Grouper 2.1.5

Support de TP

Installation des principaux modules de Grouper

Installation de Grouper

I Introduction

1 Machine Virtuelle

Ce TP s'appuie sur la machine virtuelle réalisée par Vincent Bonamy à l'occasion du Workshop sur l'installation d'ESUP V4 en mai 2013 ; merci à lui ! Elle dispose des services suivants :

- LDAP : 100 étudiants, enseignants et personnels.
  Exemples de comptes associés : etud0/esup, ens20/esup, pers99/esup.
  Compte administrateur : cn=admin,dc=esup-portail,dc=org / esup
- CAS : installé dans ***/opt/cas/tomcat-cas/,*** démarrage : service tomcat-cas start.
- ESUP-portail V4 : installé dans ***/opt/tomcat-esup,*** démarrage : service tomcat-esup start.
- Postgresql : compte administrateur : postgres /esup, compte pour ESUP-Portail et Grouper : esup4/esup. Accès direct pour l'utilisateur esup sur localhost (cf. ~/.pgpass).
- Apache : en frontal des différents tomcats : configuration dans ***/etc/apache2/sites-enabled/*** (e.g. cas.univ.fr).
- Pour chaque service un nom de domaine est défini dans ***/etc/hosts*** : ldap.univ.fr, cas.univ.fr etc.

Les utilitaires Apache-ant 1.8 , Apache-maven 3, ainsi qu'un JDK 1.6 sont également installés dans /usr/local.

2 Répertoires, fichiers de configuration et scripts

Les packages de base utilisés pour ce TP se trouvent dans ***/opt/grouper/packages.*** Toutes les installations sont réalisées dans ***/opt/grouper.*** Par exemple, le tomcat utilisé pour Grouper est dans ***/opt/grouper/tomcat-grouper.***

L'ensemble des étapes de ce TP sont scriptées pour pouvoir être rejouées facilement. Ces scripts se trouvent dans ***/home/esup/TP_Grouper/scripts.*** Il sont organisés par chapitre et les fichiers de configuration associés sont regroupés, par chapitre également, dans ***/home/esup/TP_Grouper/conf.***

L'objectif de cette vm est également de faciliter l'auto-formation et les expérimentations autour de Grouper. Pour cela, les scripts suivants permettent d'effectuer des réinitialisations ou d'automatiser les scripts à rejouer :

- reset_LDAP.sh : vide la branche groups et supprime les informations d'appartenance du côté de la branche people.
- reset_grouper-db.sh : réinitialise la base Grouper.
- reset_esup-db.sh : réinitialise la base ESUP-Portail.
- reset_vm.sh : réinitialise la machine virtuelle en supprimant toutes les applications web, les virtualhosts, les packages configurés et en réinitialisant l'annuaire et les bases de données.
- install-all.sh : utile pour chaîner les scripts et se ramener à un état spécifique de la machine virtuelle.

3 Conventions typographiques

Dans les pages de ce support, les conventions typographiques suivantes sont utilisées :

Script d'une étape du TP :

~/TP_Grouper/scripts/Chap4_grouperAPI.sh

Opération à réaliser dans une console :

```
tar -zxvf /opt/grouper/packages/grouper.api-2.1.5.tar.gz
```

Modifications apportées à un fichier de configuration :

```
hibernate.connection.url                = jdbc:postgresql://localhost:5432/grouper_2_1_5
hibernate.connection.username           = esup4
```

Fichier ou répertoire :

grouper.hibernate.properties

II Composants Grouper

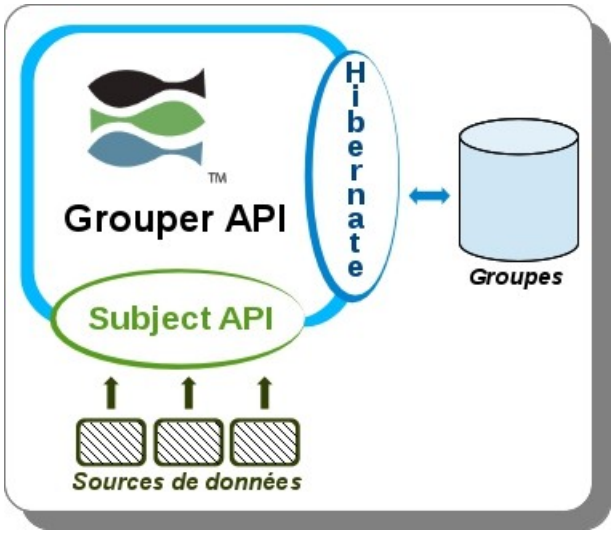
1 Vue d'ensemble

La partie centrale de grouper est constituée d'une API qui permet de manipuler les groupes stockés en base. Sur ce module viennent se greffer un certain nombre de composants : interfaces graphiques, web services, etc. Le schéma suivant représente les principaux modules de Grouper auxquels vient s'ajouter l'interface graphique ESCOGrouper, qui est une contribution du projet ESCOPortail.

Ressources :
https://spaces.internet2.edu/display/Grouper/Grouper+Wiki+Home
https://spaces.internet2.edu/display/Grouper/Administration+Guides
https://spaces.internet2.edu/display/groupertrain/Grouper+Training

Les différents composants Grouper autour de Grouper-API.

2 Détail des composants

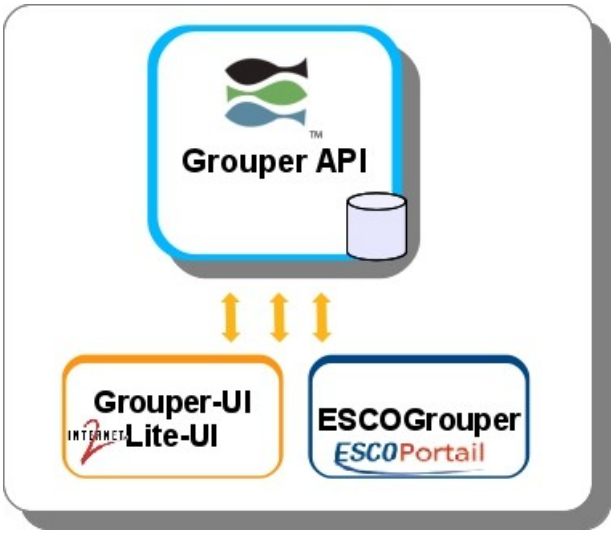


Grouper-API

Composant de base de Grouper, Grouper-API est une application java qui est intégrée dans d'autres applications.

C'est dans ce module que réside toute la logique métier relative aux groupes, dossiers, attributs, permissions, etc.

Cette API s'appuie sur l'API subject pour accéder aux sources de données contenant les éléments constitutifs des groupes.



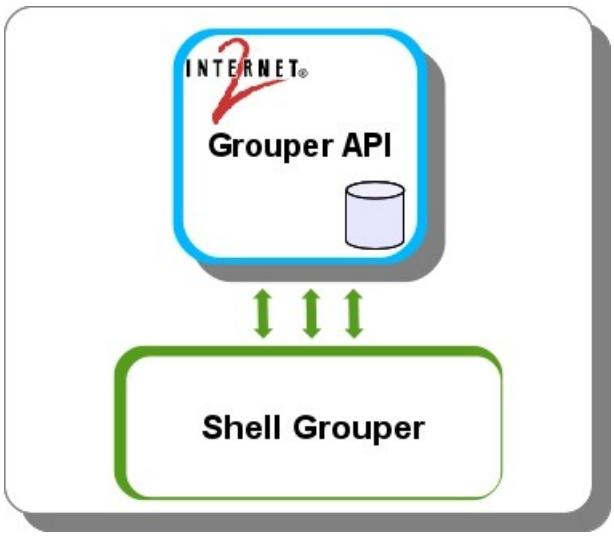
Interfaces utilisateurs

Les interfaces utilisateurs de Grouper fournies par Internet 2 sont :

- Grouper-UI : interface historique d'aministration.
- Lite-UI : interface plus récente.

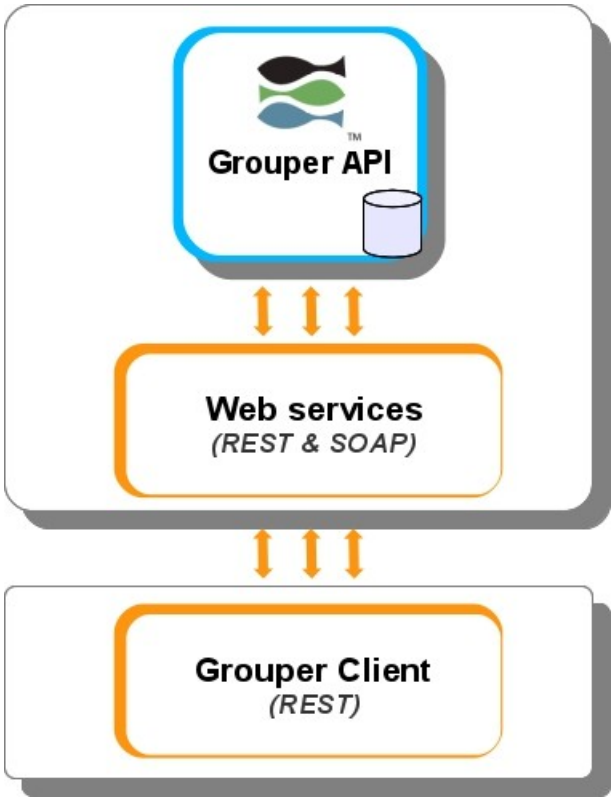
Certaines fonctionnalités sont présentes dans les deux interfaces mais certaines sont spécifiques à l'une ou l'autre. Ces deux interfaces vont être refondues dans la version 2.2.

ESCOGrouper est une contribution du RECIA qui permet de fournir une interface plus simple et conviviale aux utilisateurs.



Shell Grouper

Le shell Grouper est intégré au package de Grouper-API. Il s'agit d'un utilitaire en ligne de commande qui permet de manipuler les groupes, dossiers, attributs, privilèges, etc.
Il fonctionne en mode interactif ou batch.
Il permet également d'invoquer d'autres composants de Grouper comme le loader ou PSP.

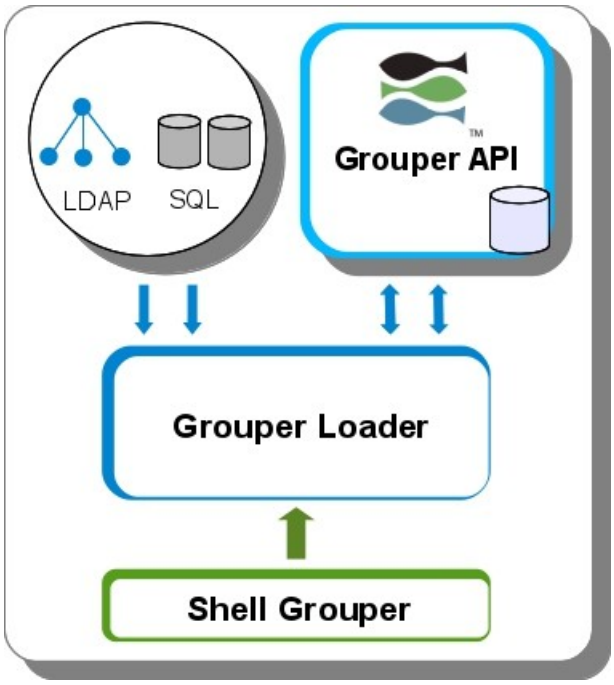


Web service

Le web service est livré sous forme d'un package spécifique et consiste en une application web qui embarque Grouper-API.
Le web service est de type REST ou SOAP.

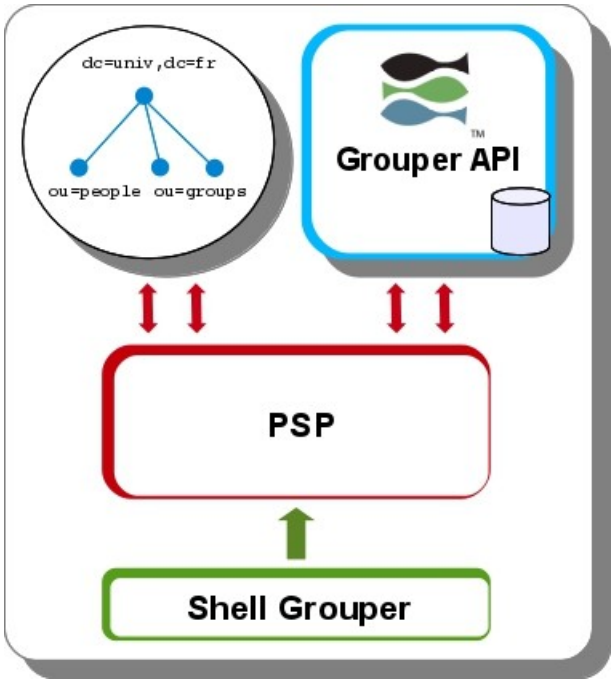
Client Grouper

Le client Grouper est également livré sous forme d'un package indépendant et peut être utilisé en ligne de commande ou intégré comme bibliothèque aux applications pour interagir avec le web service.



Grouper Loader

Composant intégré au package de Grouper-API, il permet de constituer et maintenir des groupes ou des listes de groupes à partir de sources SQL ou LDAP.
Il peut fonctionner comme un démon et être utilisé pour effectuer d'autres actions comme la publication de groupes en temps réel.
Il tend à se différencier en deux composants : le démon Grouper et le loader qui en serait un élément constitutif.



Provisioning Service Provider

Livré sous la forme d'un package spécifique, ce composant s'intègre à Grouper-API par copie de librairies et de fichiers de configuration.
Ce composant permet de publier la structure de groupes dans une branche groups d'un annuaire et de maintenir les appartenances aux groupes dans la branche people.

La publication peut être réalisée de façon complète ou incrémentale.

III Base de données Postgresql

Script : *~/TP_Grouper/scripts/Chap3_postgres.sh*

Création de la base Grouper

```
psql -U postgres -h localhost -f ~/TP_Grouper/conf/Chap3-pgsql/grouper-pgsql.sql
```

Contrôle :

```
psql -U postgres -h localhost -c "\l"
                                  Liste des bases de données
      Nom      | Propriétaire | Encodage |...   | Type caract. | ...
---------------+--------------+----------+------+--------------+---
 grouper_2_1_5 | esup4        | UTF8     |...   | fr_FR.UTF-8
```

Remarque : le fichier ***/home/esup/. pgpass*** permet a l'utilisateur esup de se connecter aux bases de données en localhost, avec les logins esup4 ou postgres, sans avoir à s'authentifier :

```
cat /home/esup/.pgpass
localhost:5432:*:postgres:esup
localhost:5432:grouper_2_1_5:esup4:esup
```

IV Installation de Grouper-API

Script : *~/TP_Grouper/scripts/Chap4_grouperAPI.sh*

Cette partie décrit la configuration et la compilation de l'API Grouper. Le jar obtenu ainsi que les fichiers de configuration seront ensuite utilisés par les autres modules, tels que les interfaces utilisateurs ou le web service.

Ressources **:**
https://spaces.internet2.edu/display/Grouper/Subject+API
https://spaces.internet2.edu/pages/viewpage.action?pageId=14517958
https://spaces.internet2.edu/display/Grouper/Member+search+and+sort+columns

Les fichiers de configuration de l'API Grouper se trouvent dans le sous-répertoire ***grouper.api/conf.***

Ils concernent le paramétrage des éléments suivants :

- l'accès à la base de données,
- l'accès aux sources de données extérieures à Grouper (e. g. la branche people d'un LDAP),
- les logs,
- Grouper.

Dans ce répertoire se trouvent également de nombreux fichiers d'exemples, très documentés.

1 Accès à la base de données Grouper : grouper.hibernate.properties

Grouper utilise Hibernate pour accéder à sa base de données. Le fichier de configuration, ***grouper.hibernate.properties*****,** est un fichier hibernate classique qui permet de paramétrer un certain nombre d'éléments : accès à la base, pool de connexions, cache, etc. Ici, seul le paramétrage de l'accès à la base de données est modifié par rapport au fichier d'origine :

```
hibernate.connection.url                = jdbc:postgresql://localhost:5432/grouper_2_1_5
hibernate.connection.username           = esup4
hibernate.connection.password           = /opt/grouper/grouper.api/conf/dbkey.txt
```

```
cd /opt/grouper
tar -zxvf /opt/grouper/packages/grouper.api-2.1.5.tar.gz
ln -s grouper.api-2.1.5 grouper.api

cd grouper.api/conf

cp ~/TP_Grouper/conf/Chap4-grouperAPI/grouper.hibernate.properties ./
cp ~/TP_Grouper/conf/Chap4-grouperAPI/morphString.properties ./
cp ~/TP_Grouper/conf/Chap4-grouperAPI/dbkey.txt ./
```

Remarques :
Le mot de passe d'accès à la base de données n'est pas en clair dans fichier hibernate mais chiffré dans un fichier externe. Pour générer ce fichier il faut saisir une clé aléatoire dans le fichier ***morphString.properties*** :

```
encrypt.key = esup225GrouPeR473
```

Puis chiffrer le mot de passe :

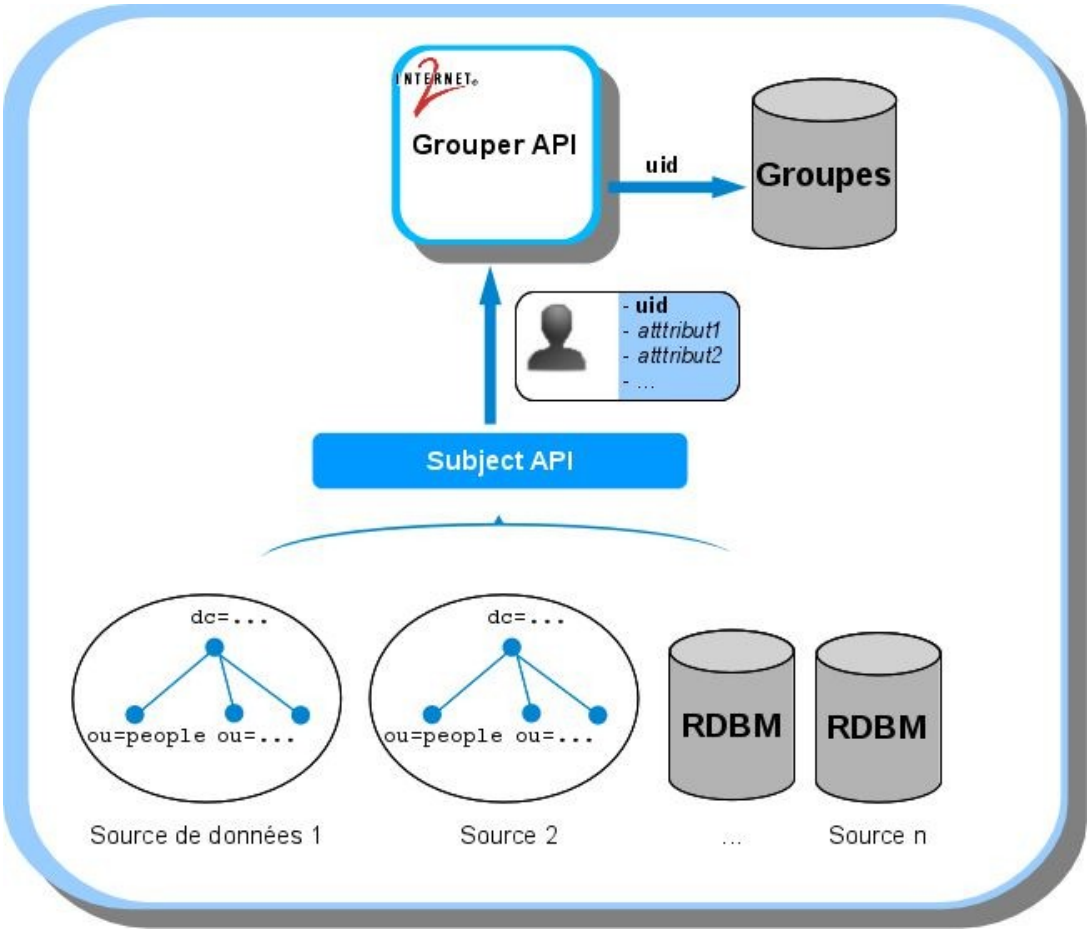
```
cd /opt/grouper/grouper.api && java -jar lib/grouper/morphString.jar dontMask
Enter the location of morphString.properties: conf/morphString.properties
Type the string to encrypt (note: pasting might echo it back): esup
The encrypted string is: q/v8lgwwcVh/UwV1fmZy+A==
```

Et enfin copier la chaîne chiffrée dans un fichier texte, ici ***dbkey.txt.***

Pour que cette procédure fonctionne, il peut être nécessaire d'installer au niveau de la JRE l'extension Java Cryptography Extension (JCE) :
http://www.oracle.com/technetwork/java/javase/downloads/jce-6-download-429243.html

2 Sources de données externes : configuration de l'API Subject

L'accès aux données constituant les groupes, en général des personnes, est réalisé via une API d'abstraction : l'API Subject. Cette API permet d'utiliser conjointement plusieurs sources de données : bases de données, annuaires, etc. Les éléments manipulés, les « sujets » , sont constitués d'un identifiant et d'une liste d'attributs. Grouper ne stocke en base que l'identifiant des sujets ainsi que l'identifiant de la source de données.



API Subject – Accès aux données utilisées pour constituer les groupes.

Le fichier de configuration associé est ***conf/sources.xml*** et pour chaque source la configuration consiste principalement à :

- Paramétrer les accès.
- Désigner l'attribut utilisé comme identifiant, l'unicité de cet attribut est requise sur l'ensemble des sources de données.
- Définir les requêtes ou filtres utilisés pour réaliser les recherches.
- Désigner les attributs remontés pour l'affichage des fiches des individus.

L'exemple suivant décrit les grandes parties de ce fichier de configuration pour la déclaration d'une source de données basée sur LDAP.

La déclaration d'une source de données commence par la classe d'implémentation de l'adaptateur, l'identifiant et la description de la source, ainsi que le type de données remontées, ici des personnes :

```
<source adapterClass="edu.internet2.middleware.grouper.subj.GrouperJndiSourceAdapter">
        <id>l**dap**</id>
        <name>Annuaire</name>
        <type>person</type>
```

Remarque : l'identifiant de la source de données, ici « ldap », est utilisé dans d'autres fichiers de configuration (e.g. PSP, ESCOGrouper).

Configuration de l'accès LDAP :

```
 <init-param>
        <param-name>INITIAL_CONTEXT_FACTORY</param-name>
        <param-value>com.sun.jndi.ldap.LdapCtxFactory</param-value>
</init-param>

<init-param>
        <param-name>PROVIDER_URL</param-name>
        <param-value>ldap://ldap.univ.fr:389</param-value>
</init-param>

<init-param>
        <param-name>SECURITY_AUTHENTICATION</param-name>
        <param-value>simple</param-value>
</init-param>

<init-param>
        <param-name>SECURITY_PRINCIPAL</param-name>
        <param-value>cn=admin,dc=esup-portail,dc=org</param-value>
</init-param>

<init-param>
        <param-name>SECURITY_CREDENTIALS</param-name>
        <param-value>/opt/grouper/grouper.api/conf/ldapkey.txt</param-value>
</init-param>
```

Les paramètres suivants permettent de spécifier les attributs correspondant respectivement à l'identifiant (stocké dans la base Grouper), le nom et la description des éléments remontés de la source de données :

```
<init-param>
	<param-name>SubjectID_AttributeType</param-name>
	<param-value>uid</param-value>
</init-param>

<init-param>
	<param-name>SubjectID_formatToLowerCase</param-name>
	<param-value>false</param-value>
</init-param>

<init-param>
	<param-name>Name_AttributeType</param-name>
	<param-value>cn</param-value>
</init-param>

<init-param>
	<param-name>Description_AttributeType</param-name>
	<param-value>description</param-value>
</init-param>
```

Trois recherches LDAP sont ensuite définies. Elles sont utilisées notamment pour rechercher les données associées aux membres d'un groupe ou encore pour rechercher des personnes en vue de les ajouter à un groupe.

```
<search>
	<searchType>searchSubject</searchType>
	<param>
			<param-name>filter</param-name>
			<param-value>(&amp;amp; (uid=%TERM%) (objectclass=eduPerson))</param-value>
	</param>

	<param>
			<param-name>scope</param-name>
			<param-value>SUBTREE_SCOPE</param-value>
	</param>

	<param>
			<param-name>base</param-name>
			<param-value>ou=people,dc=esup-portail,dc=org</param-value>
	</param>
</search>
```

```
<search>
        <searchType>searchSubjectByIdentifier</searchType>

        <param>
                        <param-name>filter</param-name>
                        <param-value>(&amp;amp; (uid=%TERM%) (objectclass=eduPerson))</param-value>
        </param>

        <param>
                        <param-name>scope</param-name>
                        <param-value>SUBTREE_SCOPE</param-value>
        </param>

      <param>
                        <param-name>base</param-name>
                        <param-value> ou=people,dc=esup-portail,dc=org </param-value>
        </param>
</search>
```

```
<search>
        <searchType>search</searchType>

        <param>
                        <param-name>filter</param-name>
                        <param-value>(&amp;amp; (uid=%TERM%) (objectclass=eduPerson)) </param-value>
        </param>

        <param>
                        <param-name>scope</param-name>
                        <param-value>SUBTREE_SCOPE</param-value>
        </param>

        <param>
                        <param-name>base</param-name>
                        <param-value>ou=people,dc=esup-portail,dc=org</param-value>
        </param>
</search>
```

Il est possible de définir des attributs virtuels calculés à partir d'autres attributs. L'évaluation de ces attributs est basée sur JEL (Java Expression Library) : http://www.gnu.org/software/jel/.

Le nom de ces attributs est composé du préfixe subjectVirtualAttribute, d'un index puis du nom de l'attribut. L'index permet d'ordonner les attributs virtuels dans le cas de dépendances.

```
<init-param>

    <param-name>subjectVirtualAttribute_0_esup_virt_attr</param-name>
        <param-value>
            $subjectUtils.defaultIfBlank(subject.getAttributeValueOrCommaSeparated('cn'), '')} \
(${subject.getAttributeValueOrCommaSeparated('supannAffectation')})
        </param-value>

</init-param>
```

Il est ensuite possible de définir les attributs utilisés pour trier les membres des groupes et effectuer des recherches dans les groupes :

```
<init-param>
        <param-name>sortAttribute0</param-name>
        <param-value>supannAffectation</param-value>
</init-param>

<init-param>
        <param-name>sortAttribute1</param-name>
        <param-value>cn</param-value>
</init-param>

<init-param>
        <param-name>searchAttribute0</param-name>
        <param-value>supannAffectation</param-value>
</init-param>

<init-param>
        <param-name>searchAttribute1</param-name>
        <param-value>cn</param-value>
</init-param>
```

Il est enfin possible de définir les attributs remontés dans les fiches des personnes :

```
<attribute>cn</attribute>
<attribute>sn</attribute>
<attribute>uid</attribute>
<attribute>department</attribute>
<attribute>mail</attribute>
<attribute>eduPersonAffiliation</attribute>
<attribute>supannAffectation</attribute>
```

```
cp ~/TP_Grouper/conf/Chap4-grouperAPI/sources.xml ./
cp ~/TP_Grouper/conf/Chap4-grouperAPI/ldapkey.txt ./
```

3 Configuration de Grouper

Les autres éléments de configuration de l'API sont définis dans le fichier ***conf/grouper.properties.*** Il permet de définir un certain nombre d'éléments tels que l'utilisation d'un groupe de confiance (administrateurs), les droits positionnés par défaut pour la création des groupes et dossiers, l'initialisation du registre, le paramétrage des hooks, etc.

Dans cet exemple, les modifications concernent :

1. Le nom de l'instance Grouper :

```
grouper.env.name = Formation ESUP
```

2. Le dossier de stockage des attributs, l'utilisation d'un groupe de confiance et leur création automatique lors de l'initialisation de Grouper :

```
configuration.autocreate.system.groups = true

groups.wheel.use                       = true
groups.wheel.group                     = esup:admin:grouper:wheel

grouper.attribute.rootStem             = esup:admin:grouper:attributes
```

3. Le paramétrage des droits par défaut pour les groupes, dossiers et attributs (suppression des droits read et view) :

```
groups.create.grant.all.admin              = false
groups.create.grant.all.optin              = false
groups.create.grant.all.optout             = false
groups.create.grant.all.read               = false
groups.create.grant.all.update             = false
groups.create.grant.all.view               = false

stems.create.grant.all.create              = false
stems.create.grant.all.stem                = false

attributeDefs.create.grant.all.attrAdmin   = false
attributeDefs.create.grant.all.attrOptin   = false
attributeDefs.create.grant.all.attrOptout  = false
attributeDefs.create.grant.all.attrRead    = false
attributeDefs.create.grant.all.attrUpdate  = false
attributeDefs.create.grant.all.attrView    = false
entities.create.grant.all.view             = false
```

```
cp ~/TP_Grouper/conf/Chap4-grouperAPI/grouper.properties ./
```

4 Configuration des logs : log4j.properties

Les logs sont configurés pour utiliser le répertoire ***/home/esup/src/grouper/grouper.api/logs/.***
Les principaux fichiers de logs sont **grouper_error.log** et **grouper_event.log.**

```
cp ~/TP_Grouper/conf/Chap4-grouperAPI/log4j.properties ./
```

Construction et initialisation de Grouper-API

```
cd /opt/grouper/grouper.api
ant dist
bin/gsh.sh -registry -check -runscript -noprompt
```

Contrôle

```
psql -U postgres -h localhost grouper_2_1_5 -c "\d"

                    Liste des relations
 Schéma |              Nom                | Type  | Propriétaire
--------+---------------------------------+-------+--------------
 public | grouper_attr_asn_asn_attrdef_v | vue   | esup4
 public | grouper_attr_asn_asn_efmship_v | vue   | esup4
 public | grouper_attr_asn_asn_group_v   | vue   | esup4
 public | grouper_attr_asn_asn_member_v  | vue   | esup4
 public | grouper_attr_asn_asn_mship_v   | vue   | esup4
 public | grouper_attr_asn_asn_stem_v    | vue   | esup4
 public | grouper_attr_asn_attrdef_v     | vue   | esup4
 public | grouper_attr_asn_efmship_v     | vue   | esup4
 public | grouper_attr_asn_group_v       | vue   | esup4
 public | grouper_attr_asn_member_v      | vue   | esup4
 public | grouper_attr_asn_mship_v       | vue   | esup4
 public | grouper_attr_asn_stem_v        | vue   | esup4
 public | grouper_attr_assign_action     | table | esup4
 public | grouper_attr_assign_action_set | table | esup4
 public | grouper_attr_assn_action_set_v | vue   | esup4
 public | grouper_attr_def_name_set_v    | vue   | esup4
 public | grouper_attr_def_priv_v        | vue   | esup4
 public | grouper_attribute_assign       | table | esup4
 public | grouper_attribute_assign_value | table | esup4
 (…)
 public | grouper_rules_v                | vue   | esup4
 public | grouper_stems                  | table | esup4
 public | grouper_stems_v                | vue   | esup4
 public | grouper_types                  | table | esup4
 public | subject                        | table | esup4
 public | subjectattribute               | table | esup4
(106 lignes)
```

V Interface graphique : Grouper-UI

Script : *~/TP_Grouper/scripts/Chap5_grouperUI.sh*

1 Paramétrage et build

Grouper est fourni avec deux interfaces utilisateurs Grouper-UI et Lite-UI. Ces deux interfaces sont regroupées dans un même package.

Le fichier de configuration est ***build.properties.*** Les deux propriétés modifiées concernent l'emplacement de grouper-API et le rôle Tomcat dans le cadre de l'authentification par défaut :

```
grouper.folder = ../grouper.api

grouper.role = grouper_user
```

```
cd /opt/grouper
tar -zxvf /opt/grouper/packages/grouper.ui-2.1.5.tar.gz
ln -s grouper.ui-2.1.5 grouper.ui

cd grouper.ui

cp ~/TP_Grouper/conf/Chap5-grouperUI/build.properties ./
cp ~/TP_Grouper/conf/Chap5-grouperUI/include.jsp ./webapp/WEB-INF/jsp/
cp ~/TP_Grouper/conf/Chap5-grouperUI/nav.properties ./conf/resources/grouper/

ant war
```

2 Déploiement

Configuration de Tomcat : ajout du rôle grouper_user et de l'administrateur GrouperSystem utilisé :

```
<tomcat-users>

        (...)

        <role rolename="grouper_user"/>

        <user username="GrouperSystem" password="esup" roles="grouper_user"/>
        <user username="ens0" password="esup" roles="grouper_user"/>
        <user username="ens1" password="esup" roles="grouper_user"/>
        <user username="ens2" password="esup" roles="grouper_user"/>
        <user username="ens3" password="esup" roles="grouper_user"/>
        <user username="etud1" password="esup" roles="grouper_user"/>
        <user username="pers1" password="esup" roles="grouper_user"/>

</tomcat-users>
```

```
cp ~/TP_Grouper/conf/Chap5-grouperUI/tomcat-users.xml \
 /opt/grouper/tomcat-grouper/conf/
```

Configuration d'Apache : configuration d'un virtual host pour grouper.univ.fr avec mod_proxy.

```
su -c "cp /home/esup/TP_Grouper/conf/Chap5-grouperUI/grouper.univ.fr \
/etc/apache2/sites-enabled"

su -c "apachectl restart"
```

Copie du war, installation du service pour le Tomcat de Grouper, et démarrage du service :

```
cp /opt/grouper/grouper.ui/dist/grouper.war \
 /opt/grouper/tomcat-grouper/webapps/

su -c "cp /home/esup/TP_Grouper/conf/Chap5-grouperUI/tomcat-grouper \
 /etc/init.d"

su -c "service tomcat-grouper start"
```

Contrôle : https://grouper.univ.fr

Notes de configuration pour l'encodage des caractères :

Le paramétrage suivant devrait permettre de résoudre les problèmes d'encodage :

Fichier ***server.xml :***

```
<Connector port="8080" protocol="HTTP/1.1" connectionTimeout="20000"
        redirectPort="8443"
        URIEncoding="UTF-8" />
```

Fichier ***/opt/esup-env/env.sh :***

```
export JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8"
```

Fichier ***/opt/grouper/tomcat-grouper/webapps/grouper/WEB-INF/jsp/include.jsp :***

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
```

VI Manipulations de base

1 Groupe de confiance

Script : *~/TP_Grouper/scripts/Chap6_base-wheel.sh*

Si la fonctionnalité est activée dans le fichier de configuration de Grouper, il est possible de définir un groupe de confiance. Les membres de ce groupe ont la possibilité, au niveau de l'interface, d'agir en tant qu'administrateurs de Grouper. Le nom du groupe est paramétré dans le fichier grouper.properties : ***esup:admin:grouper:wheel.***

Ajouter l'utilisateurs ens0 au groupe de confiance

1) Se connecter en tant que GrouperSystem/esup sur l'interface utilisateur : https://grouper.univ.fr
2) Cliquer sur Explore puis naviguer jusqu'au groupe de confiance ***esup:admin:grouper:wheel***
3) Cliquer sur **Manage members** puis sur **Add member.**
4) Rechercher l'utilisateur **ens0** et l'ajouter en tant que membre du groupe.

Se déloguer - fermer le navigateur ou supprimer les cookies et se loguer en tant que ens0/esup.

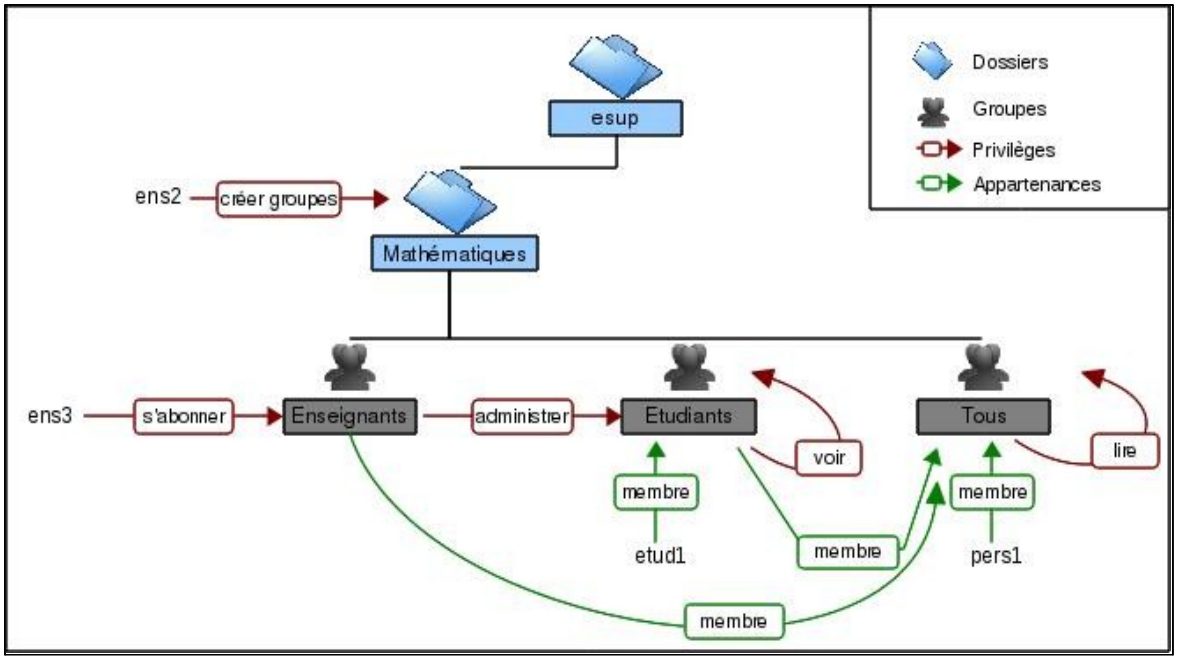
Dans le coin supérieur droit, l'utilisateur dispose maintenant de la fonctionnalité lui permettant se positionner en tant qu'administrateur :

Groupe de confiance dans Grouper-UI.

2 Création de dossiers, de groupes et attribution de privilèges

Script : *~/TP_Grouper/scripts/Chap6_base-tree.sh*

L'objectif de cette manipulation est de créer l'arborescence suivante :



Hiérarchie de groupes à créer.

1. Se connecter en tant que ens0/esup et se positionner en tant qu'administrateur (**Act as admin**).
2. Créer le dossier Mathématiques et donner à l'utilisateur ens2 le droit d'y créer des groupes :
   i. Cliquer sur **Explore** et naviguer jusqu'au dossier esup.
   ii. Cliquer sur **Create folder** .
   iii. Renseigner les champs puis cliquer sur **Save and assign privileges.**
   iv. Rechercher ens2 et lui attribuer le privilège **Create group.**
3. Se loguer en tant que ens2.
4. Créer le groupe Enseignants et donner à l'utilisateur ens3 le droit de s'abonner :
   i. Naviguer jusqu'au dossier Mathématiques puis cliquer sur **Create group.**
   ii. Renseigner les champs, puis cliquer sur **Add members.**
   iii. Rechercher l'utilisateur ens3 et lui assigner le droit **optin** (décocher membre).
5. Créer le groupe Etudiants ajouter des membres et assigner les privilèges :
   i. Revenir au dossier Mathématiques puis cliquer à nouveau sur **Create group.**
   ii. Renseigner les champs, cliquer sur **Add member,** puis ajouter comme membre l'utilisateur etud1.
   iii. Dans l'écran suivant, sélectionner la check box devant le groupe Enseignants puis cliquer sur **Assign privileges,** assigner le privilège **admin** puis cliquer à nouveau sur **Assign privileges.**
   iv. Renouveler l'opération avec le groupe Etudiants et lui assigner le privilège view.
6. Créer le groupe Tous, lui ajouter en tant que membre les groupes Enseignants et Etudiants et l'utilisateur pers1. Donner sur lui-même à ce nouveau groupe le privilège de lecture.
7. Se connecter successivement en tant que ens3, etud1 et pers1 pour voir les possibilités accordées aux utilisateurs en fonction de leurs droits et appartenances.

VII Cassification de Grouper-UI

Script : *~/TP_Grouper/scripts/Chap7_grouperUI-CAS.sh*

La cassification de Grouper-UI est fournie sous la forme d'une contribution située dans le répertoire ***contrib/yale-cas-auth/***. Pour l'activer, on utilise le mécanisme de personnalisation de l'interface Grouper.
Dans cet exemple, l'ensemble des fichiers de personnalisation sont placés dans le répertoire ***/opt/grouper/custom-grouper-ui/***. Ce répertoire contient un fichier de build additionnel qui pointe vers celui de la contribution et un fichier modifiant les règles de navigation pour prendre en compte la cassification (i.e. supprimer l'authentification Tomcat).

Le fichier de build additionnel est désigné par la propriété additionnal.build dans ***build.properties*** :

```
additional.build=${basedir}/../custom-grouper-ui/additional-build.xml
```

```
cd /opt/grouper/grouper.ui

cp ~/TP_Grouper/conf/Chap7-grouperUI-CAS/build-cas.properties \
 ./build.properties

cp -R ~/TP_Grouper/conf/Chap7-grouperUI-CAS/custom-grouper-ui /opt/grouper/
```

Le paramétrage cas doit être renseigné dans le fichier ***contrib/yale-cas-auth/build.properties*** :

```
sso.login.url          = https://cas.univ.fr/cas/login
sso.validate.url       = https://cas.univ.fr/cas/serviceValidate
grouper.server.name    = grouper.univ.fr
```

```
cp ~/TP_Grouper/conf/Chap7-grouperUI-CAS/yale-cas-auth_build.properties \
 contrib/yale-cas-auth/build.properties
cp ~/TP_Grouper/conf/Chap7-grouperUI-CAS/build-cas.xml ./build.xml
```

L'application peut ensuite être redéployée :

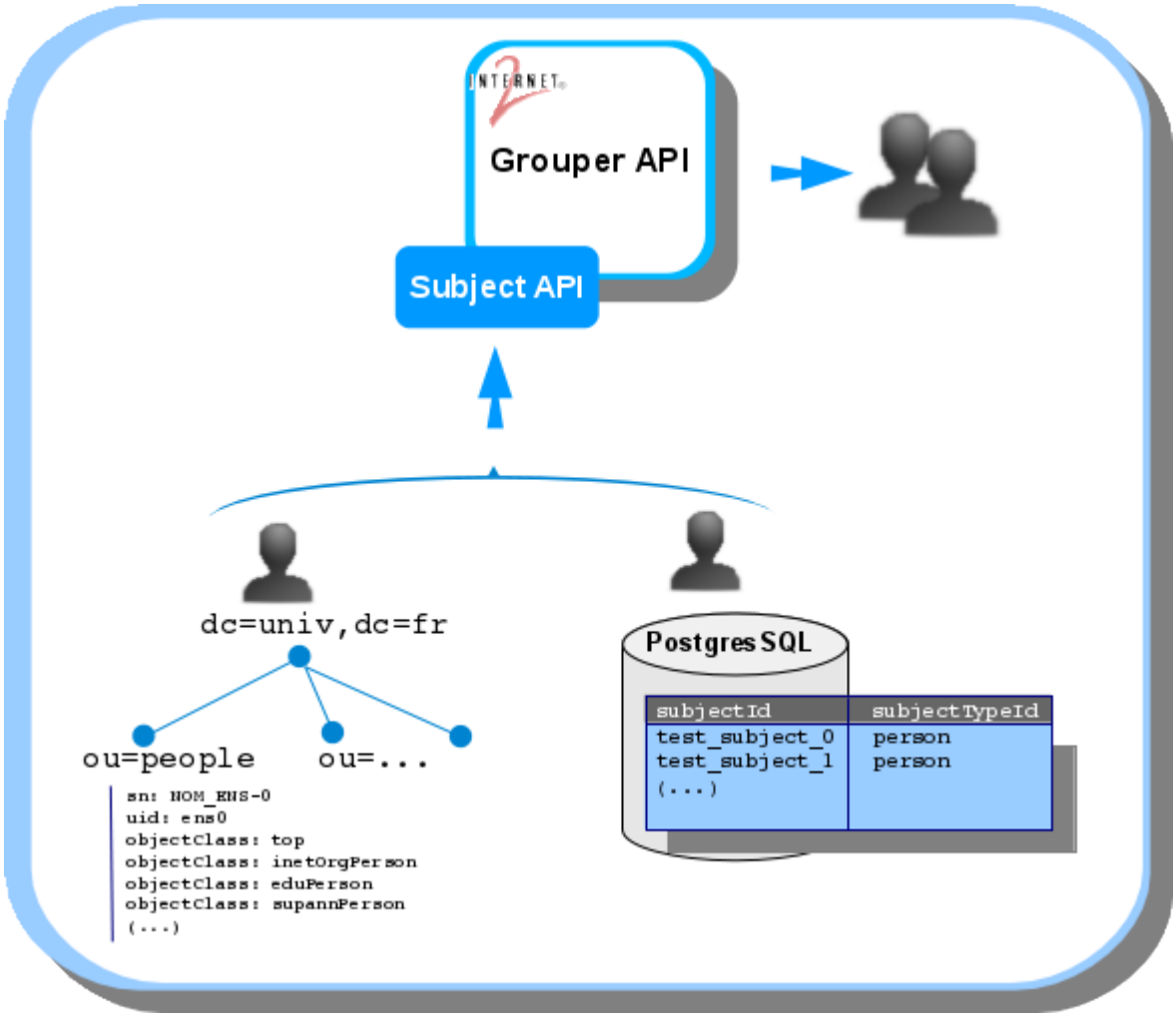
```
ant war && cp /opt/grouper/grouper.ui/dist/grouper.war \
 /opt/grouper/tomcat-grouper/webapps/

su -c "cp /home/esup/TP_Grouper/conf/Chap5-grouperUI/tomcat-grouper \
 /etc/init.d"

su -c "service tomcat-grouper start"
```

Contrôle : https://grouper.univ.fr

VIII Plusieurs sources de données : LDAP et PostgresSQL

Script : *~/TP_Grouper/scripts/Chap8_multi-sources.sh*

L'API subject, utilisée pour accéder aux données constituant les groupes, permet de configurer plusieurs sources de données, qui peuvent être de type LDAP, SQL ou encore des implémentations spécifiques. Dans l'exemple suivant, on utilise conjointement une source de type LDAP et une source de type PostgresSQL.



Intégration d'une source de données supplémentaire de type PostgresSQL.

1 Tables PostgresSQL

On utilise les tables subject et subjectattribute de la base Grouper. La création de ces deux tables de test, lors de l'initialisation de Grouper est paramétrée dans le fichier ***grouper.properties*** :

```
ddlutils.exclude.subject.tables = false
```

Ajout de quelques entrées dans ces tables :

```
psql -U postgres -h localhost grouper_2_1_5 -f
/home/esup/TP_Grouper/conf/Chap8-multi-sources/subjects.sql
```

Contrôle :

```
psql -U postgres -h localhost grouper_2_1_5 -c 'select * from subject;'

+----------------+---------------+------+
| subjectId      | subjectTypeId | name |
+----------------+---------------+------+
| test_subject_0 | person        |      |
| test_subject_1 | person        |      |
| test_subject_2 | person        |      |
| test_subject_3 | person        |      |
| test_subject_4 | person        |      |
+----------------+---------------+------+

psql -U postgres grouper_2_1_5 -h localhost -c 'select * from subjectattribute;'

+----------------+-------------+-----------------------------------+------------+
| subjectId      | name        | value                             | searchValue|
+----------------+-------------+-----------------------------------+------------+
| test_subject_0 | name        | DB User 0                         |            |
| test_subject_0 | description | Utilisateur de test 0 depuis SQL  |            |
| test_subject_0 | mail        |  test_subject_0@univ.fr           |            |
| test_subject_0 | monAttribut | Valeur A                          |            |
| test_subject_1 | name        | DB User 1                         |            |
| test_subject_1 | description | Utilisateur de test 1 depuis SQL  |            |
| test_subject_1 | mail        | test_subject_1@univ.fr            |            |
| test_subject_1 | monAttribut | Valeur A                          |            |
| (...)                                                                 |
+----------------+-------------+-----------------------------------+------------+
```

2 Ajout d'une source de données

Pour pouvoir utiliser la base Postgres, une nouvelle source de données doit être configurée dans le fichier ***sources.xml.***

Différentes implémentations d'adaptateurs sont disponibles (c.f. sources.example.xml). Ici, c'est l'adaptateur **GrouperJDBCSourceAdapter** qui est utilisé. Le fait de l'utiliser également comme fournisseur de connexion permet d'utiliser directement la configuration Hibernate de Grouper.

```
<source adapterClass="edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter">
    <id>jdbc</id>
    <name>SQL</name>
    <type>person</type>
    <init-param>
       <param-name>jdbcConnectionProvider</param-name>
       <param-value>edu.internet2.middleware.grouper.subj.GrouperJdbcConnectionProvider </param-value>
    </init-param>
```

Les requêtes pour les recherches sont spécifiées. Ces requêtes déterminent également les attributs affichés dans les fiches des utilisateurs :

```
<search>
      <searchType>searchSubject</searchType>
      <param>
         <param-name>sql</param-name>
          <param-value>
             select    s.subjectid as id,
             (select sa2.value from subjectattribute sa2 where name='name' and sa2.subjectid = s.subjectid) as name,
             (select sa3.value from subjectattribute sa3 where name='mail' and sa3.subjectid = s.subjectid) as courriel,
              (select sa4.value from subjectattribute sa4 where name='description' and sa4.subjectid = s.subjectid)
                  as description
             from subject s
             where {inclause}
          </param-value>
      </param>

      <param>
          <param-name>inclause</param-name>
          <param-value> s.subjectid = ?  </param-value>
      </param>
</search>
```

```
<search>
      <searchType>searchSubjectByIdentifier</searchType>
      <param>
          <param-name>sql</param-name>
          <param-value>
                select s.subjectid as id,
                (select sa2.value from subjectattribute sa2 where name='name' and sa2.subjectid = s.subjectid) as name,
                (select sa3.value from subjectattribute sa3 where name='mail' and sa3.subjectid = s.subjectid) as courriel,
                (select sa4.value from subjectattribute sa4 where name='description' and sa4.subjectid = s.subjectid)
                        as description
                from subject s
                where {inclause}
          </param-value>
       </param>

        <param>
           <param-name>inclause</param-name>
           <param-value>s.subjectid = ? </param-value>
        </param>
</search>
```

```
<search>
    <searchType>search</searchType>
    <param>
        <param-name>sql</param-name>
        <param-value>
            select subject.subjectid as id, namet.name as name,
                    desct.description as description, emailt.email as courriel
            from subject
            left join (select subjectid, value as name from subjectattribute
                where name='name') namet
            on subject.subjectid=namet.subjectid
            left join (select subjectid, value as description from subjectattribute
                 where name='description') desct
            on subject.subjectid=desct.subjectid
            left join (select subjectid, value as email from subjectattribute
                  where name='mail') emailt
            on subject.subjectid=emailt.subjectid
            where
                  (lower(namet.name) like '%' || ? || '%')
                  or (lower(namet.name) like '%' || ? || '%')
                  or (lower(desct.description) like '%' || ? || '%')
                  or (lower(emailt.email) like '%' || ? || '%')
        </param-value>
    </param>
</search>
```

Copie du fichier de configuration correspondant :

```
cd /opt/grouper/grouper.api/conf
cp ~/TP_Grouper/conf/Chap8-multi-sources/sources-ldap-db.xml ./sources.xml
```

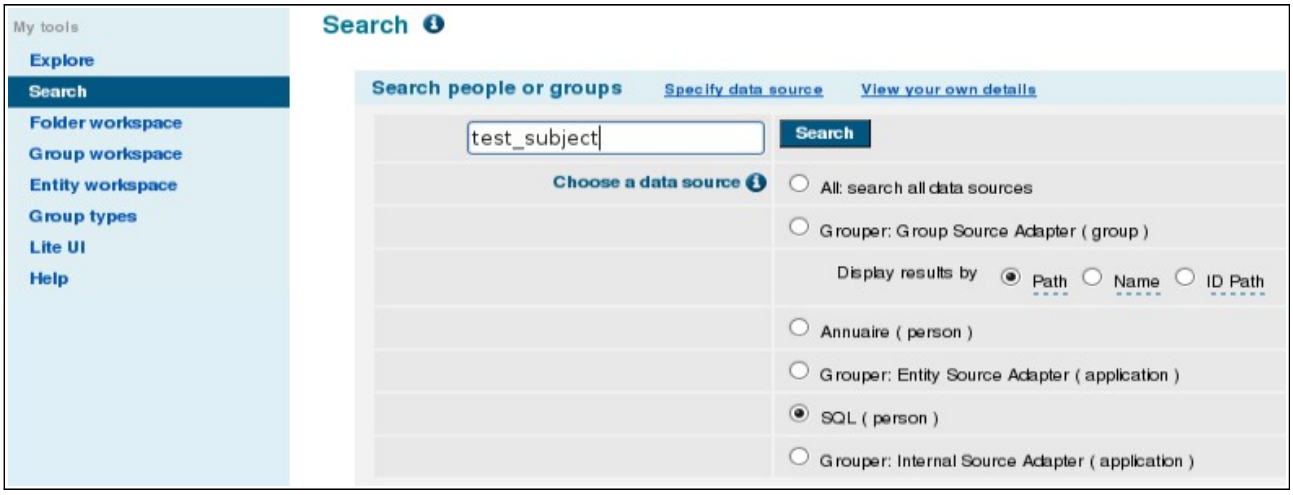
Redéploiement de Grouper-UI :

```
cd /opt/grouper/grouper.ui
ant war && cp dist/grouper.war  /opt/grouper/tomcat-grouper/webapps/
```

Contrôle :

Se connecter par exemple avec ens1/esup cliquer sur **Search** puis sur **Specify data source.**

Une nouvelle source est disponible pour la recherche de personnes.



Visualisation de la nouvelle source de données dans Grouper-UI.

IX Grouper Shell

Script : *~/TP_Grouper/scripts/Chap9_gsh.sh*

L'API de Grouper est livrée avec un shell qui permet de réaliser des opérations de maintenance, en mode interactif ou via l'écriture de scripts.
Le shell Grouper s'appuie sur l'interpréteur Java BeanShell. Outre les fonctionnalités implémentées dans le shell, la plupart des méthodes de l'API sont accessibles.

Ressources:
https://spaces.internet2.edu/display/Grouper/GrouperShell%20%28gsh%29
https://spaces.internet2.edu/display/Grouper/Useful+sample+Grouper+scripts+and+queries

1 Construction d'une structure de groupes

Dans l'exemple suivant, un script est utilisé pour créer une arborescence, positionner les privilèges et peupler les groupes.

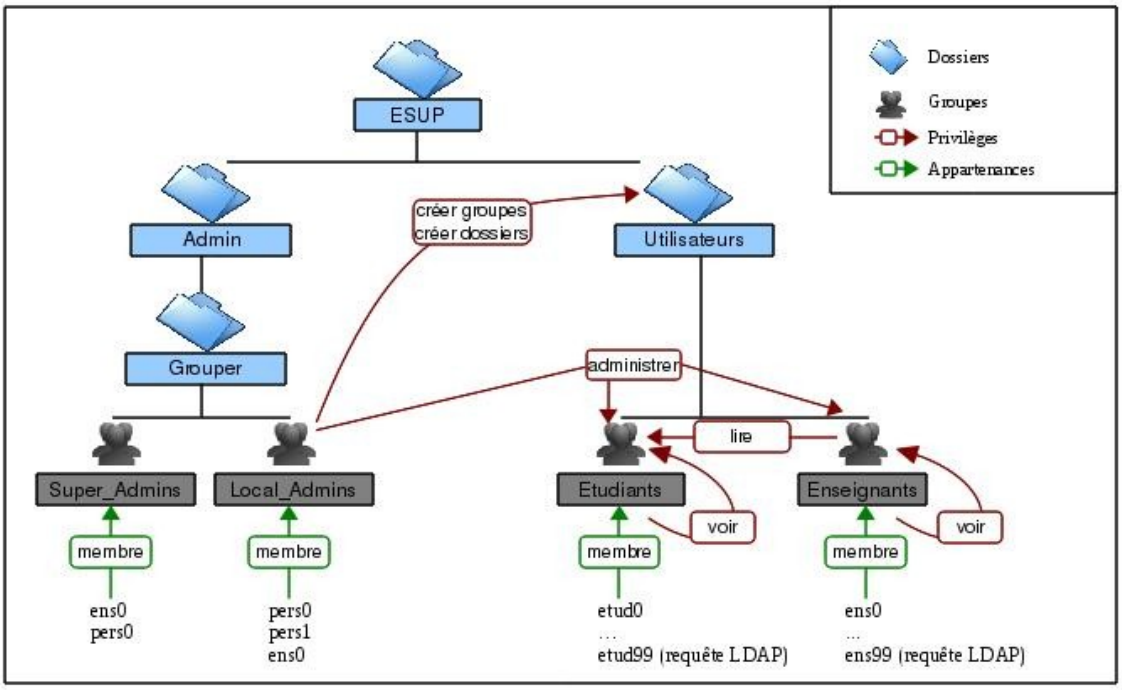
Réinitialisation de la base grouper :

```
~/TP_Grouper/scripts/reset_grouper-db.sh
```

Invocation du script de constitution de l'arborescence :

```
cd /opt/grouper/grouper.api
./bin/gsh.sh ~/TP_Grouper/conf/Chap9-gsh/grouper_tree.gsh
```

L 'arborescence créée doit correspondre au schéma suivant :



Arborescence créée via le script gsh.

Contrôle : se connecter à https://grouper.univ.fr en tant que ens0 / esup et visualiser les membres et privilèges attribués sur le groupe Etudiants.

Remarque :

Dans les scripts pour le shell grouper, les blocs d'instructions doivent être sur une seule ligne (cf https://bugs.internet2.edu/jira/browse/GRP-441). Pour contourner ce problème et améliorer la lisibilité on peut traiter les scripts gsh avec un mini script shell permettant d'échapper les retours de ligne avec \ :

```
cat /home/esup/TP_Grouper/conf/Chap9-gsh/gsh-wrapper.sh

#! /bin/bash
cat $1 | sed ':a;N;$!ba;s/\\\ *\n//g' > $1.tmp
./bin/gsh.sh $1.tmp && rm $1.tmp
```

Il suffit ensuite dans le script shell d'échapper le retour ligne avec \.

2 Import et export de la base Grouper au format xml

Export de la base :

```
cd /opt/grouper/grouper.api && bin/gsh.sh -xmlexport -noprompt /tmp/export.xml
(…)
DONE: 10:49:43: exported 2 383 records to: /tmp/export.xml
```

Réinitialisation de la base :

```
~/TP_Grouper/scripts/reset_grouper-db.sh
```

Import depuis le fichier xml :

```
bin/gsh.sh -xmlimport -noprompt /tmp/export.xml
(…)
grouper import: reading document: /tmp/export.xml, version: 2.1.5
XML file contains 2 383 records
10:58:21: Beginning import: database contains 498 records
Ending import: processed 2383 records
Ending import: database contains 2618 records
Ending import: 2120 inserts, 261 updates, and 2 skipped records
DONE: 10:58:44: imported 2 383 records from: /tmp/export.xml
Wrote record report log to:
/opt/grouper/grouper.api-
2.1.5/grouperImportRecordReport_2014_01_20__10_58_20_626.txt
```

Pour réaliser un export partiel :

```
bin/gsh.sh -xmlexport -stems esup:utilisateurs -noprompt /tmp/export.xml
```

X ESCOGrouper

1 Configuration et déploiement

Script : *~/TP_Grouper/scripts/Chap10_ESCOGrouper.sh*

Il s'agit de déployer l'interface graphique développée par le RECIA dans le cadre du projet ESCOPortail.

Remarques :
Cette interface s'appuie désormais directement sur l'API Grouper et non plus sur le web service. Les pré-requis sont maintenant les mêmes que ceux d'uPortal : JDK 6, Ant 1.8.2, Maven 3 et Tomcat 6.

Ressources :
https://sourcesup.cru.fr/docman/index.php?group_id=824
https://github.com/GIP-RECIA/esco-grouper-ui

Décompression des sources et création du lien symbolique :

```
cd /opt/grouper
unzip /opt/grouper/packages/escogrouper-grouper-2.1.5.zip
ln -s esco-grouper-ui-escogrouper-grouper-2.1.5  esco-grouper-ui
```

Copie des fichiers de configuration spécifiques :

```
cd esco-grouper-ui
cp -R ~/TP_Grouper/conf/Chap10-ESCOGrouper/custom.esup ./
cd custom.esup
ant install
```

Propriétés modifiées dans le fichier ***build.properties :***

```
escogrouper.property.file = esup.properties
maven.home=/usr/local/apache-maven/
tomcat.webapps.directory = /opt/grouper/tomcat-grouper/webapps
```

Le fichier de configuration principal est désigné par la propriété **escogrouper.property.file.** Ici il s'agit du fichier ***esup.properties :***

Type de déploiement, contexte et serveur :

```
application.type = servlet
application.context = ESCOGrouper
application.host  = escogrouper.univ.fr
```

Configuration CAS :

```
cas.mode = https
cas.host = cas.univ.fr
cas.port =
cas.uri=/cas
cas.uri.login=/login
cas.uri.logout=/logout
cas.uri.validate=/proxyValidate
cas.uri.proxyCallback=/casProxyCallback
```

Configuration LDAP :

```
ldap.host  = ldap.univ.fr:
ldap.port  = 389
ldap.dn    = cn=admin,dc=esup-portail,dc=org
ldap.dn.password = esup
ldap.basedc  = dc=fr
ldap.baseuid = ou=people,dc=esup-portail,dc=org
```

Configuration de la base Derby pour le paramétrage de l'application :

```
derby.driver  = org.apache.derby.jdbc.ClientDriver
derby.port = 1527
jderby.url = jdbc:derby://localhost:1527//opt/grouper/esco-grouper-ui/ESCODB;create=true;
derby.username = sa
derby.password = manager
derby.requireAuthentication = true
```

Les sections restantes : logs, smtp, mail des exceptions et groupes dynamiques ne sont pas utilisées dans cet exemple.

Le répertoire ***custom.esup/properties/derby/*** contient les fichiers d'initialisation de la base Derby servant au paramétrage de l'application :

- ***MyParameters.properties :*** paramètres généraux.
- ***MyStem.properties:*** paramétrage des écrans relatifs aux dossiers.
- ***MyGroup.properties :*** paramétrage des écrans relatifs aux groupes.
- ***MyPerson.properties :*** paramétrage des écrans relatifs aux personnes.
- ***MySearch.properties :*** paramétrage des écrans relatifs aux recherches.

Les modifications apportées dans ces fichiers concernent principalement deux points :

- Le nom de la source de données configurée au niveau de l'API subject (ici ldap).
  Exemple dans le fichier ***MyGroups.properties :***

```
group.property.members.map.person_ESUP.value      = ldap
```

- Les adaptations liées au schéma LDAP, qui est différent de celui utilisé par ESCOPortail.
  Exemple dans le fichier ***MyGroups.properties :***

```
group.property.members.person.cols = sn, edupersonprimaryaffiliation, uid - sn - mail
```

Dans le fichier ***converters.xml*** il faut modifier l'attribut LDAP, ici **edupersonprimaryaffiliation,** qui permet de déterminer si un individu est un enseignant, un étudiant ou un personnel :

```
<bean id="mapAttributeConverters" class="org.esco.grouperui.web.converters.MapAttributeConverters">
   <property name="attributes">
      <map>
         <entry key="#{attr['edupersonprimaryaffiliation']}" value-ref="objectClassConverter" />
         <entry key="#{attr['attribute.edupersonprimaryaffiliation']}" value-ref="objectClassConverter" />
      </map>
   </property>
</bean>
```

Dans le fichier de personnalisation i18, ***Custom_fr.properties,*** des libellés sont associés aux différentes valeurs que peut prendre cet attribut :

```
faculty = Enseignant
student = Etudiant
employee = Employ\u00E9
alum = Ancien \u00E9tudiant
researcher=Chercheur
affiliate=Invit\u00E9
oldemployee=Ancien employ\u00E9
```

Copie des fichiers de configuration Grouper et déploiement :

```
cd ..

cp  ~/TP_Grouper/conf/Chap10-ESCOGrouper/ehcache.xml ./properties/grouper
cp ~/TP_Grouper/conf/Chap4-grouperAPI/grouper.hibernate.properties
./properties/grouper
cp ~/TP_Grouper/conf/Chap4-grouperAPI/grouper.properties ./properties/grouper
cp ~/TP_Grouper/conf/Chap4-grouperAPI/log4j.properties ./properties/grouper
cp ~/TP_Grouper/conf/Chap4-grouperAPI/morphString.properties
./properties/grouper
cp ~/TP_Grouper/conf/Chap4-grouperAPI/sources.xml ./properties/grouper
cp ~/TP_Grouper/conf/Chap4-grouperAPI/grouper*.example.properties
./properties/grouper

ant init deploy
```

Configuration du virtual host Apache escogrouper.univ-tln.fr :

```
su -c "cp /home/esup/TP_Grouper/conf/Chap10-ESCOGrouper/escogrouper.univ.fr
/etc/apache2/sites-enabled/"
```

Redémarrage de Tomcat :

```
su -c "/usr/sbin/apachectl restart"
su -c "service tomcat-grouper restart"
```

Contrôle : https://escogrouper.univ.fr

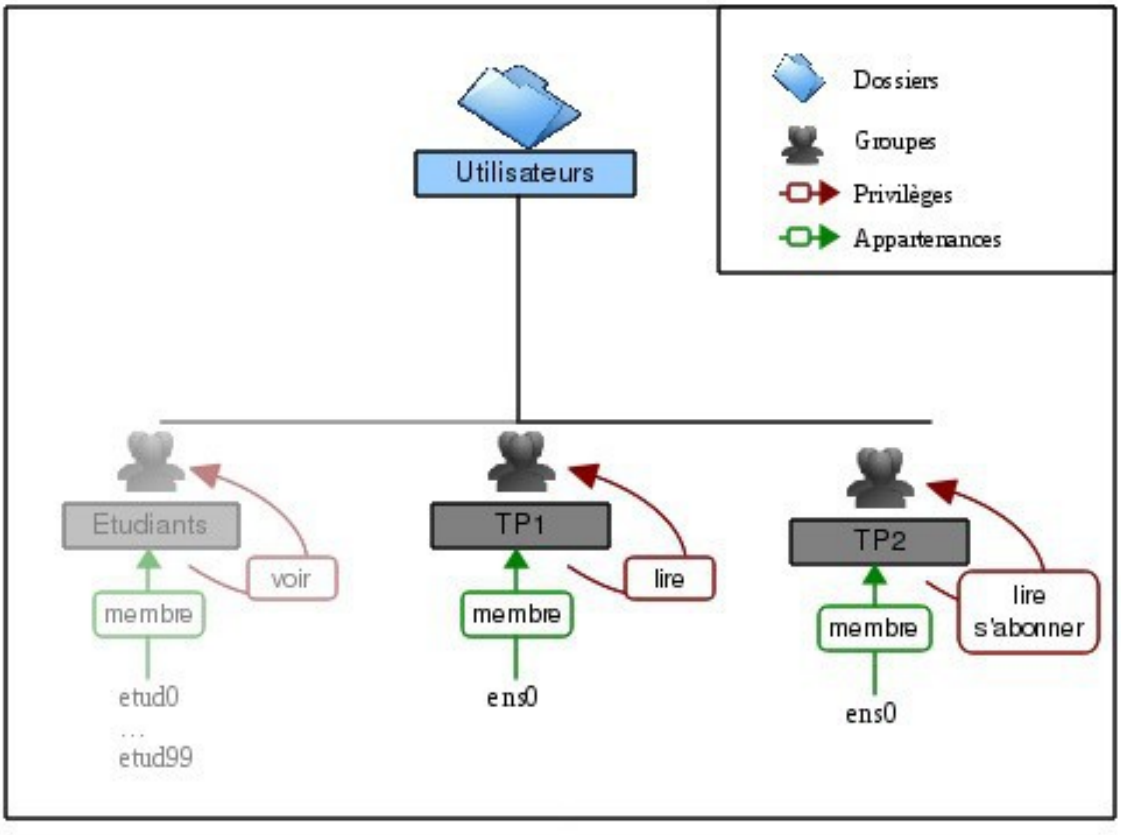
2 Notion de profile

Script : *~/TP_Grouper/scripts/Chap10_ESCOGrouper-profile.sh*

ESCOGrouper peut être paramétré en fonction du profile de l'utilisateur. Dans l'exemple suivant le profile étudiant est paramétré pour autoriser la consultation de la fiche de l'individu, des appartenances et des abonnements.

Le script gsh suivant permet d'ajouter quelques groupes à l'arborescence initiale :

```
cd /opt/grouper/grouper.api
./bin/gsh.sh ~/TP_Grouper/conf/Chap10-ESCOGrouper/escogrouper-student.gsh
```



Groupes ajoutés dans l'arborescence initiale.

Des fichiers d'exemple de configuration de profiles se trouvent dans le répertoire ***properties/profiles/.***
Celui associé au profile étudiant est modifié pour correspondre à notre exemple :

```
cp ~/TP_Grouper/conf/Chap10-ESCOGrouper/studentProfile.xml
/opt/grouper/esco-grouper-ui/properties/profiles/
```

Il s'agit de positionner un certain nombre de propriétés à true ou false :

```
<entry key="org.esco.grouperui.web.areaNavigation" value="false"/>

<entry key="org.esco.grouperui.web.person.properties" value=" true "/>

<entry key="org.esco.grouperui.web.person.properties.tab.memberships" value=" true "/>

<entry key="org.esco.grouperui.web.person.properties.tab.subscriptions" value=" true "/>

<entry key="org.esco.grouperui.web.person.properties.tab.privileges"   value="false"/>
```

L'application est redéployée :

```
cd /opt/grouper/esco-grouper-ui
ant init deploy
su -c "service tomcat-grouper restart"
```

Contrôle :
Se connecter en tant que etud0/esup :
https://escogrouper.univ.fr?profile=studentProfile

XI Provisioning Service Provider (PSP)

PSP, l'outil de publication dans un annuaire LDAP, est basé sur le protocole SPML (Service Provisioning Markup language) et utilise la technologie « Shibboleth Attribute Resolver » pour calculer les objets à publier à partir des sources de données.

Le principe général de PSP est de publier des objets sur des cibles. Un objet est constitué d'un identifiant, d'attributs et, éventuellement, de références à d'autres objets. La publication des objets sur la cible est réalisée à l'aide d'un connecteur SPML spécifique à la cible. Dans notre cas, la cible est un annuaire LDAP et on utilise donc le connecteur « SPML vers LDAP » livré avec PSP. A la condition de disposer du connecteur SPML adéquat, PSP pourrait donc être utilisé pour publier vers d'autres types de cibles.

Ressources :

https://spaces.internet2.edu/display/Grouper/Grouper+Provisioning
https://spaces.internet2.edu/display/Grouper/Grouper+Shibboleth+Integration
https://wiki.shibboleth.net/confluence/display/SHIB2/IdPAddAttribute

1 Publication initiale des groupes et des appartenances

Script : *~/TP_Grouper/scripts/Chap11_PSP.sh*

Pour déployer le module il suffit de décompresser le package et de copier les biblithèques dans le répertoire ***lib/custom/*** de Grouper API :

```
cd /opt/grouper
tar -zxvf /opt/grouper/packages/grouper.psp-2.1.5.tar.gz
ln -s grouper.psp-2.1.5 grouper.psp
cd grouper.psp
cp lib/custom/* /opt/grouper/grouper.api/lib/custom/
```

Différents exemples de configuration sont fournis avec le package dans le répertoire ***conf,*** notamment pour Active Directory ou OpenLDAP. Pour publier les groupes et appartenances, le point de départ est l'exemple concernant OpenLDAP sans l'overlay memberOf :

```
cp ~/TP_Grouper/conf/Chap11-PSP/psp-grouper-to-openldap/ldap.properties
/opt/grouper/grouper.api/conf/
cp ~/TP_Grouper/conf/Chap11-PSP/psp-grouper-to-openldap/psp-internal.xml
/opt/grouper/grouper.api/conf/
cp ~/TP_Grouper/conf/Chap11-PSP/psp-grouper-to-openldap/psp-resolver.xml
/opt/grouper/grouper.api/conf/
cp ~/TP_Grouper/conf/Chap11-PSP/psp-grouper-to-openldap/psp-services.xml
/opt/grouper/grouper.api/conf/
cp ~/TP_Grouper/conf/Chap11-PSP/psp-grouper-to-openldap/psp.xml
/opt/grouper/grouper.api/conf/
```

Dans notre exemple, les fichiers ***ldap.properties, psp-resolver.xml et psp.xml*** sont modifiés.

Fichier ldap.properties :

Le fichier ***ldap.properties*** permet de configurer les accès au LDAP ainsi que des macros qui pourront être utilisées dans les autres fichiers :

```
edu.vt.middleware.ldap.ldapUrl=ldap://ldap.univ.fr:389
edu.vt.middleware.ldap.searchScope=SUBTREE
edu.vt.middleware.ldap.bindDn=cn=admin,dc=esup-portail,dc=org
edu.vt.middleware.ldap.bindCredential=/home/esup/src/grouper/grouper.api/conf/ldapkey.txt
```

Macro utilisée dans les fichiers ***psp.xml*** et ***psp-resolver.xml :***

```
edu.vt.middleware.ldap.baseDn=dc=esup-portail,dc=org
edu.internet2.middleware.psp.groupsBaseDn=ou=groups,dc=esup-portail,dc=org
edu.internet2.middleware.psp.peopleBaseDn=ou=people,dc=esup-portail, dc=org
```

Une macro ajoutée pour cette exemple pour exclure de la publication le dossier contenant les éléments d'administration de Grouper :

```
# ESUP : The root stem of unpublished groups.
esup.grouper.admin.stem = esup:admin:grouper
```

Le type de publication, à plat : flat ou bushy : en concervant la structure arborescente des dossiers :

```
edu.internet2.middleware.psp.structure=flat
edu.internet2.middleware.psp.cnSourceAttributeID=name
```

Fichier PSP.xml

Le fichier PSP.xml permet de déterminer les éléments à publier dans l'annuaire. Dans cet exemple, les groupes et les appartenances sont publiés, les dossiers sont ignorés. La publication des groupes est spécifiée de la façon suivante :

```
<pso
    id="group"
    authoritative="true"
    allSourceIdentifiersRef="groupNames">
```

Avec :

- id : l'identifiant de la cible.
- Authoritative : attribut permettant de spécifier si la branche des groupes est gérée entièrement par le module. S'il est à true les groupes peuvent être supprimés, sinon les groupes supprimés dans Grouper restent dans l'annuaire. Les appartenances aux groupes dans la branche people sont, par contre, supprimées.
- AllSourceIdentifiersRef : le résolver qui permet d'obtenir la liste des objets à publier.

L'entrée suivante permet de calculer l'identifiant du groupe, ici le dn. Pour calculer cet identifiant le résolver groupDn, defini dans ***psp.xml***, est utilisé qui permet de prendre en compte le type de publication : flat ou bushy. La branche cible est définie via l'attribut containerId, qui utilise une macro définie dans le fichier ***ldap.properties.***

```
<identifier
        ref="groupDn"
        targetId="ldap"
        containerId="${edu.internet2.middleware.psp.groupsBaseDn}" />
```

Les entrées issues du LDAP sont discriminée via leur objectClass.

```
<identifyingAttribute
name="objectClass"
value="${edu.internet2.middleware.psp.groupObjectClass}" />
```

Permet de retrouver le dn initial pour les groupes renommés :

```
<alternateIdentifier ref="groupDnAlternate" />
```

Entrée pour calculer l'ancien dn de groupes renommés dans le cadre de la publication incrémentielle :

```
<alternateIdentifier ref="groupDnAlternateChangeLog" />
```

Les entrées suivantes définissent une liste d'attributs à publier. Les résolvers associés sont définis dans le fichier ***psp.xml*** :

```
<attribute
        name="objectClass"
        ref="groupObjectclass"
        retainAll="true" />

<attribute name="cn" />

<attribute
        name="description"
        ref="groupDescription" />

<attribute
        name="hasMember"
        ref="hasMember" />

<attribute
        name="isMemberOf"
        ref="groupIsMemberOf" />
```

Enfin deux références sont définies pour calculer l'attribut member, l'une pour les groupes, l'autre pour les personnes. L'attribut toObject fait référence à un objet défini dans ce même fichier.

```
<references name="member" emptyValue="">

        <reference ref="membersLdap"  toObject="member" />

        <reference  ref="membersGsa"   toObject="group" />

   </references>
```

Fichier psp-resolver.xml

Le fichier ***psp-resolver.xml*** permet de définir les « résolvers » qui vont être utilisés pour calculer les valeurs des attributs définis dans le fichier ***psp.xml.***

On peut regarder, par exemple, les définitions des deux références précédentes membersLdap et membersGsa.

Le resolver pour membersLdap est basé sur l'attribut member, et est dépendant du résolver GroupDataConnector. La source de données est également spécifiée : ldap, et l'attribut utilisé comme identifiant est id. Ce résolver va donc remonter les id de membres de groupes issus du LDAP.

```
<resolver:AttributeDefinition
        id="membersLdap"
        xsi:type="grouper:Member"
        sourceAttributeID="members">

        <resolver:Dependency ref="GroupDataConnector" />

        <grouper:Attribute id="id" source="ldap" />

</resolver:AttributeDefinition>
```

Le resolver pour membersGsa est également basé sur l'attribut member et dépendant de GroupDataConnector. La source de données est la source Grouper interne associée aux groupes, g:gsa et l'identifiant est le nom des groupes membres. Ce resolver retourne les noms des sous-groupes.

```
<resolver:AttributeDefinition
        id="membersGsa"
        xsi:type="grouper:Member"
        sourceAttributeID="members">

        <resolver:Dependency ref="GroupDataConnector" />

        <grouper:Attribute id="name" source="g:gsa" />

</resolver:AttributeDefinition>
```

Le resolver GroupDataConnector filtre les groupes qui ne sont pas sous le dossier ***esup:admin:grouper*** et retourne les membres et les sous-groupes.

```
<resolver:DataConnector
        id="GroupDataConnector"
        xsi:type="grouper:GroupDataConnector">

        <grouper:Filter xsi:type="grouper:MINUS">

                        <grouper:Filter
                                xsi:type="grouper:GroupInStem"
                                name="${edu.internet2.middleware.psp.baseStem}"
                                scope="SUB" />

                        <grouper:Filter xsi:type="grouper:GroupInStem"
                                name="${esup.grouper.admin.stem}" scope="SUB" />

        </grouper:Filter>

        <grouper:Attribute id="members" />

        <grouper:Attribute id="groups" />

</resolver:DataConnector>
```

PSP est invoqué via le shell Grouper :

```
cd /opt/grouper/grouper.api/ && bin/gsh.sh -psp -bulkSync
```

Contrôle : se connecter au LDAP via jxplorer

```
cd /home/esup/jxplorer/ && ./jxplorer.sh
```

Autres exemples :

Exécution en continue avec 5 minutes maximum entre deux publications :

```
bin/gsh.sh -psp -bulkSync -interval 300
```

Publication d'un groupe :

```
bin/gsh.sh -psp -sync esup:utilisateurs:etudiants
```

Calcul sans publication :

```
bin/gsh.sh -psp -diff esup:utilisateurs:etudiants
```

Publier uniquement un objet particulier :

```
bin/gsh.sh -psp -entityName member -bulkSync
```

Afficher les options disponibles :

```
bin/gsh.sh -psp
```

2 Synchronisation continue des groupes via PSP et le ChangeLog

Script : *~/TP_Grouper/scripts/Chap11_PSP-loader.sh*

Une fois la publication initiale des groupes réalisée, il est possible d'utiliser PSP pour répercuter dans l'annuaire les modifications faites dans Grouper. Ce mécanisme fait intervenir Grouper-Loader, qui est un composant permettant d'automatiser certains traitements. Pour tester cette publication en continue, il suffit de décommenter les deux propriétés suivantes dans le fichier ***grouper-loader.properties,*** pour une répercussion des modifications toutes les minutes :

```
changeLog.consumer.psp.class = edu.internet2.middleware.psp.grouper.PspChangeLogConsumer

changeLog.consumer.psp.quartzCron = 0 * * * * ?
```

D'autres propriétés dans ce fichier permettent de réaliser synchronisation complète au démarrage et/ou à intervalles réguliers.

La commande suivante permet de lancer le loader :

```
cd /opt/grouper/grouper.api/ && bin/gsh.sh -loader
```

Contrôle :

Effectuer des modifications dans Grouper, e.g. création de groupe, modification d'appartenances, puis vérifier la propagation de ces modifications dans l'annuaire.

```
cd /home/esup/jxplorer/ && ./jxplorer.sh
```

3 Installation comme service

Script : *~/TP_Grouper/scripts/Chap11_PSP-loader-deamon.sh*

Le site Internet 2 fourni des exemples de scripts qui permettent d'invoquer le loader comme service et donc de réaliser une publication dans un annuaire LDAP en continu.

Ressources :
https://spaces.internet2.edu/display/Grouper/Grouper+Book+-+Running+in+production
https://spaces.internet2.edu/display/Grouper/GrouperShell+GSH+loader+linux+service

Le script ***grouperLoader.sh*** permet d'invoquer gsh en le détachant du terminal (nohup) et d'enregistrer son pid :

```
cp ~/TP_Grouper/conf/Chap11-PSP/grouperLoader.sh  /opt/grouper/grouper.api/bin/
```

Pour installer le service le script grouper-loader est utilisé, qui invoque ***grouperLoader.sh*** en tant qu'utilisateur esup :

```
su -c "cp /home/esup/TP_Grouper/conf/Chap11-PSP/grouper-loader /etc/init.d/"
```

Il peut ensuite etre invoqué comme un script de démarrage classique :

```
esup@esup4:~$ su -c "service grouper-loader start"
Starting Grouper loader service...
Grouper loader is running...
```

```
esup@esup4:~$ su -c "service grouper-loader status"
Grouper loader is running, parent process id: 31509, process id: 31512
```

```
esup@esup4:~$ su -c "service grouper-loader stop"
Grouper loader is running with processId: 31512, waiting for exit...
Waiting for exit...
Grouper loader is stopped
```

XII Web service et Client

1 Installation du web service

Script : *~/TP_Grouper/scripts/Chap12_ws.sh*

Le web service est de type SOAP et REST et le client est de type REST et permet également d'interagir avec le LDAP.

Ressource : https://spaces.internet2.edu/display/Grouper/Grouper+Web+Services

Décompression des sources du web service :

```
cd /opt/grouper
tar -zxvf /opt/grouper/packages/grouper.ws-2.1.5.tar.gz
ln -s grouper.ws-2.1.5 grouper.ws
```

Il faut indiquer dans le fichier ***build.properties*** le répertoire de Grouper-API et indiquer le nom de l'application web :

```
grouper.dir=../../grouper.api
webapp.name=grouper-ws
```

```
cd /opt/grouper/grouper.ws/grouper-ws
cp ~/TP_Grouper/conf/Chap12-ws/build.properties ./
```

Compilation et déploiement

```
ant dist
cp build/dist/grouper-ws.war /opt/grouper/tomcat-grouper/webapps/
su -c "cp /home/esup/TP_Grouper/conf/Chap12-ws/grouper-ws.univ.fr
/etc/apache2/sites-enabled/"
su -c "apachectl restart"
su -c "service tomcat-grouper restart"
```

Contrôle :

https://grouper-ws.univ.fr/grouper-ws/status?diagnosticType=trivial
https://grouper-ws.univ.fr/grouper-ws/status?diagnosticType=db
https://grouper-ws.univ.fr/grouper-ws/status?diagnosticType=sources
https://grouper-ws.univ.fr/grouper-ws/services/GrouperService?wsdl

Exemples de requêtes http via wget :

Lister les membres d'un groupe :

```
wget 'https://GrouperSystem:esup@grouper-ws.univ.fr/grouper-
ws/servicesRest/v2_1_005/groups/esup:admin:grouper:wheel/members'
--header="Content-Type:text/x-json"
```

Recherche d'un groupe par le début de son nom :

```
wget 'https://GrouperSystem:esup@grouper-ws.univ.fr/grouper-
ws/servicesRest/v2_1_005/groups' --post-
file="/home/esup/TP_Grouper/conf/Chap12-ws/ws-approximative-name.xml"
--header="Content-Type:text/xml"
```

Trouver les groupes d'un utilisateur situés sous un dossier :

```
wget 'https://GrouperSystem:esup@grouper-ws.univ.fr/grouper-
ws/servicesRest/v2_1_005/subjects' --post-
file="/home/esup/TP_Grouper/conf/Chap12-ws/ws-groups.xml"
--header="Content-Type:text/xml"
```

2 Installation du client

Script : *~/TP_Grouper/scripts/Chap12_ws-client.sh*

Le client du web service est de type REST. Il peut être utilisé en ligne de commande ou être intégré comme librairie dans une application java.

Référence : https://spaces.internet2.edu/display/Grouper/Grouper+Client

Décompression du package (version binaire) :

```
cd /opt/grouper
tar -zxvf /opt/grouper/packages/grouper.clientBinary-2.1.5.tar.gz
ln -s grouper.clientBinary-2.1.5 grouper.client
```

Le paramétrage du client Grouper se fait dans le fichier ***grouper.client.properties.*** Il consiste principalement à renseigner l'url du web service et l'utilisateur de connexion.

```
grouperClient.webService.url = https://grouper-ws.univ.fr/grouper-ws/servicesRest/
grouperClient.webService.login =GrouperSystem
grouperClient.webService.password =esup
```

```
cd grouper.client
cp ~/TP_Grouper/conf/Chap12-ws/grouper.client.properties ./
```

Contôle :

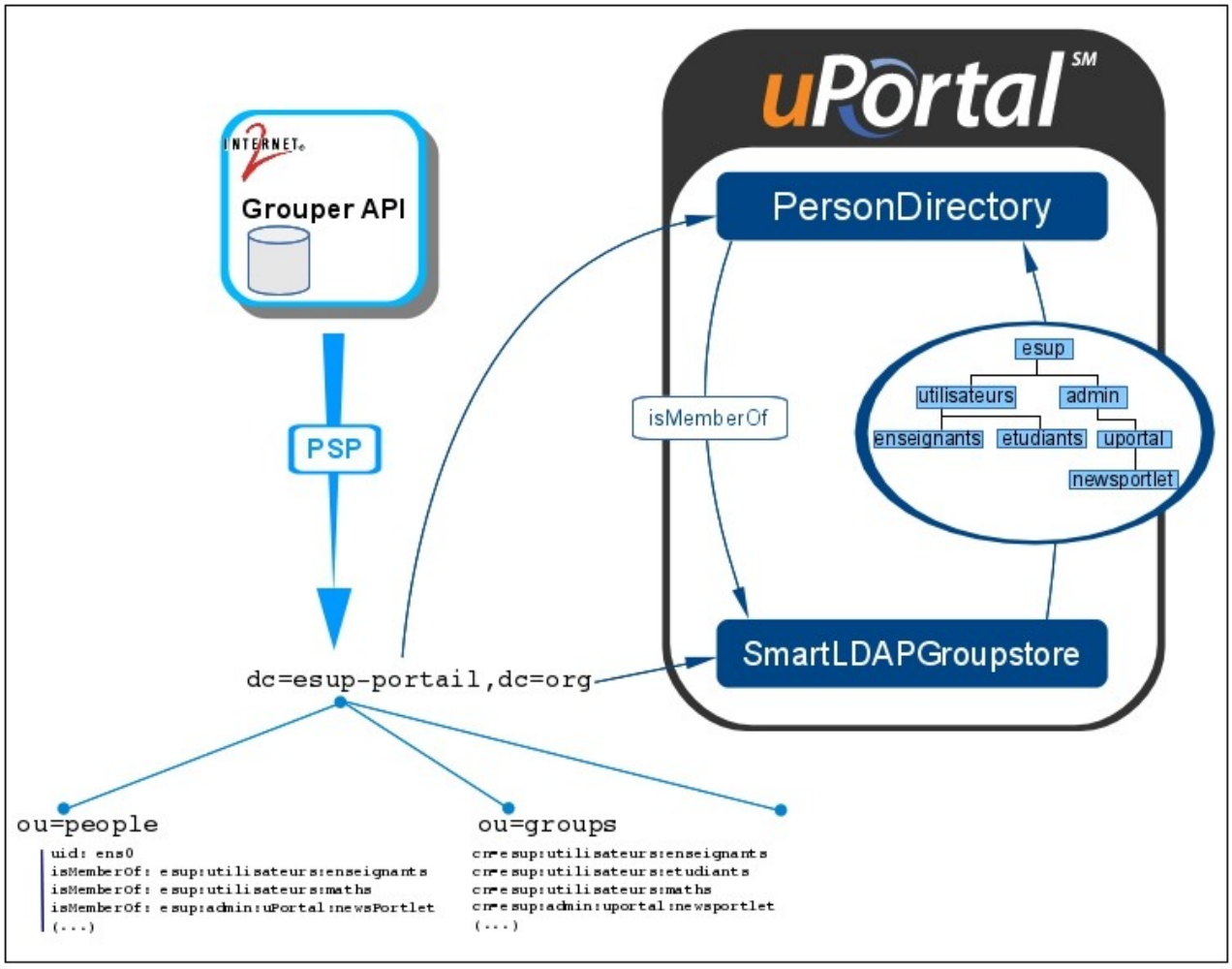
```
java  -Djavax.net.ssl.trustStore=/opt/esup-env/esup4.trustore  -jar
/opt/grouper/grouper.client/grouperClient.jar --operation=getMembersWs
--groupNames=esup:admin:grouper:wheel

GroupIndex 0: success: T: code: SUCCESS: group:
esup:admin:grouper:wheel: subjectIndex: 0: pers0
GroupIndex 0: success: T: code: SUCCESS: group:
esup:admin:grouper:wheel: subjectIndex: 1: ens0
```

XIII Intégration Grouper uPortal

Script : *~/TP_Grouper/scripts/Chap13_uPortal.sh*

L'intégration de l'arborescence de Grouper dans le portail est réalisée via le GroupStore SmartLdapGroupStore à partir des groupes publiés dans l'annuaire LDAP. Ce GroupStore utilise la branche groups de l'annuaire pour construire l'arborescence de groupes et le composant PersonDirirectory pour déterminer les appartenances des personnes à partir de leurs attributs.

Références :
http://www.esup-portail.org/pages/viewpage.action?pageId=273416195
https://wiki.jasig.org/display/UPM40/SmartLdapGroupStore
https://wiki.jasig.org/display/UPM40/Default+Person+Directory+configuration



Intégration dans uPortal via le groupstore SmartLDAPGroupstore.

1 Configuration du SmartLDAPGroupStore

La première étape consiste à activer le groupstore au niveau du composant de gestion de groupes du portail. Pour cela, il suffit de décommenter l'entrée smartldap dans le fichier ***compositeGroupService.xml :***

```
<service>
    <name>smartldap</name>
    <service_factory>org.jasig.portal.groups.ReferenceIndividualGroupServiceFactory</service_factory>
    <entity_store_factory>
        org.jasig.portal.groups.smartldap.SmartLdapEntityStore$Factory
    </entity_store_factory>
    <group_store_factory>
        org.jasig.portal.groups.smartldap.SmartLdapGroupStore$Factory
    </group_store_factory>
    <entity_searcher_factory>
        org.jasig.portal.groups.smartldap.SmartLdapEntitySearcher$Factory
    </entity_searcher_factory>
    <internally_managed>false</internally_managed>
    <caching_enabled>true</caching_enabled>
</service>
```

```
su -c "service tomcat-esup stop"
cd /opt/esup-uportal/uportal-war/src/main/resources/properties/
cp /home/esup/TP_Grouper/conf/Chap13-uPortal/compositeGroupServices.xml  ./groups/
```

La configuration du groupstore est réalisée via le fichier ***SmartLdapGroupStoreConfig.xml :***
Paramétrage de l'accès au LDAP :

```
<bean id="ldapContext" class="org.springframework.ldap.core.support.LdapContextSource">
        <property name="url" value="ldap://ldap.univ.fr:389"/>
        <property name="userDn" value="cn=admin,dc=esup-portail,dc=org"/>
        <property name="password" value="esup"/>
</bean>
```

L'arborescence de groupes est reconstruite à intervalle régulier pour répercuter les modifications de groupes publiées dans l'annuaire. Dans cet exemple, un intervalle de rafraîchissement très rapide est paramétré :

```
<bean id="groupsTreeRefreshIntervalSeconds" class="java.lang.Long">
        <constructor-arg><value>60</value></constructor-arg>
</bean>
```

Dn de la branche groupes :

```
<bean id="baseDn" class="java.lang.String">
        <constructor-arg><value>ou=groups,dc=esup-portail,dc=org</value></constructor-arg>
</bean>
```

Expression régulière pour extraire l'identifiant des groupes à partir du dn :

```
<bean id="childGroupKeyRegex" class="java.lang.String">
	<constructor-arg>
		<value>cn=(.*),ou=groups,dc=esup-portail,dc=org</value>
	</constructor-arg>
</bean>
```

Filtre LDAP pour la récupération des groupes :

```
<bean id="filter" class="java.lang.String">
	<constructor-arg><value>(objectClass=eduMember)</value></constructor-arg>
</bean>
```

Attribut utilisateur permettant de déterminer à quels groupes appartient un utilisateur :

```
<bean id="memberOfAttributeName" class="java.lang.String">
	<constructor-arg><value>isMemberOf</value></constructor-arg>
</bean>
```

Attributs de groupes pour l'identifiant, le nom et les membres des groupes :

```
<bean id="attributesMapper" class="org.jasig.portal.groups.smartldap.SimpleAttributesMapper">
	<property name="keyAttributeName">
		<value>cn</value>
	</property>
	<property name="groupNameAttributeName">
		<value>cn</value>
	</property>
	<property name="membershipAttributeName">
		<value>member</value>
	</property>
</bean>
```

```
cp /home/esup/TP_Grouper/conf/Chap13-uPortal/SmartLdapGroupStoreConfig.xml  ./groups/
```

Le portail doit être configuré pour remonter l'attribut utilisé pour déterminer les appartenances des utilisateurs, via le fichier ***personDirectoryContext.xml :***

```
<bean id="uPortalLdapAttributeSource" class="org.jasig.services.persondir.support.ldap.LdapPersonAttributeDao">
    <property name="contextSource" ref="defaultLdapContext" />
    <property name="queryAttributeMapping">
        <map>
            <entry key="username" value="${ldap.uidAttr}"/>
            <entry key="department" value="departmentNumber"/>
        </map>
    </property>

    <property name="resultAttributeMapping">
        <map>
            <entry key="eduPersonPrimaryAffiliation"> <value>eduPersonPrimaryAffiliation</value></entry>
            <entry key="eduPersonAffiliation">  <value>eduPersonAffiliation</value></entry>
            <entry key="cn"> <value>cn</value></entry>

            (…)

            <entry key="isMemberOf">  <value>isMemberOf</value></entry>
        </map>
    </property>
</bean>
```

```
cp /home/esup/TP_Grouper/conf/Chap13-uPortal/personDirectoryContext.xml ./contexts
```

Le portail est ensuite redéployé pour prendre en compte ces modifications :

```
cd /opt/esup-uportal/
ant -Dmaven.test.skip=true clean deploy-ear
su -c "service tomcat-esup start"
```

Les groupes issus du SmartLDAPGroupstore sont raccrochés sous la racine SmartLdap Root. L'instruction suivante permet de raccorder ce groupe au groupe Everyone :

```
ant data-import -Dfile=/home/esup/TP_Grouper/conf/Chap13-uPortal/Everyone.group-membership.xml
```

Contrôle : se connecter au portail avec le login/mot de passe esup/esup sur https://ent.univ.fr et ouvrir le gestionnaire de groupes via les outils d'administration :

Visualisation de la racine du SmartLDAPGroupStore via le gestionnaire de groupes du portail.

2 Utilisation des groupes dans ESUP-Portail

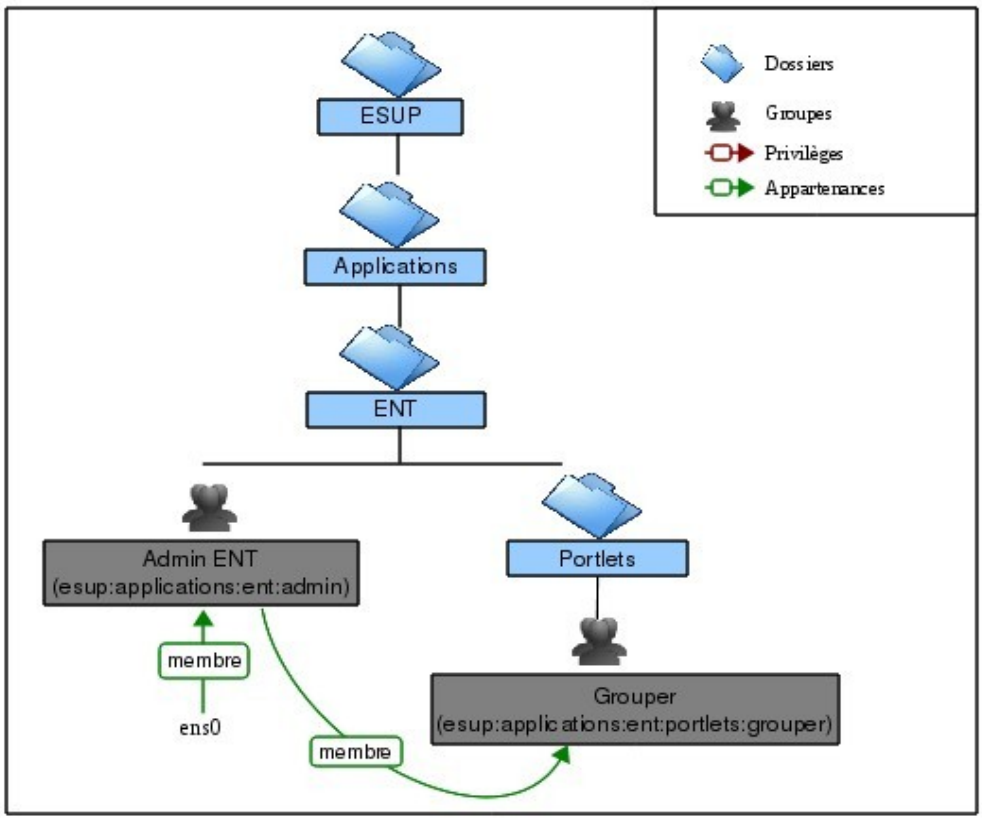
L'intégration des groupes issus de Grouper dans le portail est testée à travers trois exemples :

- Utilisation d'un groupe grouper comme sous groupe du groupe d'administration du portail.
- Utilisation d'un groupe grouper pour cibler l'audience d'un fragment.
- Utilisation d'un groupe grouper pour cibler l'audience d'une portlet.

Références :

https://wiki.jasig.org/display/UPM40/Fragment+Layout%3A+Using+xml+files
https://wiki.jasig.org/display/UPM40/Add+Portal+Admininstrator
https://wiki.jasig.org/display/UPM40/Add+Local+User+Accounts

Pour effectuer ces différents tests l'arborescence suivante est créée dans Grouper :



Groupes utilisés pour l'intégration dans le portail.

Le script gsh suivant permet de créer cette arborescence :

```
cd /opt/grouper/grouper.api
./bin/gsh.sh ~/TP_Grouper/conf/Chap13-uPortal/uportal-groups.gsh
```

Publication des groupes dans l'annuaire :

```
bin/gsh.sh -psp -bulkSync
```

Ajout d'un sous-groupe au groupe d'administration du portail

Script : ~/TP_Grouper/scripts/Chap13_uPortal-subGroup.sh

Le groupe grouper ***Admin ENT*** est ajouté comme membre du groupe uPortal ***Portal Administrators.*** Cet ajout peut être réalisé directement via l'interface ou par import xml :

```
cd /opt/esup-uportal/
ant data-import -Dfile=/home/esup/TP_Grouper/conf/Chap13-
uPortal/Portal_Administrators.group-membership.xml
```

La définition du fragment d'administration est modifiée pour prendre en compte les sous groupes du groupe d'administration :

```
ant data-import -Dfile=/home/esup/TP_Grouper/conf/Chap13-uPortal/admin-
lo.fragment-definition.xml
```

Enfin, le portail est redémarré :

```
su -c "service tomcat-esup restart"
```

Contrôle : se connecter au portail avec le login/mot de passe ens0/esup et vérifier l'accès à l'onglet d'administration.

Ciblage d'un fragment et d'une portlet

Script : ~/TP_Grouper/scripts/Chap13_uPortal-groups-fragment.sh

Dans cet exemple on ajoute un fragment ciblé sur le groupe ***esup:applications:ent:admin.*** Ce fragment contient un onglet dans lequel on affiche la portlet de gestion des groupes ainsi que Grouper-UI en iframe, ciblé sur groupe ***esup:applications:ent:portlets:grouper.***

Création du propriétaire du fragment :

```
ant data-import -Dfile=/home/esup/TP_Grouper/conf/Chap13-
uPortal/groups-admin-lo.user.xml
```

Définition d'une portlet de type iframe pour afficher Grouper UI :

```
ant data-import -Dfile=/home/esup/TP_Grouper/conf/Chap13-
uPortal/grouper.portlet-definition.xml
```

Définition du fragment :

```
ant data-import -Dfile=/home/esup/TP_Grouper/conf/Chap13-
uPortal/groups-admin-lo.fragment-definition.xml
```

Définition du layout :

```
ant data-import -Dfile=/home/esup/TP_Grouper/conf/Chap13-
uPortal/groups-admin-lo.fragment-layout.xml
```

Le portail est redémarré :

```
su -c "service tomcat-esup restart"
```

Contrôle :

- Se connecter au portail avec le login/mot de passe ens1/esup pour vérifier l'environnemnt inital de l'utilisateur ens, puis se déconnecter.
- Se connecter en tant que ens0/esup et ajouter l'utilisateur ens1 au groupe ***esup:applications:ent:admin*,** puis se déconnecter.
- Si le service associé à PSP est arrêté, répercuter la modification dans l'annuaire :

```
cd /opt/grouper/grouper.api  && bin/gsh.sh -psp -bulkSync
```

- Se connecter à nouveau en tant qu'ens1/esup pour vérifier que le nouveau fragment est bien affiché.

XIV Grouper loader : groupes automatiques

Le module Grouper Loader est un composant du démon Grouper. Il permet d'automatiser la gestion d'appartenances à des groupes à partir de sources SQL ou LDAP.

Ressources :
https://spaces.internet2.edu/display/Grouper/Grouper+Daemon
https://spaces.internet2.edu/display/Grouper/Grouper+-+Loader
https://spaces.internet2.edu/display/Grouper/Grouper+-+Loader+LDAP

Pour les exemples de chargement à partir d'une source SQL, la table subjectattributes du chapitre VIII est à nouveau utilisée.

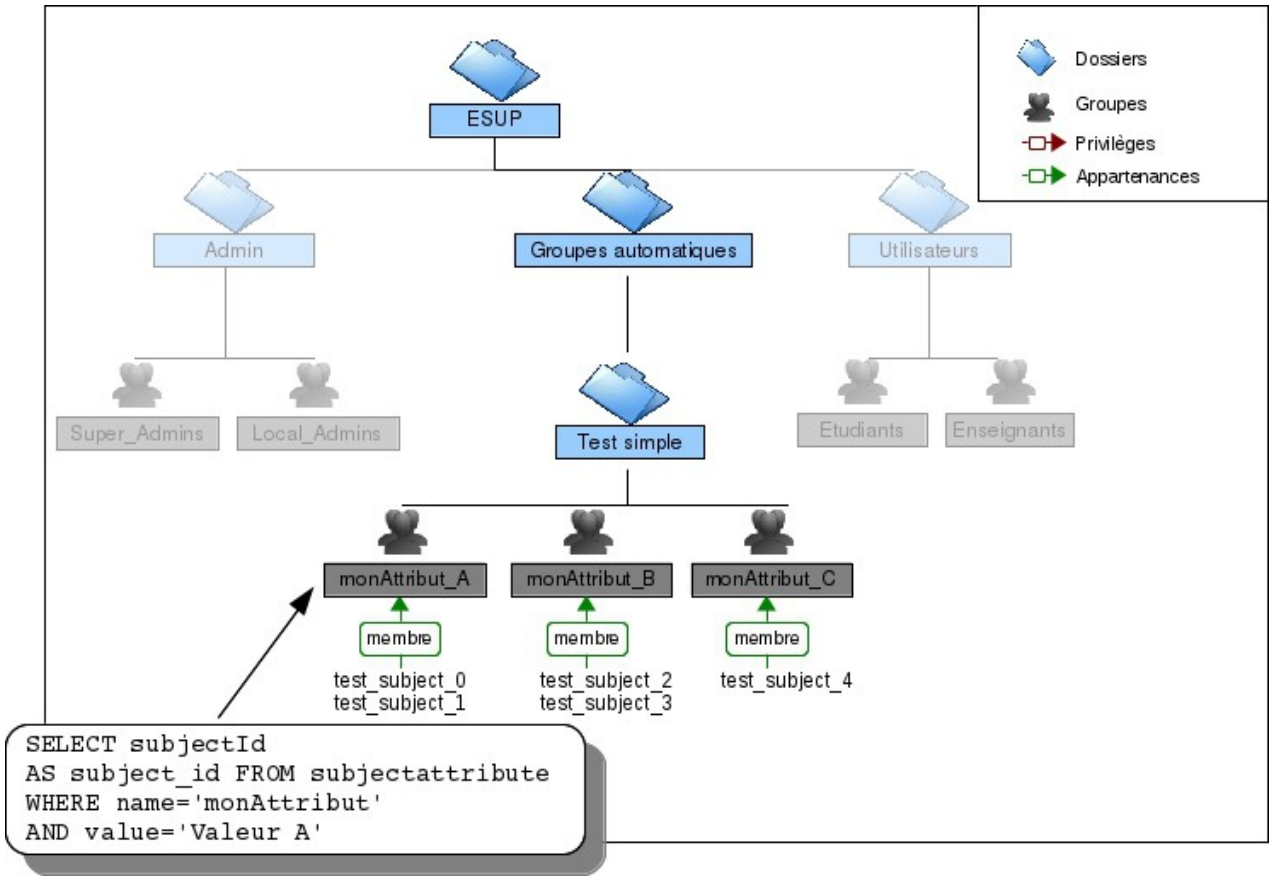
Si nécessaire, réinitialiser la table :

```
psql -U postgres -h localhost grouper_2_1_5 -f  ~/TP_Grouper/conf/Chap8-multi-
sources/subjects.sql

psql -U postgres grouper_2_1_5 -h localhost -c 'select * from subjectattribute;'

+----------------+-------------+-----------------------------------+------------+
| subjectId      | name        | value                             | searchValue|
+----------------+-------------+-----------------------------------+------------+
| test_subject_0 | name        | DB User 0                         |            |
| test_subject_0 | description | Utilisateur de test 0 depuis SQL  |            |
| test_subject_0 | mail        |  test_subject_0@univ.fr           |            |
| test_subject_0 | monAttribut | Valeur A                          |            |
| test_subject_1 | name        | DB User 1                         |            |
| test_subject_1 | description | Utilisateur de test 1 depuis SQL  |            |
| test_subject_1 | mail        | test_subject_1@univ.fr            |            |
| test_subject_1 | monAttribut | Valeur A                          |            |
| (...)                                                            |
+----------------+-------------+-----------------------------------+------------+
```

1 Chargement automatique des membres de groupes

Script : *~/TP_Grouper/scripts/Chap14_grouperLoader-simple.sh*

L'objectif de cet exemple va être de créer 3 groupes, un par valeur de l'attribut **MonAttribut** et de les alimenter automatiquement via le loader de Grouper.



Alimentation automatique de groupes à partir de requêtes SQL.

La configuration du loader se fait via le fichier ***grouper-loader.properties.*** Dans ce fichier il faut s'assurer que la propriété suivante est à true pour que les attributs nécessaires au loader soient bien initialisés :

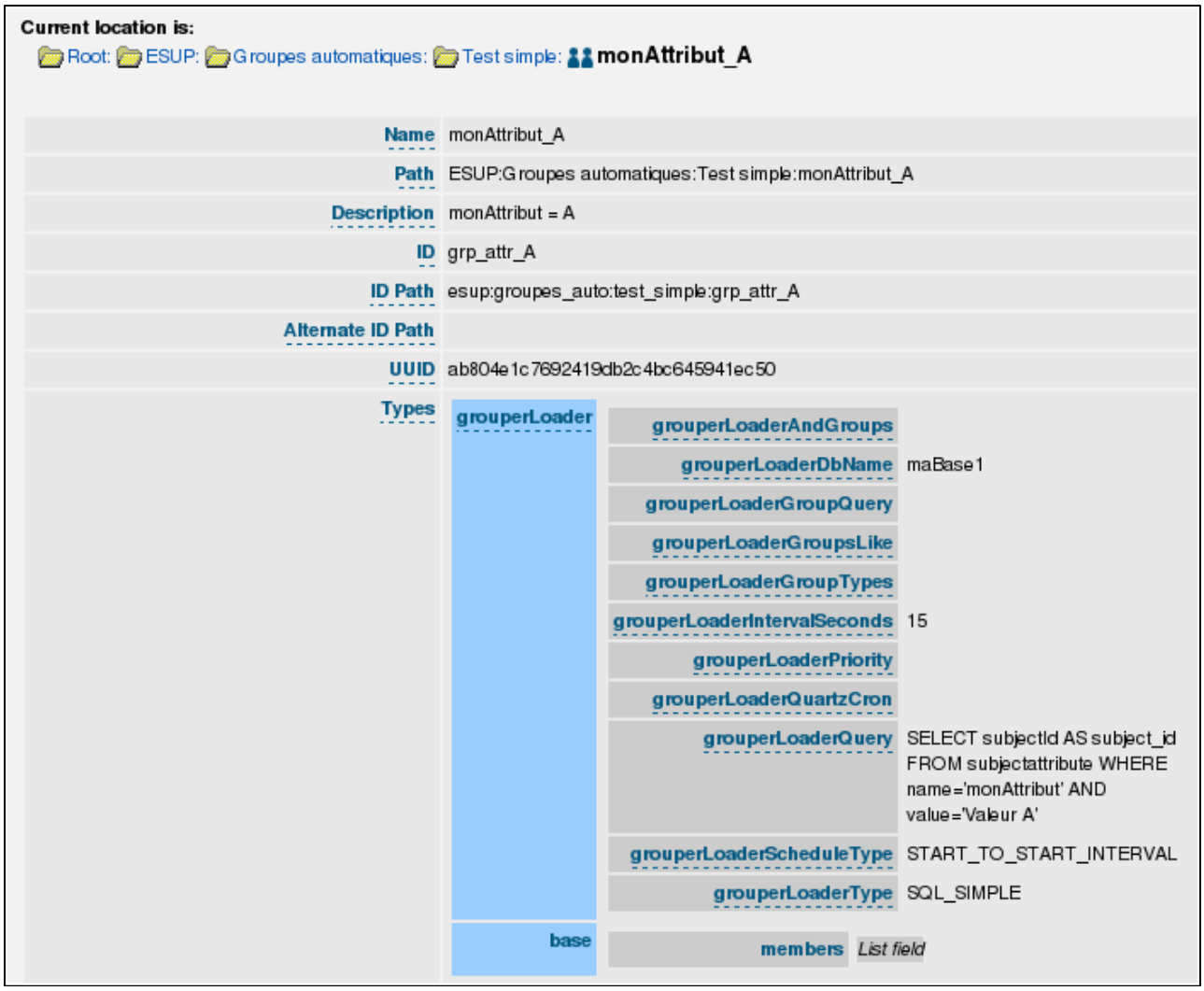
```
loader.autoadd.typesAttributes = true
```

Il faut ensuite configurer l'accès à une ou plusieurs bases de données. Le nom situé après db, ici maBase1, correspond au nom d'accès à la base qui va être utilisé ensuite au niveau de Grouper :

```
db.maBase1.user = esup4
db.maBase1.pass = esup
db.maBase1.url = jdbc:postgresql://localhost:5432/grouper_2_1_5
```

```
cd /opt/grouper/grouper.api
cp ~/TP_Grouper/conf/Chap14-grouperLoader/grouper-loader.properties ./conf
```

Pour créer un groupe automatique, il faut lui ajouter le type **grouperLoader** puis éditer les attributs correspondants :



Visualisation d'un groupe automatique dans l'interface de Grouper.

Ce type de groupe peut également être créé via le shell Grouper :

```
./bin/gsh.sh ~/TP_Grouper/conf/Chap14-grouperLoader/grouper_loader_simple.gsh
```

Le loader doit ensuite être démarré :

```
./bin/gsh.sh -loader
```

Contrôle : se connecter en tant que ens0/esup sur https://grouper.univ.fr et visualiser les membres de l'un de ces groupes. Changer l'attribut de l'un des membres et vérifier la répercussion du changement :

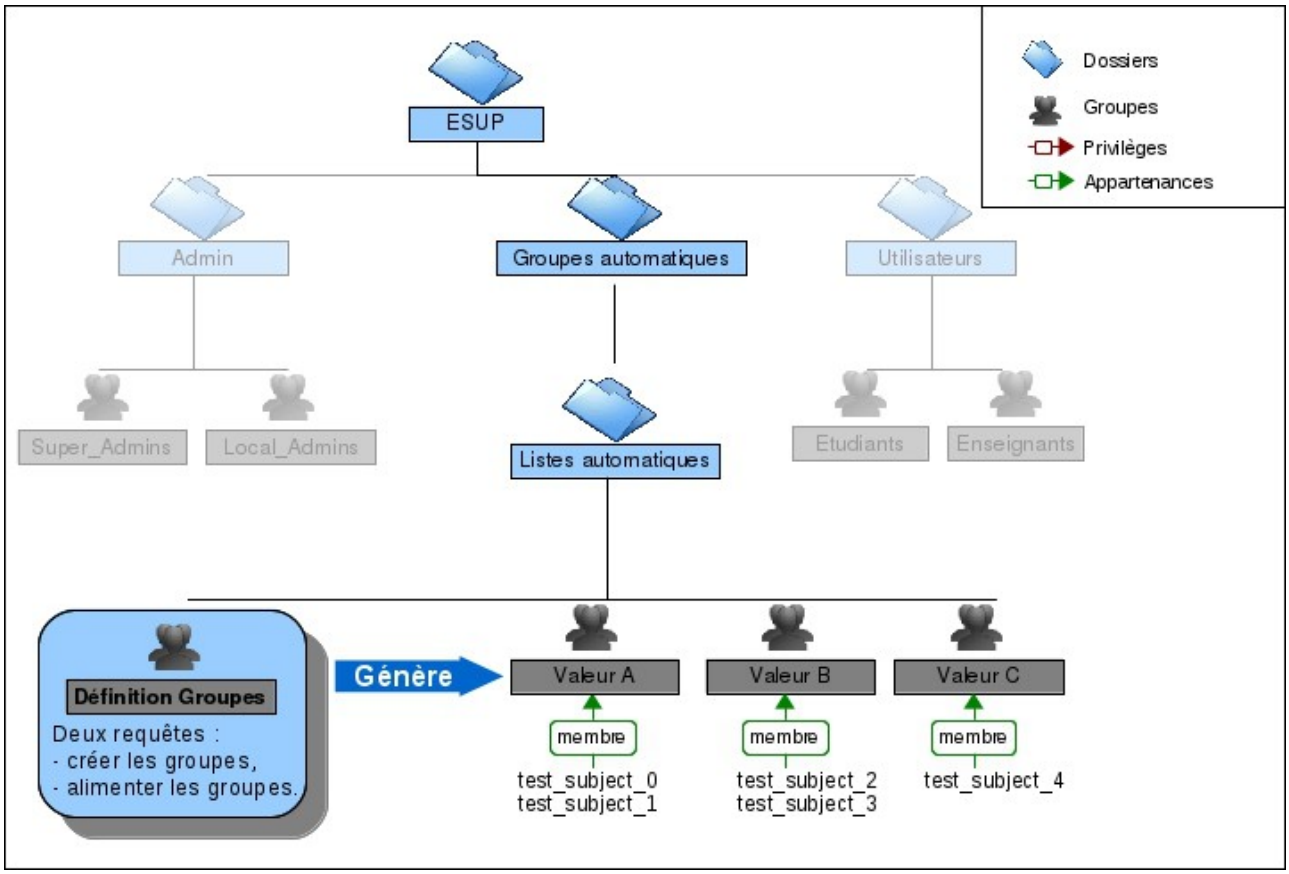
```
psql -U postgres grouper_2_1_5 -h localhost -c "update subjectattribute set
value='Valeur B' where subjectid='test_subject_0' and name='monAttribut'"
```

2 Liste de groupes automatiques

Script : *~/TP_Grouper/scripts/Chap14_grouperLoader-liste.sh*

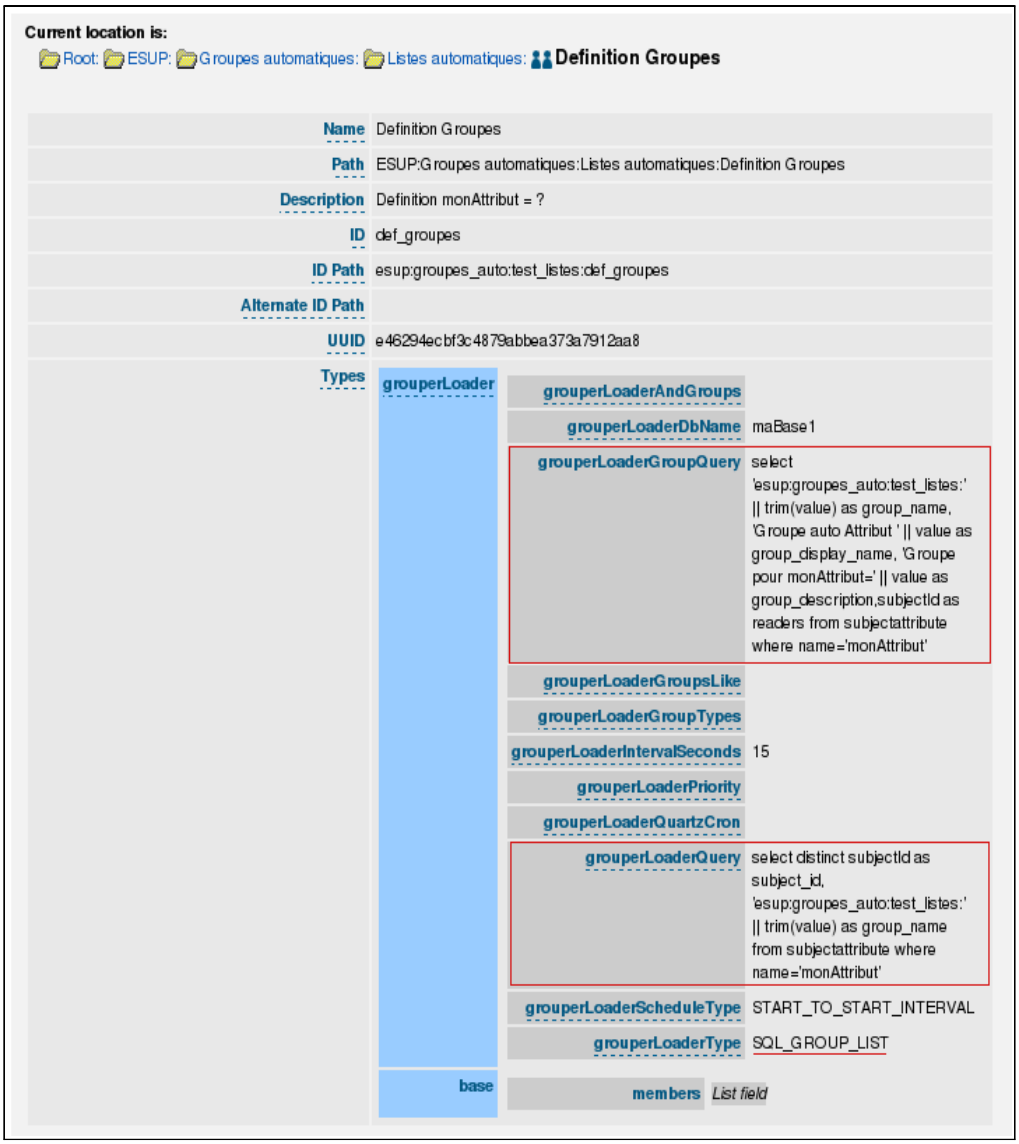
Le loader de Grouper permet également de définir automatiquement des listes de groupes avec leurs appartenances et privilèges. Pour définir une liste de groupes automatiques il faut créer un groupe avec le type **grouperLoader** et assigner la valeur **SQL_GROUP_LIST** dans l'attribut associé **grouperLoaderType.** Deux requêtes sont ensuite nécessaires. La première, spécifiée via l'attribut **grouperLoaderGroupQuery,** permet de créer les groupes avec leur noms, noms d'affichage, et description et de positionner les privilèges. La seconde associée à l'attribut **grouperLoaderQuery** a pour objectif de peupler les groupes**.**

Dans cet exemple, une liste de groupes est utilisée pour créer et peupler trois groupes , un par valeur de l'attribut monAttribut :



Groupes créés via une définition de liste de groupes.

Cette définition de liste de groupes peut être créée via l'interface graphique en attribuant au groupe le type grouperLoader et en renseignant les attributs comme dans la capture suivante :



Exemple de liste de groupes.

Pour créer le groupe via le shell grouper :

```
./bin/gsh.sh ~/TP_Grouper/conf/Chap14-grouperLoader/grouper_loader_liste.gsh
```

Puis lancer le loader :

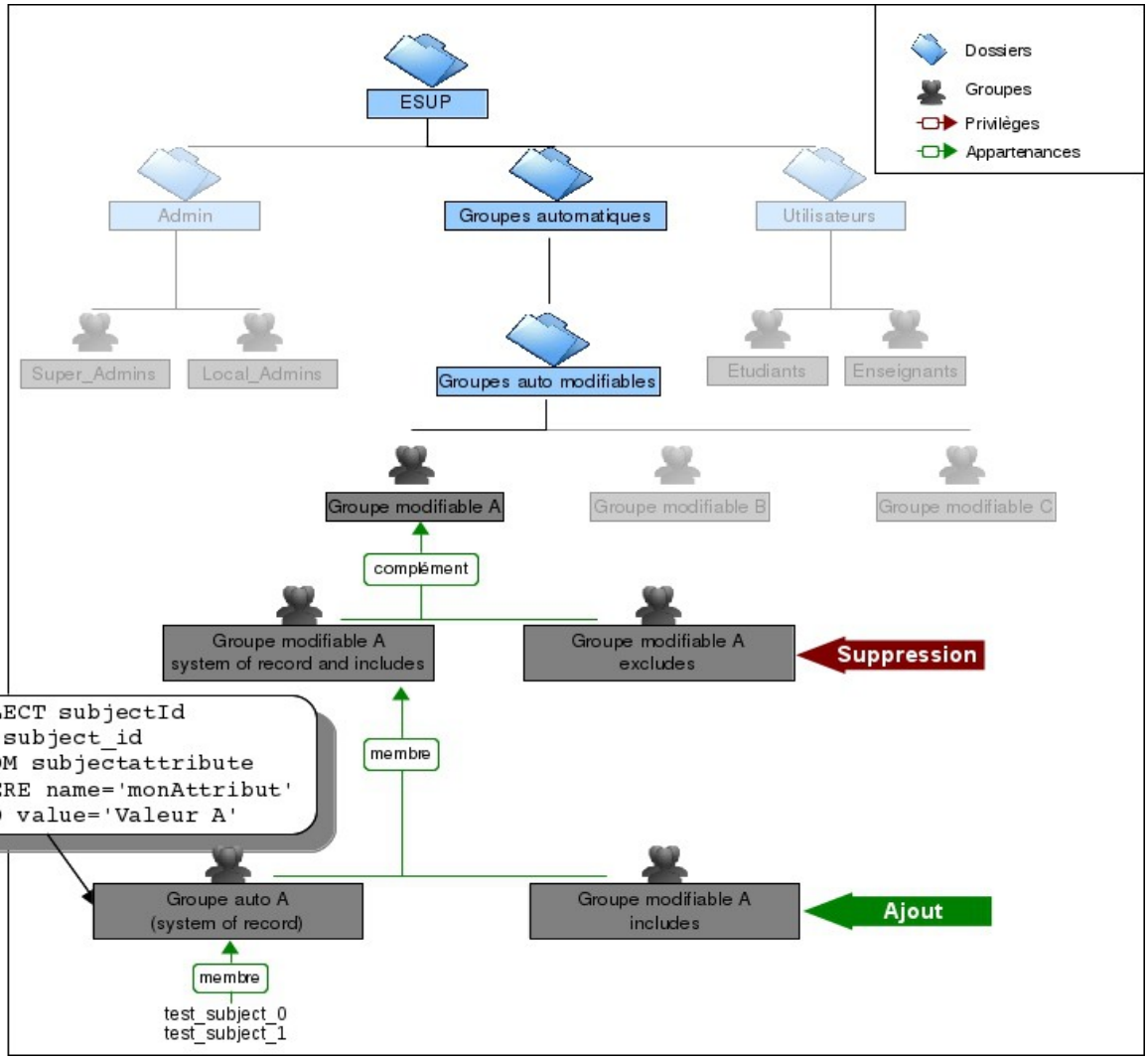
```
./bin/gsh.sh -loader
```

3 Groupes automatiques modifiables

Script : *~/TP_Grouper/scripts/Chap14_grouperLoader-incl-excl.sh*

Les groupes créés dans les exemples précédents ne permettent pas de modifier leurs membres. Pour contourner ce problème, il est possible de créer non pas un groupe mais une structure de groupes qui va permettre d'ajuster les appartenances par l'intermédiaire de deux groupes, un pour ajouter des membres et un autre pour en exclure. L'exclusion de membres s'appuie sur la possibilité, au niveau de Grouper, de créer des groupes via les opérations ensemblistes et ici c'est le complément qui est utilisé.

Le schéma suivant représente la structure de groupes obtenue :



Structure de groupe automatique avec possibilité de modifications

Pour obtenir cette structure, 2 groupes sont créés :

- **Groupe auto A** qui possède le type **grouperLoader** pour l'alimenter à partir d'une requête SQL. L'extension de ce groupe est celle de son groupe de rattachement avec le suffix _sytemOfRecord, (e.g. : grp_attr_A_systemOfRecord).
- **Groupe modifiable A** qui possède le type **addIncludeExclude,** pour créer la structure de groupes pour l'inclusion et l'exclusion de membres. Ce type de groupes peut être utilisé indépendamment du loader.

Pour que ce mécanisme puisse être utilisé, la propriété suivante doit être à true dans le fichier de configuration de Grouper ***grouper.properties :***

```
grouperIncludeExclude.use = true
```

```
cd /opt/grouper/grouper.api
cp ~/TP_Grouper/conf/Chap14-grouperLoader/grouper-incl-excl.properties
./conf/grouper.properties
```

Pour créer ces groupes via le shell Grouper :

```
./bin/gsh.sh ~/TP_Grouper/conf/Chap14-grouperLoader/grouper_loader_incl_excl.gsh
```

Le loader doit également être relancé :

```
./bin/gsh.sh -loader
```

Contrôle : se connecter en tant que ens0/esup sur https://grouper.univ.fr et modifier les membres des groupes pour les inclusions et les exclusions.

4 Chargement automatiques de groupes LDAP

Script : *~/TP_Grouper/scripts/Chap14_grouperLoader-ldap-liste.sh*

Le loader Grouper permet également de charger des groupes ou des listes de groupes depuis un annuaire LDAP. Cette possibilité du loader peut être utile, notamment, dans le cas de la prise en compte d'un existant. Dans l'exemple suivant, une liste de groupes est créée à partir de groupes existant dans un annuaire LDAP.

Ressources :
https://spaces.internet2.edu/display/Grouper/Grouper+-+Loader+LDAP
https://spaces.internet2.edu/display/Grouper/Grouper+Loader+LDAP+example

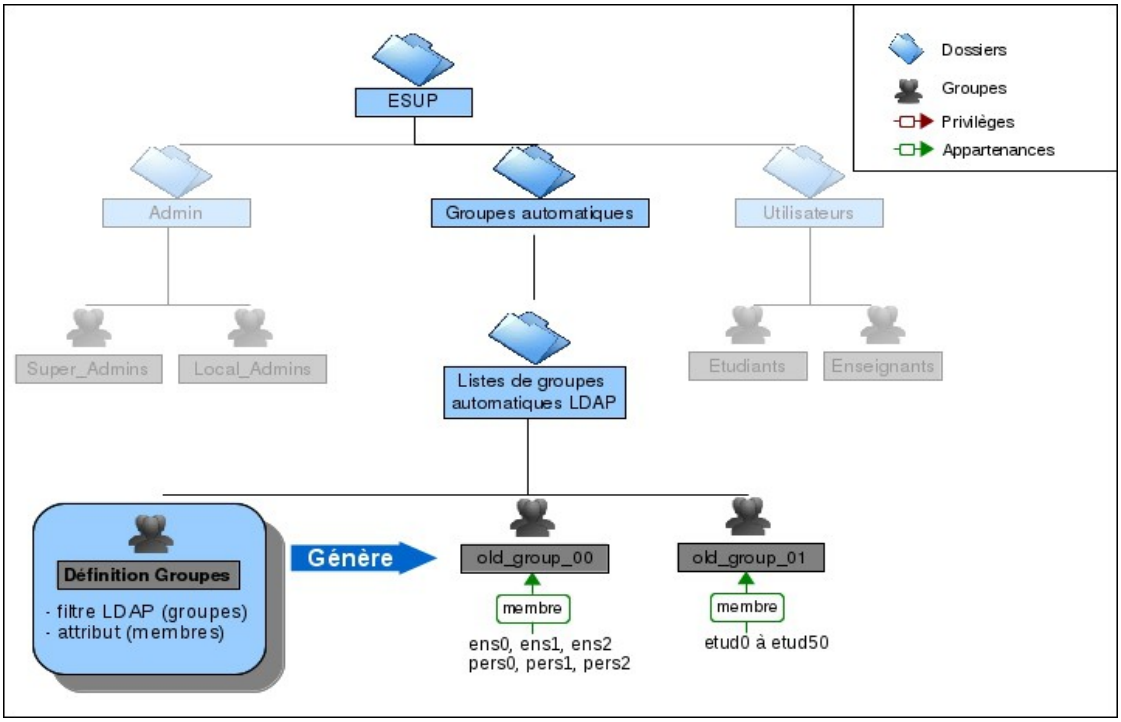
Au préalable une nouvelle branche, ou=groups_org, dc=esco-portail, dc=fr, est créée dans l'annuaire avec 2 groupes qui vont constituer la source de données pour le loader :

- old_group_00 : contient les membres ens0, ens1, ens2, pers0, pers1 et pers2.
- old_group_01 : contient les membres etud0 à etud50.

Pour créer cette branche, le fichier ***groups_org.ldif*** est chargé dans l'annaire :

```
su -c  "service slapd stop"
su -c "/usr/sbin/slapadd -v -l /home/esup/TP_Grouper/conf/Chap14-
grouperLoader/groups_org.ldif -b \"dc=esup-portail,dc=org\" -d 256"
su -c "chown -R openldap:openldap /var/lib/ldap/"
su -c "service slapd start"
```

L'objectif est de définir une liste de groupes pour créer et alimenter les groupes de cette branche dans Grouper :



Groupes importés depuis LDAP dans Grouper via une liste de groupes.

De la même façon que pour les imports depuis une base de données SQL, une ou plusieurs sources de données doivent être définies dans grouper-loader.properties :

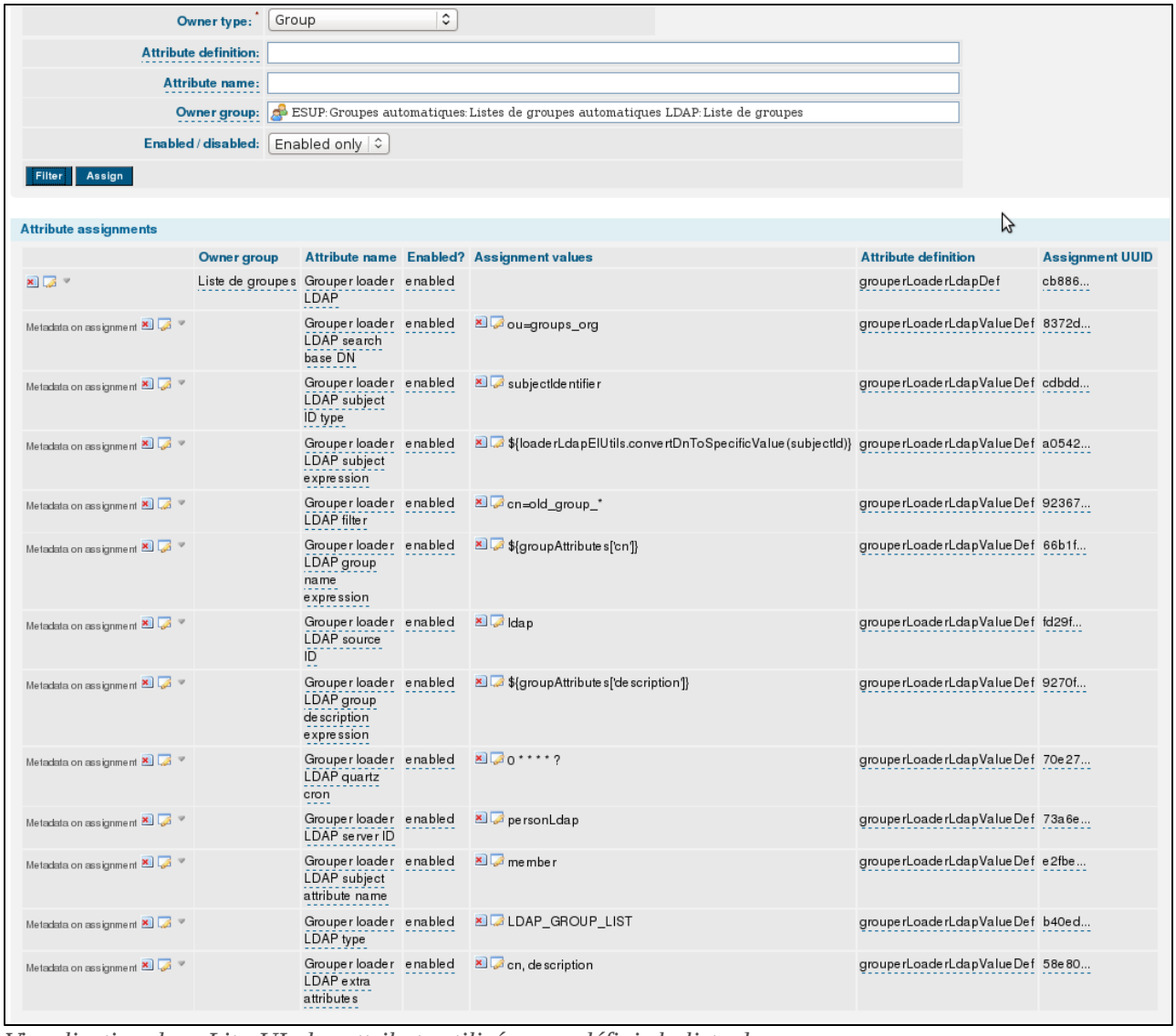
```
ldap.personLdap.url = ldap://ldap.univ.fr:389/dc=esup-portail,dc=org
ldap.personLdap.user = cn=admin,dc=esup-portail,dc=org
ldap.personLdap.pass = /opt/grouper/grouper.api/conf/ldapkey.txt
```

Remarque : dans cet exemple, le nom de la source de données est personLdap.

```
cd /opt/grouper/grouper.api
cp ~/TP_Grouper/conf/Chap14-grouperLoader/grouper-loader-ldap.properties
./conf/grouper-loader.properties
```

Pour les sources de données de type LDAP, le loader utilise le framework d'attributs. Le principe est d'assigner l'attribut **grouperLoaderLdap** au groupe qui va contenir la définition des groupes à charger, puis de le paramétrer en assignant des attributs à cette assignation (méta-datas).

Les attributs assignés à un objet ne sont visibles que depuis l'interface Lite-UI :



Visualisation dans Lite-UI, des attributs utilisés pour définir la liste de groupes.

Le script shell Grouper suivant permet de créer cette définition de liste de groupes LDAP :

```
./bin/gsh.sh ~/TP_Grouper/conf/Chap14-grouperLoader/grouper_loader_ldap_liste.gsh
```

La fin du script comporte l'instruction suivante qui permet d'invoquer le loader :

```
loaderRunOneJob(groupsDef);
```

Contrôle : se connecter en tant que ens0/esup sur https://grouper.univ.fr et visualiser les groupes créés.

Remarque : pour un exemple de groupes LDAP simples voir les scripts suivants :

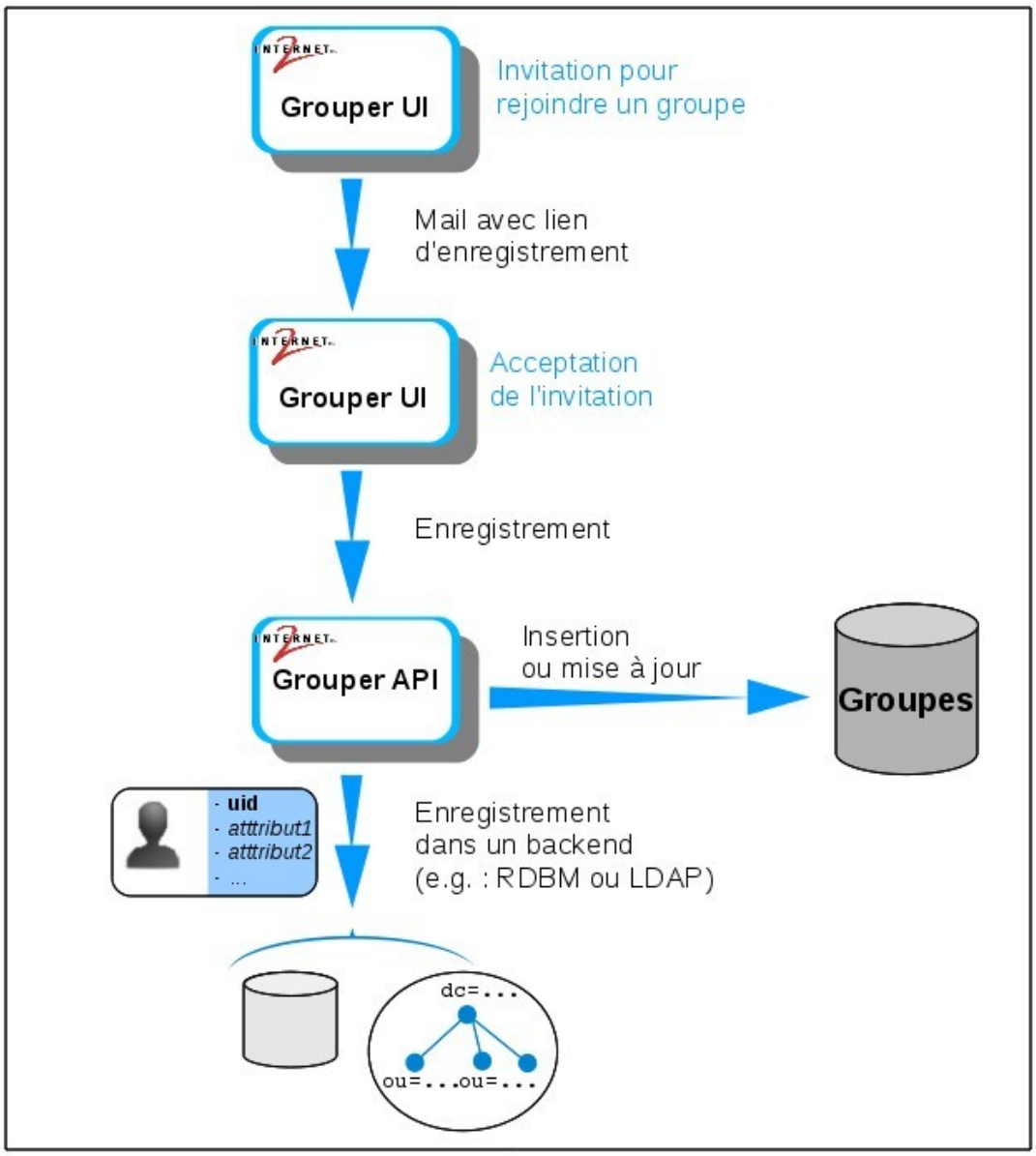
~/TP_Grouper/scripts/Chap14_grouperLoader-ldap-simple.sh
~/TP_Grouper/conf/Chap14-grouperLoader/grouper_loader_ldap_simple.gsh

XV Enregistrement de membres extérieurs

Script : *~/TP_Grouper/scripts/Chap15_externalSubjects.sh*

Nativement, Gouper permet d'enregistrer dans des groupes des personnes extérieures au SI. Cette fonctionnalité peut-être intéressante, notament dans le cas d'une fédération d'identités. Le principe est d'autoriser les personnes extérieures à s'enregistrer. Cette autorisation peut être condionnée à une invitation préalablement envoyée par mail. Les personnes enregistrées sont ensuite ajoutées automatiquement à un ou plusieurs groupes.

Ressources :
https://spaces.internet2.edu/display/Grouper/Grouper+external+subjects
https://spaces.internet2.edu/display/Grouper/Grouper+external+users+on+demo+server



Scénario possible d'enregistrement de personnes extérieures.

Dans cet exemple, une invitation sera envoyée pour rejoindre un groupe nouvellement créé. Une fois enregistrée la personne invitée sera ajoutée à ce groupe. En outre, tous les membres extérieurs seront systématiquement ajoutés au groupe ***esup:exterieurs.***

Pour simuler l'authentification de ces personnes extérieures au SI, on utilise l'authentification de base de Grouper, associée au fichier ***tomcat-users.xml.***

```
cp ~/TP_Grouper/conf/Chap15-externalSubjects/tomcat-users.xml
/opt/grouper/tomcat-grouper/conf
```

```
<user username="ext.user1@instit1.fr" password="esup" roles="grouper_user"/>
<user username="ext.user2@instit1.fr" password="esup" roles="grouper_user"/>
<user username="ext.user3@instit1.fr" password="esup" roles="grouper_user"/>
<user username="ext.user4@instit2.fr" password="esup" roles="grouper_user"/>
<user username="ext.user5@instit2.fr" password="esup" roles="grouper_user"/>
```

La configuration est réalisée dans la section ExternalUsers du fichier ***grouper.properties.*** Un grand nombre d'éléments peuvent être paramétrés dans ce fichier : attibuts, modèles de mail, délai d'expirtation, etc. A ce niveau, les propriétés suivantes sont modifiées pour utiliser une source de données interne à Grouper pour les personnes extérieures au SI :

```
externalSubject.sourceId = grouperExternal
externalSubject.sourceName = External Users
externalSubjects.autoCreateSource = true
```

On paramètre également l'ajout des personnes extérieures au groupe ***esup:exterieurs :***

```
externalSubjects.autoaddGroups=esup:exterieurs
```

Enfin, pour pouvoir visualiser les mails, on utilise une propriété de test pour le serveur smtp :

```
mail.smtp.server = testing
```

```
cp ~/TP_Grouper/conf/Chap15-externalSubjects/grouper.properties ./conf
```

Les logs associés aux mails sont redirigés vers le fichier ***grouper.api/logs/grouper_email.log :***

```
cp ~/TP_Grouper/conf/Chap15-externalSubjects/log4j.properties ./conf
```

Le groupe des utilisateurs extérieurs, ***esup:exterieurs,*** est créé :

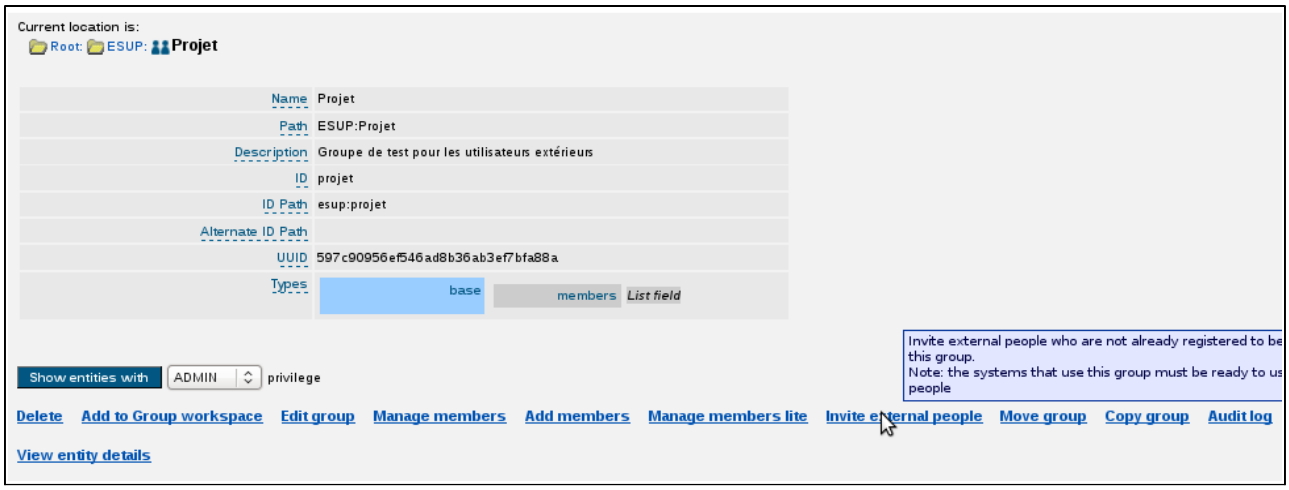
```
./bin/gsh.sh ~/TP_Grouper/conf/Chap15-externalSubjects/external-users.gsh
```

Pour finir, les interfaces utilisateurs sont paramétrées pour afficher les liens d'inviation, dans le fichier ***./conf/resources/grouper/media.properties***

```
inviteExternalPeople.link-from-admin-ui = true
 inviteExternalPeople.link-from-lite-ui = true
```

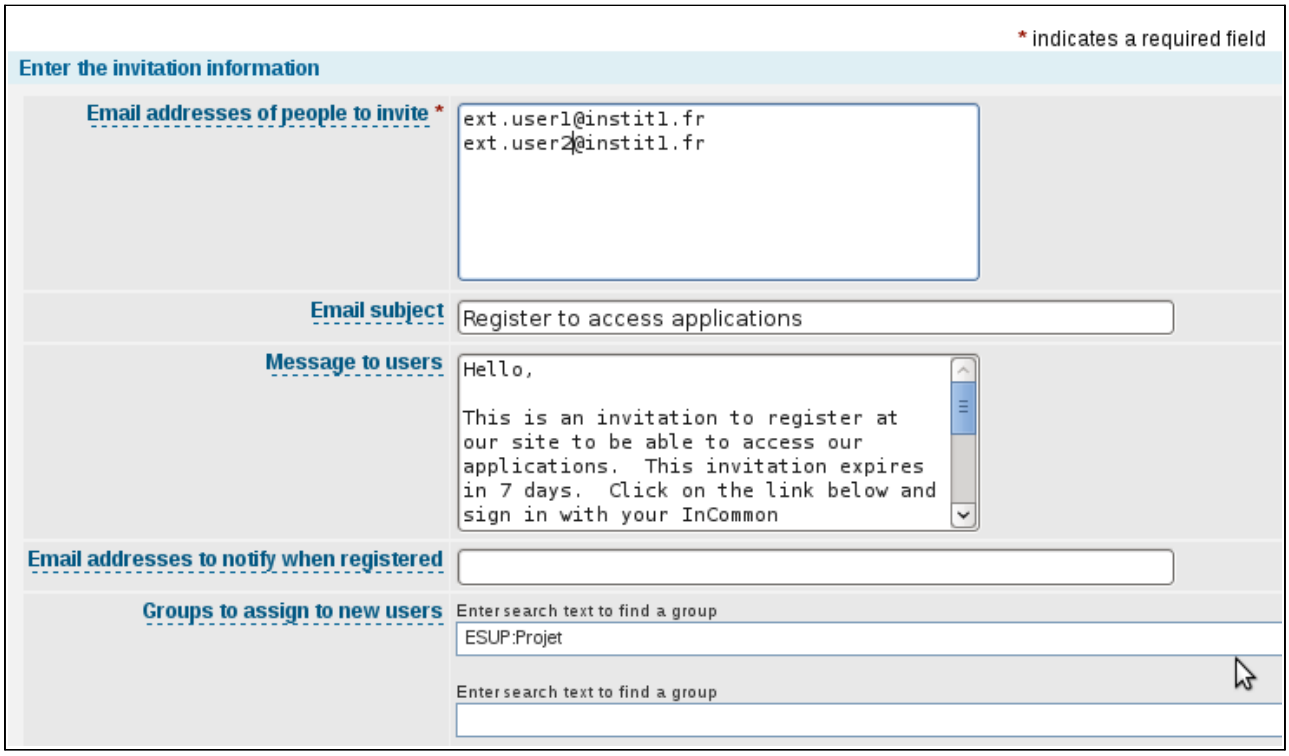
```
cp ~/TP_Grouper/conf/Chap15-externalSubjects/media.properties
./conf/resources/grouper/
```

Pour tester cette fonctionnalité, se connecter en tant qu'ens0/esup, créer un groupe, par exemple ***esup:projet*** puis cliquer sur le lien **invite external people :**



Lien d'invitation pour des personnes extérieures.

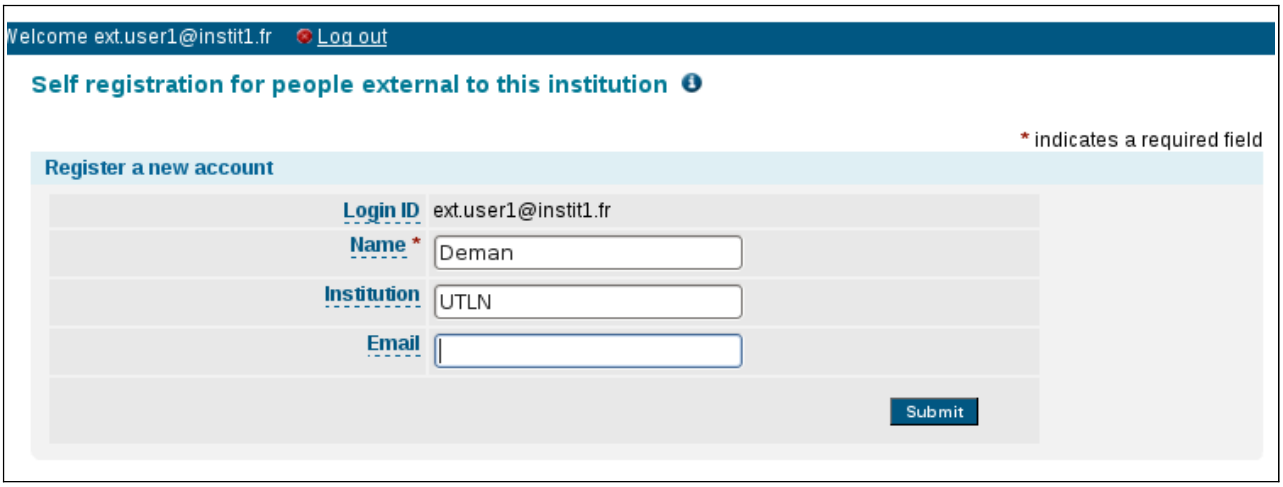
Dans l'écran suivant saisir les adresses mail des personnes à inviter :



Ecran de saisie des adresses des personnes à inviter.

Les mails envoyés avec les liens d'invitation sont loggués dans le fichier ***/opt/grouper/grouper.api/logs/grouper_email.log.*** Ourir l'un des liens dans un navigateur et se logguer avec un compte extérieur, ext.user1@instit1.fr/esup.

Ce lien est de la forme :
http://grouper.univ.fr/grouperExternal/appHtml/grouper.html?operation=ExternalSubjectSelfRegister.externalSubjectSelfRegister&externalSubjectInviteId=cd56b4712cf94f0f9ce0882f54a8c89d



Ecran d'acceptation d'inviation.

Enfin vérifier, en tant qu'ens0/esup, que cet utilisateur a bien été ajouté dans les groupes ***esup:projet*** et ***esup:exterieurs.***

XVI Synchronisation de groupes entre instances de Grouper

Script : *~/TP_Grouper/scripts/Chap16_syncGrouper.sh*

Il est possible de synchroniser les membres de groupes entre différentes instances de Grouper. Cette fonctionalité s'appuie sur le mécanisme des external subjects du chapitre précédent et utilise également le loader de Grouper ainsi que le web service. Différents types de synchonisation sont possibles : pull, push et push incrémental. Dans cet exemple la synchronisation de type pull est testée.

Pour réaliser ce test on installe une deuxième instance de Grouper et du web service à partir desquels les groupes vont être synchonisés :

```
~/TP_Grouper/scripts/Chap16_syncGrouper-Grouper2.sh
```

Vérifier l'installation de l'interface graphique en se connectant en tant que GrouperSystem/esup sur l'URL : https://grouper2.univ.fr

Vérifier l'installation du web service en récupérant les membres d'un groupe :

```
wget 'https://GrouperSystem:esup@grouper-ws2.univ.fr/grouper-
ws2/servicesRest/v2_1_005/groups/esup2:exports:grp2_groupe1/members'
--header="Content-Type:text/x-json"
```

Sur cette nouvelle instance de Grouper, deux groupes vont constituer la source de données pour la synchronisation : ***esup2:exports:grp2_groupe1*** et ***esup2:exports:grp2_groupe2.***

Sur l'instance de destination, ces groupes seront synchronisés avec les groupes ***esup:imports:grp2_groupe1*** et ***esup:imports:grp2_groupe2.*** Ceux-ci peuvent être créés via le script gsh suivant :

```
./bin/gsh.sh ~/TP_Grouper/conf/Chap16-syncGrouper/sync-groups.gsh
```

Si nécesssaire, le loader est arrêté :

```
su -c "service grouper-loader stop"
```

La configuration des groupes synchronisés est réalisée au niveau du Grouper de destination dans le fichier ***grouper.api/conf/grouper.properties.***

Le premier point est d'activer la création de la source de données pour les sujets externes dans l'instance de destination :

```
externalSubject.sourceId = grouperExternal
externalSubject.sourceName = External Users
externalSubjects.autoCreateSource = true

externalSubjects.validateIndentiferLikeEmail=false
```

Il faut ensuite paramétrer l'accès au web service via grouperClient :

```
grouperClient.esupGrouperInstance2.id = esupGrouperInstance2
grouperClient.esupGrouperInstance2.properties.grouperClient.webService.url = https://grouper-ws2.univ.fr/grouper-ws2/servicesRest
grouperClient.esupGrouperInstance2.properties.grouperClient.webService.login = GrouperSystem
grouperClient.esupGrouperInstance2.properties.grouperClient.webService.password = esup
grouperClient.esupGrouperInstance2.properties.grouperClient.webService.client.version = v2_1_005
```

Ce paramétrage est basé sur l'utilisation de clés dans les noms de propriétés, pour pouvoir paramétrer plusieurs accès.

La partie suivante permet de définir la correspondance entre les sources de données des deux instances de Grouper :

```
grouperClient.esupGrouperInstance2.source.externalUser.id = jdbc
 grouperClient.esupGrouperInstance2.source.externalUser.local.sourceId = grouperExternal
grouperClient.esupGrouperInstance2.source.externalUser.local.read.subjectId = id
grouperClient.esupGrouperInstance2.source.externalUser.local.write.subjectId = idOrIdentifier
grouperClient.esupGrouperInstance2.source.externalUser.remote.sourceId = jdbc
grouperClient.esupGrouperInstance2.source.externalUser.remote.read.subjectId = id
grouperClient.esupGrouperInstance2.source.externalUser.remote.write.subjectId = idOrIdentifier
```

Les deux groupes à synchroniser sont ensuite paramétrés :

```
syncAnotherGrouper.syncGroup0.connectionName = esupGrouperInstance2
syncAnotherGrouper.syncGroup0.syncType = pull
syncAnotherGrouper.syncGroup0.cron =  */30 * * * * ?
syncAnotherGrouper.syncGroup0.local.groupName = esup:imports:grp2_groupe1
syncAnotherGrouper.syncGroup0.remote.groupName = esup2:exports:grp2_groupe1
syncAnotherGrouper.syncGroup0.addExternalSubjectIfNotFound = true
```

```
syncAnotherGrouper.syncGroup1.connectionName = esupGrouperInstance2
 syncAnotherGrouper.syncGroup1.syncType = pull
syncAnotherGrouper.syncGroup1.cron =  */30 * * * * ?
syncAnotherGrouper.syncGroup1.local.groupName = esup:imports:grp2_groupe2
syncAnotherGrouper.syncGroup1.remote.groupName = esup2:exports:grp2_groupe2
syncAnotherGrouper.syncGroup1.addExternalSubjectIfNotFound = true
```

```
cd /opt/grouper/grouper.api
cp ~/TP_Grouper/conf/Chap16-syncGrouper/grouper.properties ./conf/
```

Grouper-UI pour l'instance de destination est redéployé pour prendre en compte la source associée aux membres extérieurs :

```
cd /opt/grouper/grouper.ui
rm /opt/grouper/tomcat-grouper/webapps/grouper.war
ant clean war
cp /opt/grouper/grouper.ui/dist/grouper.war  /opt/grouper/tomcat-
grouper/webapps/
 su -c "service tomcat-grouper restart"
```

Enfin, le loader est redémarré :

```
su -c "service grouper-loader start"
```

Contrôle : se logguer sur https://grouper.univ.f et visualiser les membres des deux groupes synchronisés.

XVII Contacts

Listes Internet2 :

- https://www.internet2.edu/communities-groups/middleware/grouper-working-group/?edit-off#group-participate

Listes ESUP :

- https://listes.esup-portail.org/sympa/info/grouper-utilisateurs
- https://listes.esup-portail.org/sympa/info/grouper-devel

Remerciements

Merci à l'Université de Toulon pour m'avoir permis de consacrer tout le temps nécessaire à la mise au point de ce didacticiel.

Merci également à Vincent Bonamy pour avoir mis à disposition la machine virtuelle du workshop ESUP.
s