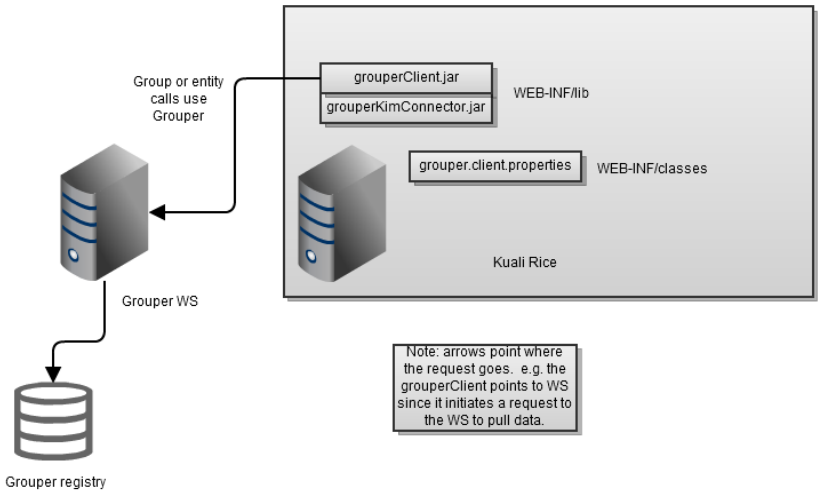
Grouper integration with Kuali Rice

This topic is discussed in the "Grouper Connectors" training video.

Grouper Integration with Kuali Rice

This section descibes the efforts to integrate Kuali Rice with Grouper. Kuali Rice KIM (Kuali Identity Management) has a Groups implementation and service, but Grouper offers different features which might scale and distribute better in an organization. For instance Grouper has unlimited folder levels, whereas Rice only has a namespace one level deep. Grouper can delegate control over folders and grouper. Also, Grouper can make composite groups so if someone stops being an active employee, they will fall out of other groups. Etc. If you wanted to run Kuali Rice and delegate some operations to Grouper, you can use the Grouper-Rice connector.

- Differences between Grouper and Kuali Rice KIM Groups
- Differences between Kuali Kim Rice identities and Grouper subjects
- Grouper Kuali Misc Functions
- Grouper Kuali Rice KIM connector design
- Grouper Kuali Rice workflow examples
- Grouper Kuali Rice Workflow Membership Provisioner



Installation of groups service

Get the grouper client, (note, this currently requires grouperClient v1.6+ unzip it or checkout and build):

```
[appadmin@lukes grouper]$ /usr/bin/svn export http://anonsvn.internet2.edu/svn/i2mi/trunk/grouper-misc
/grouperClient
[appadmin@lukes grouper]$ cd grouperClient
[appadmin@lukes grouperClient]$ ant
[appadmin@lukes grouperClient]$ cd ../..
```

Checkout the Grouper Kim connector:

```
[appadmin@lukes grouper]$ /usr/bin/svn export http://anonsvn.internet2.edu/svn/i2mi/trunk/grouper-misc
/grouperKimConnector
[appadmin@lukes grouper]$ cd grouperKimConnector/
[appadmin@lukes grouperKimConnector]$ cp build.example.properties build.properties
     --- edit build.properties, set where the grouperClient.jar was unzipped to, e.g.
     grouperClient.jar.name=../grouperClient/dist/grouperClient.jar
[appadmin@lukes grouperKimConnector]$ export JAVA_HOME=/opt/jdk1.6.0_16
[appadmin@lukes grouperKimConnector]$ export PATH=/opt/jdk1.6.0_16/bin:$PATH
     -- NOTE: you need Java 1.6 to build, but 1.5+ to run
[appadmin@lukes grouperKimConnector]$ ant
```

Copy grouperClient.jar and grouperKimConnector.jar to kr-dev/WEB-INF/lib (or whatever the Kuali rice webapp dir is if not kr-dev)

Copy grouper.client.properties to kr-dev/WEB-INF/classes (or whatever the Kuali rice webapp dir is if not kr-dev)

Edit the grouper.client.properties and set the WS connect string to the Grouper WS at your institution, include the user/pass or however you authenticate

Also insert and customize the grouperKimConnector settings in the grouper.client.properties file

Make a kr-dev/WEB-INF/classes/grouperKimOverride.xml

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
                           http://www.springframework.org/schema/tx
                           http://www.springframework.org/schema/tx/spring-tx-2.0.xsd
                           http://www.springframework.org/schema/aop
                           http://www.springframework.org/schema/aop/spring-aop-2.0.xsd">
  <bean id="kimGroupService" class="edu.internet2.middleware.grouperKimConnector.group.
GrouperKimGroupServiceImpl"/>
</beans>
```

Identify that file to Spring in Rice in rice-config.xml

```
<param name="rice.additionalSpringFiles">classpath:grouperKimOverride.xml</param>
```

If you are using the Rice sample data, you need to make sure whatever subject you are using exists in Grouper via GSH

```
addSubject("admin", "person", "admin");
addSubject("1", "person", "kr");
```

Whatever stem is configured in grouper.client.properties for kim should be created

```
kim.stem = kim
```

GSH setup:

```
grouperSession = GrouperSession.startRootSession();
new StemSave(grouperSession).assignName("kim").assignSaveMode(SaveMode.INSERT_OR_UPDATE).
assignCreateParentStemsIfNotExist(true).save();
```

If you are doing the eDocLite example, create those users and groups in GSH

```
addSubject("user1", "person", "user1");
addSubject("user2", "person", "user2");
addSubject("user3", "person", "user3");
addSubject("user4", "person", "user4");
new GroupSave(GrouperSession.staticGrouperSession()).assignSaveMode(SaveMode.INSERT_OR_UPDATE).assignName("kim:
KUALI:eDoc.Example1.IUB.Workgroup").assignCreateParentStemsIfNotExist(true).save();
new GroupSave(GrouperSession.staticGrouperSession()).assignSaveMode(SaveMode.INSERT_OR_UPDATE).assignName("kim:
KUALI:eDoc.Example1.IUPUI.Workgroup").assignCreateParentStemsIfNotExist(true).save();
addMember("kim:KUALI:eDoc.Example1.IUB.Workgroup", "user1");
addMember("kim:KUALI:eDoc.Example1.IUB.Workgroup", "user2");
addMember("kim:KUALI:eDoc.Example1.IUPUI.Workgroup", "user3");
addMember("kim:KUALI:eDoc.Example1.IUPUI.Workgroup", "user4");
new GroupSave(GrouperSession.staticGrouperSession()).assignSaveMode(SaveMode.INSERT_OR_UPDATE).assignName("etc:
webServiceUsers").assignCreateParentStemsIfNotExist(true).save();
addMember("etc:webServiceUsers", "GrouperSystem");
```

Installation of identity service

Follow the initial install instructions above. In the grouperKimOverride.xml, include this:

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
                           http://www.springframework.org/schema/tx
                           http://www.springframework.org/schema/tx/spring-tx-2.0.xsd
                           http://www.springframework.org/schema/aop
                           http://www.springframework.org/schema/aop/spring-aop-2.0.xsd">
  <bean id="kimGroupService" class="edu.internet2.middleware.grouperKimConnector.group.
GrouperKimGroupServiceImpl"/>
  <bean id="kimIdentityService" class="edu.internet2.middleware.grouperKimConnector.identity.
GrouperKimIdentityServiceImpl"/>
</beans>
```

Note that you need an email address subject attribute. Here is an example for the jdbc2 source in the Grouper sources.xml

```
<init-param>
       <param-name>subjectAttributeCol1</param-name>
       <param-value>email</param-value>
     </init-param>
     <init-param>
       <param-name>subjectAttributeName1</param-name>
       <param-value>EMAIL</param-value>
     </init-param>
```

Then you need to identify the email attribute in the grouper.client.properties

```
################################
## Kuali Identity settings
################################

kuali.identity.source.id.0 = pennperson
kuali.identity.source.nameAttribute.0 = name
kuali.identity.source.identifierAttribute.0 = PENNNAME
kuali.identity.source.emailAttribute.0 = EMAIL
kuali.identity.source.entityTypeCode.0 = PERSON

# separate a sourceId from a subjectId or sourceId
kuali.identity.sourceSeparator = ::::

# if there is this subjectId from grouper, dont untranslate to put sourceId::::subjectId
# multiple, comma separated
kuali.identity.ignoreSourceAppend.subjectIds = admin


# if the user has no email address, routing will have problems, use this default one
kuali.identity.defaultEmailAddress = idmProblems@institution.edu
```

Configure the grouper-ws.properties to send back those subject attributes:

```
# subject result attribute names when extended data is requested (comma separated)
# default is name, description
# note, these will be in addition to ws.subject.result.attribute.names
ws.subject.result.detail.attribute.names = name, description, PENNNAME, EMAIL
```

sd