v1.4.0 Grouper Web Services

Grouper Web Services as of v1.4.0

Introduction

Grouper web services (grouper-ws) is a J2EE web application which exposes common Grouper business logic through SOAP and REST. See FAQ.

To deploy the services, download the warfile and configure the property files (e.g. subject sources, databases, logging, etc). Configure authentication.

Note: there is a command line and java API web service client called Grouper Client

To implement a web service client:

1. Understand the object model. All grouper-ws services are operations based on simple data structures. The structures support Strings, ints, arrays, and structure references.
    a. Core web service API
    b. Example structure (only "getters" and "setters" are applicable properties)
    c. Each operation has many samples (authmated captures, versioned, and up to date). Here is an example
    d. Most options has a sensible default (e.g. MemberFilter defaults to All members)
    e. Lookup objects in various (consistent) ways. e.g. to delete a group, you can pass the name or uuid of the group.
2. Decide if you are using SOAP or REST (this is real REST, not Axis HTTP/XML)
    a. Both SOAP and REST support the same API
3. Inside SOAP and REST, each operation has two levels of complexity, the normal one, and the Lite one.
    a. Normal operation: can usually be batched (support a list of inputs, e.g. add multiple groups at once), supports complex inputs (arrays or structures)
    b. Lite operation: supports only inputs of scalars (no structures, no arrays... only Strings, ints, etc). In REST this also means that the request can be sent via query string only
4. If SOAP:
    a. Implement based on the WSDL. To get the WSDL of your running grouper web service deployment, use this URL: https://server.address/grouperWs/services/GrouperService?wsdl. Note, if your servlet is not grouperWs then adjust accordingly.
    b. There is a sample Java client with sample calls
5. If REST:
    a. Decide what format you want to send and receive data. grouper-ws supports XHTML, XML, and JSON, as well as query strings for input (in URL or message body)
    b. There are many samples

.NET client development guide

PHP client development guide

Guidelines For Working With Grouper Web Services

1. There is a bug we are tracking with Axis, where if you skip String params, it will mix up the params. So, if you are passing a param to a web service, make sure you pass empty strings for all null params before the param
2. Code clients with a mindset that the service might change in subtle ways. e.g. a result code might be added (check for success flag element, not success result code), an element might be added in a result object, another input element might be added to end of list, etc. Expect elements to be added in data
3. Make sure there is a property in the client of the URL and version for the service. The version of the service might change the URL (up to service deployer)...

Operations

- addMember: assign a member to a group
    - If already a member, that is ok
    - Accepts batches of members (non-Lite)
    - Accepts flag to say that any members not in batch should be removed (e.g. replace list)
- deleteMember: unassign a member from a group
    - If not a member, that is ok
    - Accepts batches of members (non-Lite)
- getMembers: return the members (including subject data) in a group (from direct or indirect membership)
    - Will accept member filter (All, Effective, Immediate, Composite)
    - Accepts batches of groups (non-Lite)
- getMemberships: under construction
    - Will accept member filter (All, Effective, Immediate, Composite)
    - Accepts batches of subjects and groups (non-Lite)
- hasMember: see if a subject is a member of a group
    - Will return true or false
    - Accepts batches of subject ids or identifers (returns batches of true's / false's) (non-Lite)
    - Will accept member filter (All, Effective, etc)

    - Can query on field (permission)
- getGroups: list groups for a subject
    - Will accept member filter (All, Effective, etc)
    - Accepts batches of subjects (non-Lite)
- groupSave
    - Create / update a group
    - Accepts batches of groups (non-Lite)
- groupDelete
    - Delete a group
    - Accepts batches of groups (non-Lite)
- getGrouperPrivileges
    - View privileges for a subject and (group or stem)
    - Can view all privileges for the subject and (group or stem) or a specific privilege
- assignGrouperPrivileges
    - Add or remove a privilege for a subject and (group or stem)
    - Will not fail if the privilege is already assigned or revoked
- findGroups
    - Can query for groups based on name, uuid, parent stem, or a substring query
    - Can create complex queries with group match (AND, OR, MINUS) (non-Lite)
- findStems
    - Can query for stems based on name, uuid, parent stem, or a substring query
    - Can create complex queries with group match (AND, OR, MINUS) (non-Lite)
- stemSave
    - Create / update a stem
    - Accepts batches of stems (non-Lite)
- stemDelete
    - Delete a stem
    - Accepts batches (non-Lite)
- memberChangeSubject
    - Change the subject of a current member
    - Accepts batches (non-Lite)

Features

- **API**
    - Batched operations (e.g. add 100 subjects to a group at once). There is a separate server-side max-in-batch param in the grouper-ws.properties.
    - Transaction support (if any fails in one batch request, rollback all in that single batch request)
- **Grouper Web Services Authentication**
    - Let container or web server handle
        - PKI
        - http-simple-auth
        - Source IP address filtering (TODO)
    - Custom authenticator
    - WS-Security
        - PKI
        - Kerberos
    - Proxying. The web service can execute operations based on an underlying user, not the authenticating user. Note the authenticating user must have appropriate permissions
- **Error Handling**
    - Error codes and error messages are sent in responses, as well as warnings. In batched mode, batches of response codes are returned. In REST, the http status code is used as well.
- **Clients**
    - Grouper will provide a quick start with Java, and it is up to users to create their own clients. The SOAP and REST are based on the HTTP documents, so any programming language will work
- **Web Service Implementation**
    - Apache Axis for SOAP, and home-grown for REST

Quick start

Checkout the appropriate projects under grouper-ws, read the README.txt in the grouper-ws/grouper-ws cvs directory

Build Script

The build script for grouper-ws is pretty basic. Generally just do the default (dist). There is also an "ant grouper" target to build a new grouper jar, and "ant quick" to do everything but generate the Axis files (takes 3 minutes).

```
C:\mchyzer\isc\dev\grouper\grouper-ws>ant
Buildfile: build.xml

dist:

clean:
   [delete] Deleting directory C:\mchyzer\isc\dev\grouper\grouper-ws\build
    [mkdir] Created dir: C:\mchyzer\isc\dev\grouper\grouper-ws\build\grouper-ws
    [mkdir] Created dir: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist
    [mkdir] Created dir: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws

compile:
    [javac] Compiling 10 source files to C:\mchyzer\isc\dev\grouper\grouper-ws\build\grouper-ws
    [javac] C:\mchyzer\isc\dev\grouper\grouper-ws\src\grouper-
ws\edu\internet2\middleware\grouper\webservices\GrouperSer
viceServlet.java:33: warning: [deprecation] getEPRForService(java.lang.String,java.lang.String) in org.apache.
axis2.tran
sport.TransportListener has been deprecated
    [javac] public class GrouperServiceServlet extends AxisServlet {
    [javac]        ^
    [javac] 1 warning

generate-aar:
    [mkdir] Created dir: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\webservices\classes
   [delete] Deleting directory C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\webservices\classes
    [mkdir] Created dir: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\webservices\classes
     [copy] Copying 13 files to C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\webservices\classes
      [jar] Building jar: C:\mchyzer\isc\dev\grouper\grouper-ws\webapp\WEB-INF\services\GrouperService.aar
      [jar] Building jar: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws.jar
    [mkdir] Created dir: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws\WEB-INF\classes
    [mkdir] Created dir: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws\WEB-INF\lib
     [copy] Copying 12 files to C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws\WEB-INF\classes
     [copy] Copying 30 files to C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws\WEB-INF\lib
     [copy] Copying 11 files to C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws
      [jar] Building jar: C:\mchyzer\isc\dev\grouper\grouper-ws\build\dist\grouper-ws.war

BUILD SUCCESSFUL
Total time: 22 seconds
The system cannot find the batch label specified - end

C:\mchyzer\isc\dev\grouper\grouper-ws>
```

Notice the generate-aar target. This is what makes the axis archive, which is all the classes needed for axis to determine the wsdl, along with the services.xml config file.

Axis is ~40 jars, though most of them are pretty axis specific. There is an ant target which will compress most of these into one jar (axisBundle.jar). Here is the ant help:

```
C:\mchyzer\isc\dev\grouper\grouper-ws>ant help
Buildfile: build.xml

help:
     [echo] Please ensure you have read the documentation -
     [echo] and created a build.properties file based on the template provided
     [echo]
     [echo] The following targets are available - type the appropriate name:
     [echo]
     [echo] 1) default (dist)
     [echo]      Simply builds, without cleaning, to the webapp.folder
     [echo] 2) clean
     [echo]      Clean the webapp folder, and classfiles, and build
     [echo] 3) generate-aar
     [echo]      Make the axis archive, which is the classfiles and services.xml that axis needs.  You need to
do this i
f you ever change anything that changes the wsdl.  You can do this automatically in dist by setting a property
in the bu
ild.properties
     [echo] 4) generate-axis-bundle-jar
     [echo]      Take all the bundlable axis jars (in lib/axis-bundle), unjar, and jar back up into one jar
     [echo]

BUILD SUCCESSFUL
Total time: 0 seconds
The system cannot find the batch label specified - end

C:\mchyzer\isc\dev\grouper\grouper-ws>
```

To do's (post 1.4.0)

1. add find subject service
2. make some params to test stuff... (junit to throw exceptions in the middle of tx?)
3. come up with formatter and code style and remove all warnings
4. add logging filter
5. fix javadoc warnings
6. look into axis 1.5 when it is out, see if error fixed, see if samples/wsdl changes, see about enums
7. add metadata service
8. add getGroups with batched groupLookup input
9. add batched privilege service, and add more url options to REST
10. add back in memberships service
11. filter getMember by privileges (find member?)
12. in rest add GET starting points with links to resources
13. improve auto-toString methods in resultMessage
14. look at acegi
15. add ip source filtering to grouper