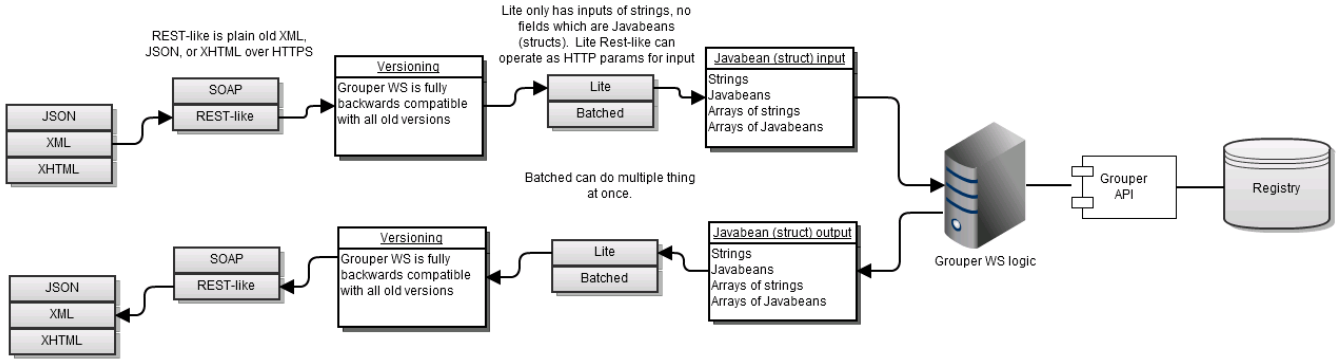
Grouper Web Services

Core web service API

These topics are discussed in the "Grouper Web Services" training series.

Grouper Web Services



- Add Member
- Add or remove grouper privileges
- Assign Attribute Def Actions
- Assign Attribute Definition Name Inheritance
- Assign Attributes
- Assign Attributes Batch
- Assign Permissions
- Attribute Definition Delete
- Attribute Definition Name Delete
- Attribute Definition Name Save
- Attribute Definition Save
- Delete Member
- External Subject Delete
- External Subject Save
- Find Attribute Definition Names
- Find Attribute Definitions
- Find External Subjects
- Find Groups
- Find Stems
- Get Attribute Assign Actions
- Get Attribute Assignments
- Get Audit Entries
- Get grouper privileges
- Get Groups
- Get Members
- Get Memberships
- Get Permission Assignments
- Get Subjects
- Group Delete
- Grouper always available web services and client
- Grouper Web Services Authentication
- Grouper Web Services FAQ
- Grouper Web Services for developers
- Grouper web services log
- Grouper Web Services Versioning
- Group Save
- Has Member
- Member change subject
- Message Acknowledge
- Message Receive
- Message Send
- Stem Delete
- Stem Save
- Web Services FAQ

Introduction

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

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 WSDL. Note, you cant get the 1.4 WSDL out of a 2.0 server, since it has backwards compatible changes from 1.5 and 1.6. You could get this 1.4 WSDL from SVN though...

Grouper client version	v1.4, v1.5, v1.6, v2.0 (server version) Endpoint	WSDL from server	WSDL from SVN
1.4	https://server.address/grouperWs/services/GrouperService	https://server.address/grouperWs/services/GrouperService?wsdl	1.4 WSDL
1.5	https://server.address/grouperWs/services/GrouperService	https://server.address/grouperWs/services/GrouperService?wsdl	1.5 WSDL
1.6	https://server.address/grouperWs/services/GrouperService	https://server.address/grouperWs/services/GrouperService?wsdl	1.6 WSDL, 1.6 WSDL in v2.0
2.0	https://server.address/grouperWs/services/GrouperService_v2_0	https://server.address/grouperWs/services/GrouperService_v2_0?wsdl	2.0 WSDL

1. a. Note, if your servlet is not grouperWs (e.g. grouper-ws) then adjust accordingly.
    b. There is a sample Java client with sample calls
2. 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. For example, in the URL you can set the content type you want back:

```
/grouper-ws/servicesRest/xml/v2_1_000/groups/aStem%3AaGroup/members/10021368
/grouper-ws/servicesRest/json/v2_1_000/groups/aStem%3AaGroup/members/10021368
/grouper-ws/servicesRest/xhtml/v2_1_000/groups/aStem%3AaGroup/members/10021368
```

1. a. Or you can set the content type of the request and it will use that for the response
    b. There are many samples
2. Understand versioning

https://spaces.at.internet2.edu/confluence/download/attachments/3014660/webServicePresentation.pptPresentation about Grouper web services

.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. Note if you send the same version in the request, you will never get a response with a different structure. Grouper WS are backwards compatible.
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)...
4. Note that Grouper WS can be setup with multiple instances. If you have database replication (even readonly), then you can setup Grouper WS application servers is multiple data centers, and you can have the Grouper Client can failover among the nodes if there are errors or timeouts.

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)
- Get Members: 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
  - 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 (many combinations of input are acceptable)
  - Can view all privileges for the subject, group, stem, specific privilege and combinations thereof
- 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)
- findAttributeDefNames
  - Can query for attributeDefNames/permissionNames based on name, uuid, inheritance, or a substring query
  - Can sort/page
- 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)
- assignAttributes
  - Assign or remove attributes (new attribute framework) from groups, stems, memberships, members, assignments, etc
  - Accepts batches (non-Lite)
- assignAttributesBatch
  - Assign or remove attributes (new attribute framework) from groups, stems, memberships, members, assignments, etc in a batch which allows more options for batch assignments than the non-lite "assignAttributes"
  - There is no Lite option for assignAttributesBatch since you can use the Lite version of assignAttributes
- getAttributeAssignments
  - Retrieves attribute assignments (new attribute framework) from groups, stems, memberships, members, assignments, etc
  - Batch or Lite
- assignPermissions
  - Assign or remove permissions from roles or individual subjects (in the context of a role)
  - Batch or Lite
- getPermissionAssignments
  - Retrieves permission assignments from roles or individual subjects (in the context of a role)
  - Batch or Lite
- Get Subjects
  - Lookup or search for subjects and attributes
  - Batch or Lite
- attributeDefNameSave
  - Create / update an attributeDefName
  - Accepts batches of attributeDefNames (non-Lite)
- attributeDefNameDelete
  - Delete an attributeDefName
  - Accepts batches of attributeDefNames lookups (non-Lite)
- getAuditEntries

- Get audit entries
- Batch or 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)
- **Authentication**
  - Let container or web server handle
    - PKI
    - http-simple-auth
    - Source IP address filtering (TODO)
  - Custom authenticator
  - WS-Security
    - PKI
    - Kerberos
  - LDAP authn
  - 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
- **JSONP is supported**, see this jira

Quick start

Note the WS is included in the Grouper Installer. Checkout the appropriate projects under grouper-ws.

Build Script

Note the WS is included in the Grouper Installer. (it will build and configure it). 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>
```

Subject attributes

1. The default attribute names (comma separated) sent back for each request are specified in grouper-ws.properties under the key:

ws.subject.result.attribute.names

2. If the caller sets T to retrieve subject detail, then the attributes will be appended to that list in grouper-ws.properties key:

ws.subject.result.detail.attribute.names

3. If the caller specifies subjectAttributeNames in the request (comma separated), then those will be appended to the list (independent of the detail attributes).

So there are central settings, and caller settings that you need to design for and specify...

Note if subjectId and subjectIdentifier are filled in with the same value, it will find by subject id or identifier.

Logging requests and responses

You can do this via the client or a proxy. If you must do this via the server, there is an experimental way to do this in v2.1.1+. You should not do this in prod, only in a testing environment.

Add a new servlet filter mapping in the web.xml

```
<filter-mapping>
    <filter-name>Grouper logging filter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

Set the filter logger to log at debug level

```
log4j.logger.edu.internet2.middleware.grouper.ws.j2ee.ServletFilterLogger = DEBUG
```

You might want to log to a dedicated file instead of putting in the grouper log...

```
log4j.appender.grouper_ws                           = org.apache.log4j.DailyRollingFileAppender
log4j.appender.grouper_ws.File                      = ${grouper.home}logs/grouper_ws.log
log4j.appender.grouper_ws.DatePattern               = '.'yyyy-MM-dd
log4j.appender.grouper_ws.MaxBackupIndex            = 30
log4j.appender.grouper_ws.layout                    = org.apache.log4j.PatternLayout
log4j.appender.grouper_ws.layout.ConversionPattern  = %d{ISO8601}: %m%n



# daemon log
log4j.logger.edu.internet2.middleware.grouper.ws.j2ee.ServletFilterLogger = DEBUG, grouper_ws
log4j.additivity.edu.internet2.middleware.grouper.ws.j2ee.ServletFilterLogger = false
```

You will see log entries like this

```
2012-05-03 09:13:18,575: [http-8088-1] DEBUG ServletFilterLogger.logStuff(98) -  - IP: 127.0.0.1, url:
/grouperWs/servicesRest/v2_1_001/groups/aStem%3AaGroup/members, queryString: null, method: PUT, content-type:
text/x-json; charset=UTF-8
request params:
request body: {"WsRestAddMemberRequest":{"actAsSubjectLookup":{"subjectId":"GrouperSystem"},"
replaceAllExisting":"F","subjectLookups":[{"subjectId":"10021368"},{"subjectId":"10039438"}]}}
respone headers: (note, not all headers captured, and not in this order)
X-Grouper-resultCode: SUCCESS
X-Grouper-success: T
X-Grouper-resultCode2: NONE
HTTP/1.1 201
Content-Type: text/x-json; charset=UTF-8
response: {"WsAddMemberResults":{"responseMetadata":{"millis":"237","serverVersion":"2.1.1"},"resultMetadata":
{"resultCode":"SUCCESS","resultMessage":"Success for: clientVersion: 2.1.1, wsGroupLookup: WsGroupLookup
[pitGroups=[],groupName=aStem:aGroup], subjectLookups: Array size: 2: [0]: WsSubjectLookup[subjectId=10021368]\n
[1]: WsSubjectLookup[subjectId=10039438]\n\n, replaceAllExisting: false, actAsSubject: WsSubjectLookup
[subjectId=GrouperSystem], fieldName: null, txType: NONE, includeGroupDetail: false, includeSubjectDetail:
false, subjectAttributeNames: null\n, params: null\n, disabledDate: null, enabledDate: null","success":"T"},"
results":[{"resultMetadata":{"resultCode":"SUCCESS_ALREADY_EXISTED","success":"T"},"wsSubject":{"id":"
10021368","name":"10021368","resultCode":"SUCCESS","sourceId":"jdbc","success":"T"}},{"resultMetadata":
{"resultCode":"SUCCESS_ALREADY_EXISTED","success":"T"},"wsSubject":{"id":"10039438","name":"10039438","
resultCode":"SUCCESS","sourceId":"jdbc","success":"T"}}],"wsGroupAssigned":{"description":"a group
description","displayExtension":"a group","displayName":"a stem:a group","extension":"aGroup","name":"aStem:
aGroup","typeOfGroup":"group","uuid":"d9094e4a7c6e4f399d7e1489c875b9f0"}}}
```

At some point we can make it more granular which requests get logged and give an option to format the request/response (indent, etc)

Fields and permissions

If you want to check to see if a subject as a group permission, or to get a list of people with a certain permissions on a group, use hasMember or getMembers, and pass the name of the field (note this list depends on your configuration):

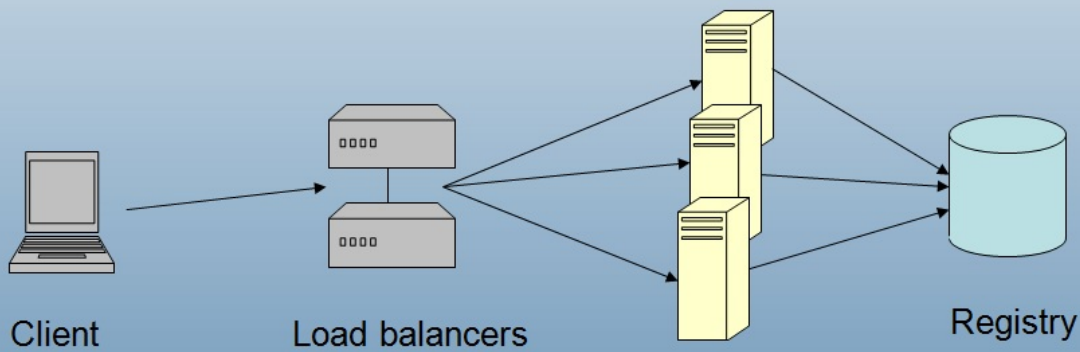
select name from grouper_fields where type != 'naming';

admins
description
displayExtension
displayName
extension
members
name
optins
optouts
readers
requireActiveEmployee
requireAlsoInGroups
updaters
viewers

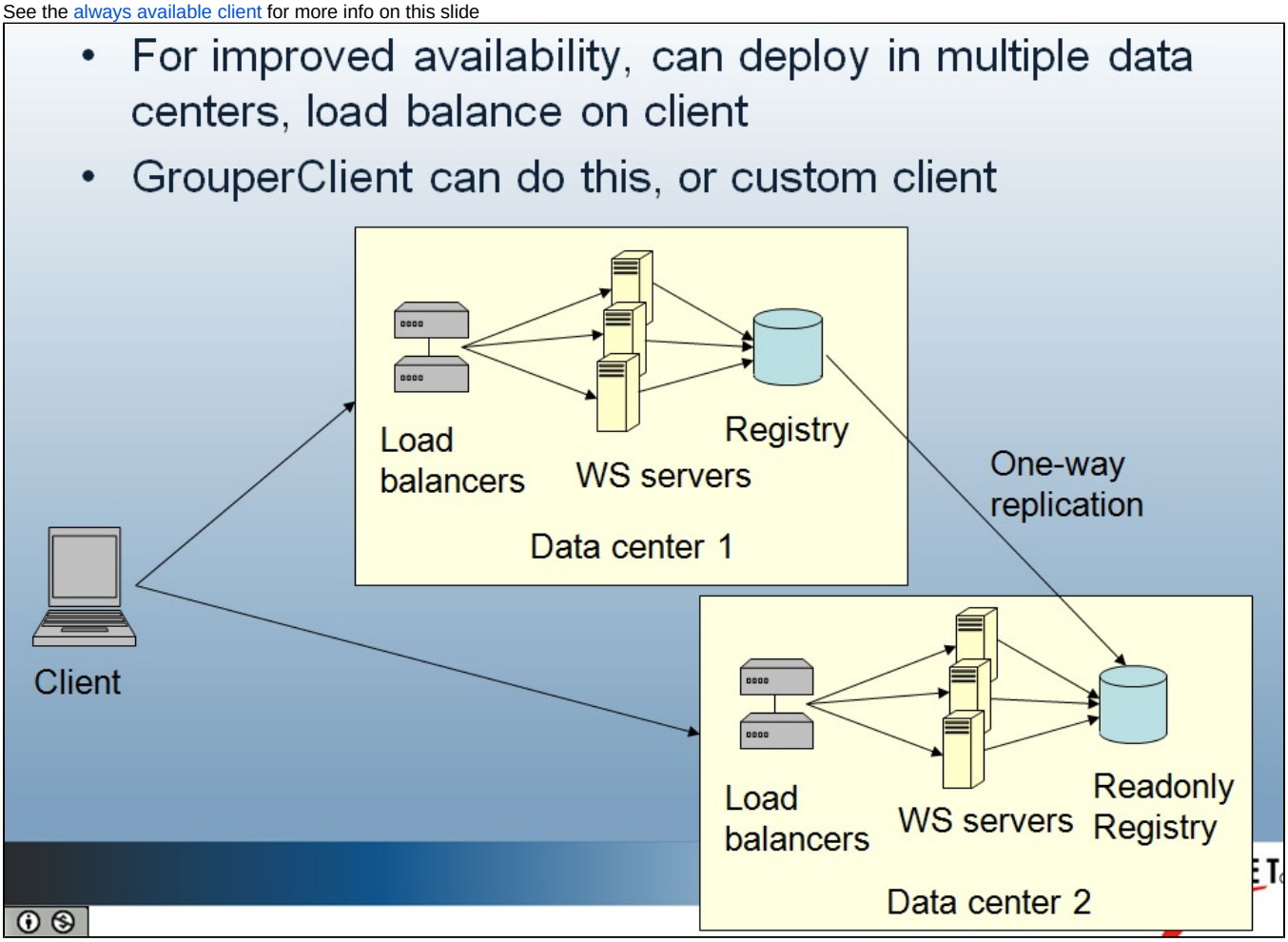
High availability

High availability

- Can have multiple app servers connected to one registry
- Might want session persistence by source IP address
- There are many ways to do this, here are two



See the always available client for more info on this slide



Children pages

- Grouper Web Services for developers
- Grouper web services log
- Grouper Web Services Versioning
- Group Save
- Has Member
- Member change subject
- Message Acknowledge
- Message Receive
- Message Send
- Stem Delete
- Stem Save
- Web Services FAQ

To do's (post 1.6.0)

1. add logging filter
2. fix javadoc warnings
3. look into axis2 1.5, see if error fixed, see if samples/wsdl changes
4. add move subject service
5. improve auto-toString methods in resultMessage
6. look at acegi
7. add ip source filtering to grouper

See Also

Always Available Web Services

Grouper Failover Client

Grouper Diagnostics