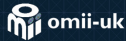


OGSA-DAI practical – developing workflows and clients

Mike Jackson
OGSA-DAI project team, The University of Edinburgh
Amy Krause
ADMIRE project team, The University of Edinburgh

GridKa School 2008

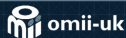


Web: www.omii.ac.uk

Email: info@omii.ac.uk

Using web services

- OGSA-DAI is accessed via web services
- Clients interact with web services via SOAP over HTTP
 - XML document exchange
 - Deduce service interface from service WSDL description
 - Construct SOAP request to invoke operation
 - Parse SOAP response from service
- But this is too low level!

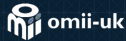


Web: www.omii.ac.uk

Email: info@omii.ac.uk

OGSA-DAI client toolkit

- Client-side Java abstractions of
 - Activities
 - Workflows
 - Resources
 - Services
- Get client-side proxies for OGSA-DAI resources and services
- Submit workflows using these proxies
- Client toolkit
 - Manages client-server communications
 - Parsing of request status into useful objects
- Developer can focus on writing applications

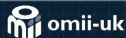


Web: www.omii.ac.uk

Email: info@omii.ac.uk

A typical client

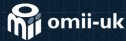
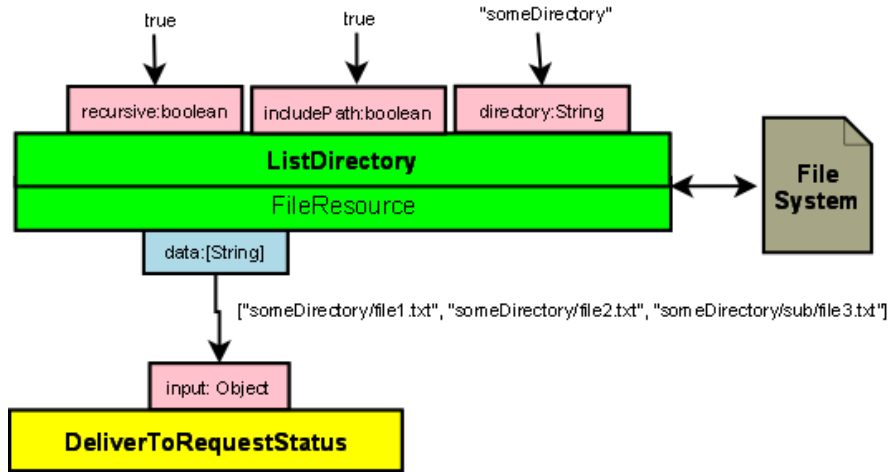
- Use client toolkit to contact server
- Get proxy for data request execution resource
- Build workflow using client toolkit activity and workflow objects
- Pass workflow to DRER proxy
- Get request status
- Get data from request status and client toolkit activities



Web: www.omii.ac.uk

Email: info@omii.ac.uk

A simple example – list files in a file system



Web: www.omii.ac.uk

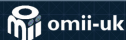
Email: info@omii.ac.uk

Contact the server and get a DRER proxy

```
URL serverURL =
    new URL("http://gks-1-121.fzk.de:21000/dai/services/");
String drerID = "DataRequestExecutionResource";

// Get server proxy.
ServerProxy server = new ServerProxy();
server.setDefaultBaseServicesURL(url);

// Get DRER proxy from server proxy.
DataRequestExecutionResource drer =
    server.getDataRequestExecutionResource(drerID)
```

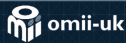


Web: www.omii.ac.uk

Email: info@omii.ac.uk

Create client toolkit activity proxies

```
ListDirectory listDir = new ListDirectory();  
DeliverToRequestStatus deliverToRequestStatus =  
    new DeliverToRequestStatus();
```



Web: www.omii.ac.uk

Email: info@omii.ac.uk

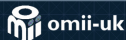
Set activity inputs and target resources

- ListDirectory takes a target resource is given input literals

```
ResourceID dataResourceID =  
    new DataResourceID("FileResource");  
  
listDir.setResourceID(dataResourceID);  
listDir.addIncludePath(true);  
listDir.addRecursive(true);  
listDir.addDirectory("someDirectory");
```

- DeliverToRequestStatus has its input connected to ListDirectory's output

```
deliverToRequestStatus.connectInput(listDir.getDataOutput());
```



Web: www.omii.ac.uk

Email: info@omii.ac.uk

Construct the workflow

- Create a workflow of the appropriate type
- Add the activities – remember to add them all!
- That's it!

```
PipelineWorkflow pipeline = new PipelineWorkflow();  
pipeline.add(listDir);  
pipeline.add(deliverToRequestStatus);
```



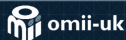
Web: www.omii.ac.uk

Email: info@omii.ac.uk

Submit the workflow to the DRER proxy

```
RequestResource requestResource = null;  
try  
{  
    requestResource =  
        drer.execute(pipeline, RequestExecutionType.SYNCHRONOUS);  
}  
catch (Throwable e)  
{  
    System.out.println(e.getMessage());  
    System.exit(1);  
}
```

- RequestResource is a proxy for the server request resource that holds the workflow
- This provides the request status



Web: www.omii.ac.uk

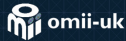
Email: info@omii.ac.uk

Get request status and data

```
RequestStatus status = requestResource.getRequestStatus();

// Is there more data from this activity?
while (listDir.hasNextData())
{
    // ListDirectory outputs a list so get an iterator over
    // the list.
    DataIterator iterator = listDir.nextData();
    while (iterator.hasNext())
    {
        // Get each list element - file name - in turn.
        System.out.println(iterator.next());
    }
}
```

- Data is provided as useful objects e.g:
 - ListDirectory outputs lists of file names – client toolkit provides an iterator over Strings.
 - TupleToWebRowSet outputs a list of XML Nodes – client toolkit provides a JDBC ResultSet
 - SQLUpdate outputs an Integer – client toolkit provides an Integer.

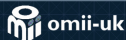


Web: www.omii.ac.uk

Email: info@omii.ac.uk

Practical

- Please form into pairs
- Introduce yourself to your partner if they're a stranger 😊
- Visit <http://www.ogsadai.org.uk/courses/gridka08/practical.html>
- Amy and Mike are here to help so just shout!



Web: www.omii.ac.uk

Email: info@omii.ac.uk

Solve the puzzle

- A relational database contains information about image files
 - Coordinates columns – x, y, z
 - Filename column – filename
- A file system contains the actual image files
- We want a workflow that
 - Queries the database
 - Splices the results of the query using column names provided by the client
 - Returns the coordinates in the request status
 - Gets the files from the file system
 - Delivers the files to an FTP server
- Form teams of four, introduce yourselves and do it!
- Visit <http://www.ogsadai.org.uk/courses/gridka08/practical.html>

Functionality needed

- Query the database
- Create tuples from information provided in a user-friendly format (1st hint)
- Separate coordinates from filenames (2nd hint)
- Deliver coordinates to the request status
- Read files from the file system
- Deliver files to FTP server (3rd hint)

1st hint – converting from a user-friendly format

- A CSV is a set of comma-separated values
 - e.g. “x”, “y”, “z”
 - More user-friendly than tuples
- CSVToTuple

2nd hint – separating coordinates from filenames

- TupleProjection projects a tuple onto a single set of columns
 - e.g. take fields x, y, z and filename and project to x, y, z
- TupleMultiProject projects a tuple onto multiple sets of columns
 - e.g. take fields x, y, z and filename and project to fields x, y, z and to field filename

3rd hint – how to repeat the host name for DeliverToFTP

- The clue is in the hint...
- ControlledRepeat activity