Outline

• Introduction to the Client Toolkit
• Accessing and Interacting with Data Services
• Activities, Requests and Sessions
• Delivery Methods
• A Data Integration Example
The Client Toolkit

- Designed to help application developers
  - Simple to use APIs
  - Conventional data access methods
  - Offers some protection from changes to underlying services
- Provides a common abstraction level for all flavours
  - Service platform is hidden
  - XML and SOAP requests hidden
  - WSI and WSRF version transparency
- Included in the latest OGSA-DAI releases
  - OGSA-DAI WSRF 2.1 (GT4)
  - OGSA-DAI WSI 2.1 (Axis and OMII)
  - OGSI (GT3) version was deprecated with release 7

Interaction with a Data Service

- Client sends a request to a data service
- A request contains a set of activities
Interaction with a Data Service

- The data service processes the request
- Returns a response document containing a result for each activity

Service Fetcher

- Provides the interface through which a client can interact with a Data Service and Data Service Resource

```java
DataService
getDataService(URL, resourceID)
perform(request)
```
Data Service Operations

• Data Service

  - `getResources()`
  - `getProperty(QName)`

• Data Service with Resource

  - `perform(Request)`
  - `perform(RequestComponent)`
  - `openSession()`
  - `closeSession()`
  - `putBlock(data)`
  - `getBlock()`
  - `getResources()`

Activities and Requests

• A request contains a set of activities

• Activities can be performed in sequence or in parallel

• An activity dictates an action to be performed
  - Query a data resource
  - Transform data
  - Deliver results

• Data can flow between activities
Client-Side Activities

- A client-side Activity class exists for each type of Activity available to a data service
- Allows easy configuration and access to result data

```java
SQLQuery query = new SQLQuery(
    "select * from GeneTable where ID<10000");
```

- Client-side activities are independent of the service platform
  - One API for interacting with both OGSA-DAI WSI and WSRF services
- Properties are also accessible in a service-independent way
  - Although published properties may differ between data services

Control-flow

- Sequence
  - Children processed one after another
- Flow
  - Children processed concurrently
- Complex structures can be formed by nesting
Sessions

• When a request is processed, it is *joined* to a session
• Sessions allow state to be stored across multiple requests

• Client Toolkit allows session requirements to be specified

Delivery Methods

- GridFTP server
- Local Filesystem
- Web Server
- FTP server
- Data Service
- DeliverTo/FromGFTP
- DeliverTo/FromURL
- DeliverTo/FromFile
- DeliverFromURL
Delivering data to another GDS

- The DataTransport port type allows to transfer data from one data service to another.
  - Supports any combination of WSI and WSRF services
- Client Toolkit provides easy access to these operations
  - via the DataService interface
  - DTInputStream and DTOOutputStream activities

Data Integration Scenario

MySQL database \(\leftrightarrow\) DS2 (GT4) \(\rightarrow\) DS3 (Axis) \(\leftrightarrow\) Oracle database

DB2 database \(\leftrightarrow\) DS1 (OMII) \(\rightarrow\) Client

- select + outputStream
- deliverFromDT + bulkload
- join query
- select + outputStream
1: Fetch services

- First fetch all three data services

```java
ServiceFetcher fetcher =
    GenericServiceFetcher.getInstance();

DataService ds1 = fetcher.getDataService(
    "http://www.epcc.ed.ac.uk/DS1",
    "DB2Resource");
DataService ds2 = fetcher.getDataService(
    "http://www.nesc.ac.uk/DS2",
    "MySQLResource");
DataService ds3 = fetcher.getDataService(
    "http://www.esnw.ac.uk/DS3",
    "OracleResource");
```

2: Send queries

- Send query requests to ds1 and ds2
- 2a: Create activities:
  - SQLQuery ➔ WebRowSet ➔ DTOutputStream

```java
SQLQuery query1 = new SQLQuery(  
    "select * from GeneTable where ID<10000";  

WebRowSet rowset1 = new WebRowSet(  
    query1.getOutput());

DTOOutputStream output1 = new DTOOutputStream(  
    rowset1.getOutput());
```
2: Send queries cont.

- 2b: Create new request and add activities

```java
ActivityRequest req1 =
    new ActivityRequest();
req1.addActivity(query1);
req1.addActivity(rowset1);
req1.addActivity(output1);
```

- 2c: Perform request!

```java
ds1.perform(req1);
```

- Repeat steps for ds2

Data Integration Scenario

MySQL database <-> DS2 <-> DS3 <-> Oracle database

DB2 database <-> DS1 <-> Client

select + outputStream

select + outputStream
3: Bulkload result data

- Pull query results to ds3 and bulkload into tables
- 3a: Create delivery and bulk load activities

```java
DeliverFromDT deliver1 = new DeliverFromDT();
SQLBulkLoad load1 = new SQLBulkLoad(
   deliver1.getOutput(), "table1");
DeliverFromDT deliver2 = new DeliverFromDT();
SQLBulkLoad load2 = new SQLBulkLoad(
   deliver2.getOutput(), "table2");
```

- 3b: Make the Data Transport connections

```java
delivery1.setDataTransportInput(output1.getDataTransport());
delivery1.setDataTransportInput(output1.getDataTransport());
```

3: Bulkload result data cont.

- 3c: Create request and add activities

```java
ActivityRequest req1 =
   new ActivityRequest();
req3.addActivity(delivery1);
req3.addActivity(load1);
req3.addActivity(delivery1);
req3.addActivity(load1);
```

- 3d: Perform the request!

```java
ds3.perform(req3);
```
4: Perform join query

- Perform a join query across the new table data
- Create activities, assemble and perform request

```java
SQLQuery join = new SQLQuery(
    "select * from table1, table2 where ...";
WebRowSet rowset = new WebRowSet(
    query.getOutputStream);

ActivityRequest req3 = new ActivityRequest();
req3.addActivity(join);
req3.addActivity(rowset);

ds3.perform(req3);`
Summary

• The Client Toolkit simplifies the development of clients that interact with OGSA-DAI data services
• Provides a common abstraction level for both supported service interfaces
  – OGSA-DAI WSI
  – OGSA-DAI WSRF
• Hides service interaction and XML from users
• Offers some protection from future changes to services