



# OGSA-DAI Client Toolkit

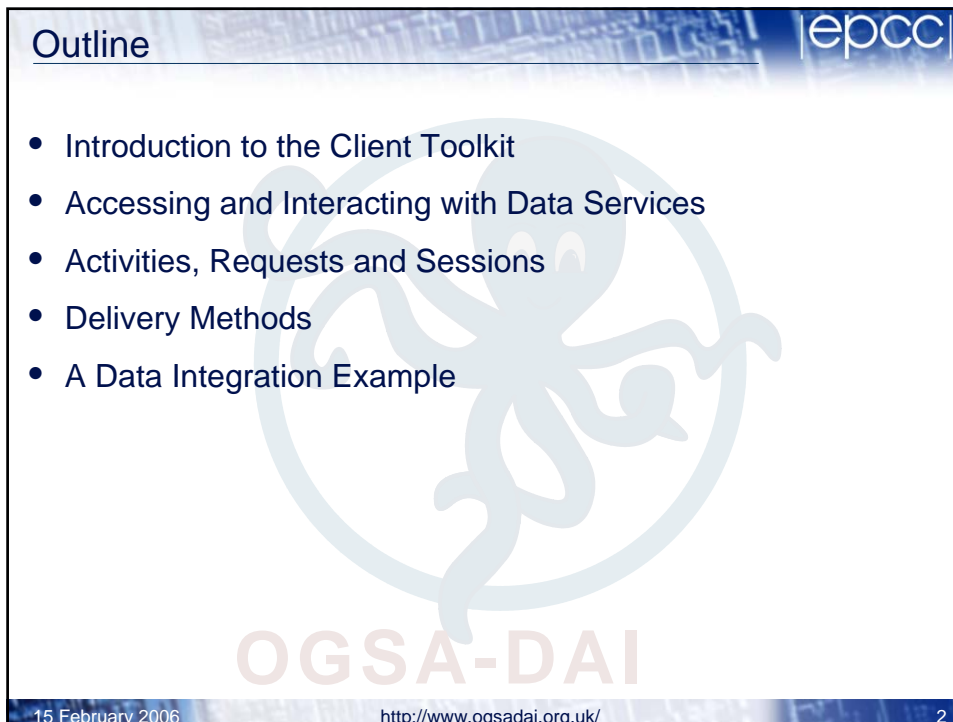
**Principles and Architectures for Structured  
Data Integration: OGSA-DAI as an example**

ISSGC06 (Ischia, Italy)  
17 July 2006

---


Amy Krause  
Applications Consultant, EPCC  
[a.krause@epcc.ed.ac.uk](mailto:a.krause@epcc.ed.ac.uk)

Tom Sugden  
Applications Consultant, EPCC  
[tom@epcc.ed.ac.uk](mailto:tom@epcc.ed.ac.uk)



## Outline

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



OGSA-DAI

15 February 2006 <http://www.ogsadai.org.uk/> 2

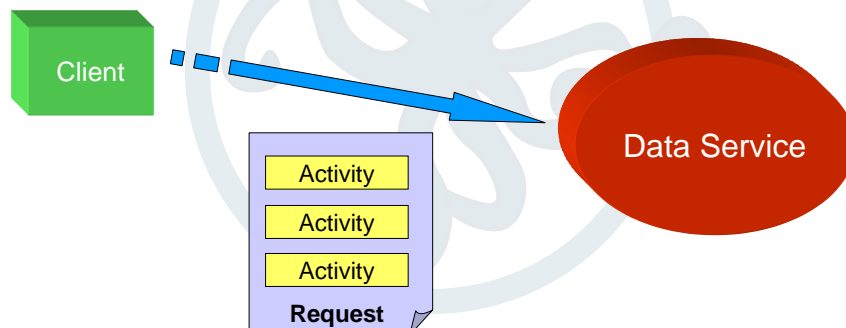
## The Client Toolkit

- 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)
  - OGSF (GT3) version was deprecated with release 7

OGSA-DAI

## Interaction with a Data Service

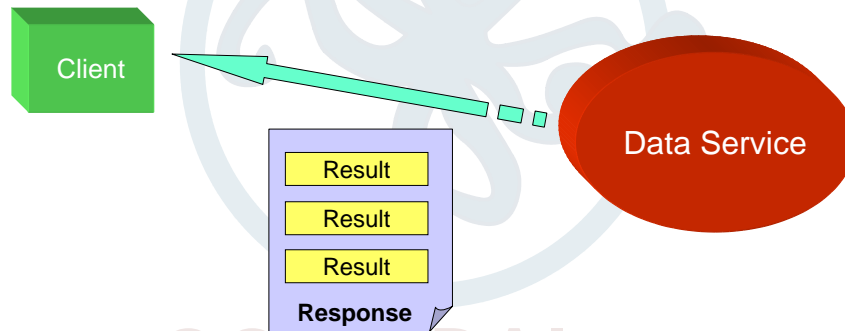
- Client sends a request to a data service
- A request contains a set of activities



OGSA-DAI

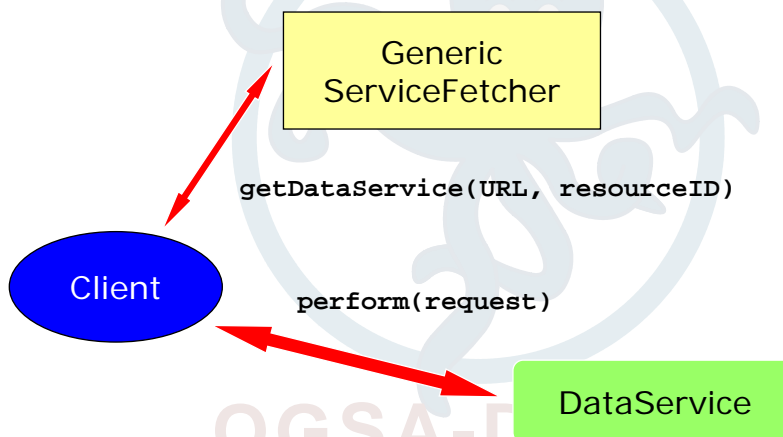
## Interaction with a Data Service

- 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



## Data Service Operations | epcc

- Data Service
  - getResources()
  - ...
  - getProperty(QName)
- Data Service with Resource
  - perform(Request)  
perform(RequestComponent)
  - openSession()  
closeSession()
  - putBlock(data)  
getBlock()

15 February 2006 <http://www.ogsadai.org.uk/> 7

## Activities and Requests | epcc

- 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

```

graph LR
    A[SQL Query Statement] -- "resultset data" --> B[Web RowSet]
    B -- "XML data" --> C[Deliver ToURL]
  
```

15 February 2006 <http://www.ogsadai.org.uk/> 8

## 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

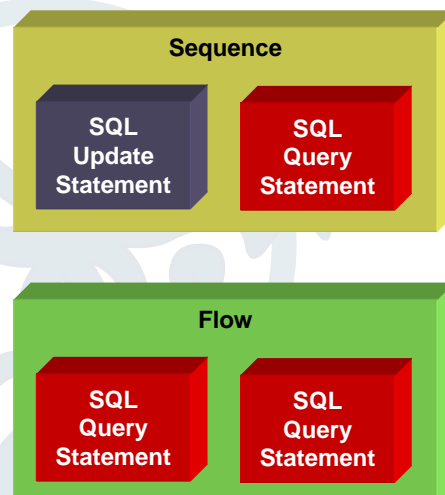
```
SQLQuery query = new SQLQuery(  
    "select * from GeneTable where ID<10000");
```

- 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

OGSA-DAI

## Control-flow

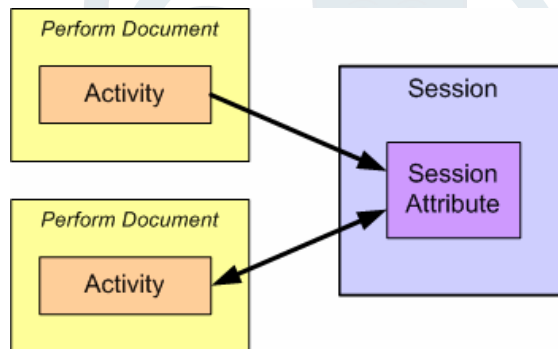
- Sequence
  - Children processed one after another
- Flow
  - Children processed concurrently
- Complex structures can be formed by nesting



OGSA-DAI

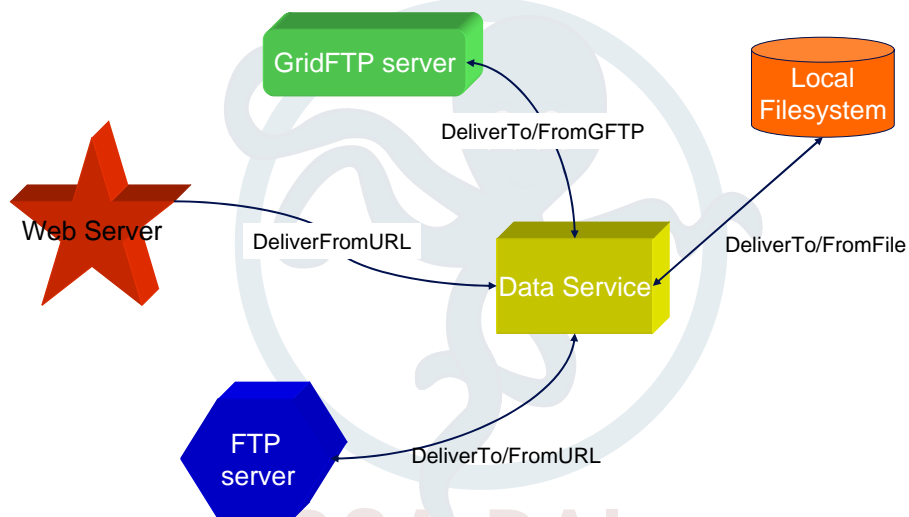
## Sessions

- 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



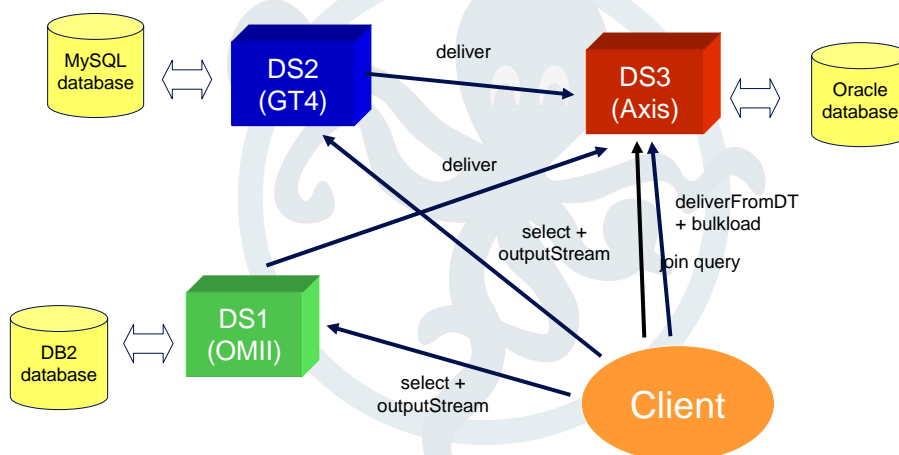
## Delivering data to another GDS

- 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 DTOutputStream activities



OGSA-DAI

## Data Integration Scenario



OGSA-DAI

## 1: Fetch services

- First fetch all three data services

```
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

```
SQLQuery query1 = new SQLQuery(  
    "select * from GeneTable where ID<10000");  
  
WebRowSet rowset1 = new WebRowSet(  
    query1.getOutput());  
  
DTOutputStream output1 = new DTOutputStream(  
    rowset1.getOutput());
```

## 2: Send queries cont.

- 2b: Create new request and add activities

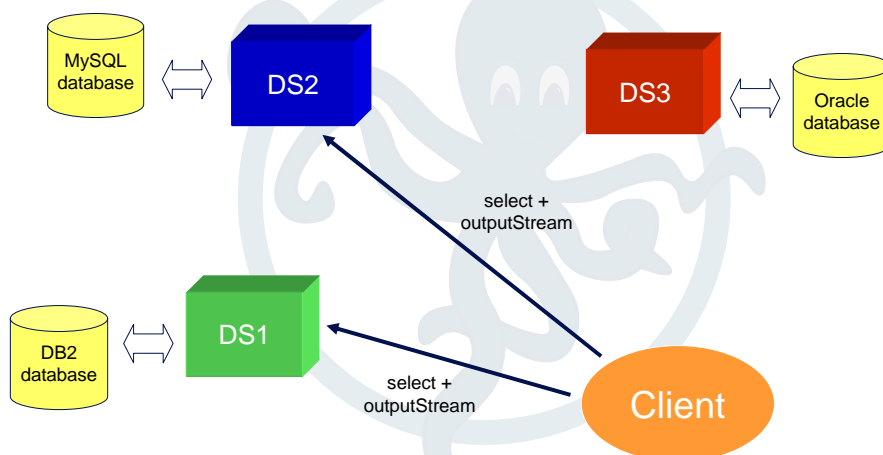
```
ActivityRequest req1 =  
    new ActivityRequest();  
req1.addActivity(query1);  
req1.addActivity(rowset1);  
req1.addActivity(output1);
```

- 2c: Perform request!

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

- Repeat steps for ds2

## Data Integration Scenario



### 3: Bulkload result data

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

```
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

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

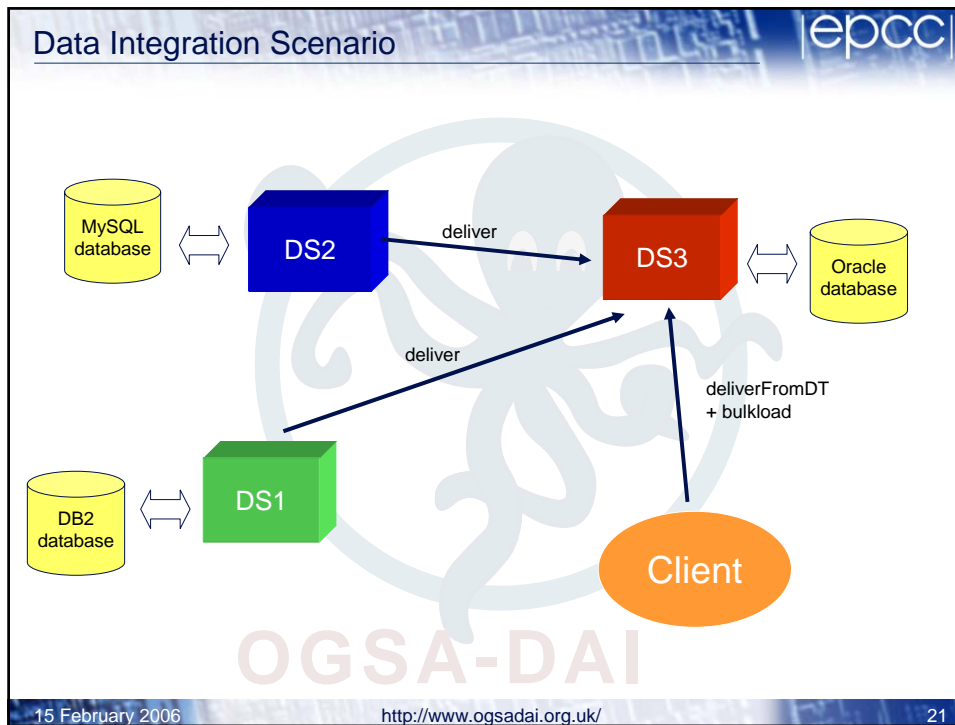
### 3: Bulkload result data cont.

- 3c: Create request and add activities

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

- 3d: Perform the request!

```
ds3.perform(req3);
```



4: Perform join query | epcc

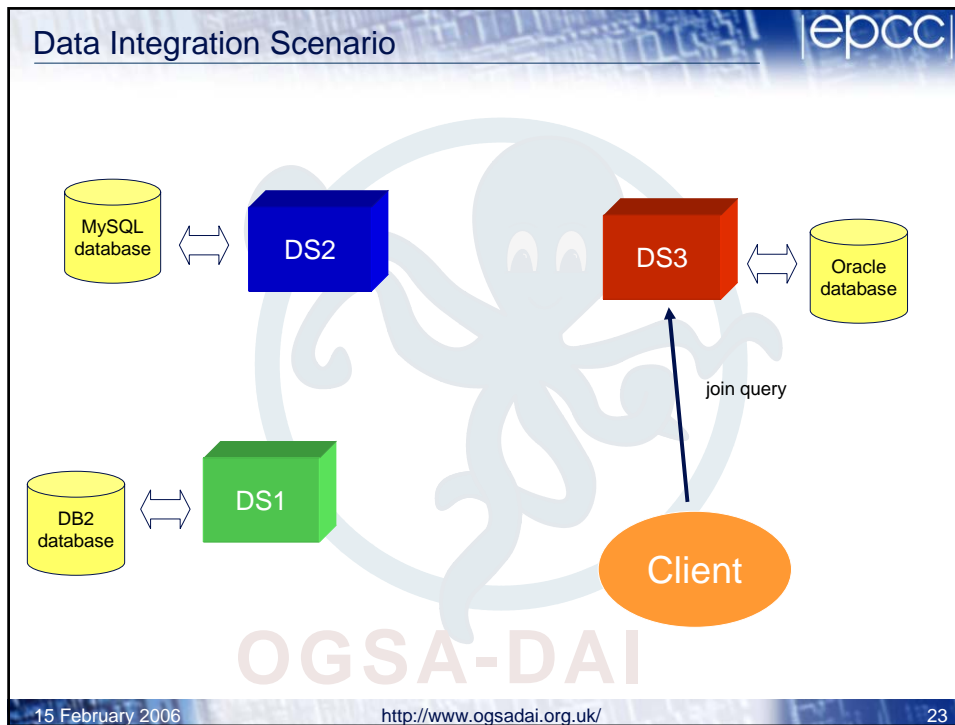
- Perform a join query across the new table data
- Create activities, assemble and perform request

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

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

ds3.perform(req3);
```

15 February 2006 <http://www.ogsadai.org.uk/> 22



- ### 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
- 15 February 2006 <http://www.ogsadai.org.uk/> 24