OGSA-DAI Tutorial: Service-based access to data on the Grid

Grid Computing 2006/2007

Femi Ajayi
National e-Science Centre
University of Glasgow

Overview

- Why data service?
- Data challenges
- What is OGSA-DAI?
- Introduction to VOTES and its challenges
- Current data issues on the Grid

Motivation

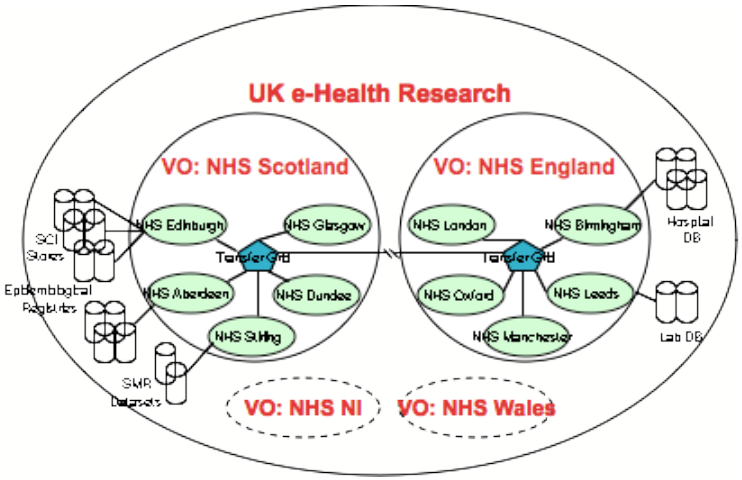
- Data on the Grid
  - Data surge
    - Growing in size, complexity and time
  - Cheaper storage
- Various data storages
  - Data resources
    - Relational databases
    - Xml files
    - Flat files
- Need ways to discover, access and integrate data
  - Unified interfaces?
- Empower e-Community through the Grid
  - Data as a services?

Data Challenges

- Data cannot be moved
  - Data stays where it is generated or stored
    - Hospitals, Research centres, Observatories, Bioinformatics institutes
  - Numerous data clusters
  - Move computation to the data (or extract data you need)
- Distributed collaborating organisations
  - Various expertise in data creation, analysis and simulations
  - Data ownership & provenance
- Diverse data collections
  - Collected in various ways and formats
  - Semantic heterogeneity

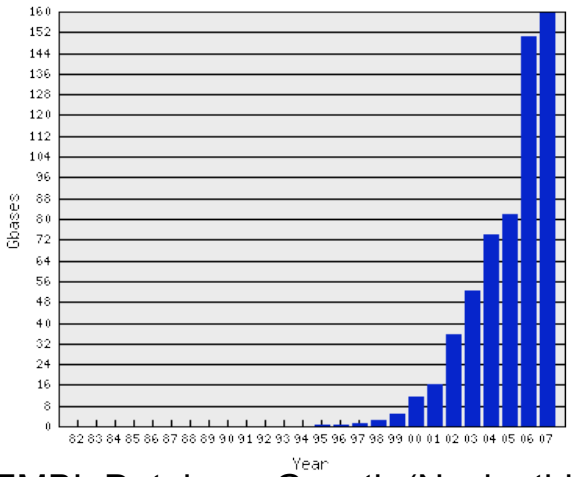
Examples

- Clinical Data: Clinical records, laboratory results, morbidity records, epidemiological studies, patient demographics, etc.
  - VOTES (Scotland): 19 SCI-Stores, SMR stores, GPASS, PACS with hundreds of thousands of clinical records to analyse (2005 Scotland Statistics: 1,036 GP Practices; 4,500 GPs; 5.3M Patients).
- Bioinformatics: Genomics data (microarray data, proteomic gel data, RNA, etc.)
  - BRIDGES: Integration of six geographically distributed research repositories to support cardiovascular functional genomics research.



e-Health Virtual Organisations

Graph created on 23/02/2007
http://www3.ebi.ac.uk/Services/DBStats/



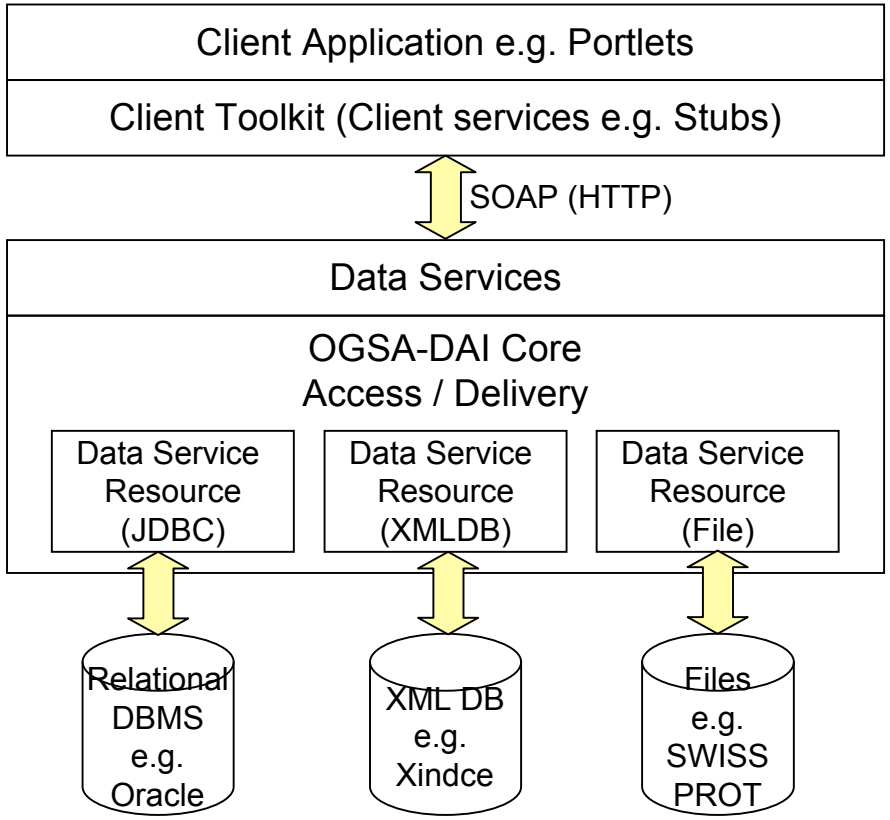
EMBL Database Growth (Nucleotide)

What is OGSA-DAI?

- To provide a service-based architecture for data access over the Grid
- To provide Grid services that offer data integration services to clients
- To specify a selection of common interfaces for different data access including relational and xml datasets.
- To seamlessly deliver data in various forms and for various purposes.
- To reduce cross-boundaries issues with regards to data access or delivery.

OGSA-DAI Architecture

- Client layer – interacts with presentation layer using WSRF/WSI standards
- Presentation layer – exposes data services as a web service
- Business logic layer – provides core OGSA-DAI functionalities, perform & response documents, data resource access, data transport, session management and property management
- Data layer – currently supports RDBMS (Oracle, SQL Server, MySQL, DB2, PostgreSQL), Xindce, and Files formats (OMIM, SWISSPROT, EMBL)



client layer

presentation layer

business logic layer

data layer

Interacting with Data Service Resources

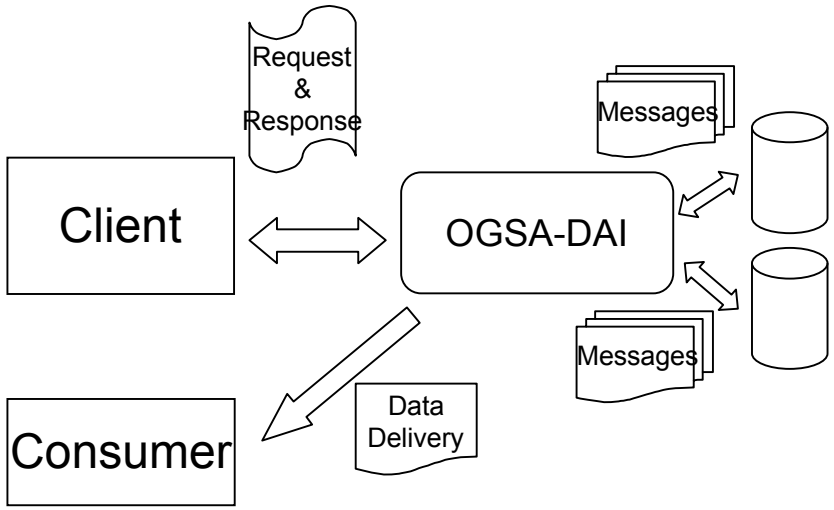
- Activities: actions to be performed
  - Data resource statement e.g. sqlQueryStatement, sqlUpdateStatement, sqlStoredProcedure, sqlResultsToXML
  - Data transformation e.g. xslTransform, gzipCompression
  - Data delivery to/from a consumer e.g. deliverFromURL, deliverToURL, deliverFromGFTP, deliverToGFTP
- Perform documents
  - Used by clients to define instruction sets for data service resources. Activities form instruction sets.
  - Data can flow from activity to activity
  - Better generated and processed using client toolkit API.
- Response documents
  - Result properties e.g. query type, cursor
  - Metadata e.g. columns definition, column properties
  - Data encapsulated in XML

Data Service Scenarios

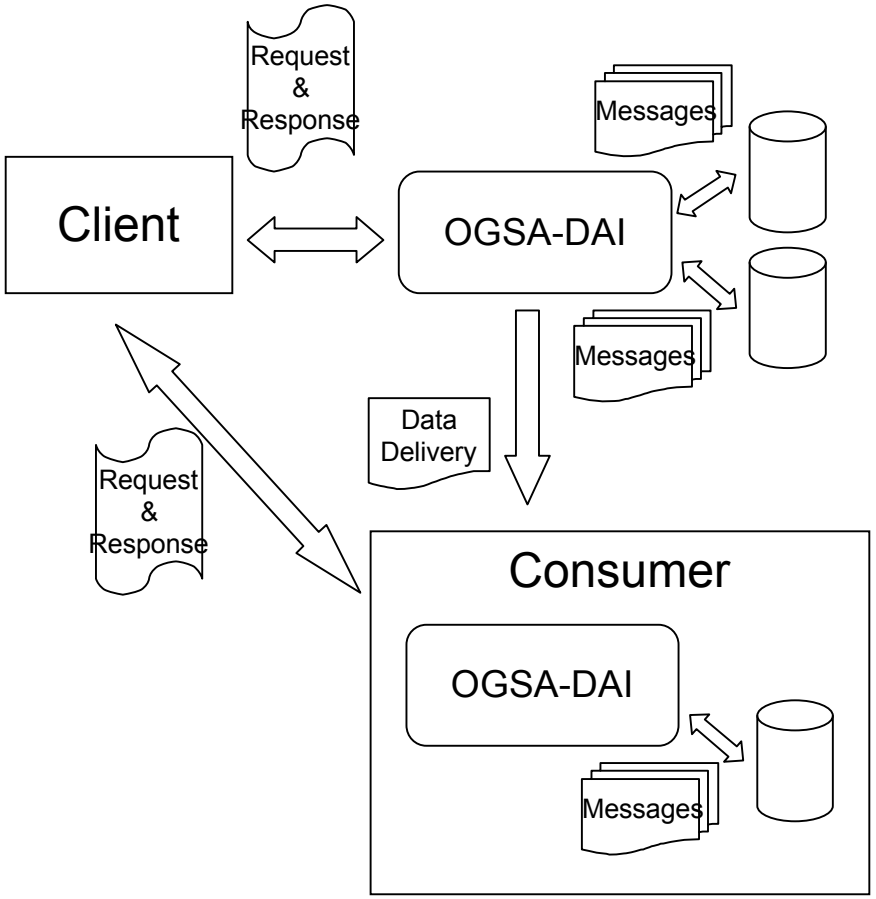
1. A client submits an SQL query that returns data
2. Extract data from one data resource and push to another. One request to access multiple data resources.
3. Multiple queries to multiple URLs in one request. To use separate activity chains.
4. Data transformations of rowsets.
5. Extract data from one FTP server to another or from one data resource to an FTP server.

OGSA-DAI Data Service Scenarios

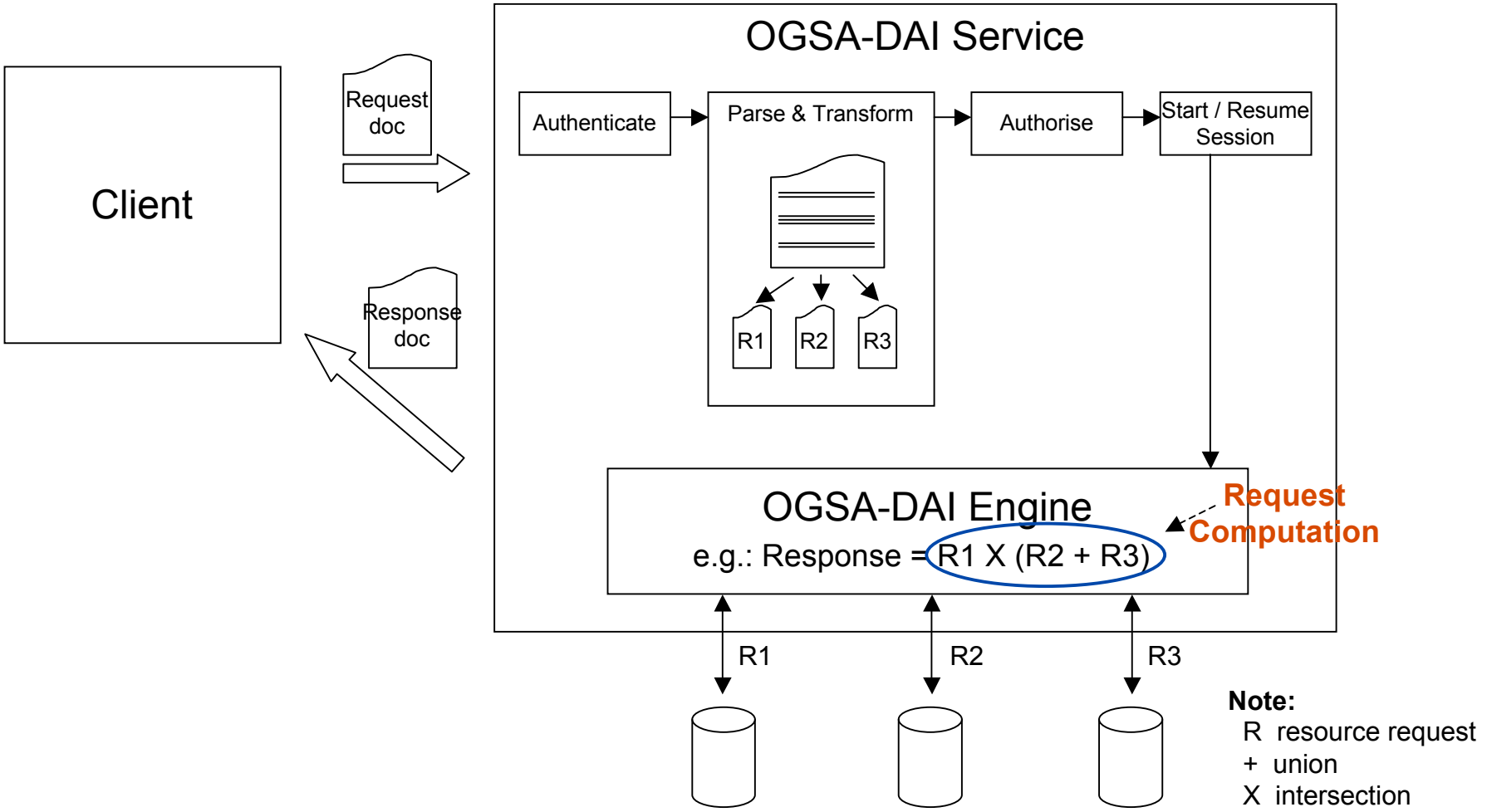
Scenario 1 & 2: Single data service, multiple data resources



Scenario 3: Multiple data services

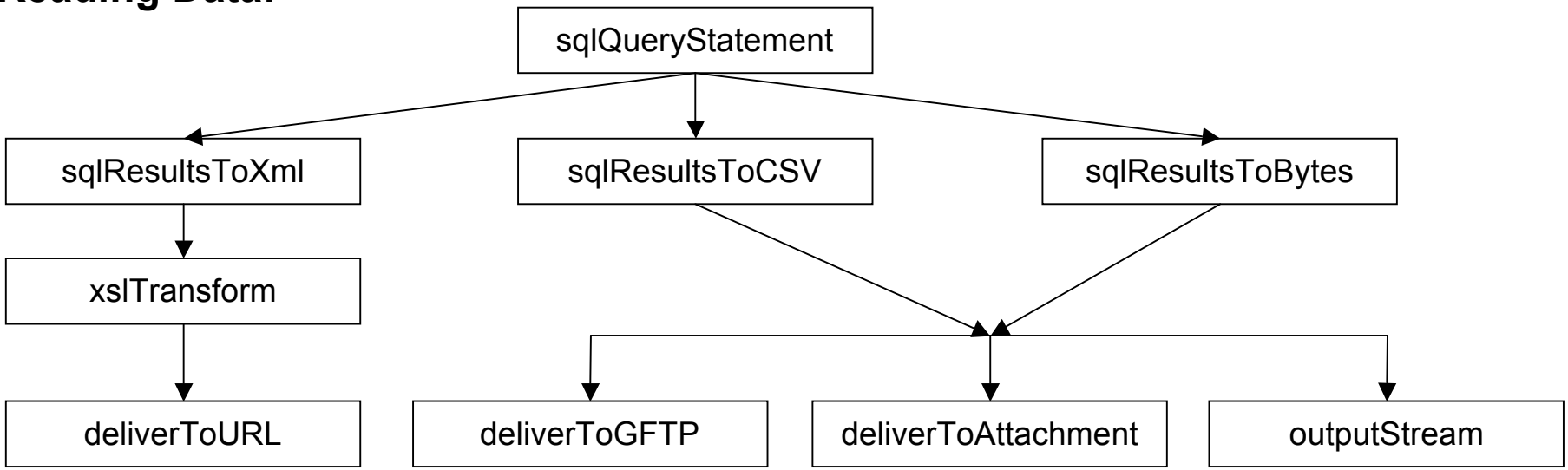


OGSA-DAI Computational Flow

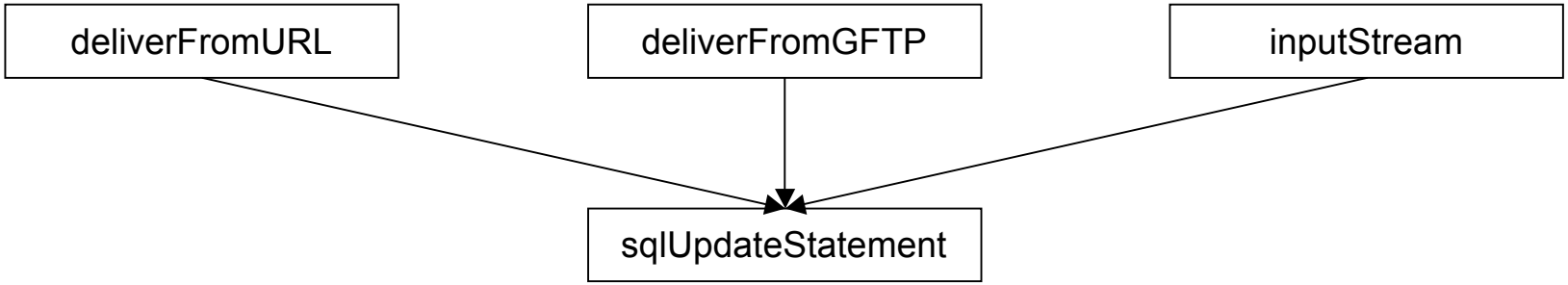


Activities

Reading Data:



Writing Data:



Perform Document

- Workflow
  - Sequence: Executes activities sequentially. Used when dependences exist between activities.
  - Flow: Executes activities concurrently (default). Used when no dependences exist between activities.

Statement activity

```
<sqlQueryStatement name="listPatient">
    <!-- value of first parameter -->
    <sqlParameter position="1" from="minValue"/>
    <!-- value of second parameter -->
    <sqlParameter position="2">69</sqlParameter>
    <expression>
                select name, postcode from patientmaster
where age &le; ? and id &ge; ?
    </expression>
    <resultStream name="queryResult"/>
</sqlQueryStatement>
```

Stream

Statement activity

```
<sqlResultsToXML name="results">
    <resultSet from="queryResult" />
    <webRowSet name="webrowset" />
</sqlResultsToXML>
```

Transform

Transform activity

```
<xslTransform name="listTransform">
  <inputXSLT>
    <xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
      <xsl:output method="text" indent="yes" />
      <xsl:template match="/">
        <xsl:value-of select=
"RowSet/data/row/col[1]"/>
      </xsl:template>
    </xsl:stylesheet>
  </inputXSLT>
  <inputXML from="queryResult"/>
  <output name="transformedResult"/>
</xslTransform>
```

Delivery activity

```
<deliverToURL name="ftpDelivery">
  <fromLocal from="transformedResult "/>
  <toURL>ftp://test.nesc.gla.ac.uk/file/file.ext</toURL>
</deliverToURL >
```

Delivery

Client toolkit APIs - could be used to generate perform documents and interpret response documents.

Response Document

- SELECT FamilyName, PostCode FROM PatientMaster WHERE Patientid LIKE '87%'

```
<?xml version="1.0" encoding="UTF-8"?>
<webRowSet xmlns="http://java.sun.com/xml/ns/jdbc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/jdbc
http://java.sun.com/xml/ns/jdbc/webrowset.xsd">
 <properties>
  …
</properties>
```

Metadata definition

```
<metadata>
 <column-count>2</column-count>
 <column-definition>
  <column-index>1</column-index>
  …
  <signed>false</signed>
  <column-display-size>35</column-display-size>
  <column-label>FamilyName</column-label>
  <column-name>FamilyName</column-name>
  <schema-name></schema-name>
  <column-precision>35</column-precision>
  <column-scale>0</column-scale>
  <table-name></table-name>
  <catalog-name></catalog-name>
  <column-type>12</column-type>
  <column-type-name>varchar</column-type-name>
  <catalog-name></catalog-name>
   <column-type>12</column-type
   <column-type-name>varchar</column-
type-name>
  </column-definition>
  <column-definition>
   …
  </column-definition>
 </metadata>
```

Metadata

```
 <data>
  <currentRow>
   ....
  <currentRow>
<columnValue>BAKER</columnValue>
   <columnValue>G 133YR</columnValue>
  </currentRow>
  <currentRow>
<columnValue>SILSTROM</columnValue>
   <columnValue>G 133YR</columnValue>
  </currentRow>
   ....
 </data>
</webRowSet>
```

Client Toolkit API

- Data services interaction
  - One API for both WSRF & WS-I services
  - Activities & Request (Queries, transformation & delivery)
  - Flow/sequence control
- Hidden service platform
- Data integration
- Extensibility

Using OGSA-DAI

- Installation
  - Download OGSA-DAI and then install
    - ant install -Ddai.container=/path/to/Web/services/container
- Exposing a Data Resource
  - Deploy an OGSA-DAI data service
    - ant deployService -Ddai.container= /usr/local/globus-4.0.1/ - \ Ddai.service.name=ogsadai/DataService
  - Deploy a data service resource
    - ant deployResource -Ddai.container=/usr/local/globus-4.0.1/ - \ Ddai.resource.file=data.service.resource.properties.postgresSqlServer
  - Add the data service resource to the data service
    - ant exposeResource -Ddai.container=/usr/local/globus-4.0.1/ - \ Ddai.service.name=ogsadai/DataService - Ddai.resource.id=SQLServerResource
- Check service is deployed correctly
  - List data service resources
    - ant listResourcesClient - Ddai.url=http://localhost:8080/wsrf/services/ogsadai/DataService

Using OGSA-DAI

- Data Service Resource

SAMPLE: *data.service.resource.properties.postgreSqlServer*
dai.resource.id= postgreSQLResource
dai.data.resource.type=[Relational | XML | Files]
dai.product.name= ***PostgreSQL***
dai.product.vendor= ***Postgres***
dai.product.version= ***7.3.10***
dai.data.resource.uri= **jdbc:postgresql://localhost:5432/testDb**
dai.driver.class= **org.postgresql.Driver** *(optional only if dai.data.resource.type=Files)*
dai.credential= [NO Certificate Provided] or leave blank
dai.user.name= ***testuser***
dai.password= ***testpwd***

- End-to-end client

```
ant dataServiceClient -
Ddai.url=http://localhost:8080/wsrf/services/ogsadai/MyDataService \
 -Ddai.resource.id=SQLServerResource -Ddai.action=examples/selectPatients.xml
```

- Client toolkit example
  - Java class e.g. TutotrialSampleQuery.java

Virtual Organisation for Trials and Epidemiological Studies (VOTES)

Objectives:

- To develop Grid infrastructure framework that would support both clinical trials and epidemiological studies.
  - Patient recruitment
  - Data collection
  - Study management
- A coordinated use and sharing of dynamic and distributed resources by a set of individuals or institutions or domains.
- Collaboration to achieve shared goals.
- No centralised point of control - diverse administrative domains
- Single sign-on, delegation, various local security integration

  - Not about submitting or running jobs
  - About accessing & integrating data
  - About distributed queries – accessing federated (or distributed) resources

OGSA-DAI Implementation in VOTES

- OGSA-DAI as a Globus Toolkit (GT4) service
- Extensive use of client toolkit API for data service interactions.
- Exposes data services and numerous data service resources
  - SQL Server 2000, Oracle 10g, PostgreSQL, MySQL
- Activities include: sqlQueryStatement, sqlResultsToXml, deliverToURL
  - Requests are mostly read only queries
  - Client application such as portlets parses the received response documents
- Executes distributed queries using a data service resource acting as a driving resource
  - Alternatively could use DQP (Distribute Query Processor)
- Data access security? A separate service.

Current Issues/Challenges

- Transactions implementation
- Data aggregation from disparate models
- Conversion between metadata
- Changing schemas
- Auto discovery of data service
  - Queries target specified resources
- Fine grain security model
- Provenance information
- Statistical disclosure
- Data access != Data integration

Resources

- OGSA-DAI Home page:
  - http://www.ogsadai.org.uk
- OGSA-DAI WSRF 2.2 User Guide:
  - http://www.ogsadai.org.uk/documentation/ogsadai-wsrf-2.2/doc/
- References:
  - M. Antonioletti, et. al. The Design and Implementation of Grid Database Services in OGSA-DAI. Concurrency and Computation: Practice and Experience, Volume 17, Issue 2-4, Pages 357-376, February 2005.
  - Sheth AP, Larson JA. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. ACM Comput. Surv. 1990; 22:183-236.

- Email: o.ajayi@nesc.gla.ac.uk