OGSA-DAI Lectures Part 2

Tom Sugden, EPCC
tom@epcc.ed.ac.uk

2nd International Summer School
on Grid Computing, Vico Equense, Italy

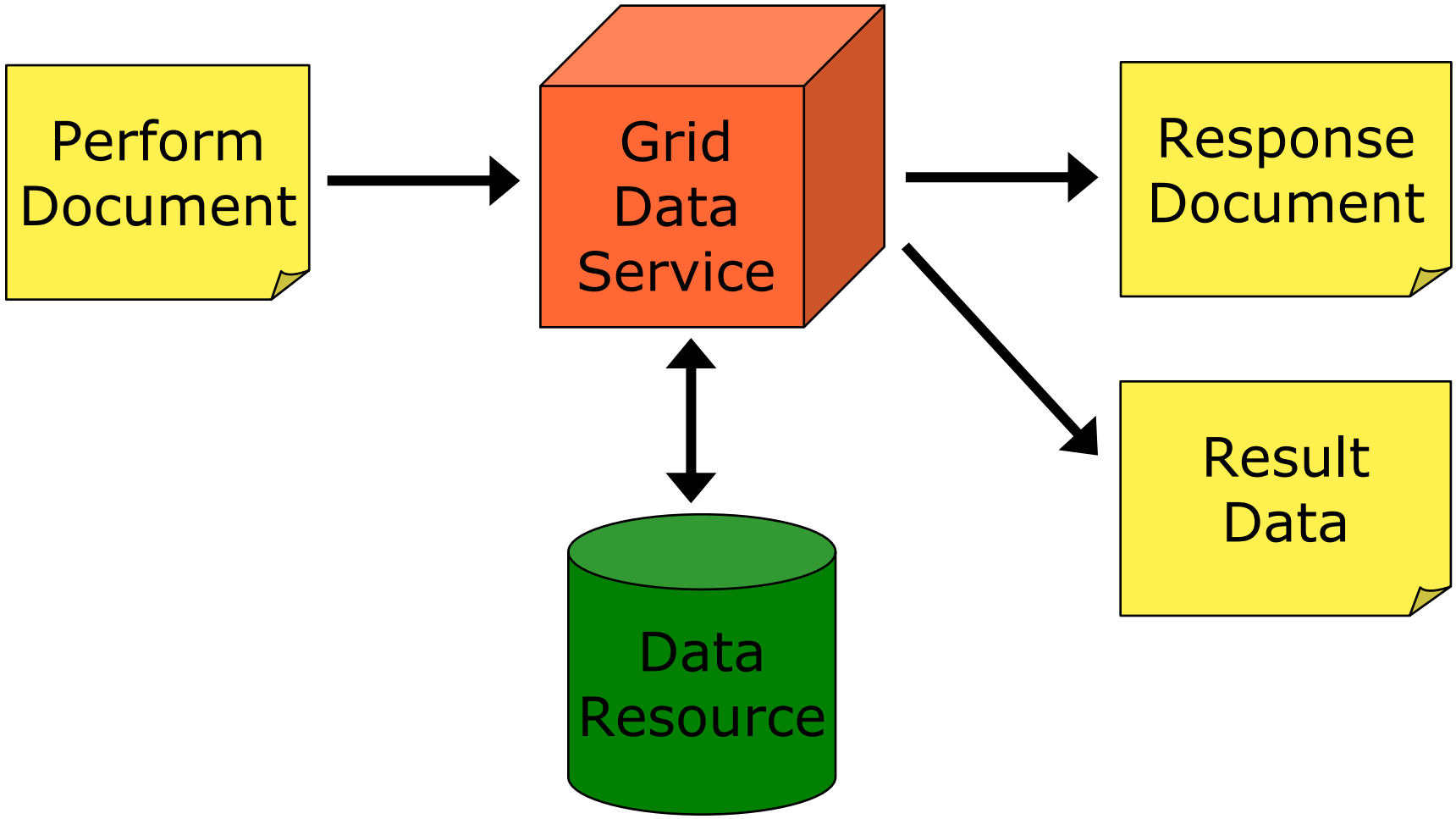
Outline

- Inside a Grid Data Service *(15 mins)*
- OGSA-DAI User Guide *(30 mins)*
- The Client Toolkit APIs *(20 mins)*
- Wrap-up *(15 mins)*

Status

- OGSA-DAI middleware
  - Release 4 of 7
  - functional and flexible
  - performance and scalability issues
- Depends on:
  - Globus Toolkit 3.2
  - Java 1.4+
  - Apache Ant
- Supports various databases
  - MySQL, Oracle, DB2, PostgreSQL, Xindice

Inside a Grid Data Service

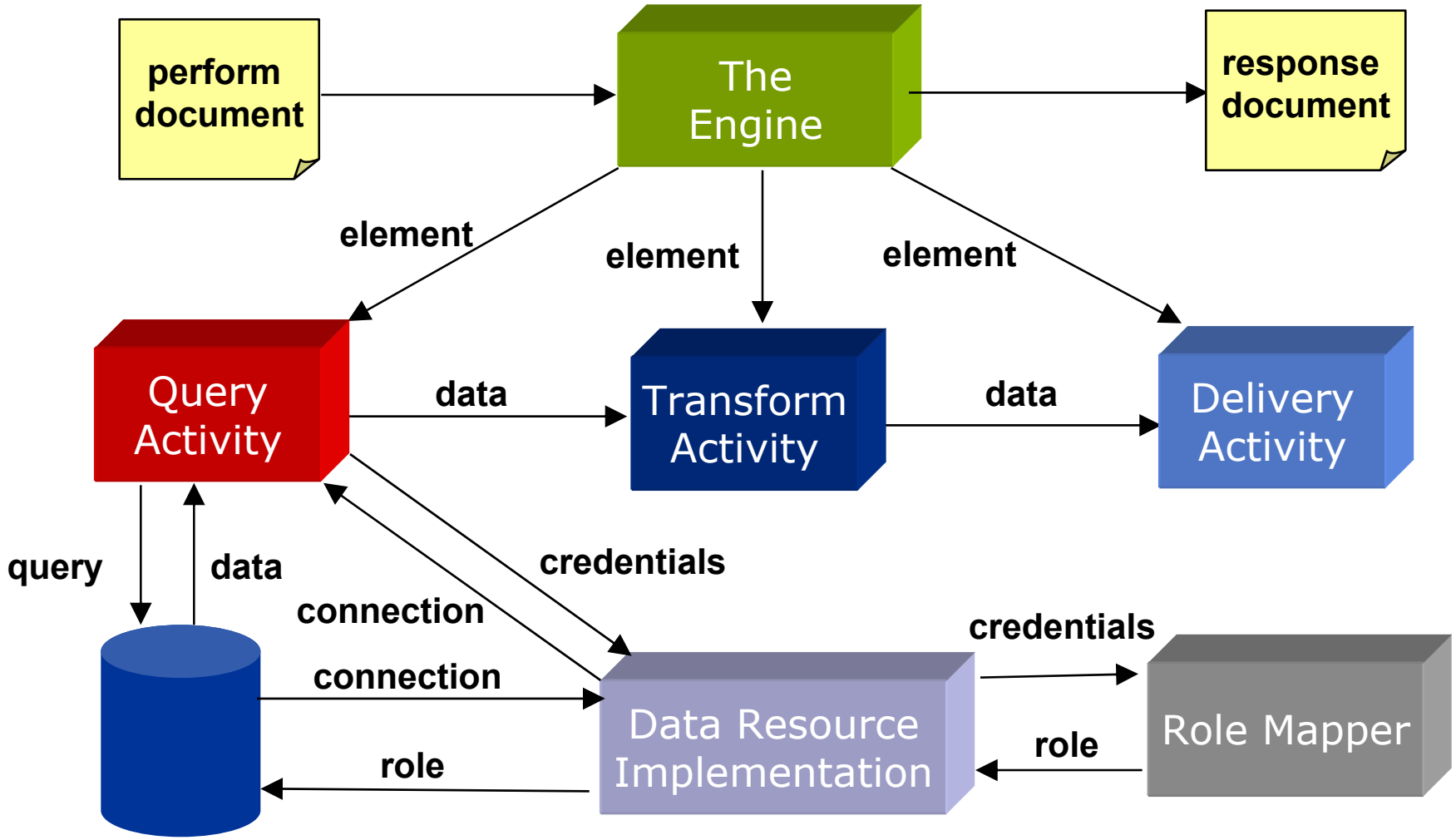

Perform Document
Grid Data Service
Response Document
Result Data
Data Resource

Overview

- Low-level components of a Grid Data Service
  - Engine
  - Activities
  - Data Resource Implementation
  - Role Mapper

- Extensibility of OGSA-DAI architecture
  - Interfaces
  - Abstract classes
  - Implementations

GDS Internals



Grid Data Service

- GDS has a document based interface
  - Consumes perform documents
  - Produces response documents
  - Additional operations for 3rd party data delivery
- Motivation for using a document interface
  - Change in behaviour ≠> interface change
  - Reduce number of operation calls
  - Extensible

The GDS Engine

- Engine is the central GDS component
- Dictates behaviour when perform documents are submitted
  - Parses and validates perform document
  - Identifies required activities implementations
  - Processes activities
  - Composes response document
  - Returns response document to GDS

Perform Documents

- Perform documents
  - Encapsulate multiple interactions with a service into a single interaction
  - Abstract each interaction into an "activity"
  - Data can flow from one activity to another

Query → Transformation → Delivery

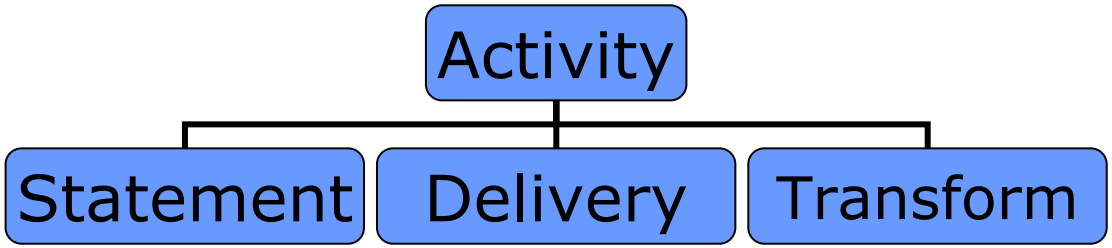
  - Not quite workflow
    - No control constructs present (conditionals, loops, variables)

Activities

- An Activity dictates an action to be performed
  - Query a data resource
  - Transform data
  - Deliver results
- Engine processes a sequence of activities
- Subset of activities available to a GDS
  - Specified in a configuration file
- Data can flow between activities

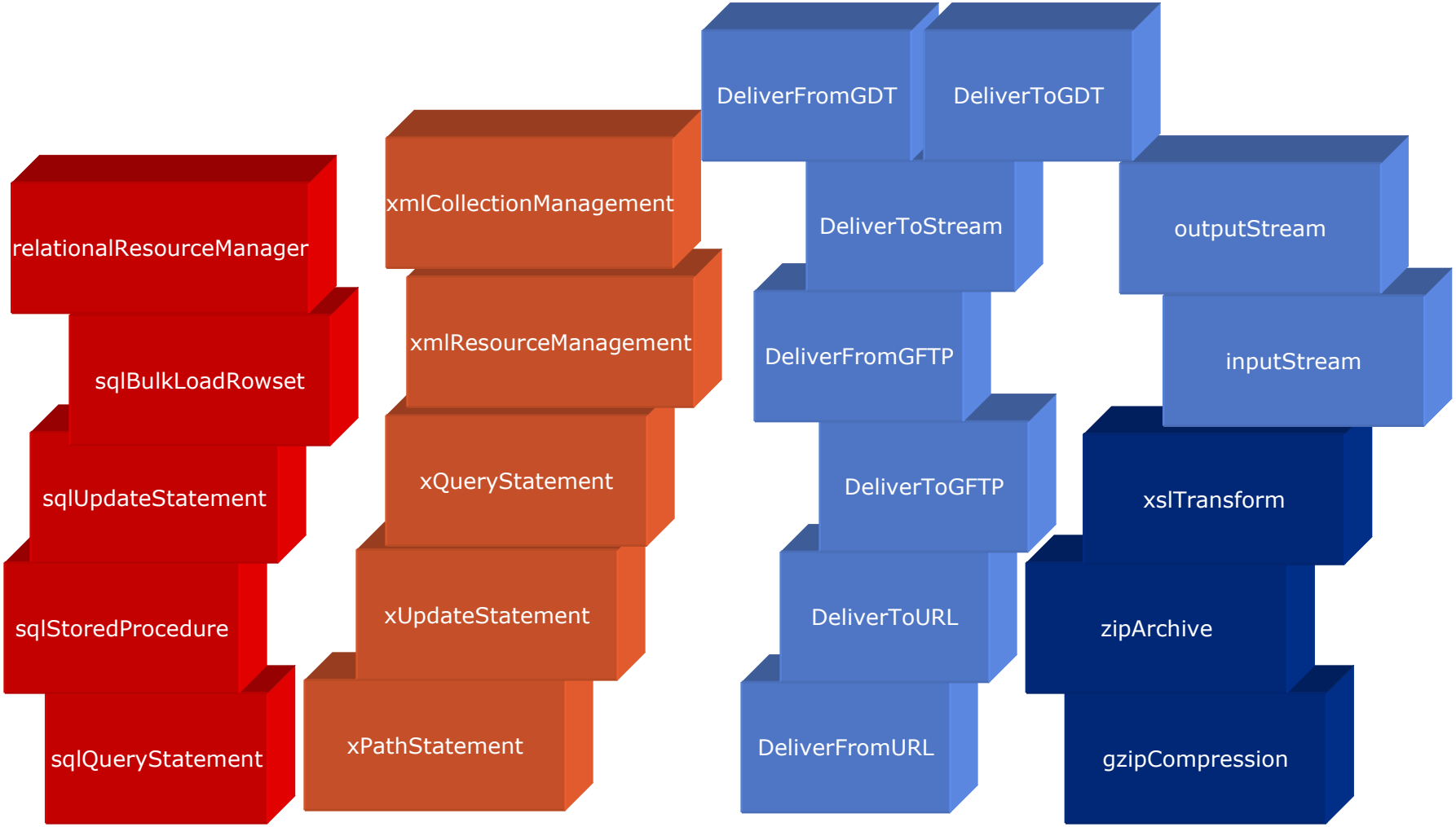
Activity Taxonomy

- Activities fall into three main functional groups



- Statement
  - Interact with the data resource
- Delivery
  - Deliver data to and from 3rd parties
- Transform
  - Perform transformations on data

Building Blocks
Predefined Activities



The Activity Framework

- Extensibility point
- Users can develop additional activities
  - To support different query languages
    - XQuery
  - To perform different kinds of transformation
    - STX
  - To deliver results using a different mechanism
    - WebDAV

- An activity requires
  - XSD schema sql_query_statement.xsd
  - Java implementation SQLQueryStatementActivity

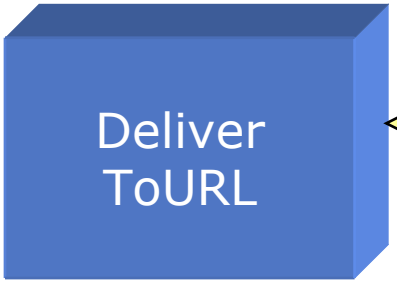
The Activity Class

- All Activity implementations extend the abstract Activity class

Activity
~ mContext: ActivityContext
+ Activity(element: Element) **~ cleanUp()** **~ initialise()** ***~ processBlock() : void*** **~ setCompleted()**

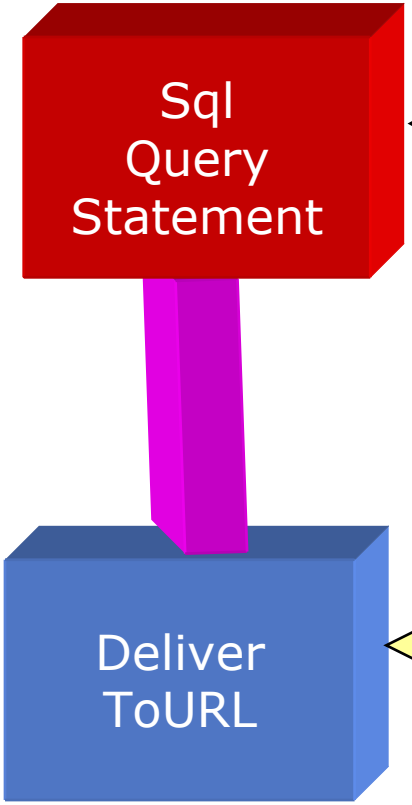
Connected Activities


```
<sqlQueryStatement name="statement">
  <expression>
      select * from myTable where id=10
  </expression>
</sqlQueryStatement>
```




```
<deliverToURL name="deliverOutput">
  <toURL>
    ftp://anon:frog@ftp.example.com/home
  </toURL>
</deliverToURL>
```

Connected Activities cont.




```
<sqlQueryStatement name="statement">
  <expression>
      select * from myTable where id=10
  </expression>
  <resultSetStream name="MyOutput"/>
</sqlQueryStatement>
```

```
<deliverToURL name="deliverOutput">
  <fromLocal from="MyOutput"/>
  <toURL>
    ftp://anon:frog@ftp.example.com/home
  </toURL>
</deliverToURL>
```

The Perform Document

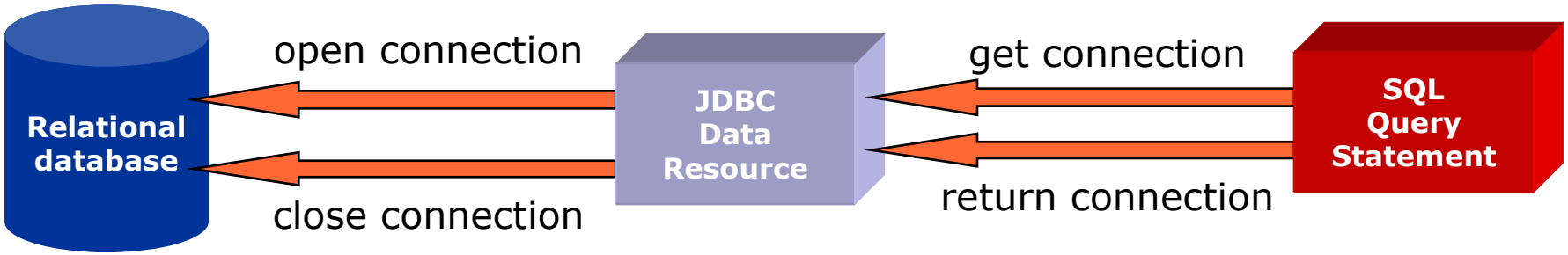
```
<?xml version="1.0" encoding="UTF-8"?>
<gridDataServicePerform
      xmlns="http://ogsadai.org.uk/namespaces/2003/07/gds/types"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://ogsadai.org.uk/namespaces/2003/07/gds/types
       ../../../../schema/ogsadai/xsd/activities/activities.xsd">

 <documentation>
  This example performs a simple select statement to retrieve one row
  from the test database then delivers the results to an FTP location.
 </documentation>

 <sqlQueryStatement name="statement">
  <expression>
      select * from littleblackbook where id=10
   </expression>
  <resultSetStream name="output"/>
 </sqlQueryStatement>

 <deliverToURL name="deliverOutput">
  <fromLocal from="output"/>
  <toURL>ftp://anon:frog@ftp.example.com/home</toURL>
 </deliverToURL>

</gridDataServicePerform>
```

Activity Inputs and Outputs

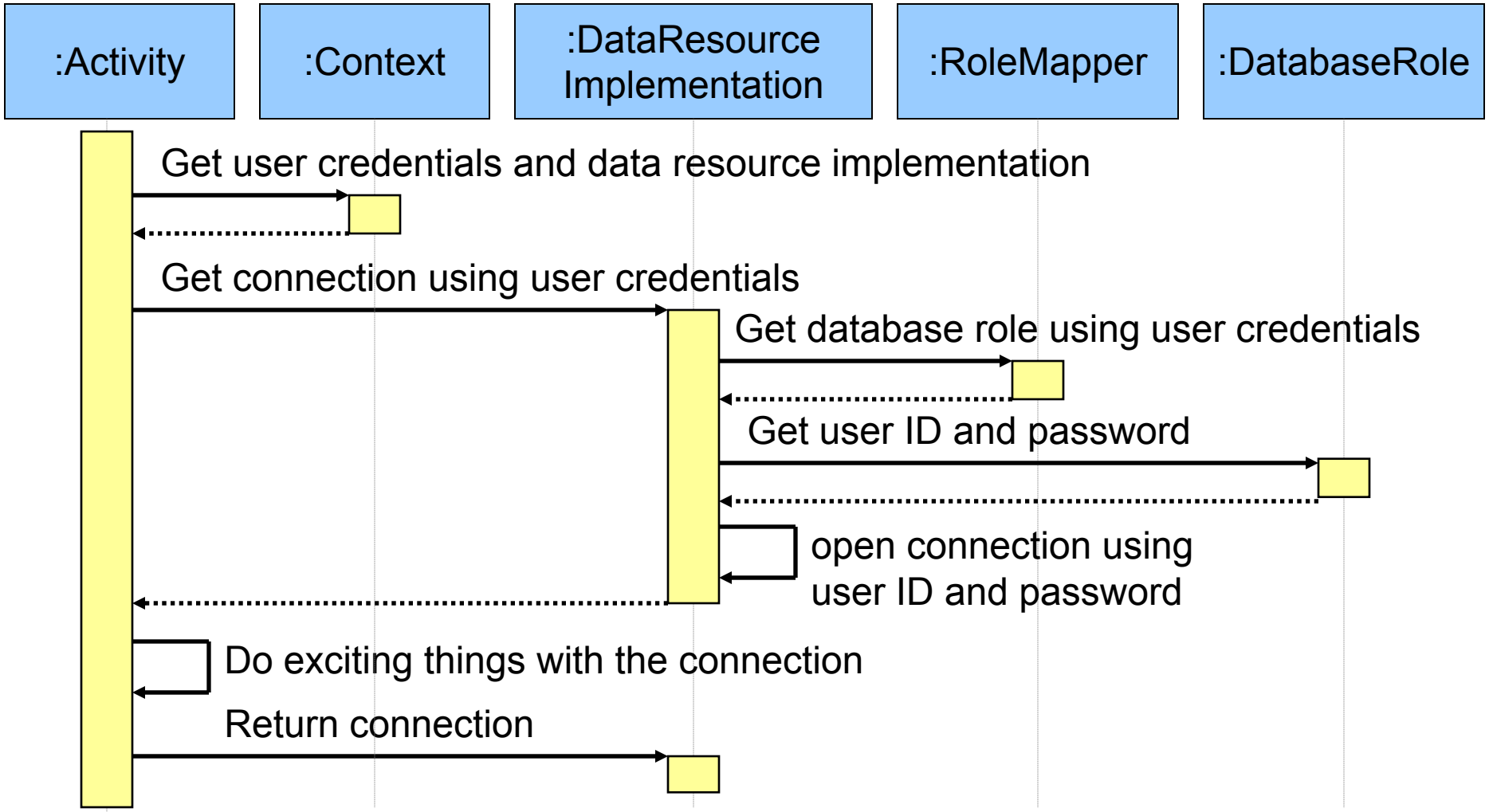
- Activities read and write blocks of data
  - Allows efficient streaming between activities
  - Reduces memory overhead
- A block is a Java Object
  - Untyped but usually a String or byte array
- Interfaces for reading and writing
  - BlockReader and BlockWriter

Data Resource Implementations

- Governs access to a data resource
  - Open/close connections
  - Validate user credentials using a RoleMapper
  - Facilitate connection pooling
- Provided for JDBC and XML:DB



Accessing Data Resource Sequence Diagram



Advantages of the Activity Model

- Avoid multiple message exchanges
  - Multiple activities within a single request
- Extensible
  - Developers can add functionality
  - Could import third party trusted activities
- Simplicity
  - Internal classes manage data flow, access to databases, etc

Issues with Activity Model

- Incomplete syntax
  - No typing of inputs and outputs
    - How do you determine the data types that can be accepted?
- Keeping implementation and XML Schema fragment in synch
- Puts workload on the server
  - May need dynamic job placement
- DAIS has factored out the perform document from the draft specs

Summary

- The Engine is the central component of a GDS
- Activities perform actions
  - Querying, Updating
  - Transforming
  - Delivering
- Data Resource Implementations manage access to underlying data resources
- Architecture designed for extensibility
  - New Activities
  - New Role Mappers
  - New Data Resource Implementations

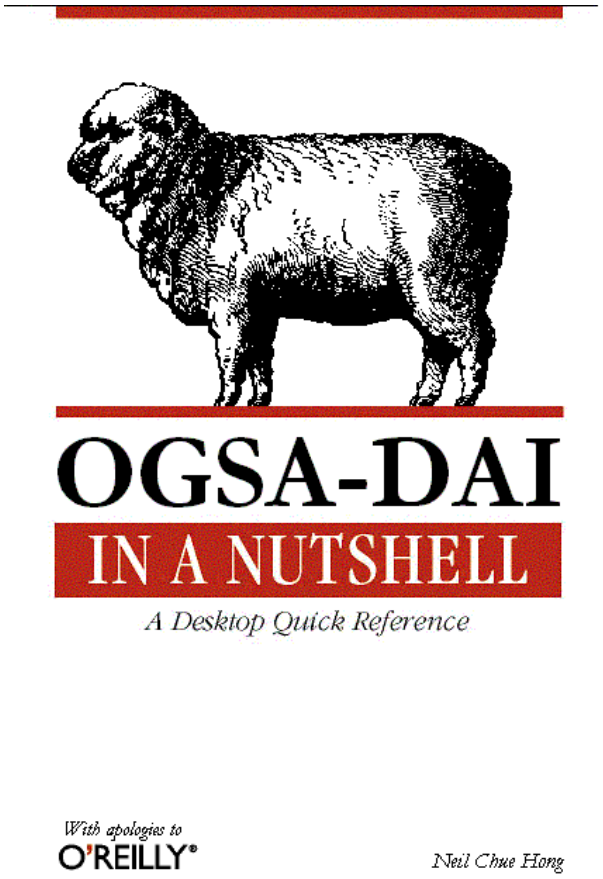
the globus alliance
www.globus.org

epcc

OGSA-DAI User Guide

OGSA-DAI in a Nutshell

- All you need to know to get started with OGSA-DAI in a handy pocket sized book!
- Updated for Version 4



Overview

- Installing OGSA-DAI
- Configuring Grid Data Service Factories
- Registering Services
- Using Grid Data Services
  - Writing perform documents
  - Using the supplied client applications
  - Using the client toolkit
- Learn by scenario

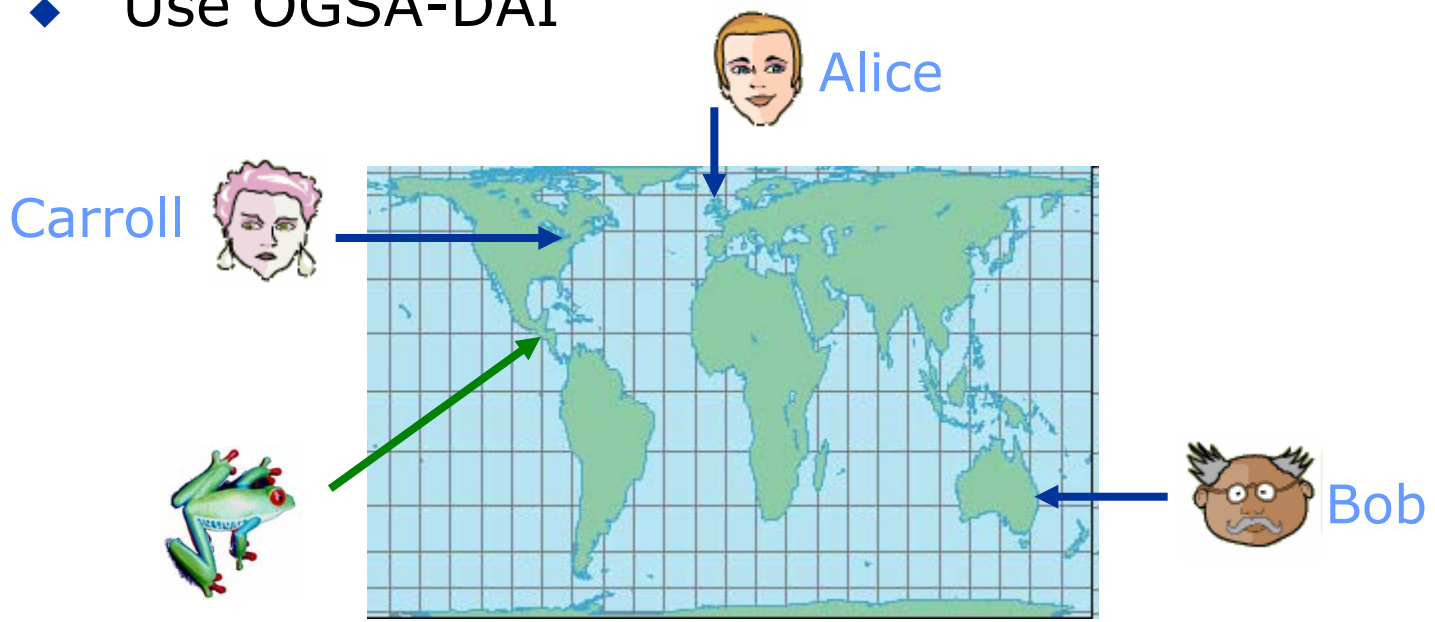
Scenario: Red Eyed Tree Frogs

Alice is a molecular biologist

- Based at the University of Edinburgh
- Mapped the genetic sequence of the Red-Eyed Tree Frog

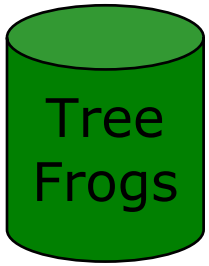
Background

- Alice wants to make her work available to the scientific community
  - Publish an on-line database
  - Use OGSA-DAI

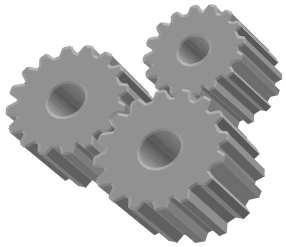


Alice's Database

- MySQL relational database
  - jdbc:mysql://localhost:3306/TreeFrogs
- Contains 1 table with 1,000,000 rows
  - GeneticSequence
- JDBC Database Driver
  - org.gjt.mm.mysql.Driver



GeneticSequence	
PK	**ID**
	Position **Chromosome** **Symbol**



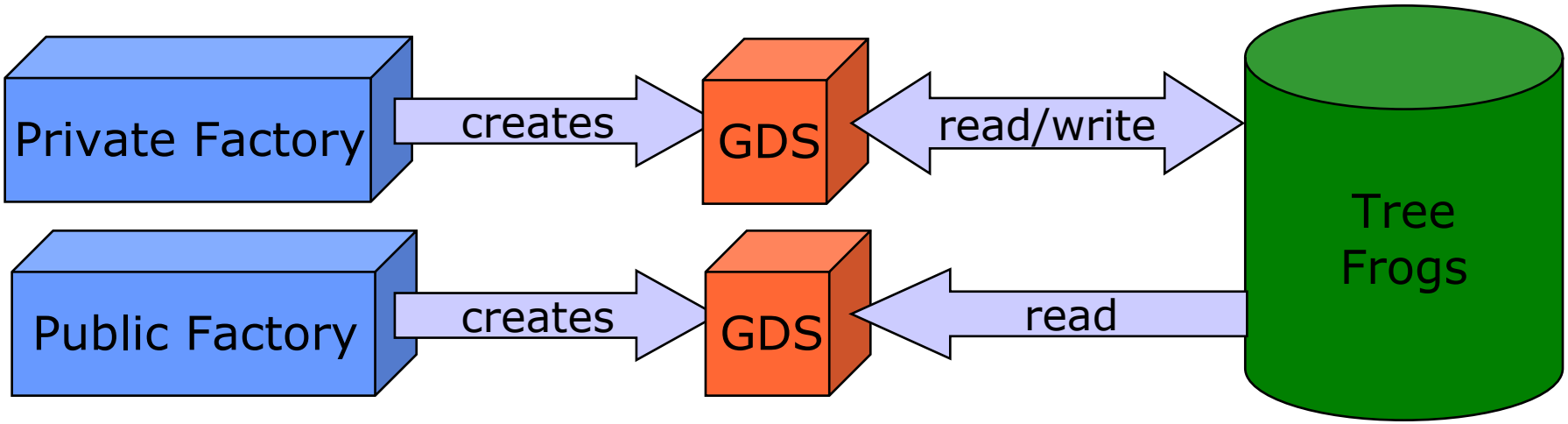
Driver

Installing OGSA-DAI

- Download OGSA-DAI software
  - http://www.ogsadai.org.uk
- Follow installation notes
  - Set-up prerequisite software
    - Java (JDK1.3 or newer)
    - Web services container (Tomcat)
    - Grid Middleware (Globus Toolkit 3.2)
    - Build tool (Ant)
    - Additional libraries (Log4J, database drivers, etc)
  - Deploy OGSA-DAI

Configuring Services

- Configure Grid Data Service Factories (GDSF)
  1. Allow specific users read/write access
  2. Allow anonymous users to search data



Part 1: Configuring Private Factory

- Allow specific users to perform
  - SQL query statements
  - SQL update statements
  - Bulk load of data

- To configure the factory:
  - Create data resource configuration file
  - Create activity configuration file
  - Create database roles file
  - Update server configuration

Data Resource Configuration

- Configuration file describes the data resource
  - Create TreeFrogsPrivate.xml
  - Base on examples\GDSFConfig\dataResourceConfig.xml

```
<dataResourceConfig>

  <!-- Database rolemap settings -->
  <roleMap implementation="...rolemap.SimpleFileRoleMapper"
    configuration="path/PrivateDatabaseRoles.xml"/>

  <!-- Database and driver settings -->
  <dataResource
    implementation="...SimpleJDBCDataResourceImplementation">
    <driver implementation="org.gjt.mm.mysql.Driver">
      <uri>jdbc:mysql://localhost:3306/treefrogs</uri>
    </driver>
  </dataResource>

</dataResourceConfig>
```

Activity Configuration

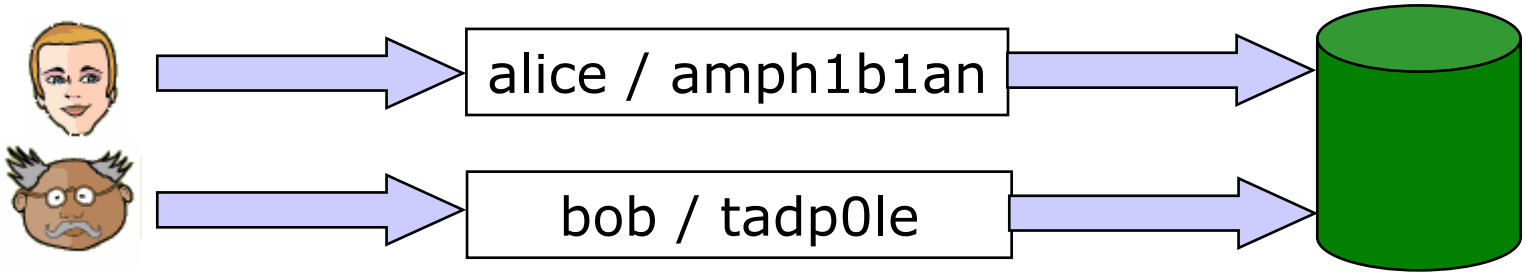
- Describes the activities that are supported by the data resource
  - Create TreeFrogsPrivateActivities.xml
  - Base on examples\GDSFConfig\activityConfig.xml

```
<activityConfiguration>
  <activityMap base=".../ogsa/schema/ogsadai/xsd/activities/">
    <!-- Activities available to GDS -->
    <activity name="sqlQueryStatement"
      implementation="package.SQLQueryStatementActivity"
      schemaFileName="path/sql_query_statement.xsd"/>
    <activity name="sqlUpdateStatement"
      implementation="package.SQLUpdateStatementActivity"
      schemaFileName="path/sql_update_statement.xsd"/>
    <activity name="sqlBulkLoadRowSet" .../>
    <activity name="deliverFromURL" .../>
  </activityMap>
</activityConfiguration>
```

Create Database Roles

- Enables access to TreeFrogs database
  - Create file PrivateDatabaseRoles.xml
  - Base on examples\RoleMap\ExampleDatabaseRoles.xml

```
<DatabaseRoles>
  <Database name="jdbc:mysql://localhost:3306/treefrogs">
    <User dn=".../CN=Alice" userid="alice" password="amph1b1an"/>
    <User dn=".../CN=Bob" userid="bob" password="tadp0le"/>
  </Database>
</DatabaseRoles>
```



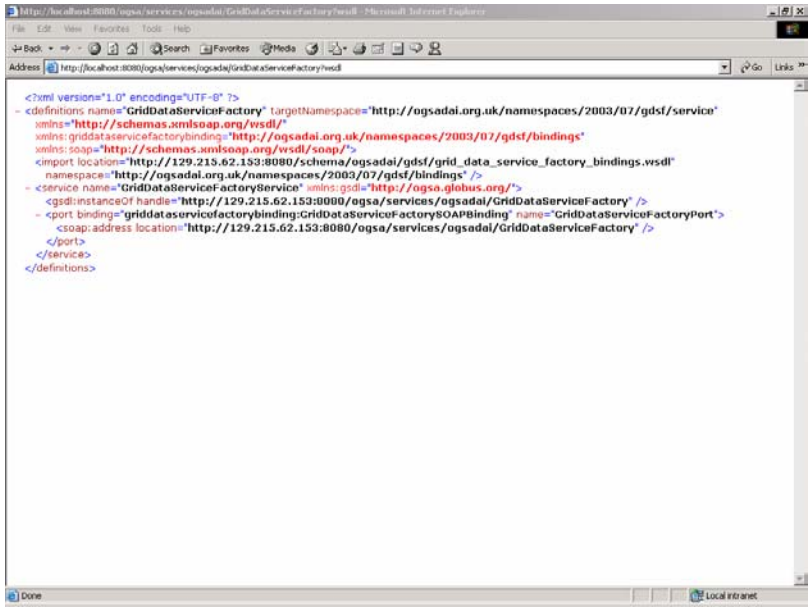
Edit Server Configuration

- Specifies the services for the container
- Loaded when Tomcat starts-up
  - Edit file server-config.xml

```
<deployment>
  ...
  <!-- GDSF-Private Service Deployment -->
  <service name="ogsadai/TreeFrogFactoryPrivate" ...>
    <parameter name="ogsadai.gdsf.config.xml.file"
      value="path/TreeFrogsPrivate.xml"/>
    <parameter name="ogsadai.gdsf.activity.xml.file"
      value="path/TreeFrogsPrivateActivities.xml"/>
    ...
  </service>
  ...
</deployment>
```

Starting the Factory

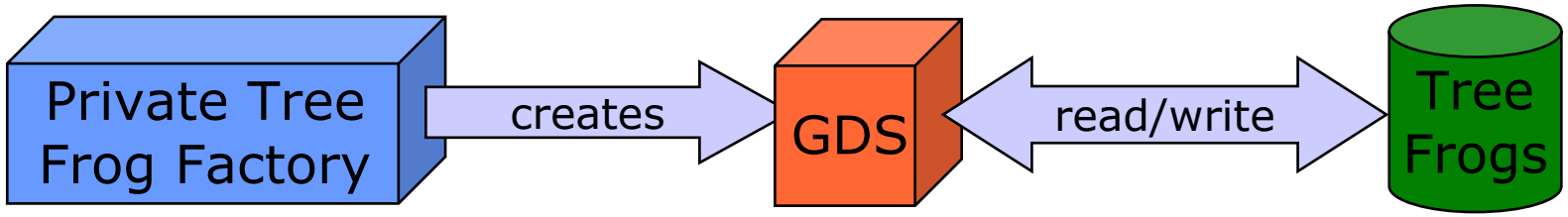
- Start service container (Tomcat)
- View the factory using a web/service browser
  - Causes factory to start up



http://localhost:8080/ogsa/services/ogsadai/TreeFrogFactoryPrivate?wsdl

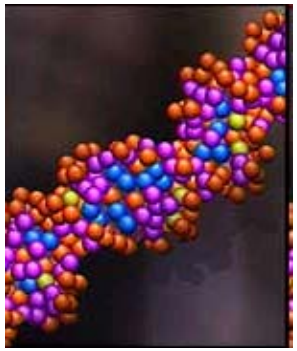
Milestone 1

- Configuration for Private Tree Frog Factory complete
- Specific users can
  - locate factory using known location
  - create GDS
  - query and update database



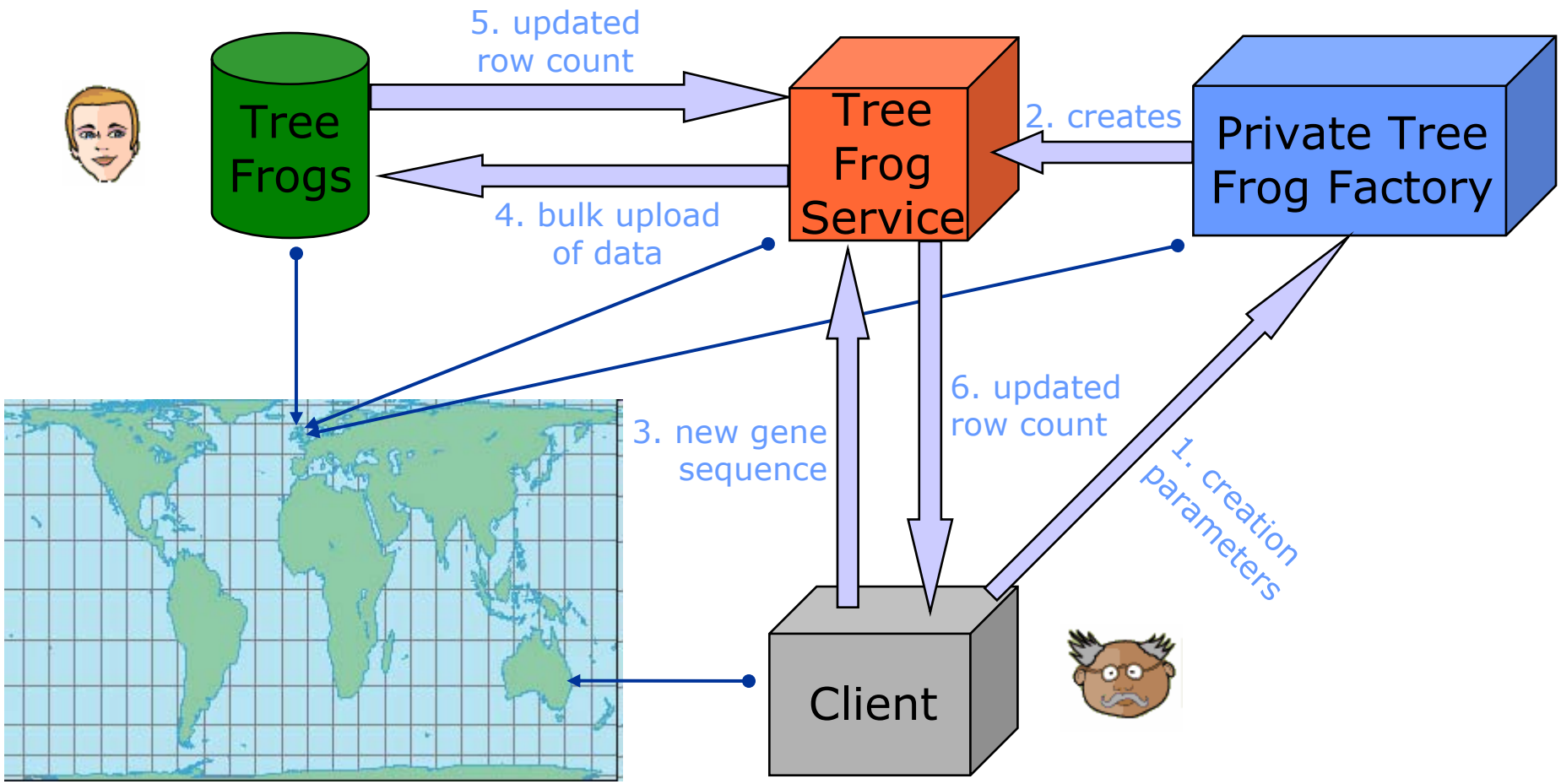
Use-case 1: Remote update

- Bob is a Professor of Biology
  - Based at the University of Sydney
  - Working in collaboration with Alice on the Red-Eyed Tree Frog genome

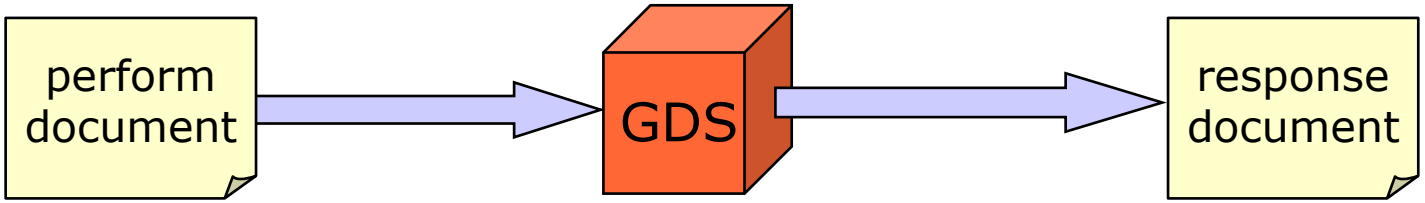


- Through Alice's OGSA-DAI services
  - Bob can contribute new sequences

Interactions



Perform Documents



- Perform documents are used to communicate with GDS
- Contain only supported activity types
  - sqlQueryStatement
  - sqlUpdateStatement
  - sqlBulkLoadRowSet

  specified in data resource configuration
- Results delivered in the response document
- Many examples provided with OGSA-DAI

Simple Query

- Select a range of chromosomes from GeneSequence
- Use sqlQueryStatement activity

```
<gridDataServicePerform ...>
  <sqlQueryStatement name="myStatement">
    <expression>
      SELECT Chromosome FROM GeneSequence
      WHERE Position > 1.1 AND Position < 1.2
    </expression>
    <webRowSetStream name="myOutput"/>
  </sqlQueryStatement>
</gridDataServicePerform>
```

Simple Query Response

- Response contained Web Row Set XML

```
<gridDataServiceResponse ...>
  <result name="myOutput" status="COMPLETE">
    <RowSet>
      ...
      <data>
        <row><col>156574335644</col></row>
        <row><col>458956403234</col></row>
      </data>
    </RowSet>
  </result>
  <result name="myStatement" status="COMPLETE"/>
</gridDataServiceResponse>
```

OGSA-DAI Clients

- Send perform documents to a GDS using a client
- OGSA-DAI provides 3 simple clients
  - Command-Line Client

```
> java uk.org.ogsadai.client.Client
        registryURL|factoryURL performDocPath
```

  - Graphical Demonstrator

```
> ant demonstrator
```

  - Data Browser

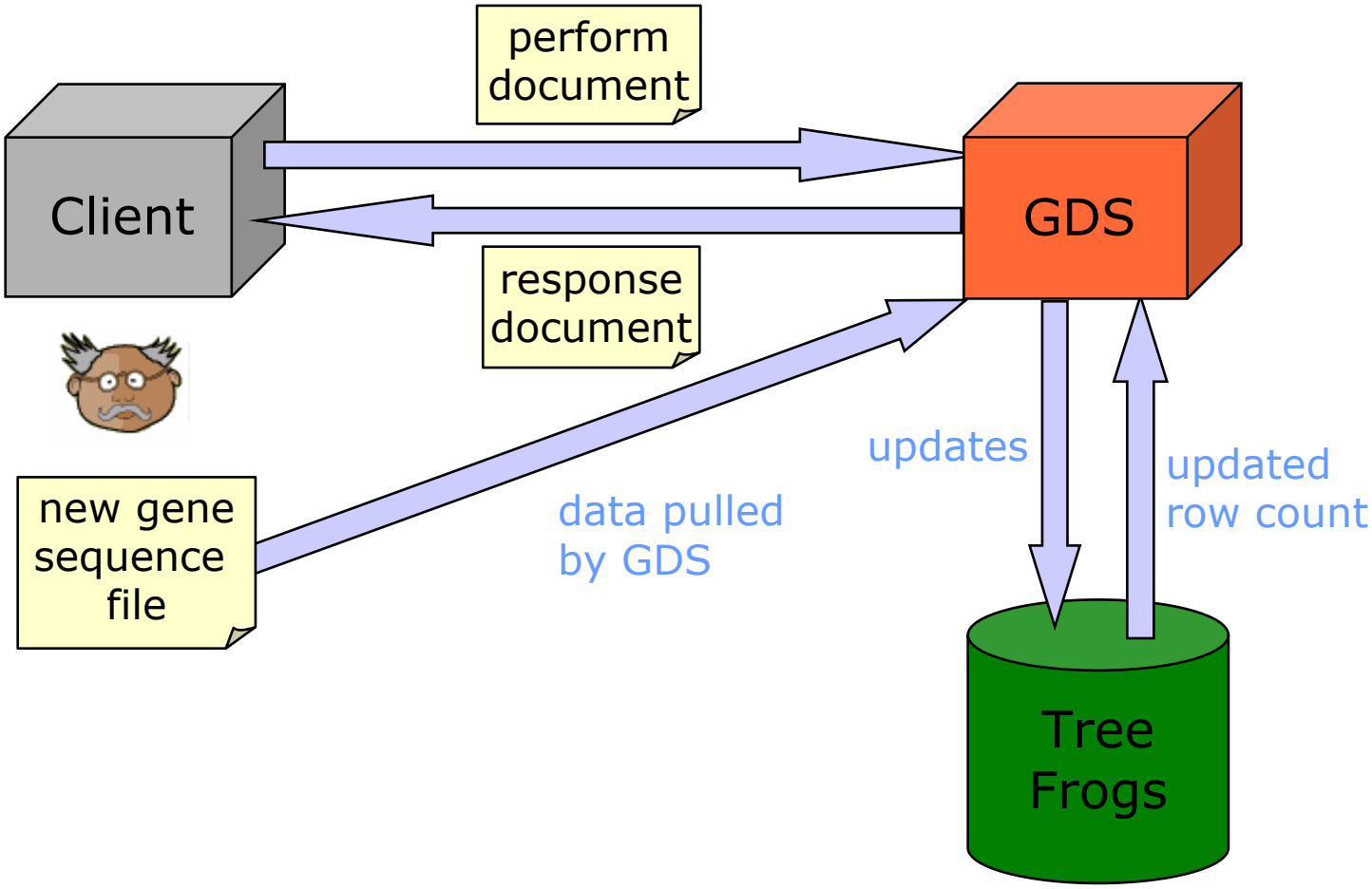
```
> ant databrowser
```

Performing Remote Update

- Bob stores his new gene sequence in a local file
- Use deliverFromURL and sqlBulkLoadRowSet activities to update remote database

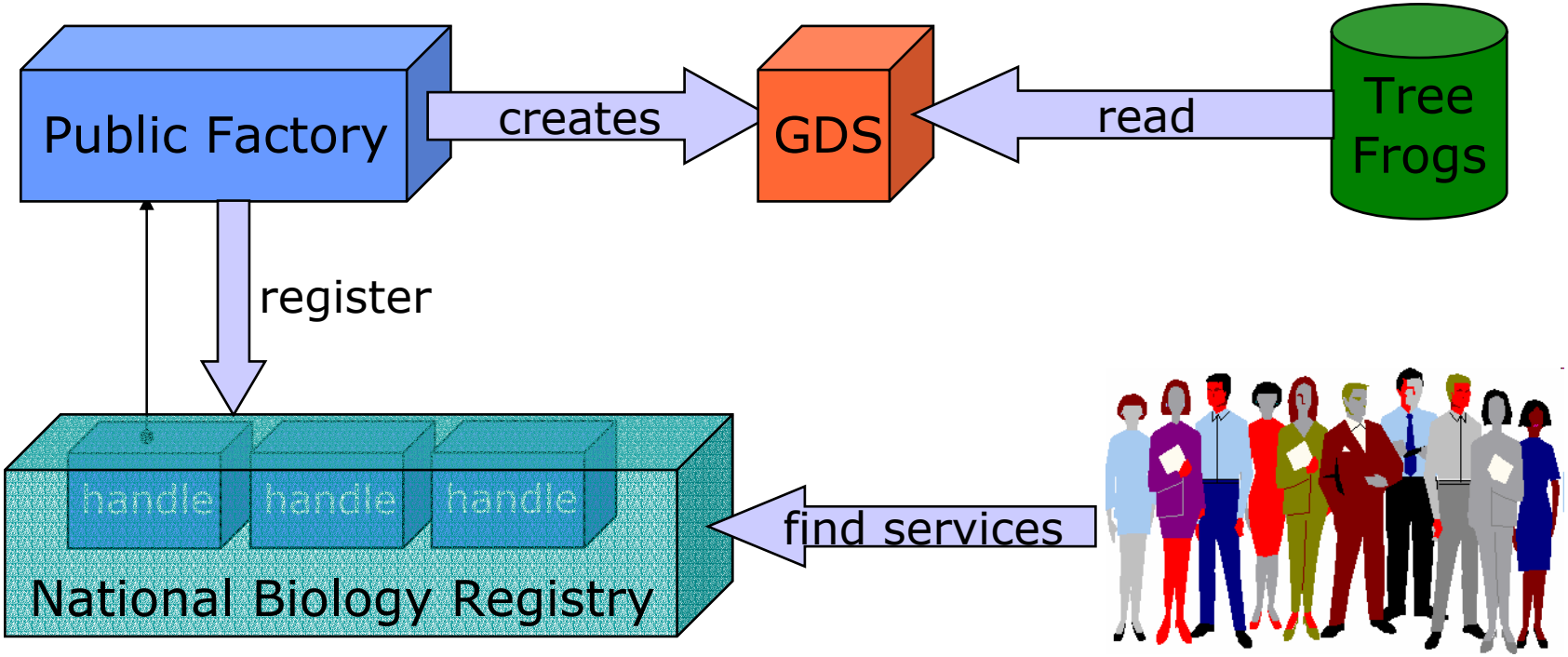
```
<gridDataServicePerform ...>
  <deliverFromURL name="myDelivery">
    <fromURL>file://path/to/newSequence.xml</fromURL>
    <toLocal name="newSequnece"/>
  </deliverFromURL>
  <sqlBulkLoadRowSet name="myBulkLoad">
    <webRowSetStream from="newSequence"/>
    <loadIntoTable tableName="GeneSequence"/>
    <resultStream name="result"/>
  </sqlBulkLoadRowSet>
</gridDataServicePerform>
```

GDS Interactions



Part 2: Configure Public Factory

- Allow anonymous users to search data
- Publish to the UK National Biology Registry



Public Factory Set-up

- Database changes
  - Alice defines **findGene** stored procedure
- Supported activities
  - SQL stored procedure
- To configure factory:
  - Create data resource configuration
  - Create activity configuration file
  - Create database roles file
  - Create service registration list
  - Update server configuration

Data Resource Configuration

- Configuration file describes the data resource
  - Create TreeFrogsPublic.xml
  - Base on examples\GDSFConfig\dataResourceConfig.xml

```
<dataResourceConfig>

  <!-- Database rolemap settings -->
  <roleMap implementation="...rolemap.SimpleFileRoleMapper"
    configuration="path/PublicDatabaseRoles.xml"/>

  <!-- Database and driver settings -->
  <dataResource
    implementation="...SimpleJDBCDataResourceImplementation">
    <driver implementation="org.gjt.mm.mysql.Driver">
      <uri>jdbc:mysql://localhost:3306/treefrogs</uri>
    </driver>
  </dataResource>

</dataResourceConfig>
```

Activity Configuration

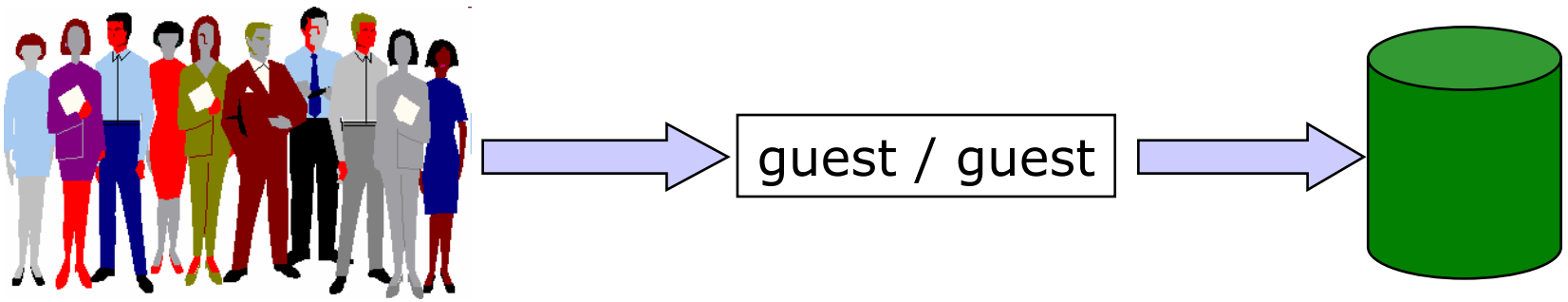
- Describes the activities that are supported by the data resource
  - Create TreeFrogsPublicActivities.xml
  - Base on examples\GDSFConfig\activityConfig.xml

```
<activityConfiguration>

  <activityMap base=".../ogsa/schema/ogsadai/xsd/activities/">

    <!- Only the sqlStoredProcedure activity
        is available to this GridDataService -->

    <activity name="sqlStoredProcedure"
      implementation="package.SQLStoredProcedureActivity"
      schemaFileName="path/sql_stored_procedure.xsd"/>

  </activityMap>

</activityConfiguration>
```

Create Database Roles

- Enables access to TreeFrogs database
  - Create file PublicDatabaseRoles.xml
  - Base on examples\RoleMap\ExampleDatabaseRoles.xml

```
<DatabaseRoles>
  <Database name="jdbc:mysql://localhost:3306/treefrogs">
    <User dn="No Certificate Provided"
      userid="guest" password="guest"/>
  </Database>
</DatabaseRoles>
```



Edit Server Configuration

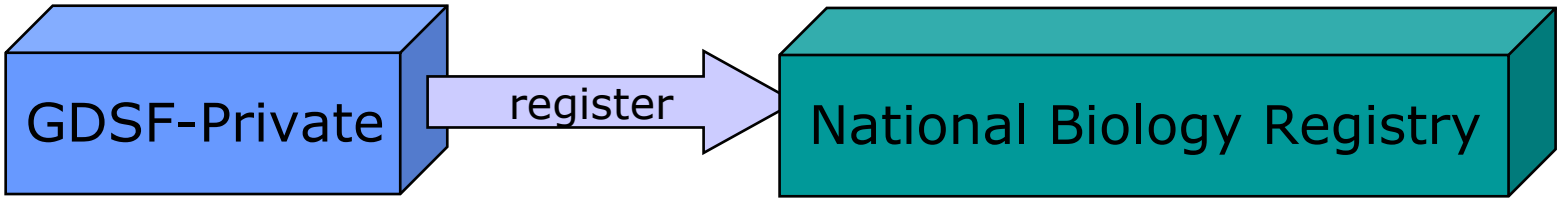
- Specifies the services for the container
- Loaded when Tomcat starts-up
  - Edit file server-config.xml

```
<deployment>
  ...
  <!-- GDSF-Private Service Deployment -->
  <service name="ogsadai/TreeFrogFactoryPublic" ...>
    <parameter name="ogsadai.gdsf.config.xml.file"
      value="path/TreeFrogsPublic.xml"/>
    <parameter name="ogsadai.gdsf.activity.xml.file"
      value="path/TreeFrogsPublicActivities.xml"/>
    <parameter name="ogsadai.gdsf.registrations.xml.file"
      value="path/TreeFrogsRegistrationList.xml"/>
    ...
  </service>
  ...
</deployment>
```

Create Service Registration List

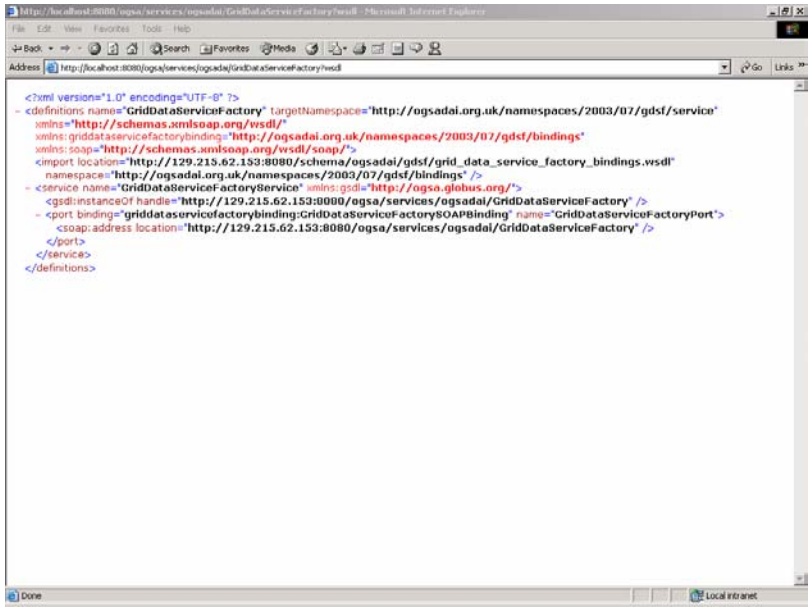
- Specifies a list of service group registries
- Factory is registered with each registry
  - Create file TreeFrogsRegistrationList.xml
  - Base on example\GDSFConfig\registrationList.xml

```
<gdsfRegistrationList ...>
  <gdsfRegistration ...
    gsh="http://www.biology.org:8080/ogsa/services/
         ogsadai/NationalBiologyRegistry"/>
</gdsfRegistrationList>
```



Starting the Factory

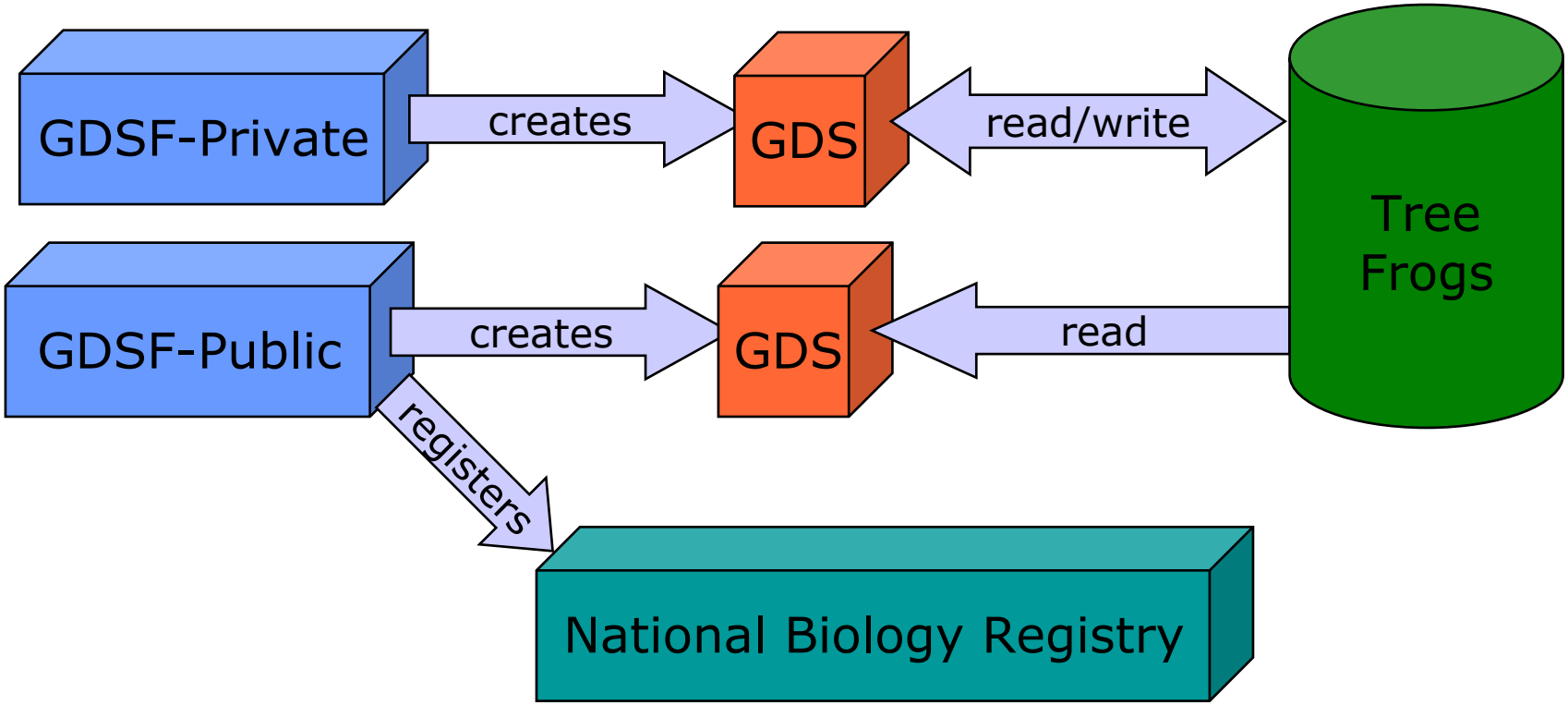
- Start service container (Tomcat)
- View the factory using a web/service browser
  - Causes factory to start up
  - Automatically registers with NationalBiologyRegister



http://localhost:8080/ogsa/services/ogsadai/TreeFrogFactoryPublic?wsdl

Milestone 2

- Configuration for Public and Private Factories complete
  - Specific users have read/write access
  - Anonymous users can search data via stored procedure

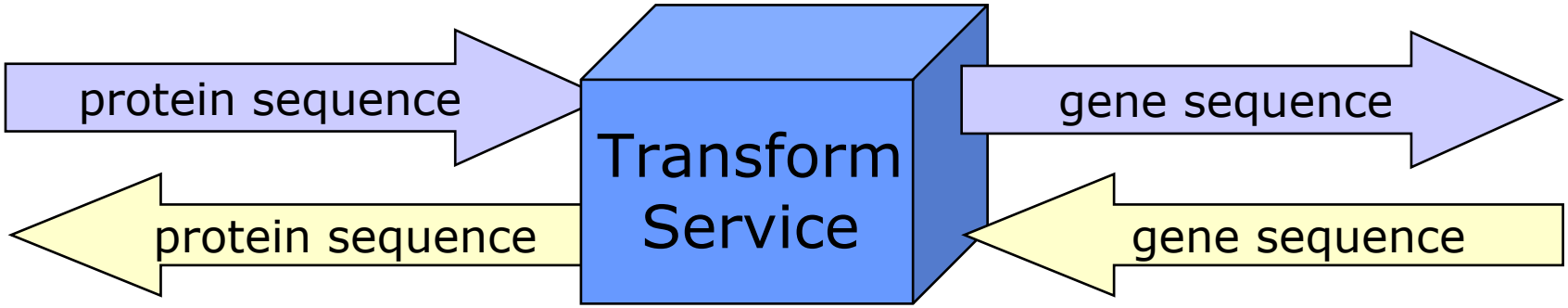


Use-case: Query with transformations

- Carroll is a biochemist
  - Works for a small drugs company in Chicago
  - Investigating toxin in saliva of Fire Bellied Toad
  - Wants to compare proteins with Red Eyed Tree Frog

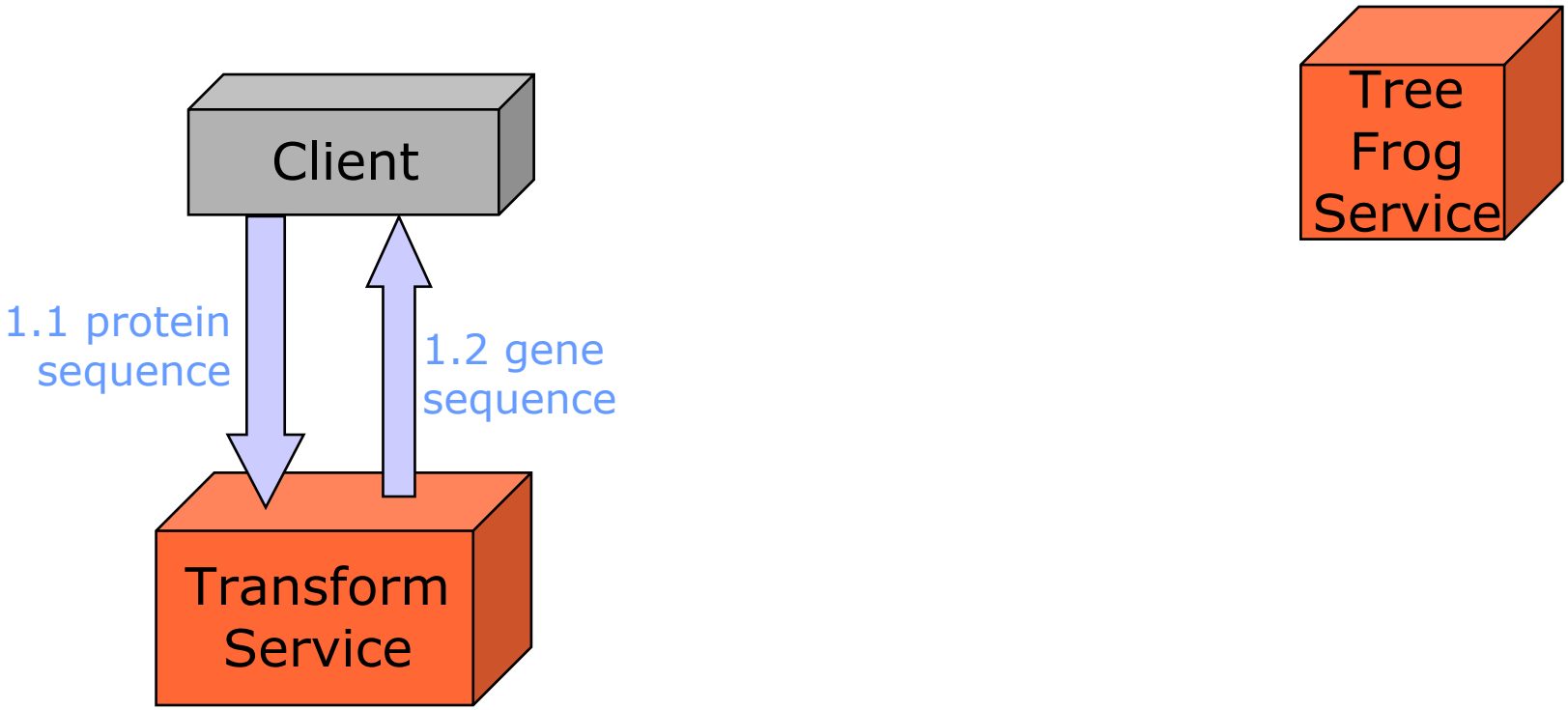
Transforming Sequences

- Carroll has a protein sequence
- Alice's data is encoded as a gene sequence
- There is a public Grid Data Transformation Service available at Newcastle University



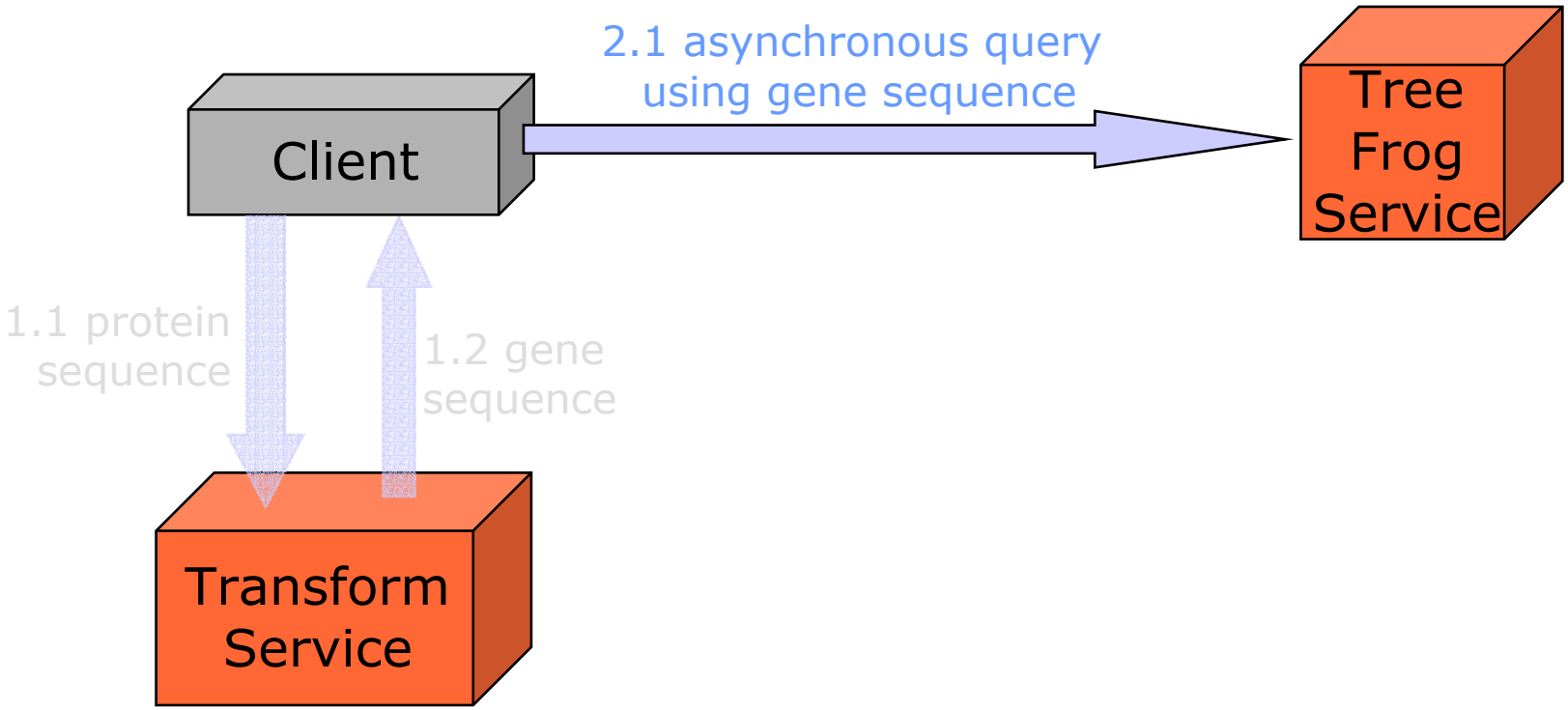
Interactions

1. Transform protein sequence needed for query



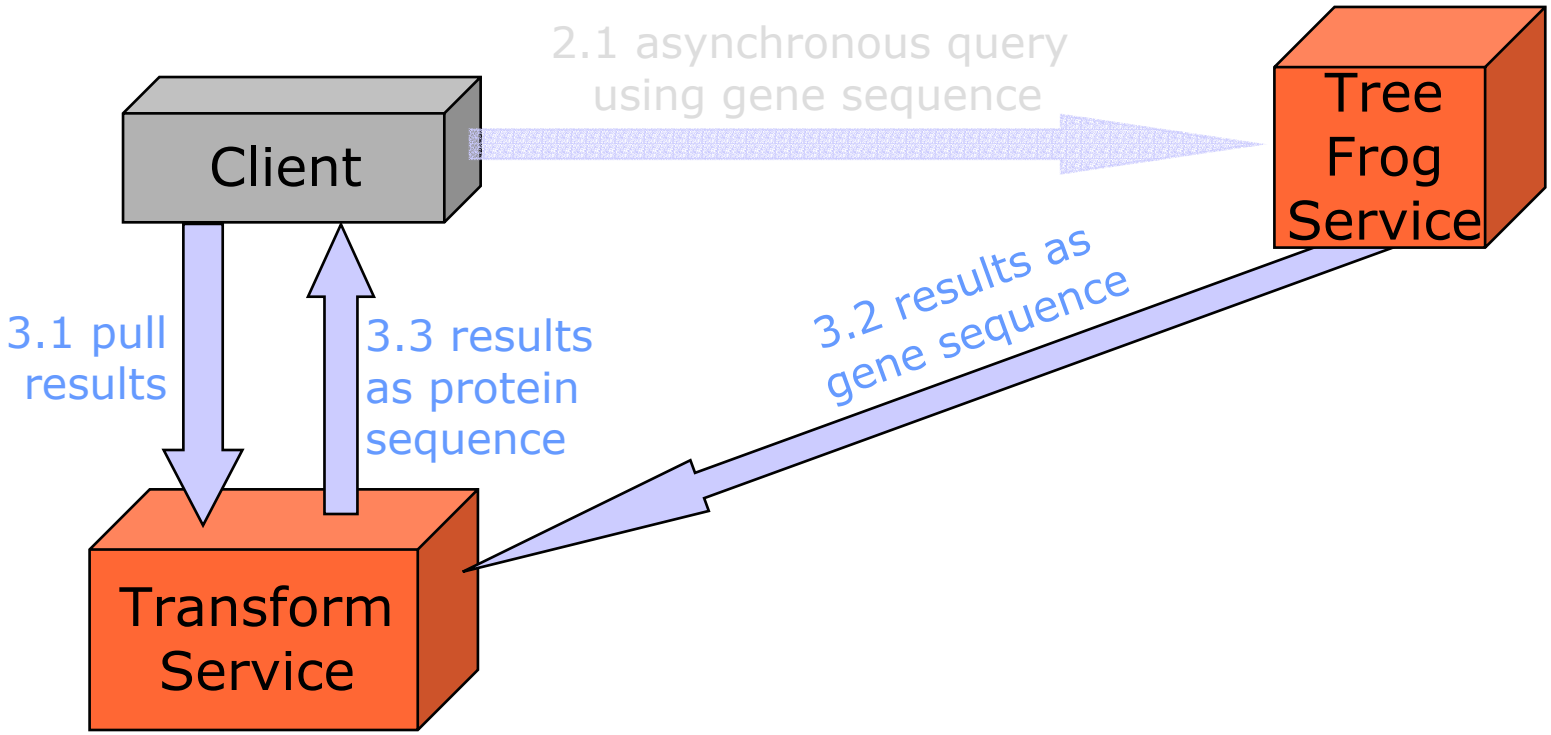
Interactions

1. Transform protein sequence needed for query
2. Query tree frog gene sequence asynchronously



Interactions

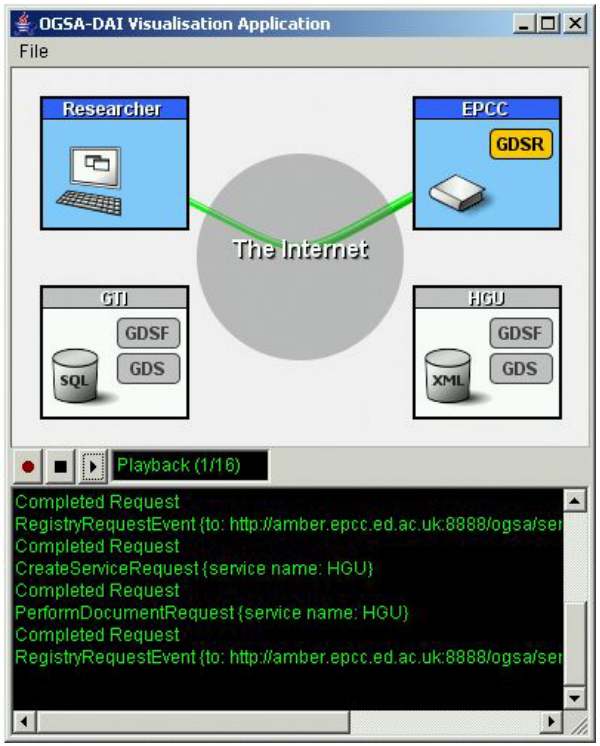
1. Transform protein sequence needed for query
2. Query tree frog gene sequence asynchronously
3. Transform results back into protein sequence



Client Toolkit

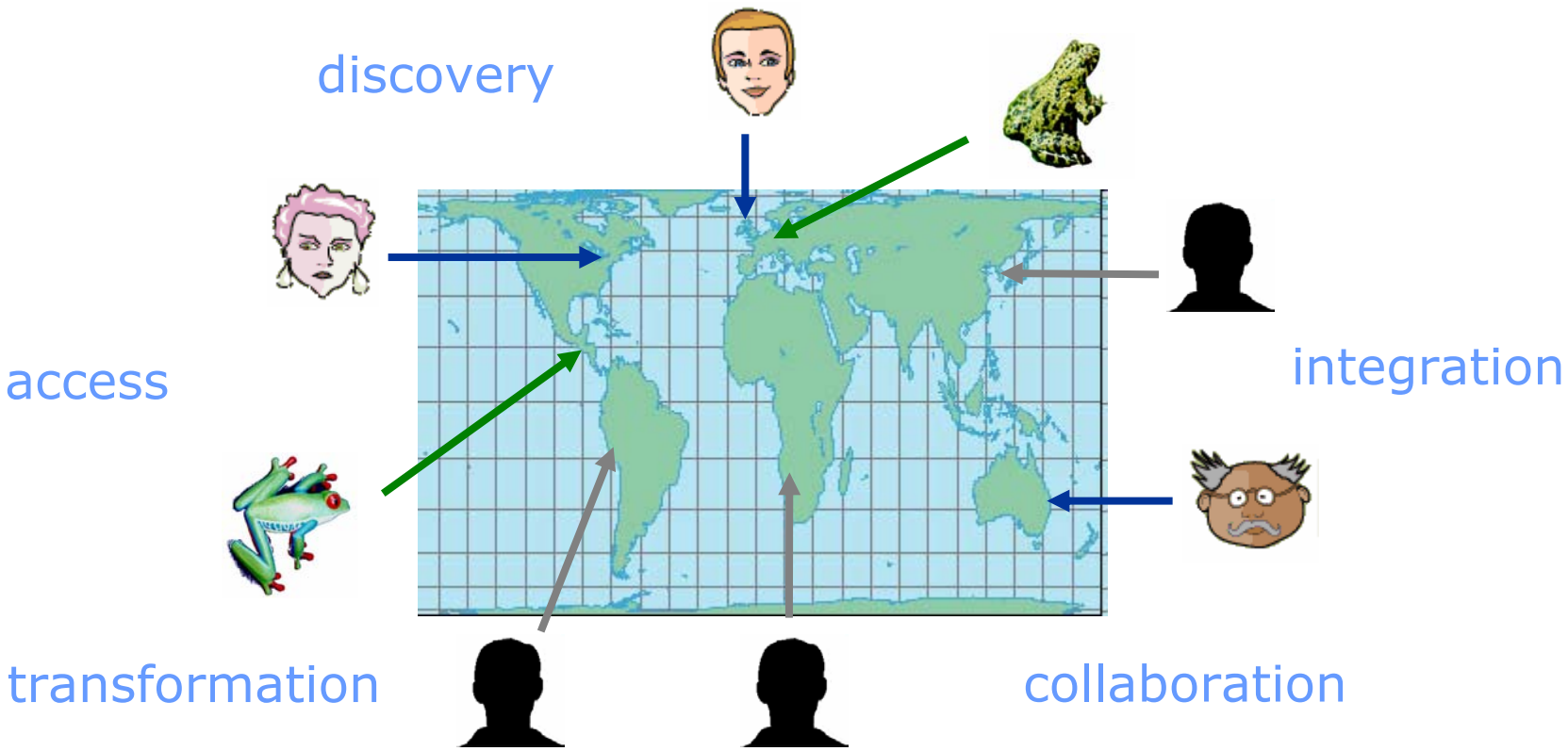
- Why? Writing XML is a pain!
- A programming API which makes writing applications easier
  - Now: Java
  - Next: Perl, C, C#?

```
// Create a query
SQLQuery query = new SQLQuery(SQLQueryString);

// Perform the query
Response response = gds.perform(query);

// Display the result
ResultSet rs = query.getResultSet();
displayResultSet(rs, 1);
```



Conclusion

- OGSA-DAI provides middleware tools to grid-enable existing databases



The Client Toolkit

Amy Krause and Tom Sugden
a.krause@epcc.ed.ac.uk
tom@epcc.ed.ac.uk

Overview

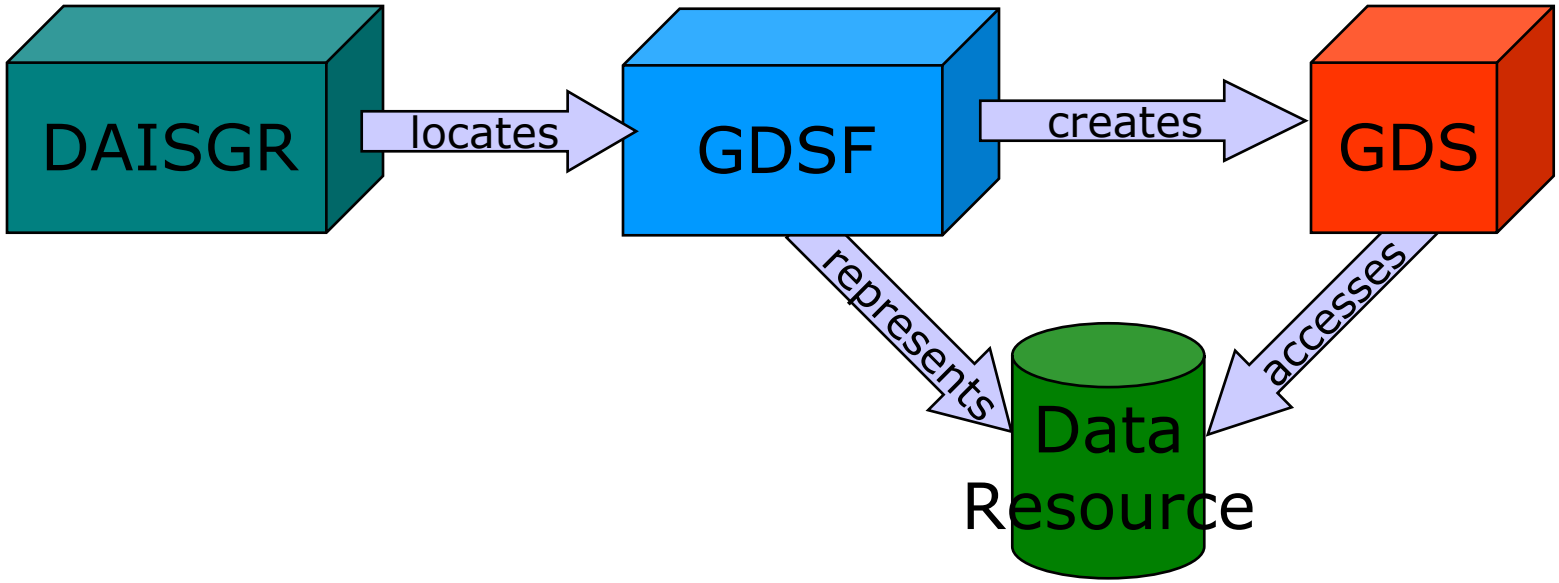
- The Client Toolkit
- OGSA-DAI Service Types
- Locating and Creating Data Services
- Requests and Results
- Delivery
- Data Integration Example

Why use a Client Toolkit?

- Nobody wants to read or write XML!
- Protects developer from
  - Changes in activity schema
  - Changes in service interfaces
  - Low-level APIs
  - DOM manipulation

OGSA-DAI Services

- OGSA-DAI uses three main service types
  - DAISGR (registry) for discovery
  - GDSF (factory) to represent a data resource
  - GDS (data service) to access a data resource



ServiceFetcher

- The ServiceFetcher class creates service objects from a URL

```
ServiceGroupRegistry registry =
        ServiceFetcher.getRegistry( registryHandle );
GridDataServiceFactory factory =
        ServiceFetcher.getFactory( factoryHandle );
GridDataService service =
        ServiceFetcher.getGridDataService( handle );
```

Registry

- A registry holds a list of service handles and associated metadata
- Clients can query registry for all Grid Data Factories

```
GridServiceMetaData[] services =
    registry.listServices(
            OGSADAIConstants.GDSF_PORT_TYPE );
```

- The *GridServiceMetaData* object contains the handle and the port types that the factory implements

```
String handle = services[0].getHandle();
QName[] portTypes = services[0].getPortTypes();
```

Creating Data Services

- A factory object can create a new Grid Data Service.

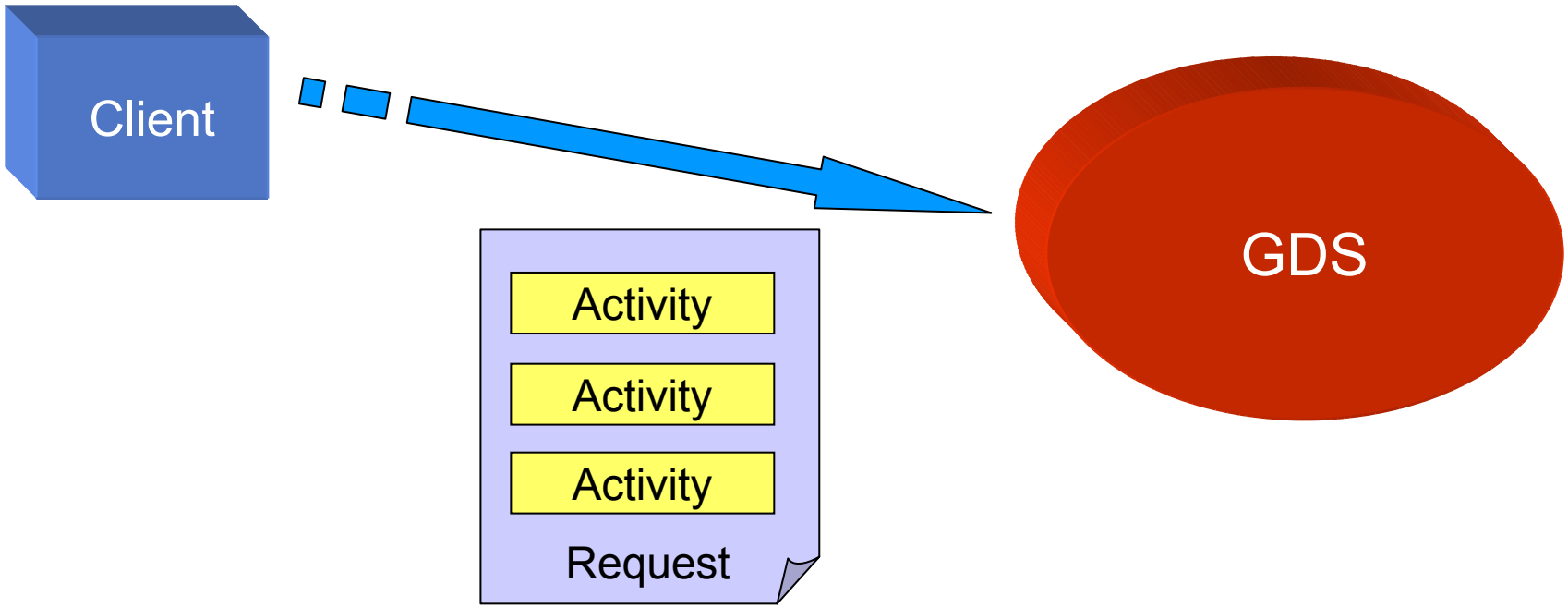
```
GridDataService service =
            factory.createGridDataService();
```

- Grid Data Services are transient (i.e. have finite lifetime) so they can be destroyed by the user.

```
service.destroy();
```

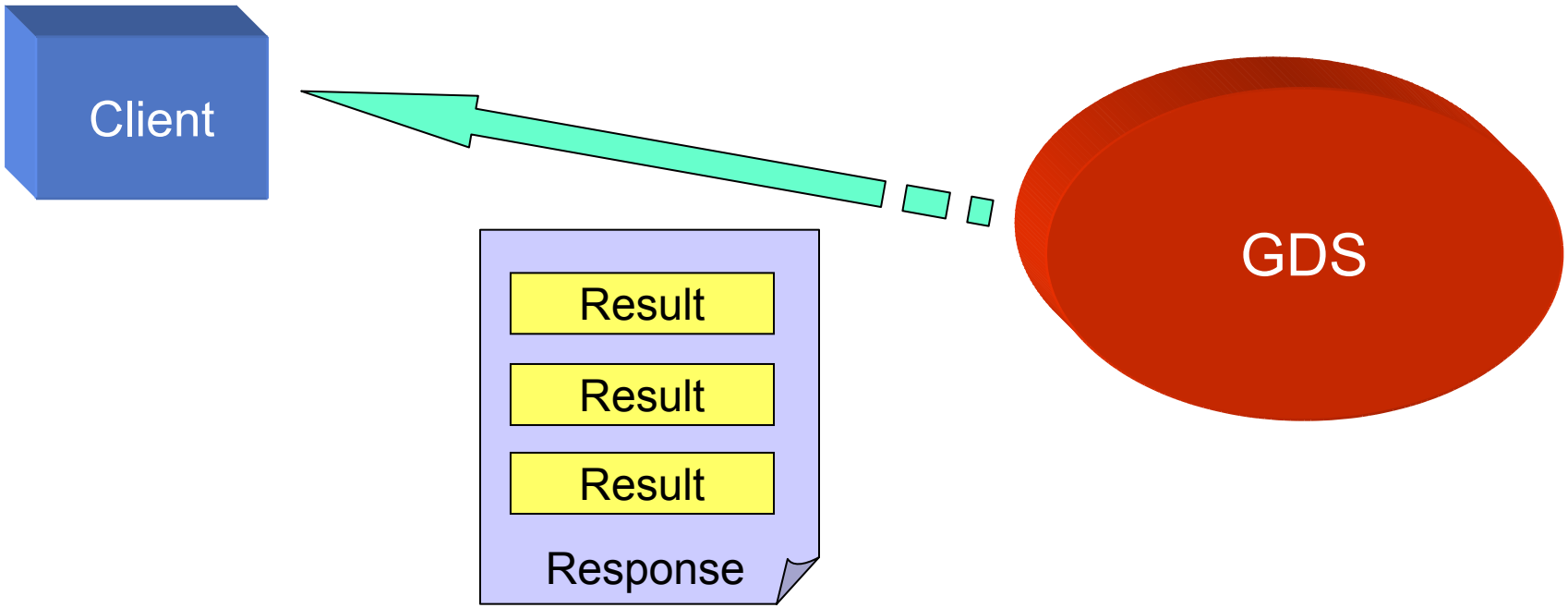
Interaction with a GDS

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



Interaction with a GDS

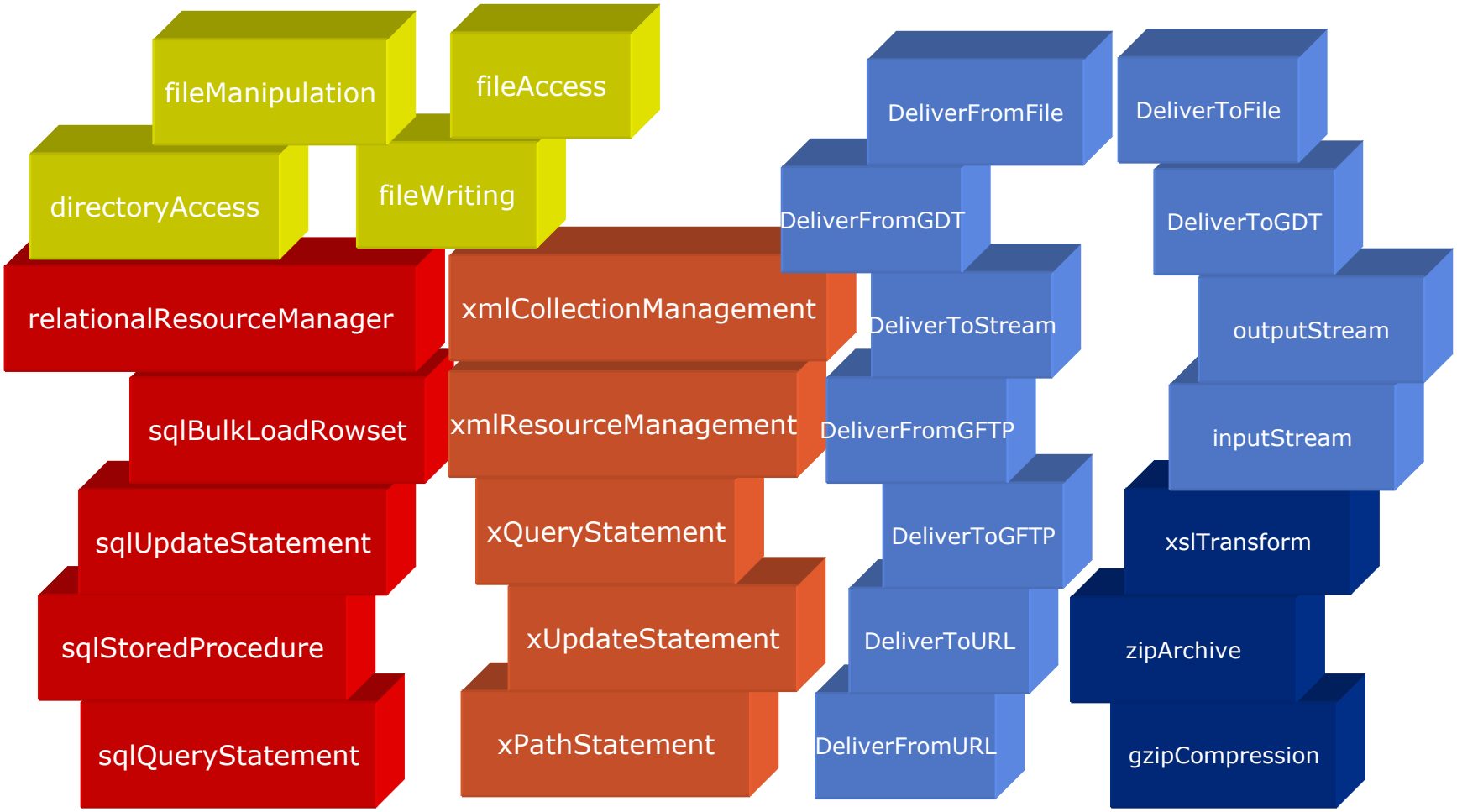
- The Data service processes the request
- Returns a response document with a result for each activity



Activities and Requests

- A request contains a set of activities
- An activity dictates an action to be performed
  - Query a data resource
  - Transform data
  - Deliver results
- Data can flow between activities

Predefined Activities



Examples of Activities

- SQLQuery

```
SQLQuery query = new SQLQuery(
   "select * from littleblackbook where id='3475'");
```

- XPathQuery

```
XPathQuery query = new XPathQuery( "/entry[@id<10]" );
```

- XSLTransform

```
XSLTransform transform = new XSLTransform();
```

- DeliverToGFTP

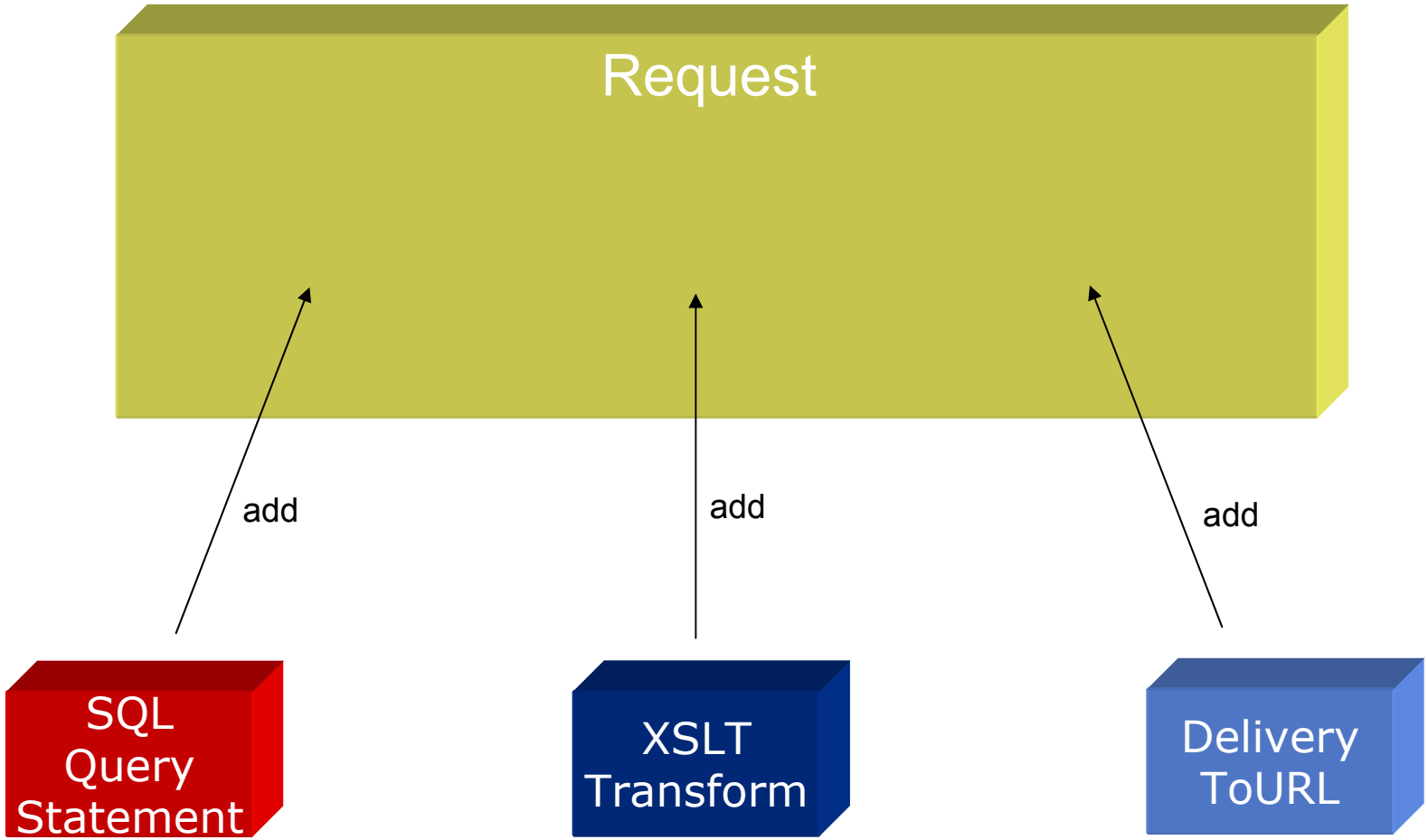
```
DeliverToGFTP deliver = new DeliverToGFTP(
       "ogsadai.org.uk", 8080, "myresults.txt" );
```

Simple Requests

- Simple requests consist of only one activity
- Send the activity directly to the perform method

```
SQLQuery query = new SQLQuery(
   "select * from littleblackbook where id='3475'");

Response response = service.perform( query );
```

Constructing a Request



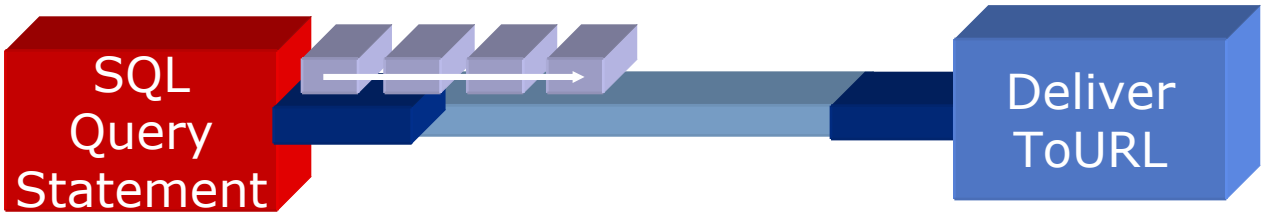
Constructing a Request cont.


```
ActivityRequest request = new ActivityRequest;
request.add( query );
request.add( transform );
request.add( delivery );
```

Data Flow

- Connecting activities

```
SQLQuery query = new SQLQuery(
   "select * from littleblackbook where id<=1000");
DeliverToURL deliver = new DeliverToURL( url );

deliver.setInput( query.getOutput() );
```



Performing Requests

- Finally… perform the request!

```
Response response = service.perform( Request );
```

- The response contains status and results of each activity in the request.

```
System.out.println( response.getAsString() );
```

Processing Results

- Varying formats of output data
  - SQLQuery
    - JDBC ResultSet:

```
ResultSet rs = query.getResultSet();
```

  - SQLUpdate
    - Integer:

```
int rows = update.getModifiedRows();
```

  - XPathQuery
    - XML:DB ResourceSet:

```
ResourceSet results = query.getResourceSet();
```

- Output can always be retrieved as a String

```
String output = myactivity.getOutput().getData();
```

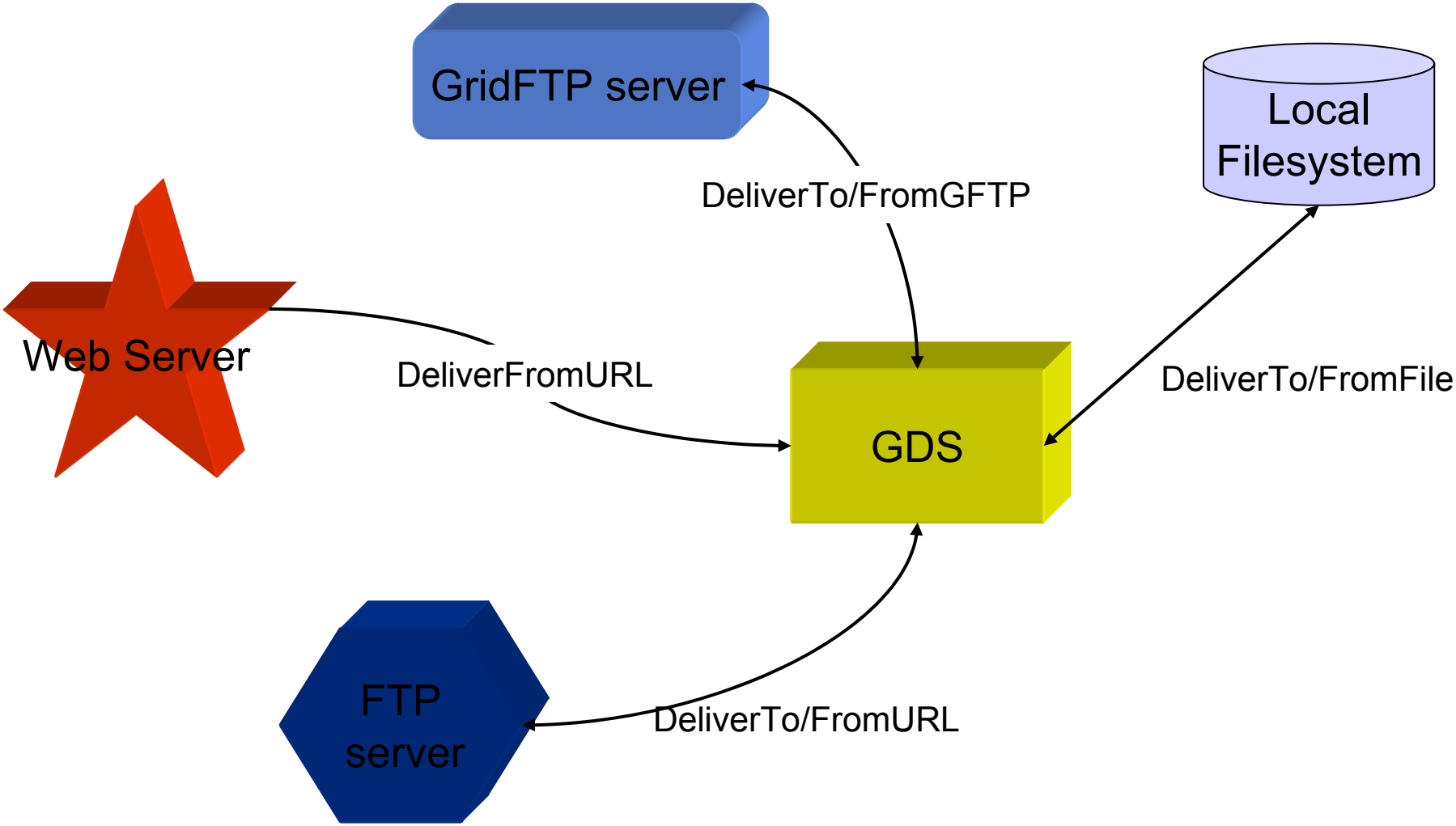
Delivery

- Data can be pulled from or pushed to a remote location.
- OGSA-DAI supports third-party transfer using *FTP*, *HTTP*, or *GridFTP* protocols.

```
DeliverToURL deliver = new DeliverToURL( url );
deliver.setInput( myactivity.getOutput() );
```

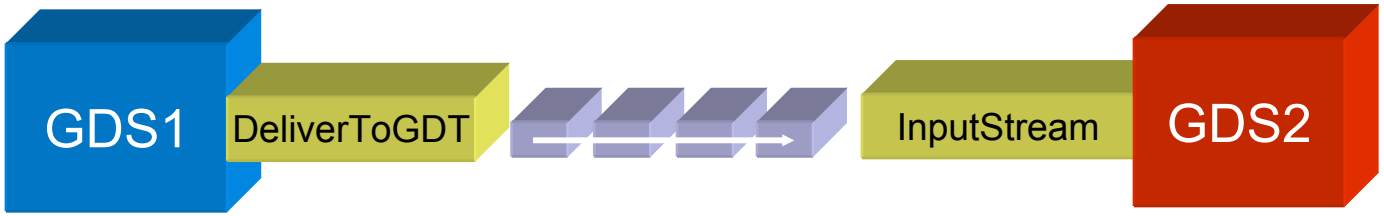
```
DeliverToGFTP deliver = new DeliverToGFTP(
    "ogsadai.org.uk", 8080, "tmp/data.out" );
deliver.setInput( myactivity.getOutput() );
```

Delivery Methods



Delivering data to another GDS

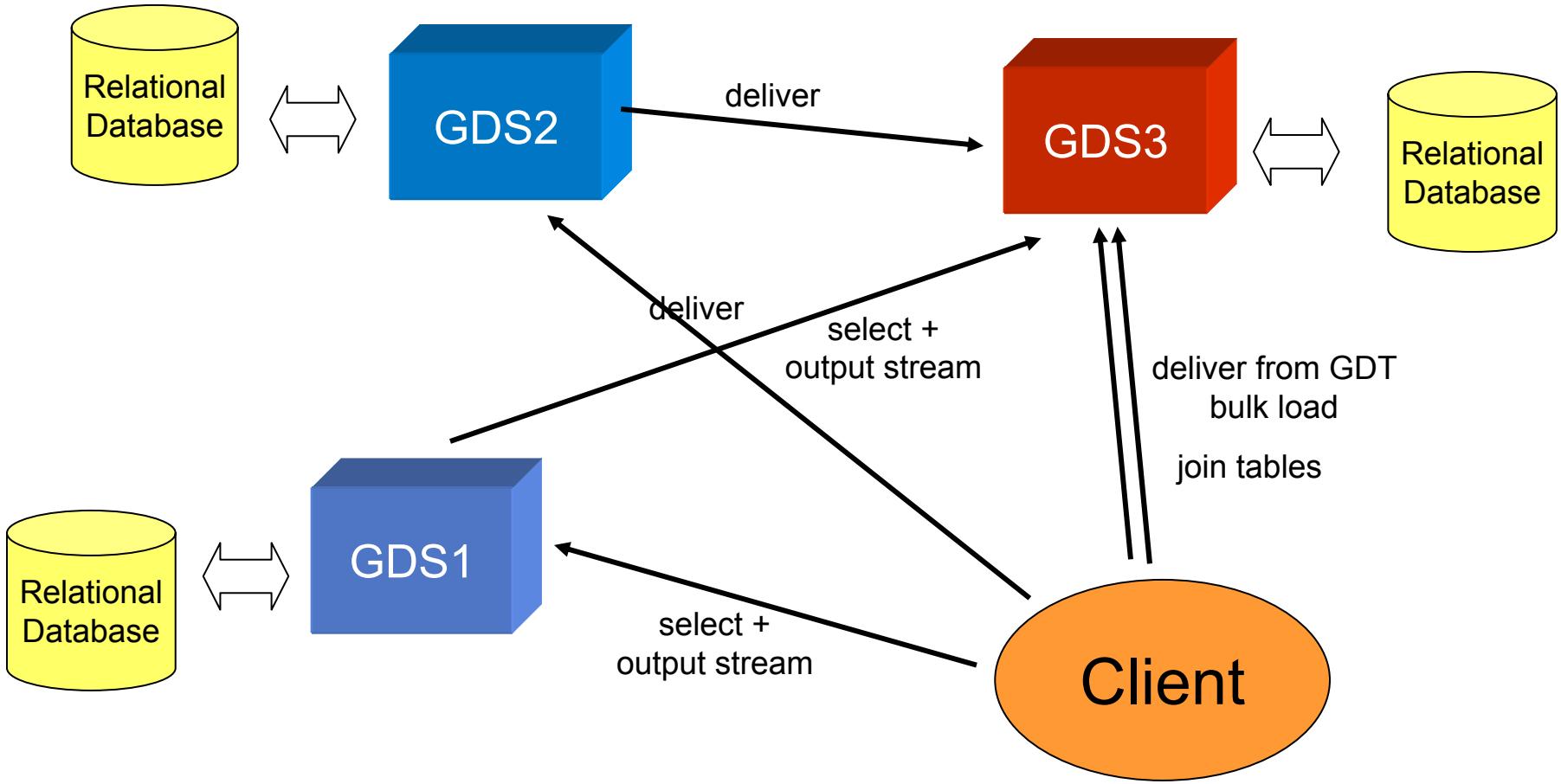
- The GDT port type allows to transfer data from one data service to another.
- An *InputStream* activity of GDS1 connects to a *DeliverToGDT* activity of GDS2
- Alternatively, an *OutputStream* activity can be connected to a *DeliverFromGDT* activity



Delivering Data

- Transfer in *blocks* or in *full*
- *InputStream* activities wait for data to arrive at their input
- Therefore, the *InputStream* activity at the sink has to be started before the *DeliverToGDT* activity at the source
- Same for *OutputStream* and *DeliverFromGDT*

Data Integration Scenario



Conclusion

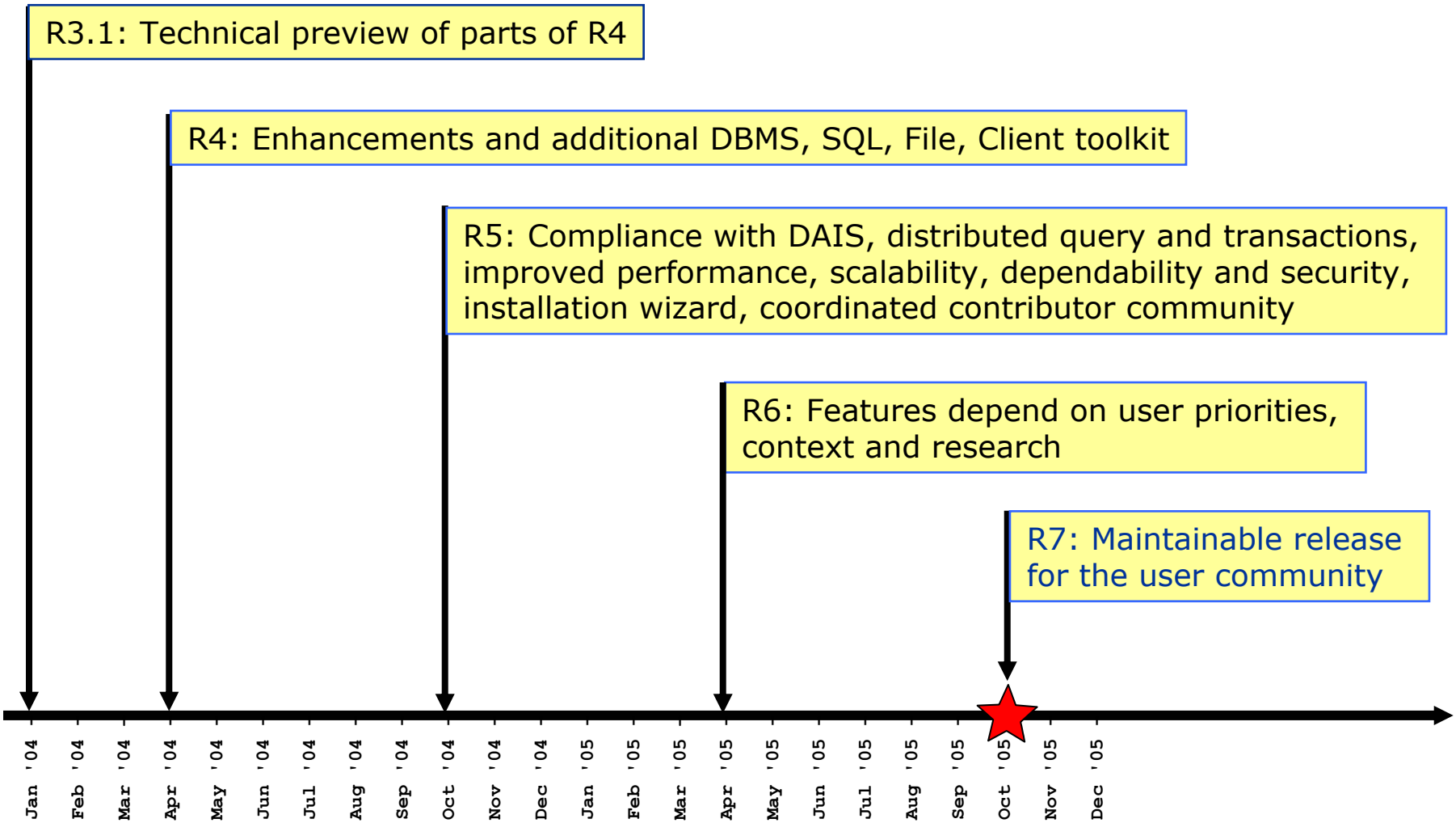
- Easy to use
  - No XML!
  - Less low-level APIs
  - improves usability and shortens learning curve for OGSA-DAI client development
- Protects developer
  - Shielded from schema changes, protocols, GT3
- Limitations
  - Metadata and service-data not addressed adequate
  - Higher-level abstraction possible (no factory)

OGSA-DAI Wrap-up

Overview

- Future Developments
- The OGSA-DAI Webpage
- Support Information
- Tutorials
- Links

Future Developments

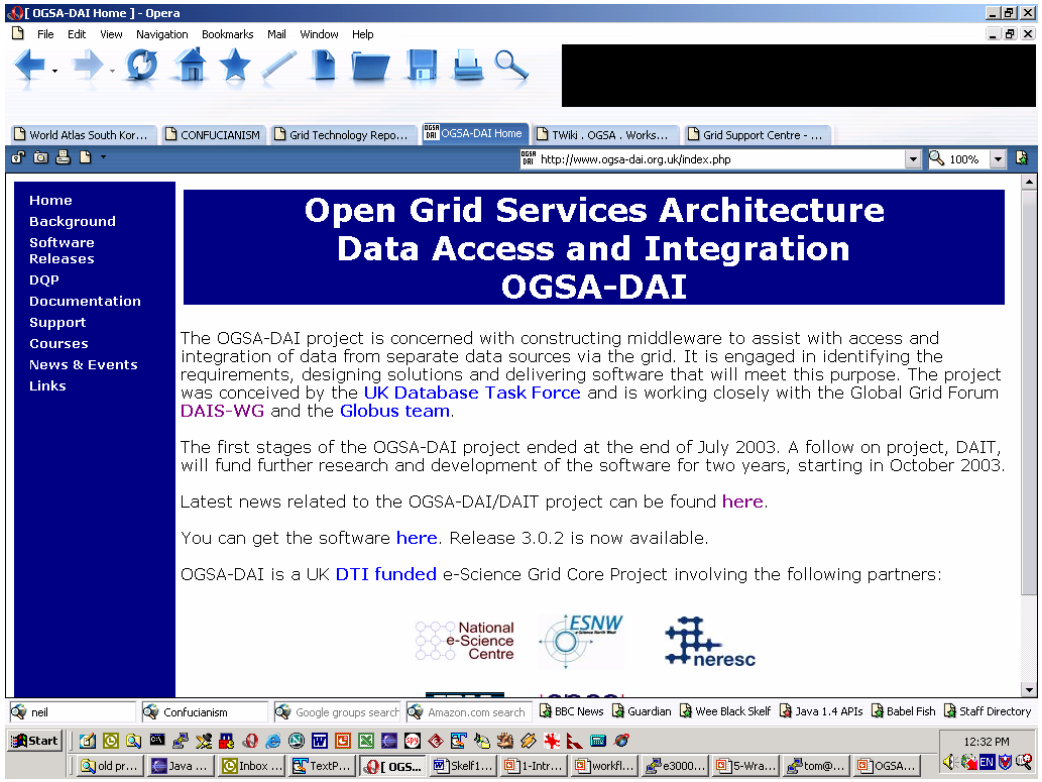


R5 → R7

- R5 October 04
  - Compliance with DAIS standards proposal
  - Distributed Relational Query Processing
  - Improved dependability and security integration
  - Extended & integrated XML and relational facilities
  - Distributed transaction participation
  - Coordinated OGSA-DAI contributor community
- R6 April 05
  - Integrated with GT4
  - New facilities depend on user priorities, context and research
  - OGSA-DAI components from contributor community
- R7 October 05
  - Maintainable release for the user community

OGSA-DAI Project Webpage

- http://www.ogsadai.org.uk



Background

News & Events

Software Releases

Documentation

Support

Training Courses

Links

Support

- Long term support for OGSA-DAI provided by UK Grid Support Centre
  - http://www.ogsadai.org.uk/support
  - support@ogsadai.org.uk
- Web forms for submission of
  - General queries
  - Problems with installation and configuration
  - Problems with usage of software
- Submissions are tracked and logged

FAQ and Mailing List

- Frequently Asked Questions
  - http://www.ogsadai.org.uk/support/faq.php
  - updated as common problems become clear
- Users mailing list
  - http://www.ogsadai.org.uk/support/list.php
  - general discussion of OGSA-DAI, data and the Grid
  - use support instead to report problems
- Suggestions for additions and improvements to support service welcome

Tutorials

- Graphical Demonstrator User Guide
- How to write an Activity Tutorial
- Using the Client Toolkit Tutorial

http://www.ogsadai.org.uk/docs/

Links

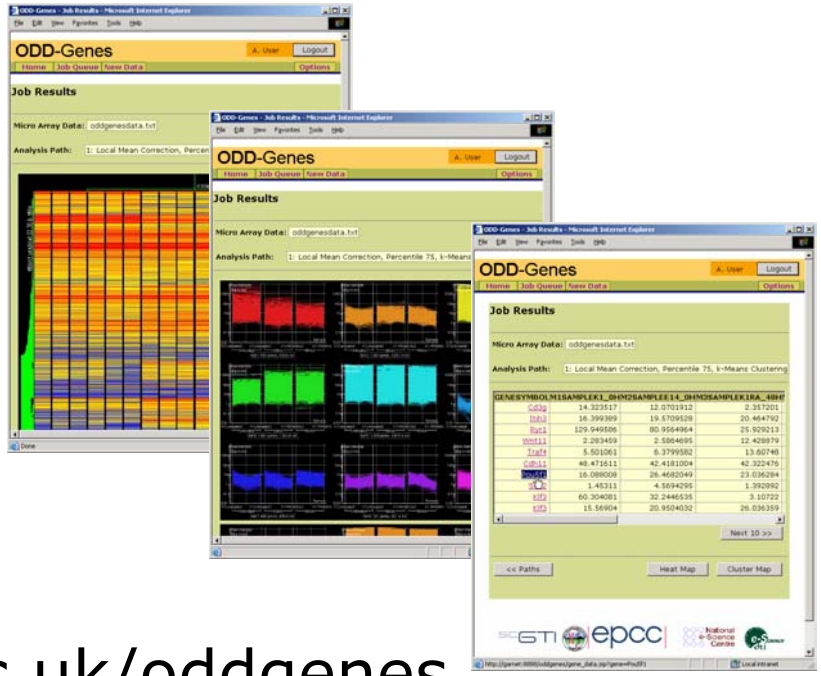
- OGSA-DAI Webpage
  - http://www.ogsadai.org.uk/
- Globus Toolkit 3
  - http://www.globus.org/ogsa
- Database Access and Integration Services (DAIS-WG)
  - http://www.gridforum.org/6_DATA/dais.htm
- Grid Technology Repository
  - http://gtr.globus.org
- ELDAS - Enterprise-Level Data Access Services (Eldas)
  - http://www.edikt.org/eldas
- Web Services Choreography
  - http://www.w3.org/2002/ws/chor

Projects using OGSA-DAI

- DQP - http://www.ogsadai.org.uk/dqp
  - Service Based Distributed Query Processor
- FirstDIG - http://www.epcc.ed.ac.uk/~firstdig
  - Data mining analysis of OGSA-DAI service-enabled data sources
- BIOGRID - http://www.biogrid.jp
  - Construction of a Supercomputer Network to meet IT needs for biology and medical science in Japan
- OGSA-WebDB - http://www.biogrid.jp
  - Provides a uniform view of heterogeneous database resources in a grid environment
- BioSimGrid - http://www.biosimgrid.org
  - A distributed database for biomolecular simulations
- More projects– http://www.ogsadai.org.uk/projects/

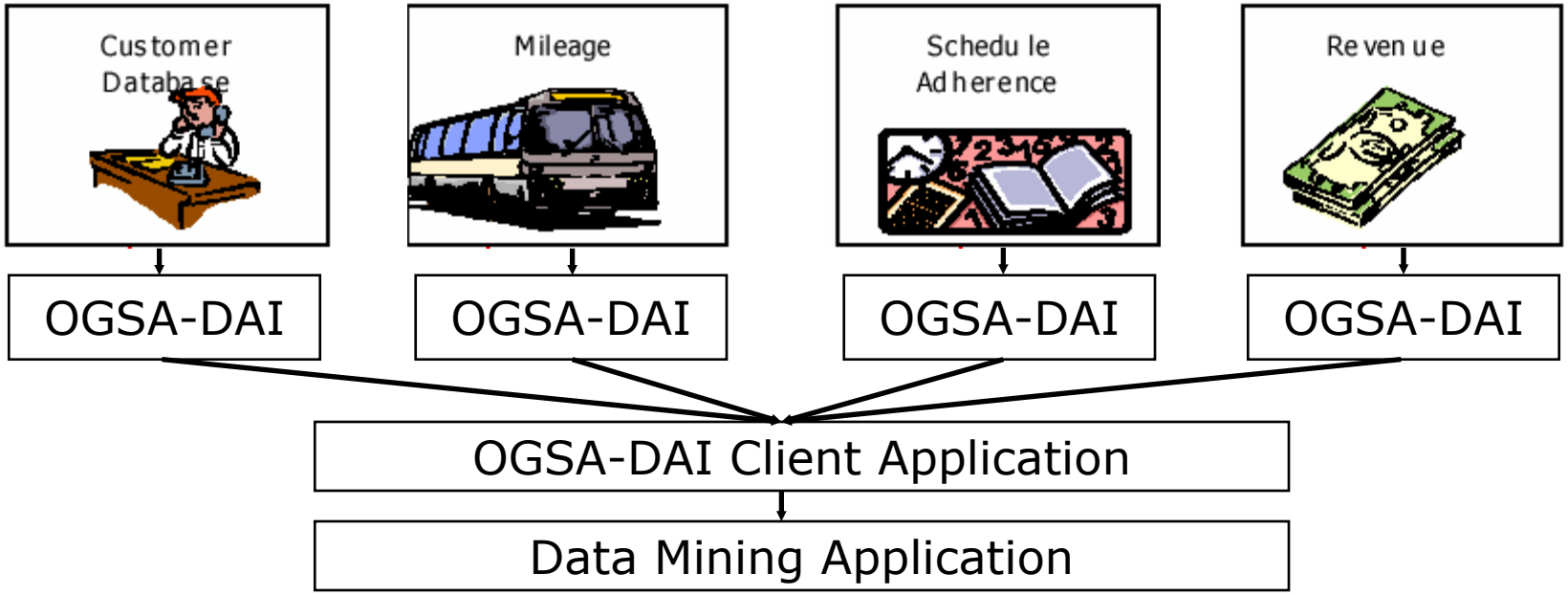
ODD-Genes

- Data Analysis for genetics
  - Sites:
    - GTI (microarray data)
    - HGU (genex data)
    - EPCC (compute server)
  - Software:
    - OGSA-DAI (Data)
    - TOG (Computation)
    - Globus Toolkit 2 and 3
  - http://www.epcc.ed.ac.uk/oddgenes



FirstDIG

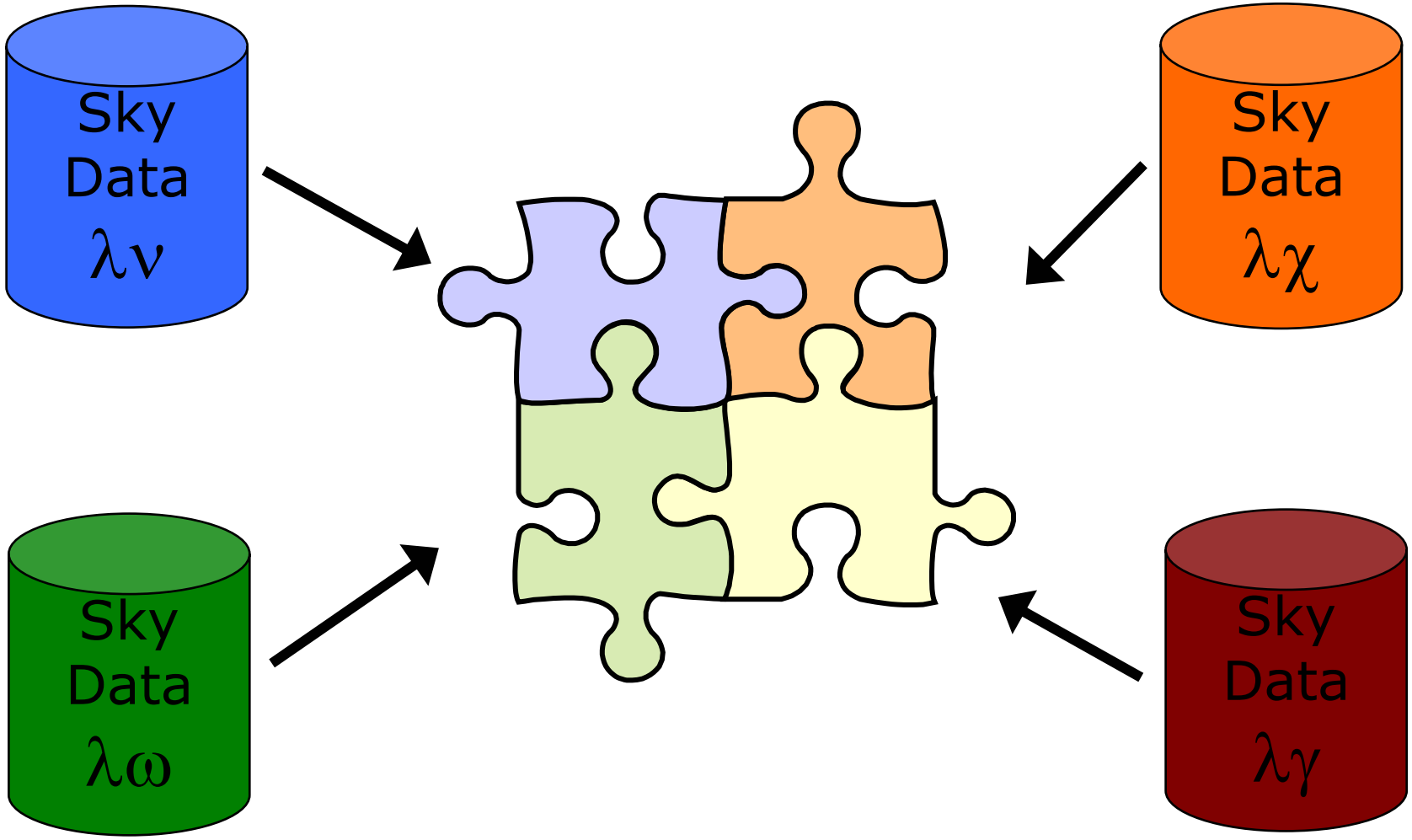
- Data mining with the First Transport Group, UK
  - Example: "When buses are more than 10 minutes late there is an 82% chance that revenue drops by at least 10%"
  - http://www.epcc.ed.ac.uk/firstdig



EdSkyQuery-G

- Collaboration between OGSA-DAI & Eldas
- Based on SkyQuery project by John Hopkins University, Baltimore, USA
- Identify astronomical objects and dropouts amongst different distributed catalogues
- Large scale data transport
- Plug-in algorithms
- Platform and DBMS independence

EdSkyQuery-G



EdSkyQuery-G Challenges

- Data formats
  - XML (WebRowSet)
  - CSV
  - Binary
  - Compressed CSV or XML
- Data transport
  - SOAP over HTTP/HTTPS
  - FTP, Secure-FTP, Grid-FTP
- Importing/Exporting data
  - Through services
  - Direct from stored procedures
  - Using native tools

SkyQuery.net

Main - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

EPCC Staff Directory | Java APIs | BBC News | Guardian Unlimited

http://www.skyquery.net/

SkyQuery.net

Home
Architecture
Algorithms
Web Services
Tutorial
Try me
Publications
Contact Us

SDSS SkyServer
VOServices.org

Surveys

ROSAT
GALEX
INTWFS
RC3
DLS
TWODF
TWOQZ
SDSS
HSTEXP
HDFN
HDFS
GOODS
UDF
TWOMASS
PSCZ
IRAS
NVSS
FIRST
AGC

Surveys ordered by wavelength. Click on a survey to see WS test.

SkyQuery.net

Table Info

Tables

Argument(if required):

Search

Select the radiobutton of a survey, the info category and optional parameter, then press Search to get info about tables and columns.

SkyQL query

```
SELECT o.objId, o.ra, o.type,
       t.objId, t.j_m, o.i
  FROM SDSS:PhotoPrimary o,
       TWOMASS:PhotoPrimary t
  WHERE XMATCH(o,t)<3.5
        AND AREA(182.358,-0.759,0.7)
        AND o.type=3 AND o.i<21 AND t.j_m<18
        AND (o.i-t.j_m)>2
```

Submit | 1 | 2 | 3 | 4

Sample queries from tutorial

o_objId	o_ra	o_type	t_objId	t_j_m	o_i	chisq	x_ra	x_dec	match
582093482743758885	182.357725113765	3	234676761	7.04	10.00888	0.0027	182.35773	-0.75849	1

Done

Start | Exceed | Inbox - Micro... | Main - Mozill... | TextPad - [Do... | Design for th... | skyquery.doc... | Microsoft Visi... | 12:48 PM

Conclusion

- Try out OGSA-DAI
  - It's free!
  - Supported
- Please send us feedback!
- Evolving and improving
  - Data integration
  - Performance and scalability
- Become involved
  - Write activities
  - Contribute to the DAIS working group

HPC-Europa

- EC-funded research visit programme
- Fully-funded, multi-disciplinary
- Visits between 3 and 13 weeks
  - EPCC in Edinburgh
  - CEPBA-CESCA in Barcelona/Catalonia
  - HLRS in Stuttgart
  - CINECA in Bologna
  - SARA in Amsterdam
  - IDRIS in Paris
- http://www.hpc-europa.com

OGSA-DAI Tutorial

- Introduction to data access and integration on the Grid using OGSA-DAI
  - Using the Data Browser
  - Writing Clients using the Client Toolkit APIs
- Start workstations in Windows mode
  - OGSA-DAI, Tomcat, MySQL and Xindice have already been configured

http://192.167.1.214:8080/tutorial