

# OGSA-DAI

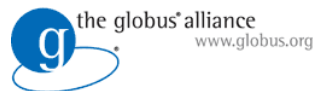
Accessing and integrating structured data using OGSA-DAI 3.0  
and links with WS-DAI

Mike Jackson and Elias Theocharopoulos  
OGSA-DAI project team, The University of Edinburgh



OGF-Europe

OGF-23



Web: [www.omii.ac.uk](http://www.omii.ac.uk)

Email: [info@omii.ac.uk](mailto:info@omii.ac.uk)

# Overview

- OGSA-DAI project
- Grids, collaboration and sharing data
- Data federation scenarios
- Possible solutions
- OGSA-DAI and data-centric workflows
- Realising the scenarios
- OGSA-DAI pros and cons
- Concealing workflows behind facades
- Standards, WS-DAI and OGSA-DAI
- Roadmap and summary

# OGSA-DAI project

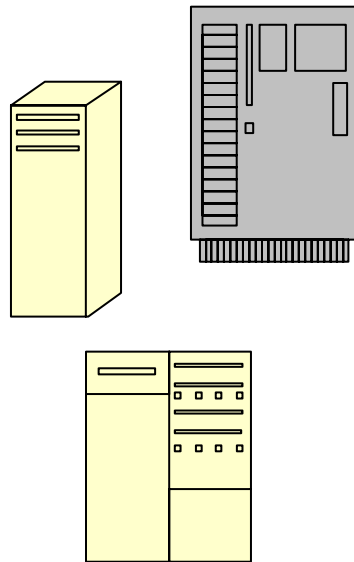
- February 2002
- Current partners
  - EPCC, The University of Edinburgh
  - National e-Science Centre, The University of Edinburgh
- OMII-UK
  - OMII, The University of Southampton
  - myGrid, The University of Manchester
  - OGSA-DAI, The University of Edinburgh
  - Funded by UK Engineering and Physical Sciences Research Council
- OMII-UK vision
  - Free open source software
  - Consultancy
  - Enable a sustained future for the UK e-Research community



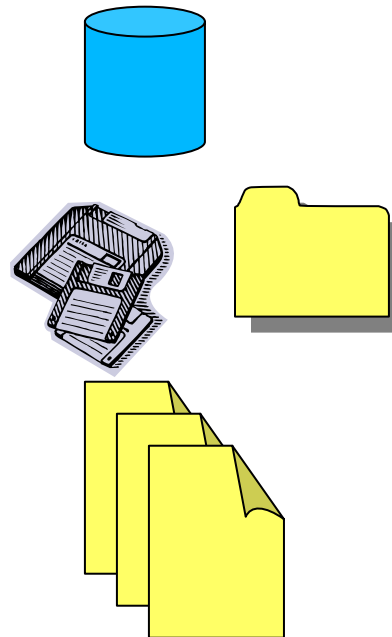
# Grids, collaboration and sharing data

# What is the Grid for?

Computational



Data

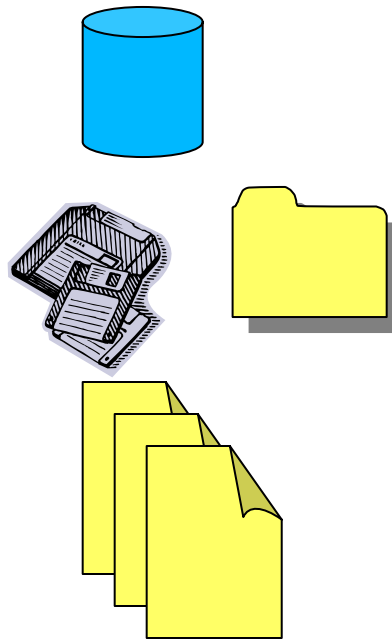


People



Sharing resources to enable collaboration

# What are we interested in?



Sharing **data** resources to enable collaboration

# Sharing data

- Share with whom?
  - World
  - Research community
  - Strategic partners or subsidiaries
  - Within a single project, organisation or business
- Open or closed communities
  - Scientific collaborations
  - Business partnerships

# What for?

- Identify, understand and exploit complex interactions between disparate variables
- Convert data into information
- Reveal new insights
  - Scientific knowledge
  - Business advantage

# Data mining

- Can better inform decision making
  - This is nothing new
- But fragmented data resources can hinder adoption
  - Distributed data resources
  - Myriad formats
  - Legacy data
- A wealth of public data increasingly available
  - House prices or crime rates by geographical area
  - Occupations according to gender or educational background
  - Occurrences of illnesses
  - ...
- Access and exploit what is within and without



# Data federation scenarios

# Heterogeneity – data models

Country	Capital
UK	London
France	Paris

```
<data>
  <columns>
    <column>Country</column>
    <column>Capital</column>
  </columns>
  <row>
    <field>UK</field>
    <field>London</field>
  </row>
  <row>
    <field>France</field>
    <field>Paris</field>
  </row>
</data>
```

```
FIELDS
Country,Capital
ENTRY=1
UK,London
ENTRY=2
France,Paris
```

CountryService

```
Country[] getCountries()
```

```
Capital getCapital(Country)
```

**(Country,Capital),(UK,London),(France,Paris)**

# Heterogeneity – products, languages, schema

- Database products
  - MySQL, Oracle, DB2, SQL Server, Postgres,...
  - eXist, Xindice,...
  - Persisted or in-memory
- Query languages
  - SQL-92, product-specific extensions,...
  - XPath, XQuery, XUpdate,...
- Database schema
  - Country, Pays, Nation, ...
  - Doctors.ID, Doctors.id, doctors.drID, Staff.NI, ...

## Scenario I – data with same schema distributed across multiple databases within an organisation

- Research organisation runs simulations in parallel for faster turn-around times
- Data partitioned and distributed across processing nodes
- Analyse the entire data set to understand current state

## Scenario II – data with different schema distributed across multiple databases within an organisation

- Transport company
- Data distributed across company sites
  - Customer contact
  - Vehicle mileage
  - Ticket revenues
  - Schedule adherence
- Combine and mine the data
- Yield information of commercial significance
  - How bus lateness affects revenue
  - How bus cancellations affects complaints

## Scenario III – data with different schema distributed across multiple databases within a group of strategic partners

- Public health providers share data on patients, illnesses and treatments
- Data distributed across the providers' sites
  - Same conceptual entities are named differently
  - e.g. PatientID, Id, PatientIdentifier, PatientNumber
- Combine and mine the data
- Yield early warning of infectious disease outbreaks

## Scenario IV – manage access to distributed data

- Manage access to public health data
- Within each partner
  - Doctors – patient names and health-related information
  - Accountants – patient names and billing/insurance information
  - Receptionists – patient names, addresses, doctors, appointments
- Between partners
  - Anonymised patient identifiers and health-related information
- Enforce access policies according to roles and rights of groups of users

# Scenario V – work with public data

- Social sciences researcher investigating correlations
  - House prices and crime rates
  - House prices and school leaving ages
  - School leaving ages and crime rates
- Visualisation
  - Combine with public data sets representing geographical areas
  - e.g. local authority areas or electoral boundaries
- Similar to scenario III

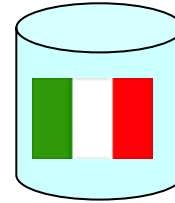
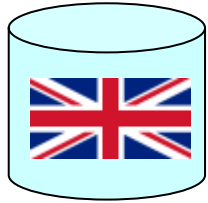
## Scenario VI – work with private and public data

- Transport company
- Combine and mine public and private data
- Understand correlations
  - House prices and complaints
  - House prices and subway line profitability
  - Crime rates in an area and assaults on subway staff at stations in those areas
- Similar to scenario III + V

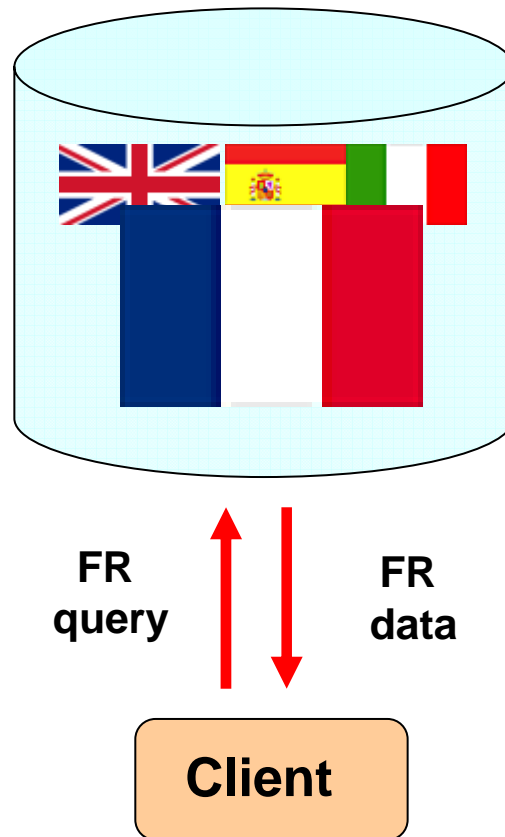


# Possible solutions

# Distributed data resources



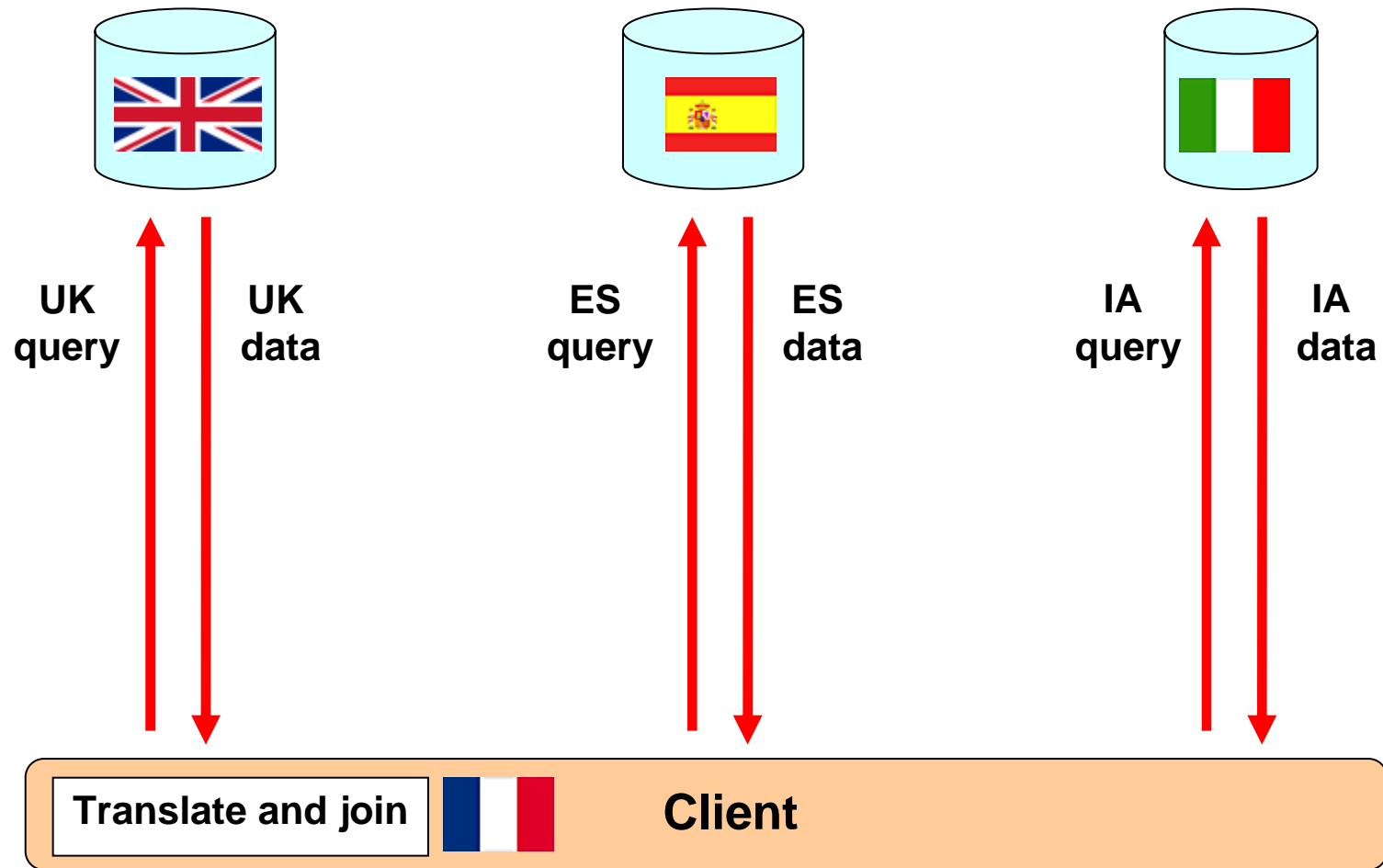
# How about a central server?



# Central server pros and cons

- Direct access
- Single point of access
- Data in common format
- Database can handle joins
  
- Initial overhead in terms of time, effort and cost
- Keeping data up to date
- Loss of control by data providers
  - Assuming they even let go
- Security and trust

# How about providing direct access?



# Direct access pros and cons

- Clients have direct access
- Data is always up-to-date
- Data providers retain control
  
- Heavyweight clients
- Heterogeneity
  - Databases, data models, query languages
  - Security protocols, authorization and authentication, ports and firewalls
- Security overheads for data providers
  - Manage firewalls and usernames/passwords for multiple clients
- Hard to use in grid/web service workflows

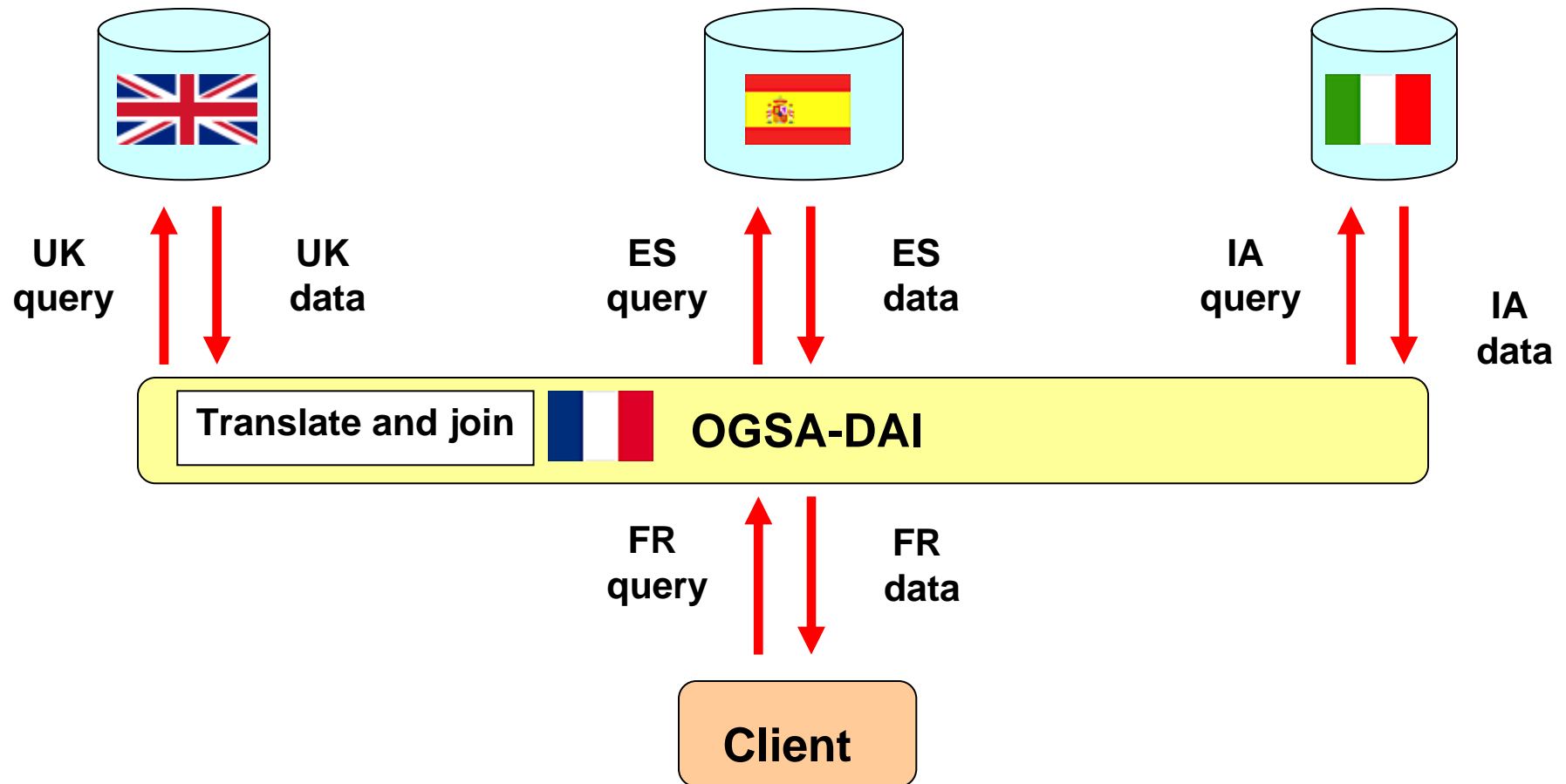


# OGSA-DAI and data-centric workflows

# OGSA-DAI's vision

- Sharing data resources to enable **collaboration**
- Data **access**
  - Structured data in distributed heterogeneous data resources
- Data **transformation**
  - e.g. expose data in schema X to users as data in schema Y
- Data **integration**
  - e.g. expose multiple databases to users as a single virtual database
- Data **delivery**
  - To where it's needed by the most appropriate means
  - e.g. web service, e-mail, HTTP, FTP, GridFTP

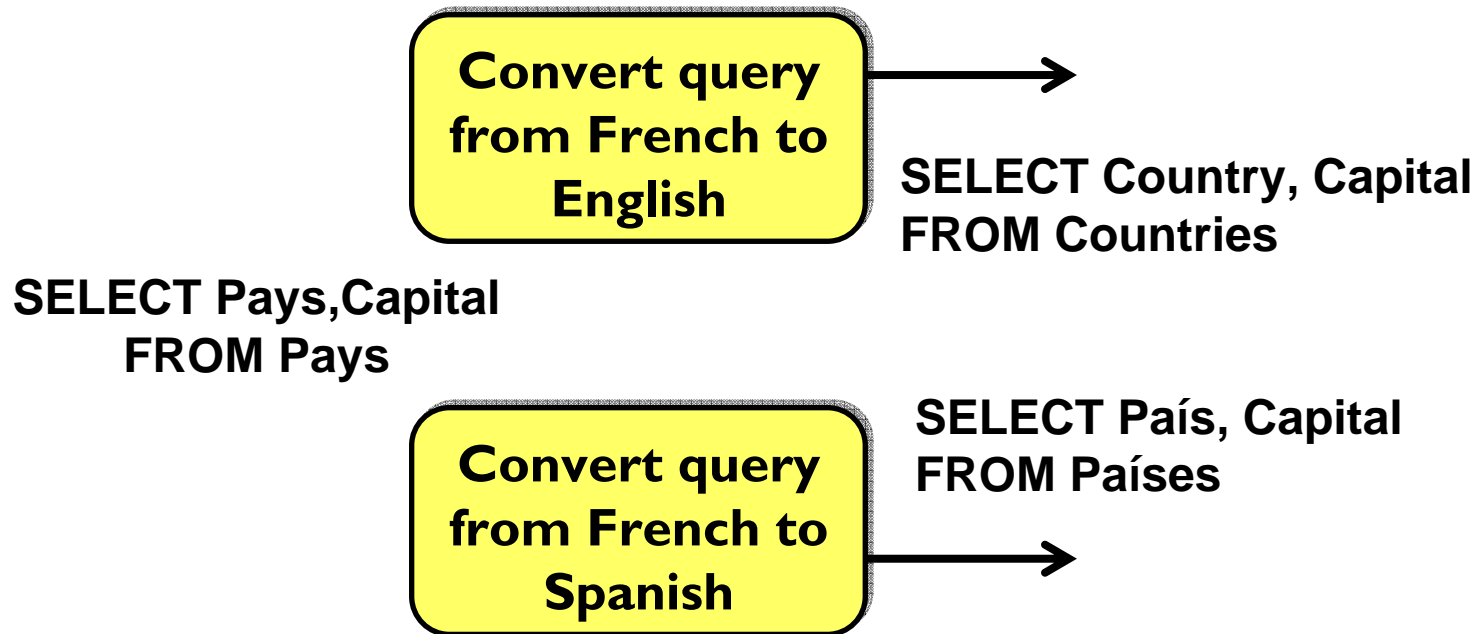
# Sharing distributed heterogeneous resources with OGSA-DAI



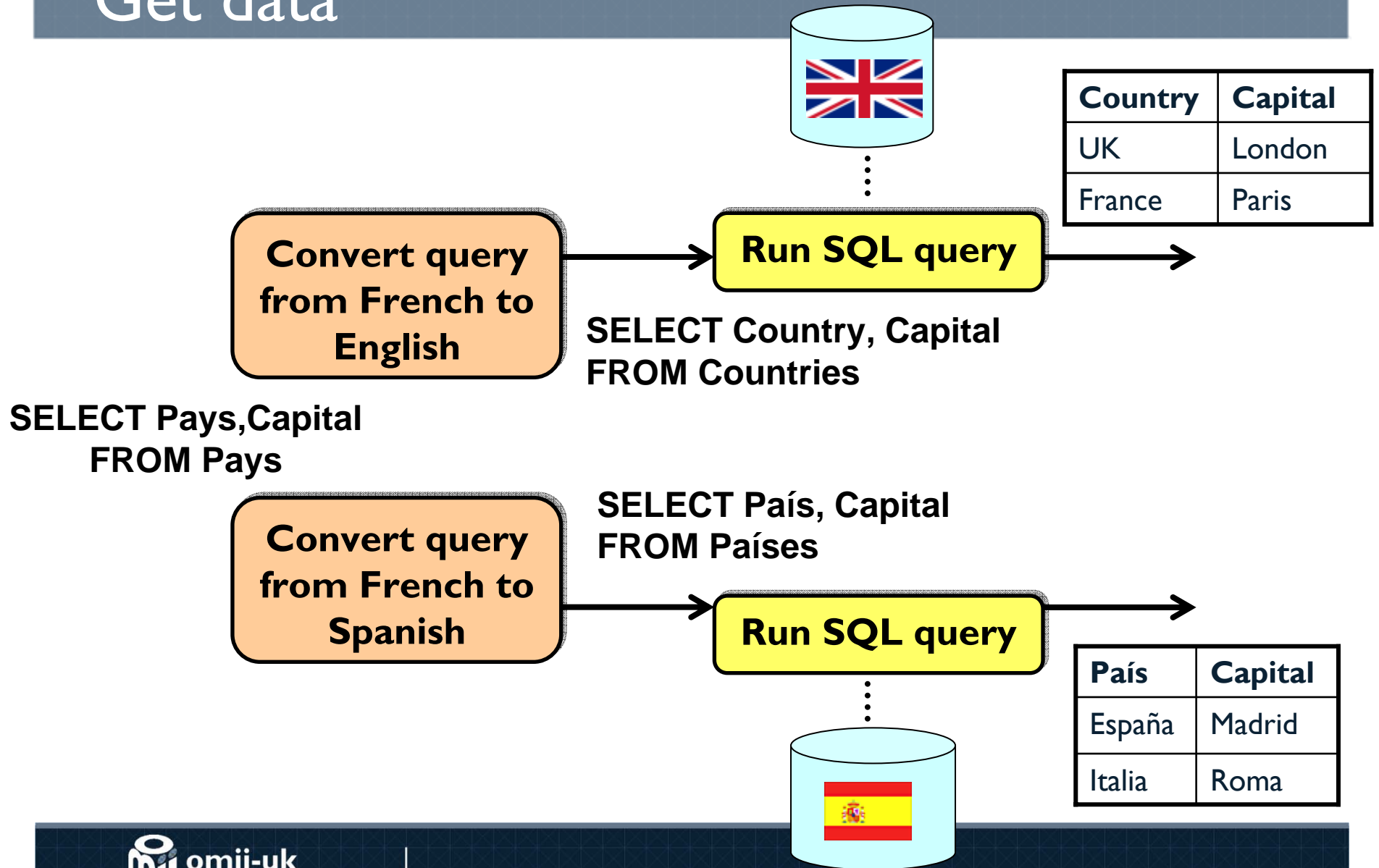
# What OGSA-DAI does

- Executes **workflows**
  - Equivalent to programs or scripts
- Workflows contain **activities**
  - Well-defined functional units
  - Data goes in, something is done, data comes out
  - Equivalent to programming language methods
- Workflows are submitted by clients
  - To an OGSA-DAI web service

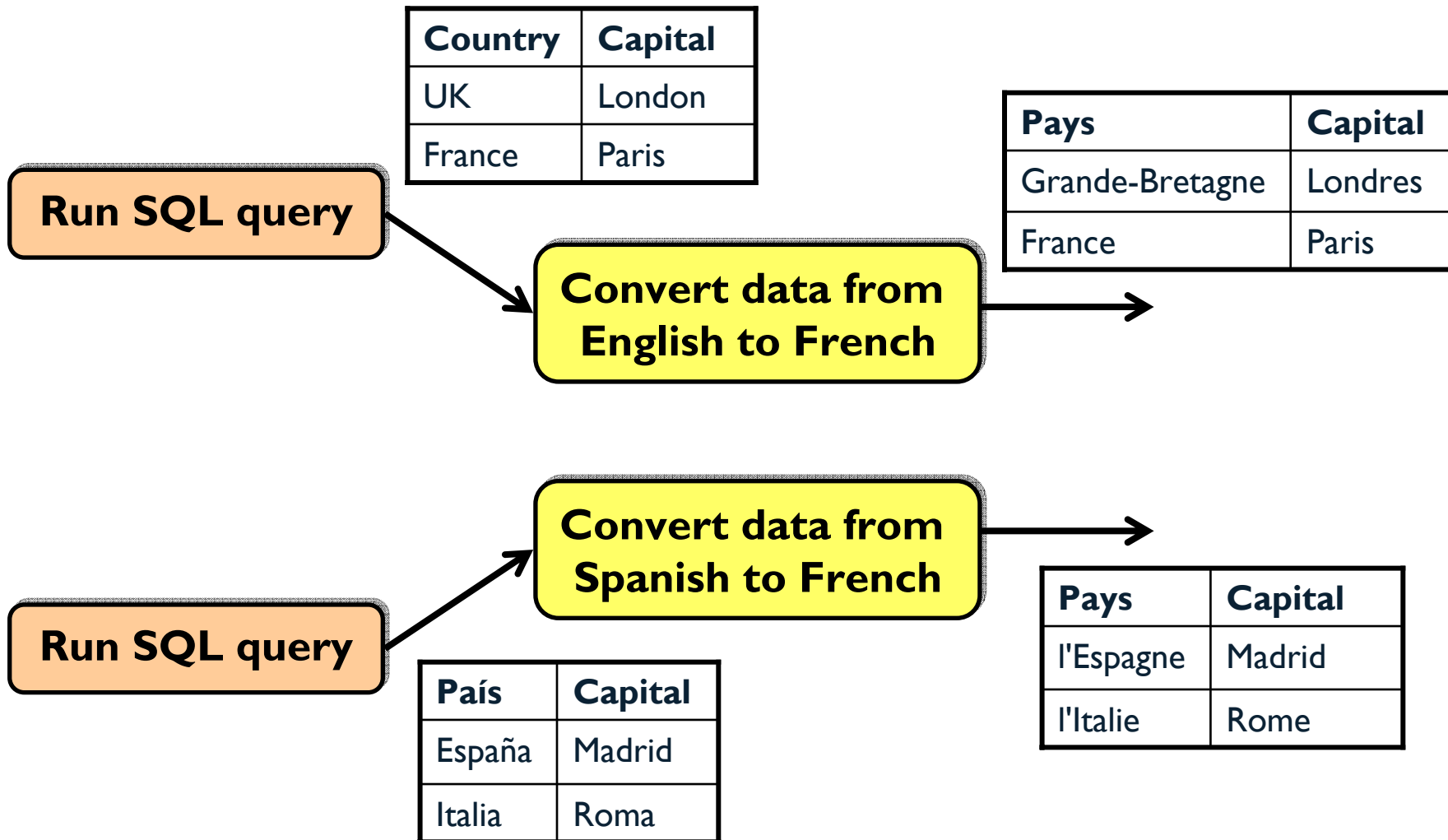
# Transform queries from a common language



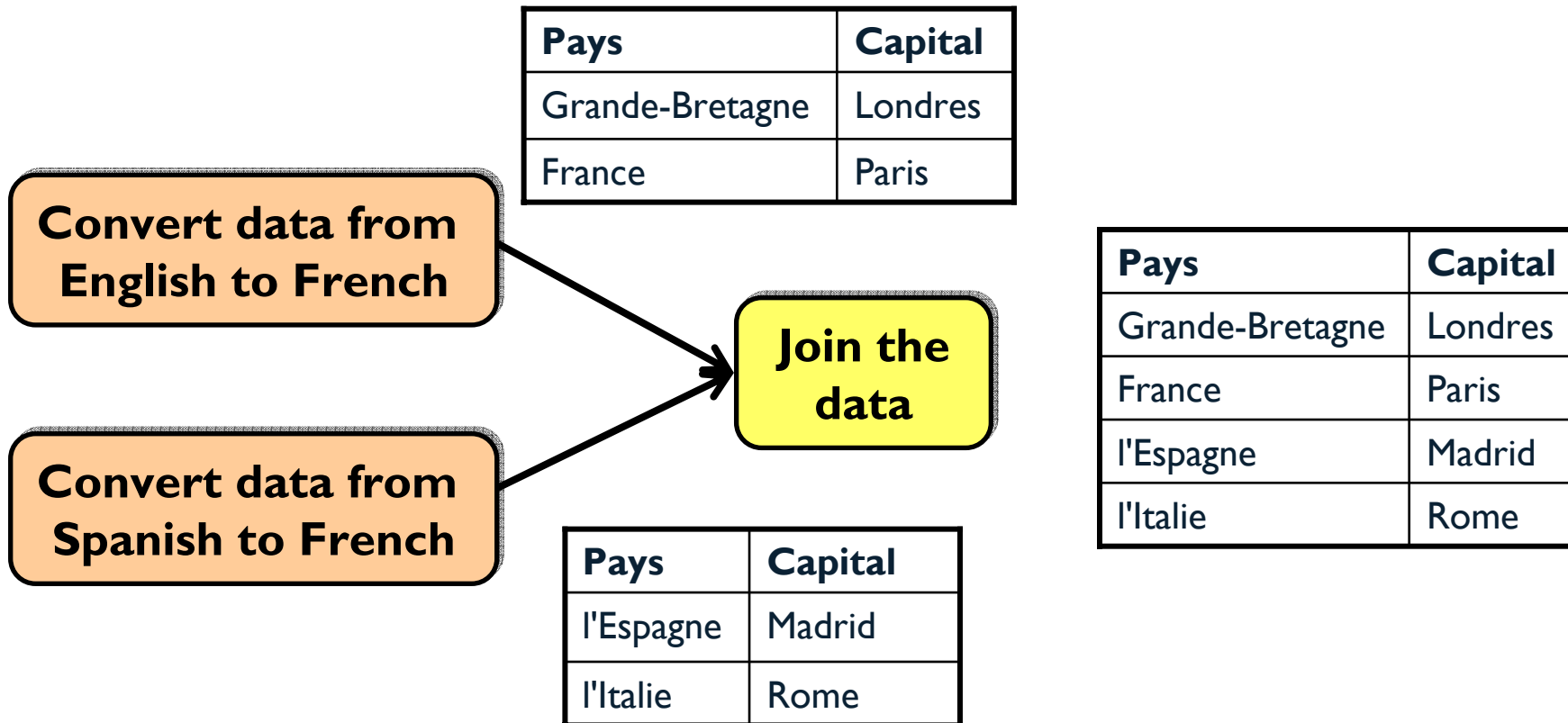
# Get data



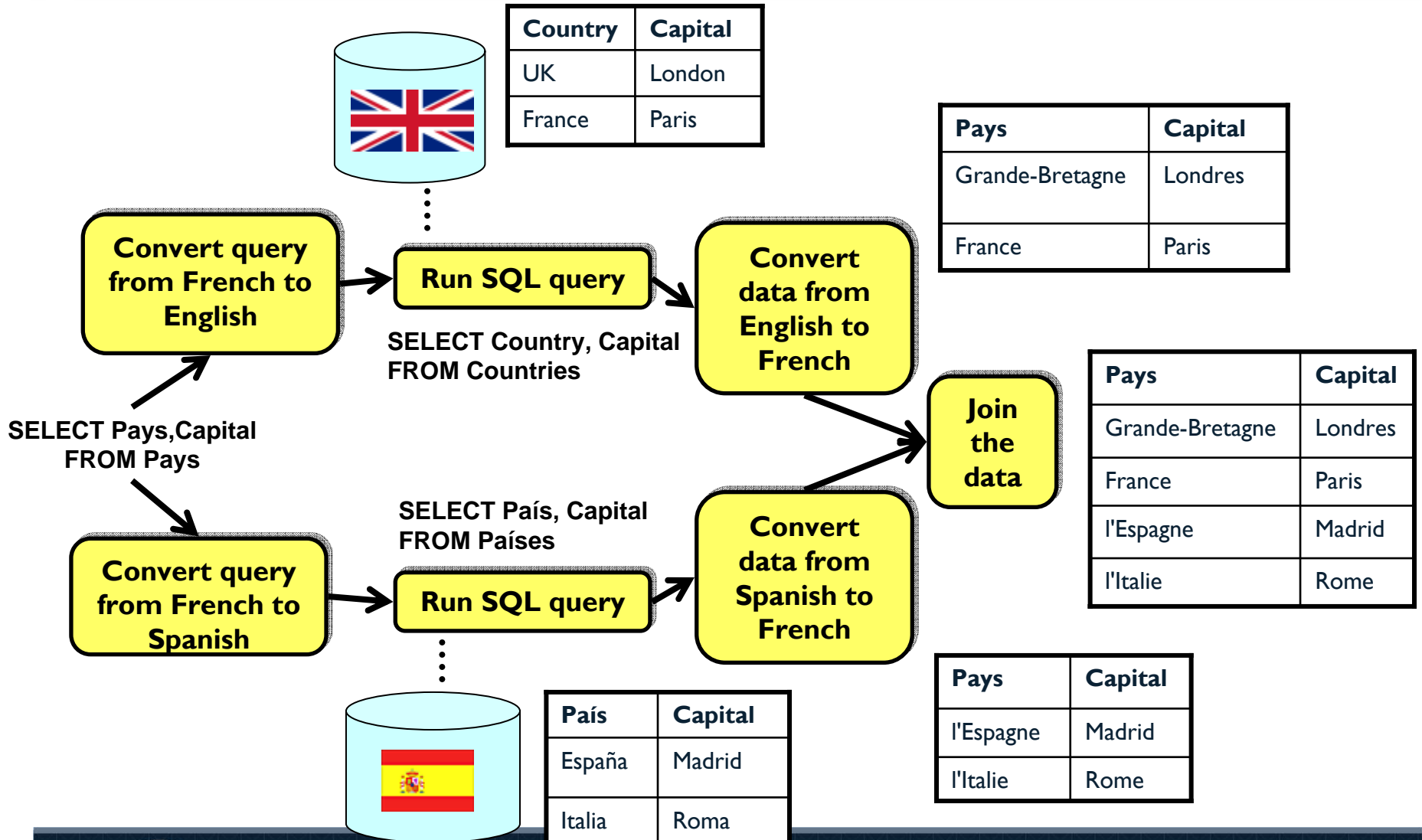
# Transform data into a common schema



# Join data in some way



# Putting it all together



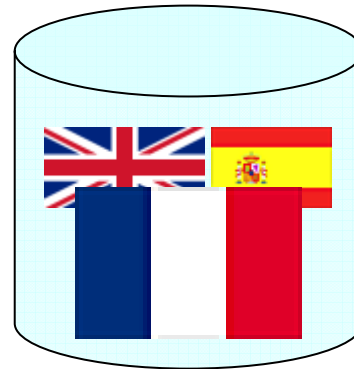
Country	Capital
UK	London
France	Paris

Pays	Capital
Grande-Bretagne	Londres
France	Paris

Pays	Capital
Grande-Bretagne	Londres
France	Paris
l'Espagne	Madrid
l'Italie	Rome

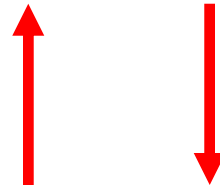
Pays	Capital
l'Espagne	Madrid
l'Italie	Rome

# How it appears to the client



**OGSA-DAI**

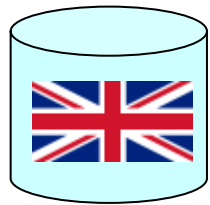
**workflow(SELECT  
Pays,Capital FROM  
Pays)**



**Client**

<b>Pays</b>	<b>Capital</b>
Grande-Bretagne	Londres
France	Paris
l'Espagne	Madrid
l'Italie	Rome

# A query-transform-update example



Run SQL query

Country	Capital
Spain	Madrid
Italy	Rome

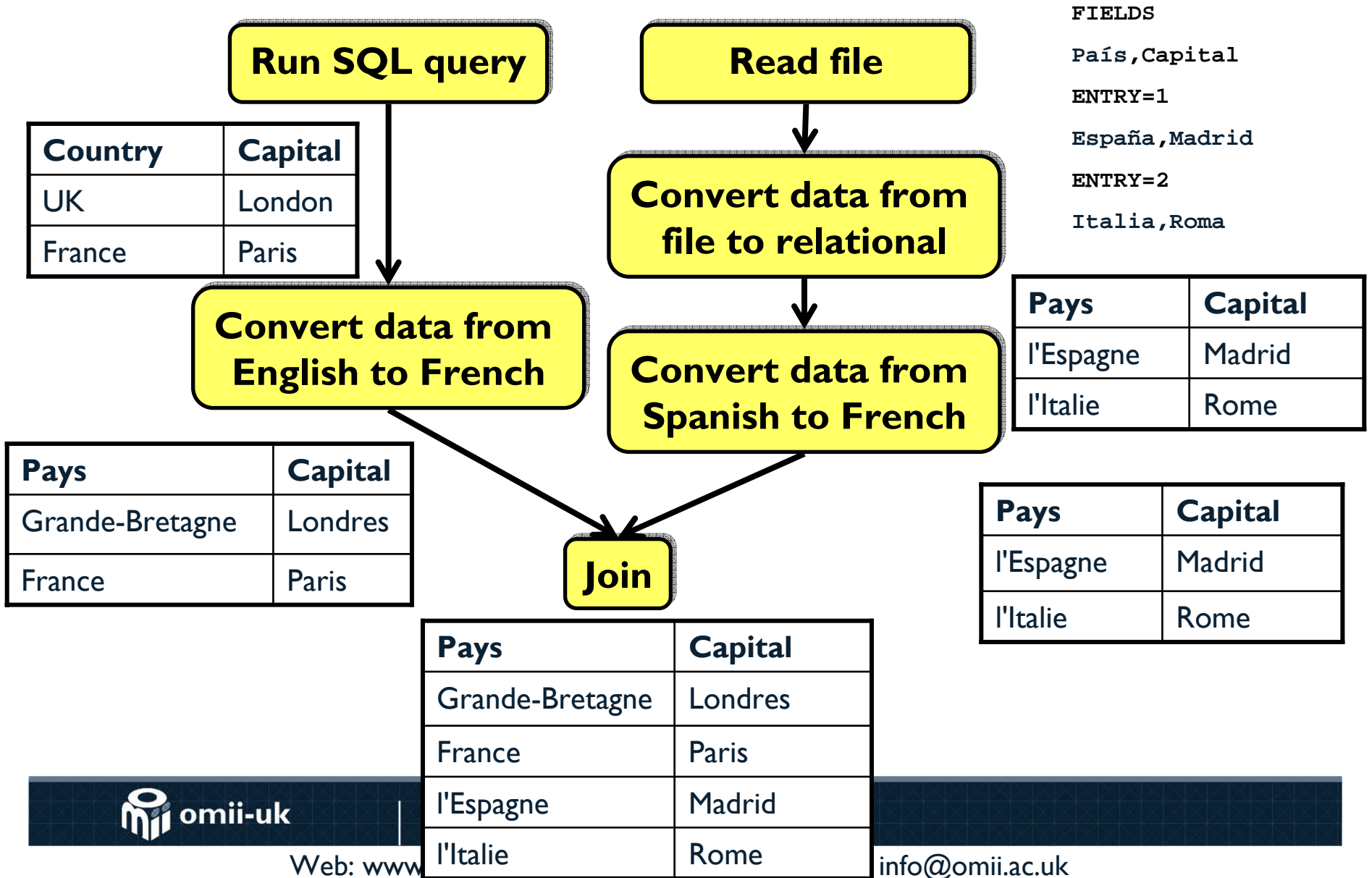
Convert data from  
English to Spanish

Run SQL update

País	Capital
España	Madrid
Italia	Roma



# A query-transform-join example



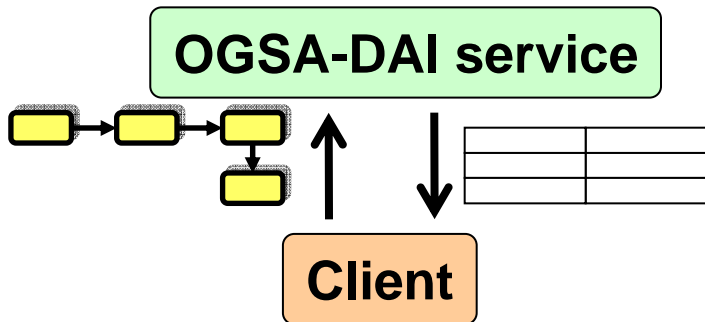
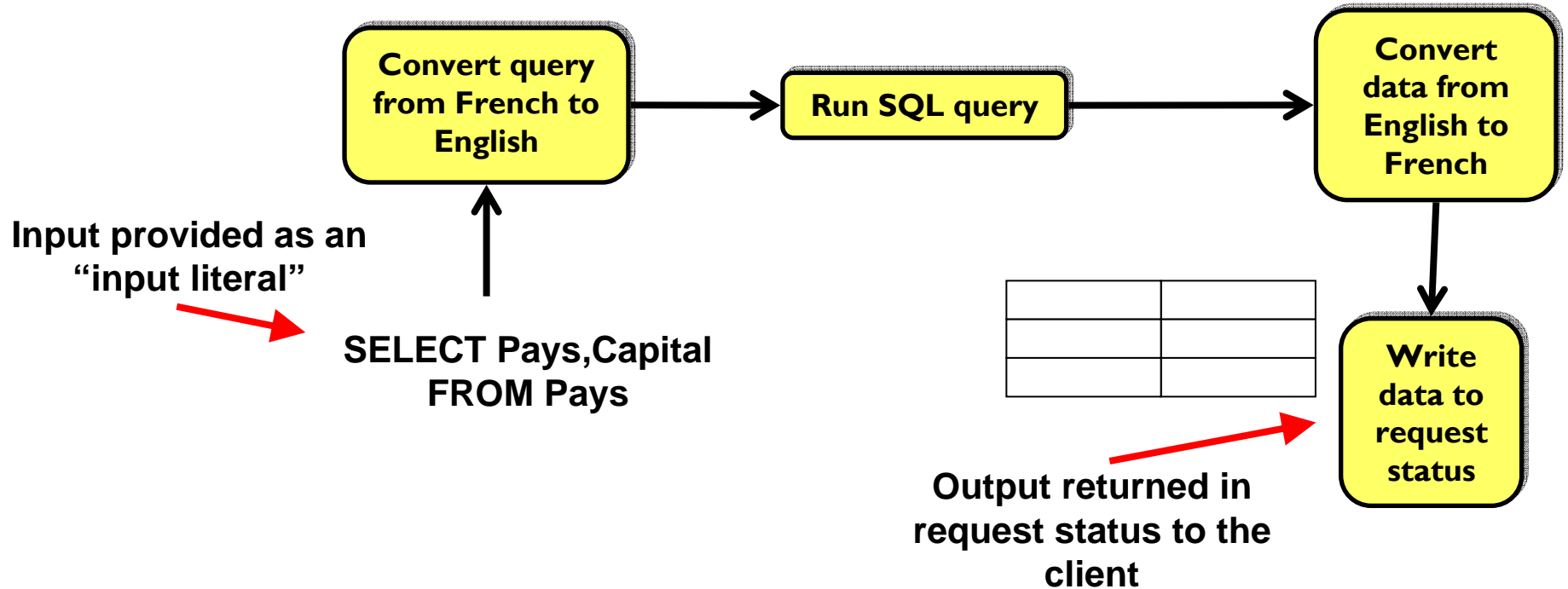
# Types of workflows

- Pipeline workflow
  - A set of chained activities executed in parallel with data flowing between the activities
- Sequence workflow
  - A set of sub-workflows each executed in sequence
  - For example
    - Sub-workflow 1 – create a database table
    - Sub-workflow 2 – bulk load data into the table
- Parallel workflow
  - A set of sub-workflows executed in parallel

# What is an OGSA-DAI web service?

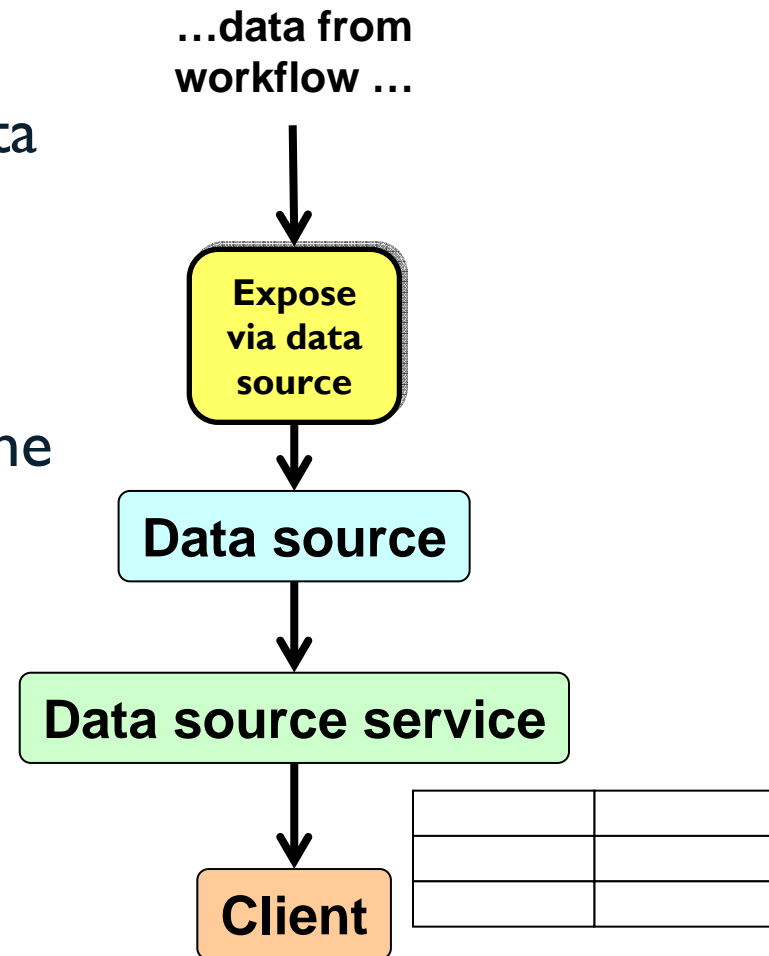
- A web service that accepts an OGSA-DAI workflow
- Passes the workflow to the OGSA-DAI **data request execution resource**
  - Manages workflow execution
- Synchronous request
  - Wait for workflow to complete
  - Return status and, if requested, data
- Asynchronous request
  - Start workflow
  - Return current status
  - Client can poll status and stream back data

# Synchronous delivery

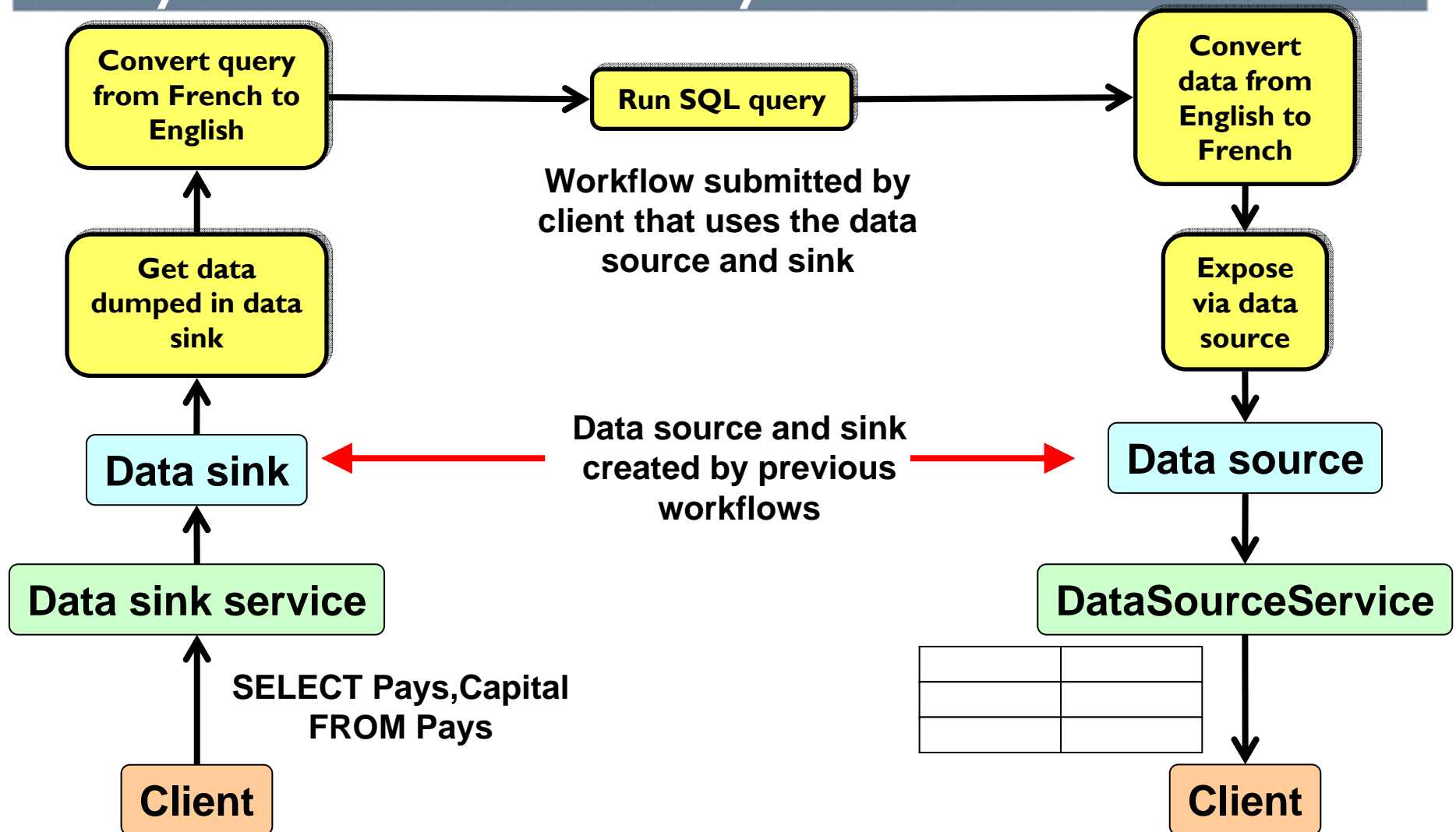


# Asynchronous delivery and data sources

- **OGSA-DAI data sources**
  - Resource for asynchronous data delivery
- **Data source service**
  - Web service
  - Get a block, N blocks or all the data
- **Scalable**
  - 1 million rows
  - OR
  - 1000 x 1000 rows



# Asynchronous delivery



## But ... a web service?

- SOAP/HTTP/XML can be a killer
- Other delivery methods
  - SOAP attachments
  - E-mail address
  - FTP server
    - Push data to FTP server on client
    - OR
    - Client can pull data from FTP server
  - GridFTP server
    - Designed for efficient data transport

# Another delivery example



**Run SQL query**

**Convert data into XML document**

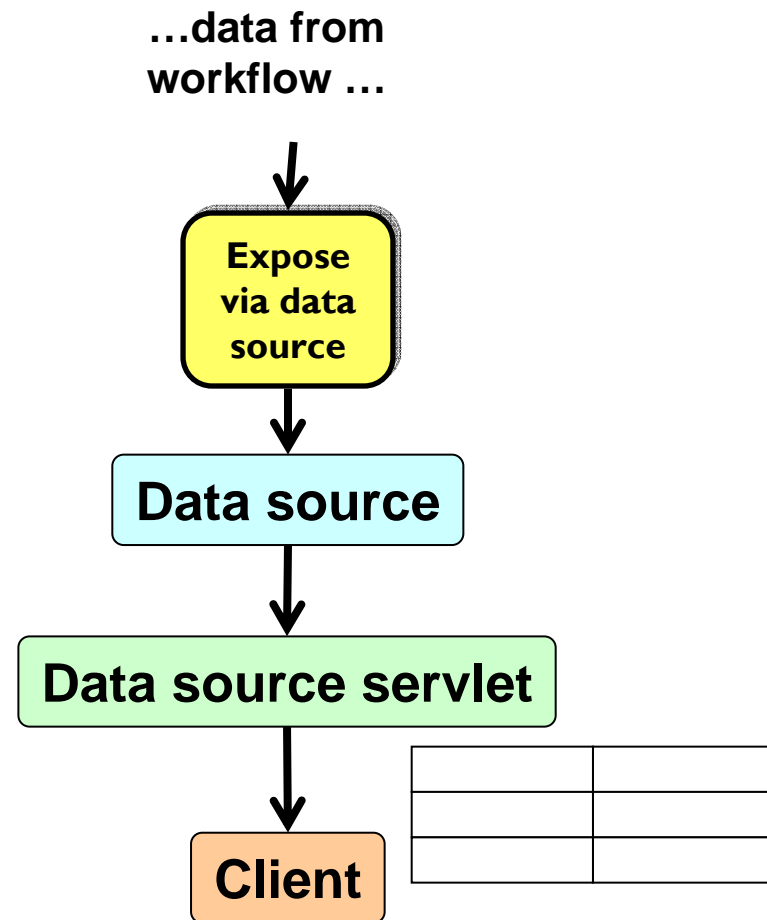
**Deliver to michaelj@epcc.ed.ac.uk**

Country	Capital
UK	London
France	Paris

```
<data>
  <columns>
    <column>Country</column>
    <column>Capital</column>
  </columns>
  <row>
    <field>UK</field>
    <field>London</field>
  </row>
  <row>
    <field>France</field>
    <field>Paris</field>
  </row>
</data>
```

# Asynchronous delivery and servlets

- Data source servlet
  - Invoke HTTP GET
  - Use URL query string to specify data source ID
- Useful in service orchestration and job submission
  - Taverna
  - GridSAM



# Other access technologies

- Web services are not required to use OGSA-DAI
  - Presentation layers are an extensibility point
- Could develop alternative presentation layers
  - Direct access layer
    - Applications interact with the OGSA-DAI core directly
  - Java RMI
  - Sockets
  - ...

# Security

- Authentication and authorization
- Access to
  - Service
  - Operation
  - Resource
  - Activity
  - Database
  - Table, column, file, XML document, XML element,...
- Read-only or read-write
- Role-based access

# Security – let the users decide

- Not our place to decide what users should use
  - Don't want to force anything on anyone
  - Extensibility points and examples only
- Security context
  - Container for anything security-related – just an interface
  - Populated by presentation layers
  - Passed around the OGSA-DAI core components

# Security context and need-to-know

- Components that need to know can interrogate the context
  - Do you have the client's distinguished name?
  - Do you have VOMS attribute for the client?
  - ...
- What components might need-to-know?
  - Activities
  - Data resources
  - Other application-specific components.,...

# Security and presentation layers

- Populate security context
- Authenticate and authorize as far as possible
  - Service, resource, activity
- Depends upon
  - Hosting environment
  - Security infrastructure
  - OGSA-DAI version's presentation layer
    - Axis/Tomcat, Globus Toolkit, ...
  - OGSA-DAI service provider

# Example – OGSA-DAI and Globus

- Policy Information Points
  - Gather information about caller and request
- Policy Decision Points
  - Use this information to authorize request
- PIPs
  - DN PIP – get DN of caller
  - Resource ID PIP – get ID of target resource
  - Workflow Resource IDs PIP – get IDs of all resources targeted by activities in a workflow
- PDPs
  - Resource Authorizer PDP – is the DN allowed to access all the resources identified in the PIPs?
    - Consults in-memory record of DNs and the resources they've created

# Example – OGSA-DAI and VOMS

- OMII-Europe VOMS plug-ins for OGSA-DAI
- Another set of Globus Toolkit PIPs and PDPs
- e.g. get user information from VOMS
- e.g. authorize access to resource based on attributes



# Security and database access

- Map security context information to database logins
- Data resource plug-in specific
  - Data resource plug-ins are customisable
- Default OGSA-DAI resources use login providers
  - Login providers are customisable
- Example – DN-based login provider
  - Maps caller DN to database user name and password
  - Mappings are stored in a text file on the server
    - Could change login provider to use a database
- Example – VOMS login provider
  - Map VOMS attributes to user name and password

# Extending OGSA-DAI

- OGSA-DAI is a framework
- Typically customised for application-specific requirements
- Extensibility points
  - Functionality – activities
  - Data – data resource plug-ins
  - Security
    - Database logins – login providers
    - Access to resources – resource authorizers
    - Access to activities – activity authorizers
    - Presentation layer security information – security contexts
  - Presentation layers



# Increasing OGSA-DAI's power

# SQL views

- Define a drPatient view to be
  - `SELECT p.id, p.name, p.age, p.sex FROM patient p, doctor d WHERE p.DrID = d.ID;`

ID	Name	Age	Sex	ZIP	Dr ID
1	Ken	42	M	IL1478305	456
2	Josie	25	F	BNI 7QP	789

ID	Name	DN
123	Greene	US-Chicago-G
456	Ross	US-Chicago-R
789	Fairhead	UK-Holby-F

- Client runs `SELECT * FROM drPatient;`
- Shorthand for complex query results
- Data access control e.g. users of drPatient
  - Cannot access a patient's ZIP
  - Are unaware of the doctor or patient tables

# OGSA-DAI SQL views

- An implementation of views in OGSA-DAI
  - Parses query
  - Maps view to SQL query over actual database
- Clients are unaware of the multiple tables
- Can define views for read-only databases
- Schema transformation
  - Map a logical schema to a physical schema

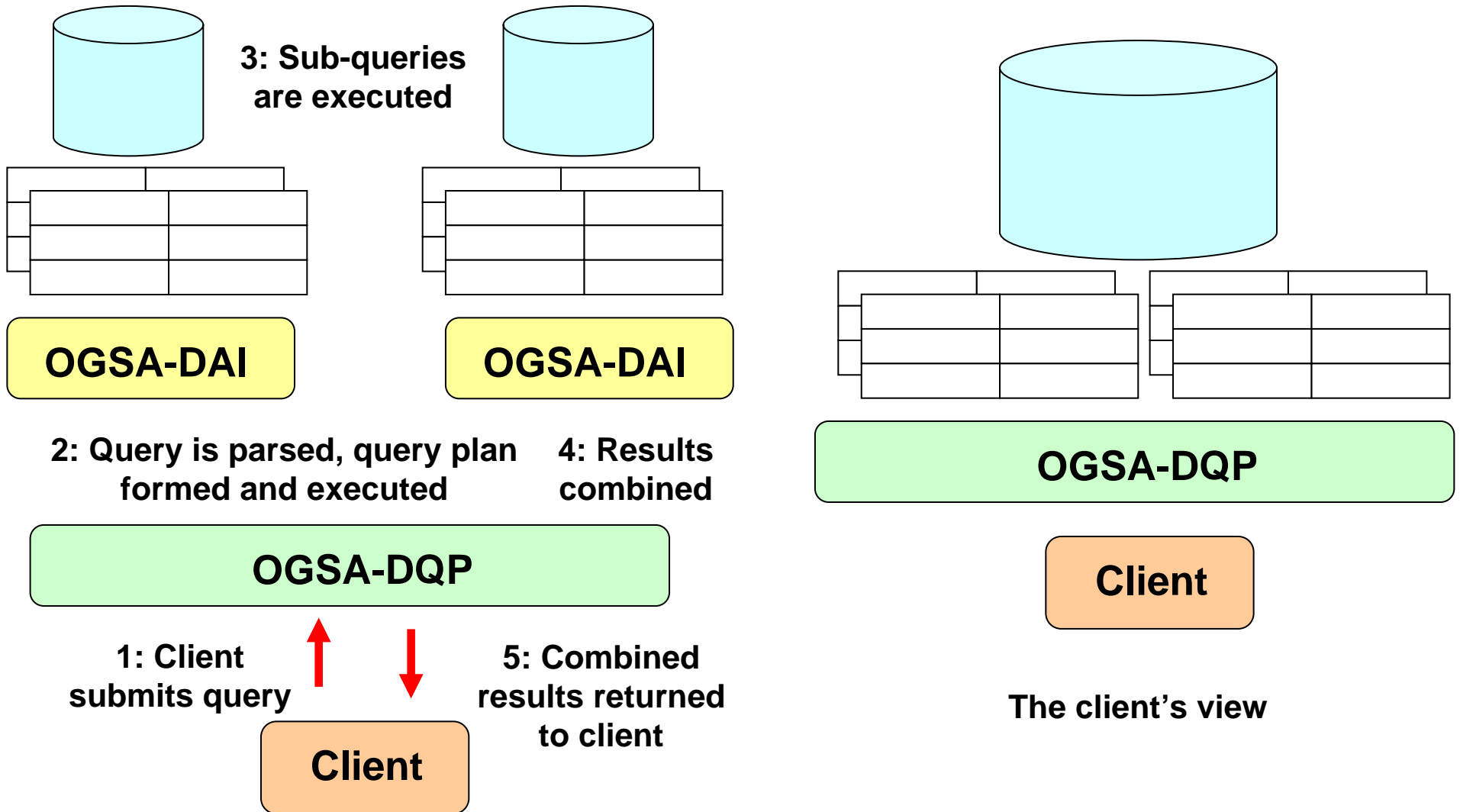
# OGSA-DAI SQL views and security

- Factor in client's security credentials
- e.g if drPatient was defined as
  - `SELECT p.id, p.name, p.age, p.sex FROM patient p, doctor d WHERE p.DrID = d.ID AND d.dn = $DN$;`
  - Replace `$DN$` by client's DN provided by Grid security components
  - Doctors can only view their own patients

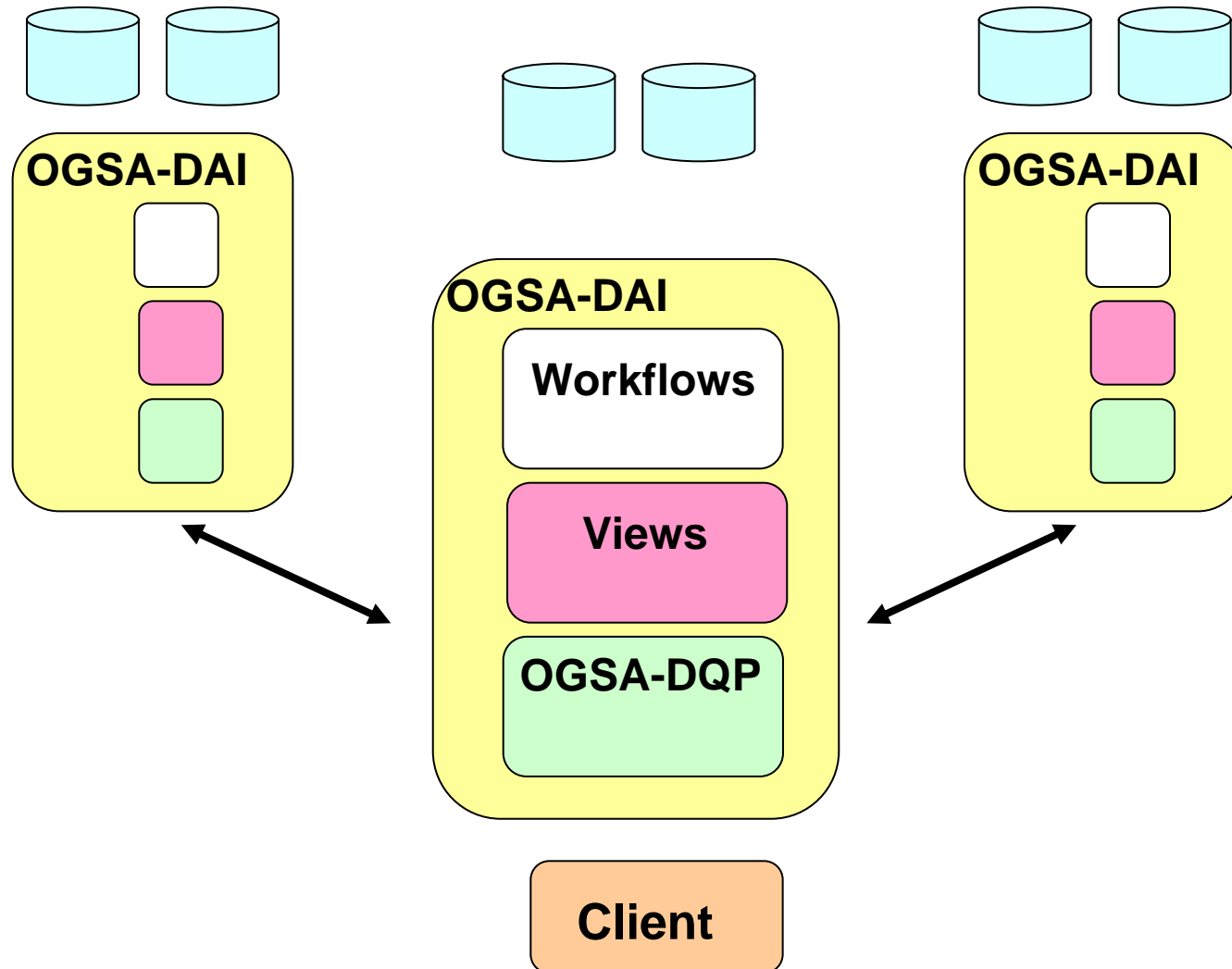
# OGSA-DQP

- OGSA-DQP utilises OGSA-DAI
  - Developed by Universities of Manchester and Newcastle
  - Rewritten for OGSA-DAI 3.0 by EPCC as part of the NextGrid project
- Distributed query processing
  - Multiple tables on multiple databases are exposed to clients as multiple tables in one “virtual database”
  - Databases can be exposed
    - EITHER within one OGSA-DAI server
    - OR via multiple remote OGSA-DAI servers
- Clients are unaware of the multiple databases

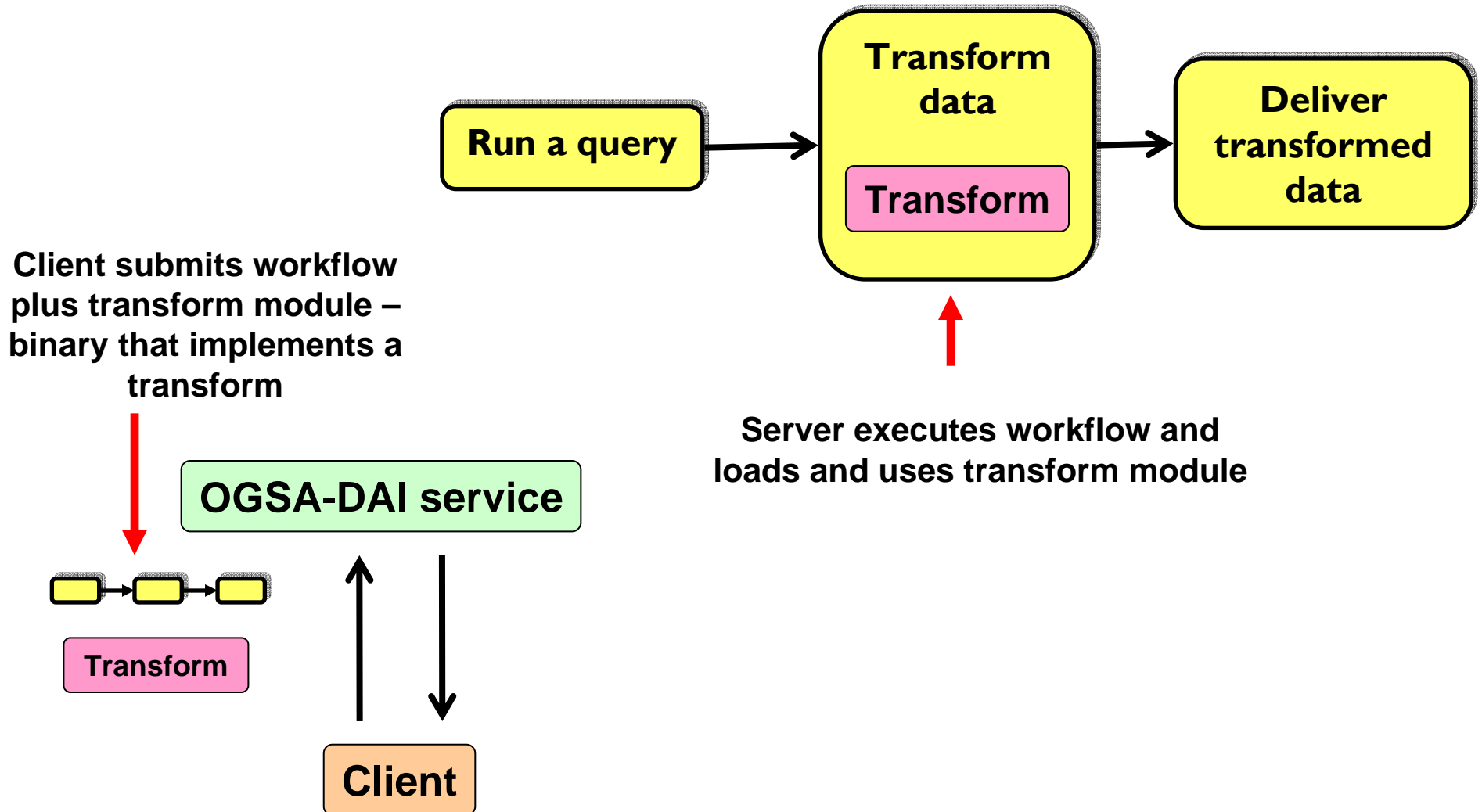
# OGSA-DQP – what it does and the client's view



# OGSA-DAI, DQP and views – the goal



# Generic transforms – more power to the client





# Realising the scenarios

## FirstDIG – data with different schema distributed across multiple databases within an organisation

- FirstDIG – First Data Investigation on the Grid
  - <http://www.epcc.ed.ac.uk/firstdig/>
  - 2003-2004
- FirstGroup plc – UK's largest transport operator
- Distributed databases
  - Customer contact
  - Vehicle mileage
  - Ticket revenue
  - Schedule adherence
- Database types
  - Relational, ODBC-enabled, COBOL files,...

# FirstDIG

- **OGSA-DAI**
  - Distributed data resource access
  - Hide heterogeneities
    - Physical database locations
    - Database drivers
    - Connection URLs
    - User names or passwords
  - Single access point for client
- **Data translation and joins done by client**
  - 2003 – OGSA-DAI was in its early days
- **Nevertheless**
  - “results of this exercise will revolutionise the way we do things in the bus industry”
    - Darren Unwin, First Group’s IT Operations Director

# BEinGRID – data with same schema distributed across multiple databases within an organisation

- Business Experiments in Grid
  - <http://www.beingrid.eu>
  - Demonstrate applicability of Grid technology to business
- BEI2 – Sales Management System
  - ENEA, CINECA, Tecnocassa, Pizza New
- OGSA-DAI
  - Standardised and uniform layer to distributed heterogeneous databases



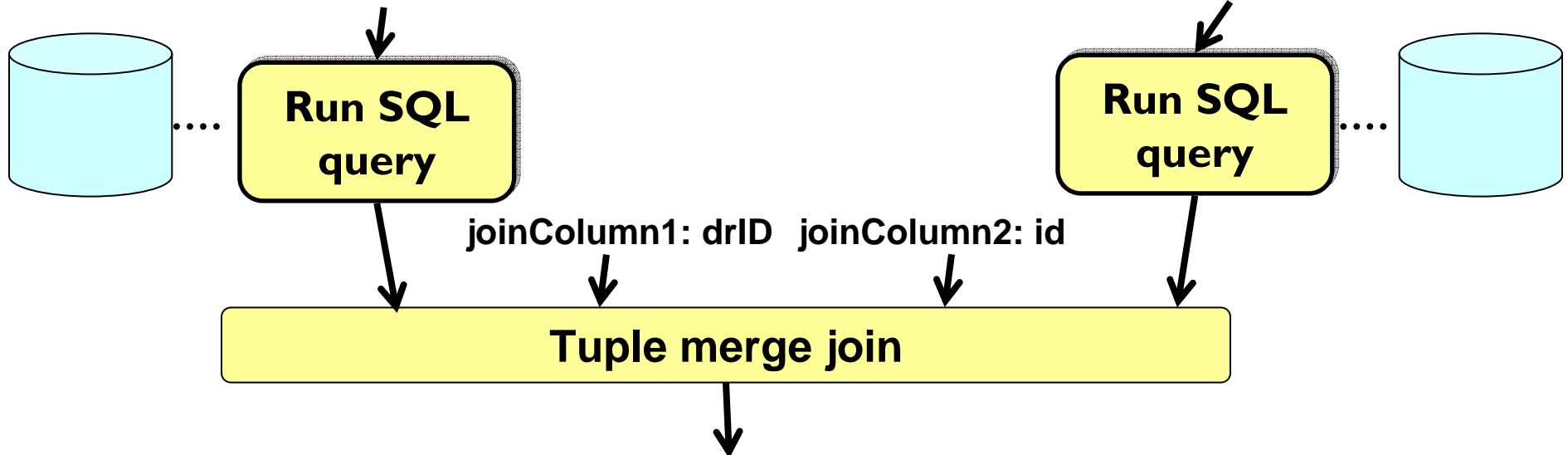
## VOTES – data with different schema distributed across multiple databases within a group of strategic partners

- Virtual Organisations for Trials and Epidemiological Studies (VOTES)
  - <http://labserv.nesc.gla.ac.uk/projects/votes/index.html>
  - UK Medical Research Council project
- Provision of data access and integration functionality for the clinical domain
- Distributed databases
  - Patient information
  - Clinical trials results
- Database types
  - Microsoft SQL Server, Access,...

# VOTES – cross-database join activity

**SELECT drID, name, id FROM  
patients ORDER by drID**

**SELECT id, drName FROM doctors  
ORDER by id**



- This is equivalent to running:

**SELECT id, name, drID, drName FROM patients, doctors WHERE  
patients.drID = doctors.id;**

- patients and doctors are in two different databases

# SIMDAT – data with different schema distributed across multiple databases within a group of strategic partners



- SIMDAT
  - <http://www.simdat.eu>
  - Grids for industrial product development
  - Automotive, aerospace, pharmaceutical, meteorological
    - e.g. Audi, BAE Systems, EADS, Meteo France, UK Met Office
- GRIA OGSA-DAI ontology and data services
  - Transparent access to files and databases
  - Industrial-strength fine-grained access control
  - Common interface to local and remote data
- Semantic Bus
  - Automotive semantic mediation for CAE/CAD
  - Access local database with same schema
  - Go through semantic mediation steps to access remote data resources

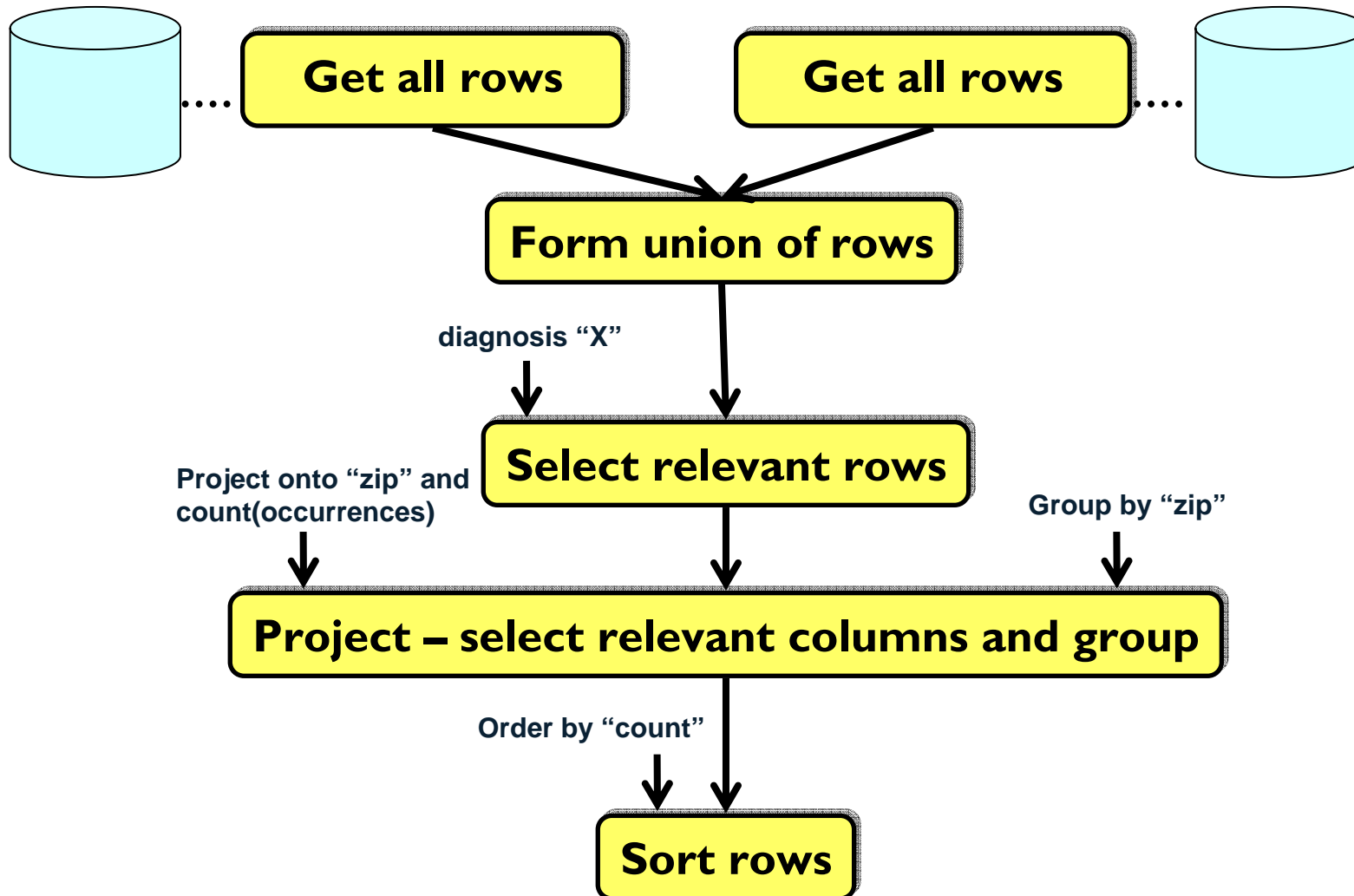
## Public Health Grid – data with different schema distributed across multiple databases within a group of strategic partners

- US Public Health Grid
  - US Centers for Disease Control
  - University of Pittsburgh
  - Tarrant Country Public Health Department
  - Dallas County Public Health Department
- Health query system
  - Look for incidences of some disease on the rise over an area
  - Historical and live data
- Health centres maintain their own databases
  - Distributed databases
  - Different products and schemas
    - e.g. PatientID, Id, PatientIdentifier, PatientNumber
  - Security and privacy is important

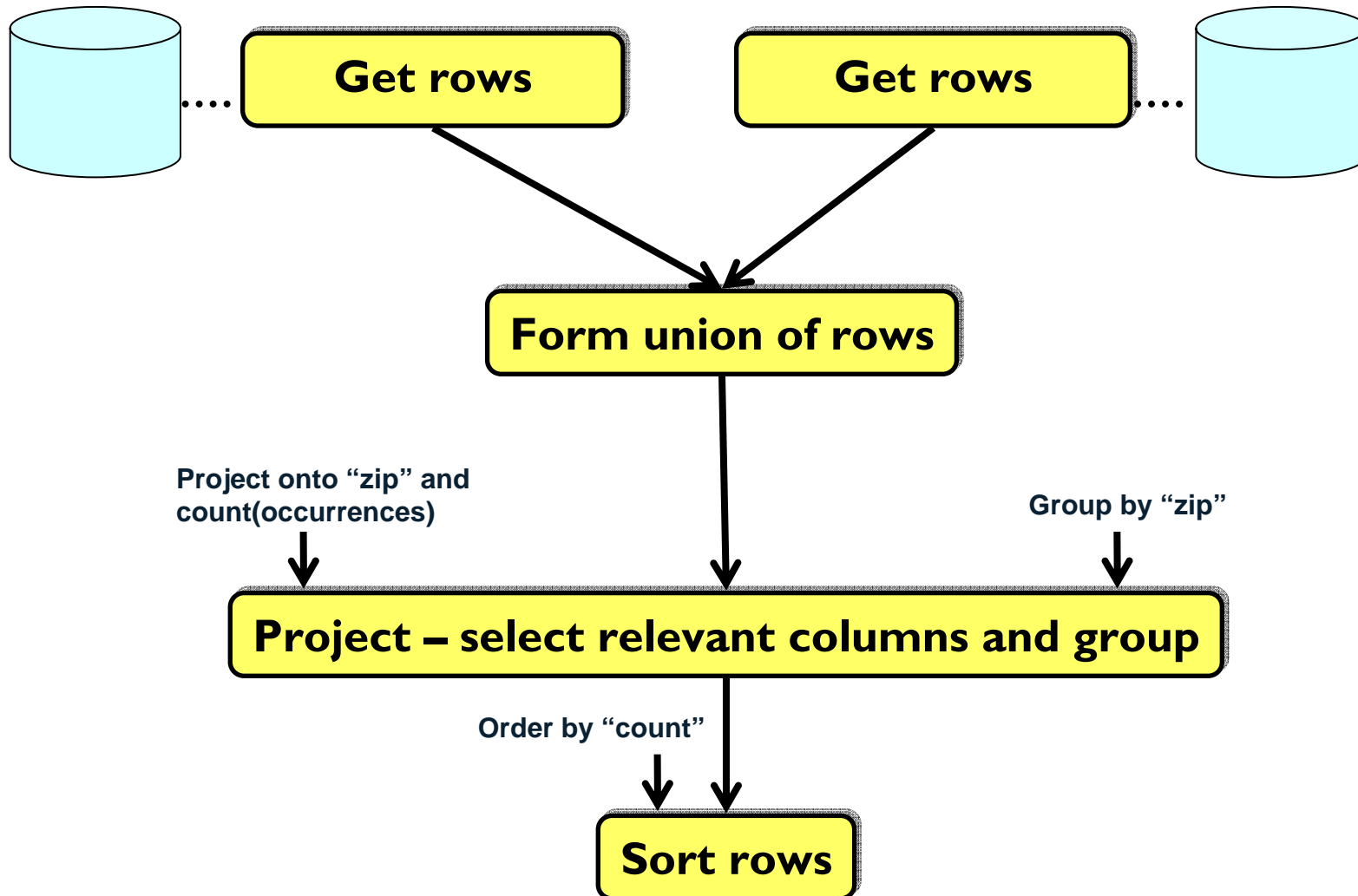
# Public Health Grid – distributed query system

- Distributed query system
  - Present all the databases as if they're a single database
  - `SELECT postal_code, COUNT(patients) FROM databases WHERE diagnosis = 'X' ORDER BY COUNT(patients)`
  - “databases” is a representation of the federated databases
- OGSA-DAI
  - Handles parsing, task distribution and integration
  - Handles heterogeneity of database products and schema
  - Reduces data transfer volumes
  - Provides a security layer

# Public Health Grid – workflow



# Public Health Grid – optimised workflow



# Public Health Grid – visualisation

- Extend the previous workflow
- Transform activity
  - SQL tuples => KML
- KML
  - GoogleMaps XML configuration file
  - Geographical area
  - Tick marks
  - Labels
- Visualise numbers of diagnoses clustered by post-codes using GoogleMaps

# ADMIRE – data with different schema distributed across multiple databases within a group of strategic partners



- ADMIRE – Advanced Data Mining and Integration Research for Europe
  - <http://www.admire-project.eu/>
  - EU 7<sup>th</sup> Framework program project
- Infrastructure for data integration and mining
  - Large scale enterprise systems
- Initial applications
  - Flood modelling and simulations
  - Customer relationship management
- OGSA-DAI is a key technology

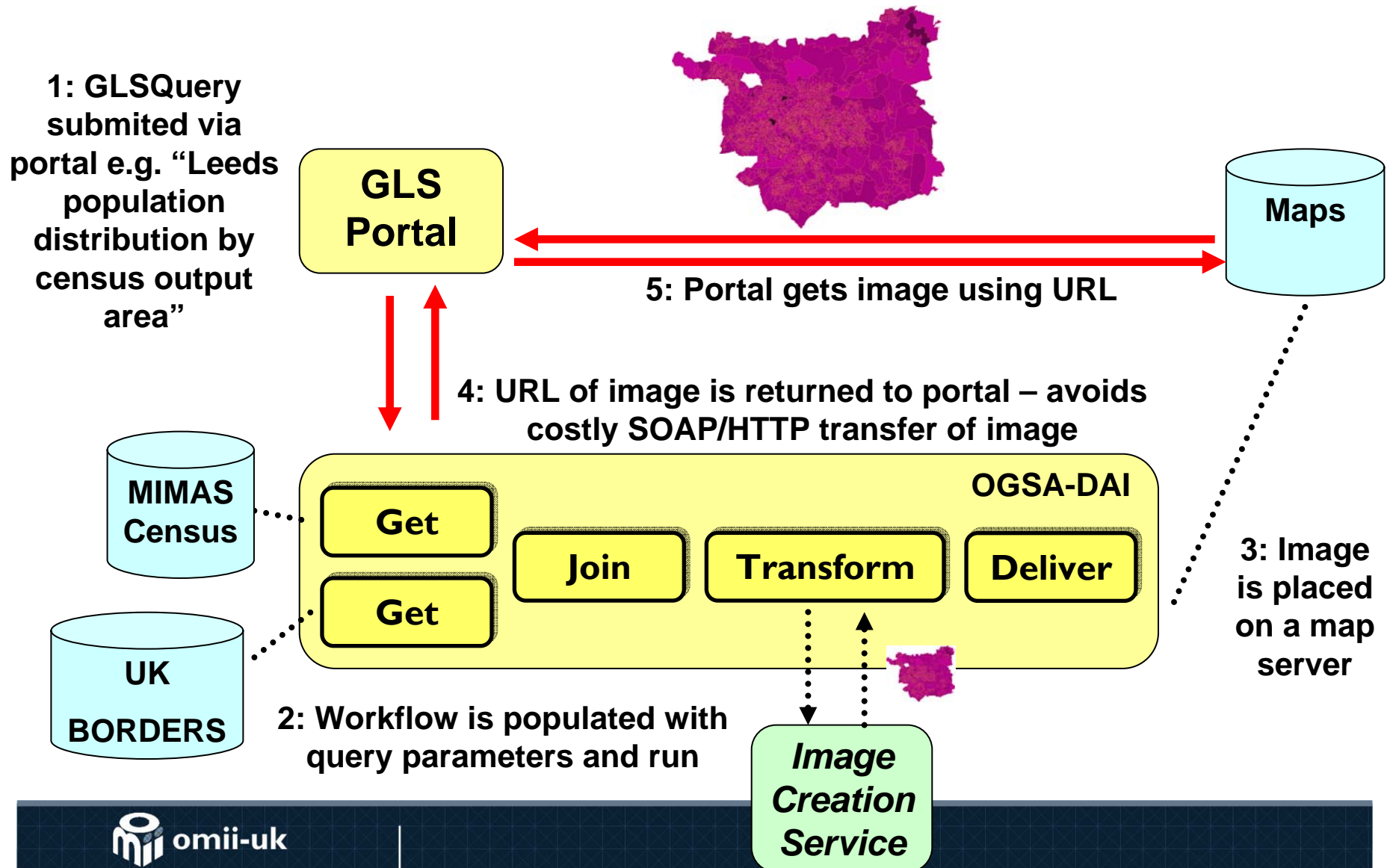
# SEE-GEO – working with private and public data

- SEcurE access to GEOspatial services
  - <http://edina.ac.uk/projects/seesaw/seegeo/index.html>
  - EDINA, MIMAS, NeSC, NCeSS
  - UK JISC project
- Geographical information systems
- Virtual integration of and access control to
  - Census data
  - Borders data
  - Data hosted by other organisations and exposed as services
- OGSA-DAI for federation of heterogeneous data resources

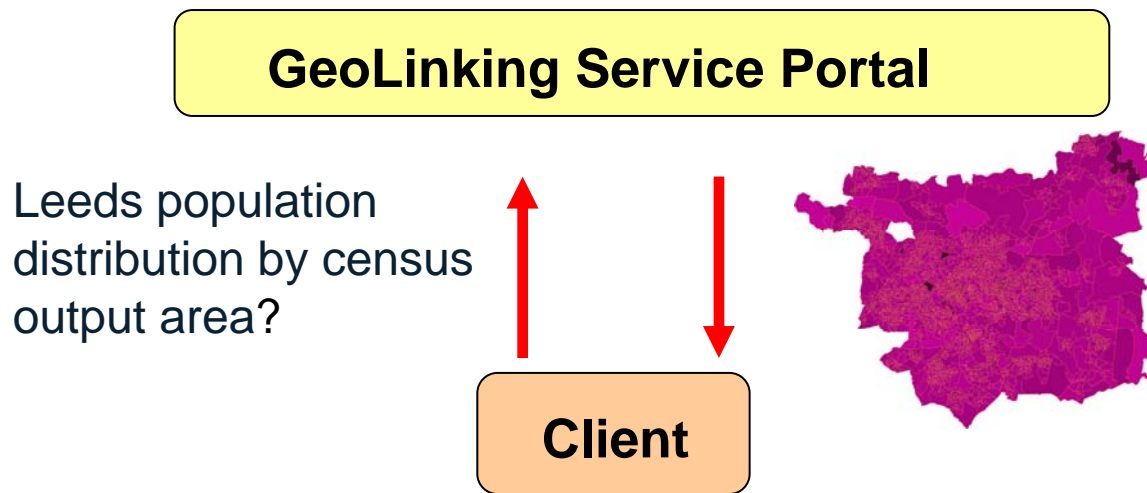
EDINA



# SEE-GEO – geo-linking service portal



# SEE-GEO – client's view



# GEOGrid – work with private data and manage access to distributed data

- Global Earth Organisation (GEO) Grid
  - <http://www.geogrid.org/>
  - National Institute of Advanced Industrial Science and Technology, Japan
- Geospatial data and services
  - Disaster mitigation
  - Environmental monitoring
  - Natural resource exploration
- Virtual integration and access control
  - Satellite imagery
  - Geological data
  - Ground-sensed data
  - Each with different ownership and access policies
- OGSA-DQP and OGSA-DAI
  - Federation of heterogeneous data resources
  - OMII-Europe VOMS components for access control



# GEOGrid – manage access to distributed data

- Virtual organisations access data resources
  - VO membership determines data that can be accessed

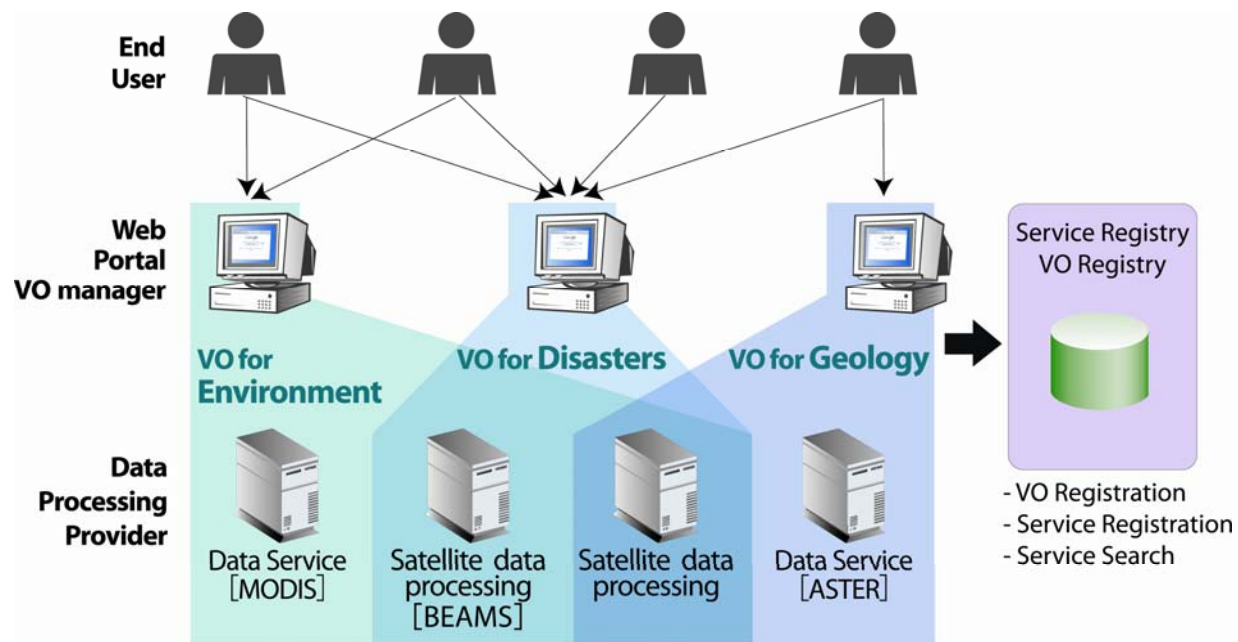


Image courtesy of Isao Kojima, GEOGrid

# What if we combined...?

- **Public Health Grid and SEE-GEO**
  - Visualisation of incidences of infection
  - Graphical query tool – select regions via drag and drop
- **FirstDIG + SEE-GEO**
  - Visualise where complaints arise the most
  - Visualise where buses are late the most
  - Factor in house prices, crime rates (socio-economic factors), congestion...



# OGSA-DAI pros and cons

# OGSA-DAI pros

- **Efficient**
  - Workflows encapsulate many operations within a single web service invocation
- **Data providers retain control of data**
- **Security layer**
  - Federation, data resource, table, document, file, row, line
- **Data transformation and joins layer**
- **Lightweight clients**

# OGSA-DAI pros

- **Extensible**
  - Data resources, activities, security, presentation layers
- **Enabling framework**
  - Save development time – focus on your application
- **100% Java open source freeware**
  - Runs under Windows, UNIX, Linux
- **Compliant with**
  - Globus Toolkit 4.0.x
  - Apache Axis/Tomcat
  - OMII 3.4.0
  - UNICORE – by OMII-Europe
  - VOMS – by OMII-Europe



# OGSA-DAI cons

- Another “thing” between you and the data
  - Web service presentation layer – optional
  - Workflow execution
- Performance implications
  - Is speed of access paramount?
  - Or are the benefits on the previous slide of more importance?
- OGSA-DAI is not a silver bullet
  - Do the strengths outweigh the weaknesses for your specific requirements?



# Hiding workflows behind facades

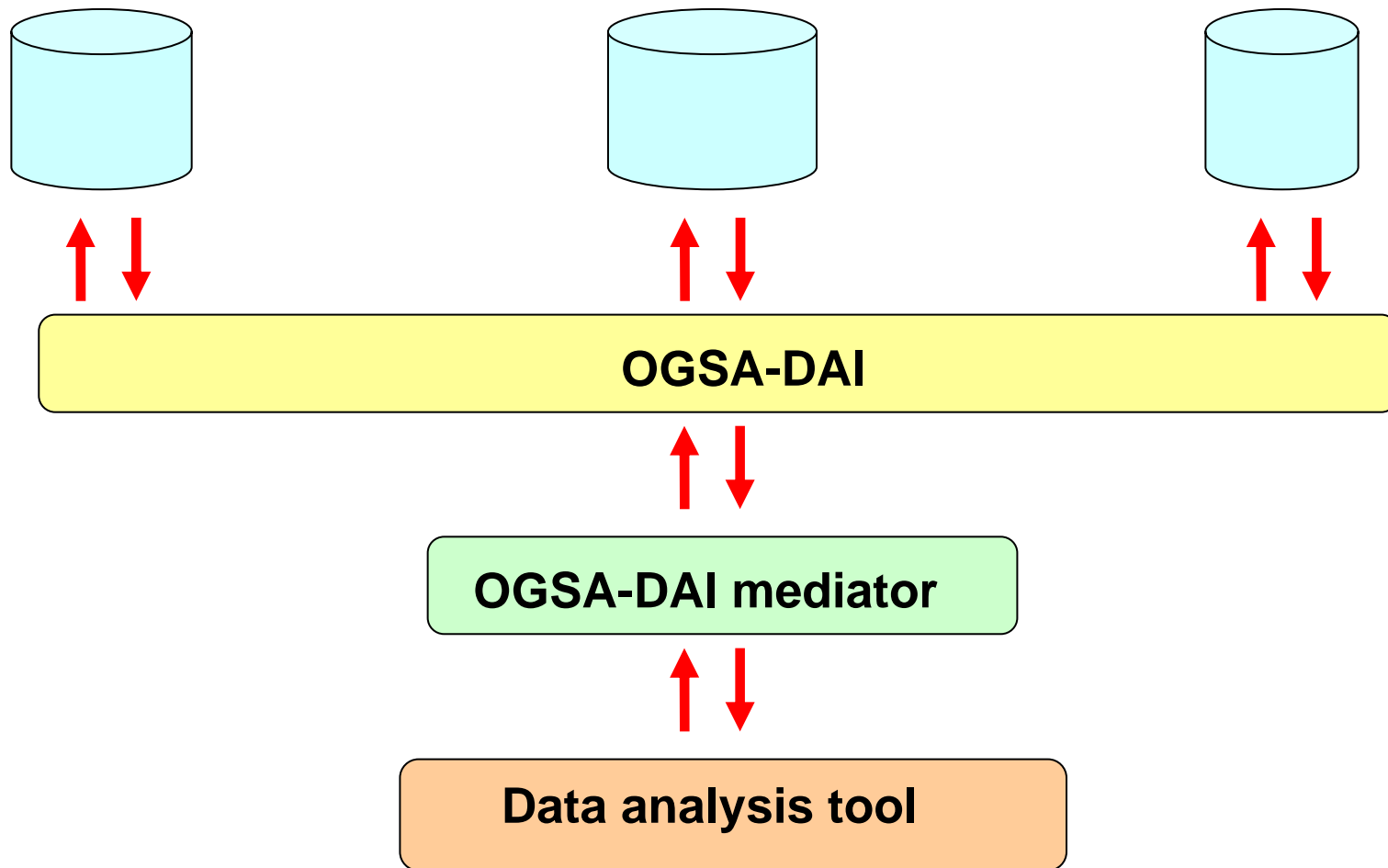
# Problems with OGSA-DAI workflows

- Too expressive
  - Infer semantics from names of activities available on OGSA-DAI server
  - Must interrogate the server
  - Problems using OGSA-DAI services in workflow engines e.g. Taverna
- A de-facto standard only
- Not compatible with many existing data analysis tools

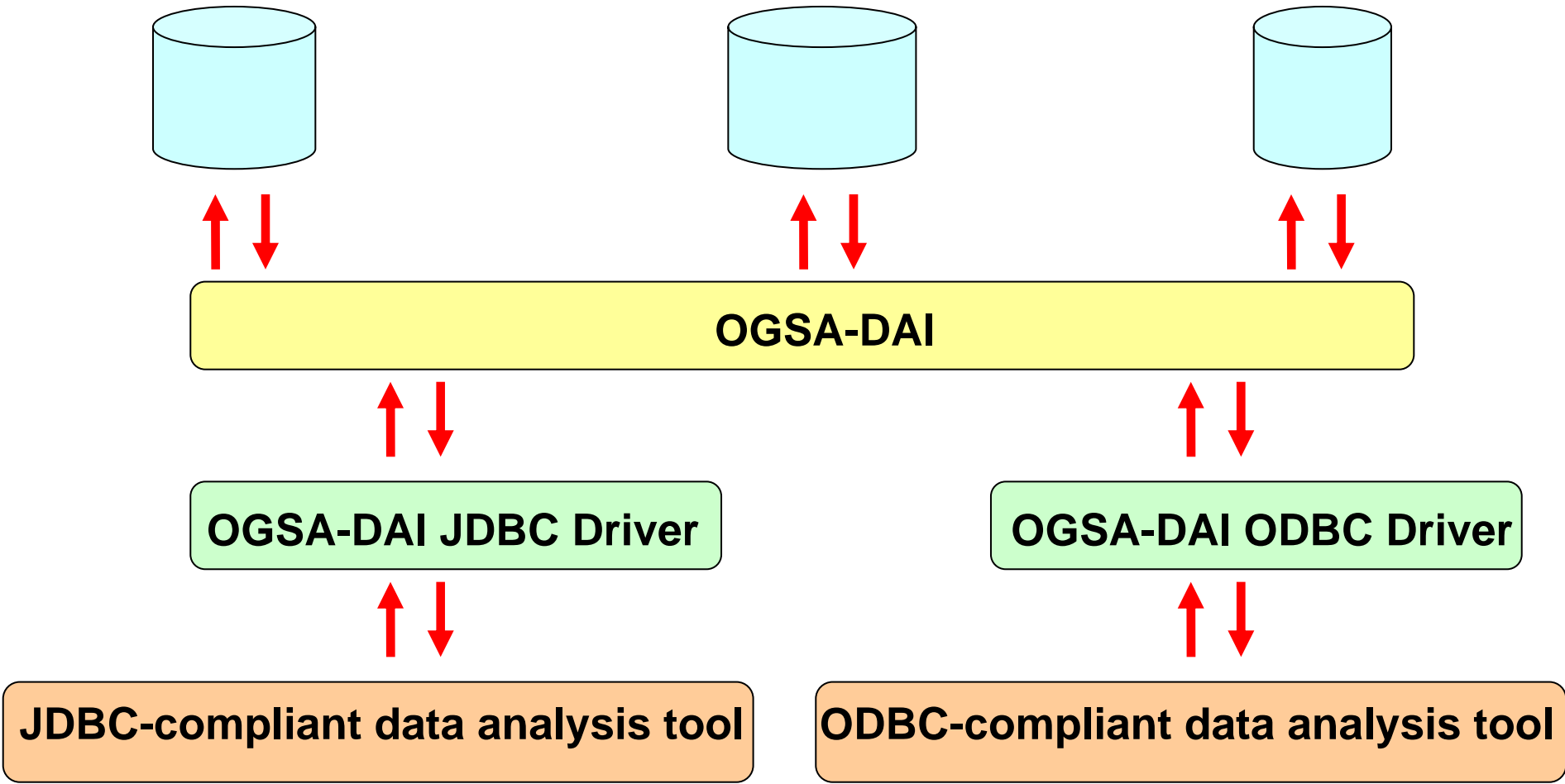
# Facades

- Define facades on top of OGSA-DAI
- Why?
  - Provide interfaces with more tightly-defined semantics
  - Comply with standards
  - Exploit existing data analysis tools
- Continue to exploit the power of workflows under-the-hood
  - “Canned workflows”
  - Templates selected and populated, executed and parsed

# Grid-enabling existing data-related products



# JDBC and ODBC-compliant facades



# JDBC and ODBC-compliant facades

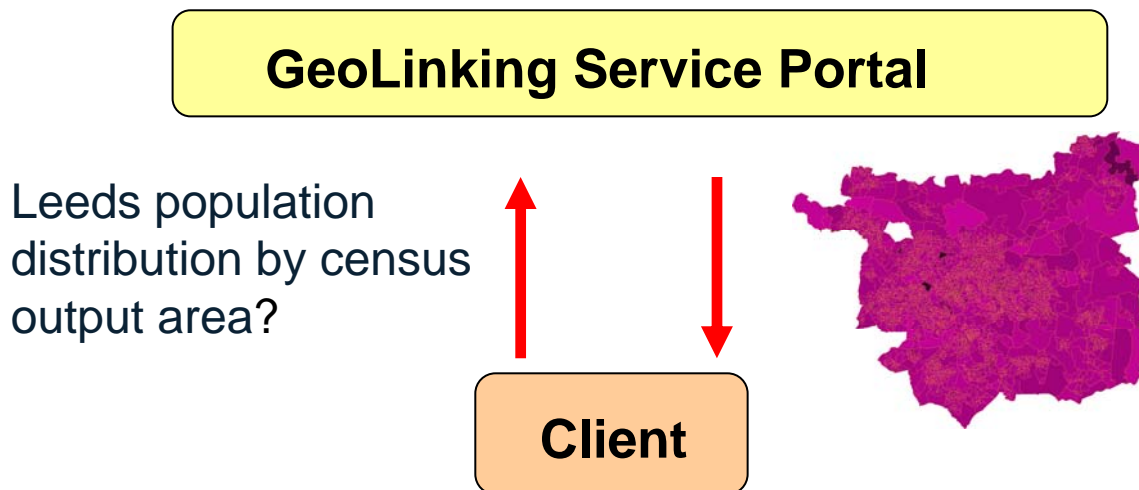
- Analyse federations of data resources managed by OGSA-DAI
- INWA project – Innovation Node Western Australia
  - <http://www2.epcc.ed.ac.uk/~inwa/>
  - EPCC, Curtin Business School Australia, Chinese Academy of Sciences Beijing
  - Secure distributed data mining of property and mortgage data
  - Use OGSA-DAI with ODBC-compliant SPSS statistical analysis tool
- JDBC driver for OGSA-DAI
  - Under development by Mathias Brito and BEinGRID
- ODBC driver for OGSA-DAI
  - Also of interest to BEinGRID

# Application-specific web services

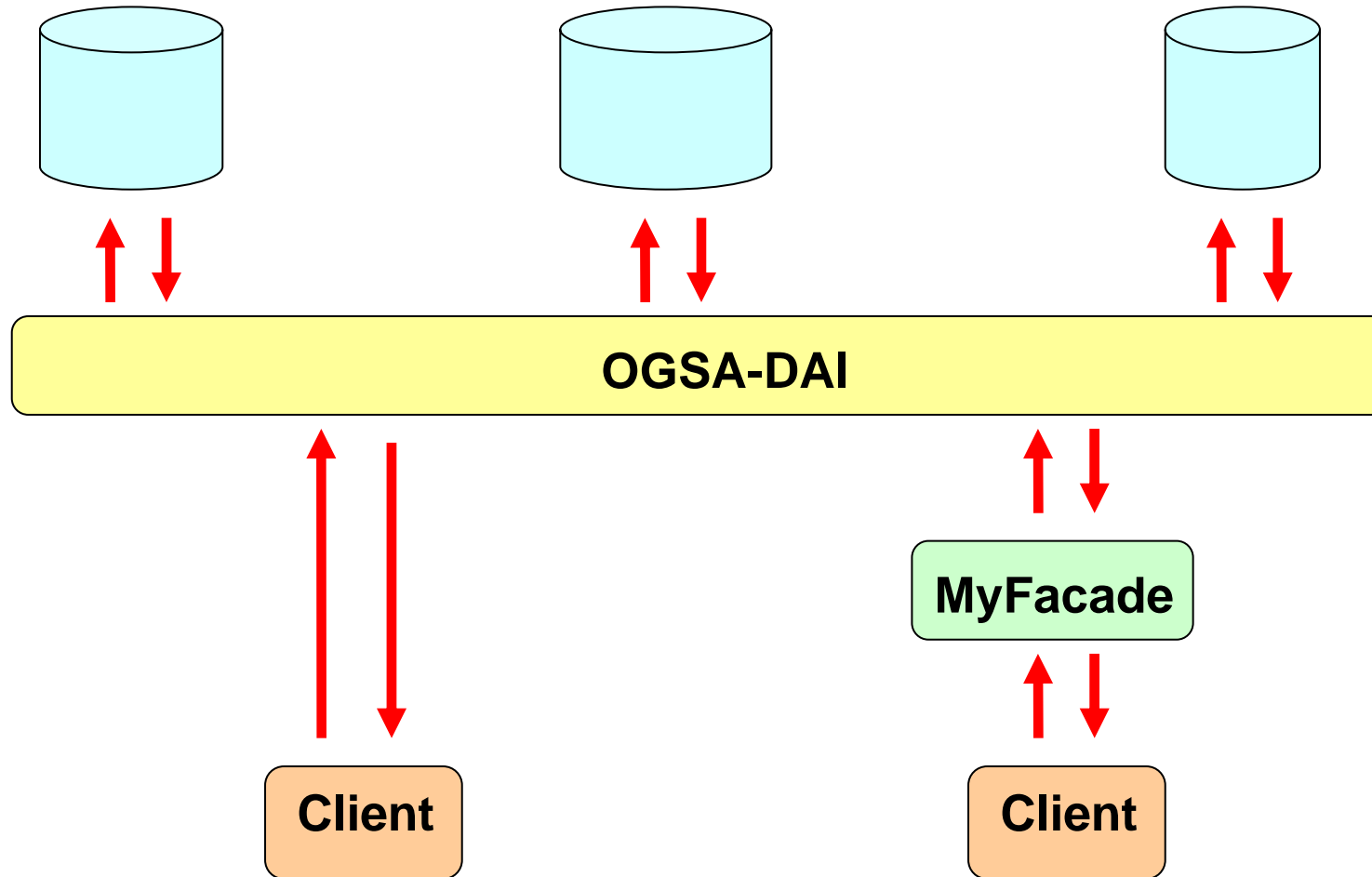
- Provide web services with application-specific operations
  - Book findByISBN(ISBN)
  - List<Book> findByAuthor(Author)
  - List<Book> findByKeyword(Word)
- Yields services with well defined semantics
- Suitable for use in workflow engines
- Precludes the use of generic data exploration, mining and manipulation tools

# SEE-GEO and GEOGrid – standards and facades

- SEE-GEO and GEOGrid's services
- OGC GeoLinking Services
  - Open Geospatial Consortium specifications
  - Specifications being hardened into standards



# Using OGSA-DAI – directly and via facades





# Standards, WS-DAI and OGSA-DAI

# WS-DAI

- OGF DAIS working group
  - Develop Web services standards for data access and integration
- WS-DAI specifications
  - Core – common to all
  - WS-DAIR – relational
  - WS-DAIX – XML
- To become standards WS-DAI needs two inter-operable implementations

## OGSA-DAI and WS-DAI – what's the difference?

- OGSA-DAI is a product
- WS-DAI is a specification
- OGSA-DAI is based upon WS-DAI specifications from 2003
  - Web services have workflow-specific operations e.g.
    - `execute(workflow)`
  - OGSA-DAI was intended as a reference implementation of WS-DAI

## OGSA-DAI and WS-DAI – what's the difference

- **WS-DAI has evolved significantly since 2003**
  - No more workflows
  - Web services have data resource-specific operations  
e.g.
    - `GetDocuments("bangles", "madness", "jam")`
    - `XPathExecute("/bands/pop/@lead='hoffs'")`
    - `GetItems(1,3)`
- **Users found OGSA-DAI workflows useful**
  - So the product stuck with them
  - Divergence from the specifications

# WS-DAIR – relational data

- Relational databases
  - Tables and rows
- SQL resource
  - Relational database
- SQL response resource
  - Results of operations on the database
  - Row sets, update counts, stored procedure results, function output parameters
- SQL row set resource
  - Rows in a row set

# WS-DAIX – XML data

- XML databases
  - Documents, collections and associated XML Schema
- XML collection resource
  - An XML database or a sub-collection within
- XML document resource
  - An XML document in a collection
- XML sequence resource
  - XML data items e.g. from XPath or XQuery statement

# WS-DAI – accessing data

- Direct access (synchronous)
  - Query runs and data is returned to client
  - Update runs and result is returned to client
- Indirect access (asynchronous)
  - Factories
  - Query runs, a resource is created and populated, resource ID and service URL returned to client
  - Client contacts service/resource for data
  - Like asynchronous delivery in OGSA-DAI

# OGSA-DAI WS-DAIX

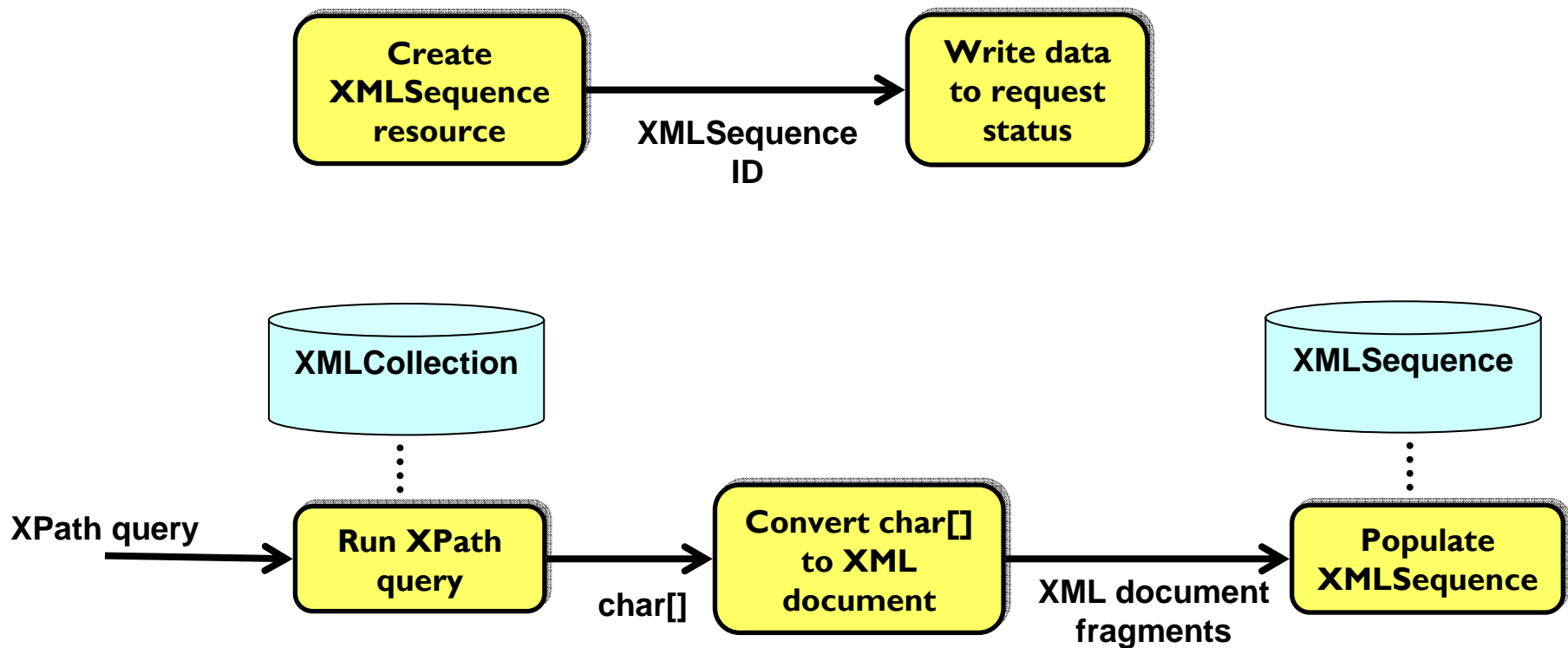
- WS-DAIX-compliant façade
  - WS-DAIX-compliant web services
  - OGSA-DAI core
- “Canned” workflows
  - Constructed in-code on the server
  - Populated with arguments from the client
  - e.g. XPath query, XML collection name, XML document,...

# OGSA-DAI WS-DAIX

- OGSA-DAI WS-DAIX services act as “clients” to core OGSA-DAI functionality
- Direct access
  - Workflow to run query
- Indirect access
  - Workflow to create resource
  - Workflow to run query and populate resource
- Clients are unaware of all this

# OGSA-DAI WS-DAIX “canned” workflows

- Indirect access to results of an XPath query



# OGSA-DAI and facades

- Pros
  - Access workflow power via well-defined interfaces
  - Facilitate inter-operability with other components
- Cons
  - Evolution of specifications into standards
    - Shifting sands – WS-DAI
    - Pulled rugs – OGSF
  - Development overhead
  - Relying on 3<sup>rd</sup> parties to implement standards e.g. GT and WSRF

# OGSA-DAI and facades

- Lightweight presentation layers
- APIs and interfaces everywhere
- Don't force but don't prohibit



# Roadmap and conclusions

# Roadmap

- **OGSA-DAI 3.0**
  - September 2007
- **VOTES join activity and document-based client extension pack**
  - November 2007
- **Servlet extension pack**
  - January 2008
- **Dynamic data resource plug-in deployment extension pack**
  - March 2008
- **Inter-OGSA-DAI data delivery activities extension pack**
  - April 2008
- **OGSA-DAI WS-DAIX 1.0**
  - May 2008

# Roadmap

- **OGSA-DQP 3.2**
  - June 2008 in conjunction with NextGrid
- **SQL views**
  - June 2008 in conjunction with BEinGRID
- **OGSA-DAI 3.1**
  - September 2008
  - Integration of extension packs
  - Bug fixes and new features
- **OGSA-DAI WS-DAIR 1.0**
  - October 2008



# Related releases

- OMII-Europe OGSA-DAI 3.0 components
  - Out now
  - Plug-ins for VOMS authorization
  - OGSA-DAI 3.0 UNICORE presentation layer
- BEinGRID components
  - OGSA-DAI data publisher
  - OGSA-DAI database triggers
  - JDBC driver for OGSA-DAI – with Mathias Brito



# Conclusion

- **OGSA-DAI**
  - Sharing data resources to enable collaboration
  - Workflows to access, transform, filter, integrate and deliver data
  - Distributed query processing
  - Facades can make workflows more usable
  - Highly customisable enabling environment
- **Facades**
  - Provide well-defined API or interface to workflows
  - Harness the power of the workflow while gaining a well-defined semantics
- **WS-DAI**
  - Specifications and standards for data access

# Further information

- WWW site : <http://www.ogsadai.org.uk>
- Info : [info@ogsadai.org.uk](mailto:info@ogsadai.org.uk)
- Users e-mail list : [users@ogsadai.org.uk](mailto:users@ogsadai.org.uk)