SparqlDQP
Stay at EPCC & NeSC

Carlos Buil Aranda

Ontology Engineering Group
Facultad de Informática
Universidad Politécnica de Madrid
cbuil@fi.upm.es

5th November 2009

Table of Contents

- Introduction
- OGSA-DAI & OGSA-DQP
- SparqlDQP
- Future work
- Way of working at
  - EPCC
  - NeSC
- Conclusions

Table of Contents

- **Introduction**
- OGSA-DAI & OGSA-DQP
- SparqlDQP
- Way of working at
  - EPCC
  - NeSC
- Conclusions

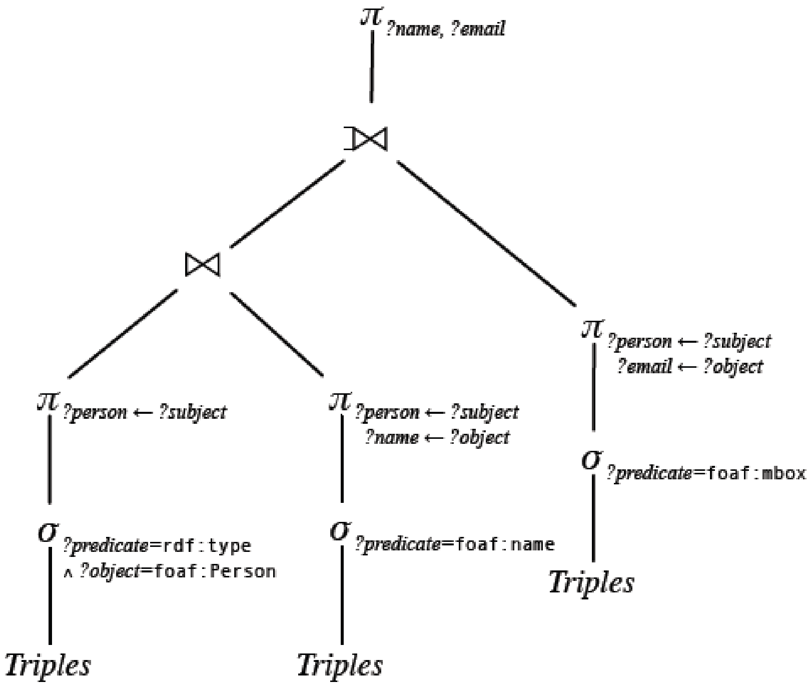
Introduction

- Can RDB2RDF Tools Feasibly Expose Large Science Archives for Data Integration?, ESWC 2009
  - No, among other reasons there is no research in Sparql Query Optimisation (authors dixit)
- Currently there are only a few approaches to federate SPARQL queries
  - Networked Graphs (Staab WWW2008)
  - Executing SPARQL Queries over the Web of Linked Data (Hartig, Bizer, Freytag, ISWC09)
  - DARQ (2006)
  - SemWIQ (Langegger, iiWAS2008)
- Problems from the previous approaches
  - They implement a basic system for optimising Sparql queries
  - They do not take into account blank nodes

Introduction

- Proposal: use existing techniques for SQL query optimisation in SPARQL
- Why?
  - A relational algebra for SPARQL, Richard Cyganiak (2005)
  - The expressive power of SPARQL, Renzo Angles and Claudio Gutierrez (2008)

```
SELECT ?name ?email
WHERE {
?person rdf:type foaf:Person .
?person foaf:name ?name .
OPTIONAL { ?person foaf:mbox ?email }
}
```



π ?name, ?email
⟕
⋈
π ?person ← ?subject
σ ?predicate=rdf:type ∧ ?object=foaf:Person
Triples
π ?person ← ?subject ?name ← ?object
σ ?predicate=foaf:name
Triples
π ?person ← ?subject ?email ← ?object
σ ?predicate=foaf:mbox
Triples

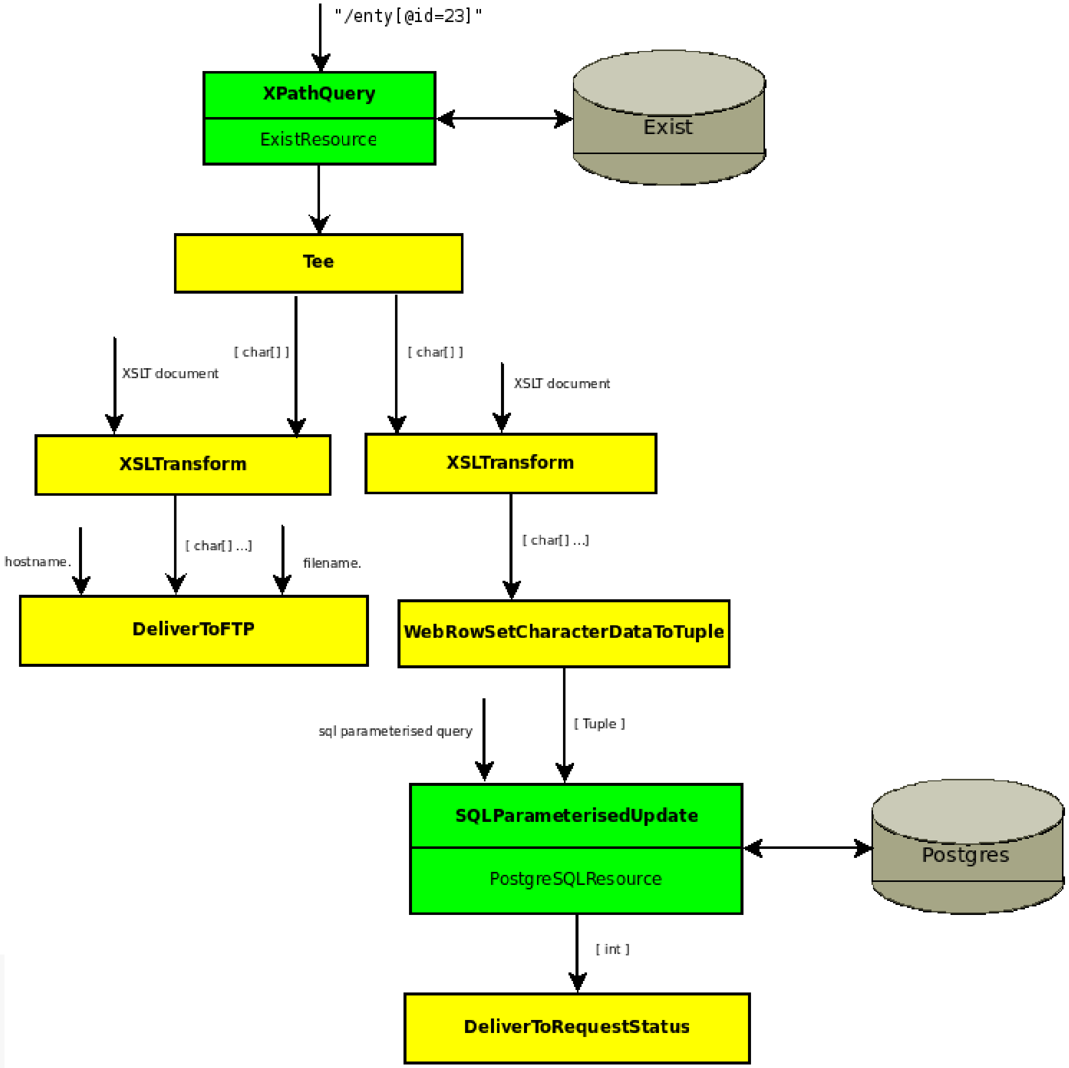
Table of Contents

- Introduction
- **OGSA-DAI & OGSA-DQP**
- SparqlDQP
- Reasoning Web Summer School
- Way of working at
  - EPCC
  - NeSC
- Conclusions

OGSA-DAI

- OGSA-DAI is
  - An extensible framework that allows to
  - Access, integrate, transform and deliver
  - Distributed and heterogeneous sources of data
  - Implements part of the WS-DAI specification
- OGSA-DAI key elements are
  - Resources (Data Resource)
  - Activities (=operations or named unit of functionality)
  - Workflows (=composition of activities)
    - Pipeline workflow (A set of chained activities executed in parallel with data flowing between the activities)
    - Sequence workflow
    - Parallel workflow
- OGSA-DAI provides indirect access to data resources

OGSA-DAI

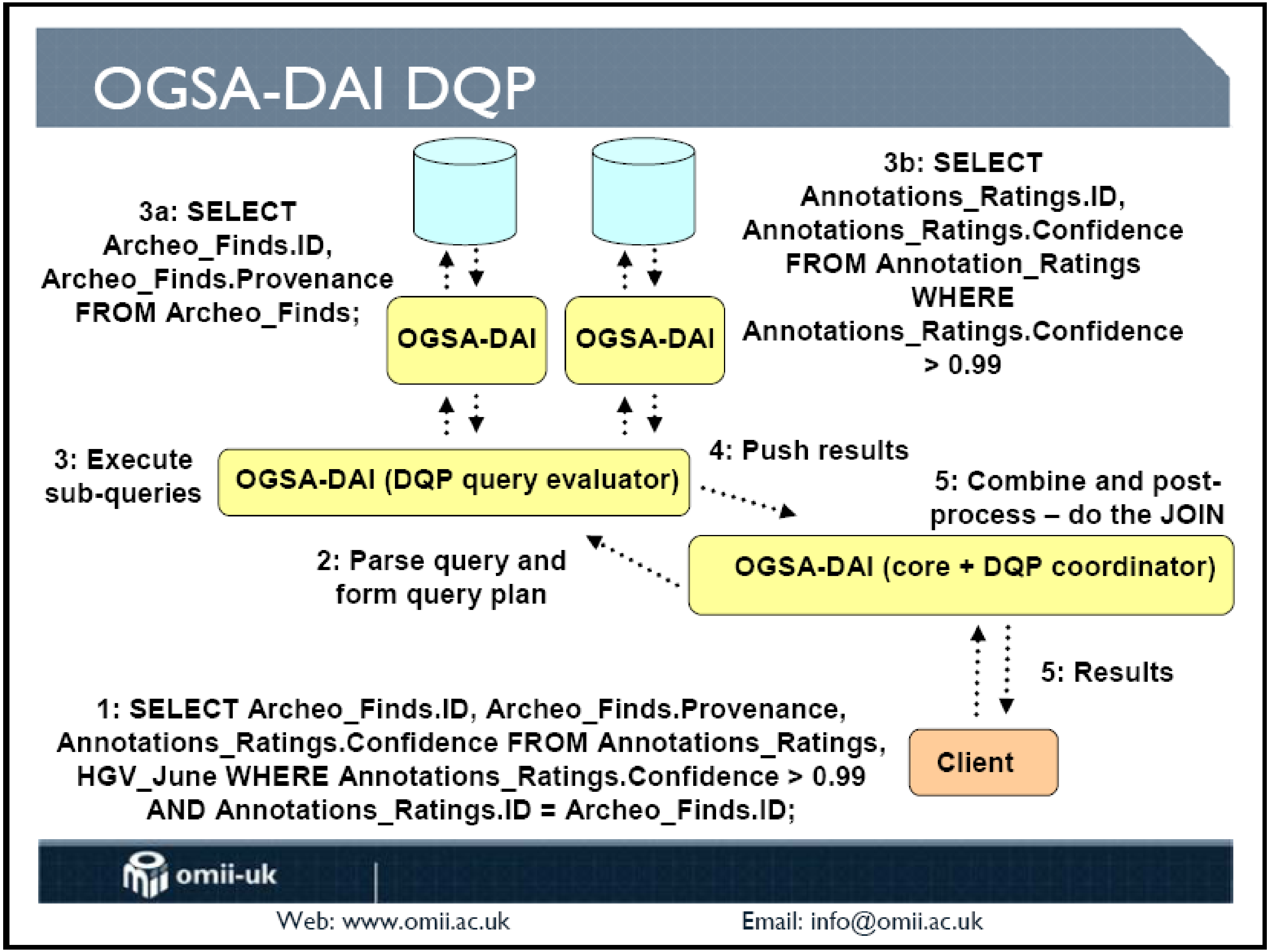

"/enty[@id=23]"
XPathQuery
ExistResource
Exist
Tee
XSLT document
[char[]]
[char[]]
XSLT document
XSLTransform
XSLTransform
hostname.
[char[] ...]
filename.
[char[] ...]
DeliverToFTP
WebRowSetCharacterDataToTuple
sql parameterised query
[Tuple]
SQLParameterisedUpdate
PostgreSQLResource
Postgres
[int]
DeliverToRequestStatus

OGSA-DQP

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

Source: http://www.ogsadai.org.uk/documentation/presentations/ogf25/OGF-25-OGSA-DAI-Tutorial.pdf

OGSA-DQP



OGSA-DAI DQP

3a: SELECT Archeo_Finds.ID, Archeo_Finds.Provenance FROM Archeo_Finds;

3b: SELECT Annotations_Ratings.ID, Annotations_Ratings.Confidence FROM Annotation_Ratings WHERE Annotations_Ratings.Confidence > 0.99

OGSA-DAI
OGSA-DAI
OGSA-DAI (DQP query evaluator)
OGSA-DAI (core + DQP coordinator)
Client

3: Execute sub-queries

4: Push results

5: Combine and post-process – do the JOIN

2: Parse query and form query plan

5: Results

1: SELECT Archeo_Finds.ID, Archeo_Finds.Provenance, Annotations_Ratings.Confidence FROM Annotations_Ratings, HGV_June WHERE Annotations_Ratings.Confidence > 0.99 AND Annotations_Ratings.ID = Archeo_Finds.ID;

omii-uk

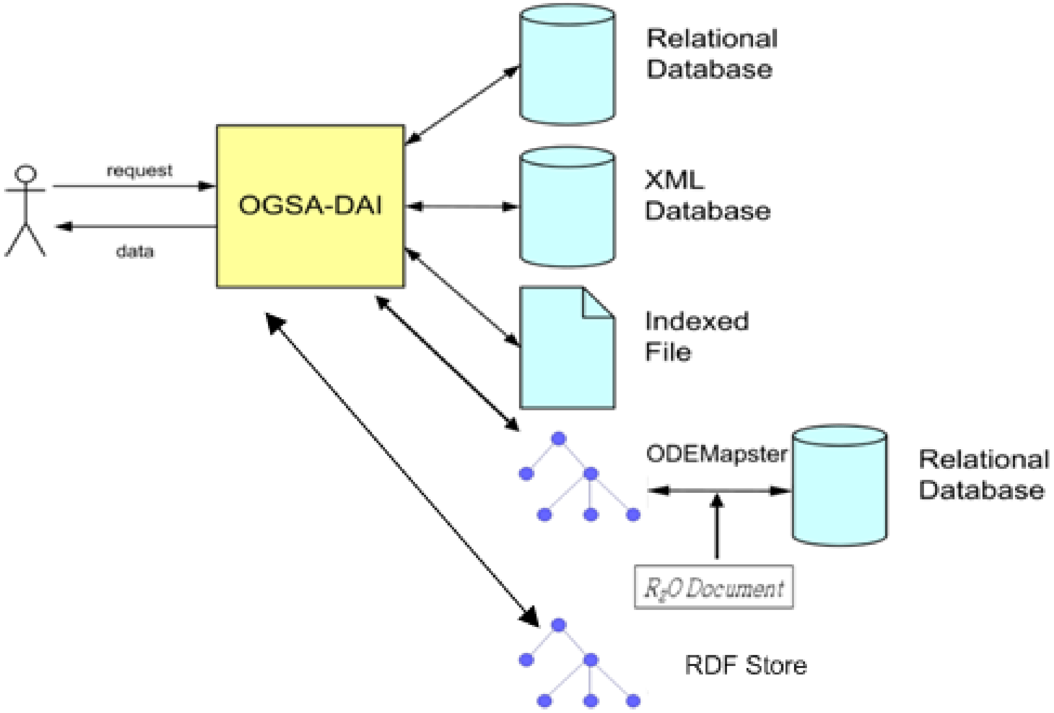
Web: www.omii.ac.uk Email: info@omii.ac.uk

Source: http://www.ogsadai.org.uk/documentation/presentations/ogf25/OGF-25-OGSA-DAI-Tutorial.pdf

Table of Contents

- Introduction
- OGSA-DAI & OGSA-DQP
- **RDF Resource & SparqlDQP**
- Future work
- Way of working at
  - EPCC
  - NeSC
- Conclusions

RDF Resource

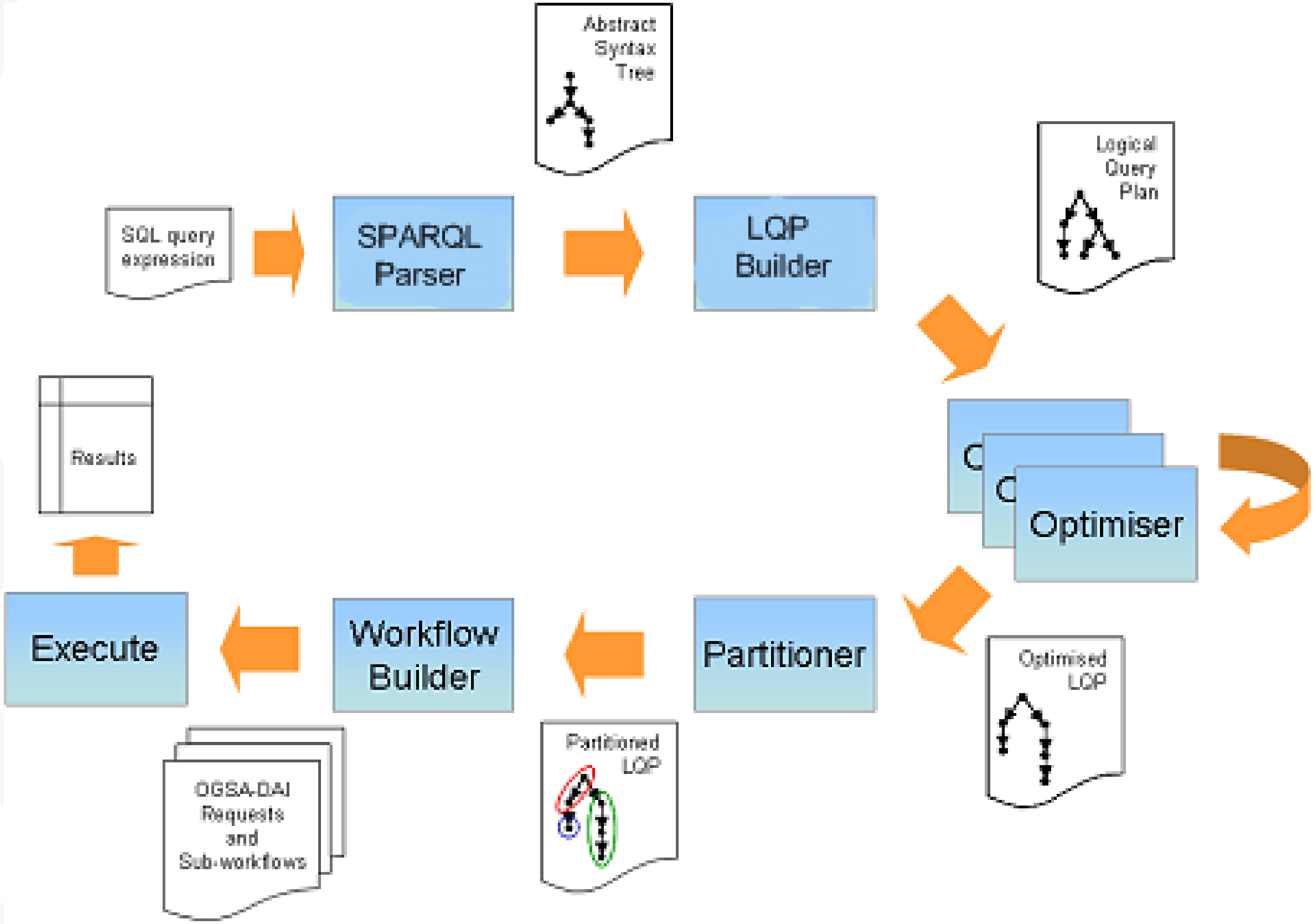


SparqlDQP

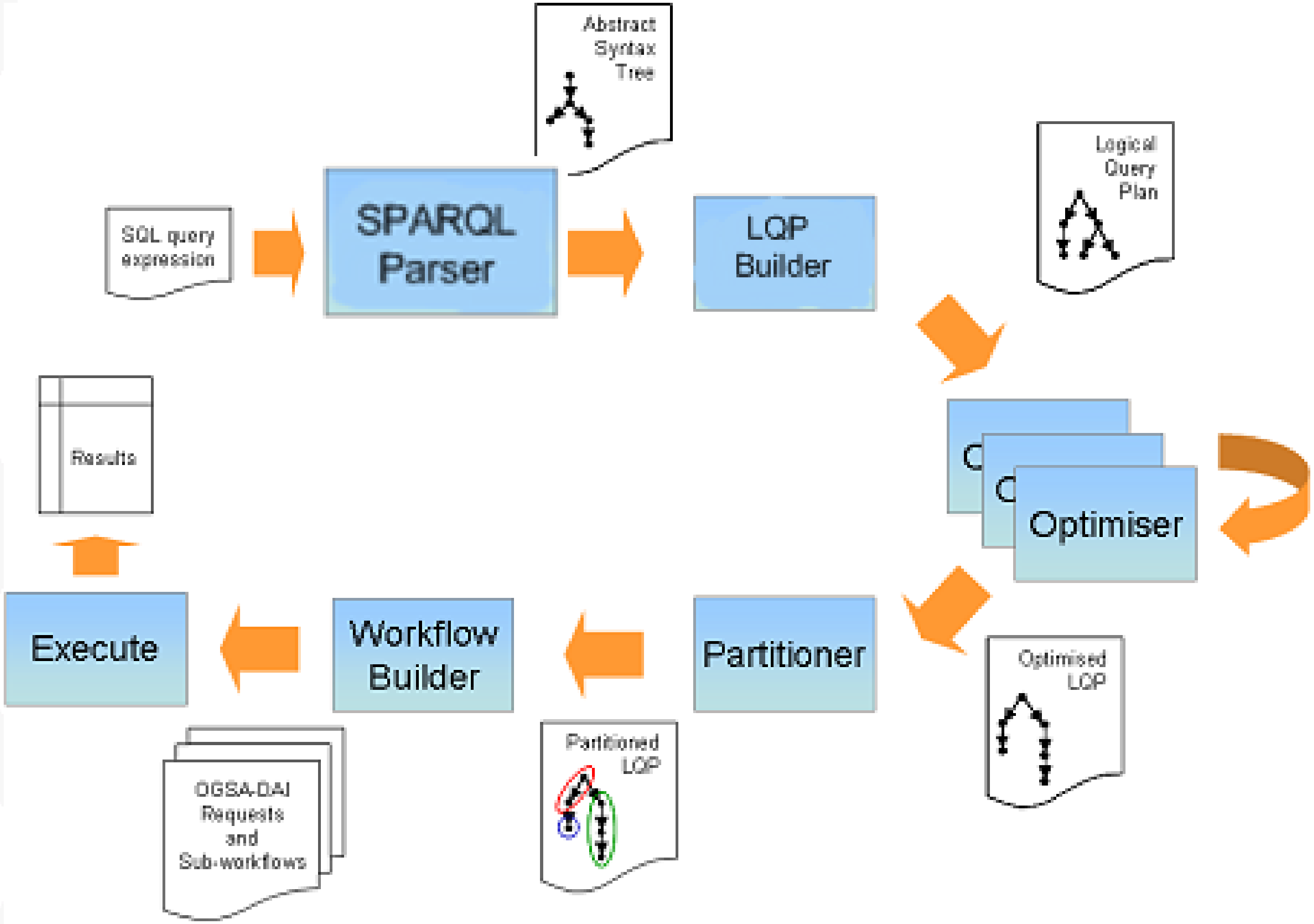
- Proposal:
  - Extend OGSA-DQP with a new query language: SPARQL
  - SPARQL is “similar” to SQL
    - Both have the same expressive power
    - It is possible to use DQP operators for SPARQL
  - Current status: optimising simple SPARQL queries

```
PREFIX p: <http://dbpedia.org/property/>
SELECT ?dbpediaResource.player ?RDFResource.club
FROM dbpediaResource: <http://dbpedia.org/sparql/>
FROM RDFResource: <http://dbpedia.org/sparql/>
WHERE{
      ?dbpediaResource.player p:cityofbirth
<http://dbpedia.org/resource/Stryn>.
      OPTIONAL {?RDFResource.player p:currentclub  ?RDFResource.club}}
```

SparqlDQP workflow

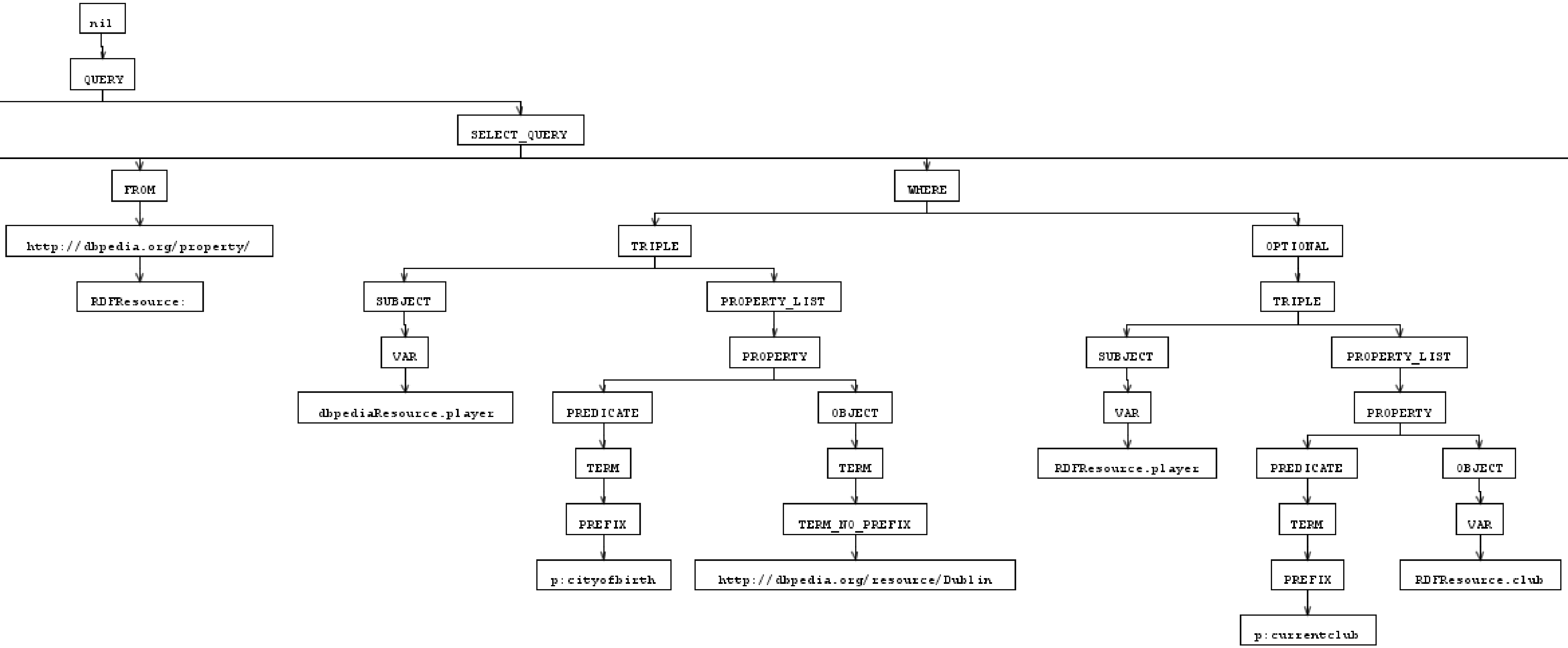


SparqlDQP workflow

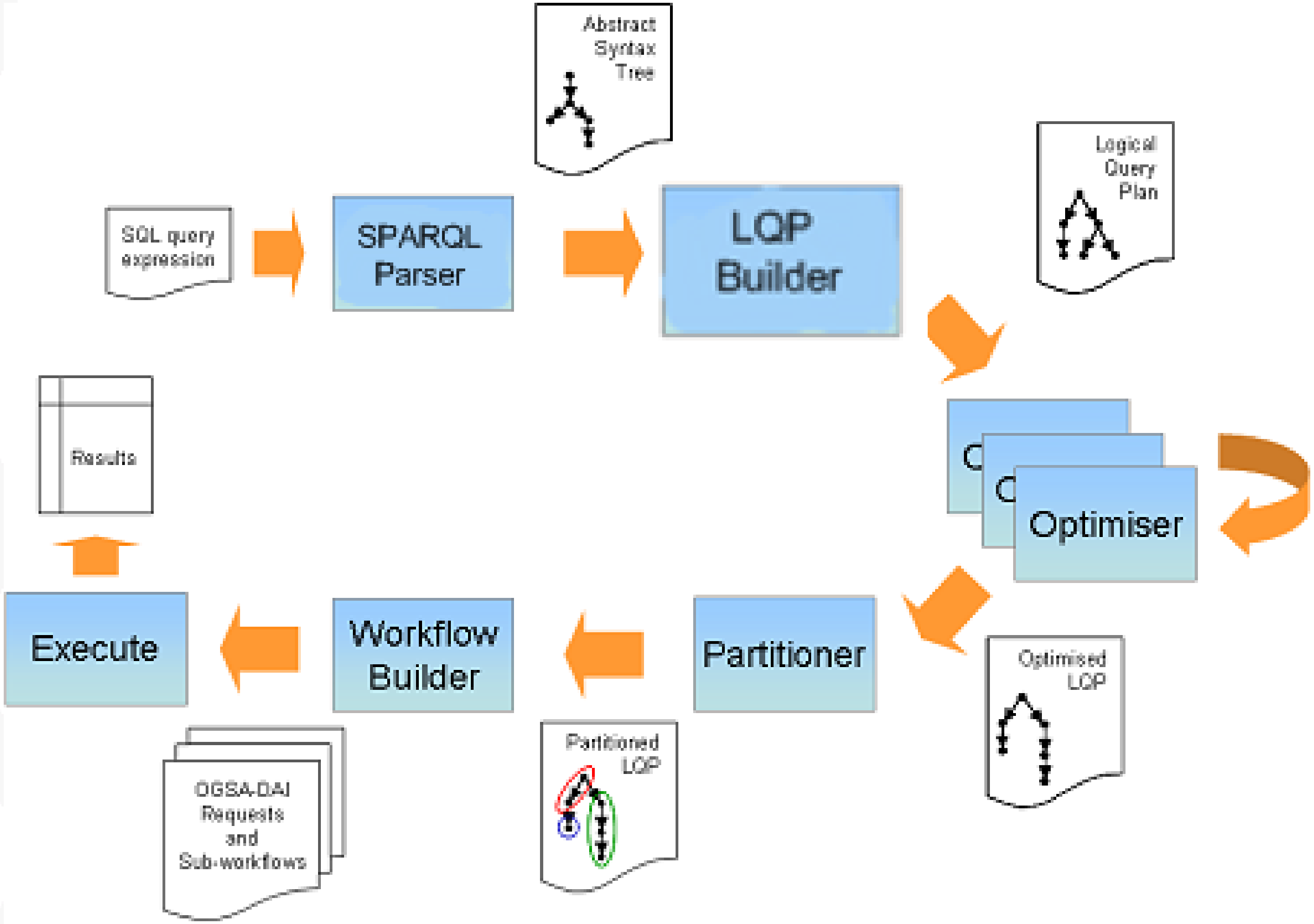


Sparql Parser

- Based on Sparql-g Grammar (which is based on the WWW Sparql recommendation)
- Use of Antlr to create an AST from the grammar

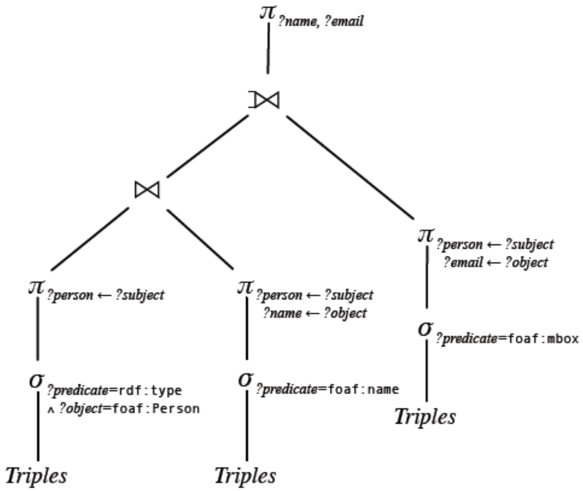


SparqlDQP workflow

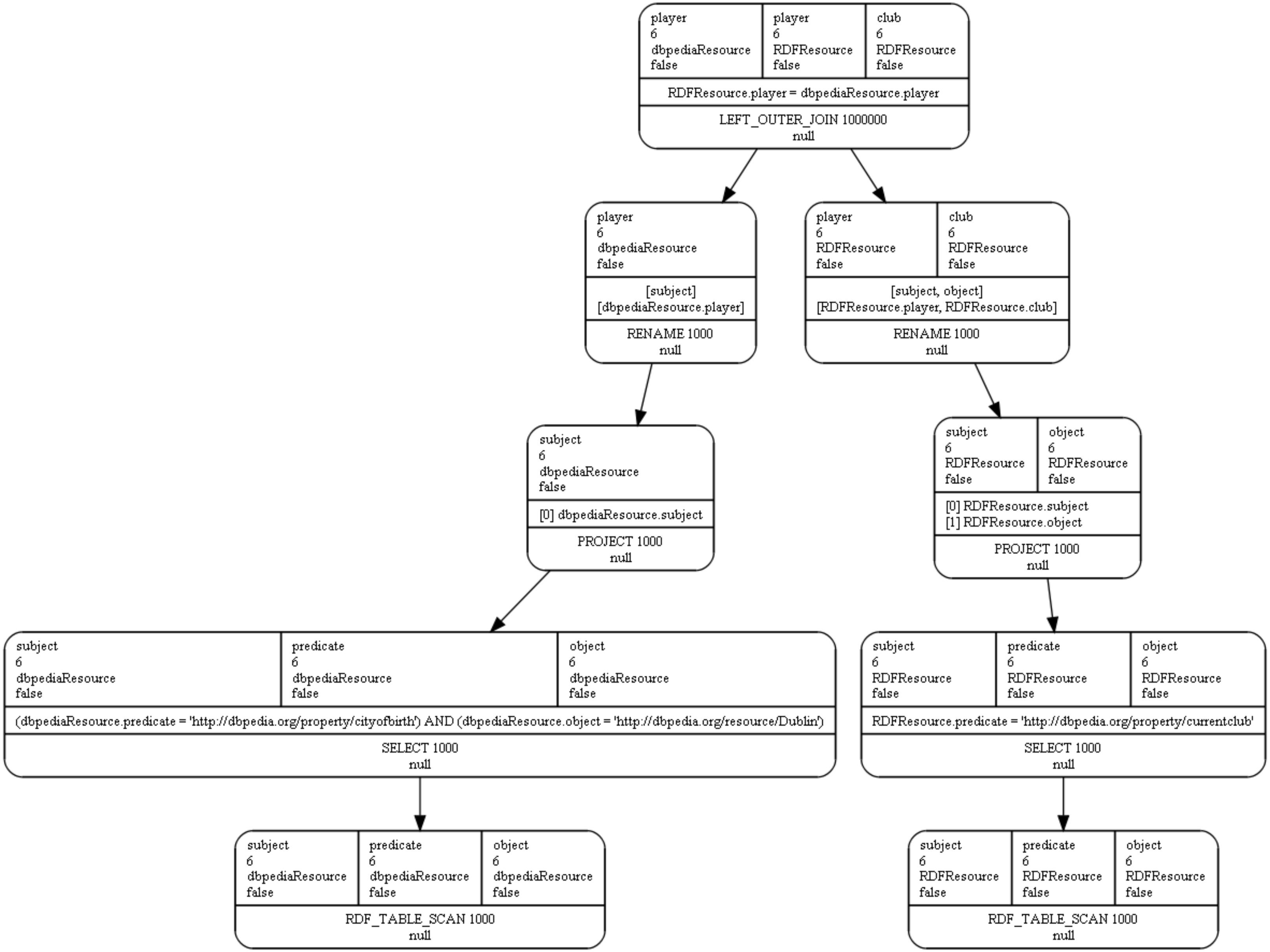


Sparql LQP Builder

- The LQP builder crawls the AST and builds an LQP
- The Sparql LQP builder uses OGSA-DQP operators
  - RDF Table Scan Operator (not DQP)
  - DQP (SQL) Select Operator
  - DQP (SQL) Project Operator
  - DQP (SQL) Inner theta join (with an equality) for triple patterns
  - DQP (SQL) Left Outer Join for OPTIONAL


π ?name, ?email
⋈
⋈
π ?person ← ?subject
σ ?predicate=rdf:type ∧ ?object=foaf:Person
Triples
π ?person ← ?subject ?name ← ?object
σ ?predicate=foaf:name
Triples
π ?person ← ?subject ?email ← ?object
σ ?predicate=foaf:mbox
Triples

Sparql LQP Builder Output



player
6
dbpediaResource
false

player
6
RDFResource
false

club
6
RDFResource
false

RDFResource.player = dbpediaResource.player

LEFT_OUTER_JOIN 1000000
null

player
6
dbpediaResource
false

[subject]
[dbpediaResource.player]

RENAME 1000
null

player
6
RDFResource
false

club
6
RDFResource
false

[subject, object]
[RDFResource.player, RDFResource.club]

RENAME 1000
null

subject
6
dbpediaResource
false

[0] dbpediaResource.subject

PROJECT 1000
null

subject
6
RDFResource
false

object
6
RDFResource
false

[0] RDFResource.subject
[1] RDFResource.object

PROJECT 1000
null

subject
6
dbpediaResource
false

predicate
6
dbpediaResource
false

object
6
dbpediaResource
false

(dbpediaResource.predicate = 'http://dbpedia.org/property/cityofbirth') AND (dbpediaResource.object = 'http://dbpedia.org/resource/Dublin')

SELECT 1000
null

subject
6
RDFResource
false

predicate
6
RDFResource
false

object
6
RDFResource
false

RDFResource.predicate = 'http://dbpedia.org/property/currentclub'

SELECT 1000
null

subject
6
dbpediaResource
false

predicate
6
dbpediaResource
false

object
6
dbpediaResource
false

RDF_TABLE_SCAN 1000
null

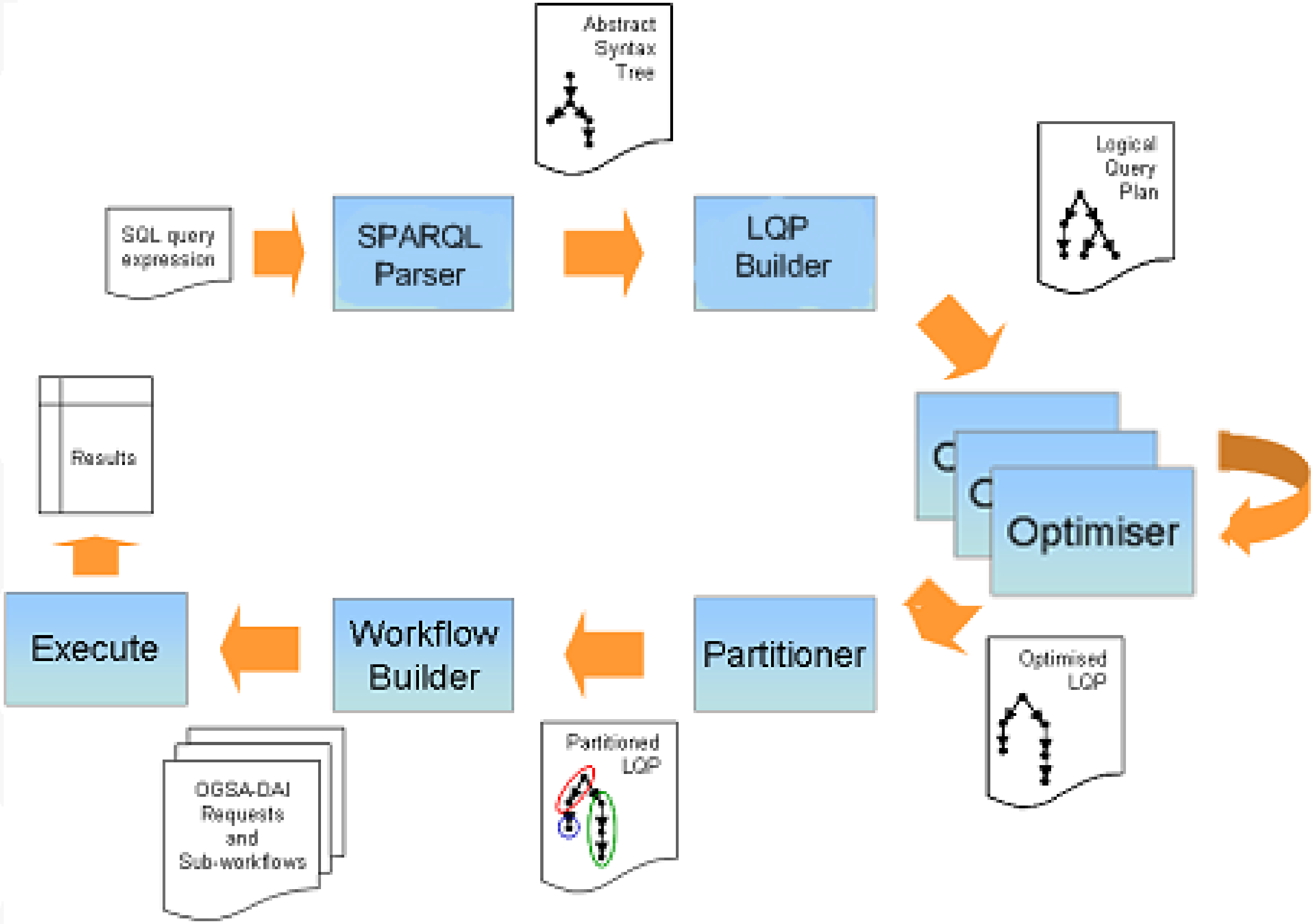
subject
6
RDFResource
false

predicate
6
RDFResource
false

object
6
RDFResource
false

RDF_TABLE_SCAN 1000
null

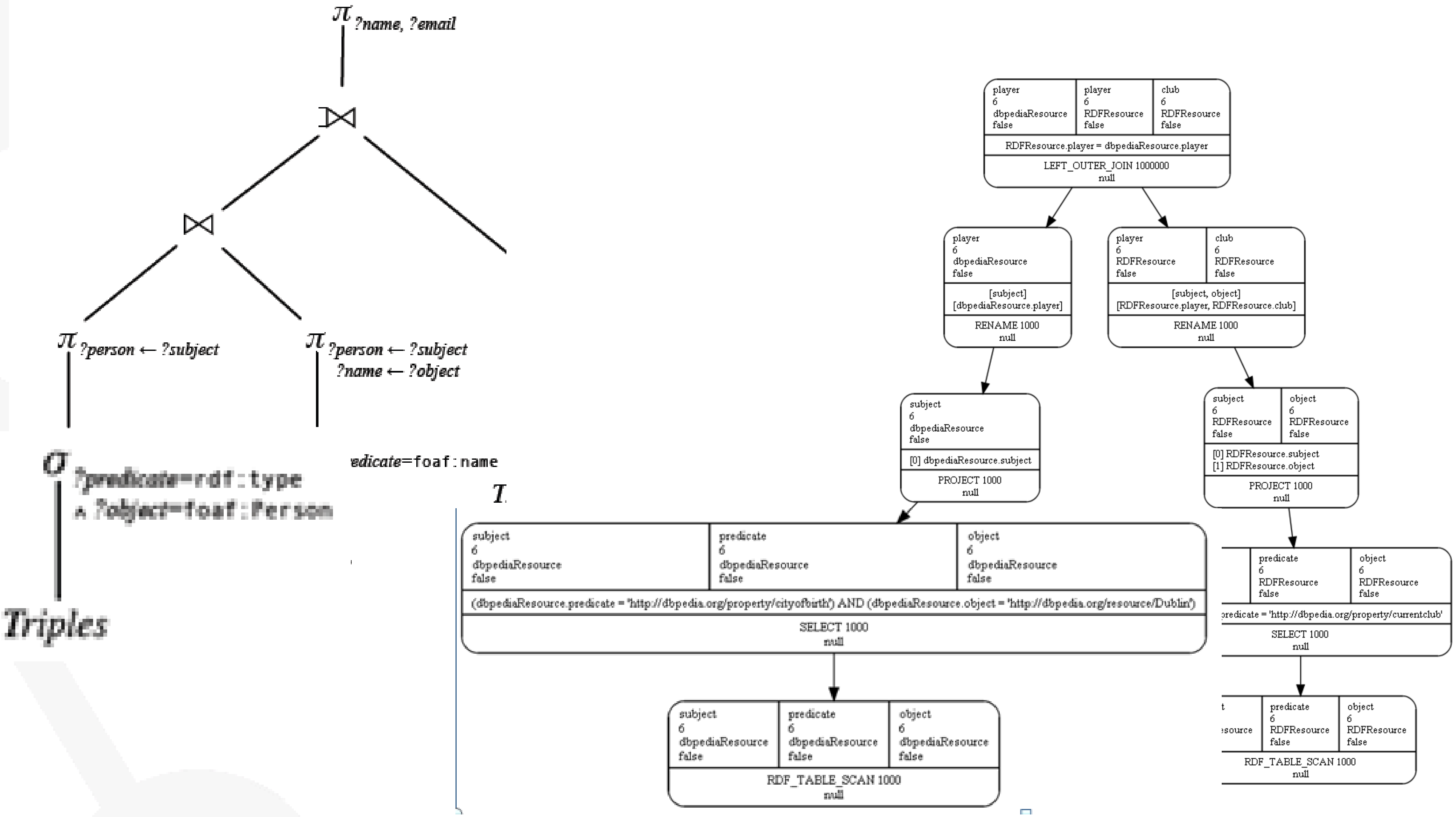
SparqlDQP workflow



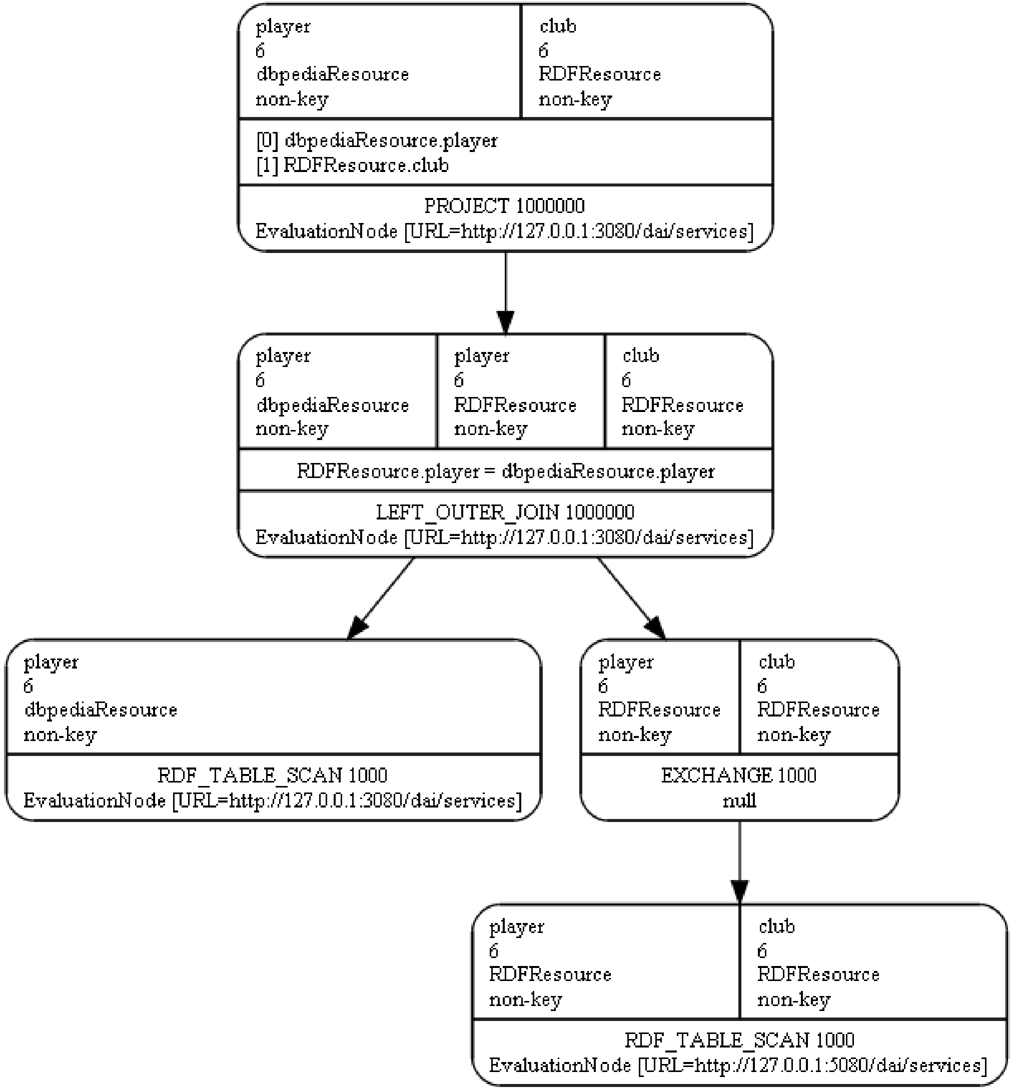
Optimisers

- Configuration file in which the existing optimisers are defined
  - Executed in a specific order
  - Currently running
    - Query normaliser
    - Partitioning Optimiser
    - RDF Table Scan implosion optimiser
- RDF Table Scan implosion optimiser
  - Merges the initial RDF scan, select and project

SparqlDQP optimisations


π ?name, ?email
⋈
⋈
π ?person ← ?subject
π ?person ← ?subject
?name ← ?object
σ ?predicate=rdf:type
∧ ?object=foaf:Person
edicate=foaf:name
T
Triples
player 6 dbpediaResource false
player 6 RDFResource false
club 6 RDFResource false
RDFResource.player = dbpediaResource.player
LEFT_OUTER_JOIN 1000000 null
player 6 dbpediaResource false
[subject] [dbpediaResource.player]
RENAME 1000 null
player 6 RDFResource false
club 6 RDFResource false
[subject, object] [RDFResource.player, RDFResource.club]
RENAME 1000 null
subject 6 dbpediaResource false
[0] dbpediaResource.subject
PROJECT 1000 null
subject 6 RDFResource false
object 6 RDFResource false
[0] RDFResource.subject
[1] RDFResource.object
PROJECT 1000 null
subject 6 dbpediaResource false
predicate 6 dbpediaResource false
object 6 dbpediaResource false
(dbpediaResource.predicate = 'http://dbpedia.org/property/cityofbirth') AND (dbpediaResource.object = 'http://dbpedia.org/resource/Dublin')
SELECT 1000 null
predicate 6 RDFResource false
object 6 RDFResource false
predicate = 'http://dbpedia.org/property/currentclub'
SELECT 1000 null
subject 6 dbpediaResource false
predicate 6 dbpediaResource false
object 6 dbpediaResource false
RDF_TABLE_SCAN 1000 null
t
esource
predicate 6 RDFResource false
object 6 RDFResource false
RDF_TABLE_SCAN 1000 null

Optimised LQP



player
6
dbpediaResource
non-key

club
6
RDFResource
non-key

[0] dbpediaResource.player
[1] RDFResource.club

PROJECT 1000000
EvaluationNode [URL=http://127.0.0.1:3080/dai/services]

player
6
dbpediaResource
non-key

player
6
RDFResource
non-key

club
6
RDFResource
non-key

RDFResource.player = dbpediaResource.player

LEFT_OUTER_JOIN 1000000
EvaluationNode [URL=http://127.0.0.1:3080/dai/services]

player
6
dbpediaResource
non-key

RDF_TABLE_SCAN 1000
EvaluationNode [URL=http://127.0.0.1:3080/dai/services]

player
6
RDFResource
non-key

club
6
RDFResource
non-key

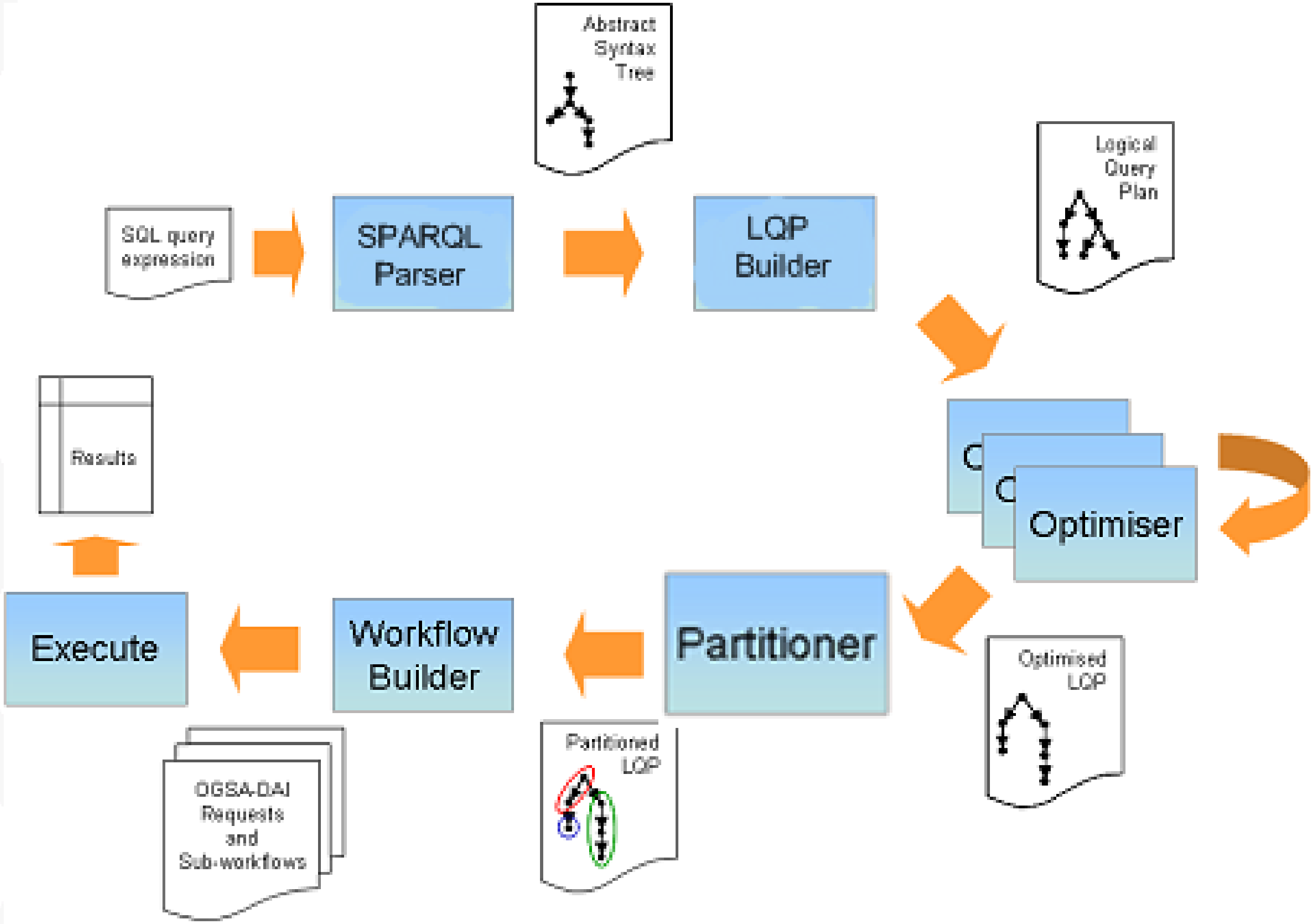
EXCHANGE 1000
null

player
6
RDFResource
non-key

club
6
RDFResource
non-key

RDF_TABLE_SCAN 1000
EvaluationNode [URL=http://127.0.0.1:5080/dai/services]

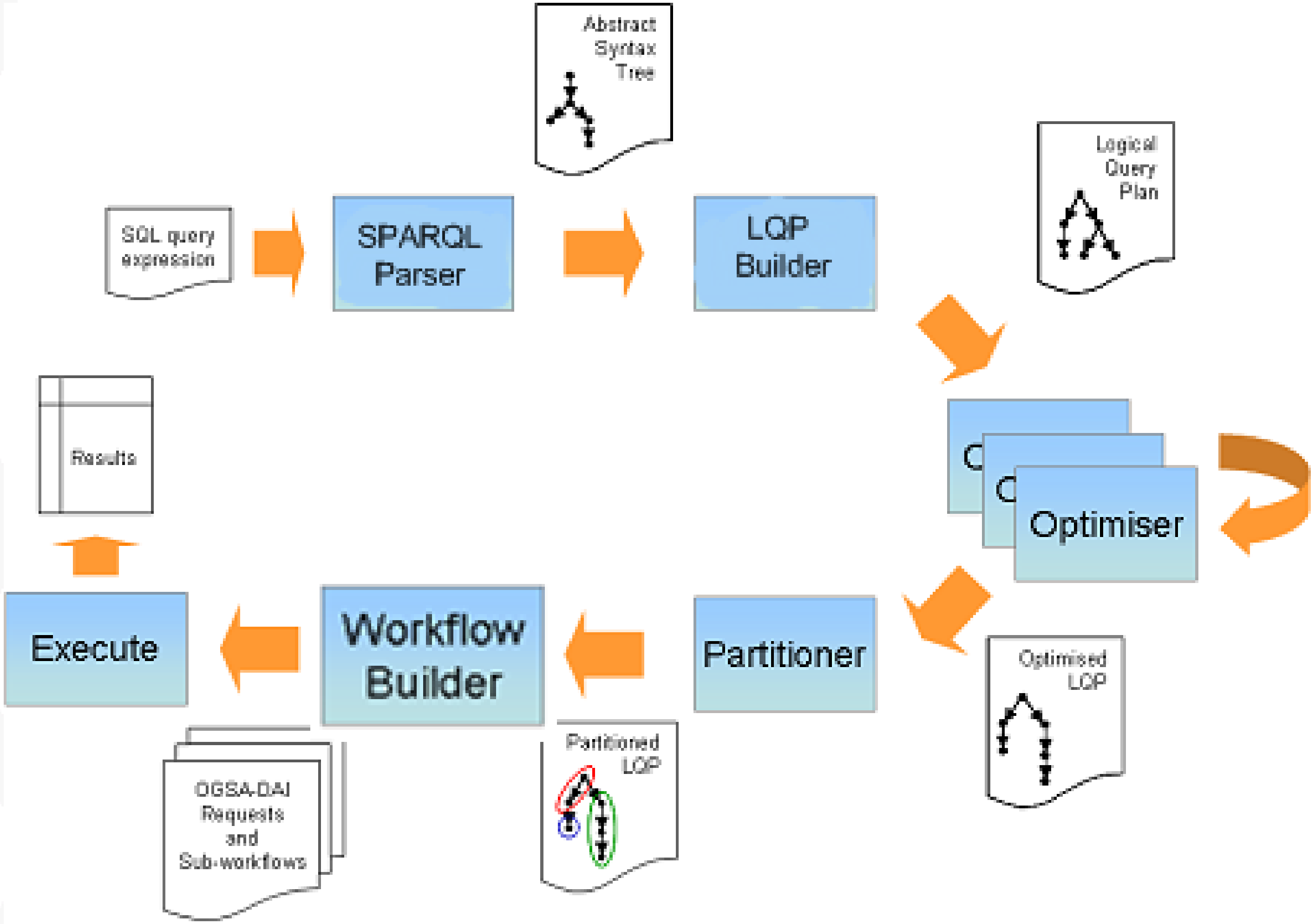
SparqlDQP workflow



Partitioner

- Partitions the SparqlLQP in the available nodes
- OGSA-DQP knows all the nodes
  - IP
  - Resources in the node
  - The partitioner is included in the optimisers chain
- OGSA-DQP knowing the LQP and the nodes creates an execution plan which is distributed across these nodes
- Important:
  - OGSA-DQP for SQL has a dictionary with the SQL database statistics
  - For SPARQL there are missing key statistics for improving the optimisations

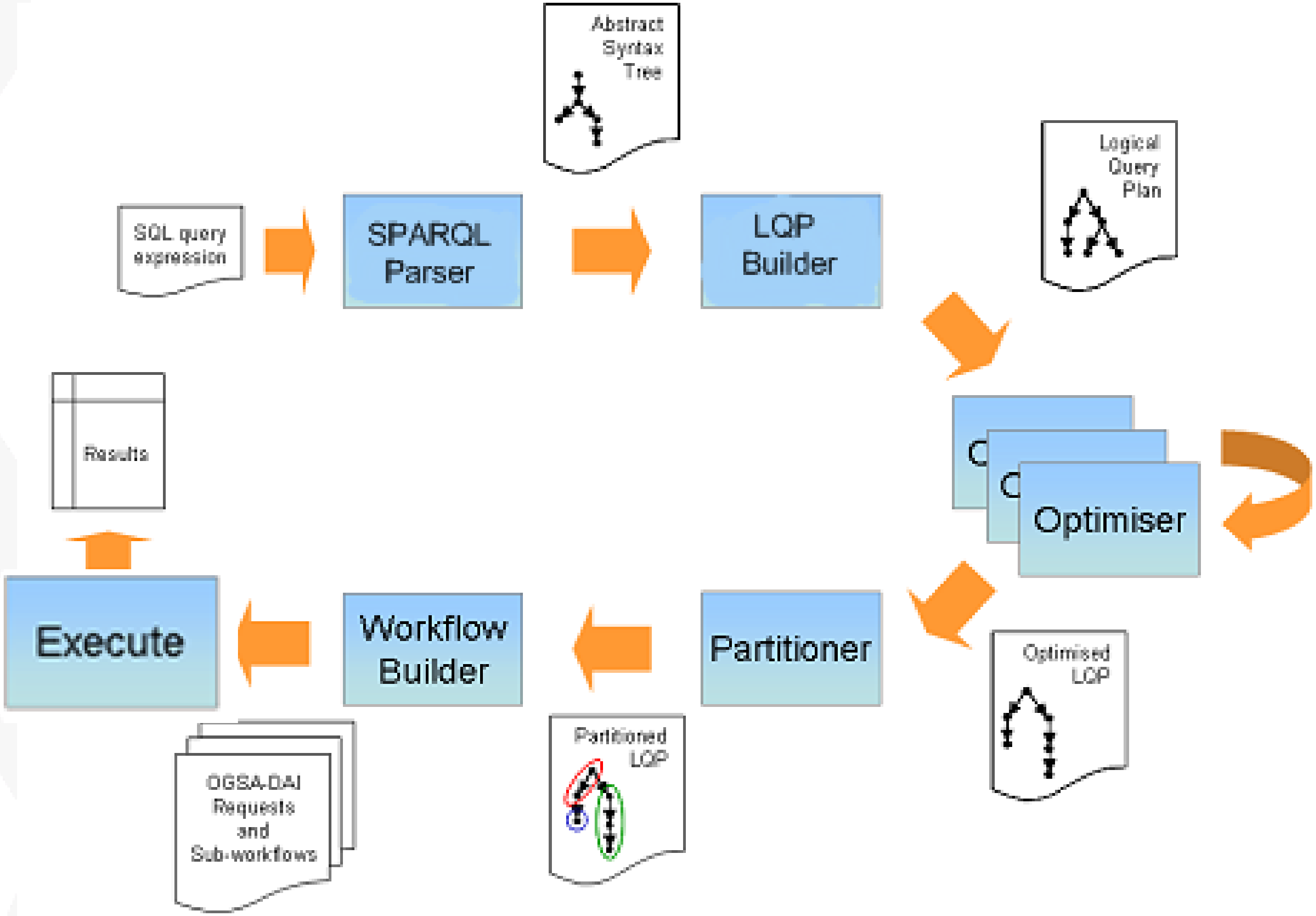
SparqlDQP workflow



Workflow Builder

- At this stage an executable workflow is created
  - It is created from the optimised LQP
- It uses the OGSA-DAI activity SQLQueryActivity
  - We do not modify anything from this activity
  - The Sparql LQP is sent to the SQL activity
  - And a workflow is created
  - This is possible because the RDF resource (which runs the final query) is totally integrated within OGSA-DAI & DQP
  - Result formats between activities accessing resources are the same
  - Possibility of creating data workflows combining RDF data and SQL data

SparqlDQP execution



The results

- Problems with the results retrieved from DBPedia: I do not get all of them, limited to a certain amount, solution on its way

```
(previous) query:
PREFIX p: <http://dbpedia.org/property/>
SELECT ?dbpediaResource.player ?RDFResource.club
FROM dbpediaResource: <http://dbpedia.org/property/>
FROM RDFResource: <http://dbpedia.org/property/>
WHERE{
      ?dbpediaResource.player p:cityofbirth
<http://dbpedia.org/resource/Stryn>.
      OPTIONAL {?RDFResource.player p:currentclub  ?RDFResource.club}}
```

Results

```
Results:
player - club -
http://dbpedia.org/resource/Jarle_Flo - null -
http://dbpedia.org/resource/H%C3%A5vard_Flo - null -
http://dbpedia.org/resource/Tore_Andr%C3%A9_Flo - null -
http://dbpedia.org/resource/Jostein_Flo -  Strømsgodset (director
of football) -
http://dbpedia.org/resource/Per_Egil_Flo - null -
```

Table of Contents

- Introduction
- OGSA-DAI & OGSA-DQP
- RDF Resource & SparqlDQP
- **Future work**
- Way of working at
  - EPCC
  - NeSC
- Conclusions

Future work (I)

- Improve the operators used by Sparql DQP
  - isURI
  - Regex
  - notBound
  - etc.
- Solve DBPedia problem
- Add statistics to SparqlDQP
- Use several RDF repositories
  - Currently only using DBPedia
  - Use Web services like http://sameas.org/ to obtain and link several RDF repositories
- Complete the SPARQL grammar

Future Work (II)

- Study the semantics of Sparql to create new optimisers
- Create new optimisers based:
  - On SPARQL
  - How SQL optimisers work in SPARQL?
- Create data workflows using other OGSA-DAI resources
- Integrate in one single query queries to RDF stores and SQL DB?

Future work (III)

- Solve problems with blank nodes
  - Example: query several foaf profiles
  - Blank nodes used as identifiers
  - Claudio Gutierrez’s group is working in exactly this problem
  - Already contacted for information

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?person ?name ?mbox WHERE
  { ?person foaf:name ?name .
    OPTIONAL { ?person foaf:mbox ?mbox}
  }
```

Table of Contents

- Introduction
- OGSA-DAI & OGSA-DQP
- RDF Resource & SparqlDQP
- Future work
- **Way of working at**
  - **EPCC**
  - **NeSC**
- Conclusions

Way of working at EPCC

- EPCC (Edinburgh Parallel Computing Centre)
- Is a software production centre mainly
- OGSA-DAI example
  - The team is focused on a product
  - The leader of the team (Ally Hume) specifies a set of actions
  - The team executes the actions
  - Example: performance was poor in a specific type of query (median filter)
    - Yourkit4java for performance measures
    - Find the problem
    - Try to solve it
    - All the team (about 6 people working on that) was very united

Way of working at NeSC

- National eScience Centre
- More academic focused than EPCC
- Every week session on what papers are we going to publish, ideas, papers published and deadlines
- More research focused
  - Example: talking to people from different groups
  - EPCC focused on the poor performance of the automatic composition of data mining workflows in ADMIRE
    - There are going to be many problems there
  - NeSC focused on wow! We are getting nice results and we will publish them!

Table of Contents

- Introduction
- OGSA-DAI & OGSA-DQP
- RDF Resource & SparqlDQP
- Future work
- Way of working at
  - EPCC
  - NeSC
- **Conclusions**

Conclusions

- Created a OGSA-DAI resource for accessing RDF data
- Created a Sparql query processor
    - Which distributes and optimises queries
- Used equivalence with SQL operators
    - A complete set of test must be performed
- Found problems using blank nodes
- For optimising Sparql queries is mandatory to understand the semantics of Sparql
- Organisations: Two different ways of working
    - EPCC: production focused
    - NeSC: research/academic focused

SparqlDQP
Stay at EPCC & NeSC

Carlos Buil Aranda

Ontology Engineering Group
Facultad de Informática
Universidad Politécnica de Madrid
cbuil@fi.upm.es

5th November 2009

SparqlDQP Features

- Adapted SPARQL grammar for distributed SPARQL
    - SELECT ?dbpedia.person FROM dbpedia WHERE {...}
- Query plan similar to SQL query plans
    - SQL optimisations can be applied to them
- Current Optimisers
    - Query Normaliser
    - Partitioner
    - RDF Table Scan Implosion
- Automatic creation of data workflows
- Distribution across the OGSA-DAI nodes

Sample Queries

```
PREFIX p: <http://dbpedia.org/property/>
SELECT ?dbpediaResource.player ?RDFResource.club
FROM dbpediaResource: <http://dbpedia.org/sparql>
FROM RDFResource: <http://dbpedia.org/sparql>
WHERE{
         ?dbpediaResource.player p:cityofbirth <http://dbpedia.org/resource/Stryn>.
   ?dbpediaResource.player p:countryofbirth  <http://dbpedia.org/resource/Norway>
         OPTIONAL {?RDFResource.player p:currentclub  ?RDFResource.club}}
```

```
PREFIX p: <http://dbpedia.org/property/>
SELECT ?dbpediaResource.player ?RDFResource.club
FROM dbpediaResource: <http://dbpedia.org/sparql>
FROM RDFResource: <http://dbpedia.org/sparql>
WHERE{" +
         ?dbpediaResource.player p:cityofbirth <http://dbpedia.org/resource/Dublin>.
         OPTIONAL {?RDFResource.player p:currentclub  ?RDFResource.club}}
```

Way of working at EPCC

- Software production Centre
  - Focused on a software product
  - High performance test