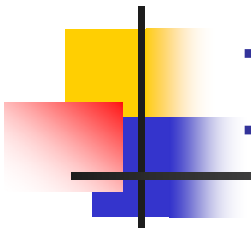
OGSA-DQP: A Service-Based Distributed Query Processor for the Grid

M Nedim Alpdemir[1], A Mukherjee[2], A Gounaris[1], Alvaro A A Fernandes[1], Norman W Paton[1], Paul Watson[2], J Smith[2]

[1] Department of Computer Science
University of Manchester

[2] Department of Computing
University of Newcastle-Upon-Tyne

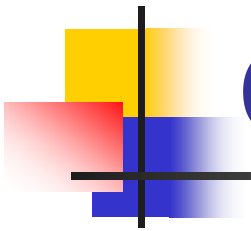


Introduction

1. High-level data access and integration services are needed if applications that have data with complex structure and complex semantics are to benefit from the Grid.

2. Standards for data access are emerging, and middleware products that are reference implementations of such standards are already available.

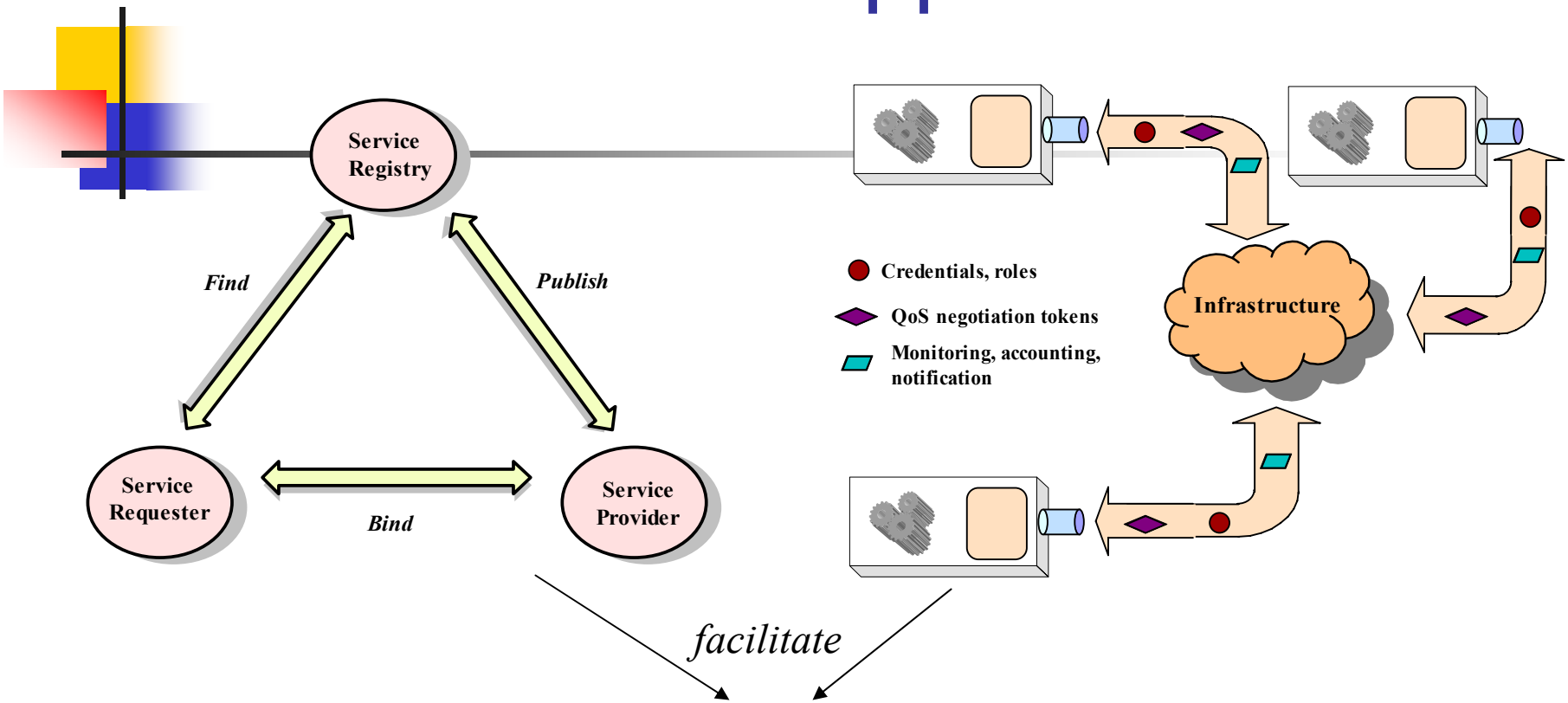
3. Distributed query processing technology is one approach to meeting (1.) given the availability of (2.).



Goals

- To expose an integrated approach to accessing and analysing data as a Grid service;
- To exploit Grid abstractions for on-demand allocation of resources required for a task;
- To provide transparent, implicit support for parallelism and distribution;
- To provide homogeneous access to heterogeneous data sources.

Service-based approaches

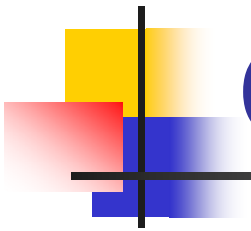


facilitate

Virtualisation of Resources

Leads to

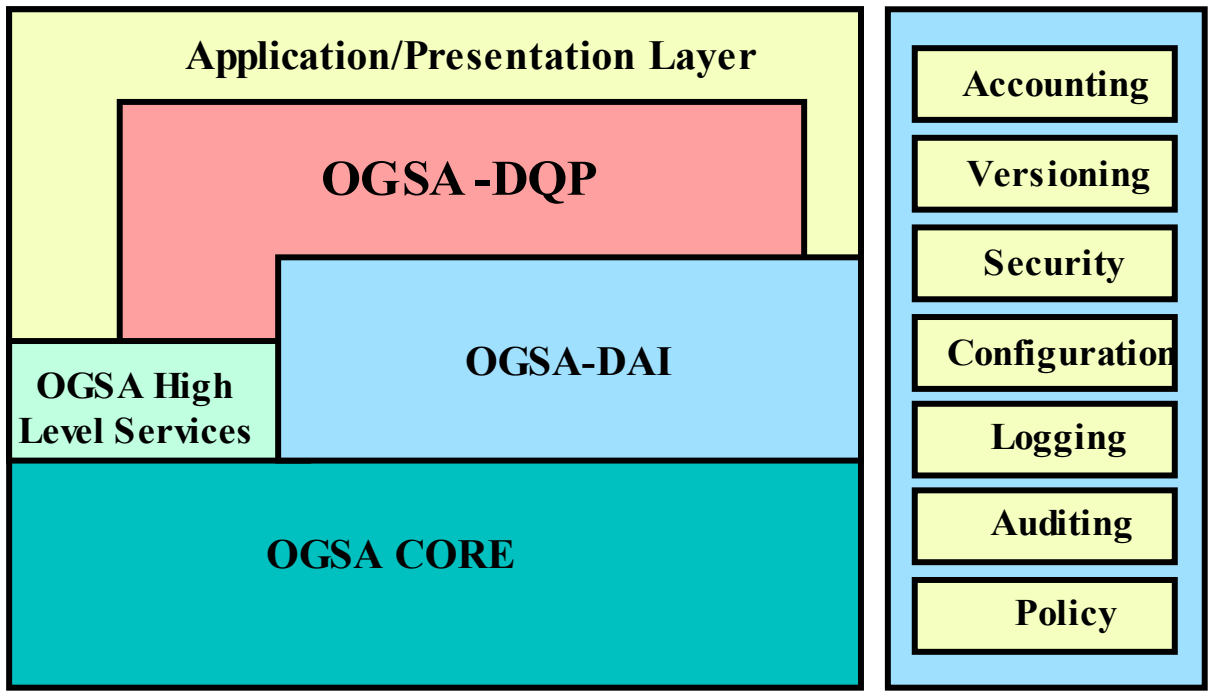
A convenient cooperation model for Distributed Systems (e.g. Grid)



OGSA-DQP is Service-Based

- Service-based in two orthogonal sense:
  - Supports querying over *data storage* and *analysis* resources made available as *services*
  - *Construction* of distributed query plans and their *execution* over the grid are factored out as services

Where Does OGSA-DQP Fit ?



Properties of OGSA-DQP

- Supports queries over GDSs and over other services available on the Grid, thereby combining data access with analysis;
- Uses the facilities of the OGSA to dynamically obtain the resources necessary for efficient evaluation of a distributed query;
- Adapts techniques from parallel databases to provide implicit parallelism for complex data intensive requests
- Uses the emerging standard for GDSs to provide consistent access to database metadata and to interact with databases on the Grid.

OGSA-DQP Extends OGSA-DAI …

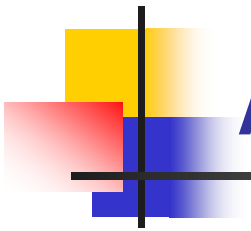
- By adding a new port type and two new services (and their corresponding factories) :
  - Grid Distributed Query (GDQ) Port Type
    - **importSchema operation**

```
GDQSR importSchema(GDQDataSourceList GDSL)
```

      - ***GDSL*** **: A document containing the list of Data Sources. The items on this list should contain the handles of the GDSFactories and URLs of service WSDLs**

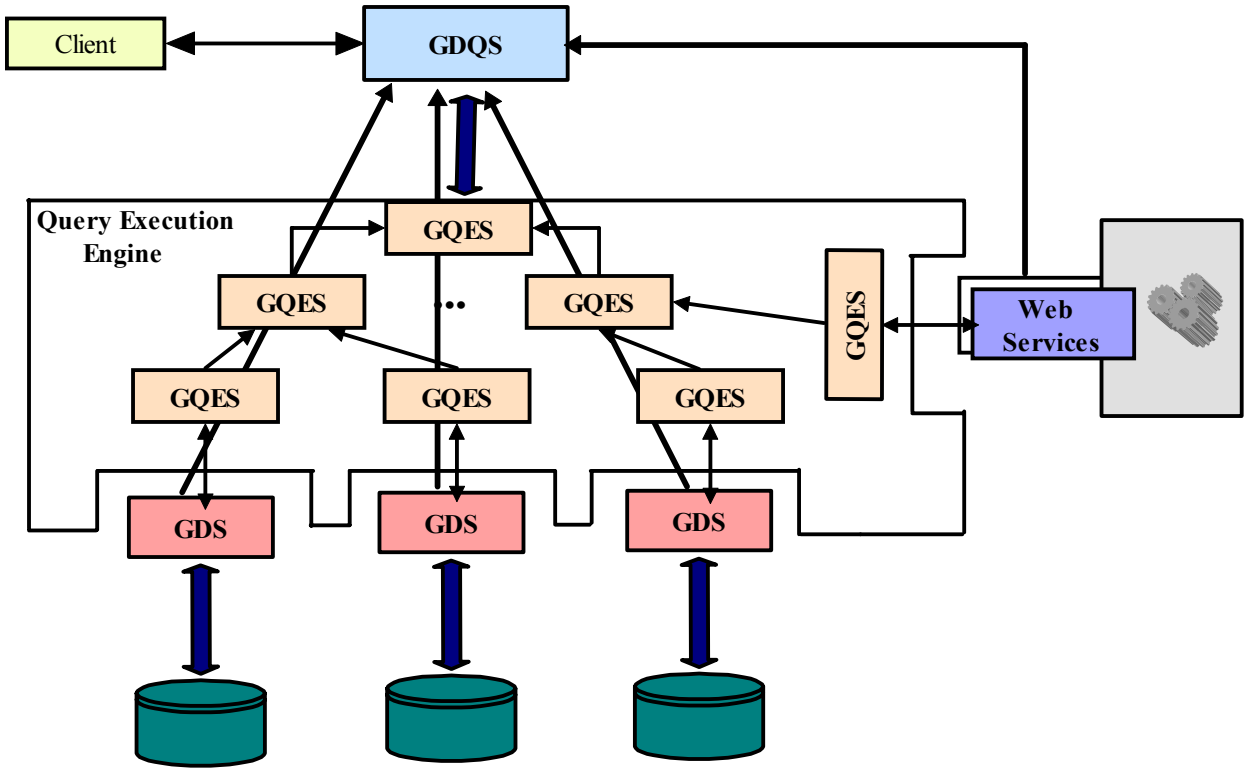
Two services : GDQS & GQES

- **Grid Distributed Query Service (GDQS)**
  - **Wraps a Query Compiler engine (Polar*) which compiles, optimises, partitions and schedules distributed query execution plans**
  - **Obtains metadata and computational resource information required for above**
  - **Creates and coordinates QQESs according to the optimised plan**
- **Grid Query Evaluator Service (GQES)**
  - **Each GQES instance is an execution node**
  - **A GQES is in charge of a partition of the query execution plan assigned to it by the GDQS**



A typical query session…

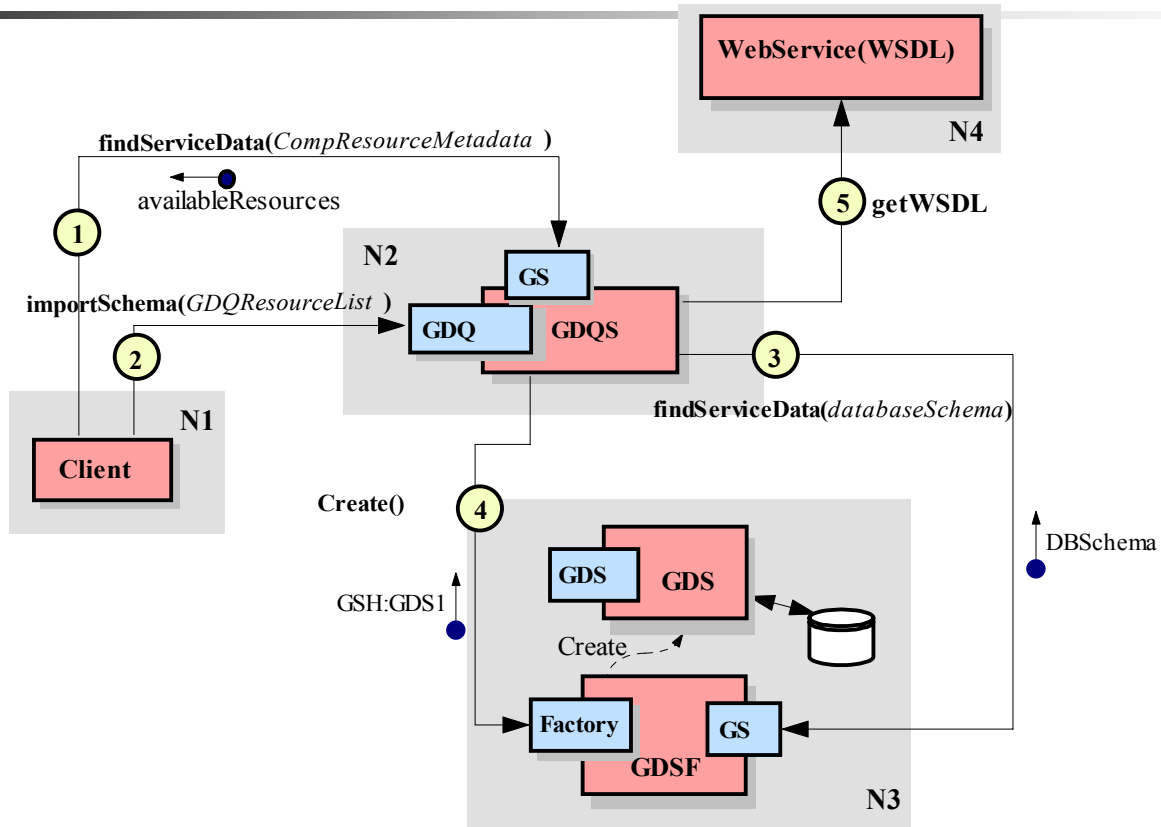
- Starts with when a GDQS instance is created
- Involves an initial set-up Phase
- Continues with Multiple query requests
- Ends when the GDQS instance is destroyed

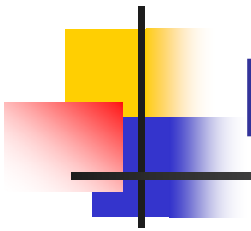

Client
GDQS
Query Execution Engine
GQES
GQES
GQES
GQES
GQES
GQES
GQES
Web Services
GDS
GDS
GDS

Setting up a GDQS

- Set-up strategy depends on the life-time model of GDQS and GDSs
  - GDQS instance is created per-client
  - But it can serve multiple-queries
  - This model avoids complexity of multi-user interactions while ensures that the set-up cost is not high
- Setup phase involves:
  - Importing schemas of participating data sources
  - Importing WSDL documents of participating analysis services
  - Collecting computational resource metadata (implicit)

Importing Schemas



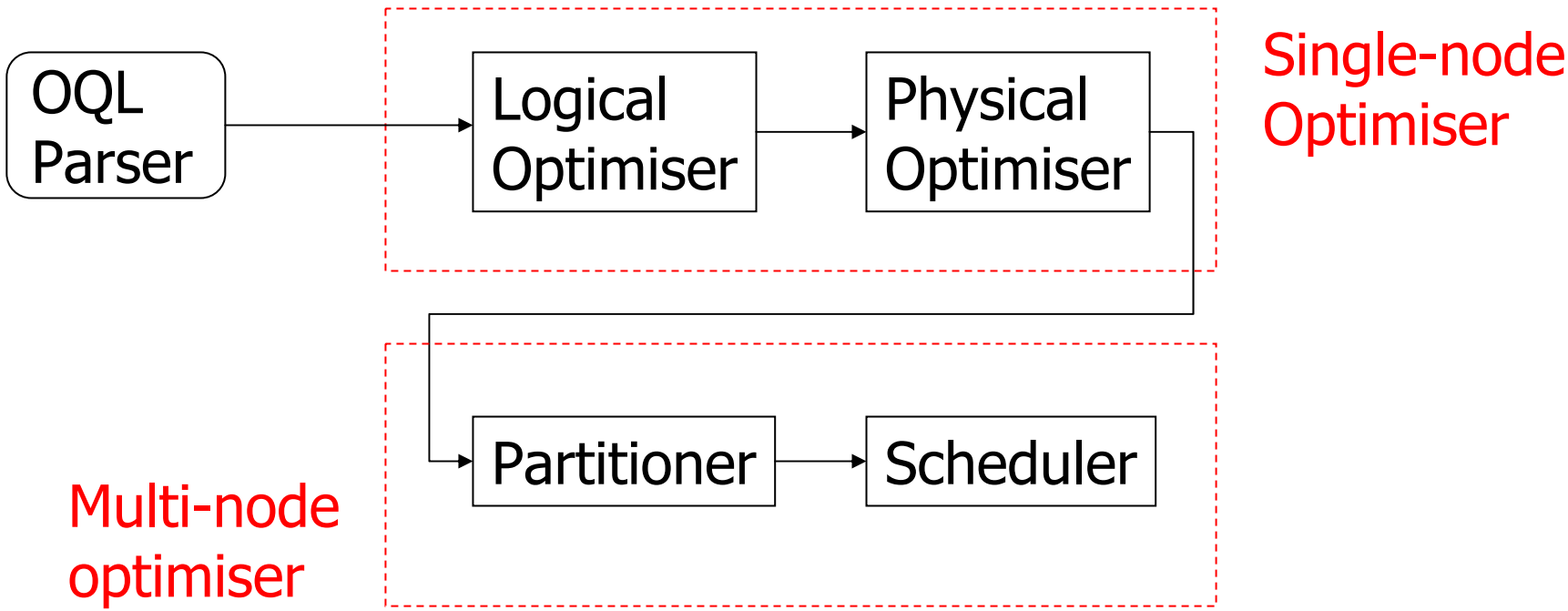


How does a query gets evaluated

An example query :

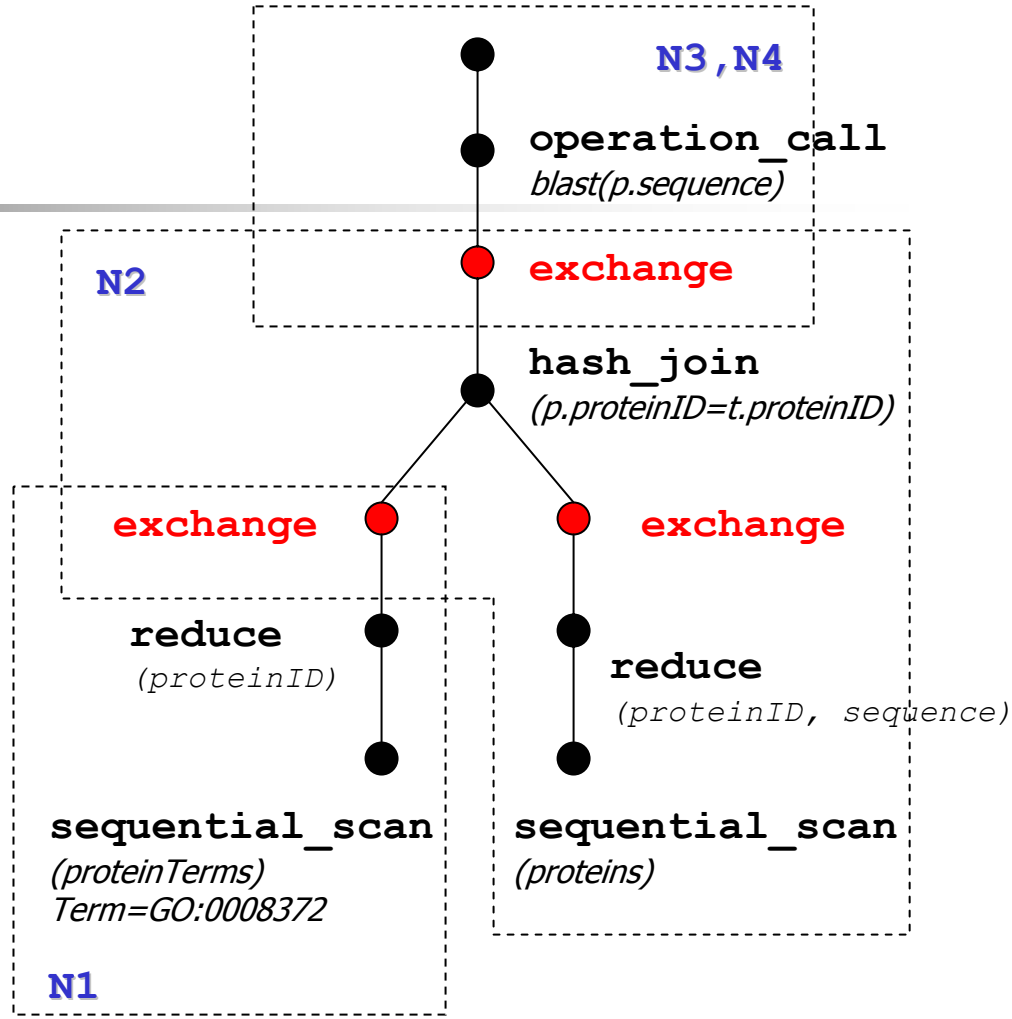
```
select p.proteinId, Blast(p.sequence)
from   proteins p, proteinTerms t
where  p.proteinId = t.proteinId and
          t.termId = 'GO:0005942 '
```

Query Compilation

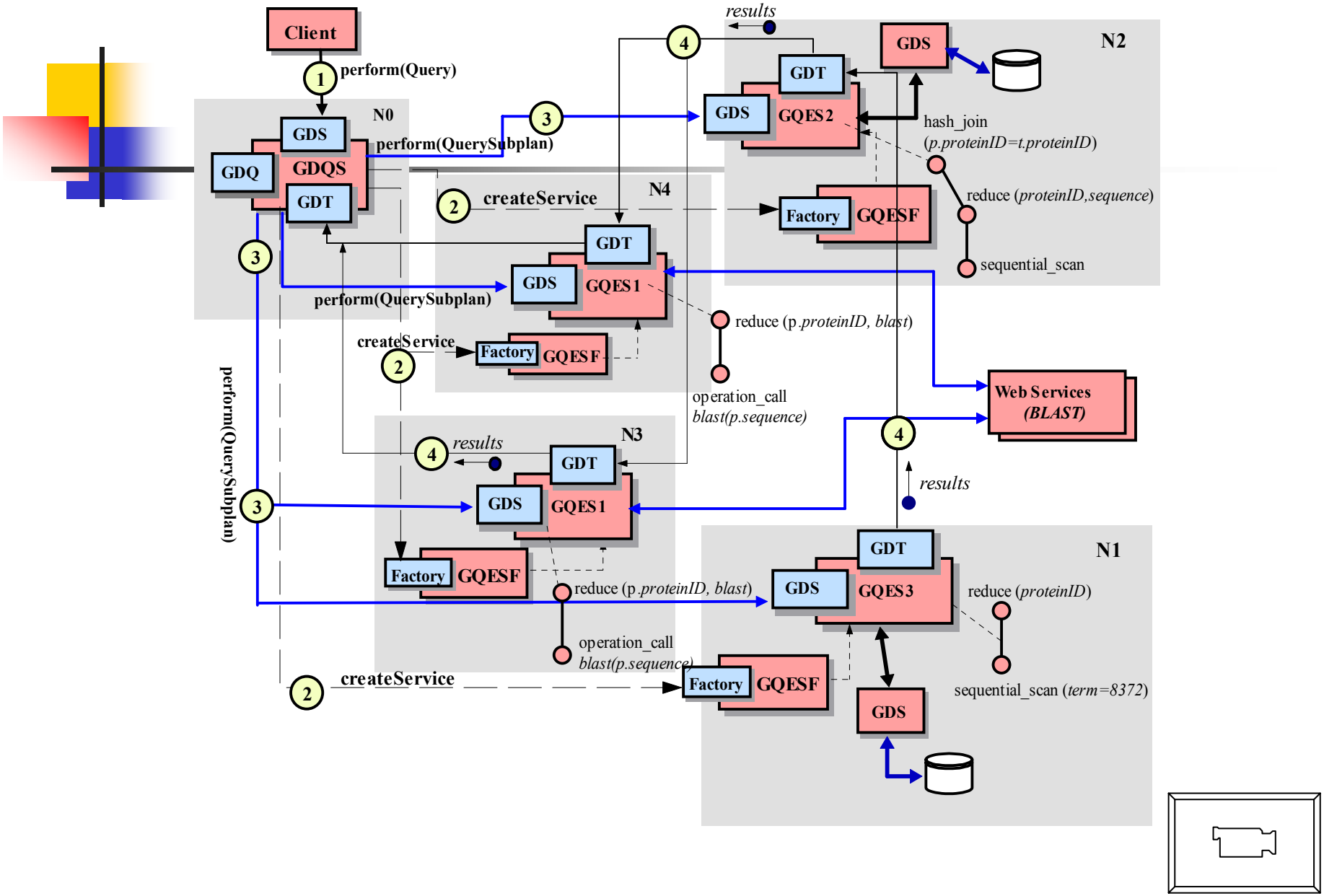


The final Query Plan

- Partitions are assigned to Grid nodes.
- Expressed by decorating parallel algebra expression.



(d) Scheduled plan

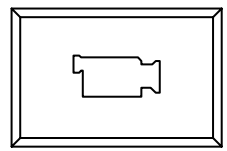

Client
1
perform(Query)
N0
GDS
GDQ
GDQS
GDT
perform(QuerySubplan)
3
2
createService
N4
GDT
GDS
GQES1
Factory
GQESF
reduce (p.proteinID, blast)
operation_call
blast(p.sequence)
results
4
N2
GDS
GDT
GDS
GQES2
hash_join
(p.proteinID=t.proteinID)
reduce (proteinID,sequence)
Factory
GQESF
sequential_scan
Web Services
(BLAST)
N3
GDT
GDS
GQES1
Factory
GQESF
reduce (p.proteinID, blast)
operation_call
blast(p.sequence)
N1
GDT
GDS
GQES3
reduce (proteinID)
sequential_scan (term=8372)
Factory
GQESF
GDS

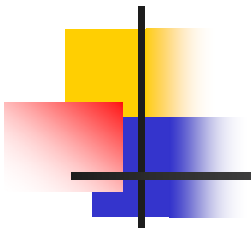
Comparison to other similar approaches

- Classical wrapper-mediator architectures address similar concerns with SB-DQP (I.e. provide declarative query support over dist. Data stores and Analysis tools) but:
  - Limited support for dynamic resource discovery via metadata queries
  - No dynamic resource allocation
  - They require custom wrappers (to interface the query engine to data sources) whereas OGSA-DQP operates on generic GDSs
  - OGSA-DQP is layered on top of a more flexible execution environment so it can provide partitioned parallelism
  - The interactions between constituting components of OGSA-DQP and external components is governed by standards based architectures (XML, WSDL, SOAP, WS/OGSA)

Continued …

- Workflow languages (WSFL, BPL4WS)
  - Procedural in nature
  - Places significant responsibility on programmers to specify the most appropriate order of execution for a collection of service requests and to obtain adequate resources for the execution of computationally demanding aspects of an app.





- Possible future work
  - Extend supported query types
  - Dynamic acquisition of computational resource metadata
  - Improve performance
  - Schema integration & conflict resolution
- Announcement !!
  - OGSA-DQP prototype release 1.0 is available from www.ogsadai.org.uk/dqp/