InfoGrid: Information resource integration

No Common Language – No Common Data Model

N. Giannadakis, M. Ghanem, Y. Guo

{ng300, mmg, yg}@doc.ic.ac.uk
Imperial College London

Grid computing [3] constitutes the amalgamation of a drive towards the standardization of existing technologies that enable the collaboration of scientists overcoming restrictions of location, distance and compatibility. The aim is to exploit the full potential of resources, computational or informational. We aim to study how a network of distributed and heterogeneous grid resources could attain maximum adaptability to languages, models, scientific uses in a service composition based knowledge discovery environment. Our application areas are bioinformatics and chem.-informatics.

No language or data model assumptions are herein made. The presence of grid layers that can handle basic service registration, etc. tasks, service execution and data mobility is premised. Accordingly, no assumption is made about the nature of the information that grid resources need to communicate. Emphasizing on the composing aspect, we move complexity from the query engine to the composition level by means of utilizing component (Grid Service) composition languages, composing service-based data resources. Service composition now becomes an information resource static integration tool.

The composed information services are essentially composed grid services and hence able to utilize all other available grid services. We assert that that execution logic of such composed programs should be enriched with logic for the provision of metadata. No assumptions should either be made for the type of metadata. Integration languages (distributed query languages) on top of the middleware can have their own requirements. A composed service will simply declare its supported languages or standards. Additional components will be created facilitating and being specialized for information resource composition.

Finally, a dynamic integration layer is proposed whereby an arbitrary integration language can declare its model and grid services which will transform a query expressed in it into a composite service that can be executed and return any results. Any such decomposing grid services will be mapping their operators to the corresponding grid services and ditto with any direct references to other resources.

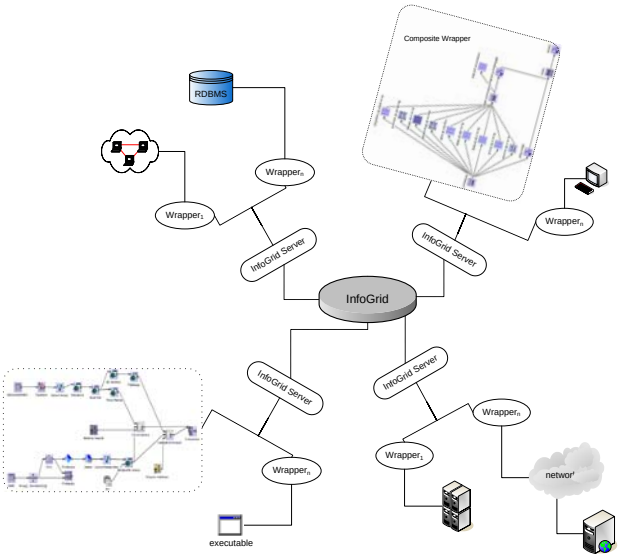

Figure 1. The InfoGrid Architecture. The figure shows the range of services that are offered by the Middleware. These can include the standard "Information Resources" such as databases, public web resources (offered themselves as Grid Services by wrappers), etc., as well as composite Grid Service.

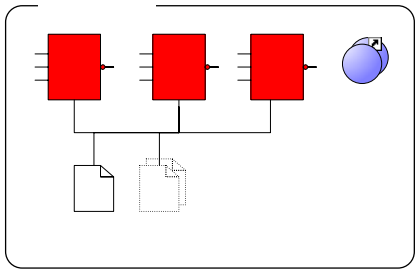

Figure 2. An InfoGrid resource is modeled as an entity with administration, execution and metadata logic. The latter is implemented in composite resources with the additional metadata execution layer.

We can consider an example that is based on two grid information resources:

1. A resource providing access to a relational database *RB*:	2. A BLAST executable *BLST*:
metadata language: ∅ (this means that in order to get the metadata one does not need to provide any parameters) **metadata model**: Relational expressed in RDF [RDF] **invocation language**: SQL **invocation model**: Relational Table	**metadata language**: Invocation parameter –help **metadata model**: Free text **invocation language**: Executable parameters and list of sequences **invocation models**: flat query-anchored, XML Blast output, tabular.

One can use the relational resource (containing biological data) to get interesting DNA sequences and then use the BLAST service for pair-wise comparisons. In the composite graph, sequences are selected from a relational resource table, using selection statements constructed with the 'Accession numbers' supplied as input. The resulting sequences are run through the BLAST resource. The **metadata** of the composition is the set of available 'Accession numbers' as produced by the metadata logic of the composition. The **invocation language** is a set of identifiers and the **model** of the results is the schema of the selected XML BLAST output.

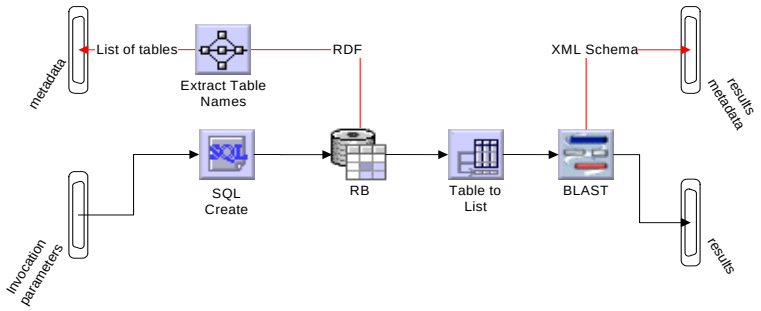

Figure 3. The graph that represents the composition for the example given. The metadata flow is denoted by the red connectors and the data flow by the black.

For an SQL-based integration language the above composition could be expressed as:

SELECT c.[-p blastn -d ecoli -m 7, TABLE2LIST(1, a(0))]
FROM RB:[*select … where a*='AccNum'] a,

Where the invocation of information resource services as well as of the other grid services (TABLE2LIST references the corresponding service) can straightforwardly be decomposed into a graph. In this example, there was no use of assumptions regarding languages or models. This high level approach allows one to delegate tasks, such as the query execution and code mobility, to the grid layers without any low level considerations.

From an information access point, our contribution is the proposed use of service composition and invocation layers as a means of integration with no language or model prerequisites. *From a service composition view*, we propose the introduction of dynamic metadata logic for services, built on top of existent composition capabilities. We aim to demonstrate that this approach can reproduce the integration capabilities of other approaches and, furthermore, that it is able to accommodate more complicated integration needs.

We are designing *InfoGrid* (the information integration effort of Imperial College's DiscoveryNet [4] project) which has the goal of addressing the needs of the scientific

community. In particular, bioinformatics and chemistry applications will constitute the application domains of our research.

Table 1. An overview of the resource characteristics for the basic information access approaches and that of InfoGrid.

Resources				
	JDBC resource	Discovery Link	OGSA-DAI	InfoGrid
Metadata language	Relational Db connection information	Special wrappers have to be available/ metadata is set statically during registration	Available through a Web Service interface	Any
Metadata model	A DatabaseMetaData Java object		XML Schema	Any
Invocation language	SQL	Handled internally by precompiled wrappers	SQL or XQuery	Any
Results model	A ResultSet object		Files (Tables or XML)	Any

Table 2. A summary of the integration approaches. We aim to prove that InfoGrid can reproduce all these.

Integration Approaches				
	JDBC	DiscoveryLink [1]	OGSA-DAI [2]	
Type	Bespoke	Federation	Query Based	Static & query based using the composition layer
Metadata	N/A (individual relational models)	Common Static Relational Model	Individual XML or relational models expressed in XML Schema	Metadata independent Using the
Language	SQL	SQL	OQL	For query based integration all languages that provide a decomposing service
Invocation	Orchestration	Abstracted	Decomposition into execution primitives with the aid of a service dictionary.	Grid execution layer

References

[1] Discovery link from IBM
[2] Open Grid Services Architecture Data Access and Integration (http://www.ogsadai.org.uk/)
[3] Global Grid Forum (http://www.gridforum.org/)
[4] DiscoveryNet project (http://ex.doc.ic.ac.uk/new/)