

Federating Heterogeneous Distributed Information Resources: the Earth Observation from the Space Domain

Stefano Nativi (*Member*)¹, Paolo Mazzetti¹, Lorenzo Bigagli¹, and Dino Giuli² (*Senior Member*)

¹ PIN Scrl – University of Florence, Piazza Ciardi 15, I-59100 Prato, Italy

nativi@pin.unifi.it

² DET, University of Florence, Via S.Marta, 3, I-50139 Florence, Italy

Keywords: Mediator-based Systems, Federated Information Systems, Environmental System Interoperability, Web Services, EOS data integration.

Abstract. We implemented a Mediator-based Federated Information System: a tightly-coupled information system based on a unified model at the conceptual level. This system provides integrated access to data managed by disparate resources. Although the generic architecture is suited for a range of applications, the challenging area of the Earth Observation from the Space domain was targeted. Federation customer applications are masked from the data resources logical and semantic heterogeneity by means of the common schema. Data source wrappers hide technical and data model heterogeneities, providing high autonomy to data resource systems. An important characteristic of the developed system is its capability of carrying out meaningful semantic integrations at both the presentation and data levels. Two kinds of mediations are utilised: pre-canned query mediation, implemented by means of mediator components and dynamic service mediation, implemented by means of facilitator components. For component integration technology we used the E-business enabling technology. An application scenario, experimented in the framework of a project funded by the Italian Space Agency, is reported.

1 The EOS Application Domain

The EOS community needs information integration, because of the large heterogeneity of both data types and methodologies used by scientists to collect and process information about studied phenomena.

Important requirements of the application domain are:

- To avoid *data overloading*, implementing semantic integration among heterogeneous data sources.
- To integrate legacy data sources and systems in an efficient and modular way;
- To device an architecture well-suited for evolution.

When data already exists in a variety of highly heterogeneous and autonomous managing systems, simply integrating this data does not result in a valid distributed information system to fully support scientific Community requirements. Effectively, what is required is a federated distributed system approach in which the individual participants in the federation (i.e. data resources) are self-contained autonomous systems, but together form a consistent wider picture—the federation.

The main objective of such federation is the information interoperation. Information interoperation is different from data and database integration, since the last approach tries to combine data sources, meanwhile information interoperation tries to combine only selected results derived from them [1].

Our goal was to enable users to issue a single query in order to search multiple information sources and, in return, receive a combined result incorporating data from across these sources.

In order to comply with EOS Community needs, we faced the following main tasks:

1. To allow information sharing, implementing interoperability among disparate (i.e. heterogeneous and distributed) application and data resources systems;
2. To facilitate the cooperation between such systems;
3. To coordinate their interaction.

The present paper describes the solutions adopted for carrying out the first task, and in a certain extent the second one.

2 System Integration Approach

We implemented a mid-tier integration approach which utilises the *wrapping* of parts of existing systems (i.e. EOS data sources) to form a Federated Information System (FIS).

The considered scenario is characterized by heterogeneous data sources which are expected to be numerous and quickly evolving. We developed a model-based mediating system, in which unified views are defined and executed at the level of conceptual models rather than at the structural level.

In order to guarantee data source autonomy, the developed FIS is a read-only system: the federation does not allow the updating (or insertion) of data into the participant resource systems, through the federation layer.

We adopted a top-down strategy for implementing the FIS: first we introduced the FIS unified information need, achieving the common model, then we plugged in the data resources, mapping their contribute to such need. In this process, the actual schema of data sources was not considered for the design of the common model schema: there was no need to include resource schemas completely, only unified view data was required. Moreover, there was no need to represent data on the federation level exactly as in the data sources, an abstracted representation is required.

The developed FIS implements a virtual integration architecture: it materializes query results only temporarily -at the time the query is posed, implementing a mechanism to translate queries against the common schema into several semantically meaningful and executable queries against data resources.

2.1 Mediator-based Solution

The system implements a mediator-based architecture, which particularly fits in on integrating disparate information sources in a quickly evolving scenario. Figure 1 represents such architecture.

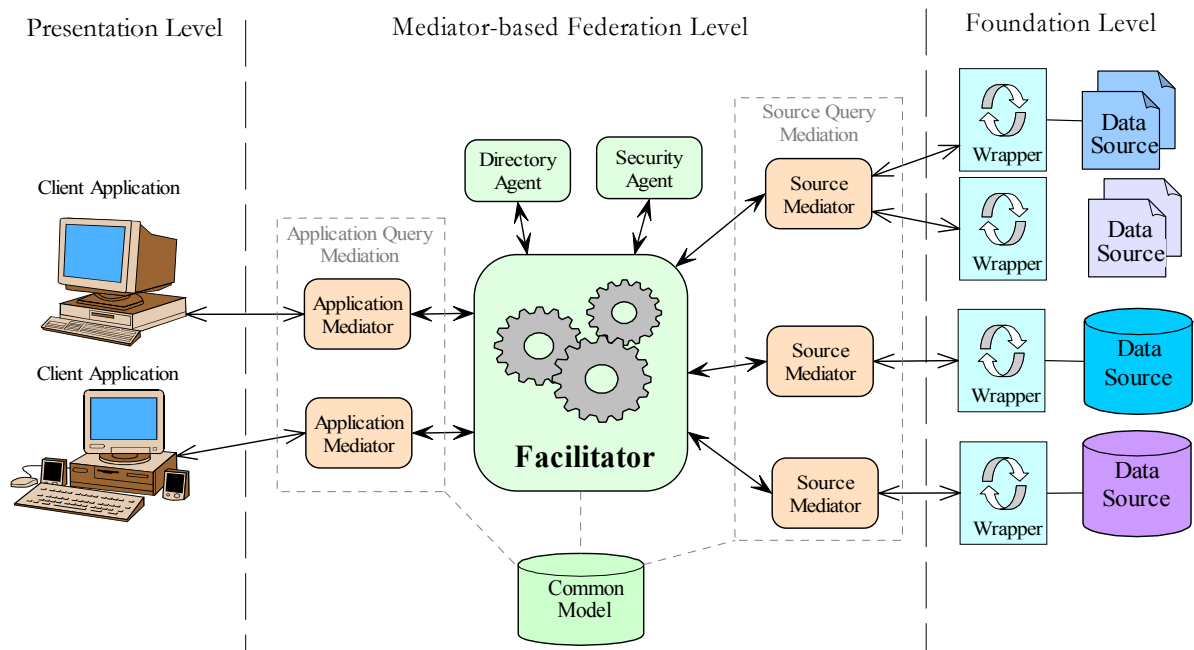


Fig. 1 Logical architecture of the mediator-based FIS

The architecture represents three logical layers:

The Presentation level contains customer applications which access a set of heterogeneous data sources (i.e. federation participants) through the federation/mediation layer.

The Mediator-based Federation level is a software component middleware that offers a uniform way to access the information stored in the data sources. The uniformity is reached by means of the following interoperation strategy: to adopt a common conceptual schema, a uniform query language and a common set of metadata for describing both the resources and their data content.

The foundation level contains the data sources; they are integrated into the federation infrastructure by means of wrappers which resolves several heterogeneities. Data sources may be structured (e.g. DB) or semistructured (e.g. XML files, flat files)

In such a configuration, the application and information resource components form a federation in which client application relies on the Facilitator to fulfil its requests, and the information sources do not renounce to their autonomy, thanks to mediators services.

The Mediator-based Federation level contains a logical facilitator which provides dynamic linkage services to the Application and Source Query Mediators (QMs); these mediators provide query mediations at the presentation and data source level, respectively. They implement semantic integration and transformation services. The major task of these mediators is to increase the information density of data shipped to client applications: to reduce data volume to be shipped to client applications, maintaining its information content [2]. In order to achieve such goal, mediators utilizes semantic integration tools. Reducing data transmission volume also reduces communication delays and costs.

QMs modularisation allows the implementation of different Application mediators, that provide different high-level services for semantic integration. That is useful for serving diverse information communities.

Data source wrappers hide technical and data model heterogeneities; the way they access wrapped data sources is transparent for mediators. Data model heterogeneity is resolved mapping different data models to a simple hierarchical model, with objects and object nesting, but e.g. no inheritance and no classes. Semantic conflicts between the data source concepts and the unified conceptual model are resolved by the Source QM.

These wrappers are well-customised components, since they deal with data model heterogeneity. Once such wrappers are implemented, the autonomy of data resource systems is very high.

3 The Common Model

The Mediation paradigm depends on models; models of resources and models of the user application needs. In accordance with Wiederhold and Genesereth [2], we consider that people, when faced with

complex tasks, categories the processes and objects to be dealt with, so that they can apply a divide-and-conquer paradigm. Hence, we introduced a hierarchical client application model. The need for conceptual clarity also supports the use of hierarchies. Such model represents the federation common model.

The introduced common model is at the conceptual level rather than at the schema level. In fact, the presented system must mediate across complex sources whose data comes from possible disjoint EOS information communities (e.g. oceanography and atmospheric communities). Therefore, the system mediated views were defined and worked out at the level of the conceptual models rather than at the structural level.

The common conceptual model introduces a common ontology, according to which data source must be modeled.

To facilitate extensibility, we introduced a common conceptual model which was obtained as a simple profile of existing standard reference models for geo-data (i.e. ISO 19107, ISO 19115 [3], and OpenGIS abstract specs: topics 5, 6, and 11. [4]). That allows the integration of the most popular types of data models, addressing semantic and schema heterogeneity.

Figure 2 represents a conceptual schema of the adopted common model.

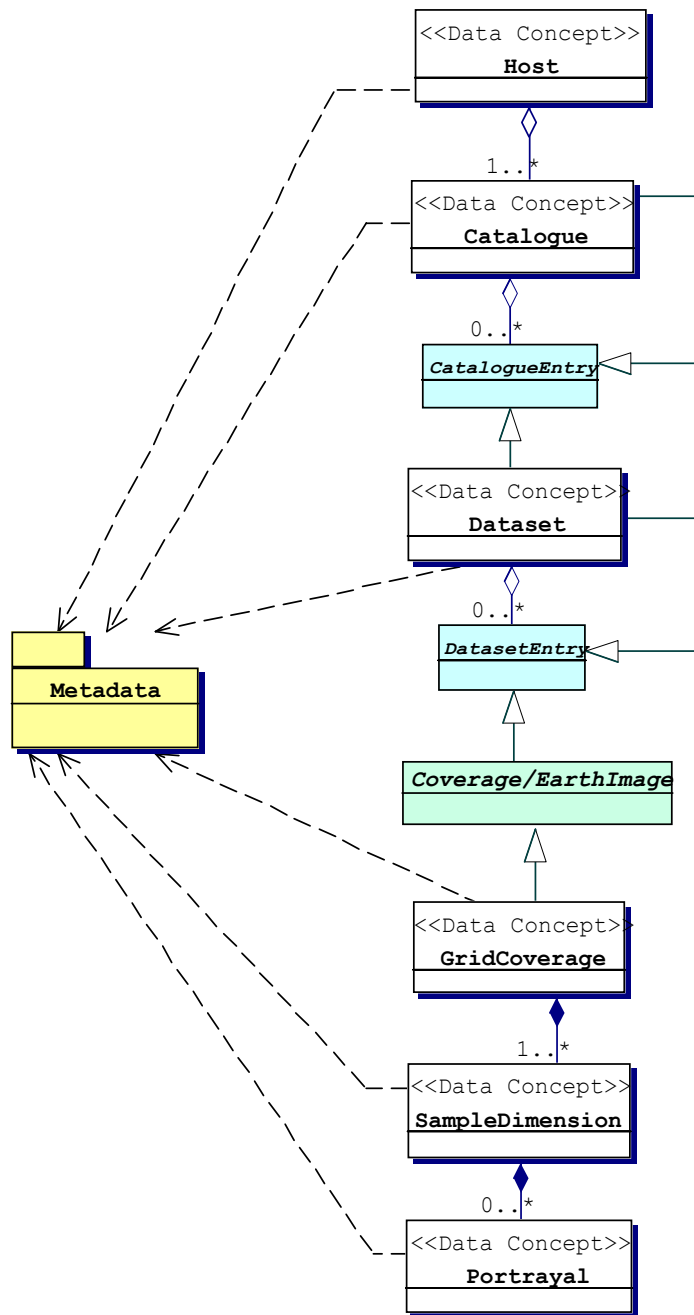


Fig. 2 Schema of the federation common model

Fundamentally, the *Coverage/EarthImage* data concept provides a n-dimensional (where n is 2, or higher) *view* of spatially distributed features; examples are: multi-band satellite imagery, aerial-photos, volumetric radar scans. In our setting, the view is geospatially registered to the Earth. A special type of

Coverage was considered: the grid cell coverage (i.e. GridCoverage) [5]. In the future, others specialisations will be supported.

A Portrayal is a possible rendering of a sample dimension: an acquisition component of a GridCoverage (e.g. a satellite band sensing).

Each data concept is characterised by a set of metadata that completely describes it.

4 System Engineering View

The logical Facilitator, depicted in Fig. 1, is actually distributed on system components deployed on multiple computational nodes. Fig. 3 shows the distribution architecture.

When Client Application sends a connection request to the Federation Facilitator, it returns back a copy of the federation directory which contains the description of published resources. Hence, the Application Facilitator manages a snapshot of the evolving federation directory; the Application Facilitator consults such directory to enable the Client Application to communicate directly with the information resources -with the assistance of high-level mediating and security services. Such solution balances the communication load and avoids bottle-necks.

Besides, the dynamic federation directory turned out to be static for the Client Application. Such compromise proved to be acceptable as the evolution rate of the resource domain is significantly greater than single Client connection length. Nevertheless, possible inconsistencies can occur and the Application Facilitator is in charge of managing them.

The federation plugs in resource components which can contain more than one data sources.

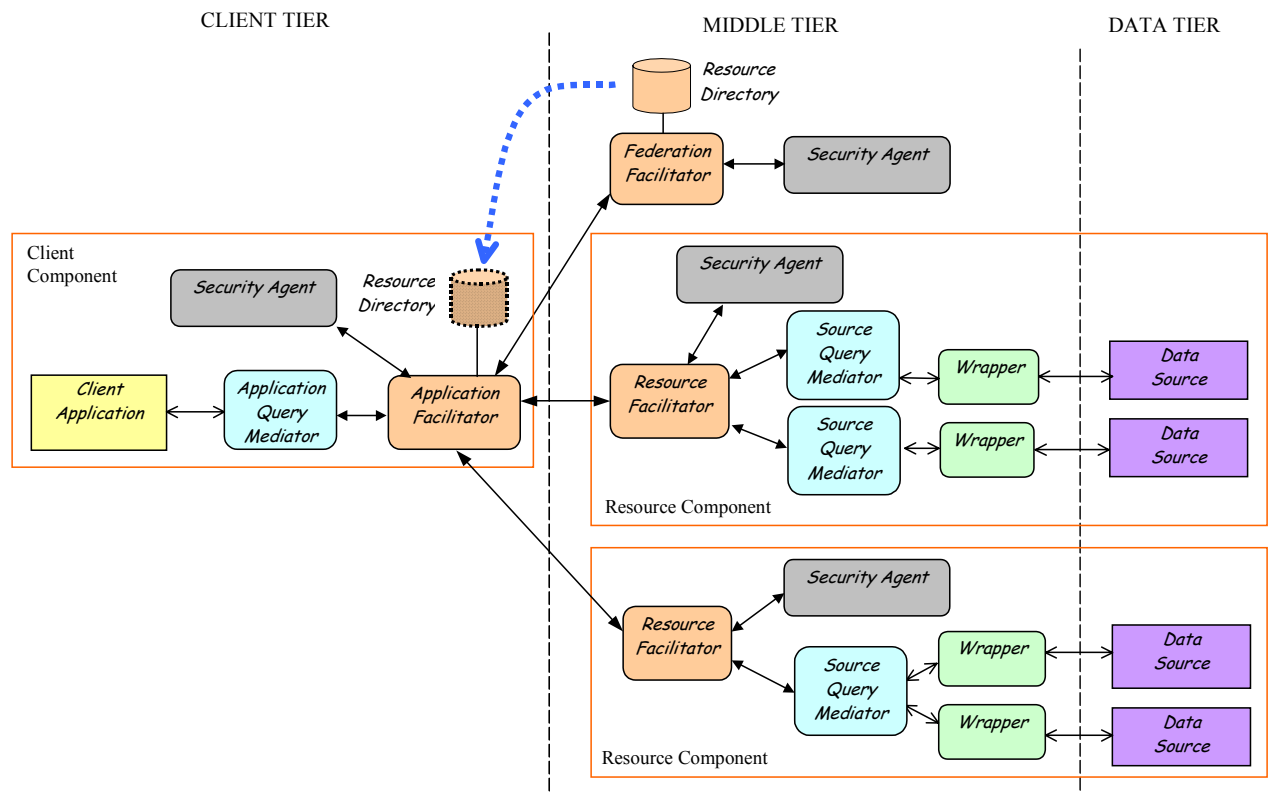


Fig. 3 Distribution logical architecture of the mediator-based system

Interconnected facilitators and mediators, as well as agents share a common communication protocol and encoding language to interact each others (see filled black arrows). Such mechanism is important to decouple components implementation from interface.

For the model mapping of a given data source, the Source QM is in charge of mapping concepts structure and semantics as well as meta-information associated with concepts. These tasks are possible because the data source was opportunely wrapped, and possible descriptive lacks were resolved.

In general, the Application QM receives and formalises a client application query, it utilises the Application Facilitator to fragment the query according to the capabilities of the data sources, and sends the fragments to the appropriate Source QMs through a Resource Facilitator. As the resources return their

individual results to the Application QM, it integrates these result fragments into a single combined result and sends that back to the Client Application.

The query process consists of the following main steps:

1. Federation Query planning (application mediator/facilitator)
2. Federation Query Decomposition (application mediator/facilitator)
3. Source Query planning (source mediator/facilitator)*
4. Source Query Decomposition (source mediator/facilitator)*
5. Query Translation (wrapper)*
6. Source Query Result integration (source mediator/facilitator)*
7. Federation Query Result integration (application mediator/facilitator)

* For each data resource involved (i.e. selected by the facilitator).

4.1 Facilitator Components

Facilitating services provide assistance in locating, understanding and querying resource capabilities and data. Facilitators have the following knowledge: [2]

1. Metadata about information resources;
2. The common conceptual model ontology;
3. The resource dataset encoding format

They utilise such meta information in routing data and information.

For example, when a new resource becomes available, the Federation Facilitator receives a machine-processable description of the new resource from the Resource Facilitator. Then the Federation Facilitator can automatically integrate the new resource, and the client applications can be served by the new resource.

Facilitators call mediators for value added services: semantic integration services and ontology mapping. For achieving such interoperation, facilitators and mediators share a common interfaces language.

The facilitator components provide complex routing services: facilitator can decompose a request into a set of sub-requests, get the sub-results and combine these to obtain the answer. Facilitator is in charge of managing and applying security policy (at the application, federation, and resource levels according to its role) invoking Security Agents. Furthermore, facilitator components implements resource discovery (at the federation or source level) and execution services. Eventually, facilitators implement data and process management services: execution services to schedule individual processes and maintain local information base, e.g. on resources at the federation scope, or data sources at the resource scope.

4.1.1 Application Facilitator

Application Facilitator can be queried to find out the resources that can provide particular information, matching resource capabilities –expressed in accordance with the common model. Eventually, the Facilitator manages the possible inconsistencies between its resource directory and the real data source availability.

Application Facilitator allows users to browse and/or query the structure –according to the common schema- of the available resources and their content, and to browse content metadata; the Application Facilitator utilises mediator services to accomplish such task.

Before routing a Client request to Resource or Federated Facilitators, it calls the client Security Agent to generate a client badge for endorsing the request, and encapsulates the badge in the request itself.

4.1.2 Resource Facilitator

The Resource Facilitator provides services to execute requests delivered by Application Facilitators (e.g. browsing and query requests) through the available Source QMs. The facilitator collects and maintains information on the data sources available at the resource level. Moreover, it applies resource authorisation policy through the resource Security Agent. Eventually, it implements a *Resource Publishing* service which allows to describe and publish/un-publish the resource as an entry of the resource directory managed by the Federation Facilitator.

4.2 QM components

QMs implement semantic integration services; they address data model and semantic heterogeneity of data sources, yielding semantic transparency.

4.2.1 Source QM

The Source QM evaluates forwarded query requests, issued by Client Applications –as far as the scope of its served data source is concerned. It maps wrapped data sources to the shared common ontology and semantics –at the federation level. It decomposes a client query (i.e. query against the federation common schema) into a set of distributed requests against local interfaced data sources. It manages local results and their semantic integration to form a unique answer compliant with the federation common domain model.

The Source QM must face information shortage, type conversions and semantic mismatches.

4.2.2 Application QM

The Application QM executes a client query against the federation common conceptual domain, through the facilitating services. It provides Client Application with an integrated *view* of federation resources. Each data source export a view which basically is a tree-structure, the integrated view is a combination of such trees applying simple association rules.

The Application QM is capable to decompose a Client Application request into a set of distinct requests against the common schema, and to re-assemble the achieved results.

The Application QM performs a caching service: it stores subset of the retrieved datasets. A large cache –like warehouse middleware- improves performances, but mediators should be kept light-weight avoiding responsibilities for persistent storage [1].

4.3 Wrapper

Wrapper components convert heterogeneous data model of data sources in a common data model according to a hierarchical metamodel. That allows Source QM to work with a unique data model. In summary, wrappers re-organise source data according to a hierarchical tree-structure.

Each wrapper implements its own interface to the information source: some wrappers simply transform the output of an information resource, besides other wrappers may modify the meaning or behaviour of information sources. Wrapper utilises a set of transformation rules and inferences in order to address data schema heterogeneity with a common perspective (i.e. data reconstructing services).

Wrapper component copes with local data access and local query interfacing (i.e. communication wrapping services and behavioural transformation wrapping services [6]), as well as source lack of computational capability.

4.4 Security

Federation-shared data may not be freely available but, e.g. provided under commercial license. Research centre may decide to provide data only to selected users and at different prices. Hence, many security problems arise when information sharing through a system federation is involved. In particular, the following security needs of the EOS Community were addressed:

- i. Access Control: data should be accessed only by authorised users;
- ii. Confidentiality: data should be read only by the proper recipient;
- iii. Non-repudiation: data download order should not be repudiable;

Access Control requires the adoption of an AAA (Authentication, Authorization, Accounting/Auditing) architecture. EOS data providers put some constraints about the final AAA architecture: for example, they dislike a centralised authorisation system preferring to decide grants on their own. On the contrary, authentication should be centralised to allow the implementation of a Single Sign On (SSO) solution.

The security subsystem was designed to satisfy these user requirements on a widely heterogeneous architecture resorting to Public Key solutions. According to Fig. 3, the security architecture comprises a central authentication service provided by the Federation Facilitator through its Security Agent. It can manage several different authentication schema exchanging proper messages with the Client Application Facilitator over a secure channel. In particular when clients connect to the federation system, a preliminary authentication phase is required. If the user passes the check he/she is considered authenticated and the Federation Facilitator sends to the Application Facilitator an information set named Badge: it comprises the username, the client IP address, and the time-stamping along with other optional information. The authentication server provides the Badge, along with its digital signature.

Then, the client can connect to the federated hosts. The first time –in a session- a client connects to a federated host, it must go through a Badge validation procedure, that is entirely transparent to the user, allowing the SSO.

The Application Facilitator sends its Badge to the Resource Facilitator whose Security Agent tries to validate it. If the Badge is valid -according to the signature and contained session information- the Client can start to browse Resource information. Indeed for each query, the Resource Facilitator delegates the Security Agent to check user grants before collecting results.

Accounting and Auditing are obtained through a log service offered by each Resource Facilitator. The use of client certificate allows for digital signature of selected entry reported in the log archive providing non-repudiability of data order.

4.5 Client Application

The Client Application GUI presents a combination of browsing and querying services. It is completely driven by the federation unified view and integrates browsing and querying of the tree-structured domain model.

Considering EOS datasets as geo-located data -sensing physical phenomena, or describing Earth properties- the client utilises a metadata query paradigm. That simplifies the data semantic abstraction and therefore the overall query process.

A Client query can be accomplished by invoking a free combination of the following primitive requests:

- ❖ **QueryFeaturesByKeyword**, which takes as input parameters one or more *KeywordObject*;
- ❖ **QueryFeaturesByArea**, which takes as input parameters a *SpatialExtentObject*;
- ❖ **QueryFeaturesByInterval**, which takes as input parameters a *TemporalExtentObject*;
- ❖ **QueryFeaturesByParent**, which takes as input parameters a *ParentObject* (i.e. any node of the tree structure of the unified view, the user utilized to formulate the query on).

These requests implement an interaction methodology (independent of any particular technology) which is based on the following basic choices: what, where, when and who).

A query mediated result is materialised in a browsing or navigational mode by the client application. As the user navigates through the result, the system progressively materializes the unvisited parts of the unified view, either using a local (i.e. managed by the Application QM) cache or submitting another request to the system. User can navigate two different contexts: the federation information content and the metadata which characterize each piece of information.

4.6 XML Encoding

Each data source exports a model of the contained information that is encoded according to an introduced XML schema: the XML encoding of the unified model. It describes the structure of all the data exchanged by the system components. The Source QMs produce documents that conform to such schema.

Source QMs utilises an encoding service -based on a set of encoding rules- which creates a dataset that is system independent and therefore suitable to be transferred. These encoding rules encode the XML

structures of a wrapped source into the XML structures of the common model. Then, the encoded dataset is transferred using a transfer service.

The client query evaluation and integration process may be viewed as the generation of an XML Document which is virtual as it is not materialized. In particular the integration process (performed by the Application QM) combines the returned XML documents, generated as responses from the different data sources involved in the query finalisation.

Each source is queried with an XML-based query language. A simple customised language has been developed as part of the project. It implements parameterised canned queries (i.e. predefined queries with variable parameters).

5 Experimentation and Application Scenario

A first implementation and experimentation of the described solution has been conducted in the framework of a project funded by the ASI (Italian Space Agency) [7]. It set up a federation of resource systems, located in the central and southern regions of Italy.

As far as data heterogeneity is concerned, the experimentation concerned two main kinds of data archives –which are the most important ones for such federation purpose:

- ❖ *Sensor imagery archives*: the historical archive of a given sensor acquisitions, on a certain spatial region, which is managed by an expert centre;
- ❖ *Case Study archives*: the archive of a set of disparate processed data which constitute a case study.

The experimentation has been successful and led to the confirmation of the presented solution for the achievement of a wide federation made up of archives managed by Italian academic and CNR (National Research Centre) institutes.

5.1 E-Business enabling technology

For component integration technology we used E-business enabling technologies. These middleware and software components removed the technical heterogeneity and addressed syntax heterogeneity of data encoding, by using XML-dialects; the result of any query, as well as the query itself is an XML document. Such technology is essential:

- to expand the traditional client-server model to fully leverage the power of web-based systems;
- to overcome the client-server-based architecture constraint of being heavily dependent on the type of hardware and software used and calling for the client to be very aware of the server and vice-versa;
- to leverage the benefits of open standards and network openness;
- to use Internet technologies to develop an application that extend beyond the traditional space and organisational borders.

In fact the true value of an E-Business architecture lies in its ability to provide integration with back-end systems and databases [8].

The adoption of a web-delivery architecture involves some issues: to strengthen the security issue and to define a generic protocol to make client and resource managers communicate.

Figure 4 reports the implemented architecture. For space sake only one *client component* is represented and it accesses only one *resource component*.

5.1.1 The Central Server

It logically represents the overall federation; it contains a web server users must connect to, in order to: subscribe/unsubscribe and start-up an interactive client session. An application server is utilised to manage the session state and several server components implemented as Java applications to: decode and interpret client messages, add/update user's login profile, centrally authenticate users, add a host to the federation list, activate/de-activate hosts.

The central server is also in charge of delivering –where necessary- the client application components to the client station: the Java Web Start technology was used; it extends the traditional Java Applet, allowing less effort in the programming phase.

5.1.2 The Client

User's client station is a PC or a UNIX workstation using a Web browser. Users utilises a Web session to contact the central web server in order to download the Java client (only when a new version is available) and starts it. The first operation is the authentication stage that the user must pass to enter the federation. The main Java components are: a human-computer high level interface (i.e. Navigation module), the Application QM and the Application Facilitator. The first one implements a human interface to provide user information discovery and concepts navigation and to reduce the overhead of working with a large federation. The other ones expose interfaces that correspond to business-level calls client needs and ensures these calls are handled by the appropriate application services. They encapsulate the details of the distributed-communication mechanism used to shunt queries from client to back-end services. These structures isolate the client from the details of how the back-end applications are accessed and allow the reuse of the back-end components across multiple client types.

5.1.3 The Federal Host Gateway

Each federal resource presents a gateway to the federation. It implements the Source QM, the Source Facilitator and the Wrapper components. It contains a web server, Application Facilitators interact with, for querying the resource. An application server is utilised to manage the session state and several server-side components implemented as Java applications to accomplish back-end tasks to: handle client requests, access and wrap the resource, authorise information retrieving, periodically communicate the gateway alive status to the *central server*, join the federation and dispatch a publish/unpublish request.

5.1.4 Message Encoding / Decoding

The Source and Application Mediators invoke a facilitating transfer service to exchange encoded data-set each others. The transfer service follows a transfer protocol which specifies packaging and transport rules. The transportation was considered only over an on-line communication medium.

We adopted SOAP (Simple Object Access Protocol) to implement the transfer services. SOAP specifies a general, asynchronous message messaging service as well as a simpler synchronous RPC service, the latter built upon the former. Although the SOAP-RPC service would suit the read-only nature of the federation data model, that basically leads to a one-way query/response data flow, we decided to implement the transfer service by means of SOAP general messaging.

This design choice increases to some extent the complexity of the facilitator component, but guarantees an effective decoupling of servers and clients with respect to their implementation language. On the other hand, a RPC strategy, by actually dictating the return value types, would possibly impose a particular language binding.

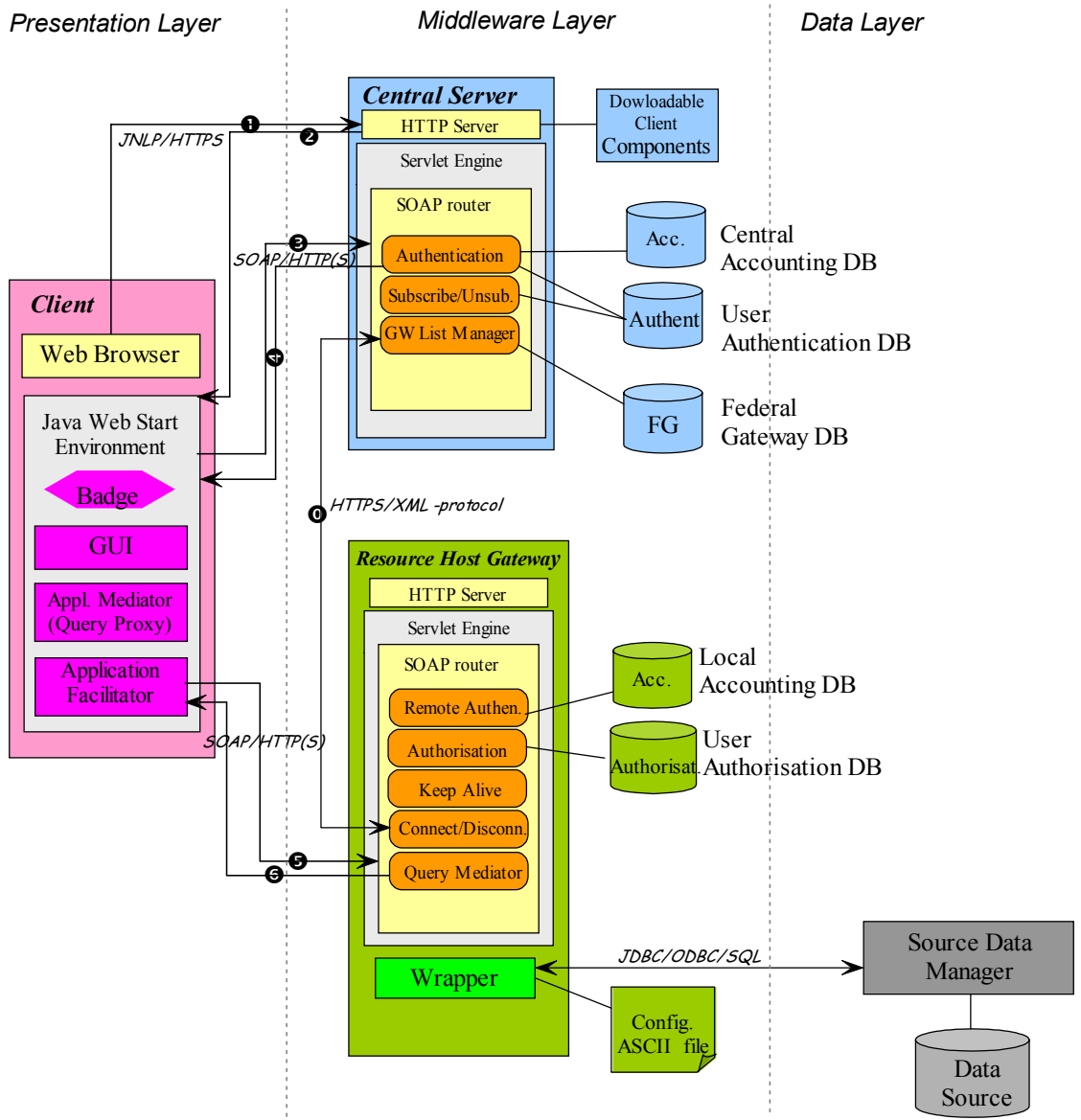


Fig 4 Computational Architecture of the System

References

- [1] G. Wiederhold, "Mediation to Deal with Heterogeneous Data Sources", in Interoperating Geographic Information Systems, Springer LNCS 1580 (Proc. Interop'99, Zurich, March 1999), pp. 1-16.
- [2] Gio Wiederhold, M. Genesereth, "The Conceptual Basis for Mediation Services", IEEE Expert, Vol. 12, n. 5, pp. 38-47, 1997.
- [3] <http://www.statkart.no/isotc211/>
- [4] <http://www.opengis.org>
- [5] The OpenGIS™ Abstract Specification, "Topic 6: The Coverage Type and its Subtypes", Open GIS Consortium, 1999.
- [6] Hull R., King R., Editors, "Reference Architecture for the Intelligent Integration of Information", The Program on Intelligent Integration of Information (I³), Advanced Research Project Agency, May 1995.
- [7]. Nativi S., Mazzetti P., Bigagli L., and Pirri F., Sistema Informativo Distribuito per Esigenze della Comunita' Scientifica Nazionale Operante nel Settore dell'Osservazione della Terra dallo Spazio, proceedings of the National Workshop IGPB 2000, Roma (2000).
- [8]. Koushik S., and Joodi P.: E-Business Architecture Design Issues, IT Professional May/June 2000, 38-43 (2000).

Mediator-based Systems (Box)

Mediation concepts were –to our knowledge- introduced by Wiederhold [B1]: mediation techniques cover a wide type of functionalities that enhance stored data prior to their use in an application. Wiederhold defined a mediator as an active lightweight software component that exploits encoded knowledge about certain sets or subsets of data to create information for a higher level of application. Mediator accesses data sources or other mediators on-demand through interactive protocols.

Mediator systems federate and integrate data from disparate sources in order to elicit information that the individual sources cannot provide independently. Mediators function is to provide integrated information, without the need to integrate the base data resources [B2].

Mediator-based architectures are become very well-known solutions as data integration solutions [B3]. Starting from the state-of-the-art of the existing “mediator-based systems”, Busse et Al. [B4] provided a more specific definition for mediator-based information systems: *A mediator-based information system is a system that offers a homogeneous, virtual and read-only access mechanism to a dynamically changing collection of heterogeneous, autonomous and distributed information sources. Access is carried out through a query language and uses a global schema.* The main software components of a mediator-based information system are wrappers, which encapsulate sources and remove technical and data model heterogeneity, and mediators, which resolve logical heterogeneity [B4].

Compared to data warehousing approach, the main difference is that mediator-based systems support combining query results from individual sources rather than combining their data. The main assets of such architecture are its scalability and extensibility [B5].

Facilitator

While mediators represent responsible, predictable and stable services, facilitators provide automation and rapid response to changing situation [1].

The facilitator provides facilitating services which implement one-stop shopping capabilities for applications [8]. Users send information and requests to the facilitator which in turn delegates to appropriate services.

These middleware services allow for the access of source information and facilitate the integration of mediated results into the user applications.

Facilitating services utilise metadata which describe capabilities and characteristics of FIS resources. The services examine user's request, select the information resources that can handle it, send the request to the appropriate mediators, get the answer, and return the request to the user.

These services are precious because they remove the burden of interoperation in a dynamic environment, from user to the federation. Users are not required to know about the existence of any available information resources at run-time; they do not need to directly manage request finalisation against such resources.

The main concern with facilitating services is on the overall system efficiency.

A facilitator can be viewed as an instant mediator to the extent that automation allows [2]. On the other hand, a mediator can be viewed as a petrified facilitator, requiring human intervention for reconfiguring it when resources, customer needs, or mediating knowledge changes [2].

[B1] Gio Wiederhold, "Mediators in the Architecture of Future Information Systems", IEEE Computer March 1992, pp.38-49.

[B2] B. Ludascher, A Gupta, M.E. Martone, "Model-based mediation with domain maps", Data Engineering.

[B3] Data Mediation Forum, http://www.dm-forum.org/eng/dmf_mot.htm

- [B4] S. Busse, R.-D. Kutsche, U. Leser, H. Weber, Federated Information Systems: concepts, terminology and architectures, Technical Report Nr. 99-9, TU Berlin, 1999.
- [B5] A. Gupta, R. Marciano, I. Zavslasky, C. Baru, "Integrating GIS and Imagery through XML-based Information Mediation", <http://www.npaci.edu/DICE/Pubs/isd99.pdf>