2013 27th International Conference on Advanced Information Networking and Applications Workshops

A Cloud-based framework for collaborative data management in the VPH-Share Project

Siegfried Benkner*, Chris Borckholder*, Marian Bubak†, Yuriy Kaniovskyi*, Richard Knight‡
Martin Koehler*, Spiros Koulouzis†, Piotr Nowakowski† and Steven Wood‡
*University of Vienna, Research Group Scientific Computing
1090 Vienna, Austria
†Institute of Computer Science / Cyfronet AGH University of Science and Technology
Krakow, Poland
‡Scientific Computing & Informatics, Sheffield Teaching Hospitals NHS Foundation Trust
Sheffield, UK

***Abstract*—The VPH-Share project objective is to store, share, integrate, and link data, information, knowledge, and wisdom about the physiopathology of the human body to enable their reuse within the virtual physiological human community. Therefore, the projects develops a modular and generic data management platform on top of a distributed Cloud infrastructure. The data management platform enables the ontological annotation of VPH-relevant datasets, their provisioning in the Cloud, and supports different data integration approaches. In this paper we present the architecture and implementation of this VPH-Share Cloud and data management platform and we go into detail about two different data integration approaches: relational data mediation, which has been realized on top of a distributed data mediation engine, and semantic data integration, which is supported on the basis of the SPARQL federation extension. Both approaches are examined on top of a project-specific scenario executed in the VPH-Share Cloud environment.**

I. Introduction

The VPH-Share project has been funded within the European Union's Virtual Physiological Human Initiative (VPH-I). It focuses on the the provisioning of a systematic framework for understanding physiological processes in the human body in terms of anatomical structure and biophysical mechanisms at multiple length and time scales. The objective of the VPH-Share project is to transform the European health care system into a more personalized, predictive, and integrated one with significant impact on health care and disease prevention. To realize these goals, the project aims to develop a novel IT infrastructure enabling profound collaborations between VPH-I members by integrating a platform for data exposure and management with a framework for hosting and executing domain-specific applications and VPH-related workflows. Initially, the project focuses on four flagship workflows from European research projects within the VPH domain (@neurIST [1], Virolab [2], euHeart [3], and VPHOP [4]) which provide data, tools and models.

The growing amount and the complexity of available information, especially within the biomedical domain, mandates the development of new mechanisms for management and sharing of this information. The biomedical information of interest for the VPH community is from various domains, complex in structure, and spread across different organizations and stakeholders. The vast amount of available data outruns the current practice of efficiently managing and sharing of this information. Therefore, the VPH-Share project develops a large-scale and unified data management platform (DMP) [5],[6] addressing these challenges. The DMP follows a process similar to incremental Extract-Transform-Load (ETL) [7] and enables on-the-fly integration of new datasets. On the one hand, this leads to an evolving dataset provided by the VPH-Share project. On the other hand, the project supports pay-as-you-go semantic data integration leading to creation of on-demand customized data spaces.

The DMP platform will be provided on the basis of a Cloud-based infrastructure, the Atmosphere Cloud environment, enabling execution of workflows and applications as well as provisioning and sharing of datasets within the community. Datasets, made available to the VPH community, are hosted within this Cloud environment by implementing the concepts of semantic services [8] and atomic services. Atomic services are realized on the basis of virtual appliances, virtual machine images preconfigured with a specific software stack. They are managed within the Atmosphere Cloud environment.

The DMP platform consists of two main parts: a tool set enabling management of data sources (which includes selection of data and semantic annotation of data) and provisioning/hosting of data. These steps are supported on the basis of a graphical tool. The second part enables the exposure of datasets in the Cloud, the VPH-Share dataset service environment.

The VPH-Share dataset service environment (DSE) has been developed on top of the Vienna Cloud Environment (VCE) [9] as well as the @neurIST data service infostructure [10]. The DSE provides RESTful and Web service interfaces for data access, integration, and mediation on the basis of relational and semantic technologies by leveraging OGSA-DAI [11], SPARQL-DQP [12], and a data mediation engine [13]. Within the DSE, we distinguish between dataset services enabling Web-based access to single data sources and virtual dataset services offering transparent access to multiple, potentially heterogeneous data sources. Both service types provide the

978-0-7695-4952-1/13 $26.00 © 2013 IEEE
DOI 10.1109/WAINA.2013.58

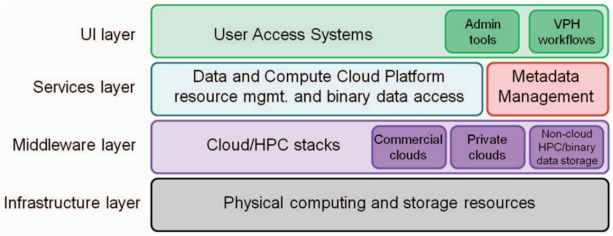


Fig. 1. The placement of the Atmosphere Data and Compute Cloud Platform in the VPH architecture

same uniform service interface.

Within this paper we focus on the provisioning of virtual dataset services as atomic services within the VPH-Share Cloud environment. Different types of atomic services have been defined: atomic dataset services, hosting a specific dataset, atomic virtual semantic dataset services, enabling the integration of multiple datasets on the basis of the SPARQL federation extension, and atomic virtual relational dataset services, offering data mediation mechanisms on the basis of SQL.

In the following sections an overview of the VPH-Share Cloud environment (Atmosphere) and the data management platform is given. Section IV goes into detail about atomic virtual dataset services and then both data integration mechanisms, semantic- and mediation-based, will be explained. A typical VPH-Share scenario, describing applicability of both mechanisms including first performance results, is described afterwards. Finally, related work concluding remarks and future plans are presented.

II. The Data and Compute Cloud Platform

The data and compute cloud platform being developed by the VPH-Share project is called Atmosphere [14]. Its goal is to enable groups of users to gain authorized access to a variety of computational and data services deployed on distributed hardware resources. Most of the technologies underpinning the cloud platform are exclusively implemented as services - applications or other necessary elements of logic that encapsulate the data they operate on, and provide secure interfaces to access them [15]. In this sense the function of the Atmosphere cloud platform is to provide a persistent infrastructure on which to deploy, instantiate, access and manage VPH services (which are understood as applications, or components thereof, fulfilling the specific needs of researchers, such as the atomic virtual dataset services further described in Section IV). This approach allows each application to evolve at its own pace thereby reducing the side effects traditionally seen in the legacy enterprise systems. The starting point for the development of Atmosphere solutions is a selection of standalone applications derived from four VPH workflows: @neurist, EUHeart, ViroLab and VPHOP. Each of these operates a selection of software tools, they are accessible only for their authors and close affiliates. With the help of Atmosphere these tools are meant to be exposed to a wider community of users and potential collaborators. In addition to the applications themselves, the goal of the cloud platform is to provide end users with control interfaces, enabling them to interact with the exposed tools in a secure and convenient manner.

At the core of the proposed framework lies the division of users into application providers, domain scientists and administrators. The cloud platform covers the entire application development lifecycle, from inception to scalable exploitation, assisting each participating class of users at each step of the design, deployment and enactment process. At the same time the VPH cloud platform must form a bridge between the world of cloud middleware services (which are typically difficult to access for inexperienced users) and the familiar OS environments in which standalone scientific applications are deployed and accessed. This view is briefly summarized in Fig. 1, showing the place occupied by the cloud platform in the layered architecture of a typical cloud PaaS offering, of which VPH-Share is an example.

The platform is implemented as a series of modules which together operate on a dedicated host (known as the core host), along with the cloud middleware stacks required to access and manage physical resources. In addition, the platform includes an internal registry where all metadata pertaining to Atomic Services and their instances is located. Finally, embeddable UI extensions are provided - these can operate in a standalone mode or be imported into a portal. All core components can be instantiated multiple times provided they share a common registry (which can be deployed on an external host) - this feature provides the platform with scalability, enabling it to grow along with the number of instantiated services and clients. Finally, the architecture includes a Cloud-based binary data federation component called LOBCDER [16]. A more thorough description of the Atmosphere platform and its subcomponents can be found at [17].

III. The VPH-Share Data Management Platform

The data management platform is a software stack subsuming the project effort towards data management, access, and integration. It basically consists of three parts: the data publication suite (DPS), the dataset service environment (DSE), and the linked data environment (LDE). While the DPS is utilized to migrate local datasets to the VPH-Share Cloud, the DSE and the LDE are utilized to expose the datasets via service interfaces to the VPH-Share stakeholders. An overview of the DMP architecture is depicted in Fig. 2.

A. The Data Publication Suite

The data publication suite provides a graphical user interface and enables management of datasets. It allows data providers to import their data and to select parts of the data to be published. The DPS follows a model-based approach for selecting data from existing sources. This makes possible the integration of additional data sources if needed in the project. Currently, modules have been developed for Microsoft SQL Databases, files, and databases supporting the OLE DB API.

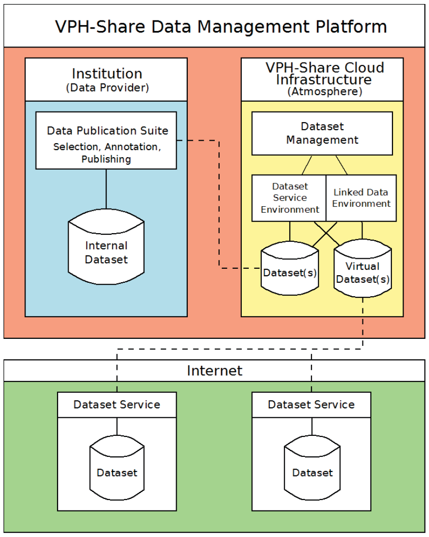


Fig. 2. The Data Management Platform: An institution is enabled to share an internal dataset with the VPH community via the data publication suite. Within the VPH-Share Cloud infrastructure many datasets are hosted and virtual datasets can be provided on top of them via the linked data and the dataset service environment. Additionally, the system enables the integration of external datasets accessable via the Internet.

In the context of this project many single table sources, which often lack a specification of data relationships, have to be integrated. The DPS extracts the defined data relationships and enables the specification of additional ones.

After the data has been selected, the DPS supports the ontological annotation of data sets by providing a drag-and-drop mechanism combined with a concept search in an ontological search base. The ontological search is based on free text specified by the user (normally the column or table name is used) and it utilizes the BioPortal REST API. Ontologies included in the search base is limited to SNOMED Clinical Terms, NCI Thesaurus, NCI Metathesaurus, HL7, and ICD-9. The ranking mechanism used by BioPortal is based on a best text match system which is not sufficient for the requirements of the VPH-Share projects. Therefore, additional ranking algorithms will be included in the DPS. Moreover, the data to be published can be de-identified by integrating project specific de-identification algorithms.

After the specific data has been selected, the relationships have been identified and the data has been annotated, the DPS enables publishing the dataset into the VPH-Share Cloud. A destination (Cloud site) can be chosen and an atomic service will be started on this site. Data transfer and exposure of the dataset as Cloud service is fully transparent. Therefore, the DPS includes additional data management services enabling the interaction between the DPS and the atomic service hosted in the Cloud. These services are utilized for the management of the dataset to be exposed and for the deployment of the dataset services.

B. The Dataset Service Environment

The dataset service environment (DSE) allows Web-based access to and integration of datasets exposed in the Cloud. The DSE supports both RESTful and Web service interfaces for data access, integration, and mediation, all on the basis of relational and semantic technologies.

Within the DSE, we distinguish between dataset services enabling Cloud-based access to single data sources and virtual dataset services offering transparent access to multiple, potentially heterogeneous data sources. Both service types provide the same uniform service interface. Dataset services support relational as well as semantic access to data sources by supporting SQL and SPARQL. Virtual dataset services can follow two different approaches. Firstly, they may integrate distributed and heterogeneous data sources on the relational level (SQL). In this case, they rely on a distributed query processing engine as well as on a data mediation engine. Secondly, data can be integrated by executing federated SPARQL queries following the SPARQL federation extension.

Both types of dataset services expose a RESTful interface and a Web service endpoint on the basis of the WS-* stack (SOAP, WSDL). Supporting both endpoint types simplifies the integration of dataset services into different types of client applications based on different programming languages (e.g. Web-based applications, fat Java client, Python applications).

The DSE has been developed on the basis of the Vienna Cloud Environment (VCE) middleware [9], [18], [19] and the @neurIST data infostructure (@neuInfo) [10]. In addition, the environment relies on multiple technologies. Among them, the OGSA-DAI [11] framework, the de-facto standard for data access and integration in the Grid, SPARQL-DQP [12], a prototype implementation of the SPARQL federation extension, and a (relational) data mediation engine [13].

In the following, the general architecture of dataset services is outlined. The data integration approaches supported by virtual dataset services will be described in detail below.

The implementation of the Web service and RESTful interfaces is based on standard Web service technologies (WSDL, SOAP, WS-*) and REST, both achieved on the basis of the Apache CXF framework [20]. The RESTful interface enables querying of datasets via plain HTTP methods. Two different types of resource representations are supported. The HTML representation enables querying of datasets via a Web browser while the XML representation is utilized for applications. Additionally, a Java-based Client toolkit enabling access to the Web service endpoints and to REST interface is provided.

Datasets services can be configured in two ways. On the one hand they support SQL-based querying of relational datasets. On the other hand semantic datasets can be queried via SPAQRL.

IV. Atomic Virtual Dataset Services

The architecture of the atomic virtual dataset services (AVDS) is depicted in Fig. 3. A AVDS is a virtual appliance with an installation of the VPH-Share dataset service environment and preconfigured Cloud services that allows the

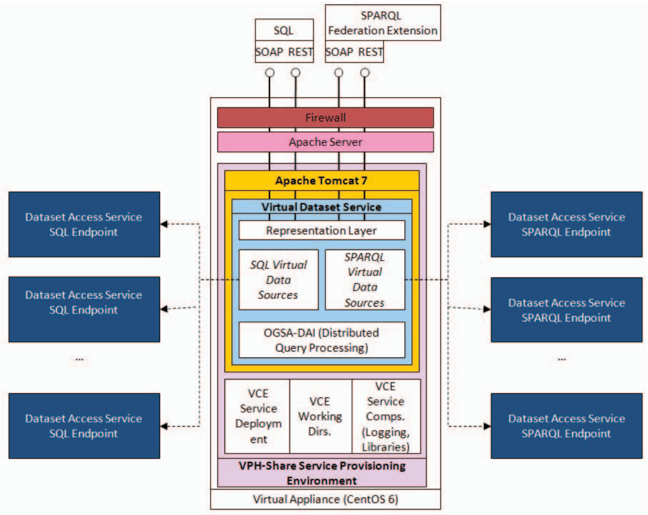

SQL
SOAP REST
SPARQL Federation Extension
SOAP REST
Firewall
Apache Server
Apache Tomcat 7
Virtual Dataset Service
Representation Layer
SQL Virtual Data Sources
SPARQL Virtual Data Sources
OGSA-DAI (Distributed Query Processing)
VCE Service Deploym ent
VCE Working Dirs.
VCE Service Comps. (Logging, Libraries)
VPH-Share Service Provisioning Environment
Virtual Appliance (CentOS 6)
Dataset Access Service SQL Endpoint
Dataset Access Service SQL Endpoint
...
Dataset Access Service SQL Endpoint
Dataset Access Service SPARQL Endpoint
Dataset Access Service SPARQL Endpoint
...
Dataset Access Service SPARQL Endpoint

Fig. 3. Architecture of Atomic Virtual Dataset Service: The virtual appliance includes an installation of the VPH-Share dataset service environment exposing a REST and a Web service interface for executing federated SPARQL and SQL queries. Therefore, the exposed virtual datasets query live data from distributed datasets via their SPARQL or SQL endpoint exposed via the same generic interface.

provisioning of virtual datasets by integrating multiple distributed and possibly heterogeneous datasets on a semantic or relational level. By following the concept of atomic services, this architecture enables on-demand deployment of new virtual datasets within the VPH initiative and hosting of them in the VPH-Share Cloud environment (Atmosphere). Additionally, this approach adds flexibility in the hardware requirements (e.g. disk space) due to the capabilities of the Cloud. In this context the capabilities of virtual datasets are outlined. Virtual datasets enable on-demand and transparent integration of datasets (physical or other virtual datasets). In contrast to a data warehouse, this approach always provides access to live data because data is not stored in the virtual dataset (just the integration rules are saved permanently).

The virtual appliance is based on CentOS, an Apache Server, Java, and the DSE release. The DSE hosts the dataset services in its own Apache Tomcat server and includes additional deployment and service management tools. The atomic service comes with a preinstalled Cloud service instance providing a semantic or a relational virtual dataset.

V. VPH-Share Data Integration Approaches

Atomic virtual dataset services enable the integration of distributed and possibly heterogeneous datasets which are exposed as dataset services as well. Hence, our system enables hierarchical integration of resources by utilizing virtual datasets (exposed by AVDS) as input datasets for other virtual datasets and so on. This is achieved by utilizing a generic and uniform Web service interface for each type of service.

Virtual datasets may integrate other datasets on a relational level on the basis of a mediation schema as well as semantically annotated datasets on the basis of the SPARQL federation extension. Both types of virtual appliances rely on the same distributed query processing engine. In the following, this engine as well as both integration approaches are described.

A. Distributed Query Processing Engine

The OGSA-DAI software framework, which supports data access and integration via Web services, integrates a service-based distributed query processing (DQP) engine. The DQP component makes it possible to federate queries and orchestrate data delivery across several distributed datasets. DQP is able to execute queries in parallel over OGSA-DAI data services and its resources in a highly optimized manner. It provides a basis for our virtual data source approach.

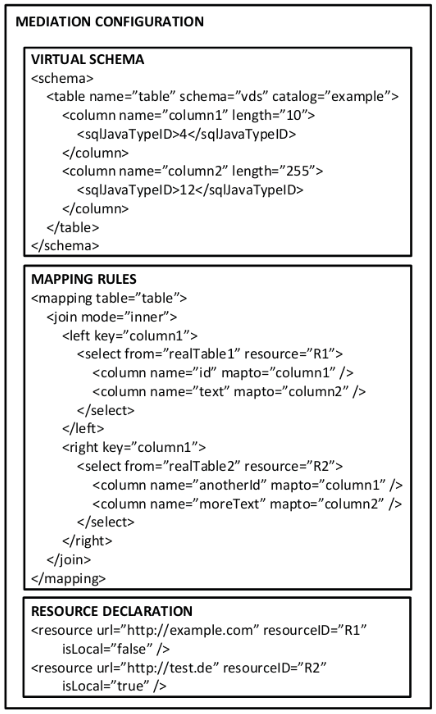
OGSA-DAI utilizes data integration as well as data access components allowing the system to orchestrate user queries onto several distributed data sources. However, the client has to be aware of data distribution and data integration. For instance, at attempt to execute a query like *'SELECT myTable.name FROM myTable, myTable2 WHERE myTable.id=myTable2.id;'* would fail, because DQP wouldn't know which resource it should address to get *myTable* or *myTable2*. The correct query would be *'SELECT ResourceID_myTable.name FROM ResourceID_myTable, ResourceID2_myTable2 WHERE ResourceID_myTable.id=ResourceID2_myTable2.id;'*, i.e., the mapping of OGSA-DAI resources to the underlying data sets has to be specified explicitly. This is a great inconvenience for the user, requiring the client to know which columns are joinable. This complication cannot be easily resolved automatically due to the semantics of column names, where issues include different data types and schemes. To address this problem a custom mapping of tables and their respective resources, including their properties, can be used to provide a truly global virtual data schema as described in the following section.

DQP is implemented as OGSA-DAI resource and its main component is the DQP coordinator. It is composed of a compiler, partitioner, optimizer and a scheduler. The coordinator parses the query, gathers the metadata from the distributed data nodes and compiles a new query plan using the linear left deep tree decomposition approach. The plan is partitioned, evaluated, optimized, if possible, and executed at a remote data source service. The scheduler utilizes gathered metadata and data source information to create and execute execution plans over multiple of execution nodes.

The generated query plan is executed on a selected set of the configured query evaluation services, which are OGSA-DAI services themselves. Each partition of the generated query plan is assigned to one evaluator. Several of them that participate in the evaluation of a query form a tree. The evaluation of individual queries takes place in the leaf nodes, up to the point where the root leaf is processed and the query evaluation is complete.

B. Relational Access and Mediation of Datasets

Relational virtual datasets as exposed by AVDS aim to transparently integrate heterogeneous, distributed relational data resources according to a virtual data schema. The participating



```
MEDIATION CONFIGURATION

VIRTUAL SCHEMA
<schema>
   <table name="table" schema="vds" catalog="example">
      <column name="column1" length="10">
         <sqlJavaTypeID>4</sqlJavaTypeID>
      </column>
      <column name="column2" length="255">
         <sqlJavaTypeID>12</sqlJavaTypeID>
      </column>
   </table>
</schema>

MAPPING RULES
<mapping table="table">
   <join mode="inner">
      <left key="column1">
         <select from="realTable1" resource="R1">
            <column name="id" mapto="column1" />
            <column name="text" mapto="column2" />
         </select>
      </left>
      <right key="column1">
         <select from="realTable2" resource="R2">
            <column name="anotherId" mapto="column1" />
            <column name="moreText" mapto="column2" />
         </select>
      </right>
   </join>
</mapping>

RESOURCE DECLARATION
<resource url="http://example.com" resourceID="R1"
      isLocal="false" />
<resource url="http://test.de" resourceID="R2"
      isLocal="true" />
```

Fig. 4. Example mediation configuration: The configuration consists of three parts (1) the global schema to be exposed; (2) the mapping rules between the global schema and the underlying datasets; (3) the OGSA-DAI specific resource definition of the underlying datasets.

data sources and complexity of formulating a specialized DQP query are hidden from the client. Client queries are executed against the virtual schema respective to the underlying data sources. These queries are mediated on demand and always result in retrieving live data.

Virtual datasets follow the Global-as-View (GaV) approach for data integration. In short, the exposed virtual schema is the global view on the data. Each relation in the virtual schema is mapped to a statically defined combination of the underlying relational resources. Queries against the virtual schema are mediated by replacing the virtual relations with the combination of appropriate queries to the mapped, local resources.

VDS is built on top of OGSA-DAI's Distributed Query Processor (DQP) extension. DQP provides, as the name suggests, means to query distributed relational resources. To achieve translation and mediation of queries against the virtual schema into a distributed query against the integrated datasets, virtual datasets intervenes when DQP is building an abstract query plan. Our data mediation engine traverses the initial virtual query plan against the global data schema of the virtual dataset which has been created by DQP. The mediation engine replaces table scans on virtual relations with the corresponding sub-query plan for that virtual relation. That results in a valid query plan on the configured data resources. This query plan is then executed via the query execution engine provided by OGSA-DQP.

To configure a virtual dataset, an XML-configuration file has to be provided. The XML configuration file has been defined as XML schema and consists of three parts:

- the virtual schema (the global schema to be exposed)
- exactly one mapping for each virtual relation (mapping between global schema and underlying datasets)
- declaration of required underlying datasets

Figure 4 shows the relevant parts of a sample mediation configuration. The first part of the XML document describes the global schema of the virtual dataset within the *schema* element. The schema element includes all virtual relations and their columns and data types that form the virtual schema. In the example, one virtual relation with the name *table* is defined. This relation has two columns (*column1 and column2*). All definitions follow OGSA-DAI's logical database schema format. It has been chosen because it is easily transformed into a set of OGSA-DAI's TableMetaData objects to be processed.

For each virtual relation a corresponding *mapping* element is defined. This element includes the mapping rules between the virtual relation and the underlying datasets. The mapping rules are expressed through a simple, XML-based domain specific language (DSL), that resembles the SQL syntax. It supports JOIN, n-ary UNION and projected SELECT operations on the local data resources. Additionally, functions may be applied to local columns. By offering a NATIVE element as drop in replacement for SELECTS, arbitrary SQL queries may be inserted. That way all SQL features supported by DQP may be utilized as part of a mapping. The integration of functions is very important to overcome possible data heterogeneities. Think of two databases storing patient names differently: dataset one stores the first and the surname in two columns, dataset two uses a single column. By including a simple *concat* function such an issue can be resolved. The sample mediation schema shows how a join on two different datasets can be expressed with this syntax.

At the bottom of Figure 4, the required local relational resources are defined using the DQP configuration syntax. In fact a DQP resource configuration may be reused as it is in a VDS configuration. Remember that each of the local relational resources can be a virtual dataset allowing for hierarchical data mediation scenarios.

When a query plan is mediated, VDS looks up the mapping definition for each table scan on a virtual relation and replaces the table scan operator with a sub-operator tree that matches the mapping. The sub-operator trees are built by so-called *ElementTranslators*. For each XML-element of the mapping a dedicated translator is utilized. These translators create DQP operators producing the required data.

The implementation of virtual datasets is minimally invasive and therefore has been restricted to extensions of the DQP *QueryPlanBuilder*. This enables to profit from existing and future features of DQP (e.g. the whole existing optimizer chain can be used as it is). A client has to only be aware of the declared virtual schema and querying a virtual dataset is accomplished in the same way as any other DQP or SQL resource deployed to OGSA-DAI. Furthermore, the clean separation of concerns between components of virtual datasets simplify extensions and modifications.

C. *Semantic Access to Distributed Datasets*

Dataset services integrate an OGSA-DAI extension enabling execution of SPARQL queries against ontologically annotated datasets. On top of these dataset services, virtual dataset services have been built upon the SPARQL-DQP engine allowing to query multiple SPARQL endpoints in a single query.

SPARQL is a data-oriented semantic querying language used to manipulate and retrieve data stored in the Resource Description Framework (RDF) format. The format builds a graph pattern, consisting of triplets in order to describe the data semantically. It gives the user the ability to compose unambiguous high-level queries as well as to execute them on diverse datasets. This usually requires the dataset to be semantically annotated and exposed through an endpoint.

Recently, the SPARQL query language has been extended by the W3C consortium with support for federated SPARQL queries. In fact, they introduced an extension for federating queries to any number of SPARQL endpoints. Thus, this language extension allows an even higher level of abstraction of distributed heterogeneous datasets. The SPARQL 1.1 federation extension adds two new operators to the protocol: the *BINDING* and the *SERVICE* operator. The first one allows to bind variables in order to constraint or filter the results of a query. More importantly, the second operator allows the specification of a remotely exposed SPARQL endpoint through a URL, where the specified sub-query will be executed.

The authors of [12] have implemented a prototype engine on top of OGSA-DQP and OGSA-DAI-RDF that supports the SPARQL 1.1 federation extension. The DQP engine uses custom parsers, optimizers and logical query plan builders to construct optimized query plans and coordinate the propagation of results, while the RDF component provides the tools, in particular the RDF activities, necessary to query a SPARQL endpoint within OGSA-DAI. As a result, SPARQL-DQP provides a single semantic access point, that needs no further configuration except that of the OGSA-DAI resource.

The prototype has been adjusted, integrated, deployed and tested within the DMP to provide users with the function of querying distributed semantic datasets in conjunction with the relational dataset mediation service.

VI. Experimental Evaluation

Both, the relational virtual dataset and the semantic virtual dataset services, have been evaluated on the basis of a VPH-Share-specific scenario. In the following, the scenario and preliminary performance results are described.

A. *Scenario and Setup*

The test setup of the experiments has been chosen according to the requirements and to the expected project-specific scenarios. Therefore, two datasets, which are hosted in Sheffield, have been selected to be utilized within a first data integration scenario.

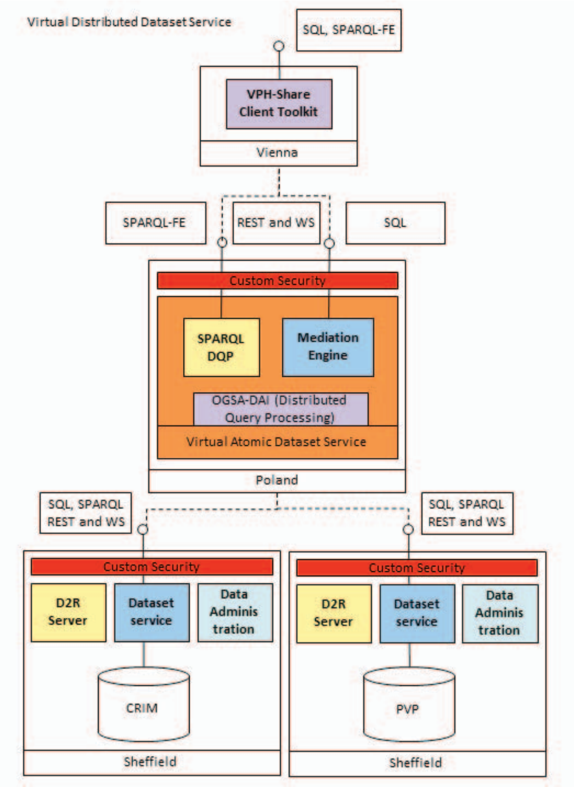


Fig. 5. Performance Evaluation Scenario: Two datasets, both hosted in Sheffield, are exposed as VPH-Share dataset services. A virtual atomic dataset service is provided via the Atmosphere Cloud environment and provides a relational global view on top of both underlying datasets. Additionally, the federated SPARQL engine is installed therein. All of them expose the same generic REST and Web service interface.

Each of them exposes an ontologically annotated dataset via the data management platforms dataset service environment. Therefore, SQL and SPARQL endpoints are provided. The first dataset follows the CRIM database schema, which has been developed in the European project @neurIST. This dataset contains 90 tables with the aneurysm data concerning 100 distinct aneurysm events and 146 patient records. The second dataset exposes the PVP data schema and holds 10 tables with personal and treatment data for 96 patients. Instead of using real data due to privacy and security issues, we confine ourselves to generated test data for testing purposes. This has no effect on the results because both datasets have been generated according to the specifics of the live-data.

In addition to both datasets, the test scenario includes one AVDS instance hosted in the VPH-Share Cloud environment. Specifically, this instance runs at Cyfronet, Poland, the main Cloud provider in the project. The AVDS instance includes a dataset service environment installation and exposes two virtual datasets via Web and RESTful interfaces, one relational and one semantic virtual dataset. Both virtual datasets are configured to integrate both datasets hosted in Sheffield, one with the SQL endpoints, the other with the SPARQL endpoints.

Finally, the queries are invoked and results delivered to an Java-based DSE client application situated in Vienna. An overview of the test scenario is depicted in Figure 5.

B. Preliminary Results

The relational virtual data source exposes a global schema with three tables. The table `event_data` returns all aneurysms from the CRIM dataset, including a CRIM specific foreign key to the patient. A second virtual table `patient_data` integrates data (height, weight, sex, etc.) from the CRIM and the PVP dataset. The last table `patient_with_events` holds the joined data of the former two tables, i.e. all aneurysms with the personal data of the corresponding patient. An overview of the mediation scenario is depicted in Fig. 5. Preliminary performance results have been retrieved by executing the following two queries against the virtual dataset:

- (A) SELECT * FROM patient_data, event_data WHERE patient_data.patient = event_data.patient
- (B) SELECT * FROM patient_with_events

To estimate the impact of mediation overhead and possibly improved optimizations, we also executed a hand-made OGSA-DQP query (C) that is equivalent to both mediated queries. As reference execution time we also executed a plain, non-distributed query (D) directly on a data node in Sheffield that fetches comparable results, but is of course restricted to a single data source and lacks integrated data.

Additionally, the semantic virtual dataset is configured with the CRIM and the PVP dataset as well. The following SPARQL-FE queries have been executed against the virtual dataset. Query E[1] extracts all patients of both of the CRIM and PVP datasets. The inner queries, denoted with the SERVICE keyword, return the IDs of their corresponding datasets, while the outer query applies the UNION operator on them. Query F is a specialized request to the CRIM dataset, that retrieves all patients of the set with the property collarSize=1. Although this query targets only one of the two datasets, the processing still goes through the SPARQL-DQP, and thus, the DQP engine.

Table I depicts the performance results. Both datasets rely on the same distributed query processing engine. The results show that data adding the relational data mediation layer to the DQP engine does not impact the overall performance. Concluding we state that utilizing the data mediation approach hides the details of data integration from the user but the user has to query the dataset on the relational basis. Additionally, new virtual tables can not be integrated on the fly but have to be specified by the administrator. Utilizing semantic virtual datasets enables querying distributed data on an ontological basis but in this version they have to be aware of the semantic datasets to be integrated.

Query	Type	Language	Mean	Median
A	Mediation	SQL	2290ms	2294ms
B	Mediation	SQL	2681ms	2464ms
C	DQP	SQL	2490ms	2235ms
D	OGSA-DAI	SQL	284ms	280ms
E	SPARQL-DQP	SPARQL	1134ms	1472ms
F	SPARQL-DQP	SPARQL	743ms	731ms

TABLE I
PRELIMINARY PERFORMANCE RESULTS: TWO QUERIES (A,B) AGAINST A MEDIATED DATA SOURCE ARE COMPARED WITH A SEMANTICALLY EQUIVALENT QUERY EXECUTED WITH PLAIN OPGSA-DQP. QUERY D HAS BEEN EXECUTED AGAINST A SINGLE DATA SOURCE. QUERY E AND F HAVE BEEN EXECUTED WITH THE FEDERATED SPARQL ENGINE.

VII. RELATED WORK

In the course over the last years, many projects with the objective of semantically integrating or linking data have emerged. Different approaches such as the Linked Data concept [21], centralized data warehouses, or data mediation approaches emerged. Projects like the Linking Open Drug Data (LODD) [22] follows the first approach (Linked Data). They have the objective of linking publicly available datasets ranging from impacts of the drugs to the results of clinical trials and establishing a Linked Data Cloud for this domain. In contrast, the VPH-Share project deals additionally with the management of the provisioning and the live-cycle of these datasets.

For instance the European Project Hypergenes [23] developed a data warehouse based infrastructure for providing semantic access to project relevant datasets. The European Project Health-e-Child [24] built a Grid-enabled network for sharing and annotating biomedical data on the basis of Grid technologies. In this case the data sources can be distributed as in the VPH-Share project. The VPH-Share project goes one step further in providing a generic toolchain enabling the utilization of an emerging dataset provided to the community. The European Project Debug IT [25] followed a similar approach towards semantic integration of distributed data sources by virtually consolidating the underlying datasets but focused on the integration of these data sources. The VPH-Share DMP objective is to support the whole life-cycle of data management, including provisioning, selection, annotation and deployment of the datasets.

VIII. CONCLUSION AND FUTURE PLANS

The VPH-Share project provides a data management platform supporting the selection, the annotation, and the publishing of VPH-relevant datasets to VPH-Share stakeholders via the Cloud. The data publication suite enables data providers to select parts of their data, annotate it with ontological concepts, and to select a Cloud site where the data is published.

The VPH-Share Cloud environment, called Atmosphere, is exclusively implemented on the basis of services. Its objective is to gain users authorized access to computational and dataset services deployed on a distributed hardware infrastructure. The platform is implemented as a set of modules operated on a dedicated host and the cloud middleware stack managing the

[1] Note: for simplicity and space this query has been shortened: SELECT ?s WHERE {{SERVICE <http://vphshare1/pvp/sparql>{ ?s <rdf#type><http://NCICB#Patient>} UNION SERVICE <http://vphshare1/crim/sparql>{ ?s <rdf#type><https://vphshare1/crim/unannotated#root>}}}

physical resources. The main concept in providing computational or data services is to encapsulate the application to be hosted together with its Web service environment in virtual appliances, called atomic services, which can be provided and scaled per click via the VPH-Share master interface.

In this paper we focused on atomic services for providing access and integration of data in the Cloud. These atomic services are implemented on the basis of the VPH-Share dataset service environment. Different types of atomic dataset services are provided: (1) atomic dataset services expose a Web service interface to single datasets, (2) atomic relational virtual dataset services rely on a data mediation engine enabling the provisioning of a taylor-made view on top of multiple possibly heterogeneous and distributed datasets, (3) atomic semantic virtual dataset services enable execution of distributed SPARQL queries (SPARQL federation extension) against multiple distributed and semantically annotated datasets.

First performance results on the basis of a typical VPH-Share scenario indicate that the benefit of using mediation as well as SPARQL federation outperforms the performance degradation. The performance of both engines is mainly determined by the slowest underlying dataset.

Currently these atomic dataset services are available on a test basis and first results have been achieved by implementing a typical VPH-Share scenario. A major future challenge is the provisioning of all atomic service types to the whole VPH-Share community. By utilizing this system at a large scale, the performance of all engines has to be evaluated in more detail, especially scalability (number of datasets involved and concurrency have to be examined). Additionally, integration of the VPH-Share security framework, including delegation of security tokens, is a major issue.

Acknowledgment

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2011-2015) under grant agreement #269978 (VPH-Share Project).

References

[1] H. Rajasekaran, P. Hasselmeyer, L. L. Iacono, J. Fingberg, P. Summers, S. Benkner, G. Engelbrecht, A. Arbona, A. Chiarini, C. Friedrich, M. Hofmann-Apitius, B. Moore, P. Bijlenga, J. Iavindrasana, H. Müller, R. Hose, R. Dunlop, A. Frangi, and K. Kumpf, “@neurIST - Towards a System Architecture for Advanced Disease Managment through Integration of Heterogeneous Data, Computing, and Complex Processing Services,” in *IEEE International Symposium on Computer-Based Medical Systems*. Jyväskylä, Finland: IEEE Computer Society Press, June 2008, copyright (C) IEEE Computer Society.

[2] VIROLAB, “EU IST STREP Project, 027446, http://www.virolab.org/,” 8 2011.

[3] EuHeart, “Integrated cardiac care using patient-specific cardiovascular modeling, http://www.euheart.eu,” 8 2011.

[4] VPHOP, “The Osteoporotic Virtual Physiological Human: http://www.vphop.eu,” 8 2011.

[5] S. Benkner, J. Bisbal, G. Engelbrecht, R. D. Hose, Y. Kaniovskyi, M. Köhler, C. Pedrinaci, and S. Wood, “Towards collaborative data management in the vph-share project,” in *Proceedings of the Intl. Workshop on Cloud Computing Projects and Initiatives, in conjunction with Euro-Par 2011*. Bordeaux, France: Springer, Aug 2011.

[6] M. Koehler, R. Knight, S. Benkner, Y. Kaniovskyi, and S. Wood, “The vph-share data management platform: Enabling collaborative data management for the virtual physiological human community,” in *Proceedings of the 8th International Conference on Semantics, Knowledge & Grids (SKG 2012*. Beijing, China: IEEE, Oct 2012.

[7] M. Franklin, A. Halevy, and D. Maier, “From databases to dataspaces: a new abstraction for information management,” *SIGMOD Rec.*, vol. 34, no. 4, pp. 27–33, Dec. 2005.

[8] C. Pedrinaci and J. Domingue, “Toward the next wave of services: Linked services for the web of data,” *Journal of Universal Computer Science*, vol. 16, no. 3, pp. 1694–1719, 2010.

[9] M. Köhler and S. Benkner, “VCE - A Versatile Cloud Environment for Scientific Applications,” in *The Seventh International Conference on Autonomic and Autonomous Systems (ICAS 2011)*, Venice/Mestre, Italy, May 2011.

[10] S. Benkner, A. Arbona, G. Berti, A. Chiarini, R. Dunlop, G. Engelbrecht, A. F. Frangi, C. M. Friedrich, S. Hanser, P. Hasselmeyer, R. D. Hose, J. Iavindrasana, M. Köhler, L. L. Iacono, G. Lonsdale, R. Meyer, B. Moore, H. Rajasekaran, P. E. Summers, A. Wöhrer, and S. Wood, “@neurist: Infrastructure for advanced disease management through integration of heterogeneous data, computing, and complex processing services,” *Information Technology in Biomedicine, IEEE Transactions on*, vol. 14, no. 6, pp. 1365–1377, November 2010.

[11] M. Antonioletti, M. Atkinson, R. Baxter, A. Borley, C. Hong, P. Neil, B. Collins, N. Hardman, A. C. Hume, A. Knox, M. Jackson, A. Krause, S. Laws, J. Magowan, N. W. Paton, D. Pearson, T. Sugden, P. Watson, and M. Westhead, “The design and implementation of grid database services in ogsa-dai: Research articles,” *Concurrency and Computation : Practice and Experience*, vol. 17, no. 2-4, pp. 357–376, 2005.

[12] C. Buil-Aranda, M. Arenas, and O. Corcho, “Semantics and optimization of the sparql 1.1 federation extension,” in *Proceedings of the 8th extended semantic web conference on The semanic web: research and applications - Volume Part II*, ser. ESWC’11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 1–15. [Online]. Available: http://dl.acm.org/citation.cfm?id=2017936.2017938

[13] A. Wöhrer, P. Brezany, and A. M. Tjoa, “Novel mediator architectures for grid information systems,” *Future Generation Computer Systems*, vol. 21, no. 1, pp. 107 – 114, 2005.

[14] P. Nowakowski, T. Bartynski, T. Gubala, D. Harezlak, M. Kasztelnik, M. Malawski, J. Meizner, and M. Bubak, “Cloud platform for vph applications,” in *8th International Conference on eScience 2012*. Chicago, USA: IEEE, Oct 2012.

[15] M. Malawski, J. Meizner, M. Bubak, and P. Gepner, “Component approach to computational applications on clouds,” *Procedia Computer Science*, vol. 4, no. 0, pp. 432 – 441, 2011, ¡ce:title¿Proceedings of the International Conference on Computational Science, ICCS 2011¡/ce:title¿. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050911001037

[16] S. Koulouzis, R. Cushing, A. Belloum, and M. Bubak, “Cloud federation for sharing scientific data,” in *8th International Conference on eScience 2012*. Chicago, USA: IEEE, Oct 2012.

[17] DICE, “The DICE team website, http://dice.cyfronet.pl/projects/details/VPH-Share,” 10 2012.

[18] M. Köhler and S. Benkner, “A service oriented approach for distributed data mediation on the grid,” in *Grid and Cooperative Computing, 2009. GCC ’09. Eighth International Conference on*, Lanzhou, Gansu, China, Aug 2009, pp. 401–408.

[19] S. Benkner, G. Engelbrecht, M. Köhler, and A. Wöhrer, “Virtualizing Scientific Applications and Data Sources as Grid Services,” *Junwei Cao (Ed.), Cyberinfrastructure Technologies and Applications, Nova Science Publishers, New York, USA*, 2009.

[20] ASF, “Apache CXF: http://cxf.apache.org/,” 5 2012.

[21] C. Bizer, T. Heath, and T. Berners-Lee, “Linked Data - The Story So Far,” *International Journal on Semantic Web and Information Systems (IJSWIS)*, 2009.

[22] W3C, “Linking Open Drug Data (LODD) , http://www.w3.org/wiki/HCLSIG/LODD,” 10 2012.

[23] European Project Hypergenes, “http://www.hypergenes.eu,” 10 2012.

[24] A. Branson, T. Hauer, R. McClatchey, D. Rogulin, and J. Shamdasani, “A data model for integrating heterogeneous medical data in the health-e-child project,” *CoRR*, vol. abs/0812.2874, 2008.

[25] D. Teodoro, R. Choquet, E. Pasche, J. Gobeill, C. Daniel, P. Ruch, and C. Lovis, “Biomedical data management: a proposal framework,” in *Stud Health Technol Inform*, 2009.