A Semantic Cloud Infrastructure for Data-Intensive Medical Research

Yuriy Kaniovskyi*, Siegfried Benkner, Chris Borckholder

Research Group Scientific Computing, Faculty of Computer Science, University of Vienna, Austria

Alfredo Saglimbeni

CINECA SuperComputing Centre, Casalecchio di Reno, Italy

Piotr Nowakowski

Institute of Computer Science / Cyfronet AGH University of Science and Technology Krakow, Poland

Tomas Pariente Lobo

Atos Information Technology Company, Madrid, Spain

Steven Wood

Scientific Computing & Informatics, Sheffield Teaching Hospitals NHS Foundation Trust Sheffield, UK

Abstract: The European Virtual Physiological Human Initiative develops a platform, called VPH-Share, for understanding physiological processes in the human body in terms of anatomical structure and biophysical mechanisms. Besides storing, sharing, integrating and linking a wide variety of heterogeneous bio-medical datasets relevant to the VPH community, the project envisions the facilitation of a secure data infrastructure, as well as search and exploration facilities based on semantic technologies. The data infrastructure and management platform are built on top of a hybrid Cloud environment. The data management platform offers tools that cover the whole life-cycle of datasets including integration, selection, semantic annotation and publishing datasets as a service. A comprehensive user interface enables end-users to search and to explore bio-medical data with support of semantic technologies, concealing the complexity of the underlying service environment. In this paper we describe the data infrastructure that has emerged in context of the project.

Keywords: bio-medical data infrastructure, data integration, data management, semantic integration, cloud computing, VPH-Share, web services, secure

1 Introduction

In a highly interconnected society, data, while constantly growing in size, remains highly decentralized. Many enterprises, institutions and practitioners pursue related goals, while resorting mostly to isolated tools and solutions. Hence, consolidation of data available in distributed heterogeneous data sources is becoming an increasingly important issue in various research fields. To gain knowledge and achieve fundamental advances, researchers often require access to a wide variety of data sources relevant to their domain. Funded by the European Union's Virtual Physiological Human Initiative (VPH-I), the VPH-Share project seeks to combine geographically scattered clinical data sources. It develops an integrated Cloud-based service framework to enable sharing, integration and access to scientific data, models and workflows pertaining to the physio pathology of the human body, ultimately aiming towards a more personalized and predictive healthcare and disease prevention.

The datasets of interest to the VPH community are highly heterogeneous, qualitatively and quantitatively multi-scale. They cover information ranging from micro to macro biology, patient data, medical images and biomedical signals. The growing amount and complexity of these data calls for new mechanisms for their management and sharing. The VPH-Share project offers a novel and comprehensive data management platform that supports these objectives including data selection, semantic annotation, integration, and publishing. It provides unified data access and querying mechanisms, with the goal of establishing a versatile data fabric for the VPH-Share medical research community.

To adequately support medical practitioners and researchers, the main user interface of the platform integrates tools for exploring, finding and querying relevant data as well as for utilizing these data in their workflows. Semantic services deployed in the VPH-Share act as coordinators of this process establishing a higher-level knowledge layer on top of the deployed data and application services. Given the sensitivity of patient-specific clinical data, the platform must also comply with legal and ethical regulations. For this reason patient-specific data cannot be made available without addressing these constraints. The platform utilizes a custom end-to-end security model and access control mechanisms. In this paper we briefly describe some security aspects as they become relevant.

From a technical perspective, the VPH-Share services are realized on top of a Cloud platform as a set of so-called atomic services. These follow the virtual appliance principle and expose datasets, applications or workflows through web service interfaces. The data management platform (1) consists of two major parts, the Data Publication Suite (DPS) and the Dataset Service Environment (DSE). As a toolset that supports management, provisioning and publication of datasets. the DPS provides a set of graphical tools to de-identify, semantically annotate and publish a data source in context of the DSE. The second part (DSE) exposes datasets through standard web service technologies to the VPH-Share Cloud network. The DSE is based on the Vienna Cloud Environment (2) and consists of two types of services: atomic dataset services are used to expose data sources through relational or semantic access, whereas atomic virtual dataset services transparently integrate multiple atomic dataset services unifying and presenting combined datasets as a single virtual one. In addition, the DSE is compliant to both SOAP and REST standards. Both can be used simultaneously to query a service. The advantage of using both standards

allows for legacy web service communication through the industry-proven SOAP interface, while REST additionally supports plain and easy to use HTTP operations to access datasets. The data management platform follows an incremental Extract-Transform-Load (ETL) process that allows provisioning of an evolving platform as new information sources become available (3). Using semantic data integration, the platform provides on-demand customized views on the available data.

In the following, we describe the anatomy and capabilities of the data infrastructure built on these two service types, with an experimental evaluation based on a project-specific scenario. Drawing conclusions from this evaluation, we present the next steps and technologies that are planned to be implemented in order to secure sustainability and scalability of the infrastructure.

The remainder of this paper is structured as follows. In Section 2 provides a short introduction to the VPH-Share project, describing its overall vision and requirements related to the data infrastructure. Section 3 describes a brief overview of the VPH-Share Cloud environment - the Atmosphere. Section 4 describes the data management platform in detail. Section 5 outlines the semantic services provided within VPH-Share. Section 6 describes the master interface – the main user interface provided for the VPH-Share platform. Section 7 presents the scenario we used for an experimental evaluation of the infrastructure followed by a discussion of the results. Finally, we wrap up with some concluding remarks, related work and future plans.

2 The VPH-Share Project

Accurate diagnosis and successful treatment of complex or rare diseases requires specific expertise that is rarely found in one place. Typically, the necessary knowledge is geographically scattered amongst many research and practical institutions. Combining this knowledge requires an infrastructure for collecting, structuring and disseminating the relevant information.

The ongoing VPH-Share project (4) develops a Cloud-based data infrastructure for sharing, finding, extracting and processing medical and patient data. This includes a distributed file storage system for medical images and other binary files. Semantic services operate atop this layer and provide knowledge-level functionality to the platform. This includes management of semantic concepts for data annotation, a metadata service repository, and data inference and deduction procedures.

Another major objective of VPH-Share is to support execution of applications and sophisticated medical workflows. The four flagship workflows originate from previous related European research projects including @neurIST (5), Virolab (6), euHeart (7), and VPHOP (8).

The main idea revolves around constructing a patient avatar to be used by medical researchers, for a particular condition. This vision is depicted in Figure 1. The project focuses on a key bottleneck - the interface with a wealth of data from medical research and clinical processes. This challenge is addressed by means of a versatile data service infrastructure. The main objectives of the data infrastructure can be summarized as follows:

- Expose data as a service. Data sources must be virtualized and made accessible through web services;
- Facilitate standardized and uniform access (interfaces) to the services;

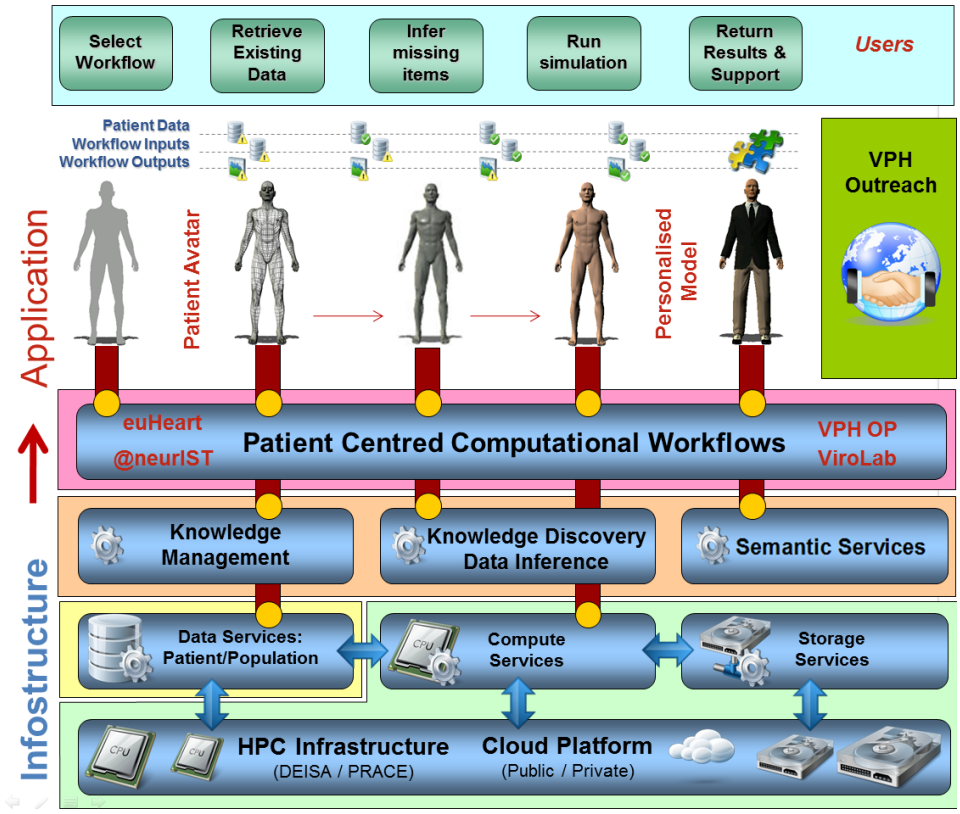

Figure 1 The VPH-Share project consists of a compositional service platform. Services are organized in several layers ranging from hosting and storage infrastructure to computational workflow appliances. The objective is to provide the end-user with a patient model resulting from a medical workflow.

- Enable search and retrieval of data through relational concepts;
- Support semantic access mechanisms and provide a high-level knowledge on the data.

The service architecture adheres to a modular design. This allows accommodating and integrating different types of data sources that can be extended in the future. These types currently include relational (MySQL, IBM DB2, Oracle, PostreSQL, Microsoft SQL Server), XML (eXist) and file storage systems (for Unix, Linux and Windows). The system is capable of transforming (CSV and XML) and delivering data as a streamed download.

The data services must support the classic SQL queries and simple structured queries based on ontological terms stored as RDF (Resource Description Framework) (9) triplets. SPARQL is used for semantic querying. The addressed data sources can either be queried directly though their respective exposing service, or through a federated service. Data sources must be in the form of web service resources. The service interface should adhere to well-established standards, provide a high-level API, a client toolkit and other utilities for provisioning, auditing and management.

The following sections will describe the platform layers from the bottom up, with a special focus on data management and intelligence.

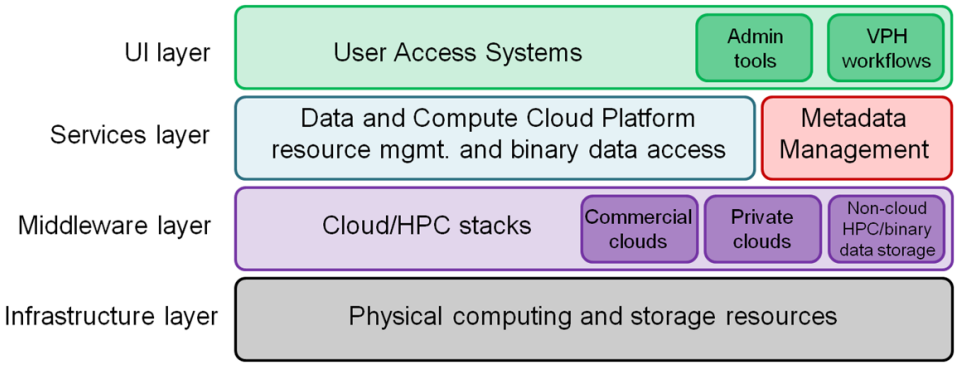

Figure 2 High-level view of VPH-Share Cloud Infrastructure

3 The VPH-Share Cloud Environment

A basis for all services deployed and used within the platform is the hosting hybrid Cloud environment called Atmosphere (10). Its main objective 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 (11). The platform offers a persistent hosting infrastructure enabling users to deploy, instantiate and manage access to the VPH-Share services. In addition, Atmosphere provides users with control interfaces, enabling interaction with the services and the environment.

The cloud platform covers the entire application development life cycle, ranging from inception to scalable exploitation. It assists each participating class of users at each step of the design, deployment and enactment process. The Cloud platform provides a large set of atomic service templates and supports building custom ones from scratch. Thus, Atmosphere forms 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 summarized in Figure 2, 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. Embeddable UI extensions allow Atmosphere features to operate in standalone mode or be imported into a portal. The core components of Atmosphere can be instantiated multiple times, provided they share a common registry. This feature allows the Cloud platform to scale along with the number of services and clients. Atmosphere also includes a Cloud-based binary data federation component called LOBCDER (12). This component stores medical images and other related binary data.

Atmosphere merges and utilizes multiple private and public Cloud sites. It enables an elastic hosting infrastructure that offers numerous compute, storage and network resources. The API endpoints of the individual Cloud sites are used to deploy services and provide statistics of the current resource load. This allows Atmosphere to monitor and manage system load. Since Atmosphere is compatible with the Amazon EC2 Cloud, it can make use of resources procured from this large-scale commercial provider to enhance its capabilities and extend its available resources.

Bound by legal constraints, the service infrastructure is protected from unauthorized access by a security framework that incorporates a number of security features as a wrapper for all deployed atomic services. These features revolve around specifying secure end-to-end communication of the services and managing access to them through the user portal. To grasp the scope of the security framework requirements, it is important to understand that VPH-Share integrates a dynamic set of diverse application services. The security framework cannot address all security requirements specific to each integrated application, especially since new applications with new requirements can be added at any given time, leaving this responsibility to service developers. In (13) we present how atomic dataset services interact with the security framework and fulfill their security-related requirements. A more detailed description of the Cloud platform and its components can be found in (14).

4 Data Management Platform

The data management platform is a software stack responsible for data management, access, and integration functionality within the project. It consists of two major parts: the data publication suite (DPS) and the dataset service environment (DSE).

The DPS enables selection, de-identification, ontological annotation and, finally, publication of data from a data source. Upon publication, the DPS uses the service provisioning tools of the DSE to expose the data in the VPH-Share Cloud. The DSE is responsible for all aspects of Cloud-enabling of data sources and exposing them through REST and SOAP-Web service interfaces. The environment also supports the provisioning of virtual dataset services that provide an integrated view on multiple data services based on the concept of schema mediation (see Appendix 9.1). In addition, an integrated linked data environment, based on semantic technologies, can be used to explore and traverse datasets using ontological concepts.

Dataset services support both relational and semantic querying. While relational SQL endpoints are primarily used by workflow services to retrieve data for processing, they can also be easily exported and utilized in legacy tools. Since the DPS ensures that all published datasets are semantically annotated with ontological concepts, they can also be queried by using SPARQL. SPARQL serves for accessing traversing, exploring and finding data by means of established semantic technologies. A conceptual view of the data management platform is shown in Figure 3.

Another substantial foundation for the data infrastructure is laid with the implementation of the federated index store (see Section 7.2). It aims at improving the performance of the data infrastructure by allowing to quickly discover conjunctions in the data, and thus to optimize the data querying and delivering processes.

4.1 Data Publication Suite

The DPS is used for publishing datasets that have been consented for research. It provides a graphical user interface and assists users through the process of exposing and managing datasets deployed in the VPH-Share Cloud, while covering all aspects of a dataset's life cycle. As depicted in Figure 3, the DPS usually resides within an institution prior to publication in the Cloud. Data providers will want to (i) import their local dataset into the tool, (ii) select the data they would like to publish, (iii) select the data that should be de-identified, and (iv) give a higher-level meaning to the data by annotating fields of

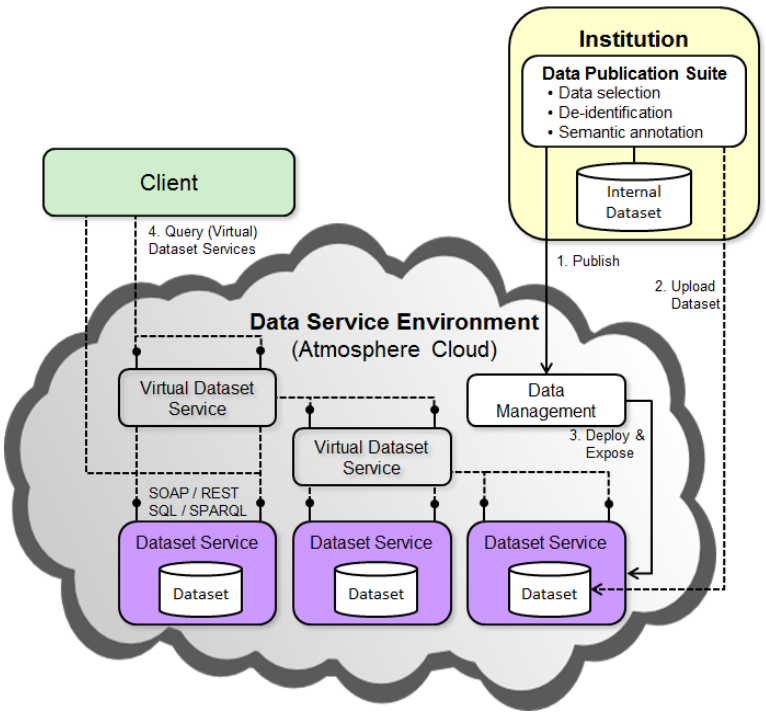

Figure 3 Conceptual view of the Data Management Platform. Data consented for research is selected, de-identified and semantically annotated as part of the publication process {1.}. The data is then uploaded {2.} to the Cloud and exposed {3.} as dataset service(s) to be queried by the Client {4.} or application services. Virtual dataset services facilitate the integration and federation of individual data sets.

the schema with ontological terms (RDF). The application can cope with structured files, Microsoft SQL databases and other OLE DB-ready databases. Due to the modular design the DPS provides extension points for additional data storage bridges if need be. It is the only client software of the DMP required for publishing fully semantically annotated and de-identified datasets to the VPH-Share Cloud.

The DPS follows an incremental Extract-Transform-Load (ETL) process that allows provisioning of an evolving platform as new information sources become available. As part of the ETL process, additional relationships between data items, e.g. foreign key constraints, can be specified. Using semantic data integration, the platform supports on-demand customized views on the data (3).

The user interface of the DPS is shown in Figure 4. Upon establishing a connection to the data source, the DPS outlines schema information on its relational tables. Additionally, the user can view relationships defined through foreign keys and other logical constraints of the data source, including properties of specific attributes. On the right hand side, the user can search for ontological terms in order to adequately annotate the dataset.

Semantic annotation is a drag-and-drop process that starts with a search for existing ontological terms. Users are able to specify free text and the DPS suggests concepts from a predefined set of ranked ontological terminologies, which is delivered by a query to the semantic services (Section 5) of VPH-Share. The translation of relational data into

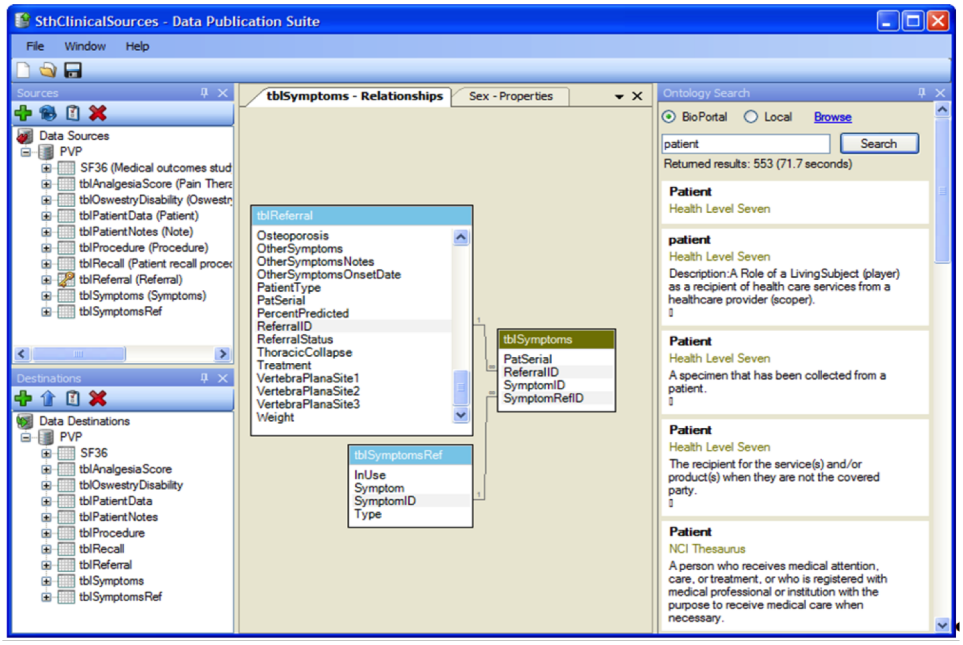

Figure 4 Data Publication Suite from left to right shows: data source and publishing destination; the data schema, relationships and properties; and the ontological term search for semantic annotation.

its semantic counterpart requires a mapping (see Appendix 9.2), generated by the DPS from the terms chosen by the user. After the annotation, the DPS publishes datasets into the Cloud as atomic dataset services. The publishing process instantiates a new atomic dataset service within the context of the DSE in Atmosphere and pushes the data to the new container. During the publication process the de-identification algorithms are applied to the selected fields. The newly created dataset service instance is then marked with an entry to the service metadata catalogue (Section 5), and is thus visible to the infrastructure. Due to the heterogeneity of the VPH-Share Cloud, the site for the service deployment can be specified by the user. After publication, the data service is remotely available and accessible through predefined endpoints.

4.2 Dataset Service Environment

The DSE distinguishes between two service types which together build up the data infrastructure. Dataset services expose individual datasets and virtual dataset services are used for dataset integration through dataset services. As such, virtual dataset services offer transparent access to multiple, potentially heterogeneous data sources via a uniform service interface specification. Therefore, a middleware is provided in order to setup and deploy appropriate dataset services as well as virtual dataset services.

The services are based on the Vienna Cloud Environment middleware (2; 15; 16) and the @neurIST data infostructure (17), which integrates the OGSA-DAI (18) framework for data access and integration of relational datasets, as well as D2R (19; 20) – a tool for publishing relational databases to the semantic web.

4.3 Dataset Services

Dataset services provide remote access to individual dataset sources published in the VPH-Share network. These services expose two representation layers and requests can be sent to either of them. The SOAP interface, on which the client API is based on, allows access for legacy applications through an industry-proven standard. The REST-based interface allows execution of queries via plain HTTP methods. Both interfaces are implemented on the basis of the Apache CXF Web Service Framework.

Dataset services can be queried through either relational SQL or semantic SPARQL language. The use of both endpoints enable the utilization within different tool chains of the project. SPARQL is used primarily for data exploration by data discovery tools such as the query builder of the master interface (Section 6.1) and SQL is used by scientific workflow services to retrieve data for processing.

Query execution of a dataset service is built upon the OGSA-DAI (18) middleware. The core functions of OGSA-DAI cover the basic exposure of a data source as a web service. In addition to the available components for functional extensions, various forms of querying, transformation and delivery of data are supported. OGSA-DAI's features are WSRF(Web Service Resource Framework)-complaint, meaning that the service exposes data sources as stateful resources in the Web. The middleware specifies job execution as a set of tasks. These tasks, so-called OGSA-DAI activities, facilitate data querying, processing and delivery capabilities for the middleware. OGSA-DAI covers basic functionalities for relational access within the dataset services. An additional component, namely D2R, is integrated to realize semantic access to the dataset. Section 4.5 provides a thorough description of semantic access to the datasets.

Contrary to the SOAP interface, the RESTful proxy additionally offers limited data management capabilities through the RESTful interface. This functionality is implemented so that workflows can modify their result-datasets or even create new deposits for them in order to prevent a reprocessing of data for a similar workflow. As such, the RESTful proxy enforces policies by differentiating access URLs to the dataset service. The URL pattern for the web application is

```
http://domain/dataset-name/access-type/endpoint
```

where access type is either a *read* or *write*, and endpoint either *SQL* or *SPARQL*. This URL structure enables policy enforcement by allowing or disallowing users to specific URL endpoints.

4.4 Virtual Dataset Services

Atomic virtual dataset services are virtual appliances within the data infrastructure. They provide virtual datasets by integrating multiple dataset services 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 network and hosting of them on the VPH-Share Cloud. This approach eases the hardware requirements (e.g. requires less disk space). The architecture of virtual dataset services is illustrated in Figure 5.

The OGSA-DAI middleware integrates a distributed query processing (DQP)(21) engine. DQP 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 an optimized manner. When executing a distributed

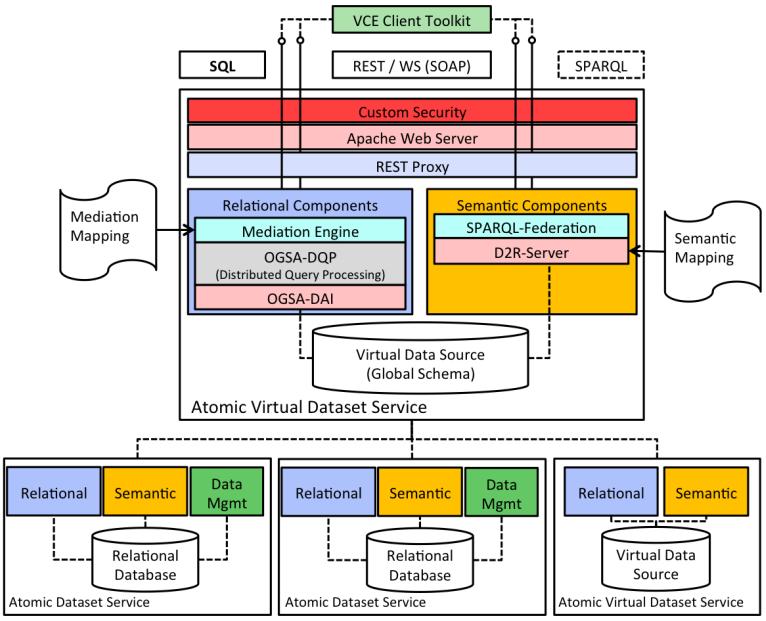

Figure 5 Atomic virtual dataset service architecture showcases relational and semantic components that integrate OGSA-DAI and D2R. Both require mappings described in the Appendices 9.

query using DQP, the workflow defines activities that communicate with remote data or mediation nodes, to obtain or deliver data from or to a remote data node. The DQP engine is implemented as within the OGSA-DAI framework 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 manipulates gathered metadata and data source information to create and execute execution plans over multiple execution nodes.

The mediation engine (22) is built on top of DQP. Relational data integration uses the Global-as-View (GaV) approach to expose a virtual schema (see Appendix 9.1) for a 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. Virtual datasets enable on-demand and transparent integration of datasets (physical or virtual). In contrast to a data warehouse, this approach always provides access to live data because data is not stored in the virtual dataset. Only the integration rules are saved permanently. As such, the system enables hierarchical integration of resources by utilizing virtual datasets as input datasets for other virtual datasets. This is achieved through the generic and uniform web service interface both service types adhere to.

Virtual dataset services also include a semantic processor for federated SPARQL queries based on the W3C SPARQL 1.1 Federation Extension specification (23) to allow a unified access to distributed semantic datasets. The semantic representation of the data enables the use of convenient knowledge discovery mechanisms, employed by the master interface

query builder to datasets based on semantic terms, while mediated datasets provide a domain specific unification of the data.

The virtual dataset service appliances are based on CentOS, an Apache Server, Java, and the DSE release. 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. These services are scalable on the Cloud and can be instantiated on-the-fly. Due to the employed technologies single dataset services are able to conduct data federation. However, whereas datasets cannot be replicated due to administrative, consistency, data transfer and legal constraints, virtual dataset services can. This virtual abstraction layer delegates and orchestrates workloads and querying across the data nodes.

Following the implementation of the security framework for the platform, atomic services operate in a protected environment. Virtual dataset services that, in contrast to single dataset services, act as mediators, do not process requests from a locally available dataset source but rely on interoperation with other dataset services in the VPH-Cloud for aggregating data received from the actual datasets. To enable communications through the security framework, a virtual dataset service impersonates the user for every adjacent operation call to the underlying dataset services by delegating the authentication information across the involved services. This delegation allows for association of an execution of a request with its initiator, even if it is part of a distributed query, which enables auditing capabilities for the data infrastructure.

4.5 Semantic Access to Datasets

SPARQL is a data-oriented semantic querying language used to manipulate and retrieve data stored in a Resource Description Framework (RDF) dataset. A RDF dataset consists of one or more graphs, which contain resources - represented by either an IRI or a literal value - and RDF statements on these resources. A statement is a triplet comprising a subject (e.g. *pantient1*), a predicate (e.g. *has*) and an object (e.g. *age* with value), asserting that the relationship denoted by the predicate holds between the subject and the object. A SPARQL query is a graph pattern and the set of all triple combinations, which match the pattern, are a result of the query. Through these mechanisms, users can query specific data rows pertaining to a single subject (e.g. *patient*), but are also allowed to submit queries pertaining to a specific predicate or object. As such, this concept gives the user the ability to compose unambiguous high-level queries. Since the DPS offers a ranked collection of ontological terms, different datasets that use similar or compatible terms to describe their resources can be linked. SPARQL queries are not restricted to a single graph or dataset. Federated SPARQL queries may span multiple datasets, which enables querying of, e.g. the *patient* concept, across all relevant datasets. The DSE implementation provides a W3C SPARQL 1.1 Protocol endpoint for semantic data access. In combination with VPH-Share's semantic discovery mechanisms, users are empowered to explore and link different data sources.

These concepts lead to the Linked Data Environment(LDE) (24), which is a specification that defines the explorable inter-linkage of structured data through RDF and SPARQL. By using resolvable URIs to denote resources, Linked Data enables the exploration of properties, relationships and values of the specified resources, similar to the use of hyperlinks in HTML. All resources are thus accessible through an URI which references a set of RDF triples describing the resource. Employing these mechanisms a user is able to traverse through the data and discover its relationships with other datasets through a web browser.

To realize semantic functionality and provide the LDE, the open source project D2RQ (20) has been integrated into the DSE. It bridges the gap between relational database and the semantic world using a mapping to translate between SQL databases and RDF triples. D2RQ provides an Apache Jena plugin which rewrites SPARQL queries into equivalent SQL queries. The mapping of relations and attributes to RDF is specified in a configuration file, using Turtle syntax (25) (see Appendix 9.2).

5 Semantic Services

A major goal of the project is to maximize utility of the resources available in the infrastructure. Thus, the platform implements a variety of semantic services and tools deployed in order to facilitate a systematic meta-data management and bind all available resources in the infrastructure to ontological terms (thus providing a higher-level meaning to them).

Although semantic services operate in the background, they are essential for locating and coordinating service, as well as user interaction. The semantic service infrastructure provides support for locating distributed services in the Web, generating semantic annotations to better support their discovery and use within VPH-Share. The main purpose of these inclusions is to import a wide range of existing description formalisms to transform and expose service descriptions as Linked Data. The resulting structure facilitates an efficient service and data annotation.

Metadata management facilities comprise components and services that allow efficient management of the metadata associated with all kinds of resources in VPH-Share. This substantially improves the data discovery process, ontological terminology indexing and information retrieval. The aim of the metadata management services is to provide knowledge-level functionality through an abstraction over VPH-Share resources. Metadata services provide tools for creating, storing, and retrieving metadata associated with any existing resource in the data infrastructure, thus providing a basis for resource discovery. The metadata catalogue supports gathering, storing and managing metadata for all VPH-Share resources, including datasets, files and other services. The exposed datasets provide their metadata in form of a VoID (Vocabulary of Interlinked Datasets) (26) document. The VoID W3C document standard is an RDF schema for describing linked datasets and their interfaces (dereferenceable HTTP URIs and SPARQL endpoints). In addition, the RDF schemas of the exposed datasets are used to facilitate intelligent semantic retrieval processes. Semantic retrieval provides users and services with a unified access to search over structured and semantically annotated datasets.

In addition, components of the metadata management support gathering and indexing of ontologies and terminologies from VPH-Share relevant external Web resources and allow their optional linkage within VPH-Share. The services support the semantic annotation process, as described in Section 4.1. Supported ontologies include SNOMED Clinical Terms, NCI Thesaurus, NCI Metathesaurus, HL7, and ICD-9. Administrative users are able to import and index new ontologies, as well as to create semantic mappings between them. This feature is centralized to ensure correct and synchronized terminology adoption for use in VPH-Share. To establish a proper collection of terminologies, a dedicated repository was deployed. In addition, a ranking system of terminology elements is provided to adequately support users through the data annotation process. The services are also utilized in the user

access system's semantic query builder to support semantic searching facilities across the data infrastructure.

6 User Access System

The VPH-Share user access system, called the master interface (4), provides a secure and user-friendly interface to the VPH-Share platform. It is designed to help formation of a collaborative environment around a complex multi-layered service infrastructure. The interface API covers features ranging from user and resource access to resource and policy management. Moreover, the portal comprises tools for discovery, exploration and traversal of the VPH-Share data infrastructure. The master interface employs a generic service invocator to allow execution of workflows in a generic manner. All information regarding service invocation is obtained through semantic services in VPH-Share.

The master interface uses the OpenID authentication standard for authorizing users. It allows users to log-in using the credentials obtained from a OpenID Identity Provider, which is BiomedTown (27). The biomedical research community of BiomedTown gathers participating parties such as data providers, developers and clinicians, forming a trust model between them and the virtual organization of VPH-Share. A login request or application call is followed by a query to BiomedTown, where the portal either grants or denies access to the platform.

The VPH-Share Cloud UI provides essential tools for declaring a number of endpoints (TCP- or HTTP-based) for services through the VPH-Share master interface. The endpoints can be searched with tools provided by the master interface and used to build application workflows involving multiple services.

6.1 Data Search Facilities

Discovery tools offered by the master interface provide an effective way to explore the resources of the platform. Using a simple text search, the interface supports discovery of services, workflows, and datasets deployed on the infrastructure.

More detailed search options are available through the use of SPARQL and RDF. Initially, a user enters a semantic term or a combination of terms, and queries the metadata services to get a list of resources that refer to that term. The returned list of resources shows how many matches each dataset resource has encountered. Further refinement through selection and filtering of additional semantic concepts can help users when searching with a broad term, such as *patient*. The datasets found can then be explored through the linked data environment.

The master interface facilitates a query builder GUI that allows constructing SPARQL queries for datasets. By using logical operators *AND* and *OR*, a user can drag the terms into the container areas for querying, as shown in Figure 6. When a concept is dragged, the user can select either an exact match, or an inclusion match for the specific term. The query builder then constructs a corresponding query and passes it to SPARQL endpoints of the dataset service.

Similar to the user interface of the DPS, the master interface provides a web page for exploring a single data source available at a dataset service. The web page allows users to (i)explore datasets in its relational form, (ii) download selected data in the CSV and XML format, and (iii) execute custom queries. Additionally, D2R libraries integrated into the

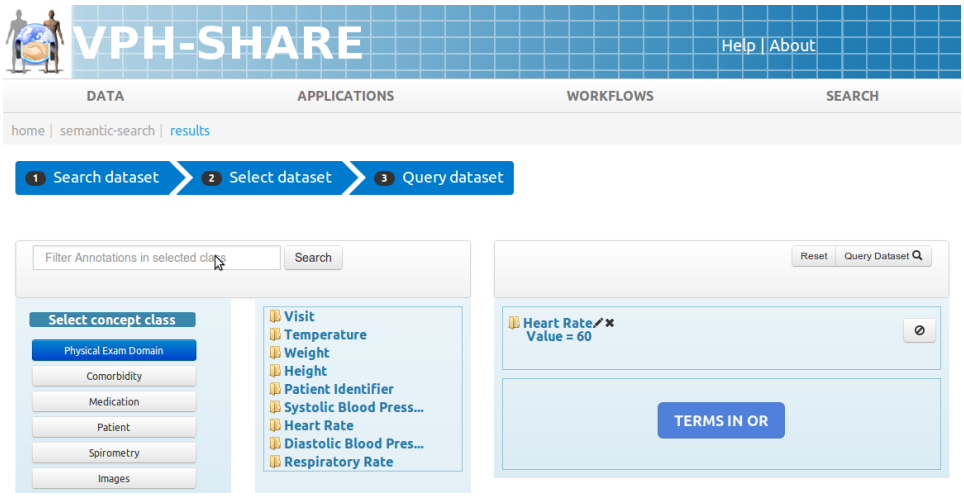

Figure 6 Master interface: query composer

dataset services provide a web page following Linked Data concepts for browsing the data using semantic terms.

7 Experimental Evaluation

This section summarizes experimental scenarios and results presented in (22) and (13), and offers a more in-depth discussion of the results and future work related to the design and development of optimization mechanisms.

Atomic virtual dataset service allow for the data infrastructure scalability. As opposed to atomic dataset services, which are not allowed to be replicated to preclude data desynchronization, atomic virtual dataset services can be replicated, with the intention of scalability in mind. Since virtual dataset services apply a big bulk of data pulling, aggregation and transformation functions, a delegation and balancing of the workload can be achieved through instantiating multiple virtual dataset services. To understand the overall performance of the data infrastructure, we carried out a series of tests, executed on top of the VPH-Share data infrastructure. These tests involved different Cloud sites of the Atmosphere and present a measure of performance to be expected from a virtual dataset service.

The testbed has been set up according to the requirements of the project. Two related dataset sources, originating from the Sheffield Teaching Hospitals containing clinical data are exposed using two separate atomic dataset services within the Vienna private Cloud. Atomic virtual dataset services on the other hand are deployed into the following three Clouds of the Atmosphere:

- Local (Vienna): The service is deployed in the same Cloud as the dataset services. The virtual dataset service instance is assigned 2 vCPUs and 1GB memory;
- Remote (Krakow): The service runs on a remote Cloud. The instance is assigned 1 vCPUs and 1GB memory;

- Commercial (Amazon EC2): The service runs on European Amazon EC2 instances the specification of VMs types can be viewed in (28).

The most important question that was addressed with regard to performance is whether the query execution is largely dominated by the transfer time of data and other network-related issues, or if more expensive and better equipped virtual instances can achieve a higher performance. Answering this question should give a strategic direction on how to best leverage the Cloud infrastructure.

For this purpose, the available data were substantially inflated through random generation to provide a more realistic measure of performance on the volume of transported data. The client application, placed in Vienna, is set to receive results in the WebRowSet XML format, which further inflates the amount of transferred data. Each request aggregates both dataset sources through a table join returning the following three classes of data:

- small - 2.000 rows, 497.211 bytes (485 KB)
- medium - 20.000 rows, 2.334.666 bytes (2.22 MB)
- large - 200.000 rows, 18.980.031 bytes (18.1 MB)

The client measures the time in milliseconds until the final result is received. Both, dataset services and virtual dataset services are tested for performance and overhead for data mediation. Each test was carried out 30 times, recording average query execution times. The first two executions were excluded, to ensure that service contexts have been fully initialized.

7.1 Performance results

Table 1 highlights a performance reference in terms of local request and data fetching, as all services are deployed in the same network in Vienna. In this first experiment, dataset services return results according to the defined data classes. The virtual dataset service aggregates data from both dataset services and apply transformations to the specified format. Median times differ only slightly from the average times presented above, as the recorded variation was rather low and had no significant outliers.

Table 1 Comparison between Atomic Dataset and Atomic Virtual Dataset Services

Data size	Dataset service	Virtual Dataset Service
Large	1599ms	3494ms
Medium	259ms	442ms
Small	132ms	283ms

Performance results of the dataset services executing data requests within the same Cloud site. Execution times roughly double when two dataset sources with the same data size are mediated and merged.

The results show how the query execution time roughly doubles when mediating two dataset services. This overhead was expected due to the intensive orchestration of the data mediation, temporary data storage, transformation and aggregation.

Table 2 provides detailed performance results comparing atomic virtual dataset services deployed on the local, remote and commercial Clouds of the Atmosphere. The table is sorted by performance.

Table 2 Cloud mediation scenario

Instance	Large		Medium		Small	
	Average	Median	Average	Median	Average	Median
Local	3591	3418	558	546	404	423
Remote	4839	4574	1160	1164	952	945
EC2 c1.xlarge	11871	11747	1589	1570	1099	1076
EC2 m2.4xlarge	12731	12638	2257	2129	1316	1264
EC2 m1.large	12771	12683	1715	1665	1120	1101
EC2 c1.medium	12894	12701	1861	1748	1130	1094
EC2 m2.xlarge	13239	13425	1600	1552	1091	1054
EC2 m2.2xlarge	14709	14777	1760	1666	1102	1080
EC2 m1.medium	14797	14752	1883	1865	1204	1148
EC2 m1.xlarge	14780	14919	1854	1808	1210	1138
EC2 m1.small	17410	17502	2257	2129	1316	1264

This scenario tests the performance of the federation services in several cloud environments. The local (Vienna) instance serves as a reference, since it is in the same cloud as the data services. The remote instance is placed in a dedicated private cloud in Poland, while the following instances are deployed in the Amazon EC2 cloud. The list is sorted by execution time performance.

While the difference in performance between the dedicated local and remote Clouds is minimal, the difference between the two instances deployed in private Clouds and that of the Amazon EC2 instances is significant. As expected, private Clouds such as the one in Vienna and Krakow typically have less traffic than a popular commercial Cloud. However, the question on whether having more resources in an instance can improve the performance of the virtual dataset service is now answered. The performance is primarily driven by the transfer rate. Nevertheless, a careful selection of a suitable instance type may provide a significant benefit.

The instances are categorized by Amazon into several instance types (28). The m1-type are general purpose instances. These are cheap and have a moderate overall network performance. The m2-type instances are still general purpose, but memory optimized instances, while the c1-type instances are equipped with additional compute resources. Finally, the hi-type instance is a special instance type optimized to work in a cluster. The instances within a type range from small to xlarge according to the increasing amount of resources they are equipped with. It is important to note, that the larger an instance is, the better the network performance it is provided with. This is reflected by the first three top performers. The results also show that the top performer is the high-computing instance c1.xlarge. In fact, a request to the virtual dataset service will spawn several dozen threads for processing, making high-performance instances suitable hosts for dataset services.

Memory-optimized instances, on the other hand, are less beneficial. Additional tests were executed to gain an insight into the performance in relation to memory allocation. While initially the working memory of the JRE was held at 512MB throughout all instances

in order to ensure comparable results, further tests showed that allocating 1GB or 256MB to the JRE yielded similar results to the ones presented here. For example an execution of a large dataset on the Vienna VM with 1GB assigned JRE memory yielded an average of 3578ms as opposed to 3591ms. On EC2 m2.4xlarge 12512ms as opposed to 12731ms.

Nevertheless, the performance results are rather modest for the tested dataset sizes. While dataset services perform relatively well, their virtual counterparts require additional resources and have a substantial mediation overhead. In addition to scaling virtual dataset services out, two additional service types come into play as a foundation to further drive the scalability and performance of the data infrastructure.

7.2 Federated Index Store

The federated index store services aim at optimizing the performance of the data infrastructure by allowing to quickly discover conjunctions within the data and thus optimize the data querying and delivering process.

As it stands, a major challenge of the data technologies is to provide some form of global data schema into which all individual data sources can be mapped. This usually also requires a global semantic model for the query terminology. Due to the heterogeneous nature of the data and the large amount of exposed datasets, using the presented services for querying may not be feasible on an extreme scale. This is because dataset services would query each and every node for the requested data, even if they do not relate to each other, or have no common data results. Processing of such queries would be extremely inefficient and time consuming.

To address this challenge, the project developed a federated index store to ingest all dataset sources, to catalogue them efficiently and to provide a subset of relevant resources for any given query. According to the requirements of the project, two services are implemented to support a scalable federation of data based on two use cases of medical workflows.

The first service focuses on subject aggregation. The feature is driven by workflows that require aggregation of datasets on a single subject (e.g. patient) and use it for their input. To reduce redundant executions, the derived results can be stored to an additional dataset and retrieved thereafter without data mediation, if the workflow requires the same data. This particular service will also be used in the query builder interface, should a user chose to explore data on a particular subject.

The second service provides information on resource intersection. It is set to support dataset services in query construction. While most of the medical datasets usually include the concept of a primary patient ID (or similar identifiers), the querying of all datasets, where no data intersections exist, is inefficient and time consuming. To avoid these costly and unproductive operations, a pre-compute database of indices is prepared. This metadata set tracks shared common subjects, so that queries run by the data integration layer are constrained. We believe that such techniques are essential for achieving the performance needed to offer VPH-Share users an acceptable pace of interaction.

The subject aggregation service can easily obtain records across multiple datasets pertaining to the same ontologically annotated subject through a simple SPARQL query. The resource intersection service, on the other hand, uses SPARQL to break queries down, acquire the count of subjects pertaining to the requested datasets and queue all permutations of the data which result in non-zero counts. This functionality supports the DQP and the mediation engine of the dataset services in building more efficient queries, leaving only those datasets for querying that are relevant for the final result.

8 Related Work

Over the last years, many projects emerged with the objective of intelligently integrating, managing and linking data, while establishing a solid and secure clinical data infrastructure as a basis for research platforms. Different approaches such as the Linked Data, centralized data warehouses, or data mediation approaches emerged. We discuss some of the many related research efforts.

The European GEMSS project (29) was one of the first undertakings aimed at developing a service-oriented Grid-based infrastructure for medical simulation, comprising data and compute services. The @neurIST project (17) focused on supporting the research and treatment of cerebral aneurysms and provided a service-oriented infrastructure including data access services, data mediation services and knowledge discovery services (30) on a relational basis.

The Linking Open Drug Data (LODD) (31) follows the Linked Data approach. They aim at linking publicly available data sources related to drugs and clinical trials for the purpose of consolidating information on drug use in specific clinical phenotype context, together with reports on drug efficacy and effects. They establish a Linked Data Cloud for this particular domain. In contrast, VPH-Share focuses on a wide variety of medical workflows, including a separate workflow engine, while delivering different perspectives on data.

The European linked2safety (32) project follows a similar approach towards semantic integration of distributed datasets as VPH-Share. They aimed at consolidating underlying datasets through the Linked Data environment, while focusing data integration and security. In addition to this, however, the VPH-Share project supports the full life-cycle of data management, including provisioning, selection, annotation and deployment of the data sources.

The European Project Hypergenes (33) built a data warehouse infrastructure for supporting unified access to project relevant datasets. In contrast, the VPH-Share project targets on a higher-level knowledge approach using semantic technologies. Not only is the data semantically enabled, but also the service infrastructure itself, through the use of semantic services as a coordinator in its background.

The European Project Health-e-Child (34) built a Grid-enabled network for sharing and annotating biomedical data bases on Grid technologies. As is the case in the VPH-Share project, dataset sources are distributed. The VPH-Share project, however, goes one step further by providing a generic toolchain enabling utilization of an emerging dataset provided to the community.

The VPH-I related project p-medicine (35) also focuses on building a semantically-enabled data infrastructure for the use in medical research workflows. It provides models for personalized treatment and disease prevention, focusing on the research of cancer. VPH-I envisions to facilitate integration with the VPH-Share infrastructures in order to provide unified access to resources of both projects.

9 Conclusion

The VPH-Share project has developed a comprehensive Cloud-based service framework for sharing of data, information, models and workflows on human physiopathology. VPH-Share utilizes different Clouds through Atmosphere, unifying private and public IT resources.

The VPH-Share dataset service environment provides support for securely accessing and integrating distributed heterogeneous datasets, while its user interface and supporting semantic services establish a knowledge-level data environment.

Semantically annotated dataset services and the Linked Data facility represent the foundation for distributed semantic data access and integration within the VPH-Share project. The annotation process of datasets is supported by semantic services in order to provide users with reasonable high-level meaning to the data. The master interface implements a dynamic query builder, which traverses through the data infrastructure using ontological terms.

The VPH-Share dataset services offer different perspectives on structured data - in relational and semantic form. Both support different means for application services and users to access the datasets. The two standard communication interfaces, SOAP and REST, allow for legacy application services, but also browser interactions with the services. Finally, the data service infrastructure supports scalable data federation techniques across multiple data nodes in both relational and semantic ways.

The master interface allows users to interact with the data infrastructure without requiring the know-how of semantic or relational technologies, while still giving them the possibility of using more advanced features and custom querying for a more deeper data exploration. While the query composer hides the details of the interaction with underlying dataset services behind SPARQL query composition, the linked data environment allows users to explore individual datasets through a semantic web browser.

The presented performance results mandate further effort towards scalability and better performance. For this reason, the federated index store is being implemented and is designed to provide the means to optimize the performance of the data infrastructure on a large scale of dataset sources. Through pre-computation of data intersections and construction of aggregated information based on semantic terms, the sub-query construction for particular datasets is expected to boost the performance of the data infrastructure by a huge margin. The integration and testing of the federated index store services will be the main focus of future work.

Acknowledgements

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

References

[1] S. Benkner, J. Bisbal, G. Engelbrecht, R. D. Hose, Y. Kaniovskyi, M. Koehler, 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.

[2] M. Koehler 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.

[3] 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.

[4] European Project VPH-Share, "http://portal.vph-share.eu/," 6 2014.

[5] 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 Management 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.

[6] VIROLAB, "EU IST STREP Project, 027446, http://www.virolab.org/."

[7] EuHeart, "Integrated cardiac care using patient-specific cardiovascular modeling, http://www.euheart.eu."

[8] VPHOP, "The Osteoporotic Virtual Physiological Human: http://www.vphop.eu."

[9] Resource Description Framework (RDF), "http://www.w3.org/RDF/."

[10] 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.

[11] 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, proceedings of the International Conference on Computational Science, ICCS 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050911001037

[12] 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.

[13] S. Benkner, C. Borckholder, M. Bubak, Y. Kaniovskyi, P. Nowakowski, D. R. Lopez, and S. Wood, "A secure and flexible data infrastructure for the vph-share community," in *the 14th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'13), Taipei, Taiwan, December 16-18, 2013.*, December 2013. [Online]. Available: http://eprints.cs.univie.ac.at/3859/

[14] DICE, "The DICE team website, http://dice.cyfronet.pl/projects/details/VPH-Share."

[15] M. Koehler 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.

[16] S. Benkner, G. Engelbrecht, M. Koehler, and A. Woehrer, "Virtualizing Scientific Applications and Data Sources as Grid Services," *Junwei Cao (Ed.), Cyberinfrastructure Technologies and Applications, Nova Science Publishers, New York, USA*, 2009.

[17] S. Benkner, A. Arbona, G. Berti, A. Chiarini, R. Dunlop, G. Engelbrecht, A. F. Frangi, C. M. Friedrich, S. Hanser, P. Hasselmeyer *et al.*, "@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, 2010.

[18] 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.

[19] C. Bizer, "D2r map - a database to rdf mapping language," 2003.

[20] Chris Bizer, "D2R Server: http://d2rq.org," 5 2012.

[21] M. N. Alpdemir, A. Mukherjee, A. Gounaris, N. W. Paton, P. Watson, A. A. Fernandes, and D. J. Fitzgerald, "OGSA-DQP: A Service for Distributed Querying on the Grid," in *Advances in Database Technology - EDBT 2004*, ser. Lecture Notes in Computer Science, E. Bertino,

S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Boehm, and E. Ferrari, Eds. Springer Berlin / Heidelberg, 2004, vol. 2992, pp. 3923–3923.

[22] S. Benkner, C. Borckholder, M. Bubak, Y. Kaniovskyi, R. Knight, M. Köhler, S. Koulouzis, P. Nowakowski, and S. Wood, "A cloud-based framework for collaborative data management in the vph-share project," in *Proceedings of the Intl. Workshop on Cloud Computing Projects and Initiatives, in conjunction with AINA*. USA: IEEE CPS, March 2013. [Online]. Available: http://eprints.cs.univie.ac.at/3674/

[23] *SPARQL 1.1 Federated Query*, W3C Std., Rev. 1.1, March 2013. [Online]. Available: http://www.w3.org/TR/sparql11-federated-query/

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

[25] Turtle - Terse RDF Triple Language, "http://www.w3.org/TeamSubmission/turtle/," 3 2014.

[26] Describing Linked Datasets with the VoID Vocabulary, "http://www.w3.org/TR/void/," 8 2013.

[27] BiomedTown portal, "http://www.biomedtown.org/," 2 2011.

[28] Amazon EC2 Instances, "http://aws.amazon.com/en/ec2/instance-types/," 8 2013.

[29] G. Berti, S. Benkner, J. W. Fenner, J. Fingberg, G. Lonsdale, S. E. Middleton, and M. Surridge, "Medical simulation services via the grid," in *First European Healthgrid Conference*, 2003.

[30] C. M. Friedrich, H. Dach, T. Gattermayer, G. Engelbrecht, S. Benkner, and M. Hofmann-Apitius, "@neulink: A service-oriented application for biomedical knowledge discovery," *Global Healthgrid: E-Science Meets Biomedical Informatics: Proceedings of HealthGrid 2008*, vol. 138, p. 165, 2008.

[31] M. Samwald, A. Jentzsch, C. Bouton, C. Kallesoe, E. Willighagen, J. Hajagos, M. Marshall, E. Prud'hommeaux, O. Hassanzadeh, E. Pichler, and S. Stephens, "Linked open drug data for pharmaceutical research and development," *Journal of Cheminformatics*, vol. 3, no. 1, 2011.

[32] Linked2Safety Project, "http://www.linked2safety-project.eu/."

[33] European Project Hypergenes, "http://www.hypergenes.eu," 8 2013.

[34] 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.

[35] F. Schera, G. Weiler, E. Neri, S. Kiefer, and N. Graf, "The p-medicine portal - a collaboration platform for research in personalized medicine," *ecancermedicalscience*. [Online]. Available: doi: 10.3332/ecancer.2014.398

Appendix

9.1 Example for relational mediation schema

An XML-configuration file has to be provided to map data sources. It 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

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. For each virtual relation a corresponding mapping element (mapping rules expressed through XML). It supports JOIN, n-ary UNION and projected SELECT operations on the local data resources. The sample mediation schema shows how a join of two different datasets can be expressed with this document. At the bottom the required local relational resources are defined using the DQP configuration syntax.

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" />
```

9.2 Example for the semantic mapping

The DPS uses a generic template to create a D2R mapping from the relational schema to RDF. For each mapped relational table a RDF Schema class is created. Each row is represented by an instance of that class, unambiguous identifiers are derived from the primary key. The table columns are also represented by resources of a column specific class, identified by combining the column name and primary key in an IRI. These class properties allow to add metadata, e.g., a unit of measurement. The value itself is a literal, referenced through the *rdf:value* predicate. Row resources and their corresponding column properties are connected through RDF statements. Instead of using the generated class names and predicates, the DPS allows selecting terms from commonly used ontologies. Appendix 9.2 shows an excerpt from a generated mapping. A table containing patient data, including the gender, is mapped to resources of the type *https://domain/dataset#PatientTable*, where each patient instance has a gender property of the type *https://domain/dataset#PatientTable_gender* with a string value.

```
map:PatientTable rdf:type d2rq:ClassMap;
 d2rq:class <https://domain/dataset#PatientTable>;
 d2rq:dataStorage map:database;
 d2rq:uriPattern "PatientTable/@@PatientTable.autoid|urlify@@".
map:PatientTable_gender rdf:type d2rq:ClassMap;
 d2rq:class <https://domain/dataset#PatientTable_gender>;
 d2rq:dataStorage map:database;
 d2rq:uriPattern "PatientTable/gender/@@PatientTable.gender|urlify@@".
map:PatientTable_gender_Property rdf:type d2rq:PropertyBridge;
 d2rq:belongsToClassMap map:PatientTable;
 d2rq:property map:has_PatientTable_gender;
 d2rq:refersToClassMap map:PatientTable_gender.
map:PatientTable_gender_Value rdf:type d2rq:PropertyBridge;
 d2rq:belongsToClassMap map:PatientTable_gender;
 d2rq:column "PatientTable.gender";
 d2rq:datatype xsd:string;
 d2rq:property rdf:value.
```