Managing Scientific Data: From Data Integration to Scientific Workflows*

Bertram Ludäscher† Kai Lin† Shawn Bowers† Efrat Jaeger-Frank†
Boyan Brodaric‡ Chaitan Baru†

Abstract

Scientists are confronted with significant data-management problems due to the large volume and high complexity of scientific data. In particular, the latter makes data integration a difficult technical challenge. In this paper, we describe our work on semantic mediation and scientific workflows, and discuss how these technologies address integration challenges in scientific data management. We first give an overview of the main data-integration problems that arise from heterogeneity in the syntax, structure, and semantics of data. Starting from a traditional mediator approach, we show how semantic extensions can facilitate data integration in complex, multiple-worlds scenarios, where data sources cover different but related scientific domains. Such scenarios are not amenable to conventional schema-integration approaches. The core idea of semantic mediation is to augment database mediators and query evaluation algorithms with appropriate knowledge-representation techniques to exploit information from shared ontologies. Semantic mediation relies on semantic data registration, which associates existing data with semantic information from an ontology. The Kepler scientific workflow system addresses the problem of synthesizing, from existing tools and applications, reusable workflow components and analytical pipelines to automate scientific analyses. After presenting core features and example workflows in Kepler, we present a framework for adding semantic information to scientific workflows. The resulting system is aware of semantically plausible connections between workflow components as well as between data sources and workflow components. This information can be used by the scientist during workflow design, and by the workflow engineer for creating data transformation steps between semantically compatible but structurally incompatible analytical steps.

*Work supported by NSF/ITR 0225673 (GEON), NSF/ITR 0225676 (SEEK), NIH/NCRR 1R24 RR019701-01 Biomedical Informatics Research Network (BIRN-CC), and DOE SciDAC DE-FC02-01ER25486 (SDM)

†San Diego Supercomputer Center, University of California, San Diego, {ludaesch,lin,bowers,efrat,baru}@sdsc.edu

‡Natural Resources of Canada, brodaric@nrcan.gc.ca

Contents

1 Introduction

Information technology is revolutionizing the way many sciences are conducted, as witnessed by new techniques, results, and discoveries in multi-disciplinary and information-driven fields such as bioinformatics, ecoinformatics, and geoinformatics. The opportunities provided by these new information-driven and often data-intensive sciences also bring with them equally large challenges for scientific data management. For example, to answer a specific scientific question or to test a certain hypothesis, a scientist today not only needs profound domain knowledge, but also may require access to data and information provided by others via community databases or analytical tools developed by community members.

A problem for the scientist is how to easily make use of the increasing number of databases, analytical tools, and computational services that are available. Besides making these items generally accessible to scientists, leveraging these resources requires techniques for *data integration* and *system interoperability*. Traditionally, research by the database community in this area has focused on problems of heterogeneous systems, data models, and schemas [She98]. However, the integration scenarios considered differ significantly from those encountered in scientific data integration today. In particular, the former usually involve "one-world" scenarios, where the goal is to provide an integration schema or integrated view over multiple sources having a single, conceptual domain. Online comparison shopping for cheapest books is an example of a one-world scenario: the different database schemas or web services to be integrated all deal with the same book attributes (e.g., title, authors, publishers, and price).

Compare this situation to the scientific information-integration scenario depicted in Figure 1. A scientist (here, an igneous petrologist) is interested in the distribution of a certain rock type (say *A-type plutons*) within a specific region. He also wants to know the 3D geometry of those plutons and understand their relation to the host rock structures. Through databases and analytical tools, our scientist can gather valuable information towards answering their scientific question. For example, geologic maps of the region, geophysical databases with gravity contours, folation maps, and geochemical databases all provide pieces of information that need to be brought together in an appropriate form to be useful to the scientist (see Figure 1).

We call this integration example a *complex multiple-worlds scenario* [LGM03]. In particular, it is not possible or meaningful to integrate the schemas of the data sources into a single common schema because the various data sources contain disjoint information. However, there are often latent links and connections between these data sources that are made by scientists. Through these *implicit* knowledge structures, the various pieces of information can be "glued" together to help answer scientific questions. Making explicit these knowledge structures–i.e. the "glue"–is therefore a prerequisite for connecting the underlying data. In scientific domains, ontologies can be seen as the formal representation of such knowledge, to be used for data and knowledge integration purposes. The problem in the geoscience domain, however, is that most of this knowledge is either implicit or represented in narrative form in textbooks, and thus not readily available for use in mediation systems as required.

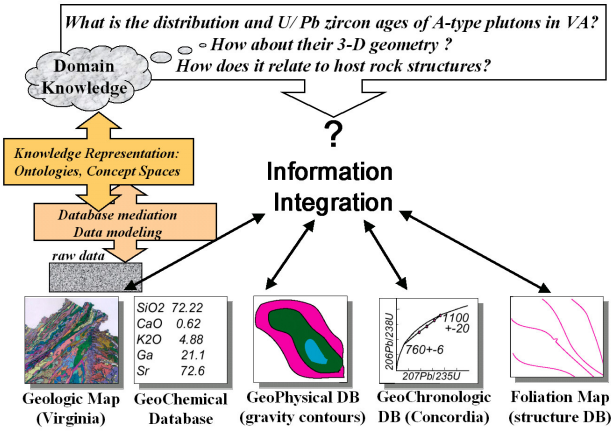

Fig. 1: Complex "multiple-worlds" integration.

As a simple example, consider a geologic map and a geochemical database (see the bottom left of Figure 1). We can link these sources thematically by establishing associations (i) between the formations in the geologic map and their constituent rock types, as indicated in the map legend and associated reports, and likewise (ii) between rock types and their mineral compositions, using e.g. the spatial overlap of formations and geochemical samples. By making these associations explicit, and recording them in domain ontologies, we can establish interesting linkages in complex multiple-worlds scenarios [LGM01, LGM03].

In this paper, we focus on two important aspects of scientific data management: *data integration* and *scientific workflows*. For the former, we present a framework that extends the traditional mediator-based approach to data integration with ontologies to yield a *semantic-mediation* framework for scientific data integration. In this framework, ontologies are

used to bridge the gap between "raw" data stored in databases, and the knowledge level at which domain scientists work, trying to answer the given scientific questions (see the left side of Figure 1).

Data integration and knowledge-based extensions such as semantic mediation deal with modeling and querying database systems as opposed to the interoperation of analytical tools, or the assembly of data sources and computational services into larger scientific workflows. As we will illustrate below, scientific workflows, however, can also benefit from knowledge-based extensions. In fact, an important goal of building cyberinfrastructure for scientific knowledge discovery is to devise integrated problem-solving environments that bring to the scientist's desktop the combined power of remote data sources, services, and computational resources from the Grid [FK99, Fos03, BFH03].

The rest of this paper is organized as follows. In Section 2 we provide a high-level overview of information integration and interoperability challenges in scientific data management. We then present two integration examples in Section 3. The first example illustrates the use of ontologies in scientific data integration using geologic maps (Section 3.1). Additional knowledge represented in a geologic-age or rock-type ontology is used to create different conceptual views of geologic maps and allows the user to query the information in novel ways. Section 3.2 illustrates scientific process integration and combines data-access, data-analysis, and visualization steps into a scientific workflow.

In Section 4 we give an overview of data-integration approaches and discuss technical issues surrounding semantic mediation. Section 5 gives an introduction to scientific workflows and a brief overview of the Kepler system. In Section 5.3 we show that semantic extensions not only benefit data integration, but can also provide new opportunities and challenges for scientific workflows. We conclude with a brief summary of our work and findings in Section 6.

2 Integration Challenges

In this section we provide a high-level overview of the data-integration and system-interoperability challenges that often await a scientist who wants to employ IT infrastructure for scientifc knowledge discovery. Many of these challenges arise from heterogeneities that occur across systems, data sources, and services that make up a scientific data management infrastructure.

Data heterogeneity has traditionally been divided into *syntax*, *structure*, and *semantic* differences [She98, VSSV02]. Scientific data analysis and information-integration scenarios like the one depicted in Figure 1 often involve additional levels, which include low-level *system* integration issues as well as higher-level *synthesis* issues. We briefly discuss these various levels of heterogeneities and interoperability challenges below (see Figure 2).

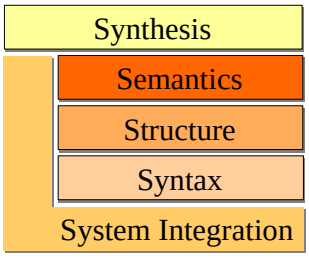

Fig. 2: Interoperability challenges.

System aspects. By system aspects we mean differences in low-level issues relating to, e.g., network protocols[1] (e.g., http, ftp, GridFTP, SOAP), platforms (operating systems), remote execution methods (e.g., web services, RMI, CORBA), and authorization and authentication mechanisms (e.g., Kerberos, GSI). Many efforts for cyberinfrastructure aim at facilitating system interoperability by providing a common Grid middleware infrastructure [FK99, Fos03, BFH03].

System entities requiring management include certificates, file handles, and resource identifiers.

Syntax. Data that is not stored in databases, or that is exchanged between applications can come in different representations (e.g., raster or vector) and file formats (e.g., netCDF, HDF, and ESRI shapefile).[2] The use of XML as a uniform exchange syntax can help resolve syntax differences, but many specialized formats prevail for practical reasons (e.g., because of tool support and efficiency).

Syntactic entities requiring management include individual files in various formats and format conversion tools.

Structure. Structure differences can arise when similar data is represented using different schemas.

[1] These are organized into a layered stack of communication protocols themselves, e.g., the TCP/IP stack and the ISO/OSI layered architecture.

[2] There are also different database representations, or *data models*, but most scientific applications today use relational database systems.

The problem of *schema integration* is a heavily studied area in databases [She98]. There are several difficulties to overcome, e.g., how to derive an integration schema, how to find and define the right mappings between source schemas and the integration schema, how to efficiently evaluate queries against virtual schemas (i.e., which are not materialized), and how to deal with incomplete and conflicting information. In Section 4.1 we give a brief overview of the mediator approach, which addresses some of these issues.

Structure entities requiring management include database schemas and queries (e.g., in SQL or XQuery).

Semantics. Storing scientific data in a database system provides solutions to a number of technical challenges (e.g., multi-user access, transaction control) and simplifies others (e.g., query execution time can be improved and certain structural heterogeneities can be resolved by defining queries that map between schemas, called *views*). However, languages for defining database schemas are not expressive enough to adequately capture most semantic aspects of data. For example, information about the kinds of objects being stored, and how those objects relate to each other or to general concepts of the domain cannot be easily or adequately expressed. Some conceptual-level information can be captured during formal database design (e.g., via an ER conceptual model [Che76]). But this information is rarely connected explicitly to the database, and thus, it cannot be used to query for data and is not otherwise usable by a database system. Traditional metadata can provide a limited form of data semantics and can help a scientist understand the origin, scope, and context of a dataset.

However, to assess the applicability and integrate different datasets for a certain study, many possible semantic heterogeneities among the datasets have to be considered: What were the measurement or experimentation parameters? What protocols were used? What is known about the accuracy of data? And most fundamentally, What concepts and associations are encoded by the data? The usability of data can be improved significantly by making explicit what is know about a dataset (i.e., by describing its semantics), and doing it in such a form that automated processing is facilitated.

Semantic entities requiring management include concepts and relationships from one or more ontologies (e.g., using a standard language such as OWL [DOS04, Usc98, OWL03] or through controlled vocabularies) and data annotations to these ontologies.

Synthesis. By synthesis we mean the problem of putting together databases, including semantic extensions, queries and transformations, and other computational services into a scientific workflow. The problem of synthesis of such workflows encompasses all previous challenges. For example, if a scientist wants to put together two processing steps A and B into the following simple analysis pipeline

$$\xrightarrow{x} \boxed{\mathsf{A}} \xrightarrow{d} \boxed{\mathsf{B}} \xrightarrow{y}$$

many questions arise: In what format does A expect its input x? Does the output d of A directly "fit" the format of B, or is a data transformation necessary? In addition to these *syntactic* and *structural* heterogeneities, *system* and *semantic* issues exist as well. For example, what mechanism should be used to invoke processes and how should data be shipped from A to B? Is it meaningful and valid to connect A and B in this way?

The main challenges for synthesis include process composition and the modeling, design, and execution of reusable process components and scientific workflows.

3 Integration Examples

In this section we provide two different examples, illustrating new capabilities of data integration using semantic extensions (Section 3.1), and of process integration via scientific workflows (Section 3.2). The underlying technologies are discussed in Section 4 and Section 5, respectively.

3.1 Geologic-Map Data Integration

Geologic maps present information about the history and character of rock bodies, their intersection with the Earth's surface, and their formative processes. Geologic maps are an important data product for many geoscientific investigations. In the following, we describe the Ontology-enabled Map Integrator (OMI) system prototype [LL03, LLB$^+$03, LLBB03] that was built as one of the first scientific data integration systems in the GEON project [GEO]. The goal of the system is to integrate information from a number of geologic maps from different state geological surveys and to provide a uniform interface to query the integrated information. Queries are "conceptual," i.e., they are expressed using terms from a shared ontology. For example, it is possible to query the integrated data using concepts such as *geologic age* and *rock type* as opposed to the terms used to describe these attributes in the underlying data. For the OMI

prototype, data from nine different states were integrated. The state geologic maps were initially given as ESRI shapefiles [ESR98] (thus, there were no syntactic heterogeneities). However, each shapefile came with a different relational schema, i.e., the data had structural heterogeneity. A shapefile can be understood as a relational table having one column that holds the spatial object (e.g., a polygon), and a number of data columns holding attribute values that describe the spatial object. For example, the relational schema of the Arizona shapefile is:

```
Arizona(AREA, PERIMETER, AZ_1000_, AZ_1000_ID,
        GEO, PERIOD, ABBREV, DESCR,
        D_SYMBOL, P_SYMBOL)
```

Here, `AREA` is the spatial column, while `PERIOD` and `ABBREV` are data columns with information on the geologic age and formation of the represented region, respectively. The maps from the other states are each structured slightly differently. For example, the Idaho shapefile has 38 columns and contains detailed lithology information, describing rocks by their color, mineralogic composition, and grain size.

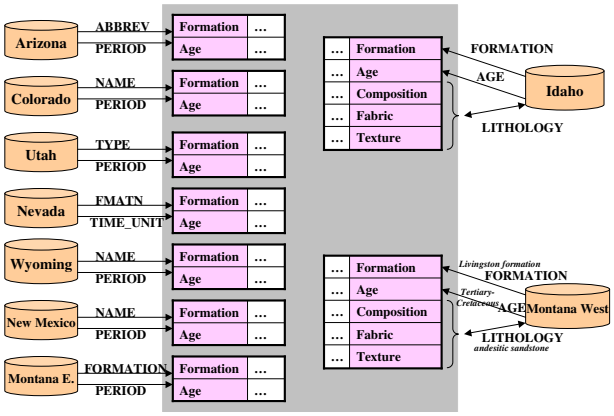

Fig. 3: Schema integration in the OMI prototype: Some elements of the local schemas (outside) are mapped to the integration schema (inside).

Figure 3 shows the association between the columns of the different source schemas and those of the integration schema. Note that the `Formation` attribute (column) in the integrated schema (in the center of Figure 3) is derived from different attributes of the source schemas, i.e., `Arizona.ABBREV`, `Colorado.NAME`, `Utah.TYPE`, and so on. The Idaho and West Montana provide additional detailed information on lithology (e.g., andesitic sandstone). The latter can be used to derive further information on the rock type associated with a region such as mineral and chemical *composition*, *fabric*, and *texture*. This assumes that a corresponding ontology of rock types is given. Using such an ontology we can infer that regions of the map marked as andesitic sandstone are also regions with sedimentary rock (because sandstone is a sedimentary rock) and that their modal mineral composition is within a certain range, e.g., $Q/(Q+A+P) < 20\%$ or $F/(F+A+P) < 10\%$, $P/(A+P) > 90\%$ and $M < 35$ [Joh02].

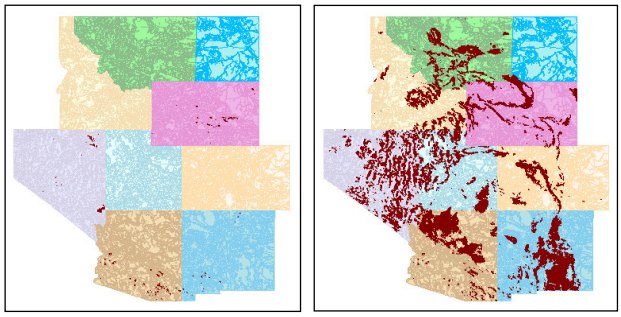

Fig. 4: Results of a query for regions with "Paleozoic" age: without ontology (left), and with ontology (right) [LL03].

Concept-based ("Semantic") Queries. The results of a simple conceptual-level query, asking for all regions with geologic age Paleozoic, are shown in Figure 4. Recall that all nine maps have geologic age information; nevertheless, few results are found when doing a simple lookup for Palezoic in the `Age` column (see the left side of Figure 4). This occurs because by only looking for Paleozoic, we have not taken into account the domain knowledge that other geologic ages such as Cambrium and Devon also fall within the Paleozoic. By using a corresponding geologic age ontology, the system can rewrite the original user query into one that looks for Paleozoic and all its "sub-ages". The result of this ontology-enabled query is shown on the right in Figure 4—where a more complete set of regions is returned.

An important prerequisite for such knowledge-based extensions of data integration systems is *semantic data registration* [BLL04a]. In a nutshell, data objects (here, the polygons making up the spatial regions) must be associated with concepts from a previously registered ontology. In the system, we use several such ontologies, for geologic age (derived from [HAC+89]) and for rock type classification (derived from [SQDO02] and [GSR+99]). In Section 4.3, we discuss further details of data registration and revisit the geologic-map integration example.

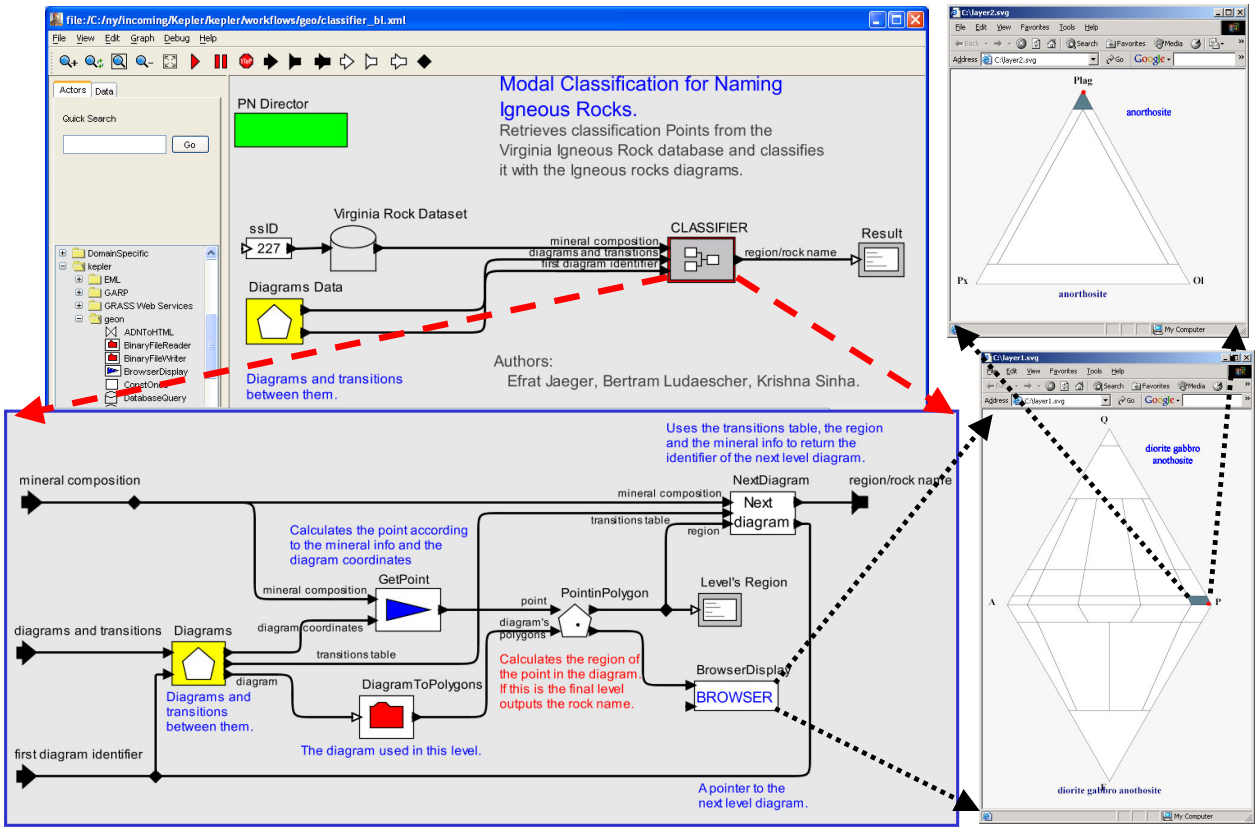

Fig. 5: Mineral Classification workflow (left) and generated interactive result displays (right).

3.2 Mineral Classification Workflow

The previous integration example was data-centric and made use of domain knowledge by using an ontology to answer a conceptual-level query. The second integration example is process-centric and illustrates the use of a scientific workflow system for automating an otherwise manual data-analysis procedure, or alternatively, for reengineering an existing data-analysis tool in a more generic and extensible environment.

The upper left window in Figure 5 shows the top-level workflow, where data points are selected from a database of mineral compositions of igneous rock samples. This data, together with a set of classification diagrams are fed into a CLASSIFIER subworkflow (bottom left). The manual process of classifying samples involves determining the position of the sample values in a series of diagrams such as the ones shown on the right in Figure 5. If the location of a sample point in a non-terminal diagram of order n has been determined (e.g., diorite gabbro anorthosite, bottom right), the corresponding diagram of order $n+1$ is consulted and the point is located therein. This process is iterated until the terminal level of diagrams is reached. The result is shown in the upper right of Figure 5, where the classification result is anorthosite).

This traditionally manual process has been automated in specialized commercial tools. Here, we show how the open source KEPLER workflow system [KEP, LAB+04] can be used to implement this workflow in a more open and generic way (Figure 5). The workflow is shown in graphical form using the VERGIL user interface [BLL+04b].[3] Note that in VERGIL, workflows can be annotated with user comments. Workflows can be arbitrarily nested and subworkflows (e.g., shown in the bottom-left of the figure) become visible by "looking inside" a composite actor.[4] The box labeled CLASSIFIER is a composite actor. VERGIL also features simple VCR-like control buttons to *play*, *pause*, *resume*, and *stop* workflow execution.

KEPLERspecific features of this workflow include a searchable library of actors and data sources (`Actor` and `Data` tabs close to the upper-left) with numerous reusable KEPLER actors. For example, the BROWSER

[3] KEPLER is an extension of the PTOLEMY II system and inherits many of its features, including the VERGIL GUI.

[4] Following PTOLEMY II terminology, a workflow component, whether atomic or composite, is called an *actor* in KEPLER.

actor (used in the bottom-right of the CLASSIFIER subworkflow) launches the user's default browser and can be used as a powerful generic input/output device in any workflow. In this example, the classification diagrams are generated on the client side as interactive SVG displays in the browser (windows on the right in Figure 5). Moving the mouse over the diagram highlights the specific region and displays the rock name classification(s) for that particular region. The BROWSER actor has proven to be very useful in many other workflows as well, e.g., as a device to display results of a previous step and as a selection tool that passes user-requested data to subsequent steps using well-known HTML forms, check-boxes, etc.

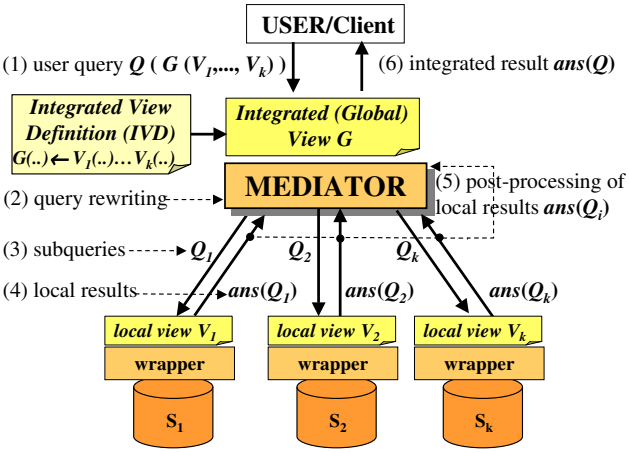

Fig. 6: Database mediator architecture and query evaluation phases (1)–(6).

4 Data Integration

In this section we first introduce the mediator approach to data integration and then show how it can be extended to a semantic mediation approach by incorporating ontologies as a knowledge representation formalism. We also present formal foundations of semantic data registration, which is an important prerequisite for semantic mediation.

4.1 Mediator Approach

Data integration traditionally deals with structural heterogeneities due to different database schemas and with the problem of how to provide uniform user access to the information from different databases [She98] (see Section 2). The standard approach, depicted in Figure 6, is to use a *database mediator* system [Wie92, GMPQ+97, Lev00].

In a mediator system, instead of interacting directly with a number of local data sources $S_1, \ldots, S_k$, each one having its own database schema V_i (also called the *exported view* of S_i), the user or client application queries an *integrated global view* G. This integrated view is given by an *integrated view definition* (IVD), i.e., a query expression in a database query language (e.g., SQL or Datalog for relational databases, or XQuery for XML databases).[5] Here, we consider the case that the integrated global view G is defined in terms of the *local views* $V_1, \ldots, V_k$ exported by the sources. This approach is called *global-as-view* (GAV).[6] For example, our geologic-map integration prototype (Section 3.1) integrates nine local source views into a single integrated global view (Figure 3). The latter is defined by query expressions like the following:

$$\texttt{G}(\text{"}\texttt{Arizona}\text{"}, \texttt{AZ.Aid}, \texttt{Formation}, \texttt{Age}, \ldots) \longleftarrow \texttt{Arizona}(\texttt{Aid}, \ldots, \texttt{ABBREV}, \ldots, \texttt{PERIOD}, \ldots), \texttt{Formation} = \texttt{ABBREV}, \texttt{Age} = \texttt{PERIOD}.$$

$$\texttt{G}(\text{"}\texttt{Nevada}\text{"}, \texttt{NV.Aid}, \texttt{Formation}, \texttt{Age}, \ldots) \longleftarrow \texttt{Nevada}(\texttt{Aid}, \ldots, \texttt{FMATN}, \ldots, \texttt{TIME\_UNIT}, \ldots), \texttt{Formation} = \texttt{FMATN}, \texttt{Age} = \texttt{TIME\_UNIT}.$$

The first rule states that the global view G is "filled" with information from the `Arizona` source by mapping the local `ABBREV` and `PERIOD` attributes to the global `Formation` and `Age` attributes, respectively. Here, spatial regions from the `AREA` column are identified via an `Aid` key attribute. To make the `Aid` attribute values unique across all sources, a unique prefix is used for each source (`AZ.Aid`, `NV.Aid`, etc.) to uniquely rename any potentially conflicting values.

Query Evaluation. Query processing in a database mediator system involves a number of steps (see Figure 6): Assuming a global view G has been defined (which is normally the task of a data integration expert), the user or an application programmer can define a query Q against the integrated view G (1). The database mediator takes Q and the integrated view definitions $G(...) \leftarrow ...V_i...$ and rewrites them into a query plan with a number subqueries $Q_1, \ldots, Q_k$ for the different sources (2). These subqueries Q_i are then sent to the local sources, where they are evaluated (3). The local answers $ans(Q_i)$ are then sent

[5] In complex multiple-world scenarios, coming up with a suitable IVD is a major problem, as one usually needs domain knowledge from "glue ontologies" to link between the sources.

[6] Sometimes it can be beneficial to provide the IVD in a *local-as-view* (LAV) manner, where the local source information is defined in terms of the global schema [Hal01]. Even mixed GAV/LAV and more general approaches exist, but are beyond the scope of this paper; see [Koc01, Len02, DT03].

back to the mediator (4), where they are further post-processed (5). Finally, the integrated result $ans(Q)$ is returned to the user (6).

There are many technical challenges in developing database mediators. For example, the complexity of the query rewriting algorithm in step (2) depends on the expressiveness of the query languages for the user query Q and for the allowed source queries Q_i. The problem of rewriting queries against sources with limited query capabilities is solved (or solvable) only for restricted languages; see, e.g., [VP00, Hal01, NL04b] for details.

4.2 Semantic Mediation

Consider again the geologic-map integration example from Section 3.1. Above we sketched how the structural integration problem can be solved by adopting a mediator approach (see Figures 3 and 6). However, the traditional mediator approach alone does not allow us to handle concept-based queries adequately. For example, in Figure 4, the conventional query for regions with Palezoic rocks yields too few results. In contrast, after "ontology-enabling" the system with a geologic-age ontology, the user query can be rewritten such that it takes into account the domain knowledge from the ontology (see the right side of Figure 4).

The crux of *semantic mediation* is to augment the mediator approach with an explicit representation of domain knowledge in the form of one or more *ontologies* (also called *domain maps* in [LGM01, LGM03]). An ontology is often viewed as an explicit specification of a conceptualization [Gru93]. In particular, ontologies are used to capture some shared domain knowledge such as the main *concepts* of a domain, and important *relationships* between them. For example, the geologic-age ontology used in our OMI prototype can be viewed as a set of concepts (one for each geologic age) that are organized as a hierarchy, i.e., a tree in which children concepts (e.g., Devon, Cambrium) are considered to be a subset of (i.e., a restricted set of ages of) the age described by the parent concept (e.g., Paleozoic). An "ontology-enabled" mediator can use the information from the geologic-age ontology to retrieve not only data that directly matches the search concept Paleozoic, but also all data that matches any subconcept of Paleozoic, such as Devon or Cambrium. Formally, subconcepts are written using the $\sqsubseteq$ symbol, e.g., Devon $\sqsubseteq$ Paleozoic and Cambrium $\sqsubseteq$ Paleozoic in our example.

Concept-based Querying Revisited. A small fragment of a more complex concept hierarchy, taken from [SQDO02], is depicted in Figure 7. This "Canadian system" defines several (preliminary) *taxonomies* (classification hierarchies) of rock types, i.e., for *composition* (Figure 7 shows a small fragment dealing with Silicate rock types); *texture* (e.g., Crystalline vs. Granular); *fabric* (e.g., Planar types such as Gneiss vs. Nonplanar ones such as Hornfels); and *genesis* (e.g., Igneous vs. Sedimentary rock). In Figure 7, subconcepts w.r.t. compositions are displayed as children to the right of the parent concept. For example, a Viriginite is a special Listwanite; moreover, we learn that Listwanite rocks belongs to the Ultramafic kind of Silicate rocks. Using this rock-type ontology, a number of new semantic queries can be formulated and executed with the prototype. Here a semantic query means a query that is formulated in terms of the concepts in the ontologies. Specifically, the lithology information provided by two of the nine state geologic maps (Idaho and Montana West) can be linked to concepts in the *composition*, *fabric*, and *texture* hierarchies, as sketched on the right in Figure 8. On the left of Figure 8, the result of a semantic query for Sedimentary rocks is displayed. Similarly, queries for composition (e.g., Silicate), fabric (e.g., Planar), and texture (e.g., Crystalline), or any combination[7] thereof can be executed.

To enable semantic mediation, the standard architecture in Figure 6 has to be extended to include some form of expert knowledge for linking between otherwise hard-to-relate sources. An important prerequisite is *semantic data registration*, i.e., the association of data objects in the sources with concepts defined in a previously registered ontology. Before going into the technical details of data registration, we first consider informally some ontology variants and alternatives for knowledge representation, ranging from "napkin drawings" to formal description-logic ontologies expressed in OWL [OWL03].

Ontology Variants and Alternatives. One of the reasons to use ontologies in scientific data integration is to capture some shared understanding of concepts of interest to a scientific community. These concepts can then be used as a semantic reference system for annotating data, making it easier to find datasets of interest and facilitating their reuse.

Alternatively, conventional *metadata* consists of attribute/value pairs, holding information about the dataset being annotated (e.g., the creator, creation-

[7] Domain experts will probably only ask for meaningful combinations. Data mining techniques can be applied to this rock-classification ontology to extract only those composition, fabric, and texture combinations that are self-consistent with the ontology. These are of course only an approximation of the actually existing combinations.

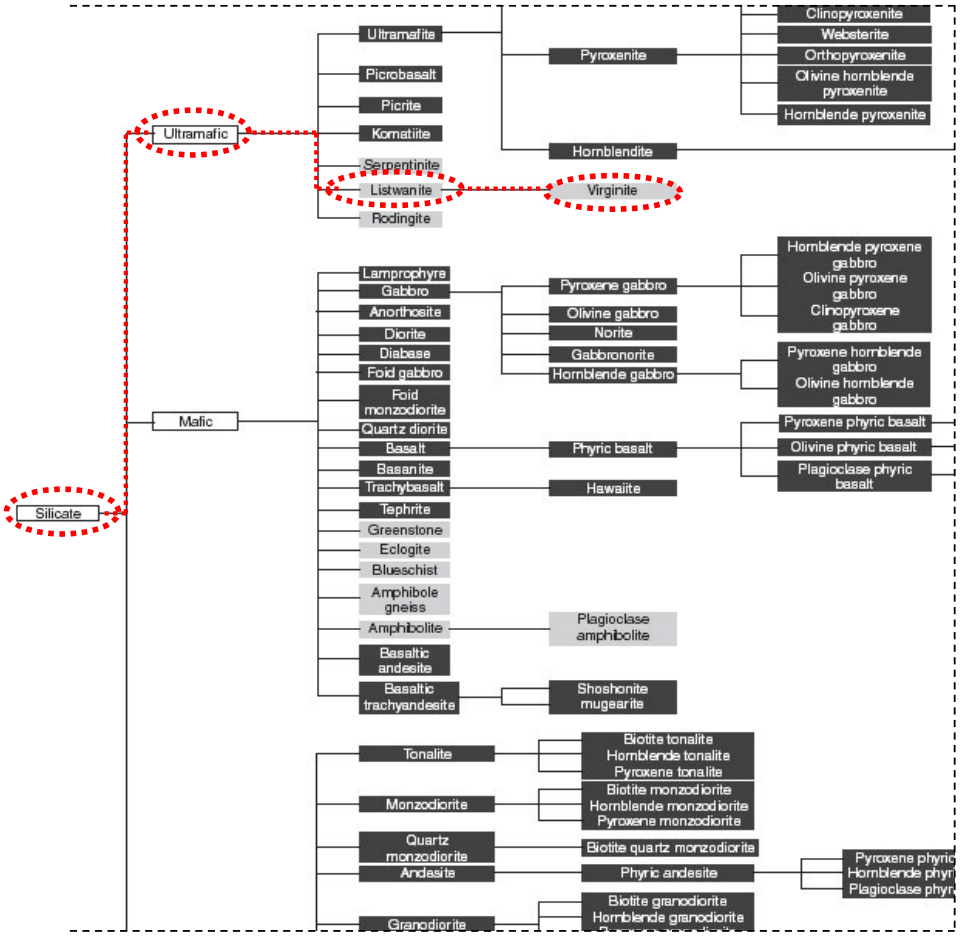

Fig. 7: Rock type classification hierarchy (fragment) based on *composition* [SQDO02].

date, owner, etc.) A *metadata standard* prescribes the set of attributes that must (or can) be used in the metadata description. Metadata standards are often defined as XML Schemas [XML01]. A sophisticated example is the Ecological Metadata Language [NCE], a community standard developed by ecologists that addresses several of the heterogeneity challenges discussed above. An EML description of a dataset can provide information on how to parse the data file (syntax), what schema to use for queries against it (structure), and even indicate some semantic information, e.g., on unit types and measurements.

Controlled vocabularies are often part of a metadata standard and are used to constrain the values of particular attributes to come from a fixed, agreed-upon set. Thus, instead of allowing, e.g., arbitrary rock names in a geologic map shapefile, it is preferable to only use those from a controlled vocabulary. In this way, searches can be guided to only use these terms. Since the terms and definitions are (ideally) agreed-upon by the community, they can also become the starting point of a more formal ontology.

Some controlled vocabularies provide relationships between the allowed terms. Similar to the relationships "narrower term"/"broader term" in a *thesaurus*, the concepts may be organized into a hierarchy or *taxonomy* (for the purpose of classification), where a child concept C is linked to a parent D, if C and D stand in a specific relationship, such as C *is-a* D or C *part-of* D. For example, the geologic age ontology can be seen as a taxonomy with the child-parent relation "*is-temporally-part-of*"; the chemical composition ontology (Figure 7) is a taxonomy with the child-parent relation "*has-composition-like*".

Finally, (formal) *ontologies* not only fix a set of concept names, but also define properties of concepts and their relationships. *Description logics* [BCM+03] are decidable fragments of first-order predicate logic, and are commonly used for specifying formal ontologies. Whereas taxonomies often result from *explicit* statements of *is-a* relationships, the concept hierarchy in a description-logic ontology is *implicit* in the axiomatic concept definitions. In description logics one defines concepts via their properties and relation-

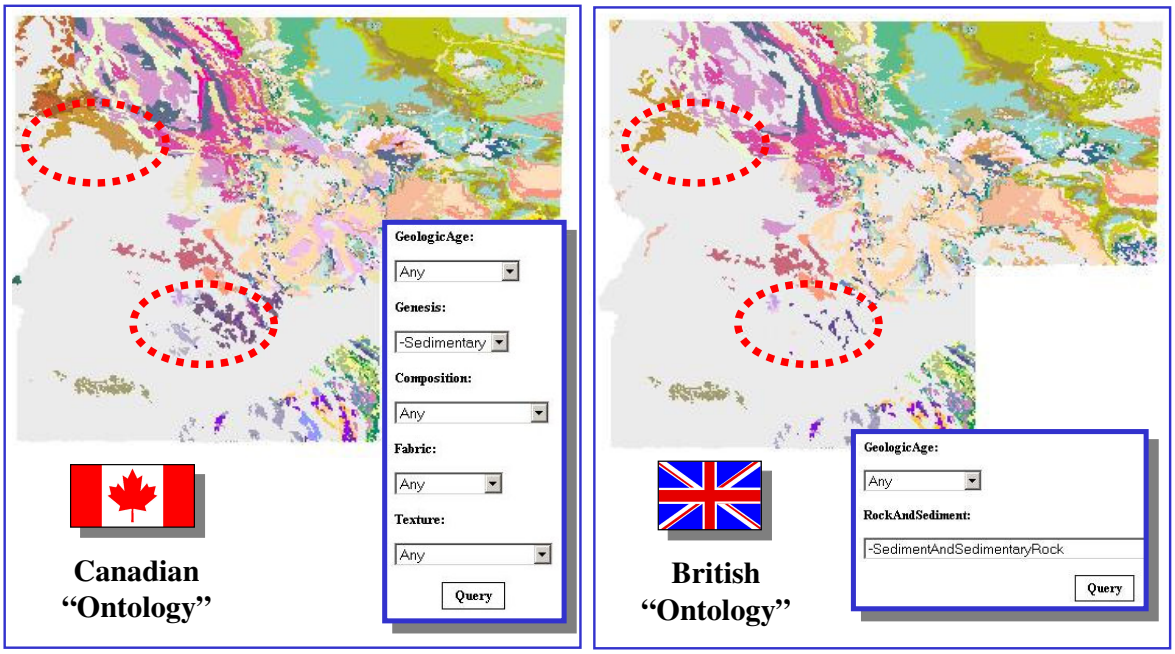

Fig. 8: Different ontology-enabled views on the same geologic maps: the Canadian system [SQDO02] supports queries along several hierarchies (*genesis*, *composition*, *fabric*, *texture*); the British system [GSR+99] provides a single hierarchy (a separate *geologic age* ontology is used in both views). Via an ontology *articulation*, data registered to one ontology can be retrieved through the conceptual view of the other ontology.

ships; the concept hierarchy is then a consequence of those definitions and can be made explicit by a description logic reasoning system. A description logic ontology consists of logic formulas (axioms) that constrain the interpretation of concepts by interrelating them, e.g., via binary relationships called *roles*. For example, the description-logic axiom

$$\mathsf{Virginite} \sqsubseteq \mathsf{Rock} \sqcap \mathsf{Listwanite} \sqcap \exists \mathsf{foundAt}.\mathsf{Virginia}$$

states that (i) instances of Virginite are instances of Rock, (ii) they are also instances of Listwanite, and (iii) they are found at some place in Virginia. It is not stated that the converse is true, i.e., that every Listwanite rock found in Virginia must be a Virginite (this could be stated by using "$\equiv$" instead of "$\sqsubseteq$"). Here, the lowercase symbol foundAt denotes a role (standing for a binary relationship, in this case between a rock types and locations); all other uppercase symbols denote concepts, each one denoting a set of concept instances, e.g., all Listwanites.

Description logics and reasoning systems for checking concept subsumption, consistency, etc. have been studied extensively over many years [BCM+03]. However, until recently there was no widely accepted syntax for description-logic ontologies. With the increased interest in using description-logic ontologies, e.g., for Semantic Web applications [BLHL01], the need for a standard web ontology language has led to the W3C OWL standard [OWL03].[8] OWL, being an XML standard, also supports *namespaces*, a URI-based reference system. In OWL, namespaces are used to help express inter-ontology *articulations*, i.e., formulas that relate concepts and relationships from different OWL ontologies.

Before defining a formal ontology for use in scientific data integration systems, a community-based effort may go through several intermediate steps, from informal to more formal representations. As a rule of thumb, the more formal the knowledge representation, the easier it is for a system to make good use of it, but also the harder it is usually to develop such a formal representation. A common starting point is an informal concept diagram or "napkin drawing" initially created by members of a community to give an overview of important items or concepts in the domain. Figure 9 shows such an informal concept diagram that resulted from a workshop with aqueous geochemistry experts. The diagram relates specific raw and derived data products, models, and scientific problems to one another. While such a diagram is useful as a communication means between domain experts, a data integration system cannot directly make

[8] OWL can be used not only for description logics (OWL-DL) but also for a simpler fragment (OWL-lite) or a more expressive version (OWL-full). In the OMI prototype, we have used simple concept hierarchies expressible in OWL-lite.

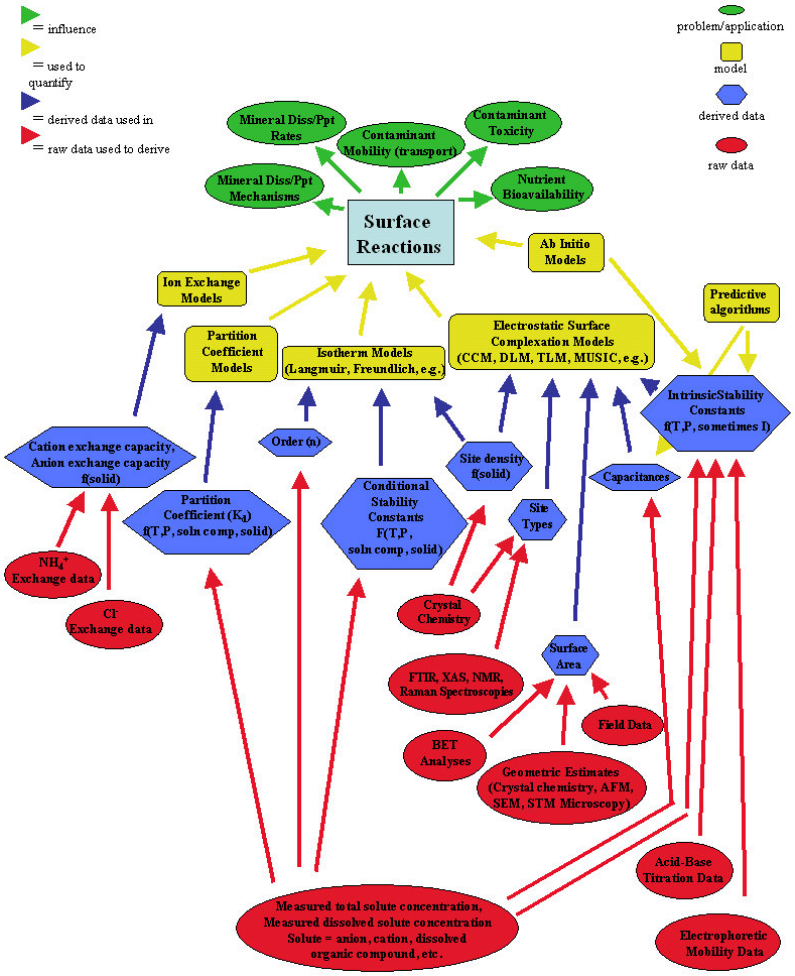

Fig. 9: Informal concept diagram ("napkin drawing"), relating raw data (red ovals), derived data (blue diamonds), models (yellow squares), and scientific problems (green ovals) in aqueous geochemistry.

use of it. One possible elaboration is the definition of metadata standards for data exchange between different community tools and applications, addressing syntactic and structural issues in data integration. Another possible subsequent step is the definition of one or more concept hierarchies (taxonomies) like in the Canadian system (Figure 7), thus enabling simple semantic mediation. A slightly more general form, *labeled, directed graphs*, use nodes for concepts and edge labels to denote arbitrary relationships between concepts. This model can be used by a data integration system to find concepts and relationships of interest via generalized path expressions (e.g., see [LHL+98]).

Finally, logic-based formalisms such as OWL ontologies can be used not only to declare concept names and their relationships, but to intensionally define new concepts relative to existing ones, and to let a reasoning system establish the logical consistency of the system and the concept hierarchy.

4.3 Semantic Data Registration

In this section, we take a closer look at the technical details of semantic mediation, in particular we present an approach called *semantic data registration* that can facilitate data integration in such a system. Figure 10 shows an overview of the architecture of our proposed framework. The main components include services for reasoning with ontologies, database mediation, registration, and data access. We assume that a *federation registry* (not shown) stores core registry information, including the database schemas of the underlying data sources, service descriptions, ontologies and ontology articulations, external semantic constraints (e.g., unit conversion rules and other scientific formulas), and registration mapping rules. formalizing the components of our framework and then discuss resource registration in more detail in the next section.

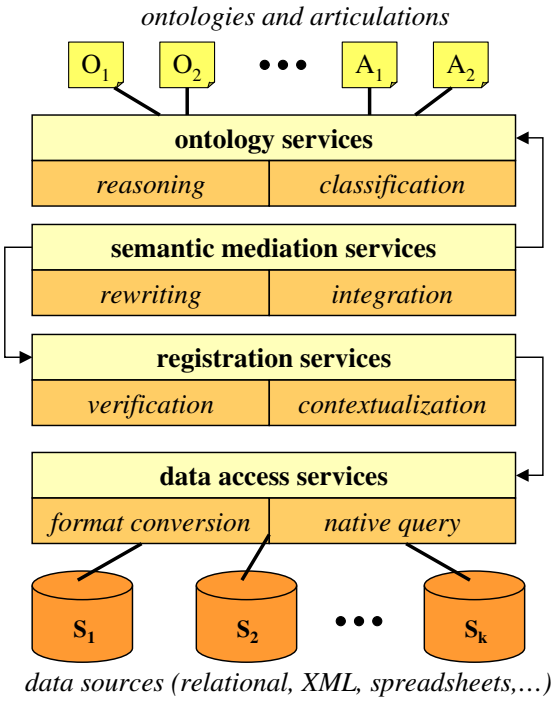

Fig. 10: Overview of the registration architecture.

First-Order Logic. We use first-order logic as a standard, underlying formalism. The *syntax* of first-order logic is defined as usual, i.e., we consider signatures Σ with predicate symbols Σ_P and function symbols Σ_F. By $\Sigma_{P.n}$ ($\Sigma_{F.n}$) we denote the subsets of n-ary predicate (function) symbols; $\Sigma_C = \Sigma_{F.0}$ are constants. *Semantics:* A first-order structure $\mathcal{I}$ interprets predicate and function symbols as relations and functions, respectively; constants are interpreted as domain elements. Given $\mathcal{I}$ and a set of formulas Φ over Σ, we say that $\mathcal{I}$ is a *model* of Φ, denoted $\mathcal{I} \models \Phi$, if $\mathcal{I} \models \varphi$ for all $\varphi \in \Phi$, i.e., all formulas in Φ are satisfied by $\mathcal{I}$. We can implement constraint checking by evaluating the query $\{\bar{x} \mid \mathcal{I} \models \varphi(\bar{x})\}$.

Ontologies (Revisited). Given the above preliminaries, we can now consider an ontology as a certain set of first-order formulas: An ontology O is a set of logic axioms Φ_O over a signature $\Sigma = \mathbf{C} \cup \mathbf{R} \cup \mathbf{I}$ comprising unary predicates $\mathbf{C} \subseteq \Sigma_{P.1}$ (*concepts*), binary predicates $\mathbf{R} \subseteq \Sigma_{P.2}$ (*roles*, *properties*), and constants $\mathbf{I} \subseteq \Sigma_{F.0}$ (*individuals*). Φ_O is usually from a decidable first-order fragment; most notably *description logics* [BCM+03]. A structure $\mathcal{I}$ is called a *model* of an ontology Φ_O, if $\mathcal{I} \models \Phi_O$.

We can view controlled vocabularies and metadata specifications as limited, special cases of ontologies. A *controlled vocabulary* can be viewed, e.g., as a fixed set $\mathbf{I} \subseteq \Sigma_{F.0}$ of *individuals* (constants); a set of named concepts $\mathbf{C}$; or even a full ontology signature Σ (if it contains relationships between terms of the controlled vocabulary). In either case, there are no axioms and hence no defined logical semantics. A *metadata specification* can be seen as an instance of an ontology having only binary predicates $\mathbf{R}$ denoting the metadata properties (e.g., *title*, *author*, *date*; see Dublin Core). Again, the absence of axioms means that no logical semantics is defined.

Namespaces. In the federation registry, we avoid name clashes between vocabularies from different scientific resources (datasets, services, etc.) by assuming each resource has a globally unique identifier i (e.g., implemented as a URI). We then rename symbols accordingly: Every symbol in Σ_i is prefixed with its resource-id i to obtain a unique vocabulary $\Sigma_i' := \{i.s \mid s \in \Sigma_i\}$, allowing new resources to join the federation without introducing identifier conflicts. For example, in the view definitions in Section 4.1, we disambiguated object identifiers by using a state prefix as a resource-id: `AZ.Aid`, `NV.Aid`, etc. A resource-id is also commonly referred to as a *namespace*. Below, by $\mathsf{id}(s)$ we denote the globally unique *resource identifier* of a symbol s.

Registering Ontologies and Articulations. An ontology O is registered by storing its logic axioms Φ_O and its signature Σ_O in the federation registry.[9] An *articulation ontology* A links two ontologies O_i and O_j and is given as a set of axioms Φ_A over $\Sigma_A = \Sigma_{O_i} \cup \Sigma_{O_j}$, thereby logically specifying inter-ontology correspondences. For example, $i.\mathtt{C} \equiv j.(\mathtt{D} \sqcap \exists \mathtt{R}.\mathtt{E})$ is an *articulation axiom* $\varphi \in \Phi_A$ and states that the concept `C` in O_i is equivalent—in terms of O_j—to those `D` having at least one `R`-related `E`. This is expressed equivalently as follows (using first-order logic syntax):

$$\forall x : i.\mathtt{C}(x) \leftrightarrow j.\mathtt{D}(x) \wedge \exists y : j.\mathtt{R}(x, y) \wedge j.\mathtt{E}(y) \quad (\varphi)$$

Note that φ is an *intensional* definition: we have not said how we can access instance objects (implicitly referred to via variables x and y), i.e., how to populate `C`, `D`, etc., as classes of objects. Finally, expressing inter-ontology articulations as ontologies achieves closure within the framework: There is no need to manage a new type of artifact and we can reuse the given storage, querying, and reasoning techniques.

Structural Data Registration. When registering a database, schema-level information and query capabilities should be included to facilitate queries by

[9] We use OWL as the concrete syntax for ontologies.

the end user or a mediator system. Specifically, the database registration information contains:

• The database *schema* Σ_D. In the case of a relational database D, we have $\Sigma_D = \{\mathbf{V}_1, \ldots, \mathbf{V}_n\}$, where each $\mathbf{V}_i$ is the schema of an exported relational view (see Figure 6).

• A set of local *integrity constraints* Φ_D. We can distinguish different types of constraints, e.g., structural constraints (such as foreign key constraints) and semantic constraints.

• A *query capability specification* Π_D. For example, Π_D may be a set of access patterns [NL04a], prescribing the input/output constraints for each exported relation. More generally, Π_D may be given as a set of view *definitions* (possibly with access patterns) supported by the source D. If D provides full SQL capabilities, a reserved word can be used: $\Pi_D = \{\texttt{SQL}\}$.

To register the structural definition of the data source, a data-access handler or *wrapper* (shown both in Figure 6 and 10) must also be provided. The wrapper provides basic services for executing underlying queries and converting data to a common format for use by the registration and mediation service.

Semantic Data Registration. A semantic data registration registers the association between data objects in a database D and a target ontology O. Let $k = \mathsf{id}(D_k)$ and $j = \mathsf{id}(O_j)$ be the unique resource identifiers of D_k and O_j, respectively.

The *semantic data registration* of D_k to O_j is given by a set of constraints Ψ_{kj}, where each $\psi \in \Psi_{kj}$ is a constraint formula over $\Sigma_D \cup \Sigma_O$. For example, the semantic data registration formula $\psi =$

$$\forall x \forall y : j.\mathtt{D}(x) \wedge j.\mathtt{R}(x, y) \leftarrow \exists z : k.\mathtt{P}(x, y, z) \wedge k.\mathtt{Q}(y, z)$$

is a constraint saying that ontology O's concept `D` and its role `R` can be "populated" using certain tuples $\mathtt{P}(x, y, z)$ from D. When the semantic data registration constraint ψ and the above articulation φ are combined into $\psi \wedge \varphi$, we see that data objects from D_k can be linked to concepts like $i.\mathtt{C}(x)$ in the ontology O_i, despite the fact that D_k was registered to O_j and not to O_i. The reason is that an *indirect* link exists to O_i via the articulation axiom φ:[10]

$$D_k \overset{\psi}{\rightsquigarrow} O_j \overset{\varphi}{\longleftrightarrow} O_i$$

For a concrete example, assume the database D_k represents a geologic map, and the ontologies O_j and O_i represent the Canadian rock classification system [SQDO02] and the British one [GSR+99], respectively. We can register D_k to the Canadian system (encoded in OWL) using mapping rules corresponding to formulas like ψ above. This allows a semantic mediation system like OMI to provide concept-based query capabilities: On the left in Figure 8, the resulting query interface with fields for composition, texture, and fabric is shown.

An important application of articulation axioms like φ relating O_j and O_i is that they can be used to query and view the data from D_k through the conceptual view provided by O_i. In our geologic map example, we can query the geologic maps using the British rock classification view (a single, large hierarchy; see Figure 8, right), despite the fact that the geologic map database was originally registered to the Canadian system. Figure 8 shows the query interfaces and results of querying the same geologic map databases, using different ontology views. Note that the highlighted differences in the results shown in the figure could have different causes. For example, they might result from asking slightly different questions, or from a different relative completeness of one ontology over the other, or from the use of only partial mapping information in the ontology articulation.[11]

Datasets as Partial Models

This section further defines the steps involved in semantic data registration. In particular, we clarify the result of semantically registering a dataset as a *partial model* and motivate the need for additional, data-object *identification steps* (see Figure 11).

Registering Partial Models. A dataset D that is registered to an ontology O contributes to the extent of the federation relative to O. Registered datasets are not materialized according to the ontology; instead the registration mappings are used to access the underlying sources when needed, similar to the way queries against an integrated views are evaluated on demand (see Figure 6). A dataset D can be registered consistently to an ontology O if it can be interpreted as a *partial model* $\mathcal{I}$ of O, denoted $\mathcal{I} \models_p \Psi_O$, which implies $\mathcal{I} \cup \mathcal{I}' \models \Psi_O$ for some unknown $\mathcal{I}'$.

A partial model differs from a true model of the ontology in that some required information may be missing. We denote the interpretation induced by applying a semantic data registration Ψ_D of database D to an ontology O as $\mathcal{I}_D$. If the latter is a partial model $\mathcal{I}_D \models_p \Psi_O$, then the model $\mathcal{I}_D \cup \mathcal{I'}_D \models \Psi_O$ contains

[10] The actual links via φ are given by the valuations that make the formula true.

[11] In our OMI prototype, we only used a rough approximation of the data registration and articulation mappings. The mappings were based on (partial) syntactic matches and did not include a systematic review by a domain scientist.

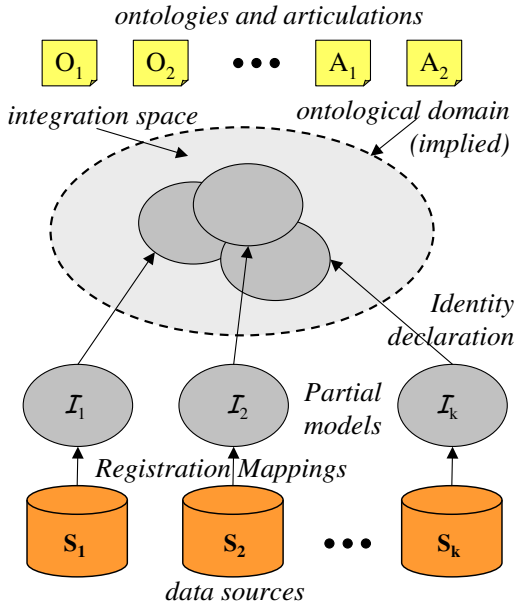

Fig. 11: Result of semantic data registration.

an unknown or hidden part $\mathcal{I}'_D$. As more sources are registered, more of $\mathcal{I}'_D$ may become known.

When an interpretation induced by a semantic data registration is not a partial model of an ontology, we say that the interpretation is *inconsistent*. An inconsistent interpretation often violates a datatype, cardinality, or disjoint constraint in the ontology. When possible, we wish to automatically verify that a semantic data registration is consistent, e.g., by ensuring that the dataset is satisfies the axioms of the ontology, or can be extended to do so.

Identification Declaration. Semantic data registration allows a dataset to be interpreted as a partial model of an ontology, but does not necessarily provide enough information to identify the same domain elements (individuals) of the integration space across multiple datasets, which is essential for data integration. The ability to identify equivalent data objects across different datasets is needed in practice because each dataset may only provide a portion of the information concerning a particular object. As shown in Figure 11, we consider an additional registration step called *identification declaration* that allows data providers to state how data objects should be identified across data sources.

Such an object identity can be defined in a number of ways. First, a semantic data registration can be augmented with *mapping tables*, which map individual data items to recognized individuals in **I**, i.e., individuals that are established instances within an ontology and come from an authoritative registry. For example, given an ontology of rock types, a mapping table can associate rock names in a geologic dataset with the unique rock types from the ontology. Second, external rules may be used for determining identity, similar to keys in a relational database.[12] For example, we may have a rule that ISBN or DOI codes uniquely identify publications, thus, registering to such a code uniquely identifies the data object. Finally, a data provider may give data-object correspondences between registered data sets. Thus, a data object is explicitly given as equivalent to another data object (although the specific identifiers of the objects may not be authoritative).

Data Integration via Semantic Registration

We identify four classes of semantic data registration expressiveness (in terms of data integration) as follows.

- *Concept-as-keyword registration.* We can consider metadata annotations using keywords from a controlled-vocabulary as (a weak form of) registration mappings. For example, we can assign a concept such as geologic-age to the dataset as a whole.[13] Such a mapping states that the dataset contains data objects, and those data objects refer to individual geological ages. However, we cannot consider or obtain each such separate (geologic-age) data object in the dataset. Clearly, such a registration can *not* be used for integration, however, it can be used for dataset discovery: We do not have access to the individual objects, so the best we can do is find the dataset that contains such objects.
- *Local data-object identification.* Local data-object identification is the typical result of a registration mapping, where local identifiers are used to identify logical data items within a dataset. In this case the identities of the individuals are local to the source, and thus, cannot be used to combine data objects from multiple sources.
- *Global data-object identification.* The result of globally identifying data objects is that it becomes possible for a mediator to recognize identical individuals in multiple datasets. The result of global data-object identification is the ability to perform object fusion [PAGM96] at the mediator.
- *Property identification.* If within a given dataset we relate two globally-identified data objects with an ontological relation in **R**, it becomes possible to join information across datasets (assuming at least one data object occurs in at least one other relation in

[12] Some description logics also support keys [LAHS03].

[13] e.g., by registering the dataset's resource-id with the concept, or registering all rows of the dataset with the concept

another source). This situation represents a stronger form of integration compared to simple object fusion, and is required for wide-scale data integration.

5 Scientific Workflows

In this section we return to the final integration challenge, synthesis (see Section 2), and illustrate how scientific workflows can be used to create new tools and applications from existing ones.

Scientific workflows are typically used as "data analysis pipelines" or for comparing observed and predicted data and can include a wide range of components, e.g., for querying databases, for data transformation and data mining steps, for execution of simulation codes on high performance computers, etc. Ideally, a scientist should be able to (1) plug-in almost any scientific data resource and computational service into a scientific workflow, (2) inspect and visualize data on-the-fly as it is computed, (3) make parameter changes when necessary and re-run only the affected "downstream" components, and (3) capture sufficient metadata in the final products. For each run of a scientific workflow, when considered as a computational experiment, the metadata produced should be comprehensive enough to help explain the results of the run and make the results reproducible by the scientist and others. Thus, a scientific workflow system becomes a scientific problem-solving environment, tuned to an increasingly distributed and service-oriented Grid infrastructure.

However, before this vision can become reality, a number of technical problems have to be solved. For example, current Grid software is still too complex to use for the average scientist, and fast changing versions and evolving standards require that these details be hidden from the user by the scientific workflow system. Web services provide a simple basis for loosely coupled, distributed systems, but core web-service standards such as WSDL [WSD03] only provide simple solutions to simple problems,[14] while harder problems such as web-service orchestration remain the subject of emerging or future standards. As part of an open source activity, members from various application-oriented research projects [GEO, SEE, SDM, BIR, ROA] are developing the KEPLER scientific workflow system [KEP, LAB+04], which aims at developing generic solutions to the process and application-integration challenges of scientific workflows.

[14] e.g. WSDL mainly provides an XML notation for function signatures, i.e., the types of inputs and outputs of web services

5.1 Scientific Workflows in Kepler

In Section 3.2 we have already presented a KEPLER workflow for mineral classification (Figure 5). KEPLER extends the underlying PTOLEMY II system [BCD+02] by introducing a number of new components called *actors*, e.g., for querying databases, for data transformations via XSLT, for executing local applications from a command line, web services via their WSDL interface, or remote jobs on the Grid, etc. In KEPLER, the user designs a scientific workflow by selecting appropriate actors from the actor library (or by dynamically "harvesting" new ones via the KEPLER web service harvester) and putting them on the design canvas, after which they can be "wired" to form the desired larger workflow. For workflow components that are not yet implemented (i.e., neither as a native actor nor as a web service or command-line tool), a special *design actor* can be used. Like regular actors, the design actor has *input ports* and *output ports* that provide the communication interface to other actors. The number, names, and data types of the ports of the design actor can be easily changed to reflect the intended use of the actor. When designing a workflow, the user connects actors via their ports to create the desired overall dataflow.[15] A unique feature of PTOLEMY II and thus of KEPLER is that the overall execution and component interaction semantics of a workflow is not buried inside the components themselves, but rather factored out into a separate component called a *director*. For example, the PN (Process Network) director used in the workflow in Figure 5 (green box) models a workflow as a process network [KM77, LP95] of concurrent processes that communicate through unidirectional channels. The SDF (Synchronous Data-Flow) director in Figure 12 is a special case of the PN director that can be used when all actors statically declare the number of tokens they consume and produce per invocation (called an actor "firing").[16] The SDF director uses this information to statically analyze the workflow, e.g., to detect deadlocks in the workflow, or to determine the required buffer size between connected actors.

5.2 Gravity Modeling Workflow

Figure 12 shows a gravity modeling *design workflow*. Unlike the mineral classification workflow discussed

[15] Control-flow elements such as branching and loops are also supported; see [BLL+04b, LAB+04].

[16] A token represents the unit of data flowing through a channel between actors. The workflow designer can choose to use, e.g., a single data value, a row from a database table, an XML element, or a whole table or file as a single token.

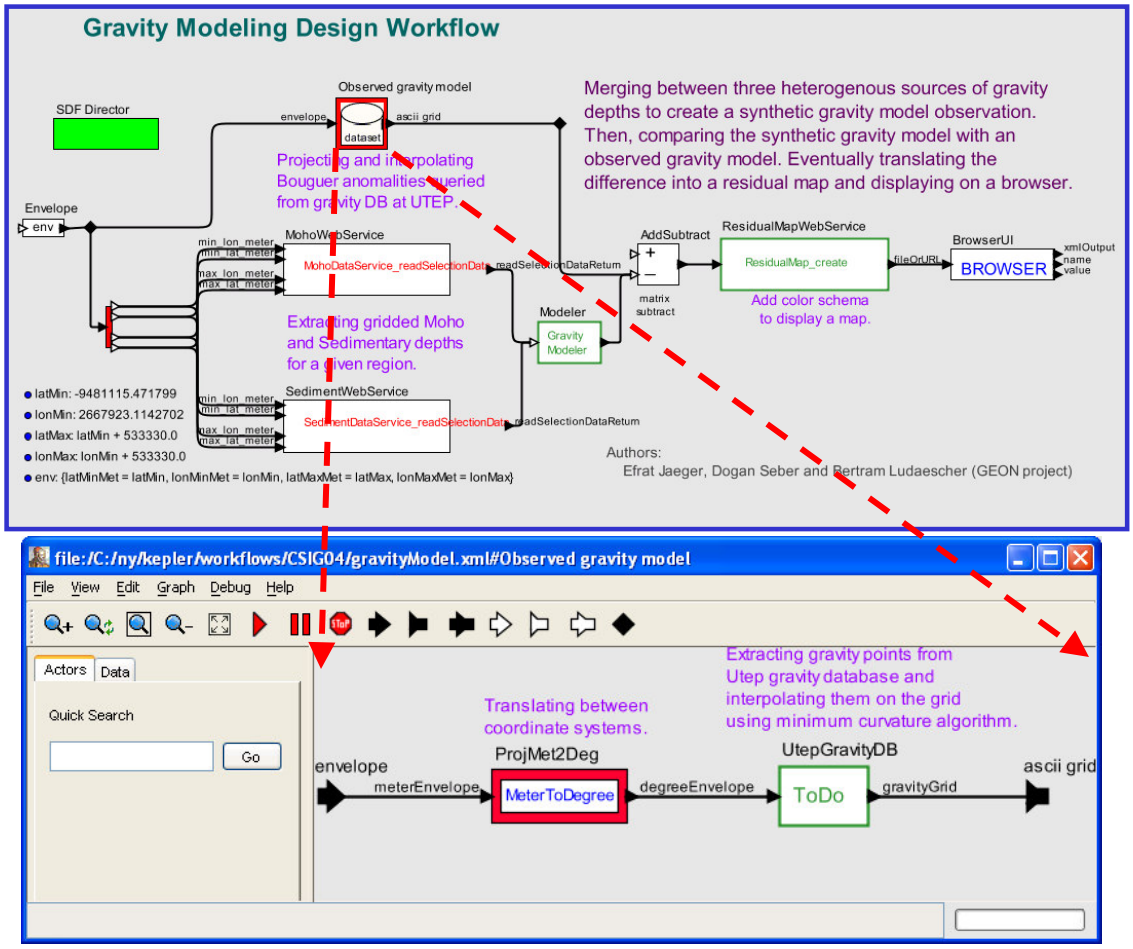

Fig. 12: Gravity Modeling Design Workflow: The main workflow (top) combines three different gravity sources into a single model: an observed model (expanded below) is compared to a synthetic model, which itself results from an integrated gravity model combining a Moho and a Sediment web service (top window, center). Outlined olive boxes are *design actors*. Unlike other actors, these are not (yet) implemented and ultimately need to be replaced by executable versions before the complete workflow can execute.

in Section 3.2, this workflow involves components that are not yet implemented. This feature allows the user of the KEPLER system to seamlessly go from a conceptual design workflow to an executable version by replacing design components with implemented ones as they become available. Another benefit of this feature is that executable subworkflows (i.e., ones that do not include design actors) can already be unit-tested and debugged early on while other parts of the workflow are still in their design phase.

The workflow depicted in Figure 12 takes a spatial envelope token (given via latitude and longitude parameters in the lower-left corner of the main window) and feeds it into two independent web services that extract gridded Moho and sedimentary depths for the enveloped region, respectively and feed them to a GRAVITYMODELER actor. This synthetic gravity model is then compared to the observed model and the result fed into a RESIDUALMAP web service actor for creating the result map. The latter is shown to the user via a standard browser. Note that a subworkflow is associated with the OBSERVEDGRAVITYMODEL actor, i.e., the latter is a composite actor. In Figure 12, the subworkflow of this composite actor is shown in the lower window. It involves two components, one to translate between coordinate system (implemented) and one to access values from the UTEP gravity database (designed). While this data access is not yet implemented, the signature of the design actor reveals that it needs to take an envelope token (in degrees) and produce from the database a gravity grid of observed data.

5.3 Semantic Workflow Extensions

An important aspect when developing scientific workflows are the structural and semantic types of actors and services they represent. Here, by *structural type* we mean the conventional data type associated with the input and output ports of an actor. KEPLER inherits from the PTOLEMY II system a very flexible structural type system in which simple and complex types are organized into an extensible, polymorphic type structure. For example, the ADDSUBTRACT actor in Figure 12, which takes the synthetic data output from the GRAVITYMODELER and compares it to the OBSERVEDGRAVITYMODEL output, is polymorphic and can operate on integers, floats, vectors, and matrices. In the case of the gravity workflow, after connecting ADDSUBTRACT with the corresponding upstream and downstream actors, the system can determine that a matrix operation is needed. If actor ports are connected that have incompatible types, the system will report a type error.

While a structural type check can guarantee that actor connections have compatible data types, they provide no means to check whether the connections are even potentially meaningful. For example, a connection between an actor outputting a velocity vector and one that takes as input a force vector may be structurally valid, but not semantically. For physical units, a unit type system has been added to PTOLEMY II to handle such situations [BLL+04b].

The idea behind *semantic types* in KEPLER is to further extend the type system to allow the user to associate a formal concept expression with a port. Thus semantic types in scientific workflows are related to the idea of semantic data registration (Section 4.3) which associates a data object having a structural type with a concept from an ontology. Our semantic type system[17] will detect, e.g., that a port of type geologic-age can not be connected to one of type texture, although both ports might have the same structural type string. Similarly, vector(force) and vector(velocity) can be detected as semantically incompatible, although their structural type vector(float) coincides.

There are several applications of a semantic type system in scientific workflows. As was the case for semantic mediation and data registration, some domain knowledge above the structural level of database schemas and conventional data types can be captured in this way. Thus semantic type information can be used for concept-based discovery of actors and services from actor libraries and service repositories, similar to the concept-based data queries in Sections 3.1 and 4.2. Based on the semantic type signature of an actor, the system can also search for compatible upstream actors that can generate the desired data or for downstream actors that consume the produced data. In this way, a workflow system with semantic types can support workflow design by only considering connections which are valid w.r.t. their semantic type.

Ontology-Driven Data Transformations. We now briefly illustrate a further application of semantic types, i.e., how they can be used to aid the construction of structural transformations in scientific workflows. Indeed a common situation in scientific workflows is that independently developed services are structurally incompatible (e.g., they use differently structured complex types, in particular different XML schemas), although they might be semantically compatible. For example, we might know that the ports P_s (source) and P_t (target) of two actors produce and consume, respectively, a matrix of gravity values. Thus, these ports should be connectible in terms of their semantic type. However, the actors may invoke independently designed web services that have different XML representations of gravity matrices, making the desired connection structurally invalid. The conventional approach to solve this structural heterogeneity is to manually create and insert a data transformation step that maps data from the XML schema used by P_s to data that conforms to the schema of P_t.

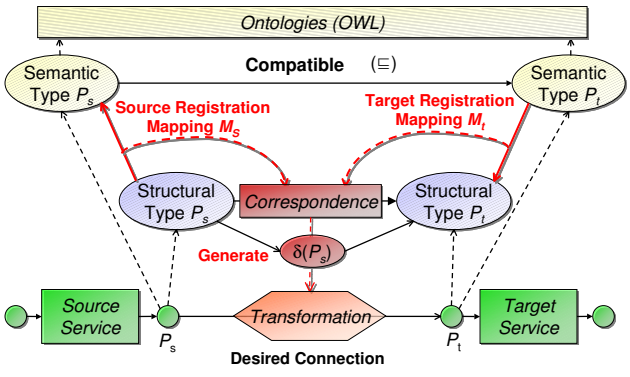

Fig. 13: Ontology-driven generation of data transformations [BL04].

The idea of ontology-driven data transformations [BL04] is to employ the information provided by the semantic type of a port to guide the generation of a data transformation from the structural type of the source port to that of the target port. Note that

[17] The semantic type system is currently under development and being added to KEPLER. More details can be found in [BL04].

the semantic type alone, when considered in isolation from the structural type cannot be used for this purpose. However, via a semantic data registration mapping (see Section 4.3), an association between structural type and semantic type is created that can be exploited for the generation of the desired transformation. Figure 13 gives an overview of the approach: The source port P_s and target port P_t are assumed to have incompatible structural types (denoted $P_s \not\preceq P_t$) but compatible semantic types (denoted $P_s \sqsubseteq P_t$). In other words, the semantic type of P_s is the same or a subtype of the one for P_t, but this is not the case for the structural types. The goal is thus to generate a data transformation δ that maps the structural type of P_s to one that is compatible with the structural type of P_t. We call a semantically valid connection *structurally feasible* if there exists such a data transformation δ with $\delta(P_s) \preceq P_t$. The core idea is that the information from the semantic registration mappings induces correspondences at the structural level that can help in the generation of the desired data transformation $\delta(P_s)$.

Correspondence Mappings. More precisely, let M_s be the source registration mapping that links between the structural type and the semantic type of P_s; similarly, let M_t denote the target registration mapping (see Figure 13). M_s and M_t can be seen as constraint formulas Ψ_s and Ψ_t as described in Section 4.3, relating data schemas to expressions of the ontology to which the data is registered. M_s can be given, e.g., as a set of rules of the form $\mathsf{q}_s \rightsquigarrow \mathsf{E}_s$, where q_s is a path expression or more general query that "marks" parts of interest in the data structure of P_s, and E_s is a concept expression over the ontology O to which P_s is semantically registered. The *correspondence mapping* $M_{st} := M_s \bowtie_O M_t$ is now given as the *semantic join* of M_s and M_t w.r.t. the ontology O: Given a rule $\mathsf{q}_s \rightsquigarrow \mathsf{E}_s \in M_s$ and a rule $\mathsf{q}_t \rightsquigarrow \mathsf{E}_t \in M_t$, the rule $\mathsf{q}_s \rightsquigarrow \mathsf{q}_t$ is in the correspondence mapping M_{st} if and only if the semantic join $\mathsf{E}_s \sqsubseteq_O \mathsf{E}_t$ holds, i.e., if the concept expression E_s yields a semantic subtype of E_t in the ontology O.

For example, assume a scientist wants to connect various ports of two actors that deal with geologic map information. Assume that the port P_s of the source actor produces XML tokens with the following structure

```
<rinfo>
    <age>...</age>
    <ccomp>...</ccomp>
    <text>...</text>
    <fab>...</fab>
</rinfo>
```

and that the port P_t of the target actor consumes XML tokens that are structured as follows:

```
<properties>
    <lithology>... </lithology>
    <geoage>...</geoage>
</properties>
```

Clearly $P_s \not\preceq P_t$, i.e., the structural types of P_s and P_t are incompatible. Now consider the source registration mapping $M_s =$

$$\begin{aligned} &\texttt{/rinfo/age} &&\rightsquigarrow O.\mathsf{geologic\_age} \\ &\texttt{/rinfo/ccomp} &&\rightsquigarrow O.\mathsf{lithology.composition} \\ &\texttt{/rinfo/text} &&\rightsquigarrow O.\mathsf{lithology.texture} \\ &\texttt{/rinfo/fab} &&\rightsquigarrow O.\mathsf{lithology.fabric} \end{aligned}$$

and the target registration mapping $M_t =$

$$\begin{aligned} &\texttt{/properties/lithology} &&\rightsquigarrow O.\mathsf{lithology} \\ &\texttt{/properties/geoage} &&\rightsquigarrow O.\mathsf{geologic\_age} \end{aligned}$$

where O is the ontology to which the structures are registered. Then the correspondence mapping M_{st} contains the rule

$$\texttt{/rinfo/age} \rightsquigarrow \texttt{/properties/geoage}$$

indicating how to transform the geologic age subelement of P_s to the one in P_t. Now assume further that $\mathsf{lithology.composition} \sqsubseteq_O \mathsf{lithology}$, i.e., that in the ontology, composition is considered a special kind of lithology information.[18] Then M_{st} also includes the correpondence rule

$$\texttt{/rinfo/ccomp} \rightsquigarrow \texttt{/properties/lithology}$$

In this simple example, the correspondence mapping M_{st} may not provide detailed enough information to determine a complete transformation from P_s to P_t.[19] In particular, we do not know in this example whether the various elements involved in the mappings are complex (i.e., contain nested elements). If so, further information would be required to automatically generate the transformation.

However, even if M_{st} does not provide enough information to automatically generate the data transformation $\delta : P_s \rightarrow P_t$ (see Figure 13), the obtained correspondences are still valuable. For example, a workflow engineer who needs to develop customized data-transformation actors for a scientific workflow can use these correspondences as a "semantic" starting point to define the needed transformation. Also, correspondences can be exploited by varous database schema-matching tools [RB01], and used to infer additional structural mappings.

[18] For a detailed description of the $\sqsubseteq_O$ relation see [BL04].

[19] Note that if `age`, `ccomp`, `lithology`, and `geoage` are "simple" elements, i.e., only contain PCDATA, the transformation can be generated from the correspondences alone.

6 Conclusions

To answer specific scientific questions, a scientist often repeatedly performs similar tasks and data management steps. For example, scientists select appropriate analysis methods to address the given question, search relevant data or create new data, determine whether existing data can be used for the desired analyses, pre-process and integrate data as needed, and so on. There are a number of significant data and tool integration challenges that need to be overcome to enable more of the increasingly data-driven "e-science", and to allow the scientist to spend less time on labor-intensive, error-prone manual data management.

In this paper we have given an overview of the different kinds of data integration and interoperability challenges in scientific data management. After reviewing the traditional mediator approach to data integration, we have presented a knowledge-based extension called *semantic mediation* that facilitates (i) the linking of hard-to-relate data sources by "going through" shared ontologies, and (ii) new types of concept-based queries against the data. Semantic mediation requires a "deeper modeling" of data which can be achieved by semantically registering existing data sources to one or more ontologies. We have presented some of the technical details of semantic data registration and illustrated semantic mediation using examples from the GEON project [GEO].

While semantic mediation addresses the problem of data integration, it does not provide a mechanism to integrate other applications and tools into data analysis "pipelines". For this kind of process integration problem, we have proposed the use of scientific workflow systems like Kepler, to provide an open and extensible problem-solving environment for designing and executing workflows. Kepler is built on top of the Ptolemy ii system and inherits from it many useful features, including an actor-oriented design methodology to create more reusable workflow components, an extensible type system, and an intuitive graphical user interface. Kepler specific extensions include database querying and data transformation actors, and actors for executing web services, command line tools, and remote jobs on the Grid [LAB+04]. A large remaining problem is the generation of data transformations that are often necessary to connect two independently designed actors or web services. To this end we have proposed an ontology-driven data transformation approach that exploits semantic data registration information to generate correspondence mappings, which in turn aid the generation of the desired data transformations.

References

[BCD+02] S. S. Bhattacharyya, E. Cheong, J. Davis II, M. Goel, C. Hylands, B. Kienhuis, E. A. Lee, J. Liu, X. Liu, L. Muliadi, S. Neuendorffer, J. Reekie, N. Smyth, J. Tsay, B. Vogel, W. Williams, Y. Xiong, and H. Zheng. Heterogeneous Concurrent Modeling and Design in Java. Technical Report Memorandum UCB/ERL M02/23, EECS, University of California, Berkeley, August 2002.

[BCM+03] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BFH03] F. Berman, G. Fox, and A. Hey, editors. *Grid Computing: Making the Global Infrastructure a Reality*. John Wiley & Sons, 2003.

[BIR] Biomedical Informatics Research Network Coordinating Center (BIRN-CC), University of California, San Diego. `http://nbirn.net/`.

[BL04] S. Bowers and B. Ludäscher. An ontology-driven framework for data transformation in scientific workflows. In *Proc. of the 1st Intl. Workshop on Data Integration in the Life Sciences (DILS)*, volume 2994 of *LNCS*, pp. 1–16, 2004.

[BLHL01] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.

[BLL04a] S. Bowers, K. Lin, and B. Ludäscher. On integrating scientific resources through semantic registration. In *Proceedings of the 16th International Conference on Scientific and Statistical Databases (SSDBM)*, 2004.

[BLL+04b] C. Brooks, E. A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, and H. Zheng. Heterogeneous Concurrent Modeling and Design in Java (Volumes 1-3). Technical report, Dept. of EECS, University of California, Berkeley, 2004. Technical Memoranda UCB/ERL M04/27, M04/16, M04/17.

[Che76] P. Chen. The Entity-Relationship Model: Towards a Unified View of Data. *ACM Transactions on Database Systems (TODS)*, 1(1):9–36, 1976.

[DOS04] M. C. Daconta, L. J. Obrst, and K. T. Smith. Chapter 8: Understanding Ontologies. In *The Semantic Web: A guide to the future of XML, Web Services and Knowledge Management*, pp. 181–238. Wiley, 2004.

[DT03] A. Deutsch and V. Tannen. MARS: A System for Publishing XML from Mixed and

Redundant Storage. In *Intl. Conference on Very Large Data Bases (VLDB)*, 2003.

[ESR98] ESRI. ESRI Shapefile Technical Description. `http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf`, 1998.

[FK99] I. Foster and C. Kesselman, editors. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1999.

[Fos03] I. Foster. The Grid: Computing without Bounds. *Scientific American*, April 2003.

[GEO] NSF/ITR: GEON: A Research Project to Create Cyberinfrastructure for the Geosciences. `www.geongrid.org`.

[GMPQ+97] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman, V. Vassalos, and J. Widom. The TSIMMIS Approach to Mediation: Data Models and Languages. *Journal of Intelligent Information Systems*, 8(2), 1997.

[Gru93] T. R. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5:199–220, 1993.

[GSR+99] M. R. Gillespie, M. Styles, S. Robertson, C. R. Hallsworth, R. W. O. Knox, C. D. R. Evans, A. A. M. Irving, J. W. Merritt, A. N. Morigi, and K. J. Northmore. BGS Rock Classification Scheme, Volumes 1–4. Technical report, 1999. `http://www.bgs.ac.uk/bgsrcs/home.html`.

[HAC+89] W. B. Harland, R. Armstrong, A. Cox, C. Lorraine, A. Smith, and D. Smith. *A Geologic Time Scale 1989*. Cambridge University Press, 1989.

[Hal01] A. Halevy. Answering Queries Using Views: A Survey. *VLDB Journal*, 10(4):270–294, 2001.

[Joh02] B. R. Johnson. Geologic Map Unit Classification, ver. 6.1, Draft, 2002. USGS.

[KEP] Kepler: A System for Scientific Workflows. `http://kepler-project.org`.

[KM77] G. Kahn and D. B. MacQueen. Coroutines and Networks of Parallel Processes. In B. Gilchrist, editor, *Proc. of the IFIP Congress 77*, pp. 993–998, 1977.

[Koc01] C. Koch. *Data Integration against Multiple Evolving Autonomous Schemata*. PhD thesis, Technische Universität Wien, 2001.

[LAB+04] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao. Scientific Workflow Management and the Kepler System. *Distributed and Parallel Systems*, 2004. (submitted for publication).

[LAHS03] C. Lutz, C. Areces, I. Horrocks, and U. Sattler. Keys, Nominals, and Concrete Domains. In *Proc. of the 18th Intl. Joint Conf. on Artificial Intelligence IJCAI*, 2003.

[Len02] M. Lenzerini. Data Integration: A Theoretical Perspective. Tutorial at PODS, 2002.

[Lev00] A. Levy. Logic-Based Techniques in Data Integration. In J. Minker, editor, *Logic Based Artificial Intelligence*, pp. 575–595. Kluwer, 2000.

[LGM01] B. Ludäscher, A. Gupta, and M. E. Martone. Model-Based Mediation with Domain Maps. In *17th Intl. Conf. on Data Engineering (ICDE)*, Heidelberg, Germany, 2001. IEEE Computer Society.

[LGM03] B. Ludäscher, A. Gupta, and M. E. Martone. A Model-Based Mediator System for Scientific Data Management. In Z. Lacroix and T. Critchlow, editors, *Bioinformatics: Managing Scientific Data*. Morgan Kaufmann, 2003.

[LHL+98] B. Ludäscher, R. Himmeröder, G. Lausen, W. May, and C. Schlepphorst. Managing Semistructured Data with FLORID: A Deductive Object-Oriented Perspective. *Information Systems*, 23(8):589–613, 1998.

[LL03] K. Lin and B. Ludäscher. A System for Semantic Integration of Geologic Maps via Ontologies. In *Semantic Web Technologies for Searching and Retrieving Scientific Data (SCISW)*, Sanibel Island, Florida, 2003.

[LLB+03] K. Lin, B. Ludäscher, B. Brodaric, D. Seber, C. Baru, and K. A. Sinha. Semantic Mediation Services in Geologic Data Integration: A Case Study from the GEON Grid. In *Geological Society of America (GSA) Annual Meeting*, volume 35(6), November 2003. .

[LLBB03] B. Ludäscher, K. Lin, B. Brodaric, and C. Baru. GEON: Toward a Cyberinfrastructure for the Geosciences – A Prototype for Geological Map Interoperability via Domain Ontologies. In *Workshop on Digital Mapping Techniques*. AASG and U.S. Geological Survey, June 2003. .

[LP95] E. A. Lee and T. Parks. Dataflow Process Networks. *Proceedings of the IEEE*, 83(5):773–799, May 1995. `http://citeseer.nj.nec.com/455847.html`.

[NCE] NCEAS. Ecological Metadata Language (EML). `http://knb.ecoinformatics.org/software/eml/`.

[NL04a] A. Nash and B. Ludäscher. Processing First-Order Queries with Limited Access Patterns. In *ACM Symposium on Principles of Database Systems (PODS)*, Paris, France, June 2004. .

[NL04b] A. Nash and B. Ludäscher. Processing Unions of Conjunctive Queries with Negation under Limited Access Patterns. In *9th Intl. Conf. on Extending Database Technology (EDBT)*, LNCS 2992, pp. 422–440, Heraklion, Crete, Greece, 2004.

[OWL03] OWL Web Ontology Language Reference, W3C Proposed Recommendation, December 2003. `www.w3.org/TR/owl-ref/`.

[PAGM96] Y. Papakonstantinou, S. Abiteboul, and H. Garcia-Molina. Object Fusion in Mediator Systems. In *Proc. of the 22nd Intl. Conf. on Very Large Data Bases (VLDB)*, pp. 413–424, 1996.

[RB01] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.

[ROA] ROADNet: Real-time Observatories, Applications and Data management Network. `roadnet.ucsd.edu`.

[SDM] Scientific Data Management Center (SDM). `http://sdm.lbl.gov/sdmcenter/`, see also `http://www.npaci.edu/online/v5.17/scidac.html`.

[SEE] NSF/ITR: Enabling the Science Environment for Ecological Knowledge (SEEK). `seek.ecoinformatics.org`.

[She98] A. Sheth. Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics. In M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman, editors, *Interoperating Geographic Information Systems*, pp. 5–30. Kluwer, 1998.

[SQDO02] L. Struik, M. Quat, P. Davenport, and A. Okulitch. A Preliminary Scheme for Multihierarchical Rock Classification for use with Thematic Computer-based Query Systems. Technical Report 2002-D10, Geological Survey of Canada, 2002. `http://www.nrcan.gc.ca/gsc/bookstore/free/cr_2002/D10.pdf`.

[Usc98] M. Uschold. Knowledge level modelling: concepts and terminology. *The Knowledge Engineering Review*, 13(1):5–29, 1998.

[VP00] V. Vassalos and Y. Papakonstantinou. Expressive Capabilities Description Languages and Query Rewriting Algorithms. *Journal of Logic Programming*, 43(1):75–122, 2000.

[VSSV02] U. Visser, H. Stuckenschmidt, G. Schuster, and T. Vogele. Ontologies for Geographic Information Processing. *Information Processing, Computers and Geosciences*, 28(1):103–117, 2002.

[Wie92] G. Wiederhold. Mediators in the Architecture of Future Information Systems. *IEEE Computer*, 25(3):38–49, 1992.

[WSD03] Web Services Description Language (WSDL) Version 1.2. `http://www.w3.org/TR/wsdl12`, June 2003.

[XML01] XML Schema, W3C Recommendation. `www.w3.org/TR/xmlschema-{1,2}`, May 2001.