Using Semantic Web Technologies for Exploratory OLAP: A Survey

Alberto Abelló, Oscar Romero, Torben Bach Pedersen, *Senior Member, IEEE*, Rafael Berlanga, Victoria Nebot, María José Aramburu, Alkis Simitsis

Abstract—This paper describes the convergence of some of the most influential technologies in the last few years, namely data warehousing (DW), On-Line Analytical Processing (OLAP), and the Semantic Web (SW). OLAP is used by enterprises to derive important business-critical knowledge from data inside the company. However, the most interesting OLAP queries can no longer be answered on internal data alone, external data must also be discovered (most often on the Web), acquired, integrated, and (analytically) queried, resulting in a new type of OLAP, *exploratory OLAP*. When using external data, an important issue is knowing the precise semantics of the data. Here, SW technologies come to the rescue, as they allow semantics (ranging from very simple to very complex) to be specified for web-available resources. SW technologies do not only support capturing the "passive" semantics, but also support active inference and reasoning on the data. The paper first presents a characterization of DW/OLAP environments, followed by an introduction to the relevant SW foundation concepts. Then, it describes the relationship of multidimensional (MD) models and SW technologies, including the relationship between MD models and SW formalisms. Next, the paper goes on to survey the use of SW technologies for data modeling and data provisioning, including semantic data annotation and semantic-aware extract, transform, and load (ETL) processes. Finally, all the findings are discussed and a number of directions for future research are outlined, including SW support for intelligent MD querying, using SW technologies for providing context to data warehouses, and scalability issues.

Index Terms—Business Intelligence, Data Warehousing, OLAP, ETL, Semantic Web, Reasoning

1 Introduction

Business Intelligence (BI) is aimed at gathering, transforming and summarizing available data from existing sources to generate analytical information suitable for decision-making tasks. The most widely used approach to BI has been the combination of Data Warehousing (DW), On-Line Analytical Processing (OLAP) technologies and the Multidimensional (MD) data model (see [1]).

DW/OLAP technologies have been successfully applied for analysis purposes, but always in a well-controlled "closed-world" scenario, where the set of data sources is rather static, and well structured data is periodically loaded in batch mode applying heavy cleansing transformations. However, the eruption of XML and other richer semi-structured formats like RDF has opened up much more heterogeneous and open scenarios than those of such traditional in-house DW applications.

In [2], Inmon outlines the opportunity and importance of using unstructured and semi-structured data (either textual or not) in the decision making process. Nowadays, Web 2.0 sites and Linked Open Data initiatives are becoming sources of huge amounts of valuable semi-structured data. Currently no one questions the need of adding all this information to the traditional corporate analysis processes. A significant amount of information and thus, knowledge, can be found in "unconventional" data sources like Web portals, social media, unstructured or less-structured data stores like product reviews, customer complaints, e-mails, and so on.

Enterprises have started to look into such rich information sources to increase their profits and improve their products and services. As an example, populating a business report that shows the effect of a product campaign in a specific time period may require combining information from historical, structured data like product sales and customer data, residing in a DW, with sentiments extracted from Big Data (e.g., tweets) relating to products promoted by the respective campaign (see [3], [4]).

Thus, companies want to *explore* all these new data opportunities and include them in their OLAP analyses, leading to a new type of OLAP: *Exploratory OLAP*.

The main difference of *Exploratory* OLAP from *Traditional* OLAP is naturally the issue of exploration: of new data sources, of new ways of structuring data, of new ways of putting data together, of new ways of querying data. Whereas *Traditional* OLAP is

- *A. Abelló and O. Romero are with Polytechnic Univ. of Catalonia-BarcelonaTech, Spain. E-mail: [aabello|oromero]@essi.upc.edu*
- *T. B. Pedersen is with Aalborg Univ., Denmark. E-mail: tbp@cs.aau.dk*
- *R. Berlanga, V. Nebot, M.-J. Aramburu are with Univ. Jaume I, Spain. E-mail: [berlanga|romerom|aramburu]@uji.es*
- *A. Simitsis is with HP Labs, USA. E-mail: alkis@hp.com*

performed in a "closed-world" scenario based only on internal data, an essential part of *Exploratory* OLAP is to discover, acquire, integrate, and analytically query *new external data*.

The Semantic Web (SW) has been conceived as a means to build semantic spaces over Web published contents so that Web information can be effectively retrieved and processed by both humans and machines for a great variety of tasks.

A recent paper [5] introduced the concept of *fusion cubes* to mean cubes that, based on a core of internal multidimensional data, gradually merge in (fuse with) external data, in order to support *self-service BI*. The paper provides a motivating example, which in our view captures the essence of exploratory OLAP well, and shows why SW technologies are needed in this scenario. The example concerns a group of concerned citizens (watchdogs) that want to monitor if the fishing catches being landed in the various EU countries respect the overall limits set up by the EU marine protection schemes and how they are related to marine protection areas. The watchdogs want to analyze the data by Time, Location, and Species, where each of these three dimensions should be organized into a hierarchy of levels, e.g., Day-Week-Month-Year, Port-Province-Country-Region, and Subspecies-Species-Family. To do this, they must integrate statistical catch data (in a flat tabular format) with geographical data about marine protection areas (from public database, in SW format), fish population data (from various research databases, in a multitude of formats ranging from comma separated files to SW data), and finally with ontology data describing geo and species hierarchies (in SW formats).

Reasoning capabilities are needed to perform the complex integration and resolve conflicts, e.g., contradicting catch data or species classifications. In our view, SW technologies are powerful enough to both model all these different types of data **and** provide the needed reasoning capabilities on top.

Several industrial OLAP tools already use data semantics to some extent. A notable example is the TARGIT Decision Suite (formerly TARGIT BI Suite) [6] which uses extended semantics to do so-called "meta-morphing". The TARGIT meta-morphing model extends the traditional multidimensional model with associations between measures and dimensions/levels, and records and learns from the users' behavior, e.g., what combinations of measures and dimensions/levels are used, which type of charts is used to display results, etc. This enables users to ask questions that are in some sense "in-complete", using the semantics and learned user preferences to fill in the gaps, and thus enabling easier and more intuitive interaction. However, the semantics are captured in a closed, internal format, and is applicable only to already known internal cube data.

To enable *Exploratory* OLAP, there is thus a great need for capturing semantics in an open and powerful way that can apply seamlessly across both internal and (newly discovered) external data. We believe that SW technologies are a good choice for this.

- To support the **discovery** of relevant data, it is essential that the meaning of the data is "declared" in an accurate, rich, and unambiguous way, so the right data can be found. The rich ontology languages found in SW are ideal for this.
- To support the data **acquisition**, external sources must be queried in a precise, yet efficient, way, to avoid having to download complete large data sets, which are perhaps never used again, and where most of the content is irrelevant to the particular query. SW query languages and technologies, such as SPARQL and SPARQL endpoints serve this purpose very well.
- To support the data **integration**, facilities must be provided to resolve conflicts in the data, to combine data from many different formats and sources, and to structure data in a multidimensional format. Again, SW technologies such as reasoning provide a powerful foundation for this.
- To support analytical **querying**, measure data must be aggregated along the OLAP dimensions. Both measures and dimensions (hierarchies and levels) can now be based on external data, and it is thus very important to capture the exact semantics of the multidimensional data and its lineage, i.e., providing not just a result, but also a precise specification of its meaning and where it came from. Again, SW technologies have the rich modeling constructs to support this.

Thus, *the goal of this paper is to survey how SW technologies can aid in data discovery, acquisition, integration, and analytical querying of external data, and thus serve as a foundation for Exploratory OLAP*.

We note that a solid foundation for *Exploratory* OLAP is also ideal for the wider scenario of "Exploratory BI", where deep analytics and data mining are performed on the exploratory OLAP cubes. However, considering the full range of *Exploratory* BI is beyond the scope of this paper.

As DW mainly involves the integration of disparate information sources, semantic issues are highly important for effectively discovering and merging data. These semantic issues are similar to those faced in the SW. As a consequence, SW technologies have recently been applied to some DW tasks such as Extract, Transform, and Load (ETL) processes, MD design and validation, and so on. Although they are usually limited to *Traditional* OLAP scenarios (see for example [7]), we will show that SW technologies can also be useful in highly heterogeneous and open scenarios.

The main purpose of this paper is twofold: a) to survey and categorize how SW technologies have been applied to solve the new requirements of *Exploratory* OLAP systems, and analyze the associated

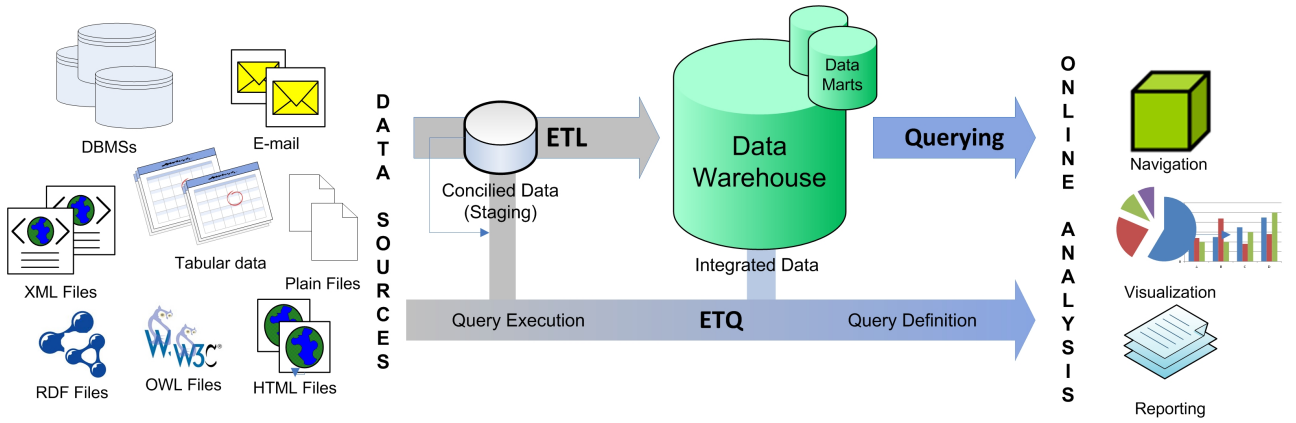

Fig. 1. DW/OLAP Elements and Data Flows

feasibility and benefits; and b) to use the analysis to identify future challenges where the potential to use SW technologies for *Exploratory* OLAP is high, but current technologies are lacking, i.e., a "position paper" approach.

To classify current approaches for the survey, we define five separate categorization criteria (i.e., *Materialization*, *Transformations*, *Freshness*, *Structuredness*, and *Extensibility*) and show that along all five dimensions there are challenges to overcome. It is important to notice that the first three criteria are related to data provisioning, while the latter two are more related to the data sources and the schema design. Thus, we will below survey and analyze how SW technologies are applied by current work to solve both data schema design and data provisioning requirements.

Moreover, the feasibility of applying SW technologies needs to be analyzed because, adding semantics generally increases system complexity. In order to analyze the classical trade-off between expressiveness and complexity, in the context of each particular work, we have used three more criteria specifically related to SW (namely *Reasoning*, *Computation*, and *Expressiveness*).

We believe that these eight dimensions allow to (1) cover the most relevant aspects of the usage of SW technologies towards *Exploratory OLAP*, and (2) separate different issues to facilitate the analysis and solution. Aspects that we find less relevant for this purpose, and thus do not consider, include the type of data storage (relational, NoSQL, etc.), any distribution or parallelism in data storage or computations, the specific types of analysis performed (OLAP only, or also specific types of data mining, etc.), and the specific technologies and systems used.

Our main conclusion from the survey part is that SW technologies are a promising way to approach the involved semantic integration issues. Although contributions and problems in data provisioning are more relevant than for modeling, there are less approaches in the former, the reason probably being not only the difficulty, but also that the relationship between data provisioning and SW is not mature enough. For the expressiveness/complexity tradeoff, our main conclusion is that researchers tend to use ad hoc algorithms on top of more mature standard services and maintain a medium level of expressiveness with easier computation at the expense of incomplete (but sound) inferences. *In the position part of the paper, we list challenges related to schema design, data provisioning, as well as semantic and computational issues*. The main contributions of this paper can be summarized as:

1) Propose a set of five novel criteria to categorize DW/OLAP systems,
2) Analyze how these criteria affect the need for semantics and the feasibility of the design and data provisioning processes,
3) Analyze how semantic-aware reasoning techniques can aid,
4) Survey and categorize existing DW/OLAP work according to the five criteria and the three reasoning criteria, and
5) Identify research trends in this area.

The paper is organized as follows. Section 2 introduces the basic concepts on DW and SW, and describes the categories used in the comparison of the different papers. Section 3 introduces the survey part of the paper by presenting a summarized comparison of the different approaches reviewed. The following two sections present the details of the survey: Section 4 surveys traditional and new features of data schema design and Section 5 does the same for data provisioning processes. Section 6 provides the position part of the paper through a global discussion of the main issues of the paper and potential research lines to address them. Finally, conclusions and future research directions are provided in the last section.

2 Methodology

Nowadays, a new trend of OLAP work has emerged, which applies SW technologies to mainly address data integration issues and the automation of data processing. The purpose of this paper is to categorize the

main requirements of these new OLAP approaches, as well as to show how SW technologies can help to fulfill the new requirements.

As there are many papers proposing a large variety of system features, in this section we present a methodology that guides this survey and produces a clear picture of this intricate area.

We first present the characteristics of *Traditional* OLAP use cases to frame the area of interest of our survey. Then, five criteria related to the different relevant aspects of DW/OLAP systems are defined. By means of these criteria, in the rest of the paper, current approaches are categorized. Furthermore, the five criteria define a space that allows us to locate *Exploratory* OLAP use cases and to distinguish them from *Traditional* OLAP use cases. In addition, we use another three criteria related to expressiveness, reasoning and complexity, to characterize existing work with regard to SW technologies.

2.1 The structure of OLAP systems

OLAP technology is aimed at gathering, transforming and summarizing available data from existing sources to generate analytical information suitable for decision-making tasks. Traditionally, OLAP has been associated with data warehouses (DW), following the three layered structure shown in Fig. 1, namely:

- the *data sources layer*, which consists of all the potential data of any nature (e.g., relational, object-oriented, semi-structured, and textual) that can help to fulfill the analysis goals,
- the *integration layer*, which transforms and cleanses the data gathered from the sources, as well as stores them in an appropriate format for the subsequent analysis (i.e., the DW), and
- the *analysis layer*, which contains a number of tools for extracting information and knowledge from the integrated data and presenting it to the analysts (i.e., OLAP cubes, charts, reports, etc).

As it is clear from this description, the integration model of *Traditional* OLAP systems (DW/OLAP) is based on a global schema (i.e., the DW schema), which is seen as a view over the underlying data source schemas (which is usually known as Global as View, or GaV for short). In this integration model, query answering is simple. The external data sources are (implicitly) assumed to be known in advance as are the user needs guiding the design of the global schema. This works well when the sources and requirements are indeed known in advance, but encounters problems when this does not occur. For those cases, more flexible integration models are needed. In particular, the integration of external data schemas in terms of a global schema (often in the form of a global domain ontology) has been studied (see [8]). From the global schema, local schemas can be derived; i.e., the local schemas are seen as (more specialized) views of the unified general global schema. The resulting integration model (usually known as Local as View, or LaV for short) is thus highly extensible, at the expense of considerably more complicated query answering. Therefore, in this integration model the reasoning power of SW technologies is especially needed.

DW/OLAP systems use a special data model, the multidimensional data model (MD), for the integration layer. Here, factual data gathered from the data sources layer must be expressed in terms of numerical measures and categorical hierarchical dimensions. The semantics of this model consists of representing any interesting observation of the domain (i.e., measures) in its context (i.e., dimensions). The typical processes in charge of transforming data from the data sources layer to the integration layer are called ETL processes. In some cases (e.g., to enable fast loading and querying at the expense of delivering only partly cleansed/transformed data at first), the order of the steps are switched, or interleaved, leading to Extract, Load, Transform (ELT) and Extract, Transform, Load, Transform (ETLT), where the transformations are (partially) delayed to provide fresh but less refined data. Taking this to the extreme, we have Extract, Transform, Query (ETQ) by delaying transformations to the last minute and serving data directly to the user on demand. Thus, given a MD query derived from a particular analysis goal, an ETQ process directly extracts the required information from the data sources, and transforms it to fit into the OLAP results.

ETQ processes are becoming essential for performing analyses that involve external data published in the Web, and therefore they usually deal with semi-structured, streamed and dynamic data sources (e.g., [9], [10]). Fig. 1 shows how ETQ processes can interact with the DW/OLAP data flow. Thus, an ETQ process can take "fresh" data from the ETL staging area, blend it with both external and DW integrated data (e.g., dimension hierarchies), and eventually deliver the results to the analytical tools. Notice that ETQ processes can also live apart from traditional DW/OLAP, avoiding thus the need of loading the integrated data into a DW. It can be noticed that *Exploratory* OLAP systems are tightly related to ETQ processes.

SW technologies can help in all DW/OLAP layers in order to support *Semantic-aware* and *Exploratory* OLAP systems. In the *data sources layer*, they can aid in capturing the precise semantics of the data sources. In the *integration layer*, they can be used to specify the transformations and capture the data lineage. In the *analysis layer*, they can help specifying the semantics of the presented information and reasoning about it. Finally, SW technologies can serve as a proper basis for defining ETQ processes, since most external data is now being published as linked data (see [10], [11]).

As far as we know, the amount of approaches that use SW technologies in the first two layers is large, whereas there are very few proposals that apply them

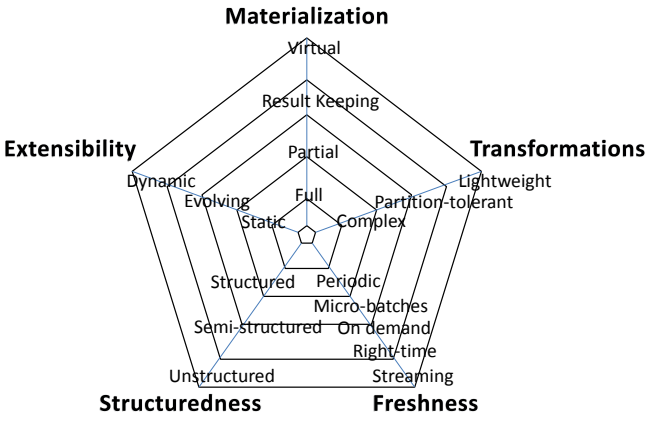

Fig. 2. OLAP Systems Categorization Criteria

to the third layer and ETQ processes. For this reason, our primary focus is to investigate the main issues of the first two components. More specifically, in this survey we analyze how SW technologies can aid in the integration and provisioning of data, and in the MD schema design of OLAP systems.

2.2 OLAP use cases categorization criteria

In this section we propose a series of criteria that aims to capture the main aspects of emerging DW/OLAP systems. The criteria aim to reflect the main components of a traditional DW/OLAP system, and how these components are evolving to cover the new requirements posed by the new scenarios (e.g., social networks, linked data and big data). The resulting categorization schema aims to identify the commonalities and differences of emerging approaches in terms of the changes they propose with respect to the traditional components of DW/OLAP systems, and is the result of long discussions and analysis of some previous work around the new scenarios for OLAP (e.g., fusion cubes in [5]).

The proposed categorization of DW/OLAP systems relies on five criteria. These criteria are generally orthogonal, but for certain types of DW/OLAP systems there will be correlations between them, as discussed below. The criteria are of a functionality-only nature (i.e., independent of the underlying technologies used to provide them). The first three criteria are related to data provisioning, while the latter two are closer to the data sources and the data schema design. The criteria are shown in Fig. 2. Generally, the center of the figure represents *Traditional* use cases, whereas the outer rim are use cases that are *Exploratory* or otherwise somehow harder, in terms of the required reasoning computation. SW technologies can provide large benefits even in the central area, where *Semantics-aware Traditional* systems/use cases are located, but in the complex cases found at the outer rim, where *Exploratory* requirements appear, having sufficient semantics and reasoning power is essential.

2.2.1 Materialization

Starting from the top of the figure, we firstly find *Materialization*. This criterion concerns the level of materialization of the integrated data. In *Traditional* DWs, all the integrated data is fully materialized (i.e., `Full`) often including a so-called data staging area for performing transformations and cleansing. At the other extreme, `Virtual` DWs extract data from sources at query time, integrate them on the fly, return the result to the user, and then throw away the integrated data. Notice that the ETQ processes described in the previous section fall in this category. A compromise, where some data is materialized, while other data, typically data with many changes, are extracted at query time, is sometimes used (see [12]). Closer to the `Virtual` DW, the `Result Keeping` approach first extracts data on-demand from sources and computes the result on the fly (e.g., for displaying in a dashboard), but then stores/keeps the results to allow repeated requests for the same result to be delivered quickly (see [9]). Complex ETL flows may actually have subparts each residing in different categories (i.e., `Partial`). For example, it is common to have an "on-line" flow that performs fast, but less thorough, on the fly integration in main memory for immediate use, while a parallel "off-line" flow performs more thorough integration for historical use and stores all data persistently (see [13]). Here, SW technologies can be used to describe the data and the results, as well as the steps inbetween.

2.2.2 Transformations

Proceeding clockwise, the next one is *Transformations*. This criterion concerns the level of transformations applied to the source data during the integration process. In *Traditional* DWs, it is common to apply many `Complex` and significant transformations, e.g., creating versions of data (i.e., Slowly Changing Dimensions -SCD- in [14]), significant cleansing, computing holistic aggregates, etc. At the other end of the spectrum, some use cases demand only `Lightweight` transformations that can be done quickly on the fly (even for streaming data), e.g., moving averages, simple and approximate aggregations, renaming/removing columns, etc. As mentioned above, such light data quality improvement and integration are sometimes complemented with a parallel flow performing complex transformations for later use. As a middle category, some systems apply transformations that are non-simplistic, but `Partition-tolerant` and thus parallelizable, such as categorizing values, etc. SW technologies provide a powerful framework for describing the transformations and for managing the lineage of results through these.

2.2.3 Freshness

The next criterion is *Freshness*, which concerns how often the data integration process is performed (i.e.,

how often the DW is refreshed). *Traditional* DWs were refreshed Periodically (e.g., daily, in batch mode). A variation of this is Micro-batches where the refreshment is run often (e.g., every 15 or 30 minutes), on the smaller batch of data accumulated in that period. Other DWs (e.g., the Virtual DWs mentioned above) refresh the data On demand, when requested by users. More recently, there has been a trend to refresh the DW even more frequently (e.g., with propagation delays of at most a minute or so). This is often referred to as near real time (or Right time) DWs. Finally, we can have Streaming data at very high rates like thousands of items per second, and handled more or less immediately: the so-called data stream approach. Thus, this criterion is somewhat related to *Materialization*. For Full *Materialization*, all levels of *Freshness* make sense and the same applies specifically for the specific data covered by Partial *Materialization*. The categories Result Keeping and Virtual are tied to On demand *Freshness*. This criterion can be heavily affected by the usage of SW technologies (e.g., for formally stating the freshness and the associated criteria and policies).

2.2.4 Structuredness

The next criterion is *Structuredness* which concerns which types of data are found in the data sources or, more specifically, how Structured the least structured type of source data is. In *Traditional* OLAP cases, all sources consist of structured data, typically relational tables or in a few cases structured spreadsheets. More recently, Semi-structured data sources such as XML, RDF and OWL have become more common. Lately, Unstructured data such as text files, HTML files, and other like e-mails or twits (part of the "big data" movement) have become important sources. SW technologies can be used for all three kinds, but their power is especially necesary to manage the more complex needs of semi- and unstructured data.

2.2.5 Extensibility

The next and last criterion is *Extensibility*. This criterion concerns how Dynamic the set of data sources can be, i.e., how easily new data sources could be brought into the system. In *Traditional* DW/OLAP, the same (mostly internal) *Static* data sources are used over and over, and new sources are only brought in at new major DW releases (i.e., at most a few times per year). Recently, there has been a trend to include new data sources, often from external data suppliers, into an existing DW more often in order to answer new questions, making the source set Evolving. In this context, an Evolving system is able to adapt their MD schemas to evolving data sources as well as user requirements. These systems have been widely studied in the literature (see reviews from [15], [16], [17]), and they can be considered within the traditional DW/OLAP boundaries.

Finally, some cases can be very *Exploratory*, looking for new sources most of the time, in order to answer specific, but constantly changing questions, making the source set completely Dynamic. This criterion should be seen as how easy it is to evolve the schema, more than how often it is actually done, which does not depend on the technologies used, but only on the business needs. While it makes good sense to use SW technologies to describe even rather static DWs, it becomes essential to have a powerful semantics and reasoning framework for the Dynamic case.

2.2.6 Discussion

DW/OLAP systems take many forms, but we see two separate "rings" in the five-dimensional space emerging, with many possible stands in between. The inner ring (core) is *Traditional* OLAP, which is well understood and aims at answering a rather static and well-defined set of questions mostly on structured data. The outer ring is what we will call *Exploratory* OLAP which aims at answering new and constantly changing questions on a much wider range of data. We note that a number of similar terms have been suggested for the broader case of BI systems (covering not only DW/OLAP, but also analytics/data mining), including "live BI" in [18], "on-demand BI" in [19], "ad-hoc BI" in [20], "open BI" in [21], "situational BI" in [22], or lately "fusion cubes" in [5]. However, given our focus on (multidimensional) data modeling and data acquisition, we think that the term *Exploratory* OLAP is more precise and better captures its essence. The most important criterion for distinguishing between them is *Extensibility*. *Traditional* OLAP cases have a Static set of data sources, while *Exploratory* OLAP often, or all the time, brings in new data sources, making them Evolving or Dynamic. For the *Structuredness* criterion, *Traditional* OLAP tends to use mostly Structured data, while *Exploratory* cases also use Semi-structured and Unstructured data sources (e.g., text, social media data, etc). For the *Materialization* criterion, *Traditional* OLAP typically uses a Materialized DW, including an intermediate materialized data staging area. In contrast, *Exploratory* OLAP will often use a Virtual approach where data is pulled from sources on-demand, although some level of materialization is possible (e.g., caching). Data staging areas are typically not used. For the *Transformations* criterion, *Traditional* OLAP will typically have Complex transformations such as maintaining SCDs and computing precise holistic aggregates, while *Exploratory* OLAP will only employ transformations that can be performed sufficiently fast (perhaps on streaming data) and in parallel (i.e., the transformations have to be Lightweight or at least Partition-tolerant). We note that even such transformations can in fact require heavy computations (e.g., machine learning computations, data/text analytics, User Defined Func-

tions -UDFs-, etc). Also, such transformations often blend several traditional stages all in one complex step (e.g., starting with business processes, over ELT-ing, analytics, to reporting and visualization). For the `Freshness` criterion, *Traditional* OLAP will typically employ `Periodic` updates, perhaps executed very frequently (i.e., `Micro-batches`), while *Exploratory* OLAP will typically extract some data either `On demand`, in `Right-time`, or from `Streaming`.

A given system is often not clearly of either type, but rather somewhere in the continuum between them, facilitating more or less *Exploratory* use cases. Also, both kind of cases will often co-exist in the same organization or even as a "hybrid case" in the same system, with more *Traditional* cases, or system parts, for the core day-to-day analytical tasks on internal data, and other more *Exploratory* cases/parts for ad hoc analyses.

To summarize, while SW technologies can provide significant benefits even for *Traditional* systems, they become indispensable to handle the complexities and dynamics found in *Exploratory* ones.

2.3 SW technologies for OLAP systems

SW technology is aimed at providing the necessary representation languages and tools to express semantic-based metadata. This focus on semantics is very useful for *Exploratory* OLAP systems, where the vast amount of unstructured or semi-structured sources demand new semantic-aware solutions that enable machine processable data integration.

SW technologies can aid the development of *Exploratory* OLAP systems in two aspects: on the one hand, ontologies serve the purpose of formally conceptualizing both the domain of interest and the business concepts. On the other hand, by means of semantic annotation, different data sources can be mapped to ontology concepts, resulting in a homogeneous conceptual space where we capture the meaning of the integrated elements.

Most ontology languages, such as the Web Ontology Language (OWL; the W3C recommendation), have strong foundations in logics and differ from other semantic-aware technologies in that they are machine processable and support reasoning. Thus, we can describe concepts and relationships but also infer implicit knowledge from that explicitly stated. Two main families of logic-based languages currently underlie most of the research done in this direction: Description Logics (DL) and Datalog-related logics (see [23] and [24], respectively). As discussed in [25], both paradigms can be used to establish ontologies, but from different points of view.

DL-based languages, such as OWL, assume a decentralized approach and information is stored separated from data. Thus, one talks about *terminology* and *instances asserted*. DL also follows the open-world assumption and, accordingly, a DL ontology can have many different interpretations. Furthermore, Datalog follows a centralized viewpoint, the closed-world assumption and the unique name assumption. A direct consequence is that DL ontologies are more difficult to model but they better deal with incomplete data (such as Web data), whereas Datalog ontologies are more intuitive for the database community but might not be that interesting for integration cases with missing or partial information. In our case, DL suits decision making processes where not having information about a fact does not necessarily mean it is not true; e.g., what-if analysis.

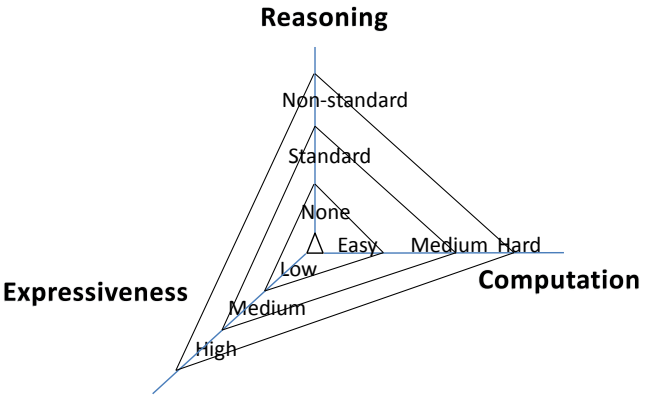

Fig. 3. SW Technologies Categorization Criteria

In this paper, we pay special attention to the language used to describe the ontology and semantic annotations, as the more expressive the semantics of this language, the more machine-processable the annotations, but also the more computationally expensive. In the next section, we describe three criteria that capture this trade-off and categorize state-of-the-art OLAP systems literature w.r.t. to their use of logic-based ontology languages.

2.4 SW technologies categorization criteria

Although logic-based languages are very appealing for their semantic-awareness and reasoning features, reasoning is computationally hard. For this reason, most research done in this direction is focused on balancing the language expressiveness and the reasoning services provided according to each scenario (see [23]). This trade-off is traditionally captured in terms of three criteria (represented in Fig. 3): *Reasoning* capabilities provided, language *Expressiveness*, and *Computation* complexity. Without loss of generality, in the remainder of this paper we focus on how research on OLAP makes use of logic-based ontology languages and the trade-off offered w.r.t. these criteria.

2.4.1 Reasoning

Starting clockwise from the top, the *Reasoning* criterion concerns the inference algorithms needed. We mainly talk about the use of `Standard` reasoning

services (such as subsumption), `non-Standard` inferences (such as schema matching, transitive closures, temporal reasoning [26], and so on) and no use of reasoning (i.e., `None`). We say a reasoning service is `Standard` if it is supported by most reasoners. The typical inferences provided by DL reasoners are concept satisfiability, subsumption and query answering (see [23]). Concept satisfiability and subsumption sit at the terminological level, whereas query answering also deals with instances. Relevantly, very few DL languages (e.g., DL-Lite in [27] and the OWL2 QL profile, based on DL-Lite) properly support query answering which means that, in practice, query answering is prohibitively costly for large data sets, such as those in OLAP scenarios. Thus, most DL languages are typically used at the terminological level. Concerning Datalog, since terminology and instances are not separated, its reasoning services are query-oriented and its most typical inference is query answering. Among `non-Standard` reasoning services we focus underline: concept matching, the least common subsumer, and computing the functional transitive closure. Discovering functional dependencies, by means of computing the functional transitive closure, would also be of great interest for the OLAP community because, as discussed in Section 4, discovering MD schemas is nowadays mainly based on functional dependencies. All these inferences sit at the terminological level. However, the matching and least common subsumer have been shown to be costly and, in some cases, matching turns out to be undecidable (see [23]). For computing functional dependencies at the terminological level we need to compute the functional transitive closure that is prohibitively costly in the general case. However, its feasibility has recently been shown in [28] for less expressive DLs. In case no reasoning is used (i.e., `None`), the ontology is primarily used as a dictionary or common terminology among peers.

2.4.2 Computation

The next criterion is *Computation*. For this axis we do not mean classic theoretical computational complexity, but instead the feasibility of computing certain reasoning tasks under certain assumptions (i.e., in a given scenario). An expensive inference (e.g., computing the transitive closure of all properties in an ontology) computed once may indeed be more feasible than a relatively less complex reasoning task (e.g., computing subsumption in OWL DL ontologies) conducted relatively often (e.g., over a very large ontology and triggered by a certain event in the application GUI). Thus, this axis refers to how feasible certain reasoning tasks are from a practical point of view. Accordingly, by `Easy` we mean feasible tasks, `Medium` means a lot of computer time is needed (but still feasible), and `Hard` means infeasible (either in practice for large data sets -which is the most typical scenario for OLAP systems- or theoretically proven).

2.4.3 Expressiveness

Finally, the *Expressiveness* criterion concerns to which extent we can describe the domain semantics by means of the terminological constructs at hand. For this axis, we distinguish between `Low` (mainly semantic annotations such as RDF triples or simplistic ontology-based descriptions such as concept taxonomies), `Medium` (ontology languages such as DL-Lite, OWL2 profiles and most Datalog languages, which are known to have a limited expressiveness but able to deal with reasoning in terms of data complexity, i.e., considering the instances to resolve the inference algorithms) and `High` (highly expressive ontology languages, such as OWL DL, still feasible for inferences at the terminological level, but not in terms of data complexity). We do not consider more expressive languages such as OWL Full because even their cheapest inferences are prohibitive for OLAP.

2.4.4 Discussion

Both DL and Datalog approaches tackle the same problem: modeling ontologies to overcome the (potential) lack of semantics, which is the core of our discussion. However, their inherent characteristics make them tackle this problem from different perspectives. The relationship between the latter two criteria for most DL and Datalog languages is nowadays clear for `Standard` reasoning services. However, many OLAP solutions are built on top of semantic-aware technologies and need `Non-standard` reasoning services. In these cases, the feasibility (in terms of the computation axis discussed above) of using reasoning services for certain ontology languages under certain assumptions (e.g., functional transitivity for multidimensional schemas) must be explored.

In the next sections, we analyze how semantic-aware solutions and specifically ontology languages can be used to overcome the difficulties discussed in Section 2.2 as we move away from the center along any of the five axes there presented (i.e., *Extensibility*, *Materialization*, *Transformations*, *Structuredness* and *Freshness*) with regard to the categorization (i.e., *Reasoning*, *Expressivity* and *Computation*) for SW technologies presented in this section.

3 Comparison

In previous sections, we have introduced the four main stages of a DW/OLAP system (discovery, acquisition, integration and querying) and later, we have introduced a set of criteria to categorize current approaches. As we explained before, our main focus is to investigate how SW technologies can aid throughout these stages.

In practice, current approaches are traversal to these four conceptual stages and thus, they cannot be classified according to them. As in classical software design approaches, current solutions either focus on

Criterion/Category	Traditional (·)	Medium (⊙)	Exploratory (⊙)
Structuredness	Structured	Semi-structured	Unstructured
Materialization	Full	Partially	Virtual
Transformation	Complex	Not only Relational algebra	Relational algebra
Freshness	Periodic	Some sources on-demand	On-demand
Extensibility	Hard to add sources	Sources easily added	Sources added automatically
Criterion/Category	Easy (☺)	Feasible (🕒)	Difficult (☹)
Reasoning	None	Standard	Non-standard
Expressiveness	RDF/Taxonomies	Datalog, OWL2 profiles, DL-Lite	OWL-DL
Computation	Less than Polynomial	Polynomial	Exponential

TABLE 1
Generalized categories

the MD schema design of OLAP systems (i.e., at the schema level), or on integration and provisioning of data (i.e., at the data/instances). For each of these two categories (schema vs. data), we have identified representative papers, and subsequently divided them into two subcategories (for a total of four categories of papers), depending on whether SW technologies are applied to satisfy the requirements of *Traditional* OLAP systems (here denoted *Semantic-aware* OLAP systems), or to support (to some extent) the new set of requirements of *Exploratory* systems. The papers were selected based on our experience, depending on how well they exemplify the categories.

As previously explained, being *Traditional* or *Exploratory* is not boolean, but a continuum with a blurry border. Also, inside each of the five criteria in Fig. 2, there is a continuum. Thus, to facilitate the visual comparison, we have simplified the descriptive scale of each criterion into only three generic categories, depending on how close they are to the mid point of the space (i.e., *Traditional* OLAP). Table 1 summarizes these fifteen resulting categories, together with the nine categories for the SW criteria in Fig. 3. Notice that the max (resp. min) of these three categories does not necessarily coincide with the max (resp. min) of the corresponding edge, because the purpose of the table is just to outline the differences in the approaches (i.e., the table shows the max -resp. min- found in the analyzed papers). These exactly coincide for *Structuredness* and *Materialization*; for *Transformation*, the typical `Lightweight` set of operations we found is `Relational Algebra` (so we reflected this in the table); in the case of *Freshness*, the more *Exploratory* papers we found considered `On-demand` refresh; finally, with regard to *Extensibility*, we mapped papers automatically adding sources to `Dynamic`. To decide the classification of each work in each category, we checked the content of the corresponding papers looking for matchings to the definitions in Section 2. If not enough information was provided to classify some work in a given category, we crossed out the corresponding cell.

Table 2 summarizes the findings in the analyzed work, the details of which will be provided in Sections 4 and 5. This table is aimed at evaluating existing work under the specific criteria, and allows us to visually analyze the correlation between being more *Exploratory*, using semantics, and the incurred cost.

For each of the relevant papers identified, we show its position in each of the five DW criteria, and also the value in the three SW ones. Horizontally, the table is divided into four parts corresponding to *semantic-aware MD design*, *multidimensional query definition*, *semantic-aware ETL processes*, and *ETQ processes*. When one paper deals with both issues, MD design and data provisioning, it appears twice in the table and is analyzed from both perspectives (which may result in apparently contradictory classifications, caused by the different viewpoints of the analysis).

Let us start from the upper part of Table 2 (i.e., semantic-aware MD design). As previously discussed (and further discussed in Section 4.1) the *Transformation* and *Freshness* OLAP criteria do not apply for these papers. These criteria are more related to data than to schema and have not been typically considered for MD design. Focusing on the other criteria, the common characteristic of all papers is that they completely materialize the DW. Four of them deal with semi-structured data, and among these, two facilitate to some extent their inclusion in the DW. The exception to this is Neumayr et al. [34], because they present an extension of the work (i.e., [35]) which adds the *Exploratory* part. As a general rule, when a work only deals with the schema, it allows `High` *Expressiveness*, but dealing with schemaless unstructured data requires to lower the *Expressiveness* to `Medium` in order to process the huge amount of data in a DW. In any case, the computation would consume a lot of resources, except if one restricts the *Expressiveness* and at the same time only deal with the schema of a well structured database.

The second part of the table corresponds to more *Exploratory* design systems (see Section 4.2). We can see that in this case, the common characteristic to all of them is that they are able to deal with unstructured data, as soon as ontological mappings are provided. Also common to all of them is that they to some extent facilitate the addition of new sources to the DW. However, most of them materialize extracted data, do it off-line and allow only light-weight transformations.

		Reference	Str.	Mat.	Tra.	Fre.	Ext.	Rea.	Exp.	Com.
Data Schema Design	Sem.-aware	Niemi et al. [29], [30]	⊙	·	×	×	·	🕒	☺	🕒
		Priebe et al. [31]	⊙	·	×	×	·	🕒	☺	🕒
		Bakhtouchi et al. [32]	·	·	×	×	·	🕒	☹	☹
		Prat and Akoka [33]	·	·	×	×	·	🕒	☹	☹
		Neumayr et al. [34], [35]	·	·	×	×	⊙	🕒	🕒	🕒
		Abelló et al. [36]	⊙	·	×	×	⊙	🕒	🕒	☹
		Romero and Abelló [37]	⊙	·	×	×	⊙	🕒	🕒	☹
	Exploratory	Kämpgen et al. [11]	⊙	·	⊙	·	⊙	☺	☺	🕒
		Romero et al. [38]	⊙	·	⊙	⊙	⊙	🕒	☹	☹
		Nebot et al. [39]	⊙	·	⊙	·	⊙	🕒	☹	☹
		Khouri et al. [40]	⊙	·	⊙	·	⊙	🕒	☹	☹
		Kämpgen et al. [41]	⊙	⊙	⊙	⊙	⊙	☺	☺	🕒
		Nebot and Berlanga [10]	⊙	⊙	⊙	⊙	⊙	🕒	☹	☹
Data Provisioning	Sem.-aw.	Niemi et al. [29], [30]	⊙	·	⊙	·	⊙	🕒	☺	🕒
		Skoutas and Simitsis [42], [43]	⊙	·	⊙	×	⊙	🕒	☹	☹
		Skoutas et al. [44]	⊙	·	⊙	×	⊙	🕒	☺	☺
		Romero et al. [38]	⊙	·	⊙	⊙	⊙	🕒	☹	☹
	Expl.	Nebot and Berlanga [10]	⊙	⊙	⊙	⊙	⊙	🕒	☹	☹
		Pedersen et al. [9]	⊙	⊙	⊙	⊙	⊙	☺	×	☺
		Kämpgen et al. [41]	⊙	⊙	⊙	⊙	⊙	☺	☺	🕒

TABLE 2
Summarized comparison

Kämpgen et al. [41] and Nebot and Berlanga [10] seem to move in a more *Exploratory* space, though. Not surprisingly, `High` *Expressiveness* results in the need of heavy computations. As the two last approaches in this part show, the only way to avoid heavy computations is to avoid reasoning.

The third part of the table corresponds to semantic-aware ETL (see Section 5.1). We can see that the common characteristics in this case are *Extensibility*, *Materialization*, and *Structuredness*. The work in this area tries to automate to some extent the generation of data flows into a DW. Then, they allow more or less unstructured data and simple transformations. Relevantly, since some of the approaches work at the conceptual level, they do not pay attention to more physical characteristics like *Freshness*. Due to the inherent complexity of the problem, some drastically limit their *Expressiveness* in order to be feasible. The rest of them result in high *Computation* needs.

Finally, the last rows in the table summarize the work related to ETQ (see Section 5.2). They all avoid the complete materialization of data and thus improve their *Freshness*. Some facilitate *Extensibility* more than others, but all allow to deal with semi and/or unstructured data. Given the cost of querying this kind of data, they avoid reasoning or keep it to a minimum (e.g., Pedersen et al. [9] does not even use RDF semantic constructs). Also they do not deal with heavy-weight transformations, but keep this as simple as possible. Despite this fact, the needed computing power to run them is not low, in general.

Retaking our watchdogs example in the introduction, a semantic aware data design could be used if we can (beforehand) define the mappings from all the different data sources to the ontology showing the watchdogs vocabulary (i.e., we must know in advance that we will use statistical catch data, geographical information about marine protection areas, etc). Once this technical work is done by someone in the watchdogs group, the others could use the ontological knowledge to navigate and analyze all these data, which would most probably have been replicated onto their server for the sake of performance. Oppositely, if such planning cannot be done beforehand and the watchdogs need to discover the different data sources on the fly (or there is no such expert in the group able to define the mappings), then there should be a public ontology where the publisher of catching information and marine protection areas map their contents. Given that, the watchdogs could use exploratory techniques to navigate that public ontology and find what is worth to be analyzed among the available sources.

With regard to data provisioning, given the same premises as in the data design, semantic aware techniques can help to design the extraction processes, which may include complex cleaning algorithms, because the data is extracted beforehand and locally stored. This would need the intervention of some expert inside the group. On the contrary, if no complex cleaning is necessary and we do not want to replicate the data in our server, the watchdogs could use an exploratory approach to issue simple queries directly to the data providers. The ontological knowledge should facilitate the integration of those data on the fly. We do not preclude the expert from participating,

but just consider that his/her intervention can and should be minimized.

4 Data Schema Design

MD design is a well-known paradigm in the area of DW and databases in general, always related to OLAP tools. It was popularized by Ralph Kimball at the logical level in [14].

Multidimensionality is based on the fact-dimension dichotomy. This paradigm aims at analyzing the fact (or subject of analysis) instances, from different analysis dimensions (i.e., points of view). Several measures (i.e., metrics) are available for each fact instance in order to gain insight. Furthermore, the MD model also provides foundations to study/analyze the available measures at various aggregation levels determined by the hierarchical structure of the dimensions. Indeed, aggregation is one of the main characteristics of the MD model, setting foundations for the well-known roll-up and drill-down operators.

Conceptually, it entails that the fact is related by means of to-one relationships (i.e., functional dependencies) to the dimensions, which in turn identify the fact. Thus, a fact has no atomic identifier but instead a compound one composed by the set of dimension values univocally identifying the factual data (from now on we will refer to these identifiers as MD identifiers). Finally, dimension hierarchies are composed of aggregation relationships between the dimension levels. Discovering these kinds of relationships is crucial in the design of the OLAP cubes and in turn of the DW.

In next subsections, we first review the *Traditional* OLAP work using SW technologies in the design and then, we discuss more *Exploratory* approaches, where no clearly delimited design phase exist.

4.1 Semantic-aware Multidimensional Design

The main features of *Traditional* OLAP systems directly impact on how MD design has been tackled for such systems. Since these systems largely materialize the integration layer, designers arrange the integrated data in an MD fashion ready to be exploited by non-expert users. `Complex` and `Periodic` ETL processes guarantee a quality threshold for the integrated data, which comes from `Static` and `Structured` sources (typically relational databases). In some cases, this data is completed with data coming from external or non-structured sources.

Thus, *Traditional* MD design first focuses on identifying the needed subset of source data to answer the end-user analytical needs and then arranges them. Since sources are mainly relational and static, the process can be automated up to some extent by inferring a mapping between the two schemas. In these scenarios, MD schemas are identified by exploring the data sources in order to discover MD patterns fulfilling the MD integrity constraints. Although no standard is available, much work has been devoted to identify these constraints (e.g., [45]), which can be summarized as a formal definition of the MD space and the notion of well-formed hierarchies, in order to preserve a correct aggregation of data.

To automate MD design, classical approaches (e.g., [14]) focus on the organization of data and assume relational (or homogeneous) and well-structured sources and therefore, they are hardly effective (or feasible) in heterogeneous scenarios with disparate sources. Indeed, the more automatable they are, the more tied to a specific formalism or language (typically relational sources). Consequently, they do not tackle the integration of different data models.

At this point, some work proposed semantic-aware approaches to integrate external data. Basically, these first approaches follow the same principles but starting from an integrated view of the sources in XML, RDF, or ontologies. SW technologies are a promising foundation for integrating heterogeneous data and most work further exploring this direction can be classified either as those focusing on Web data (or similar scenarios), where the presence of the SW technologies is granted, and those using SW technologies to tackle integration in any scenario.

Among the first ones, one may find the work discovering MD schemas from XML (a review and deep discussion of DW approaches for XML and Web data can be found in [46]). In the general case, however, these approaches can be clearly improved by using more expressive SW formalisms that facilitate integration by incorporating a reference semantic layer whereto every domain concept and relationship can be mapped.

The suitability of DL for data modeling was claimed back in 1998 (see [47]) and by then the first work acknowledging the benefits of modeling the DW by using such formalisms appeared in the DWQ project. This line of work highlights the need of capturing functional dependencies and aggregation relationships to model the DW. A few papers focusing on Datalog to model the DW also appeared at that time. Although Datalog has been overlooked for modeling DWs until very recently, it was, indeed, the first logic-based approach proposed for DW modeling in [48]. This work tackled the very same problems just discussed for DL: how to deal with functional dependencies and summarizability. A proper Datalog extension was presented to do that.

However, the state of the art on DL and Datalog at the time did not provide strong evidence about the feasibility of such approaches, and reasoning algorithms (needed for validation purposes) over such languages were computationally expensive and thus unfeasible for real cases. For this reason, it was not until ten years later that the first work presenting semi-automatic methods to support the MD design task by

Fig. 4. Ontologies for Semantic Annotations

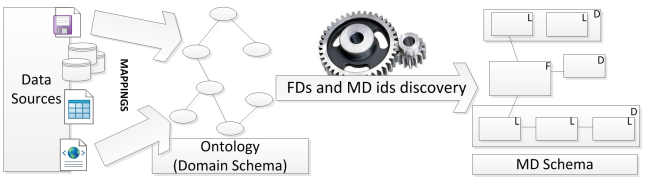

Fig. 5. Ontologies for Domain Modeling

exploiting the SW technologies appeared, empowered by recent advances both in DL and Datalog.

4.1.1 Ontologies for Semantic Annotations

Fig. 4 depicts one trend in this category of work, where the ontology plays a passive role. Thus, the data sources are checked (mainly sampling data using data mining techniques) to identify functional dependencies and MD identifiers and then annotate the findings in a reference ontology, following a GaV integration model. From this asserted information, facts are identified and, from each fact, a star-shaped schema is produced by deploying correct hierarchies and aggregations by means of functional dependencies. Consequently, note that the ontology is a repository of semantic annotations aimed at hiding heterogeneities. Reasoning is used to perform satisfiability and consistency checks (in the general case, reducible to subsumption checking) to validate the asserted knowledge, whose complexity varies according to the ontology language expressiveness.

As shown in Table 2 (see Section 3), all approaches in this section (see Data Schema Design - Sem.-aware) assume `Full` materialization of the DW. As a common trend, although they use SW technologies to overcome heterogeneities, the final DW produced mostly follows the *Traditional* assumptions and thus, do not fulfill the requirements of *Exploratory* cases.

Those papers following the trend depicted in Fig. 4 (i.e., [29], [30], [31], [32], [33], [34], [35]) neither provide support for tackling the system *Extensibility* in an automatic way nor deal with `Semi-structured` or `Unstructured` data. More specifically, Niemi et al. [29], [30] and Priebe et al. [31] have `Low` *Expressiveness* and thus yield `Medium` *Computation* (e.g., OWL 2.0 RL). Somehow, these approaches can be considered precursors to current *Exploratory* efforts aimed at bridging the gap between OLAP and linked data (e.g., [49]). Oppositely, Bakhtouchi et al. [32] and Prat et al. [33] try to tackle any scenario and choose a expressive standard DL language such as OWL DL. Thus, they provide `High` *Expressiveness* and yield `Hard` *Computation*.

Finally, Neumayr et al [34] and Anderlik et al. [35] present a detailed layered approach consisting of a *flat* domain (i.e., with no MD meaning), a hierarchy domain (showing roll-up relationships) and an MD domain. By means of integrity constraints (Datalog rules without head) and Datalog inference capabilities, they guarantee that the asserted information does make MD sense (i.e., preserves the MD constraints). Thus, they propose to analyze traditional data by using external ontologies as *semantic dimensions* (former work in this line was proposed in [50], in which a domain ontology serves as a basis to define an MD schema for aggregating instance data).

For all these papers adding new data sources is reduced to linking the sources to the reference ontology. Thus, they provide a certain degree of automation to model the DW only once the sources have been linked (i.e., mapped) to the reference ontology. This approach to add new data sources has two main consequences: the ontology languages used are less expressive (e.g., DL-Lite) but still the reasoning tasks carried out are computationally expensive: besides `Standard` reasoning such as subsumption checking, they need to compute the transitive closure of functional dependencies (a `Non-standard` service). Yet, these papers would rather use ad hoc algorithms than `Non-standard` reasoning services. Nevertheless, linking sources to the ontology can turn out to be hard in some scenarios (because mappings must be provided). Thus, in any case, adding new sources is time-consuming and it is not suitable for very `Dynamic` scenarios but for those where new data sources are added from time to time and intended to stay in the setting.

All in all, the main difference between these approaches is the ontology language used, distinguishing two main options: those focusing solely on RDF and those aiming to support a more generic and expressive approach. However, we note that both options have paid little attention to aggregation relationships up till now.

4.1.2 Ontologies for Domain Modeling

Fig. 5 sketches an alternative that achieves a larger degree of automation, while still in a *Traditional* scenario. In this case, sources are mapped to a reference ontology (with no MD meaning) and reasoning is exploited to identify functional dependencies and/or MD identifiers over the ontology rather than relying on the designer explicitly asserting them; all in all, this makes the ontology to play an active role. In general, expensive algorithms are triggered to identify MD patterns (by checking the MD constraints) and then, an MD schema based on this knowledge is created.

Most of this work assumes that an ontological representation of the sources is available (i.e., they are

working with Web data and taking the presence of SW technologies for granted).

Abelló and Romero [36] examine the ontology to look for MD identifiers (in order to identify facts), which is known to be expensive in the general case due to the large amount of instances a DW must deal with. To yield a tractable complexity, this work proposes the use of a reference ontology to pre-identify candidates and drastically reduce the number of tests (i.e., data samplings) to be carried out. `Standard` reasoning inferences (mainly subsumption checking) are used to spot out potential identifiers. However, ad hoc algorithms are introduced to deal with transitivity at the terminological level for OWL DL.

Romero and Abelló [37] identify, for each fact, potential dimensional concepts which are arranged in well-formed hierarchies by means of functional dependencies. In this sense, we can build an MD schema from the inferred functional dependencies. As common background, this kind of work suffers from the problem of computing the transitive closure of functional dependencies. In general, the transitive closure cannot be computed with `Standard` reasoning services and therefore [28] proposes a trick to simulate it in a $DL-Lite_A$ ontology by means of certain answers, while other similar approaches use ad hoc algorithms. Fortunately, transitive functional dependencies have been considered in OWL 2.0, under certain constraints, which opens the door for tackling this issue with `Standard` reasoning services (to at least some extent).

To our knowledge, no work has tackled the problem of discovering dimensions or MD identifiers from Datalog-based ontologies. However, Neumayr et al. [34] set foundations towards discovering functional dependencies by means of query answering over their Datalog metamodel.

4.2 Multidimensional Query Definition

In an *Exploratory* OLAP scenario, MD design should be driven by the user queries. This is because *Exploratory* users expect to be aware of new external data that fulfill their current business requirements, which are not covered by their internal corporate data. In this scenario, MD design must be flexible enough to allocate new incoming external data and to conciliate them with both internal data and user requirements.

To fulfill user requirements, an *Exploratory* OLAP system should ask for `Fresh` data in a highly `Dynamic` environment (e.g., the Internet), where an integration layer is hard to materialize. Thus, heterogeneity among sources becomes the main issue, also the lack of structure in the sources, which claims for new techniques to identify MD patterns on the fly.

4.2.1 Ontologies for Semantic Linked Modeling

Fig. 6 sketches the idea behind the work in this category. The main difference with regard to Fig. 5 is the

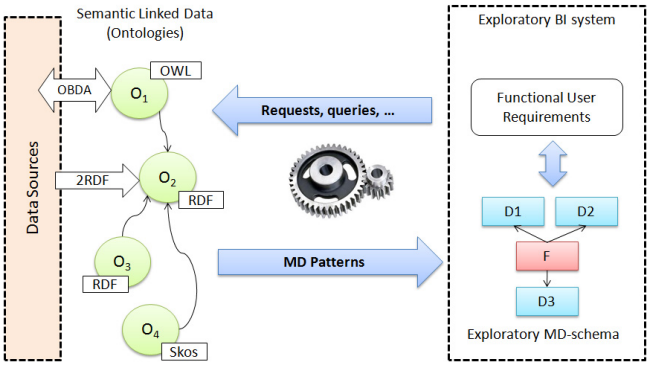

Fig. 6. Ontologies for Semantic Linked Modeling

lack of a centralized view of the domain (in terms of a single ontology) but the presence of several domain ontologies. Typically, data sources are published as semantic linked data, and users specify their requirements in the form of queries to existing catalogues of published datasets, which can be accessed via SW query languages (e.g., SPARQL). *Exploratory* MD design requires methods that should be user-driven and highly extensible. Semantic-aware approaches have been shown appropriate for conciliating user-requirements and existing data sources during the MD design phase. *Extensibility* is usually achieved through a loose coupling between conceptual schemas and user requirements. Reasoning is applied for functional dependency discovery and for validating the resulting MD schemas, which can be expressed as DL expressions. However, a fully *Exploratory* MD design method has not been proposed yet. This would basically consist of discovering, selecting and automatically linking those available data sources that best fit the user requirements. Assuming that all data sources are published under SW standards (e.g., Linked Open Data -LOD-), the main issues that we have to address are the following:

- Discover datasets that fulfill the user requirements, and get the subsets that are of interest;
- Extract useful relationships between the retrieved subsets, to discover potential MD facts; and
- Validate the discovered facts, for mixed datasets that may have incompatible semantics, that need to be integrated.

In the area of MD modeling, several user-driven and semantic-aware approaches have been proposed, which to some extent follow the integration models above. The feature common to all of them is that they allow the integration of `Unstructured` data by just mapping them to the available ontologies.

Firstly, Kämpgen et al. [11] propose a borderline *Traditional* DW to store and analyze statistical Linked Data (sLD) through OLAP queries. However, in this approach, there is no MD design because the DW only accepts sLD already expressed as MD data. It uses SPARQL as query language, and does not perform any kind of reasoning over semantics.

Romero et al. [38] follow the GaV philosophy. Thus, all data sources are tightly integrated into one global schema (called the annotation ontology). Then, an MD schema is constructed and validated constrained to the user's requirements, which are also specified in terms of the annotation ontology. Finally, ETL processes are semi-automatically designed to populate the generated model (see also Section 5.1). According to *Extensibility*, this work can be considered `Evolving`, since the inclusion or change of data sources implies the re-definition of the global schema, as well as their derived MD schemas and ETL flows.

Nebot et al. [39] propose a semantic DW, where both external and internal data are expressed, linked and stored as SW data (OWL format). Users specify their requirements by picking up concepts from the stored ontologies. Then, the system extracts the necessary logical modules to set up a global ontology from which facts and dimensions will be validated and generated from the DW.

Although this approach can be considered a *Traditional* DW, it has several features that make it *Exploratory*. First, if data sources are already expressed in OWL/RDF like in LOD, then their inclusion into the DW is straightforward. If not, a process of semantic annotation will be necessary. Mappings between ontologies and data sources of the DW can be semi-automatically obtained as in current semantic integration literature. Second, as the MD design is performed according to the user requirements, which are expressed in terms of the current ontologies in the DW, the resulting schemas are fitted to their specific needs at any moment. However, like Romero et al. [38], any change in the data sources will imply the re-definition and validation of the generated schemas and ETL flows. It must be pointed out that both approaches, [38] and [39], apply reasoning to discovering facts as well as validating the generated MD schemas.

Khouri et al. [40] propose another user-driven semantic integration approach. In this case, the authors propose a pre-defined global ontology into which data source ontologies are loosely integrated. User requirements are expressed as queries over the global ontology, which are executed to build up the DW conceptual schema (local ontology). In this approach, reasoning is applied to classify and validate the classes of the local ontology.

Kämpgen et al. [41] propose to directly query sLD to fulfill OLAP queries expressed in MDX, which resembles an *Exploratory* ETL flow as those described in Section 5.2. As Khouri et al. [40], SPARQL is used and no reasoning tasks are performed.

Finally, an alternative approach for extracting semantic dimensions from external ontologies is presented by Nebot and Berlanga [10]. In this case, two statistical measures are proposed for extracting MD-shaped hierarchies from the concept hierarchy. It is worth mentioning that these approaches deal with the concept taxonomic hierarchy (i.e., is-a relationships) to define roll-up operations, whereas most approaches mentioned in Section 4.1 deal with role chains.

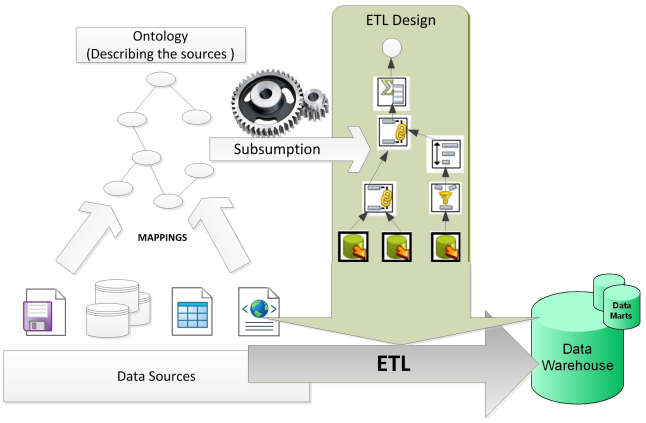

Fig. 7. Ontologies for ETL Modeling

5 Data Provisioning

Another challenge towards *Exploratory* OLAP is to shift from DW-centric, *Traditional* ETL flows to broader data flows consisting of complex analytic operations, involving a plurality of different data types and sources, spanning multiple execution engines, and running under different freshness requirements and at different paces, ranging from slow or frequent batches to micro-batches or real-time processing.

As with *Traditional* and *Exploratory* OLAP, where the latter requires a solution that captures, transforms, and presents fresh data in order to answer changing questions, we also see the need for *Exploratory* ETL processes. Depicted in Fig. 1, we have named these processes *Extract, Transform, and Query (ETQ)* in order to differentiate them from traditional ETLing. ETQs should be able to gather data, apply computations, and produce dynamic reports or populate dashboards directly from –potentially evolving– user requirements. In addition, ETQ processes deviate from *Traditional* processing in that they may affect various stages of the design, as for example they may be used to populate DW constructs or to answer a business query by fetching data directly from the sources (like on-demand ETL in [12]).

With the widespread adoption of new technologies in the Web, such as XML and other richer semi-structured formats like RDF, important and useful information is being captured in a large variety of data sources. Thus, real-world ETL scenarios have to deal with the integration of heterogeneous information sources. Dealing with the problem of accessing structured, less-structured, or unstructured data in an integrated and transparent way still has open challenges. Semantic annotation appears as a promising means for dealing with such issues, as it provides a semantic abstraction that can support both homogeneous access to disparate data sources and resource discovery.

In this context, semantic-aware ETL processes are those that take into account these semantic annotations to improve the integration processes required in OLAP solutions. In the rest of this section, we first review work related to semantic-aware ETL processes and then, we discuss preliminary efforts that deal with heterogeneous and unconventional data sources, which are dynamically incorporated to the analysis.

5.1 Semantic-Aware ETL Processes

During the initial steps of a *Traditional* ETL project, the main goal is to construct a conceptual ETL design that identifies the data sources that are useful to the project and describes the corresponding data transformations needed to map these sources to the target DW concepts. For achieving that, it is imperative to identify and understand the semantics of both the data sources and the target data stores. Although this is at large an open problem, some approaches propose the use of SW technologies to facilitate and to automate the design and construction of the ETL part.

5.1.1 Ontologies for ETL Modeling

Fig. 7 sketches (using Pentaho PDI icons) an approach to use SW technology for ETL. It uses a global ontology for mapping all the involved data stores to it. This idea resembles the LaV paradigm, where the application ontology, constructed as a conceptual schema of the domain, corresponds to the global schema and the semantic descriptions of the data stores, in terms of classes and properties defined in the ontology, correspond to the views describing the local schemas. However, the use of an OWL ontology, instead of a global schema, provides a formal model on which automated reasoning mechanisms may be applied. Furthermore, in ETL, it is not sufficient to consider the integration problem as a query rewriting problem, since the transformations taking place in real-case ETL scenarios usually include operations, that cannot be easily captured by a query rewriting process (see [42]).

The mixture of SW and ETL technologies is still quite immature. However, the preliminary efforts in this area have shown a potential solution to the automation of ETL designs (mainly at the conceptual level) given a set of business requirements, focusing on being more *Exploratory* along *Extensibility* and *Structuredness*, by facilitating the load of semi-structured (i.e., XML or RDF) data. With respect to other categorization criteria presented in Section 2.2, the entire spectrum of *Transformations* presented in Fig. 2 could be supported by the methods proposed. However, in practice, so far only simple and relational-style operations seem to be considered. How to generalize this work to capture a richer set of transformations (like user-defined functions) remains to be seen. *Freshness* is basically left aside and not even mentioned in some of the work, since they mainly deal with the conceptual level of ETL design, whereas this criterion involves lower design levels (i.e., physical). The same holds for *Materialization*, and all the authors more or less implicitly propose just to load the results into some kind of data storage.

Regarding the reasoning mechanisms used, if any, they are always `Standard` (i.e., subsumption) and only involve the schema, never the data. Despite that, given the *Expressiveness* chosen by some authors, *Computation* turns out to be `Hard` in most of them. Only Skoutas et al. [44] reduce the computational needs by restricting the reasoning to simple taxonomies.

Firstly, Niemi et al. [29] describe methods for OLAP cube construction using SW technology. The authors use a generic OLAP ontology as an upper ontology for all OLAP cubes. This ontology defines only general OLAP concepts and it is independent of the application area. Per application need, they consider domain-specific ontologies (e.g., CarModel, Branch, Country) based on the upper one. We may need to define ontology mapping transformations describing how the source data should be converted to conform to the global domain ontology. In order to integrate data from different sources, the authors consider an RDF data format and an RDF query language.

As an extension to their work, Niemi et al. [30] discuss in more detail the method for automating the construction of OLAP schemas. Again, the source and target schemas are considered as known. The mapping among the source data and the OLAP schema is done by converting the data in RDF using ontology maps. Then, the relevant source data are extracted by RDF queries generated using the ontology describing the OLAP schema. At the end, the extracted data are stored in a database and analyzed using typical OLAP techniques. Both these approaches aim at an end-to-end design approach, but they have two main limitations. First, they both require prior knowledge of the source and target schemas and second, they consider only simple data transformations.

Skoutas and Simitsis [42], [43] present an approach to ETL design using SW technology that elaborates more on the complexity of the data transformations required for integrating source data from heterogeneous sources into a DW. This work deals with a core ETL design challenge: the structural and semantic heterogeneity. For example, two sources S1 and S2 may contain similar information under two different schemas or they may use different representation formats. This approach uses ontologies to formally and explicitly specify the semantics of the source and DW schemas and thus, to automate to a large extent the ETL generation. It also assumes that the source and target schemas are previously known. A conceptual ETL design is obtained, whose generation involves the automatic derivation of the mappings from the source to the target attributes, along with the appropriate ETL transformations. However, computationally and

semantically complex ETL operations like pivot and slowly changing dimensions are not supported.

Skoutas et al. [44] extend the above mentioned work and propose using a graph-based representation as a conceptual model for the source and target data stores and based on that, the ETL transformations can be identified by means of graph transformations, instead of generic reasoning (which results to be computationally cheaper). It is described how the operations comprising the ETL process can be derived through graph transformation rules, the choice and applicability of which are determined by the semantics of the data with respect to an attached domain ontology.

The techniques discussed so far consider as given the source and target schemas and search for the mapping of one to the other, by benefiting from ISA relationships in a reference taxonomy. As a further step, Romero et al. [38] aim at building ETL and MD conceptual designs (see Section 4.2) starting from a set of data sources and a set of business requirements (e.g., expressed as service level objectives). This approach first analyzes the data sources mapped against a domain ontology. Then, in an iterative fashion, it produces two results: (a) an MD design that satisfies the requirement at hand (e.g., fact and dimension tables, relationships among them, etc.); and (b) a flow of conceptual ETL operations required to feed the MD schema by connecting it to the related data sources.

5.2 ETQ Processes

As earlier mentioned, *Exploratory* OLAP aims at analysing "fresh" data coming from a wide range of data sources. Fig. 1 shows that the provision of this kind of data would require alternative ETL processes, which we called ETQ. As discussed before, this area is generally unexplored by current approaches, but it contrasts with the great demand of this kind of processes within the community of OLAP.

SW technologies can play a crucial role in the development of ETQ processes. On the one hand, they facilitate to some extent the processing of unconventional data sources, containing semi-structured or unstructured data. Adding semantics to such data helps in shaping them under some notion of structure. This can be very helpful, given the effort spent in ETL research and materialized in ETL tools for dealing with structured information. On the other hand, SW technologies are enabling a new emerging scenario (i.e., the Web of Data) that offers new possibilities to publish semantic data for easy consumption by analytical tools. In this new scenario, virtualization and freshness can be fully achieved thanks to the use of SW references (e.g., URIs), which allows applications to access the required linked data at any time.

Thus, the main aim of ETQ processes is to avoid as much as possible complex computations over the imported data, so that these data can be included as fast as possible within the analytical queries (*Freshness* criterion). Unlike traditional ETL processes, ETQ processes can also be defined over unconventional data sources (*Structuredness* criterion), such as those usually found on the Internet. Current approaches partially cover some of these aspects by adopting RDF-based technology. However, applying reasoning within these processes is still an unexplored issue.

Firstly, Nebot and Berlanga [10], as a step towards the definition of semantic-based ETQ processes, propose a method to generate fact tables directly from semantic data published as RDF(S) and OWL. This method starts with the target MD query, which must be expressed in terms of concepts and properties of the source ontologies, and then it performs a series of transformations that guarantee that the generated factual data conforms to both the MD query and the source semantics. The main drawback of this approach comes from its computational complexity, since it requires to compute the aggregation patterns within the ontologies through a reasoner.

Pedersen et al. [9] provide an example of ETQ processes proposed to extend OLAP with external data. This method allows the execution of OLAP operations that involve data contained in external XML (not even RDF) sources through XPath queries. In doing so, external data can be used as dimensions and/or measures of OLAP cubes, offering high *Extensibility*, and *Freshness* of the results. Still, the XML sources need to be logically integrated within the OLAP system, preventing it from being completely `Dynamic`.

Finally, Kämpgen et al. [41] directly define OLAP operations over external data expressed in RDF and annotated using the RDF Data Cube vocabulary (QB). The authors define an MD schema based on QB and map OLAP operations in this schema to SPARQL queries over the sources. In this approach, the *Extensibility* criterion is enabled through the use of RDF and QB to annotate the data sources. One drawback of this approach is that the annotation with QB is manually performed, which implies a bottleneck in the design of the transformation processes, affecting the *Extensibility* criterion. Another limitation of this approach comes in terms of efficiency, as it depends on the SPARQL processor, which has been criticized as inefficient for complex MD queries.

6 Challenges

In this section, we summarize our findings and identify a list of challenges that require a fresh look in the future. We divide our discussion between the two areas of interest in this survey, namely schema design and data provisioning, but we also comment on whether SW technologies are ready to fully support the needs of next generation OLAP systems. Fig. 8 depicts the locality of these challenges (shown as numbers from 1 to 8) with respect to the various components of the ecosystem described in Fig. 1.

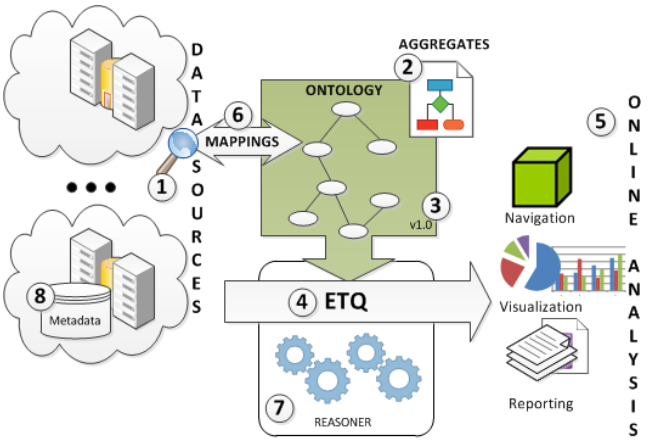

Fig. 8. Challenges in next generation OLAP

6.1 Schema Design Challenges

Most of the existing work for bringing together OLAP and SW technologies focuses on schema design issues. But our analysis on the work surveyed in Section 4 reveals that there are still unresolved challenges.

The common characteristic of the existing approaches is that these can deal with unstructured (or less structured) data, as soon as ontological mappings are provided. Getting these mappings is not trivial. In many cases, appropriate domain ontologies do not even exist. Thus, a first challenge involves the automatic (or semi-automatic) derivation of such mappings (♯1 in Fig. 8). Research on other, but related fields, like schema mappings and data exchange has provided very useful results that have not been exploited yet by research on DW design (see [51]).

Another limitation going toward more Exploratory schema designs is formed by existing approaches that assume that the extracted data are materialized, which typically happens off-line and is based on complex transformations. As we discussed, the reality and modern trends require a departure from this practice.

Our survey also revealed that there is a lack of SW tools for fully supporting OLAP functionality. An important substance of MD design is aggregation, as aggregation relationships are needed to rollup data. For automating DW design, we need to be able to reason on aggregations (♯2 in Fig. 8). However, these are not part of the OWL standard and only preliminary theoretical results are available. In [52], the authors argue that calculating aggregates under an open-world assumption is hard. As a step toward tackling this challenge, [53] shows that part-whole relationships can be represented in DL (despite that some of their properties cannot be modeled) and [54] explains how some kinds of aggregates can actually be managed using Datalog.

Recent work has dealt with the problem of ontology evolution and versioning in the SW scenario, even though it is in its infancy (see [55]). Future work should investigate how this kind of tools can be integrated with traditional DW evolution like in [17] to address the temporal issues in *Exploratory* OLAP scenarios (♯3 in Fig. 8). On the other hand, temporal analysis has not yet been approached by *Exploratory* OLAP work. However, one must bear in mind that temporalities usually lead to hard non-standard reasoning (see [26]).

6.2 Data Provisioning Challenges

As we presented in Section 5, there are preliminary efforts that use SW technologies to automate and to improve data provisioning. Our main conclusion is that SW technologies can be a powerful tool for data and semantic integration. Counterintuitively, reality shows that most approaches so far have dealt with the design problem, letting the data integration part largely unexplored.

The mixture of SW and ETL technologies is still quite immature. Preliminary efforts proved that automating the ETL design is doable. But existing approaches solve the problem partially and mostly at the conceptual level. Example solutions proposed involve creating ETL designs based on an analysis of business requirements and providing solutions for loading semi-structured data. Although these efforts seem to be in the correct direction, additional work is needed to move to a more *Exploratory* level (♯4 in Fig. 8).

A challenge that follows from our discussion on MD design issues involves the richness of integration operations that can be automated with SW technologies. Current practice considers ETL processes as a series of relational-style operations, but in practice ETL is much more complex. Dealing with non-relational operations, like user defined functions, machine learning based computations, etc. is not straightforward. Dealing with complex data types like bag-semantics, handling nulls, etc. is not trivial either. Considering non-persistent data, like streaming data, events, etc. complicates the semantics involved too.

Referring back to our criteria, *Freshness* and *Materialization* are not even considered by existing approaches to semantic-aware integration. Future research needs to couple the power coming from the semantic richness offered by SW technologies to the physical design, characteristics, and performance of integration designs (♯5 in Fig. 8). For example, the derivation of integration operations from the ontological mappings should take into account performance related issues (e.g., data selectivity or availability) to optimize the design created.

Despite it not being reflected in Table 2, another relevant characteristic is the integration model being followed. Clearly, *Exploratory* OLAP systems need to use LaV to be able to dynamically integrate data sources, without defining the global schema and mappings in advance (♯6 in Fig. 8). However, the computational cost of answering queries in this kind of architecture (especially for highly expressive logics) is well known.

The more expressive a language is, the more expensive the reasoning (being easily undecidable). Thus, to dynamically add data sources, we have to pay higher computational costs, which easily becomes infeasible.

6.3 Semantic and Computational Challenges

Being *Exploratory* means automating the access not only to schema but also to data to some extent. This implies we have to be able to cope with semantics and reasoning at the instance level (♯7 in Fig. 8). Therefore, most *Exploratory* approaches imply `Hard` *Computation* if *Expressiveness* is not drastically reduced. It is easier for any reasonably expressive ontology language to perform reasoning tasks on the terminology without asserted instances (combined complexity) than query answering over the asserted axioms (data complexity). The fact that work in ETL related to reasoning is also done at the schema level, without actually involving data, supports this assertion. Nevertheless, recent approaches such as OBDA (Ontology-Based Data Access, see [56]) open new directions. Thus, tractable DLs, such as DL-Lite, allow us to exploit query answering by querying the terminology.

Typically, Datalog has been used to reason on the instances of databases. It provides query answering (the model and data are somehow intertwined) in polynomial time under a closed-world assumption. Definitely, this would be the case for a DW supporting decision-making based only on the data it physically contains (i.e., it is a database and assumes a closed-world). However, if a system has to be truly *Exploratory*, we cannot guarantee that the necessary information will always be available. It makes sense to think of decision-making in open-world scenarios (e.g., what-if analysis), considering also those data on the Internet that could be dynamically added to our analysis. Thus, more research is necessary on efficient reasoning algorithms under an open-world assumption, maybe constrained to specific types of schemas (e.g., MD).

Also, there is a controversy related to the polynomial complexity of Datalog, since it is only achieved for standard query answering (i.e., against a fixed program) and failing to guarantee this restriction (which happens to be rather restrictive and not assumable in OLAP) it becomes EXPTIME-complete (see [25]). It is also argued that tractable DLs like the DL-Lite family provide query answering in LogSPACE (the same complexity as relational databases) and therefore, query answering in such families can be reduced to querying a relational database. This assumption is behind concepts such as OBDA. However, in practice, current OBDA tools have problems to match relational database response time.

Datalog and DL represent two different ways to deal with ontologies. In our opinion, however, both approaches cannot adequately cope with large amounts of instances and although these techniques can facilitate integration, in general, we do not envision a pure logic-based approach. We need semantics and we need reasoning, but it looks like they are not compatible in the presence of a high data volume.

The column `Computation` in Table 2 describes the worst case. Thus, *Computation* and *Expressiveness* are usually correlated (`High` *Expressiveness* implies `Hard` *Computation*). Nevertheless, it is worth mentioning that in practice some of these approaches do not achieve the worst case complexity, presenting for most ontologies a `Medium` *Complexity* for fair expressive languages. Moreover, some approaches may prefer to maintain `Medium/High` *Expressiveness* with easier *Computation* at the expense of incomplete (but sound) inferences, when complete reasoning is not required.

It is worth noting, that no work is using `Non-standard` reasoning services (e.g., least common subsumer, pattern matching, etc). Researchers prefer to use ad hoc algorithms on top of more mature `Standard` services. Thus, for example, Abelló and Romero [36], [37] incur higher *Computation* because of implementing ad hoc algorithms on top of the reasoner. We think that constraining or at least detecting some kind of expressions (e.g., MD queries) in the reasoners could be a solution to the puzzle (i.e., similar to detecting star-join patterns in relational query optimizers). It would not (only) be a trade-off between language expressiveness and reasoning cost, the kind of queries allowed should also come into play. So far, we described how OLAP can benefit from SW, but it is also true that research on how SW can benefit from OLAP, considering the latter as an enabling paradigm for making complex processing and reasoning more efficient.

From a different perspective, the computational problem could be tackled by increasing the computer power, e.g., by using a cluster of parallel machines (see [29]). In this sense, [57] addresses the challenge of distributed reasoning by using MapReduce. A drawback in this approach is the current structure of the SW. The existence of data niches hidden behind endpoints hinders the analysis of data. On the one hand, these conceal the contents of the different sites. On the other hand, data to be analyzed has to be moved from one site to another due to the limited query functionalities offered. RDF Data Cube Vocabulary (a W3C recommendation in [49]) goes towards solving this, by offering decisional functionalities.

Finally, we should also question whether the management of the SW is the most adequate for decision support (♯8 in Fig. 8). One of the main difficulties is integrating data, and the complete independence of repositories does not facilitate it. A more hierarchical organization (trading autonomy for consistency) would be more appropriate. At least, some registries showing the contents, availability, and data quality of each source are clearly needed.

7 Conclusions

This paper first classified OLAP schema design and data provisioning approaches that leverage SW technologies, based on the following criteria: *Materialization*, *Transformations*, *Freshness*, *Structuredness*, and *Extensibility*. It then analyzed the SW technologies according to the criteria of *Reasoning*, *Computation*, and *Expressivity*. The main conclusion was that SW technologies are indeed a promising approach for the new and challenging research area of *Exploratory OLAP*. The paper then identified a number of challenges for future research that must be met to fullfill theis promise, related to schema design and data provisioning, as well as semantic and computational issues.

Acknowledgment

This work has been partly supported by the Spanish Ministerio de Ciencia e Innovación under projects TIN2011-24147 and TIN2011-24747.

References

[1] S. Rizzi, A. Abelló, J. Lechtenbörger, and J. Trujillo, "Research in data warehouse modeling and design: dead or alive?" in *DOLAP*, 2006.

[2] W. Inmon, D. Strauss, and G. Neushloss, *DW2.0*. Morgan Kaufmann, 2008.

[3] A. Simitsis, K. Wilkinson, M. Castellanos, and U. Dayal, "Optimizing analytic data flows for multiple execution engines," in *ACM SIGMOD Conf.*, 2012.

[4] A. Ghazal, T. Rabl, M. Hu, F. Raab, M. Poess, A. Crolotte, and H.-A. Jacobsen, "BigBench: towards an industry standard benchmark for big data analytics," in *ACM SIGMOD*, 2013.

[5] A. Abelló, J. Darmont, L. Etcheverry, M. Golfarelli, J.-N. Mazón, F. Naumann, T. B. Pedersen, S. Rizzi, J. Trujillo, P. Vassiliadis, and G. Vossen, "Fusion cubes: Towards self-service business intelligence," *IJDWM*, vol. 9, no. 2, 2013.

[6] M. Middelfart and T. B. Pedersen, "The meta-morphing model used in targit bi suite," in *ER Workshops*, 2011.

[7] R. Berlanga, O. Romero, A. Simitsis, V. Nebot, T. B. Pedersen, A. Abelló, and M. J. Aramburu, "Semantic web technologies for business intelligence," in *Business Intelligence Applications and the Web: Models, Systems and Technologies*. IGI Global, 2011.

[8] A. Y. Levy, "The information manifold approach to data integration," *IEEE Intelligent Systems*, vol. 13, 1998.

[9] D. Pedersen, J. Pedersen, and T. B. Pedersen, "Integrating XML data in the TARGIT OLAP system," in *ICDE*, 2004.

[10] V. Nebot and R. Berlanga, "Building data warehouses with semantic web data," *DSS*, vol. 52, no. 4, 2012.

[11] B. Kämpgen and A. Harth, "Transforming statistical linked data for use in OLAP systems," in *I-SEMANTICS*, ser. ACM Int. Conf. Proc., 2011.

[12] U. Dayal, M. Castellanos, A. Simitsis, and K. Wilkinson, "Data integration flows for business intelligence," in *EDBT*, 2009.

[13] P. Vassiliadis and A. Simitsis, "Near Real Time ETL," in *New Trends in Data Warehousing and Data Analysis*, 2009.

[14] R. Kimball and M. Ross, *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. J. Wiley & Sons, 2002.

[15] P. Vassiliadis, M. Bouzeghoub, and C. Quix, "Towards quality-oriented data warehouse usage and evolution," *Information Systems*, vol. 25, no. 2, 2000.

[16] M. Golfarelli, J. Lechtenbörger, S. Rizzi, and G. Vossen, "Schema versioning in data warehouses: Enabling cross-version querying via schema augmentation," *DKE*, vol. 59, no. 2, 2006.

[17] M. Golfarelli and S. Rizzi, "A survey on temporal data warehousing," *IJDWM*, vol. 5, no. 1, 2009.

[18] M. Castellanos, U. Dayal, and M. Hsu, "Live business intelligence for the real-time enterprise," in *From Active Data Management to Event-Based Systems and More*, 2010.

[19] M. Essaidi, "ODBIS: towards a platform for on-demand business intelligence services." in *EDBT/ICDT Workshops*, 2010.

[20] H. Berthold, P. Rösch, S. Zöller, F. Wortmann, A. Carenini, S. Campbell, P. Bisson, and F. Strohmaier, "An architecture for ad-hoc and collaborative business intelligence," in *EDBT/ICDT Workshops*, 2010.

[21] J.-N. Mazón, J. J. Zubcoff, I. Garrigós, R. Espinosa, and R. Rodríguez, "Open business intelligence: on the importance of data quality awareness in user-friendly data mining," in *EDBT/ICDT Workshops*, 2012.

[22] V. Markl, "Situational business intelligence," in *BIRTE*, 2008.

[23] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge Univ. Press, 2003.

[24] S. Ceri, G. Gottlob, and L. Tanca, "What you always wanted to know about datalog (and never dared to ask)," *IEEE TKDE*, vol. 1, no. 1, 1989.

[25] P. F. Patel-Schneider and I. Horrocks, "Position paper: a comparison of two modelling paradigms in the semantic web," in *WWW*, 2006.

[26] B. Motik, "Representing and querying validity time in rdf and owl: A logic-based approach," *J. Web Sem.*, vol. 12, 2012.

[27] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev, "The DL-Lite Family and Relations," *J. of AI Research*, vol. 36, 2009.

[28] O. Romero, D. Calvanese, A. Abelló, and M. Rodriguez-Muro, "Discovering functional dependencies for multidimensional design," in *DOLAP*. ACM, 2009.

[29] T. Niemi, S. Toivonen, M. Niinimäki, and J. Nummenmaa, "Ontologies with semantic web/grid in data integration for OLAP," *Int. J. Semantic Web Inf. Syst.*, vol. 3, no. 4, 2007.

[30] M. Niinimäki and T. Niemi, "An ETL process for OLAP using RDF/OWL ontologies," *JoDS*, vol. 5530, 2009.

[31] T. Priebe, A. Reisser, and D. T. A. Hoang, "Reinventing the Wheel?! Why Harmonization and Reuse Fail in Complex Data Warehouse Environments and a Proposed Solution to the Problem," in *Tagung Wirtschaftsinformatik*, 2011.

[32] A. Bakhtouchi, L. Bellatreche, S. Jean, and Y. Aitameur, "MIR-SOFT: mediator for integrating and reconciling sources using ontological functional dependencies," *Int. J. of Web and Grid Services*, vol. 8, no. 1/2012, 2012.

[33] I. M. Nicolas Prat and J. Akoka, "Multidimensional models meet the semantic web: Defining and reasoning on OWL-DL ontologies for OLAP," in *DOLAP*. ACM, 2012.

[34] B. Neumayr, S. Anderlik, and M. Schrefl, "Towards ontology-based OLAP: datalog-based reasoning over multidimensional ontologies," in *DOLAP*. ACM, 2012.

[35] S. Anderlik, B. Neumayr, and M. Schrefl, "Using domain ontologies as semantic dimensions in data warehouses," in *ER*, ser. LNCS. Springer, 2012, vol. 7532.

[36] A. Abelló and O. Romero, "Ontology driven search of compound IDs," *KAIS*, vol. 32, no. 1, 2012.

[37] O. Romero and A. Abelló, "A framework for multidimensional design of data warehouses from ontologies," *DKE*, vol. 69, no. 11, 2010.

[38] O. Romero, A. Simitsis, and A. Abelló, "GEM: Requirement-driven generation of ETL and multidimensional conceptual designs," in *DaWaK*, ser. LNCS, vol. 6862. Springer, 2011.

[39] V. Nebot, R. Berlanga, J. M. Pérez-Martínez, M. J. Aramburu, and T. B. Pedersen, "Multidimensional integrated ontologies: A framework for designing semantic data warehouses," *JoDS*, vol. 5530, 2009.

[40] S. Khouri, I. Boukhari, L. Bellatreche, E. Sardet, S. Jean, and M. Baron, "Ontology-based structured web data warehouses for sustainable interoperability: requirement modeling, design methodology and tool," *Computers in Industry*, vol. 63, 2012.

[41] B. Kämpgen, S. O'Riain, and A. Harth, "Interacting with statistical linked data via OLAP operations," in *Int. Workshop on Interacting with Linked Data (ILD 2012), Extended Semantic Web Conf. (ESWC)*. CEUR-WS.org, 2012.

[42] D. Skoutas and A. Simitsis, "Ontology-based conceptual design of ETL processes for both structured and semi-structured data," *Int. J. Semantic Web Inf. Syst.*, vol. 3, no. 4, 2007.

[43] A. Simitsis, D. Skoutas, and M. Castellanos, "Representation of conceptual ETL designs in natural language using Semantic Web technology," *DKE*, vol. 69, no. 1, 2010.

[44] D. Skoutas, A. Simitsis, and T. K. Sellis, "Ontology-Driven Conceptual Design of ETL Processes Using Graph Transformations," *JoDS*, vol. 13, 2009.

[45] J.-N. Mazón, J. Lechtenbörger, and J. Trujillo, "A survey on summarizability issues in multidimensional modeling," *DKE*, vol. 68, no. 12, 2009.

[46] J. M. Pérez, R. Berlanga, M. J. Aramburu, and T. B. Pedersen, "Integrating data warehouses with web data: A survey," *IEEE TKDE*, vol. 20, no. 7, 2008.

[47] D. Calvanese, M. Lenzerini, and D. Nardi, "Description logics for conceptual data modeling," in *Logics for Databases and Information Systems*. Kluwer, 1998.

[48] M.-S. Hacid, P. Marcel, and C. Rigotti, "A rule-based data manipulation language for OLAP systems," in *DOOD*, ser. LNCS, vol. 1341. Springer, 1997.

[49] W3C, "The RDF Data Cube vocabulary. W3C Working Draft," http://www.w3.org/TR/vocab-data-cube, 2012.

[50] R. Dánger and R. Berlanga, "Analysis of ontological instances - a data warehouse for the semantic web," in *ICSOFT (ISDM/EHST/DC)*. INSTICC Press, 2007.

[51] P. G. Kolaitis, "Schema mappings, data exchange, and metadata management," in *PODS*, 2005.

[52] D. Calvanese, E. Kharlamov, W. Nutt, and C. Thorne, "Aggregate queries over ontologies," in *ONISW*. ACM, 2008.

[53] U. Sattler, "Description logics for the representation of aggregated objects," in *ECAI*. IOS Press, 2000.

[54] J. Seo, S. Guo, and M. S. Lam, "Socialite: Datalog extensions for efficient social network analysis," in *ICDE*, 2013.

[55] E. Jiménez-Ruiz, B. C. Grau, I. Horrocks, and R. B. Llavori, "Supporting concurrent ontology development: Framework, algorithms and tool," *DKE*, vol. 70, no. 1, 2011.

[56] A. Poggi, D. Lembo, D. Calvanese, G. D. Giacomo, M. Lenzerini, and R. Rosati, "Linking data to ontologies," *JoDS*, vol. 10, 2008.

[57] J. Urbani, S. Kotoulas, J. Maassen, F. van Harmelen, and H. E. Bal, "WebPIE: A Web-scale Parallel Inference Engine using MapReduce," *J. Web Sem.*, vol. 10, 2012.

Torben Bach Pedersen is a Professor of Computer Science at Aalborg University, Denmark. His research concerns business intelligence and big data, especially "Big Multidimensional Data" - the integration and analysis of large amounts of complex and highly dynamic multidimensional data. He is an ACM Distinguished Scientist, an IEEE Senior Member, and a member of the Danish Academy of Technical Sciences.

Rafael Berlanga is Professor of Computer Science at Universitat Jaume I, Spain, and leader of the Temporal Knowledge Bases group. His current research concerns novel text mining methods for performing Business Intelligence over very large resources such as the Semantic Web and social media. He has published more than 25 contributions to high-impact international journals and more than 50 contributions to international conferences.

Victoria Nebot obtained her Ph.D. in computer science from the Universitat Jaume I, Spain in 2013. She is currently a post-doctoral researcher in the Temporal Knowledge Bases Group (TKBG) in Universitat Jaume I. Her main research is focused on analyzing and exploiting semistructured and complex data derived mainly from the Semantic Web. Also, she is interested in text mining and information extraction methods for the Semantic Web.

Alberto Abelló is Tenure Track professor. PhD in Informatics, UPC. Local coordinator of the Erasmus Mundus PhD program IT4BI-DC. Active researcher with more than 50 peer-reviewed publications and H-factor 18, his interests include Data Warehousing and OLAP, Ontologies, NOSQL databases and BigData management. He has served as Program Chair of DOLAP and MEDI, being member also of the PC of other database conferences like DaWaK, CIKM, VLDB, etc.

María José Aramburu is associate professor of Computer Science at Universitat Jaume I. She received the PhD degree from the University of Birmingham (UK) in 1998. Author of articles in international journals such as Inf. Proc. & Management, Decision Support Systems as well as papers in international conferences such as ICDE, DEXA, and ECIR, her main research interests include knowledge repositories, decision support systems and integration of information.

Oscar Romero is a tenure-track lecturer at UPC. He obtained his PhD in Computing in 2010. His research interests concerns data modeling and storage for next generation data warehousing systems, focusing on distributed data management, management and optimization of data intensive flows, semantic-aware information integration and the development of user-centric functionalities for service-oriented BI.

Alkis Simitsis is a senior research scientist in the Analytics Lab at HP Labs. His research career spans multi-objective optimization of hybrid analytics flows, real-time business intelligence, massively parallel processing, query processing/optimization, data warehouses, ETL, web services, and user-friendly query interfaces focusing on keyword search and NLP techniques. He has published more than 90 papers in refereed international journals and conferences in the above areas and has served in various roles in many program committees including SIGMOD, VLDB, ICDE, and EDBT. He is a member of ACM, IEEE, and the Technical Chamber of Greece.