Fusion Cubes:
Towards Self-Service Business Intelligence

Alberto Abelló, School of Informatics, Universitat Politècnica de Catalunya, Barcelona, Spain

Jérôme Darmont, Université de Lyon (Laboratoire ERIC), Lyon, France

Lorena Etcheverry, Computer Science Institute, Universidad de la Republica, Montevideo, Uruguay

Matteo Golfarelli, DISI - Department of Computer Science and Engineering, University of Bologna, Bologna, Italy

Jose-Norberto Mazón, Department of Software and Computing Systems, Universitat d'Alicante, Alicante, Spain

Felix Naumann, Department of Information Systems, Hasso Plattner Institute, Potsdam, Germany

Torben Bach Pedersen, Department of Computer Science, Aalborg University, Aalborg, Denmark

Stefano Rizzi, Department of Computer Science and Engineering, Università di Bologna, Bologna, Italy

Juan Trujillo, Department of Information Systems and Languages, Universitat d'Alicante, Alicante, Spain

Panos Vassiliadis, Department of Computer Science, University of Ioannina, Ioannina, Greece

Gottfried Vossen, Department of Information Systems, Universität Münster, Münster, Germany

ABSTRACT

Self-service business intelligence is about enabling non-expert users to make well-informed decisions by enriching the decision process with situational data, i.e., data that have a narrow focus on a specific business problem and, typically, a short lifespan for a small group of users. Often, these data are not owned and controlled by the decision maker; their search, extraction, integration, and storage for reuse or sharing should be accomplished by decision makers without any intervention by designers or programmers. The goal of this paper is to present the framework we envision to support self-service business intelligence and the related research challenges; the underlying core idea is the notion of fusion cubes, i.e., multidimensional cubes that can be dynamically extended both in their schema and their instances, and in which situational data and metadata are associated with quality and provenance annotations.

Keywords: *Business Intelligence, Data Cube, Data Fusion, Data Integration, Data Warehouses, ETL, Metadata Discovery, Metadata Quality, Open Data, Schema Discovery*

DOI: 10.4018/jdwm.2013040104

INTRODUCTION

Today's business and social environments are complex, hyper-competitive, and highly dynamic. When decisions have to be made quickly and under uncertainty in such a context, the selection of an action plan must be based on reliable data, accurate predictions, and evaluations of the potential consequences. *Business intelligence* (BI) tools provide fundamental support in this direction. For instance, in medium and large companies, BI tools lean on an integrated, consistent, and certified repository of information called a *data warehouse* (DW), which is periodically fed with operational data. Information is stored in the DW in the form of multidimensional cubes that are interactively queried by decision makers according to the OLAP paradigm (Golfarelli & Rizzi, 2009). In this work, we call *stationary* the data that are owned by the decision maker and can be directly incorporated into the decisional process. Stationary data may take either operational or multidimensional form; in both cases, their quality and reliability is under the decision maker's control. In a corporate scenario, the data stored in the company DW and information system are stationary.

However, well-informed and effective decisions often require a tight relationship to be established between stationary data and other data that fall outside the decision maker's control (Pérez et al., 2008; Trujillo & Mate, 2011; Golfarelli, Rizzi, & Cella, 2004; Darmont et al., 2005). These valuable data may be related, for instance, to the market, to competitors, or to potential customers, and are sometimes called *situational* data (Löser, Hueske, & Markl, 2008):

We call situational those data that are needed for the decisional process but are not part of stationary data. Situational data have a narrow focus on a specific domain problem and, often, a short lifespan for a small group of decision makers with a unique set of needs.

In some cases, situational data can be retrieved (for free or for a fee) in a semi-structured form by accessing established data providers, such as *DBpedia* (Auer et al., 2007, cross-domain), *ProductDB* (productdb.org, commerce domain), *Geonames* (sws.geonames.org, geography), or *DATA.GOV* (www.data.gov, public institutions); for instance, in DBpedia the structured content extracted from Wikipedia is mapped onto a cross-domain ontology and can be queried using SPARQL. In other cases, situational data are chaotically scattered across heterogeneous and unstructured sources available on the Web (e.g., opinions expressed by users on social networks, ratings of products on portals, etc.). In general, situational data tend to be highly dynamic in contrast to stationary data, which are used to address a large set of decisional problems and impose a slow and careful management. A quick comparison of the main features of situational and stationary data is reported in Table 1.

The capability of incorporating situational data into the decision process gives rise to a new class of applications which, in the context of BI 2.0, are often labeled as *situational BI*, *on-demand BI*, or even *collaborative BI*. In this paper we use the term *self-service BI* to emphasize that the search, extraction, and integration of situational data should be accomplished by users through a continuous interaction with the application, without any mediation or intervention by analysts, designers, or programmers.

As also emphasized by Gartner Inc., self-service BI appears to be the big wave in BI since 2010 (Horwitt, 2010). The key idea is to enable non-expert users to make well-informed decisions when required, by letting them navigate information in a "surf and save" mode, meaning that data can be stored for reuse or sharing. Among the main applications for self-service BI in a corporate scenario we mention brand positioning, pricing, competitor monitoring, policy planning, and risk management; the domains involved range from retail, telecommunications, and entertainment to finance and public services, such as health and transportation.

This trend also finds an echo at smaller-scale levels, i.e., in non-governmental organizations, watchdog groups (non-profit groups

Table 1. Stationary vs. situational data

	Stationary Data	Situational Data
Form	structured	semi-structured or unstructured (textual)
Source	DW, databases, ...	data providers, portals, forums, blogs, ...
Integration	at design time	at runtime
Lifespan	years	days to weeks
Reusability	high	low
Availability	24h	no guarantees
Reliability	cleansed and certified data	very variable
Quality	integrity and consistency	user ratings, owner ratings, freshness...

that critically monitor the activities of governments, industry, or other organizations), Web communities, and even at the average citizen's level. In fact, people increasingly demand open data (e.g., the Spanish *indignados* and the New Zealand initiative on open governmental data), which they need to access easily from the Web, to mix with private (stationary) data, to analyze with intelligent on-line tools, and to share with their peers or even world-wide.

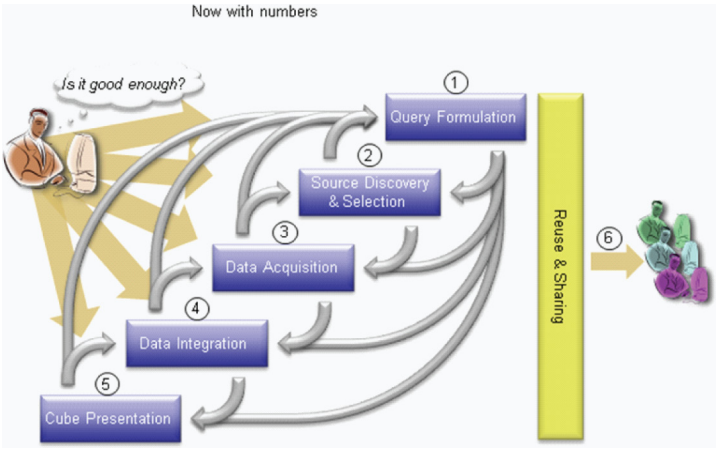
Example: Watchdog Scenario

A European watchdog group monitors fishing activities in the EU. The stationary data they use to this end are the official fishing statistics for member states, which can be accessed from epp.eurostat.ec.europa.eu in a sort of pivot table (i.e., in multidimensional form). These data can be analyzed by region, year and caught species. Now they wish to evaluate whether marine protected areas near fishing regions positively or negatively affects catches, thus evaluating whether EU investment in protecting marine areas have visible effects in terms of the volume of fishing catches. This requires incorporating and cross-analyzing situational data that are heterogeneous in many respects. Two kinds of data must be considered: (i) geographical data to consider protected marine areas, and (ii) fish population data from these marine protected areas. While geographical data can be acquired from the World Database on Protected Marine Areas (www.wdpa-marine.org), fish populations must be acquired from research papers. Field observations by activists taken for instance from the Greenpeace blogs could also come into play.

Achieving the decisional goal described above would require a long and cumbersome process for manually extracting situational data and integrating them with stationary data. The aim of this paper is to present the framework we envision to partially automate this process so as to support self-service BI. The typical interaction scenario we foresee can be sketched as follows (see Figure 1); each step is described in more detail in the sections below:

1. A user poses an OLAP-like *situational query*, i.e., one that cannot be answered on stationary data only. In our running example, the watchdog group wants to compare the volume of catches with the observations of the same species in bordering protected marine areas;
2. The system discovers potentially relevant data sources (i.e., sources related to geographically protected marine areas and fish populations);
3. The system fetches relevant situational data from selected sources. In our example, retrieved situational data should be related to European countries, for several years and certain fish species;

Figure 1. Self-service BI: A functional view



4. The system integrates situational data with the user's data, if any. Observations from protected marine areas should be aggregated by country, species, and year in order to be integrated with statistical data about catches;
5. The system visualizes the results and the user employs them for making her decision. Our running example would visualize data in a map in order to see both catches and fish populations and easily determine if there is a correlation between them;
6. The user stores and shares the results, either with a certain group only, or publicly, so that they can be reused (possibly by others) in subsequent analyses. New results from our running example can be useful as a starting point to include the investment done by each country in protected marine areas and the revenue in form of fishing catches.

The entire scenario is strictly supervised by the user, who evaluates the outcome of each step and possibly triggers iterations to the previous steps to improve the quality of the results.

The steps of the querying process are discussed in the remainder of this paper. Some of these steps require challenging research problems to be solved; however, as shown in Section DEPLOYMENT, the overall framework is quite realistic. Before going into details we present an overview of the architecture we envision to support our framework and survey the related literature.

Components and Architecture for Self-Service BI

The main contribution of our proposed architecture is the *fusion cube*. Different from a "traditional" multidimensional cube, a fusion cube can be dynamically extended both in its schema and its instances; besides, each piece of data and metadata in a fusion cube is associated with a set of annotations that describe its quality from different points of view, the source it was taken from, its freshness and estimated validity, its reliability, and so on. In our example, the stationary cube would be populated by fishing statistics, which would situationally be augmented by domestic regulations as well as field observations. We imagine the stationary cube to follow the traditional format of a star or a snowflake schema, facts or dimensions of which will be augmented by situational data into a fusion cube. These additional data is preferably attached as RDF (Resource Description Framework) data in triple form. More precisely, both facts and dimensions can be

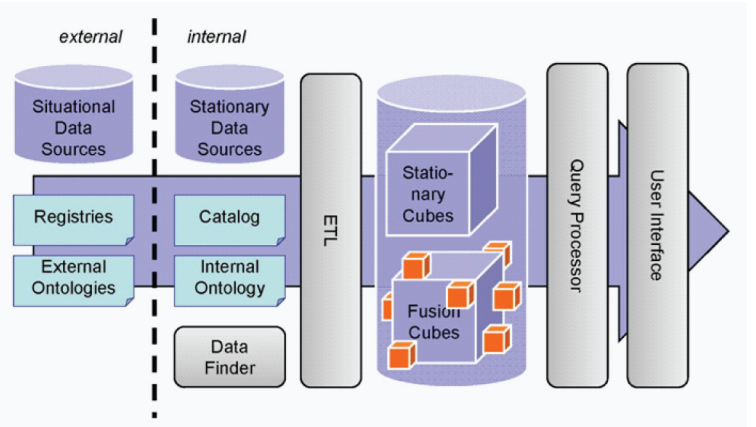
augmented by additional attributes referencing entities described in RDF format. In this way, traditional warehouse contents can be combined with semi- or unstructured contents as required by a given application.

Figure 2 shows the other components of our architecture. The *user interface* enables users to pose situational queries in an OLAP-like fashion. Queries are then handed on to a *query processor*, which supports declarative queries and translates them to executable query processing code. The posed query can refer to a stationary cube or to a fusion cube already defined by another user; alternatively, the user may need to create a new fusion cube, in which case new situational data to populate it must be found. To this end, the *data finder* uses external *registries* as well as external ontologies or it just accesses the metadata already in the *catalog* as well as in the internal ontology. Registries can be complex services or just a simple search engine. The catalog as well as the internal ontology include metadata of *internal data sources* (the available cubes and fusion cubes, operational databases, and ETL flows) as well as those of already known *external data sources* together with their quality attributes.

Recently, more and more different types of *external data sources* have appeared, each with its specific features, its information content, and its providers such as opinions, reviews, annotations, profiles, data from competitors, sensor networks and social network data, etc. (Agrawal, Das, & Abbadi, 2011). This huge variety of different and heterogeneous external data sources are often referred as *Big Data* (Cohen et al., 2009). The most notable examples are listed below, together with some annotations about their integration with multidimensional data:

- An increasing number of data sources have been using the *RDF* data model for their datasets. Some works in the literature are focused on building DWs from RDF data (Niinimäki & Niemi, 2009; Nebot et al., 2009); however, the integration of RDF data into DWs is still done partially and on-the-fly, providing limited analysis capabilities and with strict time constraints;
- *Linked data* is the practice of using the Web to share and connect related data; in linked data, relationships between pieces of data are given a semantics and can be used for reasoning (Bizer, Heath, & Berners-Lee, 2009). Some examples of this are traceability approaches (Mate & Trujillo, 2012) and semantic tagging (Bizer, Heath, & Berners-Lee, 2009). The automatic processing of linked data and its combination with multidimensional data will presumably be a hot topic in BI over the next years;

Figure 2. The fusion cube architecture



- *Open data* are becoming more and more popular, as it is widely recognized that certain data should often, but not always be freely available to everyone to use and republish as they wish. In particular, if there is no fee for these data, they can play a prominent role in self-service BI for extending the internal information content;
- *Social networks* allow their participants to interact and continuously provide data. This results in huge data collections that give precious insights about people attitudes, so there is an enormous interest in analyzing these data. Though some tools, such as MicroStrategy, enable social network data analysis, much work still must be done to improve the quality of the returned data so as to facilitate their integration with the internal data. In addition, legal and ethical boundaries of privacy must be preserved;
- The terms *opinion mining a*nd *sentiment analysis* refer to the process of capturing and describing the general feelings or opinions of a group of people about a certain fact. The source data for this process is typically found in the Web in an unstructured form, e.g., in blogs. The resulting information can be crucial for decision making (Xu et al., 2011), but its integration with multidimensional data is still a challenging problem (Perez et al., 2008).

In a next step the *ETL* component extracts structure from situational data, integrates them with internal data, and stores them into fusion cubes. Fusion cubes may refer to their local storage for materialization purposes or RDF augmentations as indicated above (also see the section of storage for details). Finally, the user interface shows the results to the user by emphasizing their provenance and estimated quality. As will also be described later, the deployment of all these components can take various forms ranging from tightly-coupled subsystems to loosely-coupled services.

We mention that the architectural components of Figure 2 are chosen to support the process steps shown in Figure 1. For example, the User Interface will support query formulation as well as cube presentation, and the Data Finder together with the ontologies will be responsible for source discovery and selection.

RELATED WORK

While each of the following subsections points to the most relevant works within the scope of specific steps of the process shown in Figure 1, in this section we survey some approaches that are related to self-service BI in a more general sense.

Early research has focused on integrating so-called complex data (i.e., heterogeneous and diversely structured data from multiple Web sources) into decision-support processes. Boussaid et al. (2008) propose a complex data warehousing methodology that exploits XML as a pivot language. This approach includes the integration of complex data into an operational data store in the form of XML documents, their dimensional modeling and storage in an XML data warehouse, and their analysis with combined OLAP and data mining techniques. See also (Zorrilla et al., 2011) for a comprehensive investigation of the main issues arising from the interaction between BI applications and the Web. Pedersen et al. propose operators for extending existing cubes with external XML data, including the *decoration* operator that adds a new dimension (Pedersen, Pedersen, & Riis, 2004) and the *extension* operator that adds a new measure (Pedersen, Pedersen, & Pedersen, 2008). However, this line of work is restricted to XML data only and considers neither source discovery, nor collaborative and data quality aspects.

Löser et al. (2008) create a case for situational BI. Their vision relies on a cloud architecture and on an extensible algebraic model for executing complex queries over unstructured and (semi-) structured data; in particular, unstructured text retrieved from documents is seen as a primary source for information. The process envisioned for query answering encompasses five stages:

1. **Data retrieval** (e.g., by accessing the company Intranet or using Web crawlers);
2. **Data extraction** (to transform unstructured data into semi-structured data);
3. **Data cleansing** (e.g., to complete missing data);
4. **Schema generation** (to give some multidimensional structure to cleansed data);
5. **Query processing** (based on traditional database operators).

Thiele and Lehner (2011) envision two technical solutions to support situational BI. The first are *spreadmarts*, meant as analysis systems running on a desktop database that are created and maintained by an individual or a group; like a normal data mart, a spreadmart includes an ETL stage as well as an analysis front-end, but with no guarantees about the quality and consistency of information. The second solution are *analytical mashups* that combine existing data and services from a variety of sources to create new contents; in this approach, the traditional ETL process is replaced by a *fetch*, *shape*, and *pick* process driven by a single individual or department. In this context, migrating current DW architectures for supporting situational data can be cumbersome; in this sense, Jörg & Dessloch (2009) propose adapting current ETL tools to ensure a consistent state of the DW when data sources are delivered with lower latency (as is the case for situational data).

An architecture for collaborative BI is depicted by Berthold et al. (2010); here, the term "collaborative" is used to emphasize that combining social software with BI allows the decision-making process to be enriched with opinions of experts. The authors also plan to accommodate self-service queries, mainly by adopting sophisticated mash-up technologies. Enabling collaboration is also a goal of Cabanac et al. (2007), who annotate multidimensional cubes to model and share the expertise of decision makers. Another approach to collaborative BI is the one by Golfarelli et al. (2012), where several data marts are included in a peer-to-peer network; though the decision making process is dynamically enhanced through knowledge sharing, no ad-hoc integration of situational data taken from external sources is supported.

A platform for on-demand BI services is proposed by Essaidi (2010); here, however, the term "on-demand" means that a *software-as-a-service* model is adopted, not that situational data can be dynamically integrated.

QUERY FORMULATION

Posing a situational query does not simply imply writing some MDX commands or using a traditional OLAP front-end, but it typically requires a more complex process aimed at extending the information previously available. In the worst case, no relevant information is available at all within internal data sources, so an additional *discovery* phase becomes necessary to trigger the query process (details below). In the following subsections we explore the issues related to both situations.

OLAP Querying

A traditional OLAP session is typically composed of a sequence of multidimensional queries, each obtained by recursively applying an OLAP operator to the previous one. Self-service BI comes into play when the internally available cubes are not sufficient to completely satisfy users' situational needs, so either their data or their multidimensional schemata must be extended. In the first case, a query matches a stationary cube schema but some relevant situational dimensional data and measures values are missing. For instance, in our watchdog example, the Eurostat site could not initially include the instances related to some countries that are candidates to join the EU and the related values for the measures in the fishing activities cube. In the second case, some new measures and/or dimensions are required that are not available in the stationary cubes. For instance, an additional attribute reporting a new categorization of species could be needed by a query. Note that such functionalities are a

very significant generalization of the decoration and extension operators by Pedersen et al. (2004; 2008).

To help users cross the borders of stationary cubes, a new OLAP operator must be devised. We call this operator *drill-beyond*: its main goal is to provide the user with an easy-to-use interface for specifying *how* and *where* a cube should be extended. Instead of simply running a new query on a stationary cube, the drill-beyond operator triggers the process summarized in Figure 1 (and described in detail in the following sections) and collects the information necessary to run it. The drill-beyond metaphor could be applied to every type of OLAP interface; in the following we will describe it in terms of the conceptual elements of a cube. We envision a user can drill-beyond:

- …the *schema* by clicking on either a dimensional attribute or a fact. In the first case the meaning is that an additional attribute describing or aggregating the clicked one is needed. In the second case, either a new dimension or a new measure is needed. For example, the user could click on the "species" attribute and choose an "add child" menu item to require that an additional attribute "fat content category" is added to further describe species;
- …the *instances*: by clicking on a dimensional attribute to specify that additional instances for that attribute are needed. Note that the measure values related to the new attribute values must be retrieved, too. For example, clicking on the "country" attribute and choosing the "add instances" menu item starts the search of the required instances.

After choosing *where* the cube must be extended, the user must provide some additional information about *how* to extend it. Since the information required cannot be precisely expressed in terms of the available schema, we believe that, to enable maximum flexibility, drill-beyond should include a question answering system or it should be keyword-based. The main drawback of keyword search is its limited capabilities in returning precise answers; on the other hand, though some studies show that coupling a question answering system with a DW can increase the answer precision and facilitate in structuring the results (Ferrández & Peral, 2010), their effectiveness in open domains is still limited. In both cases, keywords and/or a natural language query should be provided by the user. For example, in our *add instances* scenario appropriate keywords could be "fisherman activities in Croatia".

Since the complexity of the process depicted in Figure 1 strictly depends on the characteristics of the acquired information, the user could provide additional parameters to define a trade-off between completeness of the acquired data and acquisition/integration effort. From this point of view, sources can be distinguished into *internal* (listed in the internal catalog, though not available into fusion cubes yet) and *external* (listed in an external registry or retrieved from the Web). Limiting the search to internal sources could be useful to reduce the integration effort. Users could also be interested in distinguishing between open data providers and fee-based data providers, or even in limiting the search to a given provider.

Notably, though managing the drill-beyond operator can be complex, a recommendation system could significantly reduce the query formulation effort by suggesting possible drill-beyond actions previously done by the same user or by other users in the same querying context (Jerbi et al., 2009; Aligon et al., 2011).

Cube Discovery

In traditional corporate scenarios, we can take for granted that one or mores stationary cubes storing cleansed and certified data (see Table 1) exist and can serve as the foundation for drilling beyond. However, this is not always the case; for instance, in the watchdog scenario, there will initially probably be no cubes, but only Web data in some (non-multidimensional) format—a typical case for this type of self-service BI scenarios. The user thus only has some vague ideas about the possibly available data and how

to structure and analyze it, and the system should aid in transforming these ideas into a concrete cube schema.

Here, we must "bootstrap" the process by running *cube discovery* on selected situational data sources, e.g., by discovering dependencies among attribute values to identify dimension hierarchies, and by looking at cardinalities, data types, and data values to identify measures. For example, the system could suggest "Fisherman's License Type" as a possible hierarchy level in a Fisherman dimension and "Volume of Catches" as a measure. Realistically, this process can be at most semi-automated: the user provides some keywords to outline facts, dimensions, and measures; some relevant patterns are found and possible results are returned to the user, who then either confirms or proposes alternatives. In some sense, this resembles the (now discontinued) *Google Square* but with a significantly more powerful BI-related functionality to discover not only simple structure in data, but also the much more relevant concepts of dimensions with hierarchies and measures. To this end we can build on the existing work in cube discovery for various source types, e.g., for relational data (Jensen, Holmgren, & Pedersen, 2004), for XML data (Jensen, Møller, & Pedersen, 2001), and for ontology-based data (Romero & Abelló, 2010); however, there is a large potential in further developing cube discovery techniques to address more of the data sources found on the Web and in particular to incorporate collaborative aspects (see Section COLLABORATIVE ASPECTS for details). In any case, discovery results in a (possibly virtual) cube and its associated multidimensional schema.

SOURCE DISCOVERY AND SELECTION

In traditional BI the relevant data sources (mostly relational and legacy databases) are known at design-time, so standard ETL techniques can be applied to these sources to cleanse and integrate data for populating multidimensional cubes. Conversely, in self-service BI other (heterogeneous and possibly unstructured) data sources must be included in the decision process; these sources cannot be integrated at design-time since (i) by definition, situational queries cannot be anticipated, and (ii) most useful data sources (such as blogs and social networks, RDF data models and linked data) are not controlled by the user and its company, so their features and their availability may change quickly over time.

When a situational query is posed, there is a need for systematically analyzing the different potential sources of information at hand. If an external data source has already been discovered and employed by any users, its features, content, quality, and the ETL processes necessary to load it into fusion cubes are already stored in the catalog. The specific data required may be cached in a fusion cube, so they can either be directly accessed or be reloaded from the source using the ETL procedures in the catalog, depending on the source availability and the desired data freshness. Conversely, if all locally-available data are insufficient, new data sources must be discovered. In our watchdog scenario, keywords "Europe fish capture" could be sent to a search engine to search for potentially-relevant sources such as blogs, RDF data, or linked data; or the Twitter API could be used to discover if the hash tag (#) *#europefishcapture* exists.

A relevant issue related to this phase is that of maintaining an acceptable query response time while simplifying the source discovery and selection process. MapReduce approaches (Pavlo et al., 2009) should be considered for this task, because (i) data processing can be split into different nodes (each for a different source), thus optimizing the query response time, and (ii) sources can be added and removed without severely altering the structure of the whole system. This is relevant in our scenario as we may use a node for exploring possible hash tags in social networks, and then, after data processing, we may conclude that this source does not provide significant data; then that node can be removed without interfering with the nodes in charge of processing the available

data for fish captures. One particular example of this trend is the HIVE platform (Thusoo et al., 2009), based on Hadoop (White, 2010) and MapReduce, that provides analytical capabilities over heterogeneous sources of data.

Quality Assessment

One of the most widely accepted definitions of data quality is based on Deming's "fitness for use" concept (Tayi & Ballou, 1998): A user can only assess the level of quality of a set of data for a particular task to be executed in a specific context, according to a set of criteria, thus determining whether or not these data can be used for that purpose (Strong, Lee, & Wang, 1997). Therefore, when assessing the data quality of the discovered sources, the focus should be on analyzing how current data quality negatively affects the subsequent data analysis. A crucial problem here is to determine which data quality criteria should be considered for selecting situational data according to the use given to them. Even more, when dealing with both internal and external data sources, we should be able to determine a satisfactory trade-off between the quantity and the quality of the retrieved data. Thus, users should be allowed to specify some quality criteria to be used for ranking the results, such as:

- **Relevance to the keywords:** Standard criteria from information retrieval could be used here, taking both data and their metadata, e.g., tags, into account;
- **Integrability:** The overall quality of the results is higher if the situational data share some identifiers with the internal cubes or if a corresponding transcoding function is available. If such a transcoding is not available a semi-automatic technique should be adopted (Po & Sorrentino, 2011);
- **Owner rating:** Source owners themselves often rate their data in terms of completeness, correctness, and freshness;
- **User rating:** Collaborative filtering (Herlocker et al., 2004) or recommendation approaches (Hillet al., 1995) can be used to exploit the unbiased ratings provided by users, that previously acquired that data.

Recalling our running example, when the watchdog group acquires the observations of fish species in protected marine areas, *credibility* of sources must be considered for selection purposes. Also, depending on the kind of required data analysis, different levels of *accuracy* can be allowed: e.g., the watchdog group may accept lower data accuracy if data analysis consists of a map visualization for quick and representative results, but writing a sound denunciation would require discovering and selecting sources for acquiring very accurate data. In this direction, Espinosa, Zubcoff, and Mazón (2011) propose a set of experiments to systematically analyze the behavior of different data quality criteria on the data sources and how they affect the results of a data mining process.

DATA ACQUISITION

Once a set of promising data sources has been identified, the actual data acquisition processes must be performed. This task is highly dependent on the data source type and poses several challenges. In this section we discuss how situational queries can be reformulated over data sources, highlight some issues that may arise during query execution, and discuss the interactive role of users in the data acquisition process.

Query Reformulation and Processing

We call *query reformulation phase* the process of building queries to retrieve situational data. These queries are built according to the information requirements derived from the queries stated by the user, that were already discussed

in Section QUERY FORMULATION, and the metadata of data sources, stored in the catalog. In our running example, a query is needed to retrieve the measures values in the fishing activities cube for those countries that are candidate to enter the EU.

The most challenging reformulation problem arises when a cube is enriched through drill-beyond. Through the metadata (stored in the catalog) of the data sources to be used, the user query must be properly reformulated against the schemata of these data sources. To effectively represent source schemata and establish expressive mappings between these schemata and those of the available cubes, ontologies or controlled vocabularies may be used. The work by Golfarelli et al. (2012) provides an example of a mapping language to support query reformulation across multidimensional cubes.

Reformulated queries have to be executed against the selected data sources. Data access methods are heterogeneous; from this point of view data sources can be distinguished into those that provide a querying mechanism that allows specific items to be selected from the data source, and those that only allow a bulk download of all the data in the source. In the first category we find, for example, *SPARQL endpoints* (Prud'hommeaux & Seaborne, 2008) that allow SPARQL queries to be executed over RDF data to retrieve specific RDF triples, and Web APIs that enable predefined queries providing results in different formats like JSON or XML. In the second category we find, for instance, published RDF documents that are not available through SPARQL endpoints or csv files for download.

Source metadata, in particular machine-processable descriptions of available data, are crucial in this context, not only to reformulate queries but also to interpret the results obtained. For instance, RDF data are inherently rich in semantics, especially when classes and predicates from well-known vocabularies such as *Friend Of A Friend*, *Simple Knowledge Organization System*, or *Schema.Org* are used. On the other hand, the descriptions of data available through Web APIs are usually given as plain text, making automatic processing of results harder.

Finally, it is important to consider that external data sources may impose restrictions on query execution (e.g., timeouts to minimize resources blocking or billing policies based on the amount of queries per day as in most Web APIs). These restrictions should guide the data finder in the query formulation phase and may lead, for instance, to splitting big queries into smaller ones or designing caching mechanisms to avoid redundant calls to pay-per-use data sources.

Is This Good Enough?

Usually, information retrieval techniques are evaluated in terms of two measures: *precision*, which represents the percentage of retrieved results that are indeed relevant for the task at hand, and *recall*, which represents the percentage of total relevant results that were retrieved (Baeza-Yates & Ribeiro-Neto, 2010). There is a well-known trade-off between these two measures: Recall must be sacrificed to improve precision, and vice versa. In building fusion cubes we anticipate a similar problem, but user interaction can help to deal with this. Here is a possible scenario:

1. Initially, the system reformulates the user query, prioritizing recall against precision (get as much results as it can) and performs queries over several candidate data sources;
2. Once situational data are retrieved and returned, the user may decide that they are not suitable for the task at hand (e.g., because incomplete or unrelated to the original query). The user may then, for example, remove a source from the set of candidate data sources, resolve conflicts that may arise from retrieving data from multiple sources, or state that the retrieved data is not enough;

3. The actions performed by the user in Step 2 may lead to new queries that retrieve more data, perhaps more precise than those initially retrieved.

Another key factor here is query response time. A user may decide to sacrifice precision to obtain quicker results. If retrieved data are not satisfactory then she may ask the system to retrieve more results, instead of initially spending too much time to get results that exceed her expectations.

DATA INTEGRATION

Once situational data are selected and acquired, they have to be combined with stationary data to feed fusion cubes. One of the challenges here is to define mechanisms to integrate heterogeneous data sources in a self-service, on-demand fashion. To this end we suggest the following steps:

1. *Extract* structure from different situational data sources;
2. *Map* both schemata and instances of data sources;
3. *Reconcile* them with stationary data to obtain a fusion cube.

This process should be "quality-aware" in the sense that the user can always give feedback on the integration process to improve it. Once heterogeneous data are fused, they should be properly stored to be reused and shared when required.

Structure Extractors

Huge amounts of data on the Web are "structurable" and are good candidates for integration in a fusion cube. Good examples of structurable data are *Google Fusion Tables* (Gonzales et al., 2010), *WebTables* (Cafarella et al., 2008) and wrappers from the *Lixto* project (Gottlob, 2005). There are other works that focus on extracting relations from apparently unstructured data sources (Bozzon et al., 2010; Löser et al., 2011). We refer readers to Laender et al. (2002) for a survey on Web data extractors.

Importantly, these approaches focus on discovering and extracting structured information assets from the Web into a plain tabular format. However, this is not enough in our context, since the fusion cubes have a multidimensional structure, which allows OLAP access in a way that comes more natural to human analysts. Therefore, the concepts extracted from situational data must be annotated as measures, dimensions, or hierarchy levels. To do so, domain constraints or heuristics derived from the multidimensional model should be used. For instance, in our running example, observations of one species in protected marine areas may appear in a table published in a research paper in which the quantity of observations is a measure represented in each cell of the table and the dimension is the specific marine area determined in the header of the table. Also the semantic knowledge of the specific application domain has a crucial role for annotating concepts. For example, "fish population data from marine protected areas" is related to some statistical values that are likely to be related to a fact or its measures (due to the semantics of the population concept), while "protected marine area" could be annotated as a dimension due to its geographical meaning.

Schema and Instance Mapping

In the mapping step, schemata from both situational data sources and fusion cubes are mapped. Schema and instance mapping approaches are based on matching those elements of two different schemata that semantically correspond (Rahm & Bernstein, 2001). Although schema and instance mapping have been traditionally configured and performed in a manual manner, fusion cubes require their automation.

Interestingly, full integration is not required by fusion cubes, since data sources may partially represent useful situational data. In our scenario, "fish population data in a marine protected area" can be acquired from a table in a research

paper. Also, this table can contain useful data, e.g., fish size or weight. Therefore, the fusion cube scenario can be seen as a pay-as-you-go approach to integration in the sense of (Sarma, Dong, & Halevy, 2008) in which set of initial mappings are improved over time.

As in traditional data integration, schema elements must be matched based on clues in the schema and data, such as element names, types, data values, schema structures, and integrity constraints (Doan & Halevy, 2005); semantic knowledge from the domain (e.g., domain ontologies) should also be used.

To increase the level of automation, situational data sources can also take advantage of the previous annotations and mappings related to multidimensional schemata, like done by Cabanac et al. (2007). By using these annotations, several mappings between the situational data sources and fusion cubes can be derived. For example, the multidimensional normal forms proposed by Lechtenbörger & Vossen (2003) can be useful for matching purposes as they allow checking that every fact is based on a set of dimensions that determine the granularity adopted for representing the fact measures, and that dimensions are organized as hierarchies of levels by means of functional dependencies.

Reconciling Stationary and Situational Data

Apart from the important task of cleaning data (duplicate removal and conflict resolution), stationary and situational data needs to be reconciled in order to standardize and format data, i.e., data scrubbing. Recalling our running example, when the watchdog group compares volume of catches with the observations of the same species, units used in measurements should be homogenized (e.g., weight or number of individuals). A well-known tool used for carrying out these tasks is *Google Refine* (code.google.com/p/google-refine/), which provides functionalities for working with messy data, cleaning it up, transforming it from one format into another, extending it with web services, and linking it to databases.

Storing Situational Data

Once situational data have been integrated, they can be materialized in a fusion cube with the aim of being reused when required. To facilitate their combination with other cubes, they must be semantically annotated thus allowing users to explore new possibilities of analysis by composing situational sources in new cubes.

Our idea for materializing fusion cubes is to use an RDFS vocabulary similar to the one presented by Cyganiak, Reynolds, & Tennison (2012). By using RDF we can take advantage of the existing semantic Web technologies that make the publication, exchange, and querying of RDF data simpler. In particular, Etcheverry & Vaisman (2012) propose to define both schema and instances of fusion cubes as RDF graphs describing the multidimensional model (i.e., dimension instances are sets of roll-up functions, and fact instances are points in a multidimensional space, as proposed by Cabibbo & Torlone, 1997). This large volume of RDF data can partly be stored and queried efficiently using specialized RDF warehousing engines, so-called *triple-stores* (Liu, Thomsen & Pedersen, 2011). However, these triple-stores need to be extended to better support typical OLAP queries, the exact way of doing this depends on whether the triple-store is based on an RDBMS (Liu, Thomsen, & Pedersen, 2011) or not (Neumann & Weikum, 2008). Since fusion cubes can be seen as an extension to traditional cubes, we also need a mechanism provide interfaces for specifying this extension: the drill-beyond operation as explained in Section QUERY FORMULATION. In this sense we envision an extension of the drill-across OLAP operator to navigate between fusion cubes and cubes, and vice versa.

Interestingly, fusion cubes resemble other "composable" artifacts, such as *Web mashups* (Yu et al., 2008), which are Web applications generated by combining content, presentation, or application functionality from disparate Web sources. Indeed, both Web mashups and fusion cubes aim to combine Web applications and situational data, respectively, in a value-adding manner.

Is This Good Enough?

During the integration step, a user can interact with the system to accept, reject, or change an existing mapping, or even to propose a new one. To facilitate this task, the proposed mappings must be ranked based on some similarity measure that evaluates to what extent two elements from different schemata represent the same concept. On the one hand, this measure should be related to the cube schema so that two elements can be compared based on their annotations: A mapping that matches dimension *d* with hierarchy level *h* is better than a mapping that matches *d* with measure *m*. On the other hand, data quality measures should take semantics into account: Although the strings "observations" and "catches" are lexically different, they are likely to be mapped, since they both refer to a finite collection of items (either watched or captured).

CUBE PRESENTATION

Data visualization is "a form of knowledge compression: it is a way of squeezing an enormous amount of information and understanding into a small space" (McCandless, 2010). Visualization techniques in BI are typically used to highlight outstanding facts, tendencies, patterns, and relationships; data are presented to users mainly via traditional metaphors (tabular lists, pie-charts, etc.) with an extra touch of interaction, such as interactive dashboards. However, fusion cubes have an inherent feature that traditional cubes do not have: While cubes are typically considered to be the cleanest (error-free) form of data representation, fusion cubes combine a quality-assured part, referring to the data coming from the DW, with a variable-quality part including the situational data acquired under the user's guidance.

Is This Good Enough?

Since the fused data must be approved by the user, the ultimate "is this good enough?" question is raised whenever a user is about to integrate situational data within her report and use them for further analysis. *Quality filtering* for incoming data becomes a key ingredient of self-service BI as users must be able to select data based on quality-related characterizations (either of the data themselves or their providers). These quality characterizations can cover any of the quality criteria that were mentioned in previous steps, such as accuracy, freshness, completeness, reliability, or any other criterion fitting under the "fitness for use" (Tayi & Ballou, 1998) umbrella. A catalogue with user-generated (alternatively, crowd-wisdom; in extremis, automated) characterizations for the quality of incoming data would probably help for this task. These quality characterizations for incoming data can be shown to users in an intuitive way and in sync with the rest of the representation. As typically happens in data visualization, the usage of the *Gestalt visualization principles* for highlighting properties of data can be exploited to this end. For example, unreliable data can be highlighted in red colors (indicating danger), or possibly, out-of-date data can be depicted in smaller font size, and so on (Tidwell, 2006).

REUSE AND SHARING

A main characteristic of Web 2.0 is the massive surge of online social collaboration platforms, which utilize and emphasize the fact that groups have greater combined knowledge and problem-solving ability than any single individual. This ability to metaphorically "stand on the shoulders of each other" can be used to great advantage within self-service BI: "star ratings" for data sources, collaborative filtering for suggesting

relevant data sources or analyses, and data sources shared in libraries and augmented with annotations.

A few systems offer basic functionality in this direction. For instance, *Google fusion tables* (Gonzalez et al., 2010) offers cloud-based storage of basic relational tables that can be shared with others, annotated, fused with other data with standard operators, and finally visualized. The *illo* system (illo.com) allows non-business users to store small cubes in a cloud-based environment, analyze the data using powerful mechanisms, and provide visual analysis results that can be shared with others and annotated. However, there is further potential for generalizing and extending the collaborative aspects of BI, taking into account more powerful data semantics and advanced types of collaboration. In the following we discuss collaborative aspects for each of the phases described in the previous sections:

- **Collaborative query formulation:** At this stage the user is formulating a query against an existing cube, extended with a keyword-based search for situational data. The system can take advantage of the previous actions of other users in the same or similar cases, e.g., suggesting "popular queries" related to the data or Amazon-like "users who looked at this dataset asked these queries." A more long-term approach is to publish the data for commenting, and getting the user community to give suggestions for what to do with specific data. Finally, collaborative cube discovery is also possible: the decisions of other users on how some (non-cube) source data has been translated into measures and dimensions with hierarchies can provide valuable guidance for subsequent users;
- **Collaborative source discovery:** Here, the issue is not only to find data, but even more to find relevant data of high quality. Tags provided by other users can help determine the exact content of sources and "star ratings" of data sources can help in choosing the source with the best quality. Ratings can also more generally be used to provide recommendations for data sources;
- **Collaborative data acquisition:** Once sources are discovered, the actual data must be acquired. An interesting aspect here is what to do if the desired data are not (yet) available online. The community can be asked for data; concretely, an open, shared, cloud-based database, such as Freebase (www.freebase.com), could be established where others can upload desired data to be used by the whole community, i.e., an extension and generalization of the concept of *crowd-sourcing* as recently suggested by Franklin et al. (2011);
- **Collaborative data integration:** The data integration task is often the most difficult in BI projects, and can benefit the most from collaboration. For example, a user might want to add a location dimension to the fusion cube. Building it from scratch, filling it with data, and specifying all relevant hierarchies is a daunting task, but it is quite likely that others have already accomplished (large) parts of it, and subsequently shared them. Ten relevant location dimensions could be available; which one to pick depends on the best fit to your data (e.g., coverage of the whole world and not just Europe) in combination with good "star ratings" and additional "ground truth" provided by collaborative ontologies or even Wikipedia. Similarly, (definitions of) measures can be shared. Another traditionally hard task is data cleansing; here, procedures for specific data (such as de-duplication of names and places) can be shared and reused;
- **Collaborative data presentation:** Query result presentation can also benefit from collaboration, building on existing works within the so-called *meta-morphing* for BI analytical frontends (Middelfart & Pedersen, 2011) that combine knowledge about the multidimensional schema with knowledge about the historical behavior of

the current and other users. For example, based on the previous choices of other users, the system may learn which types of charts are relevant (popular) for what data. However, the existing work must be significantly extended and generalized to cope with the much more diverse data and user universes imagined for our scenario.

DEPLOYMENT

We now describe how we envision a deployment of fusion cubes, thereby indicating that our vision can be realized without extraordinary effort. Indeed, there are already various architectures that do similar things. For instance, Google Squared (which is now discontinued) has combined information retrieval, text extraction, and query processing. The Stratosphere project (www.stratosphere.eu), which explores the power of massively parallel computing for complex information management applications, tries to combine all those components relevant to fusion cubes at query time. Another relevant project is Asterix at UC Irvine (asterix.ics.uci.edu), a scalable parallel platform for semi-structured data management described by Alsubaiee et al. (2012). There are essentially two extreme alternatives for deploying our system, namely *tightly-coupled* and *service-based,* which are described next.

Tightly Coupled Deployment

The first implementation option is a tightly coupled, in-house architecture. In this case, all components are deployed locally and connected together. This is straightforward for the right-hand side components of Figure 2 (the "inside"), while it is challenging for the left-hand side components. To gain some control over them, external registries as well as ontologies can periodically be crawled, looking for potentially interesting nuggets of information, changes, or additions. When something is found, an (incremental) ETL process is triggered to bring these data into the fusion cubes through quality evaluation, cleansing, and transformation stages. The corresponding metadata are added to the catalog, making the information available to users.

Besides making on-line querying feasible, the main advantage of this kind of implementation is that control is gained not only over availability, but also over the quality of external data. On the one hand, it can be measured, while on the other it can be improved upon (possibly off-line). From a user's point of view, the internal catalog contains more metadata than that in the original registry, and since external data are not under our control (i.e., they can change, or even disappear), by taking a snapshot non-volatility can be guaranteed. Moreover, extraction processes can be reused, resulting in optimizing the usage of resources (i.e., bandwidth to bring data from the original source, as well as CPU consumed by the ETL component). Efficiency is also improved by the coupling mechanisms between components (either internal or external) because, being ad-hoc, they can implement specific protocols instead of generic ones.

On the disadvantage side, dedicated storage devices are needed to store the snapshots taken from the external sources. Since we assume that user needs are quite unique and situational, knowing them in advance is unlikely, resulting also in a certain waste of resources. However, as shown in Figure 2, our architecture provides for this. Alternatively, we could bring only the metadata and location of the external source into the catalog. This would save not only disk space, but also the difficulty of dealing with updates in the external data (which now become volatile). In this case, the ETL process is always triggered at query time (either over the original source or its local copy), but applying relatively complex line cleansing algorithms is unrealistic.

Service-Based Deployment

At the other extreme, deployment can also be based on loosely-coupled services. In this case, provisioning anything in advance is not needed, which is quite appropriate in this environment because of the unpredictable workload

and resources consumed by analyses, and we dynamically find a provider for the data, software, and infrastructure we need. This would probably impact response time, but will give users a chance to do without mediation or feedback loops:

- Consuming *data as a service* is a must in our envisioned system, since answering situational queries requires finding a provider for the data not available internally. Having a true registry (not only a search engine) of potential data sources and some Web service interface would be the most appropriate choice to be able to automate the process to some extent (AbuJarour, Naumann, & Craculeac, 2010). For example, it is not possible to access data in the *deep Web* with simple search engines; in this case, the provider should register the source and describe its contents, so that it can be easily found;
- Consuming *software as a service* (SaaS) provides location independence and dynamic discovery not only of the above mentioned external registries, but also of data cleansing tools or other services that could help users to find, choose, or transform the data. Deploying the system by means of SaaS would also make its management and evolution more flexible. Another important characteristic we would obtain is a broad network access to all components. Finally, a business opportunity could be generated by offering in-house components to external consumers. On the other hand, working with a fixed set of external providers would be advisable for performance reasons.

CONCLUSION AND BEGINNINGS

During the past decade, the acceptance of information technology has arguably been faster than ever. Resources that were only available to large organizations are now offered to anyone online as software (SaaS). Simultaneously, the Web keeps on growing as a tremendous source of information. In this context, end-users should be able to integrate both private and situational data and analyze them without any direct help, a process we term self-service BI. We even envision a time when users will be able to do this from a small pocket-size device.

In this paper, we presented our vision of what self-service BI shall be in the near future, which challenges it poses, and what can be done to meet these. We detailed how the classical data warehousing process shall be complemented with search engine-like querying (the drill-beyond OLAP operator), external data source discovery and selection, and on-demand ETL and data storage. This process shall be as transparent and accessible to the average user as possible, and a collaborative aspect can be present in each of its steps. We also outlined an architecture, centered on the idea of fusion cubes, to support our vision.

Although some technologies already go part of the way toward self-service BI, such as applications that make the exploitation of structured data very easy or cloud BI solutions, most of them are still aimed at BI specialists and many research challenges remain:

- It will be interesting to investigate to what degree more advanced data model and query algebra, with associated query processing techniques, can empower the concept of fusion cubes. A more advanced data model should mix the superior analysis capabilities of multidimensional data models with the flexibility and generality of semi-structured data models, while borrowing useful concepts from the semantic Web. It will also be interesting to develop a cloud-based fusion cube deployment infrastructure with support for these features, which, when customized to meet the high performance requirements of certain application scenarios, constrains the suitability of some hardware platforms and the efficiency of algorithms;

- As in most cloud-based applications, various security and privacy aspects must also be dealt with, including a simple and easy-to-use mechanism for balancing data privacy and data sharing in fusion cubes that can act as a privacy shield for non-technical users when dealing with sensitive data. Some users or members of a group of users may indeed want to provide data for a given analysis and share the results, without necessarily disclosing source data publicly or even to other group members. Web data availability on the long run, the assessment of data and/or data sources, and digital rights management issues must also be dealt with;
- Collaborative intelligence features require a formal foundation and associated techniques; many features will have to be seamlessly hidden by user interfaces, e.g., ETL, OLAP and linguistic tools, to make self-service BI usable by non-specialists;
- Finally, from a financial point of view, we need a novel pay-as-you-go payment model for fusion cubes that allows the traditionally huge startup costs for a BI system to be amortized both over the users sharing data, and over time.

ACKNOWLEDGMENT

This article was conceived during Dagstuhl Seminar 11361 “Data Warehousing: From Occasional OLAP to Real-time Business Intelligence”.

REFERENCES

AbuJarour, M., Naumann, F., & Craculeac, M. (2010). Collecting, annotating, and classifying public Web services. In *Proceedings International Conference on Cooperative Information Systems*, Hersonissos, Greece (pp. 256-272).

Agrawal, D., Das, S., & Abbadi, A. E. (2011). Big data and cloud computing: Current state and future opportunities. In *Proceedings International Conference on Extending Database Technology*, Uppsala, Sweden (pp. 530-533).

Aligon, J., Golfarelli, M., Marcel, P., Rizzi, S., & Turricchia, E. (2011). Mining preferences from OLAP query logs for proactive personalization. In Proceedings Advances in Databases and Information Systems, Vienna, Austria (pp. 84-97).

Alsubaiee, S., Behm, A., Grover, R., Vernica, R., Borkar, V., Carey, M. J., & Li, C. (2012). ASTERIX: Scalable warehouse-style web data integration. In *Proceedings International Workshop on Information Integration on the Web,* Phoenix, AZ.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. G. (2007). DBpedia: A nucleus for a web of open data. In *Proceedings International Semantic Web Conference*, Boston, MA (pp. 722-735).

Baeza-Yates, R. A., & Ribeiro-Neto, B. A. (2010). *Modern information retrieval* (2nd ed.). Reading, MA: Addison-Wesley.

Berthold, H., Rosch, P., Zoller, S., Wortmann, F., Carenini, A., & Campbell, S. … Strohmaier, F. (2010). An architecture for ad-hoc and collaborative business intelligence. In *Proceedings of EDBT/ICDT Workshops*, Lausanne, Switzerland.

Bizer, C., Heath, T., & Berners-Lee, T. (2009). Linked data - The story so far. *International Journal on Semantic Web and Information Systems*, *5*(3), 1–22. doi:10.4018/jswis.2009081901.

Boussaid, O., Darmont, J., Bentayeb, F., & Rabaséda, S. L. (2008). Warehousing complex data from the web. *International Journal on Web Engineering Technology*, *4*(4), 408–433. doi:10.1504/IJWET.2008.019942.

Bozzon, A., Brambilla, M., Ceri, S., & Fraternali, P. (2010). Liquid query: multi-domain exploratory search on the web. In *Proceedings International World Wide Web Conference*, Raleigh, NC (pp. 161-170).

Cabanac, G., Chevalier, M., Ravat, F., & Teste, O. (2007). An annotation management system for multidimensional databases. In *Proceedings International Conference on Data Warehousing and Knowledge Discovery*, Regensburg, Germany (pp.89-98).

Cabibbo, L., & Torlone, R. (1997). Querying multidimensional databases. In *Proceedings International Workshop on Database Programming Languages*, Estes Park, Colorado (pp. 319-335).

Cafarella, M. J., Halevy, A. Y., Wang, D. Z., Wu, E., & Zhang, Y. (2008). WebTables: Exploring the power of tables on the web. *Proceedings VLDB Endowment*, *1*(1), 538–549.

Cohen, J., Dolan, B., Dunlap, M., Hellerstein, J. M., & Welton, C. (2009). Mad skills: New analysis practices for big data. *Proceedings VLDB Endowment*, *2*(2), 1481–1492.

Cyganiak, R., Reynolds, D., & Tennison, J. (2012). *The RDF data cube vocabulary*. Retrieved from http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html

Darmont, J., Boussaid, O., Ralaivao, J. C., & Aouiche, K. (2005). An architecture framework for complex data warehouses. In *Proceedings International Conference on Enterprise Information Systems*, Miami, FL (pp. 370-373).

Doan, A., & Halevy, A. Y. (2005). Semantic integration research in the database community: A brief survey. *AI Magazine*, *26*(1), 83–94.

Espinosa, R., Zubcoff, J. J., & Mazón, J.-N. (2011). A set of experiments to consider data quality criteria in classification techniques for data mining. In *Proceedings International Conference on Computational Science and its Applications* (pp. 680-694).

Essaidi, M. (2010). ODBIS: Towards a platform for on-demand business intelligence services. In *Proceedings EDBT/ICDT Workshops*, Lausanne, Switzerland.

Etcheverry, L., & Vaisman, A. (2012). Enhancing OLAP analysis with web cubes. In *Proceedings Extended Semantic Web Conference*, Heraklion, Crete (pp. 469-483).

Ferrández, A., & Peral, J. (2010). The benefits of the interaction between data warehouses and question answering. In *Proceedings EDBT/ICDT Workshops*, Lausanne, Switzerland.

Franklin, M. J., Kossmann, D., Kraska, T., Ramesh, S., & Xin, R. (2011). CrowdDB: Answering queries with crowdsourcing. In *Proceedings ACM International Conference on Management of Data* (pp. 61-72).

Golfarelli, M., Mandreoli, F., Penzo, W., Rizzi, S., & Turricchia, E. (2012). OLAP query reformulation in peer-to-peer data warehousing. *Information Systems*, *37*(5), 393–411. doi:10.1016/j.is.2011.06.003.

Golfarelli, M., & Rizzi, S. (2009). *Data warehouse design: Modern principles and methodologies*. McGraw-Hill.

Golfarelli, M., Rizzi, S., & Cella, I. (2004). Beyond data warehousing: What's next in business intelligence? In *Proceedings International Workshop on Data Warehousing and OLAP*, Washington, DC (pp. 1-6).

Gonzalez, H., Halevy, A. Y., Jensen, C. S., Langen, A., Madhavan, J., Shapley, R., et al. (2010). Google fusion tables: Web-centered data management and collaboration. In *Proceedings ACM International Conference on Management of Data* (pp. 1061-1066).

Gottlob, G. (2005). Web data extraction for business intelligence: The Lixto approach. In *Proceedings Conference Datenbanksysteme in Business, Technologie und Web Technik* (pp. 30-47).

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, *22*(1), 5–53. doi:10.1145/963770.963772.

Hill, W. C., Stead, L., Rosenstein, M., & Furnas, G. W. (1995). Recommending and evaluating choices in a virtual community of use. In *Proceedings Conference on Human Factors in Computing Systems* (pp. 194-201).

Horwitt, E. (2010). *Self-service BI catches on*. Retrieved from http://www.computerworld.com/

Jensen, M. R., Holmgren, T., & Pedersen, T. B. (2004). Discovering multidimensional structure in relational data. In *Proceedings International Conference on Data Warehousing and Knowledge Discovery* (pp. 138-148).

Jensen, M. R., Møller, T. H., & Pedersen, T. B. (2001). Specifying OLAP cubes on XML data. *Journal of Intelligent Information Systems*, *17*(23), 255–280. doi:10.1023/A:1012814015209.

Jerbi, H., Ravat, F., Teste, O., & Zurfluh, G. (2009). Applying recommendation technology in OLAP systems. In *Proceedings International Conference on Enterprise Information Systems* (pp. 220-233).

Jörg, T., & Dessloch, S. (2009). Near real-time data warehousing using state-of-the-art ETL tools. In *Proceedings International Workshop on Business Intelligence for the Real-Time Enterprise* (pp. 100-117).

Laender, A. H. F., Ribeiro-Neto, B. A., da Silva, A. S., & Teixeira, J. S. (2002). A brief survey of Web data extraction tools. *SIGMOD Record*, *31*(2), 84–93. doi:10.1145/565117.565137.

Lechtenbörger, J., & Vossen, G. (2003). Multidimensional normal forms for data warehouse design. *Information Systems*, *28*(5), 415–434. doi:10.1016/S0306-4379(02)00024-8.

Liu, X., Thomsen, C., & Pedersen, T. B. (2011). 3XL: Supporting efficient operations on very large OWL Lite triple-stores. *Information Systems*, *36*(4), 765–781. doi:10.1016/j.is.2010.12.001.

Löser, A., Hueske, F., & Markl, V. (2008). Situational business intelligence. In Proceedings Business Intelligence for the Real-Time Enterprise (pp. 1-11).

Löser, A., Nagel, C., Pieper, S., & Boden, C. (2011). Self-supervised Web search for any-k complete tuples. In *Proceedings International Workshop on Business intelligencE and the WEB* (pp. 4-11).

Mate, A., & Trujillo, J. (2012). A trace metamodel proposal based on the model driven architecture framework for the traceability of user requirements in data warehouses. *Information Systems*, *37*(8), 753–766. doi:10.1016/j.is.2012.05.003.

McCandless, D. (2010). *The beauty of data visualization*. TED Conference Series. Available from http://www.ted.com/talks/david_mccandless_the_beauty_of_data_visualization.html

Middelfart, M., & Pedersen, T. B. (2011). The meta-morphing model used in TARGIT BI suite. In *Proceedings ER Workshops*, Brussels, Belgium (pp. 364-370).

Nebot, V., Llavori, R. B., Pérez-Martínez, J. M., Aramburu, M. J., & Pedersen, T. B. (2009). Multidimensional integrated ontologies: A framework for designing semantic data warehouses. *Journal on Data Semantics*, *13*, 1–36. doi:10.1007/978-3-642-03098-7_1.

Neumann, T., & Weikum, G. (2008). RDF-3X: a RISC-style engine for RDF. *Proceedings of the VLDB Endowment*, *1*(1), 647–659.

Niinimäki, M., & Niemi, T. (2009). An ETL process for OLAP using RDF/OWL ontologies. *Journal on Data Semantics*, *13*, 97–119. doi:10.1007/978-3-642-03098-7_4.

Pavlo, A., Paulson, E., Rasin, A., Abadi, D. J., DeWitt, D. J., Madden, S., & Stonebraker, M. (2009). A comparison of approaches to large-scale data analysis. In *Proceedings ACM International Conference on Management of Data* (pp. 165-178).

Pedersen, D., Pedersen, T. B., & Riis, K. (2004). The decoration operator: A foundation for on-line dimensional data integration. In *Proceedings International Database Engineering and Applications Symposium* (pp. 357-366).

Pedersen, T. B., Pedersen, D., & Pedersen, J. (2008). Integrating XML data in the TARGIT OLAP system. *International Journal of Web Engineering and Technology*, *4*(4), 495–533. doi:10.1504/IJWET.2008.019945.

Perez, J. M., Berlanga, R., Aramburu, M. J., & Pedersen, T. B. (2008). Towards a data warehouse contextualized with web opinions. In *Proceedings IEEE International Conference on e-Business Engineering*, Macau, China (p. 697-702).

Pérez, J. M., Llavori, R. B., Aramburu, M. J., & Pedersen, T. B. (2008). Integrating data warehouses with Web data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, *20*(7), 940–955. doi:10.1109/TKDE.2007.190746.

Po, L., & Sorrentino, S. (2011). Automatic generation of probabilistic relationships for improving schema matching. *Information Systems*, *36*, 192–208. doi:10.1016/j.is.2010.09.004.

Prud'hommeaux, E., & Seaborne, A. (2008). *SPARQL query language for RDF*. Retrieved from http://www.w3.org/TR/rdf-sparql-query/

Rahm, E., & Bernstein, P. A. (2001). A survey of approaches to automatic schema matching. *The VLDB Journal*, *10*(4), 334–350. doi:10.1007/s007780100057.

Romero, O., & Abelló, A. (2010). A framework for multidimensional design of data warehouses from ontologies. *Data & Knowledge Engineering*, *69*(11), 1138–1157. doi:10.1016/j.datak.2010.07.007.

Sarma, A. D., Dong, X., & Halevy, A. Y. (2008). Bootstrapping pay-as-you-go data integration systems. In *Proceedings ACM International Conference on Management of Data* (pp. 861-874).

Strong, D. M., Lee, Y. W., & Wang, R. Y. (1997). Data quality in context. *Communications of the ACM*, *40*(5), 103–110. doi:10.1145/253769.253804.

Tayi, G. K., & Ballou, D. P. (1998). Examining data quality - Introduction. *Communications of the ACM*, *41*(2), 54–57. doi:10.1145/269012.269021.

Thiele, M., & Lehner, W. (2011). Real-time BI and situational analysis. In M. Zorrilla, et al. (Eds.), *Business intelligence applications and the Web: Models, systems and technologies* (pp. 285–309). Hershey, PA: IGI Global. doi:10.4018/978-1-61350-038-5.ch013.

Thusoo, A., Sarma, J. S., Jain, N., Shao, Z., Chakka, P., & Anthony, S. … Murthy, R. (2009). HIVE – A warehousing solution over a map-reduce framework. In *Proceedings International Conference on Very Large Data Bases* (pp. 24-28).

Tidwell, J. (2006). *Designing interfaces - patterns for effective interaction design*. Sebastopol, CA: O'Reilly.

Trujillo, J., & Mate, A. (2011). *Business intelligence 2.0: A general overview*. Retrieved from http://cs.ulb.ac.be/conferences/ebiss2011/

White, T. (2010). *Hadoop: The definitive guide*. Yahoo Press.

Xu, K., Liao, S. S., Li, J., & Song, Y. (2011). Mining comparative opinions from customer reviews for competitive intelligence. *Decision Support Systems*, *50*(4), 743–754. doi:10.1016/j.dss.2010.08.021.

Yu, J., Benatallah, B., Casati, F., & Daniel, F. (2008). Understanding mashup development. *IEEE Internet Computing*, *12*(5), 44–52. doi:10.1109/MIC.2008.114.

M. E. Zorrilla, J.-N. Mazon, O. Ferrandez, I. Garrigos, F. Daniel, & J. Trujillo (Eds.). (2011). *Business intelligence applications and the Web: Models, systems and technologies*. Hershey, PA: IGI Global. doi:10.4018/978-1-61350-038-5.

Alberto Abelló is tenure-track 2 professor at Universitat Politècnica de Catalunya (Polytechnical University of Catalonia). He received his PhD from UPC in 2002. He is also a member of the MPI research group (Grup de recerca en Modelització i Processament de la Informació) at the same university, specializing in software engineering, databases and information systems. His research interests are databases, database design, data warehousing, OLAP tools, ontologies and reasoning. He is the author of articles and papers presented and published in national and international conferences and journals on these subjects (e.g., Information Systems, DKE, IJDWM, KAIS, CAiSE, DEXA, DaWaK and DOLAP). He has also served in the international program committees (e.g., DaWaK and DOLAP), being chair of DOLAP and MEDI. Currently, he is the local coordinator at UPC of the Erasmus Mundus Joint Master IT4BI and Erasmus Mundus Joint Doctorate IT4BI-DC.

Jérôme Darmont is a full professor of computer science at the University of Lyon 2, France, and the director of the ERIC laboratory. He received his PhD in 1999 from the University of Clermont-Ferrand II, France, and then joined the University of Lyon 2 as an associate professor. He became full professor in 2008. His research interests mainly relate to database and data warehouse performance (auto-optimization, benchmarking...) and security (data privacy and availability), especially in the context of cloud business intelligence. He has published more than 90 peer-reviewed papers. He is a member of several editorial boards and has served as a reviewer for numerous conferences and journals. Along with Torben Bach Pedersen, he initiated the Cloud Intelligence workshop series in 2012.

Lorena Etcheverry is a PhD candidate at the Computer Science Institute, Universidad de la República, Uruguay, under the supervision of Alejandro Vaisman. She received her MSc in Computer Science from the same University in 2010. She is a teaching and research assistant, and a member of the Information Systems group since 2003. As such, she has been involved in several undergraduate courses, in particular Databases Fundamentals and Data Warehousing Systems Design. Her PhD research is focused on designing a representation of multidimensional data using semantic web technologies, in particular RDF, that allows not only to publish this kind of data but also to directly perform OLAP operations over it via SPARQL queries.

Matteo Golfarelli received the PhD degree for his work on autonomous agents in 1998. Since 2005, he is an associate professor, teaching information systems, database systems, and data mining. He has published more than 80 papers in refereed journals and international conferences in the fields of pattern recognition, mobile robotics, multiagent systems, and business intelligence that is now his main research field. He coauthored a book on data warehouse design. He is co-chair of the DOLAP12 and of the MiproBIS Conference and member of the editorial board of the International Journal of Data Mining, Modeling, and Management *and of the* International Journal of Knowledge-Based Organizations*. His current research interests include distributed and semantic data warehouse systems, and sentiment analysis.*

Jose-Norberto Mazón is an assistant professor at the Department of Software and Computing Systems in the University of Alicante (Spain). He obtained his PhD in Computer Science from the University of Alicante (Spain). His research interests are: open knowledge and business intelligence, open data, and model driven development. He has published over 70 articles in his field of interest in major national and international journals, conferences and workshops. He has also been co-organizer of the Business Intelligence and the Web (BEWEB) workshop series, and is a member of program committees of several international conferences and workshops.

Felix Naumann studied mathematics, economy, and computer sciences at the University of Technology in Berlin. After receiving his diploma (MA) in 1997 he joined the graduate school "Distributed Information Systems" at Humboldt University of Berlin. He completed his PhD thesis on "Quality-driven Query Answering" in 2000. In 2001 and 2002 he worked at the IBM Almaden Research Center on topics around data integration. From 2003 - 2006 he was assistant professor for information integration at the Humboldt-University of Berlin. Since 2006 he holds the chair for information systems at the Hasso Plattner Institute at the University of Potsdam in Germany.

Torben Bach Pedersen is a full professor of computer science at Aalborg University (AAU), Denmark, where he heads the Center for Data-Intensive Systems. He received his PhD in computer science from AAU in 2000 and his MSc in computer science and mathematics from Aarhus University in 1994. Before joining AAU in 2000, he worked as a business intelligence specialist in industry for six years. His research interests span business intelligence topics such as data warehousing, OLAP, and data mining, with a focus on non-traditional and complex types of data such as web-related data, spatial and spatio-temporal data, music data, energy data, and medical data. He has published more than 100 peer-reviewed papers on these topics. He has served as PC Chair for DaWaK 2009 and 2010 and DOLAP 2010, General Chair for SSTD 2009, and on numerous program committees, including VLDB, ICDE, and EDBT. He is a member of the SSTD Endowment and the DOLAP Steering Committee.

Stefano Rizzi received his PhD in 1996 from the University of Bologna, Italy. Since 2005 he is Full Professor at the University of Bologna, where he is the head of the Data Warehousing Laboratory and teaches Business Intelligence and Software Engineering. He has published more than 100 papers in refereed journals and international conferences mainly in the fields of data warehousing, pattern recognition, and mobile robotics, and a research book on data warehouse design. He joined several research projects on the above areas and has been involved in the PANDA thematic network of the European Union concerning pattern-base management systems. He is member of the steering committee of DOLAP. His current research interests include data warehouse design and business intelligence, in particular multidimensional modeling, OLAP preferences, and collaborative business intelligence.

Juan Trujillo is a Full-time Professor at the Department of Software and Computing Systems in the University of Alicante (Spain) and the leader of the Lucentia Research Group. His main research topics include Business Intelligence applications, Business Intelligence 2.0, Big Data, data warehouses' development, OLAP, data mining, UML, MDA, data warehouses' security and quality. He has also participated in the official registration of different tools related to Data Warehouse modelling. He has advised 10 PhD students and published more than 150 papers in different highly impact conferences such as the ER, UML or CAiSE, and more than 30 papers in highly ranked international journals (JCR) such as the DKE, DSS, ISOFT, IS, or JDM. He has also been co-editor of five special issues in different JCR journals (e.g. DKE). He has also been PC member of different events and JCR journals such as ER, CIKM, ICDE, DOLAP, DSS, JDM, or DKE, and PC Chair of DOLAP'05, DAWAK'05-'06 and FP-UML'05-'09.

Panos Vassiliadis received his Diploma in Electrical Engineering and his PhD from the National Technical University of Athens in 1995 and 2000, respectively. He joined the Department of Computer Science of the University of Ioannina in 2002. Vassiliadis has conducted research in the area of data warehousing (metadata management, OLAP, and quite emphatically, ETL) for long. He has also worked on top-k and context-based querying and web service management. Following a common thread in his work, he is currently investigating how the rigorous modeling of data, software and their interdependence can be exploited for the design, visualization and evolution management of data-centric software ecosystems. Vassiliadis is the co-editor of the book Fundamentals of Data Warehouses *(Springer). He has several publications in international journals and conferences and an active service to the scientific community as a reviewer and PC chair.*

Gottfried Vossen is a Professor of Computer Science in the Department of Information Systems at the University of Muenster in Germany, and an Honorary Professor at the University of Waikato Management School in Hamilton, New Zealand. He received his master's and PhD degrees as well as the German habilitation in 1981, 1986, and 1990, resp., all from the Technical University of Aachen in Germany. He is the European Editor-in-Chief of Elsevier's Information Systems - An International Journal. *His research interests include conceptual as well as application-oriented challenges concerning databases, information systems, process modeling, and Web 2.0 applications, cloud computing, big data, as well as implications. Vossen is an author or co-author of more than 200 publications, and an author, co-author, or co-editor of more than 20 books on databases, business process modeling, the Web, e-commerce, cloud computing, and computer architecture.*