Business Model Ontologies in OLAP Cubes

Christoph Schütz, Bernd Neumayr, and Michael Schrefl

Johannes Kepler University Linz, Austria
{schuetz,neumayr,schrefl}@dke.uni-linz.ac.at

Abstract. Business model ontologies capture the complex interdependencies between business objects. The analysis of the hence formalized knowledge eludes traditional OLAP systems which operate on numeric measures. Many real-world facts, however, do not boil down to a single number but are more accurately represented by business model ontologies. In this paper, we adopt business model ontologies for the representation of non-numeric measures in OLAP cubes. We propose modeling guidelines and adapt traditional OLAP operations for ontology-valued measures.

Keywords: Business Intelligence, Business Modeling, Resource-Event-Agent, Resource Description Framework.

1 Introduction

Business model ontologies capture the complex interdependencies between business objects. More specifically, business model ontologies represent knowledge about the creation of value by economic agents (or actors) through the utilization, production, and exchange of economic resources (or value objects) [1,2,3]. For example, the company *BMW* produces the *Z4* car model which it sells to customers in exchange for money. Knowledge represented in multiple ontologies can be extracted and collected into a single ontology, thereby establishing a common vocabulary. For example, the definition of *BMW* as a company and *Z4* as a car model establishes a common vocabulary for ontologies in the automotive industry. The usage of this vocabulary by different ontologies allows for their comparison.

In Online Analytical Processing (OLAP), business analysts examine real-world facts which are quantified by measures and organized in a multidimensional space (see [4, p. 977 et seq.]), the OLAP cube. The dimensions of this OLAP cube are hierarchically organized. Dedicated query operations allow for the aggregation of facts along the dimension hierarchies as well as the selection of relevant facts. For example, the revenues of car sales are available by country for each quarter of a year. Using the roll-up operation, an analyst could obtain the revenues of car sales by year rather than quarter and by continent rather than country. Furthermore, the analyst could choose to view only car sales in Europe and Asia. In traditional OLAP systems, however, the query operations are restricted to numeric measures.

C. Salinesi, M.C. Norrie, and O. Pastor (Eds.): CAiSE 2013, LNCS 7908, pp. 514–529, 2013.

Many real-world facts are more naturally represented by business model ontologies than numeric measures. The analysis of business model ontologies, however, eludes traditional OLAP systems but OLAP cubes can be extended with ontology-valued measures. As a consequence, OLAP operations become available for the aggregation of knowledge from various business model ontologies in order to gain new insights into a business situation. For example, in India, *BMW* sells cars to upper-class customers whereas the company *Suzuki* sells cars mainly to the middle class. Based on this knowledge alone, *Suzuki* might not perceive *BMW* as a rival. By including knowledge from other countries, however, *Suzuki* might come to the conclusion that *BMW* is indeed a potential threat to its Indian business, with *BMW* already selling cars to the middle class in Europe. On the other hand, by including knowledge from the production department, *Suzuki* might identify synergies in the supply chain, leading to the appreciation of *BMW* as a prospective partner.

In this paper, we introduce ontology-valued measures for facts in OLAP cubes. We propose to model these measures based on the concepts defined in the well-researched and widely-accepted REA business model ontology [1]. We adapt the REA modeling methodology in order to fit the particularities of OLAP systems. We use RDF for the formalization of ontology-valued measures. We associate named RDF graphs [5] with facts, yielding cubes of contextualized RDF data similar to contextualized knowledge repositories for the Semantic Web [6]. We redefine traditional OLAP operations using the SPARQL query language for RDF data in order to select knowledge from various facts and condense this knowledge into a more abstract and concise form.

The remainder of this paper is organized as follows. In Sect. 2, we illustrate the modeling of facts with ontology-valued measures. In Sect. 3, we adapt traditional OLAP operations for ontology-valued measures. In Sect. 4, we briefly present a proof-of-concept prototype. In Sect. 5, we review related work. We conclude with a summary and an outlook on future work.

2 Facts with Ontology-Valued Measures

Base facts, as opposed to derived facts which are the result of OLAP operations, explicitly assign values to measures. Traditional OLAP cubes allow base facts only at the finest level of granularity. We advocate the use of base facts at coarser levels of granularity which serve as shared facts for the definition of common knowledge inherited by the lower-level base facts.

2.1 Base Facts

We base the representation of ontology-valued measures in OLAP cubes on the REA business model ontology [1]. We make use of common business patterns [7] but restrict our considerations, with the addition of types and groups, to the core concepts of REA. The core concepts in REA are economic **R**esources, **E**vents, and **A**gents. The main focus of interest are economic events which use, consume, or produce economic resources or cause flows of stock. In the course of economic events, agents provide and receive control over these resources. Events always occur in duality, each event having a reciprocal, compensatory event. Economic resources, events, and agents may be collected into groups or characterized by types.

The Resource Description Framework[1] (RDF) for the Semantic Web serves as the language for the formalization of ontology-valued measures. In RDF, knowledge is represented as triples of subject, predicate, and object. In such triples, the subject and the predicate are resources, the object is either a resource or a literal. Every resource is denoted by a Uniform Resource Identifier (URI) which may be shortened using prefixes. In the graphical representation, ovals represent resources used as subjects or objects, arrows represent predicates, and rectangles represent literals. For simplicity, and since the formalization of REA does not require their inclusion, we omit blank nodes.

RDF Schema[2] (RDFS) allows for the definition of classes and properties in order to structure RDF data. Classes are sets of RDF resources. Properties are directed relationships from a domain class to a range class. The `rdf:type` property explicitly defines membership of a resource in a class. Classes and properties may also be arranged in hierarchies of sub-classes and sub-properties. Notice, however, that definitions in RDFS, unlike a database schema, do not impose any constraints on the instantiating RDF data. Rather, RDFS permits logical inferencing through automated reasoning.

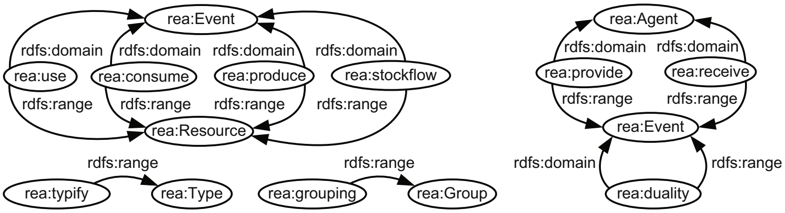


Fig. 1. Definition in RDF Schema of the REA business model ontology's core concepts and their relationships

We use RDFS for the definition of the REA ontology's core concepts and the relationships between these concepts (Fig. 1). Each REA concept translates to an RDFS class, each relationship between concepts translates to an RDFS property. Unlike standard REA models, the RDF formalization has relationships that are directed. The RDF resources `rea:Resource`, `rea:Event`, `rea:Agent`, `rea:Type`, and `rea:Group` are implicitly defined as classes since they are the domain and/or range of properties. The other RDF resources are implicitly defined as properties since they have a domain and/or range. In order to describe occurrences of real-world events, the defined classes and properties are instantiated.

[1] `http://www.w3.org/RDF/`

[2] `http://www.w3.org/TR/rdf-schema/`

Depending on the application domain, specializations of the generic REA concepts and their relationships can improve the expressiveness of an ontology. For example, a competitor analysis might consist of analyses of sales events and value chains. Therefore, an ontology for a competitor analysis introduces specialized events (Fig. 2). On the one hand, sales events are provided in exchange for a payment; sales have a sold quantity, payments generate revenue. On the other hand, there are production events; a production has a produced quantity. Tool models, product models, and payment types are kinds of types. Logical reasoning over ontology-valued measures based on the specialized set of classes and properties may return more informative results than direct instantiations of the generic classes and properties alone. Whether or not a generic class or property is specialized depends on the scope of the analysis.

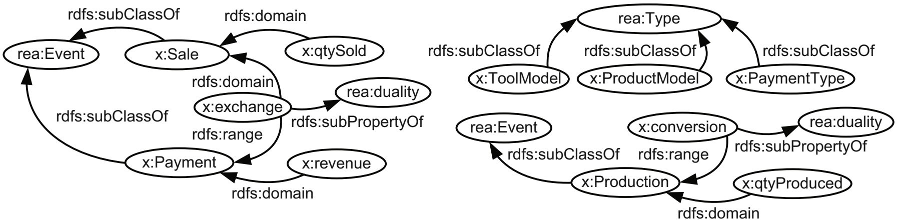


Fig. 2. Specializations of the generic REA concepts for competitor analysis

The RDF formalization of REA differs from standard REA modeling which distinguishes between metamodel, application model, and runtime model [7]. The metamodel defines the general concepts and the relationships between them. In the formalization, the definitions in Fig. 1 correspond to the metamodel. The application model applies the metamodel to a particular domain. In the RDF formalization, the specializations in Fig. 2 correspond to the application model. The runtime model captures the occurrences of events, individual resources and agents. In the RDF formalization, the instantiations of the generic REA classes and properties as well as their specializations correspond to the runtime model. This approach differs notably from other formalizations of REA where the instances of the generic concepts constitute the application model [8]. The approach is similar to the object-oriented implementation of a sample problem provided by Hruby [7, p. 133 et seq.].

The realization of the application model as a specialization of the metamodel, and with it the possibility to instantiate the metamodel as part of the runtime model, has several advantages. First, since the generic concepts are specialized only where needed for the analysis, the ontologies remain concise and focused on the important aspects, which facilitates the task of the analyst. Second, the ontologies are compatible with the decidable subsets of the Web Ontology

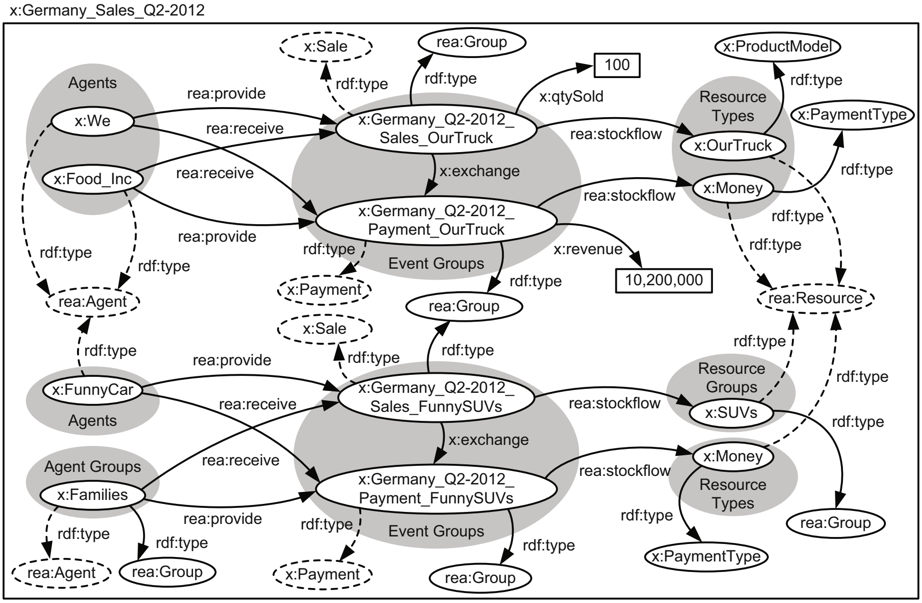


Fig. 3. The German sales department's competitor analysis for the second quarter of 2012 using an adaptation of REA for OLAP (inferred knowledge dashed)

Language[3] (OWL), which is important for compatibility with existing tools and the efficiency of reasoning.

Figure 3 illustrates a fictitious scenario in the automotive industry which instantiates the generic REA concepts and their specializations. The German sales department of a car company conducts a competitor analysis for the second quarter of 2012. From the sales department's point of view, the competitor analysis consists of conducting a survey on what products the different companies in the industry sell to customers. The analyzing company (`x:We`) sells one hundred vehicles of its truck model (`x:OurTruck`) to a producer of aliments (`x:Food_Inc`). In exchange, the company receives a payment of money which generates a revenue of 10,200,000. Another car company (`x:FunnyCar`) sells SUVs to families in exchange for payments of money.

The ontology in Fig. 3 captures economic events at various levels of granularity. In this regard, the ontology extends the standard REA modeling methodology. The REA ontology originates from accounting information systems which are transactional and not primarily intended for analysis. A high level of detail characterizes standard REA modeling. For example, an ontology following the standard REA modeling methodology might represent that the *Millers* bought a car of the *FunnySUV* product model with serial number *7329* on *May 15th, 2012*. OLAP systems, however, use data from data warehouses which typically abstract from individual economic events. Rather, a data warehouse contains data of

[3] `http://www.w3.org/TR/owl-semantics/`

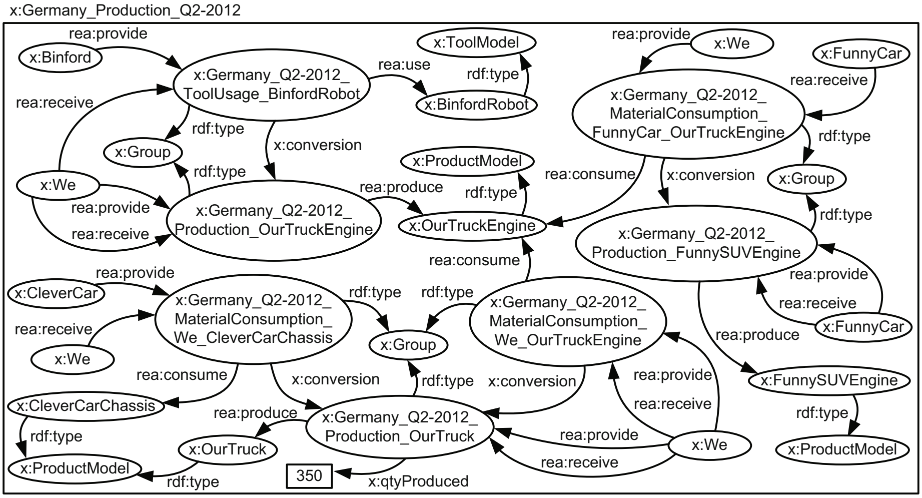


Fig. 4. The German production department's competitor analysis for the second quarter of 2012

interest at an adequate level of detail [4, p. 977]. For example, an ontology-valued measure might represent the knowledge that *Families* bought *SUVs* from *FunnyCar* in the second quarter of 2012. This use of groups and types instead of individual resources, events, and agents is non-standard REA methodology. We propose a variation of the REA modeling methodology for OLAP which allows for the use of groups and types instead of individuals.

Variations from the standard REA modeling methodology should be made explicit. For example, in Fig. 3, the sales events are explicitly defined as instances of class `x:Group`. Through inference, these groups are automatically classified as instances of class `x:Sale` and transitively as instances of class `rea:Event`. Similarly, the exchanged resources are explicitly defined as instances of `x:ProductModel` and `x:PaymentType`, respectively, and thus instances of `rea:Type`. Through inference, these types are automatically classified as instances of `rea:Resource`. The automatic classification stems from the RDFS definitions. When types and groups are used like individual events, resources, et cetera, the RDFS inference engine views them as individuals. The explicit definition as groups and types, however, reminds the analyst of the deviation from the standard modeling methodology.

The ontology in Fig. 3 is similar in intent to a sales cube in traditional OLAP. Still, the ontology has a different focus and is less-structured than a traditional OLAP cube; the ontology is also multi-granular. The ontology emphasizes the relationships between producers and customers whereas a traditional OLAP cubes's focus is on numbers. The ontology is less-structured since sales events and payments of the rival company (`x:FunnyCar`) have no figures available.

The ontology is multi-granular in the sense that sales of one company are captured by product model (x:OurSUV) whereas sales of the other company (x:FunnyCar) are captured by product category (x:SUVs).

The production department's analysis of the business situation (Fig. 4), where the interdependencies of the companies in the value chain are the focus of interest, is another case of knowledge better represented by an ontology than a numeric measure. The company produces a truck engine (x:OurTruckEngine which, together with the chassis from a supplier (x:CleverCar), is converted into a car model (x:OurTruck). The same truck engine is also used by another car manufacturer (x:FunnyCar) as the basis for an SUV engine (x:FunnySUVEngine). This ontology is another example of REA modeling with adaptations for OLAP because the events abstract from the individual workers who carry out the tasks. For example, FunnyCar provides the production event of x:FunnySUVEngine. In this case, the individual worker is of no interest for the analysis.

2.2 Shared Facts

We use the Dimensional Fact Model (DFM) [9] as the conceptual modeling approach for the multidimensional organization of OLAP cubes. In the DFM, the main modeling primitives are fact schemas and dimensions. Dimensions consist of several dimension attributes, the aggregation levels, which are ordered from most to least granular. A fact schema represents a real-world event of interest and defines a set of measures which quantifies the fact. Instances of the fact schema represent occurrences of the event. Measures may be aggregated along the aggregation levels of the dimensions.

Typically, in the DFM, instances of the fact schema, which are the base facts of the OLAP cube, exist only for the most granular level. Using optional aggregation paths, however, we work around this restriction of the DFM in order to allow for base facts at multiple levels of granularity. The base facts at coarser levels of granularity serve as shared facts which define common knowledge inherited by the lower-level base facts. Thus, rather than duplicating common knowledge in every base fact, shared facts make explicit the existence of such shared knowledge and its scope, which facilitates the task of the analyst.

Figure 5 illustrates the fact schema, *Strategy*, of a three-dimensional cube for competitor analysis. Arcs between levels indicate aggregation paths, for example,

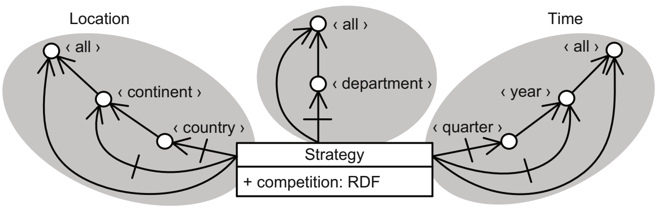


Fig. 5. The *Strategy* fact schema in DFM notation

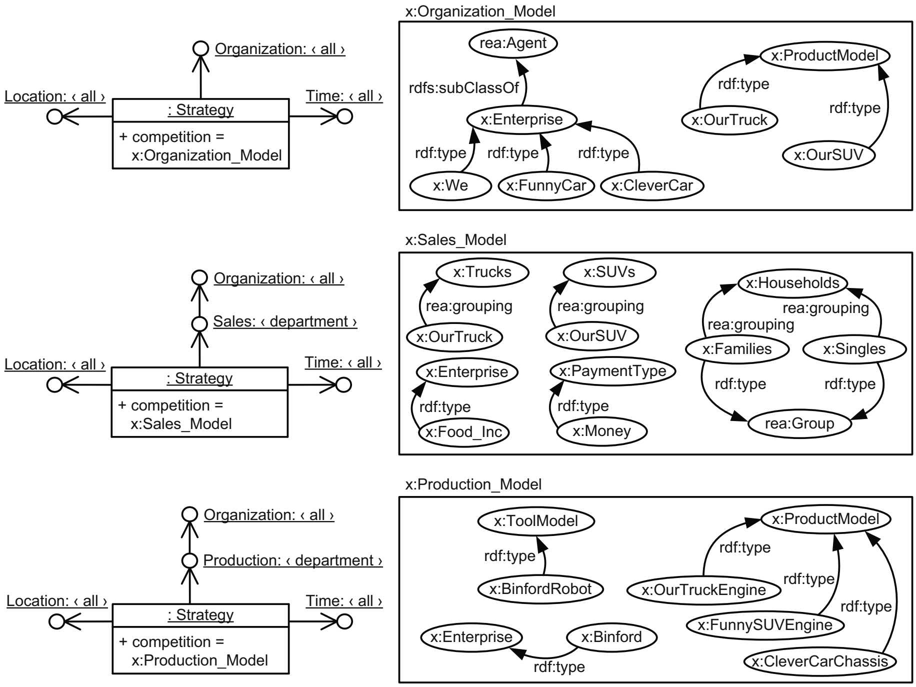
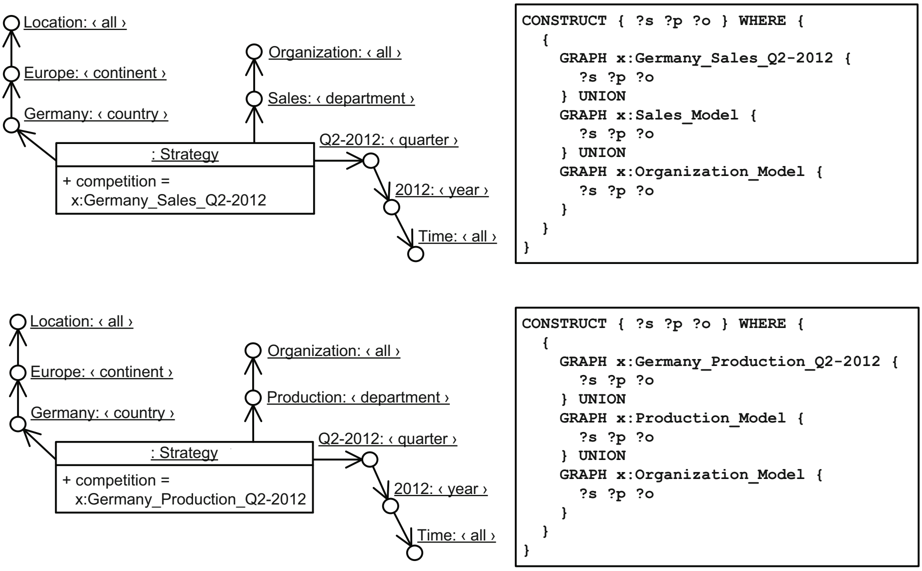


Fig. 6. Shared facts of the *Strategy* fact schema in Fig. 5

countries may be aggregated to continents. Optional aggregation paths are marked with a dash and denote the possibility that some instances of the fact schema may not roll-up to a particular level. The possibility of optional aggregation paths is important for the representation of base facts at a granularity level that is not the most detailed. For example, an instance of the *Strategy* fact schema may be defined at the *department* level for all locations and all points in time. The granularity level of this fact is not the most detailed. The fact may serve as a shared fact which contains knowledge inherited by the facts at more detailed levels of abstraction.

For each ontology-valued measure, a base fact assigns a named RDF graph, the knowledge that is valid in the context of the fact. Named graphs denote sets of RDF triples and allow for the representation of context and provenance of RDF data [5]. In this sense, OLAP cubes with ontology-valued measures are similar to contextualized knowledge repositories for the Semantic Web [6]. The dimensions set the context for the knowledge that is encoded in the RDF data of the cube.

In Fig. 6, different base facts instantiate the *Strategy* fact schema and assign named RDF graphs as values for the *competition* measure. The knowledge represented in RDF graph `x:Organization_Model` is valid in the context of the whole organization for all locations and all points in time. The knowledge represented



```
CONSTRUCT { ?s ?p ?o } WHERE {
  {
    GRAPH x:Germany_Sales_Q2-2012 {
      ?s ?p ?o
    } UNION
    GRAPH x:Sales_Model {
      ?s ?p ?o
    } UNION
    GRAPH x:Organization_Model {
      ?s ?p ?o
    }
  }
}
```

```
CONSTRUCT { ?s ?p ?o } WHERE {
  {
    GRAPH x:Germany_Production_Q2-2012 {
      ?s ?p ?o
    } UNION
    GRAPH x:Production_Model {
      ?s ?p ?o
    } UNION
    GRAPH x:Organization_Model {
      ?s ?p ?o
    }
  }
}
```

Fig. 7. Specifications of how base facts inherit knowledge from the shared facts in Fig. 6

in RDF graphs `x:Sales_Model` and `x:Production_Model` is valid only for the sales and production department, respectively.

The hierarchical organization of the dimensions, which is defined in the DFM, establishes an implicit order between facts which can be used for the definition of a common vocabulary local to specific regions of the OLAP cube. For a formal definition of the hierarchical order of fact instances, we refer to existing work on data warehousing [10,9] and contextualized knowledge repositories [6]. From a context point of view, all knowledge that is valid in the general context is also valid in the specific context. Therefore, the measure values define a common vocabulary which is applicable to different parts of the cube. The lower-level base facts inherit knowledge represented in the higher-level ancestor base facts which are thus shared facts. The facts in Fig. 6 are such shared facts.

We use SPARQL [4] queries to obtain, for a given base fact, the RDF triples inherited from the ancestor base facts (Fig. 7). For example, the German sales department's competitor analysis for the second quarter of 2012 inherits the definitions specific to the whole organization as well as the definitions for the sales department. Likewise, the German production department's analysis for the same period of time inherits the definitions for the organization and the production department. The result contains the union of all RDF triples from a base fact and its ancestor base facts. The generation of the queries can be automated. We use a similar query structure for the roll-up of facts in Sect. 3.

[4] `http://www.w3.org/TR/sparql11-query/`

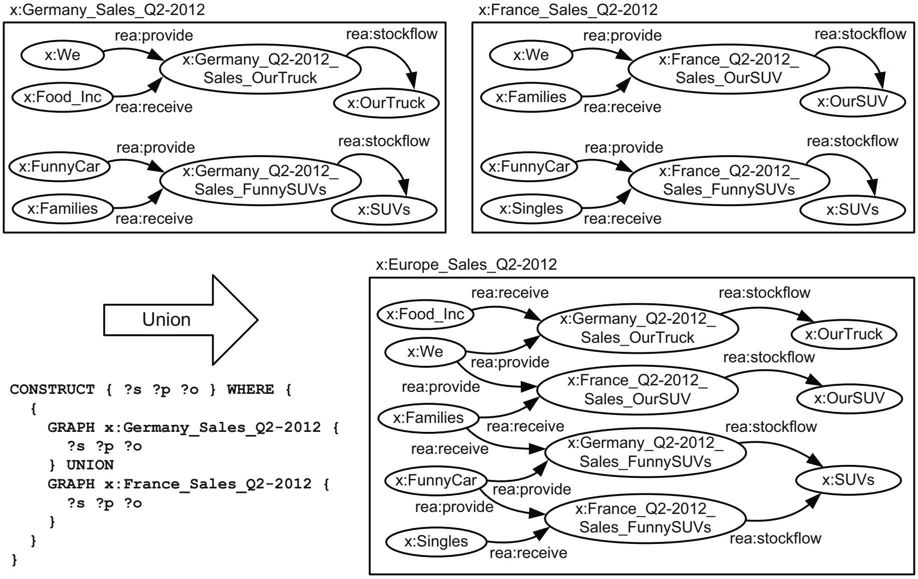
3 OLAP with Ontology-Valued Measures

Roll-up, slice, and dice are the most common OLAP operations. A roll-up changes the OLAP cube's level of granularity by summarizing the data. Slice and dice select a set of facts from an OLAP cube based on criteria. In this section, we adapt the most common OLAP operations for cubes with ontology-valued measures.

In traditional OLAP, the roll-up operation retrieves measure values from multiple facts and applies an aggregation function on this set of values. The application of the aggregation function transforms the set of values into a single value. The result is a summarized view of real-world events.

For ontology-valued measures, the roll-up operation merges ontologies from multiple facts and applies an abstraction pattern on the merged ontology. This abstraction pattern transforms the knowledge that is represented in the merged ontology such that the propositions are more general but also more intelligible for the analyst, which often involves a reduction of the propositions. In the RDF formalization, merge and abstract are defined in terms of SPARQL queries over the RDF graphs that are stored in the facts.

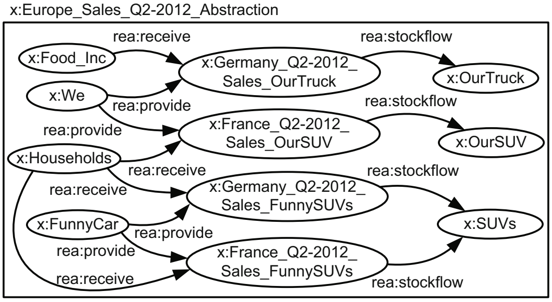
In the most basic case, RDF graphs are merged by forming the union of triples. The result RDF graph contains every triple that exists in at least one of the input graphs. The same query principle is also used for obtaining the inherited RDF triples. Figure 8 illustrates the merging of the competitor analyses of the sales departments in Germany and France by forming the union of the RDF triples



```
CONSTRUCT { ?s ?p ?o } WHERE {
  {
    GRAPH x:Germany_Sales_Q2-2012 {
      ?s ?p ?o
    } UNION
    GRAPH x:France_Sales_Q2-2012 {
      ?s ?p ?o
    }
  }
}
```

Fig. 8. Merging facts by forming the union of RDF triples

```
DELETE {
  ?s rea:receive ?o .
}
INSERT {
  ?g rea:receive ?o .
}
WHERE {
  {
    ?s rea:receive ?o .
    ?o rdf:type x:Sale .
    ?s rea:grouping ?g .
  }
}
```



```
DELETE {
   ?s1 ?p1 ?o1 .
   ?s2 ?p2 ?o2 .
} INSERT {
   x:Sales_FunnyCar_SUVs ?p1 ?o1 .
   ?s2 ?p2 x:Sales_FunnyCar_SUVs .
} WHERE {
  {
   ?s1 ?p1 ?o1 .
   ?s1 rdf:type x:Sale .
   x:FunnyCar rea:provide ?s1 .
   ?s1 rea:stockflow x:SUVs .
  } UNION {
   ?s2 ?p2 ?o2 .
   ?o2 rdf:type x:Sale .
   x:FunnyCar rea:provide ?o2 .
   ?o2 rea:stockflow x:SUVs .
  }
}
```

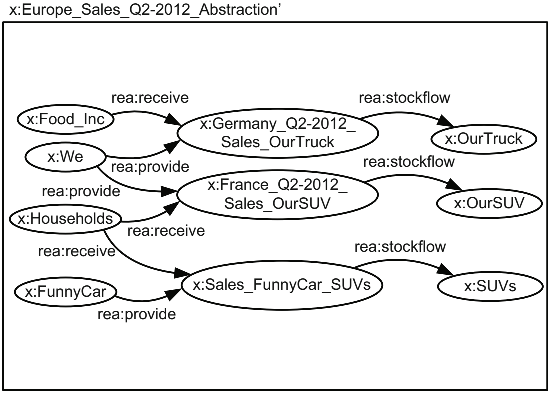
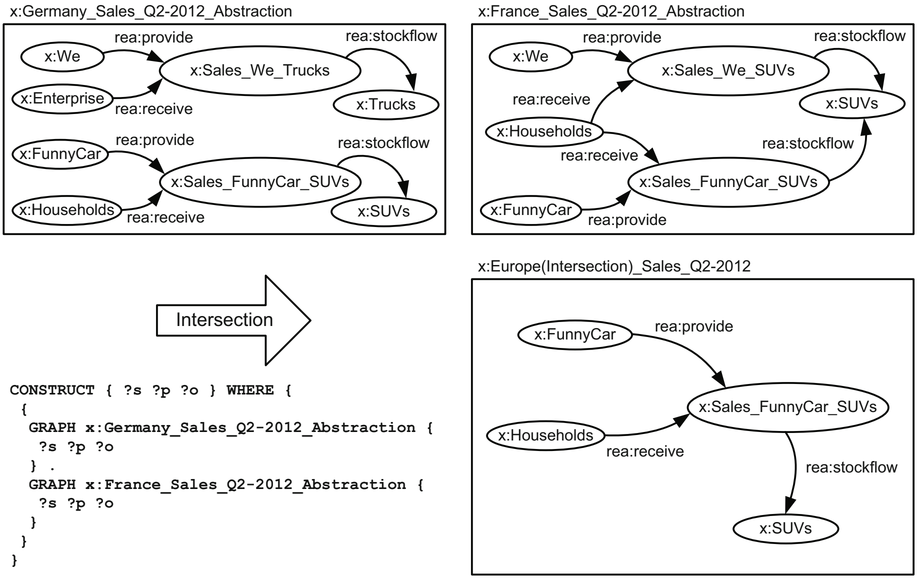


Fig. 9. Incremental abstraction of a fact's RDF data, starting from the merged RDF graph `x:Europe_Sales_Q2-2012` in Fig. 8

of the base facts. The result is an RDF graph which belongs to a derived fact at a more abstract level of granularity than the input base facts.

We use the SPARQL update language[5] to specify abstraction patterns for RDF graphs. A set of RDF triples that fulfil a certain graph pattern should be deleted from the RDF graph. Instead of the deleted RDF triples, new triples are inserted into the RDF graph. This approach corresponds to graph pattern matching and replacement in common model transformation languages [11,12]. Figure 9 illustrates the incremental application of abstraction patterns on RDF data. As the first step, clients participating in sales events are moved to a coarser granularity. All agents that participate in a sales event through the `rea:receive` property are replaced by their group, if they are part of a group. This condition applies to `x:Families` and `x:Singles`, which are replaced by `x:Households`. This condition does not apply to `x:Food_Inc`. As the second step, sales of `x:FunnyCar` that involve a stock flow of `x:SUVs` are grouped together. The new, compound sales event participates in all triples, be it as subject or object, that either of the original sales events participated in.

[5] http://www.w3.org/TR/sparql11-update/



```
CONSTRUCT { ?s ?p ?o } WHERE {
 {
  GRAPH x:Germany_Sales_Q2-2012_Abstraction {
   ?s ?p ?o
  } .
  GRAPH x:France_Sales_Q2-2012_Abstraction {
   ?s ?p ?o
  }
 }
}
```

Fig. 10. Merging facts by forming the intersection of RDF triples

Facts may also be merged by forming the intersection of the contained RDF triples. The result graph contains every triple that exists in all of the input graphs. This variant of the merge operation is useful for obtaining knowledge that is true regardless of the context. For example, merging facts with intersection may retrieve knowledge which has been stable over a period of time. Likewise, it may also retrieve knowledge which applies to several countries or departments of the company.

In many cases, merging facts with intersection will require abstraction to be performed on the input RDF graphs prior to forming the combination of the RDF triples. With intersection, the more detailed the representation of economic events, the fewer the triples that are in the result. Figure 10, for example, illustrates a roll-up over abstractions of the competitor analyses of the sales departments in Germany and France. In this example, the abstractions are merged by forming the intersection of triples. The original input RDF graphs, as seen in previous examples, were too finely grained for obtaining results. Without prior abstraction of the input RDF graphs, in this case, the result RDF graph would not contain any RDF triples. Similarly, some queries will require RDFS reasoning to be performed before execution.

In traditional OLAP, slice and dice correspond to the selection of facts based on some user-defined criteria. When applied to ontologies, slice and dice are special cases of abstraction. Slice and dice reduce the number of RDF triples. Thus, slice and dice correspond to an abstraction with only a delete part but no insertions.

4 Implementation

We provide a proof-of-concept prototype[6] with basic management and analysis capabilities for OLAP cubes with ontology-valued measures. Ontology-valued measures are stored in the Jena[7] RDF framework's native tuple store (TDB) which supports the SPARQL query and update language as well as RDFS reasoning. The multidimensional organization is stored in an object-relational database whereas TDB contains only a set of RDF graphs.

In the object-relational database, each OLAP cube has a table with one column for each ontology-valued measure and each row corresponds to a base fact. The columns store names of RDF graphs. The order of the base facts is established by a multilevel cube [10] which allows for the definition of facts at multiple levels of granularity. Facts at higher levels of granularity serve as shared facts with represent common knowledge shared by the finer-grained descendant base facts. For each fact in this cube, the ancestor facts are calculated and stored at the time of insertion, providing an index for the OLAP operations, which is particularly important for the retrieval of inherited knowledge. The implementation is an extension of the prototype implementation for the management of hetero-homogeneous data warehouses [13].

We provide methods in Java for the merging and abstraction of facts with ontology-valued measures as well as the extension of facts with inherited knowledge. Merging is implemented as a binary operation which merges two RDF graphs at a time and returns the merged RDF graph as result. Abstraction takes an RDF graph and an abstraction pattern, formalized as a SPARQL update, as input and returns a modified RDF graph as result. Nesting of operations is possible by applying the methods on results of methods. All methods also work with the identifying dimensional attributes as parameters and, in this case, automatically retrieve from the object-relational database the names of the RDF graphs associated with the facts. The extension of facts with inherited knowledge works only for the identifying dimensional attributes as parameters and cannot be applied on results.

5 Related Work

The application of semantic technologies to business intelligence promises to bridge the gap between the technical specifications of the systems designer and the knowledge of the business analyst [14,15]. Multidimensional ontologies from the *Semantic Cockpit* project [14] describe the semantics of dimensions and measures in order to facilitate the formulation of OLAP queries and the interpretation of results. Similarly, the Business Intelligence Model (BIM) [15,16] aims at presenting the data in terms familiar to the business analyst. Nebot et al. [17] investigate the multidimensional organization of ontologies and the semi-automated extraction of OLAP cubes with numeric measures from these

[6] The prototype is available on `http://hh-dw.dke.uni-linz.ac.at/`

[7] `http://jena.apache.org/`

ontologies. Other work [18] deals with the use of domain ontologies as semantic dimensions in OLAP cubes. These approaches are complementary to the use of ontology-valued measures or could be useful as the basis for modeling ontology-valued measures.

There are approaches which use the REA ontology for the development of a traditional OLAP system. These approaches, however, differ significantly from the use of REA as the basis for modeling ontology-valued measures. On the contrary, these approaches emulate the structure of traditional OLAP systems in REA. REAL-D [19] uses a variant of the REA ontology extended with concepts for time and location. Numeric measures are stored in a snowflake schema. Economic events are facts and the numeric properties of the events are measures. Resources, agents, time, and location are the dimensions. Likewise, the posting and account business patterns [7] keep track of transactions and allow for the aggregation of attributes associated with these transactions.

InfoNetOLAP [20] extends traditional OLAP to graph data. Graphs are associated with dimensional attributes. The edges of the graphs are weighted. Typical applications of InfoNetOLAP are co-author graphs, social graphs, spatio-temporal applications, et cetera, over different time periods, geographic locations, et cetera. A distinction between informational roll-up and topological roll-up is made, which is similar to the basic idea of merge and abstract for ontology-valued measures. InfoNetOLAP, however, is not intended for the representation of complex knowledge. Rather, InfoNetOLAP is another means of structuring numeric measures. Business model ontologies as a data model for measures are much more expressive, powerful, and versatile than simple graphs. Also, the graphs in InfoNetOLAP are much more structured and uniform. In this regard, by using business model ontologies in OLAP cubes, we go beyond the InfoNetOLAP approach.

6 Summary and Future Work

Numeric measures alone fail to adequately represent many complex real-world facts. Business model ontologies can fill the gap but their analysis eludes traditional OLAP systems. By extending OLAP cubes with ontology-valued measures, the analyst can leverage the expressivity of business model ontologies without losing the analysis capabilities of OLAP systems. The REA business model ontology may serve as the basis for modeling such ontology-valued measures.

Future work will explore the use of other business model ontologies, depending on the context of the analysis, as the basis for the representation of ontology-valued measures. Among these business model ontologies are e^{3}value [21] and its domain-specific extensions, for example, e^{3}forces [22] for modeling Porter's five forces for industry analysis. Likewise, the use of composite indicators [23] could prove beneficial. Of particular interest is the use of a variety of different business model ontologies within the same OLAP cube and the analysis across different ontologies.

Future work will also integrate ontology-valued measures into the *Semantic Cockpit* [14] which leverages semantic technologies for business intelligence in order to support business analysts with the retrieval and interpretation of data. Whereas semantic dimensions [18] allow for the use of knowledge from existing domain ontologies in OLAP queries, ontology-valued measures employ business model ontologies in order to represent complex real-world facts.

Acknowledgments. This work was supported by the FIT-IT research program of the Austrian Federal Ministry for Transport, Innovation, and Technology under grant FFG-829594.

References

1. Geerts, G., McCarthy, W.: An ontological analysis of the economic primitives of the extended-REA enterprise information architecture. International Journal of Accounting Information Systems 3, 1–16 (2002)
2. Gordijn, J., Akkermans, H., van Vliet, H.: Business modelling is not process modelling. In: Mayr, H.C., Liddle, S.W., Thalheim, B. (eds.) ER Workshops 2000. LNCS, vol. 1921, pp. 40–51. Springer, Heidelberg (2000)
3. Weigand, H., Johannesson, P., Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T.: On the notion of value object. In: Martinez, F.H., Pohl, K. (eds.) CAiSE 2006. LNCS, vol. 4001, pp. 321–335. Springer, Heidelberg (2006)
4. Elmasri, R., Navathe, S.: Fundamentals of database systems, 5th edn. Pearson International Edition. Addison Wesley, Boston (2007)
5. Carroll, J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. In: 14th International Conference on World Wide Web, pp. 613–622. ACM (2005)
6. Serafini, L., Homola, M.: Contextualized knowledge repositories for the semantic web. Web Semantics: Science, Services and Agents on the World Wide Web 12-13, 64–87 (2012)
7. Hruby, P.: Model-driven design using business patterns. Springer, Heidelberg (2006)
8. Gailly, F., Laurier, W., Poels, G.: Positioning and formalizing the REA enterprise ontology. Journal of Information Systems 22(2), 219–248 (2008)
9. Golfarelli, M., Maio, D., Rizzi, S.: The Dimensional Fact Model: a conceptual model for data warehouses. International Journal of Cooperative Information Systems 7(2-3), 215–247 (1998)
10. Neumayr, B., Schrefl, M., Thalheim, B.: Hetero-homogeneous hierarchies in data warehouses. In: 7th Asia-Pacific Conference on Conceptual Modeling, pp. 61–70. Australian Computer Society, Darlinghurst (2010)
11. Jouault, F., Kurtev, I.: Transforming models with ATL. In: Bruel, J.-M. (ed.) MoDELS 2005. LNCS, vol. 3844, pp. 128–138. Springer, Heidelberg (2006)
12. Wimmer, M., Kappel, G., Kusel, A., Retschitzegger, W., Schoenboeck, J., Schwinger, W.: Surviving the heterogeneity jungle with composite mapping operators. In: Tratt, L., Gogolla, M. (eds.) ICMT 2010. LNCS, vol. 6142, pp. 260–275. Springer, Heidelberg (2010)
13. Schütz, C.: Extending data warehouses with hetero-homogeneous dimension hierarchies and cubes: A proof-of-concept prototype in Oracle. Master's thesis, Johannes Kepler University Linz, Austria (2010)

14. Neumayr, B., Schrefl, M., Linner, K.: Semantic cockpit: An ontology-driven, interactive business intelligence tool for comparative data analysis. In: De Troyer, O., Bauzer Medeiros, C., Billen, R., Hallot, P., Simitsis, A., Van Mingroot, H. (eds.) ER Workshops 2011. LNCS, vol. 6999, pp. 55–64. Springer, Heidelberg (2011)
15. Jiang, L., Barone, D., Amyot, D., Mylopoulos, J.: Strategic models for business intelligence. In: Jeusfeld, M., Delcambre, L., Ling, T.-W. (eds.) ER 2011. LNCS, vol. 6998, pp. 429–439. Springer, Heidelberg (2011)
16. Barone, D., Topaloglou, T., Mylopoulos, J.: Business intelligence modeling in action: A hospital case study. In: Ralyté, J., Franch, X., Brinkkemper, S., Wrycza, S. (eds.) CAiSE 2012. LNCS, vol. 7328, pp. 502–517. Springer, Heidelberg (2012)
17. Nebot, V., Berlanga, R.: Building data warehouses with semantic web data. Decision Support Systems 52, 853–868 (2012)
18. Anderlik, S., Neumayr, B., Schrefl, M.: Using domain ontologies as semantic dimensions in data warehouses. In: Atzeni, P., Cheung, D., Ram, S. (eds.) ER 2012. LNCS, vol. 7532, pp. 88–101. Springer, Heidelberg (2012)
19. O'Leary, D.E.: REAL-D: a schema for data warehouses. Journal of Information Systems 13(1), 49–62 (1999)
20. Chen, C., Zhu, F., Yan, X., Han, J., Yu, P., Ramakrishnan, R.: InfoNetOLAP: OLAP and mining of information networks. In: Yu, P.S., Han, J., Faloutsos, C. (eds.) Link Mining: Models, Algorithms, and Applications, pp. 411–438. Springer, Heidelberg (2010)
21. Gordijn, J., Wieringa, R.: A value-oriented approach to e-business process design. In: Eder, J., Missikoff, M. (eds.) CAiSE 2003. LNCS, vol. 2681, pp. 390–403. Springer, Heidelberg (2003)
22. Pijpers, V., Gordijn, J.: e^3*forces*: Understanding strategies of networked e^3*value* constellations by analyzing environmental forces. In: Krogstie, J., Opdahl, A.L., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 188–202. Springer, Heidelberg (2007)
23. Barone, D., Jiang, L., Amyot, D., Mylopoulos, J.: Composite indicators for business intelligence. In: Jeusfeld, M., Delcambre, L., Ling, T.-W. (eds.) ER 2011. LNCS, vol. 6998, pp. 448–458. Springer, Heidelberg (2011)