A UML Profile for Variety and Variability Awareness in Multidimensional Design: *An application to agricultural robots*

Houssam Bazza
IDS Team, Abdelmalek Essaadi University
Tangier, Morocco
houssam.bazza@etu.uae.ac.ma

Sandro Bimonte
TSCF, INRAE Clermont-Ferrand
Aubière, France
sandro.bimonte@inrae.fr

Stefano Rizzi
DISI, University of Bologna
Bologna, Italy
stefano.rizzi@unibo.it

Jean Laneurit
TSCF, INRAE Clermont-Ferrand
Aubière, France
Jean.Laneurit@inrae.fr

Hassan Badir
IDS Team, Abdelmalek Essaadi University
Tangier, Morocco
badir.hassan@uae.ac.ma

ABSTRACT

Variety and variability are an inherent source of information wealth in schemaless sources, and executing OLAP sessions on multidimensional data in their presence has recently become an object of research. However, all models devised so far propose a "rigid" view of the multidimensional content, without taking into account variety and variability. To fill this gap, in this paper we propose V-ICSOLAP, an extension of the ICSOLAP UML profile that supports extensibility and type/name variability for each multidimensional element, as well as complex data types for measures and levels. The real case study we use to motivate and illustrate our approach is that of trajectory analysis for agricultural robots. As a proof-of-concept for V-ICSOLAP, we propose an implementation that relies on the PostgreSQL multi-model DBMS and we evaluate its performances. We also provide a validation of our UML profile by ranking it against other meta-models based on a set of quality metrics.

KEYWORDS

UML Profile, Big Data, Variety, Variability, Multi-model Databases, Multidimensional Modeling

1 INTRODUCTION

Big Data have now pervaded most applications domains (marketing, industry, agriculture, health, etc.) thanks to the enhanced analysis capabilities enabled by the *6Vs* feature [15], namely, Volume, Velocity, Variety, Value, Veracity, and Variability. In particular, variety refers to the possibility of having data of different kinds coexist: highly structured (e.g., relational databases), semi-structured (coming from sensors, social networks, etc.) or unstructured (such as photos and videos). Conversely, variability refers to the increase in the range of the values and the continuous data changing in terms of structure and meaning. In order to store, manage, and analyze these data, NoSQL DBMSs have been created; they overcome the rigid structure of relational DBMSs to natively support models —such as document-based, graph-based, key/value— that better comply with the complex nature of Big Data. These new models are characterized by a high degree of flexibility; the structure of data is not fixed, it can be modified and extended in each single piece of data to grant a comprehensive support to variability and evolution. For this reason, such models are often called *schemaless*.

Big Data analysis rests on several methods lent from OnLine Analytical Processing (OLAP), as well as from Artificial Intelligence, Data Mining, and Statistics. In particular, OLAP enables the interactive exploration of data represented according to the *multidimensional model*, which relies on the concepts of analysis subjects (*facts*), analysis coordinates (*dimensions*), and numerical indicators (*measures*). Dimensions are organized into hierarchies of *levels*, which allow measures to be aggregated at different granularities. Multidimensional design has been extensively investigated for almost three decades, at both the logical and the conceptual levels. At the logical level, the most remarkable result lies in the definition of *star* and *snowflake* schemata as best practices for implementing multidimensional data on relational databases [20]. At the conceptual level, several models have been proposed to represent multidimensional content independently of its implementation, either by extending the Entity/Relationship model, or UML, or introducing ad-hoc graphical formalisms (e.g., [20, 28]).

Carrying out multidimensional modeling at the conceptual level has several advantages; for instance, it streamlines design because accurate conceptual schemata can be quickly derived using a *data-driven approach* [20]. However, to the best of our knowledge, all conceptual models devised so far propose a "rigid" view of the multidimensional content, in that they do not take into account the flexibility that is intrinsic in schemaless data. On the other hand, variety and variability have been recognized to be an inherent source of information wealth in schemaless sources, and executing OLAP sessions in their presence has recently become an object of research [12, 16]. Besides, very few works are focused on multidimensional conceptual models for complex data, such as streams, spatial networks, fuzzy and social networks [5, 8, 24].

To fill this gap, in this work we propose V-ICSOLAP, an extension of the ICSOLAP UML profile [9] that explicitly supports variety (in terms of complex data types) and variability (in terms of both extensibility and type/name variability) at the conceptual level. We chose to extend ICSOLAP because (i) it has been successfully applied in real case studies in different domains [9], and (ii) UML profiles offer a formal, non-ambiguous, and readable graphical notation, which is highly beneficial in the context

of a complex design activity such as variety/variability-aware multidimensional modeling.

one of the most remarkable advantages of conceptual schemata is that they allow designers to focus on the information content by abstracting all implementation details away. Indeed, V-ICSOLAP could be adopted to accurately set a multidimensional view of data in different architectural scenarios:

- **Data warehouses**. A *data warehouse* (DW) is a database aimed at supporting decision-makers in analyzing useful data from heterogeneous sources [23]. The approach followed in DW systems is called *schema-on-write*, because multidimensional schemata are decided at design time and forced onto data (typically, in relational form) at the time of writing them in the DW, so as to enable efficient querying via OLAP tools; thus, conceptual modeling is a crucial phase of DW design because it determines the information content of the DW and, as a consequence, the possible queries that end-users can formulate.
- **Data lakes**. *Data lakes* have been emerging as repository systems for storage, processing, and analysis of schemaless (typically, NoSQL) data, in which data are kept in their original format and are processed to be queried only when needed [13]. Due to the evolving nature of schemaless data, adopting a schema-on-write approach could be complex, so a *schema-on-read* approach, which leaves data unchanged in their structure until they are accessed by the end-user, is normally preferred. Though here the multidimensional schemata are not decided at design time but at query time, conceptual modeling is necessary to let end-users choose a useful perspective for analyzing data through the OLAP paradigm [16].
- **Self-service business intelligence**. Schema-on-write approaches fall short also when data sources are external to the company, unreliable, have a short lifespan, and are required for analyses related to situational needs of advanced users such as data scientists [26]. Even here a schema-on-read approach is necessary, and conceptual modeling is the first step to enable OLAP querying [17].
- **Lakehouses**. The most recent architectural pattern devised in this context is the *lakehouse*, which aims at unifying warehouses and lakes to support both efficient OLAP querying and state-of-the-art analytics [35]. From a technical point of view, this is achieved by combining low-cost storage in an open format accessible by a variety of systems with powerful management and optimization features. OLAP querying requires a multidimensional view of data, so even in this case conceptual modeling is mandatory.
- **Multi-model data warehouses**. A *multi-model DBMS* (MMDBMS) is a data processing platform that natively supports different data models with a fully integrated backend, thus providing unified data governance, management, and access via a single query language while still granting performance, scalability, and fault tolerance [27]. It has been recently shown that storing multidimensional data on an MMDBMS brings further advantages over classical relational DWs, namely, reducing the cost for ETL procedures and ensuring better extensibility and evolvability thanks to the use of schemaless models [7]. Remarkably, a multi-model data warehouse can store data according to the multidimensional model and, at the same time, let each of its elements be natively represented through the most appropriate model; thus, it can be seen as another way to bridge the architectural gap between data lakes and DWs.

Importantly, whichever the architectural scenario adopted, V-ICSOLAP should be considered as the first step of a process aimed at enabling decision makers to query multidimensional content. Although automating this process is out of the scope of this paper, we provide a proof-of-concept for V-ICSOLAP by introducing a case study concerning the analysis of the activities of agricultural robots and showing how it can be implemented in a multi-model DW so as to take full advantage of the flexibility introduced by this new profile. Specifically, we rely on the PostgreSQL MMDBMS and we evaluate its performances for a set of representative OLAP queries. Besides, to prove the effectiveness of the proposed profile, we provide a metrics-based validation based on the quality framework proposed by [3, 30].

The paper is organized as follows. Section 2 discusses the related work; Section 3 presents the motivation and the requirements by means of a real case study; Section 4 presents the V-ICSOLAP profile; Section 5 provides a metrics-based assessment of V-ICSOLAP; Section 6 describes our proof-of-concept for V-ICSOLAP. Finally, Section 7 draws the conclusion and outlines our future work.

2 RELATED WORK

Introducing complex data in multidimensional conceptual models has received a lot of attention in the spatio-temporal context. Many proposals extend the Entity/Relationship model and UML to take into account spatial, temporal, and spatio-temporal data in DWs [19, 32]. Other kinds of complex data, such as topological spatial networks [8], stream data [5], and social network data [24] have also been studied, proposing extensions to the concepts of levels, measures, and fact and support new OLAP operators. Table 1 summarizes the main features of the main conceptual multidimensional models devised. In particular, these models are classified according to the following properties: (i) the support for complex data, (ii) the support for variability (name and type), (iii) the support for extensibility, (iv) the formalism used, (v) the implementation in a CASE tool, (vi) the presence of a formal metamodel, and (vii) the proposal for an implementation (automatic or set of manual guidelines) in a DBMS.

Table 1 shows that spatio-temporal complex types received an important attention, but no work has focused on generic complex data and graph data. Moreover, extensibility is not supported by any work, while variability is only (partially) supported by a specific work on multi-representation of spatial DWs. Indeed, to the best of our knowledge, no approach gives explicit support to variability. The closest we could find are some works about multi-resolution in the context of spatial DWs, which support multi-valued attribute values according to some semantics (such as scale and usage) [4, 29].

In the context of transactional databases, some works propose conceptual models for graph data [18] that are very similar to our approach, as well as for JSON documents [10, 11]. In particular, JSON-related approaches provide a graphical formalism to represent the complexity of documents, but they do not represent variability and extensibility. Other works start from an E/R diagram or a UML class diagram modeling an (either transactional or multidimensional) database and provide its implementation

Table 1: Conceptual multidimensional models and their features

Paper	Complex data types	Variability	Extensibility	Formalism	Case tool implementation	Meta-model	DBMS implementation
[9]	Spatial and temporal types	No	No	UML profile	MagicDraw	Yes	Oracle
[8]	Spatial topological graph	No	No	UML profile	MagicDraw	Yes	Oracle
[6]	Stream	No	No	UML profile	MagicDraw	Yes	Esper
[24]	Tweets	No	No	Ad-hoc	No	No	Oracle
[1]	No	No	No	UML	-	Yes	No
[19]	Spatial types	No	No	UML Profile	Eclipse	Yes	No
[28]	No	No	No	UML Profile	Rational Rose	Yes	Oracle
[20]	No	No	No	Ad-hoc	Indyco	Yes	No
[36]	Spatial and temporal data types	No	Yes	Ad-hoc	No	Yes	Oracle
[25]	Trajectory	No	No	UML Profile	No	Yes	No
[4]	Spatial types	Type	No	UML	Perceptory	No	No
[29]	Spatial types	Type	No	Ad-hoc	No	Yes	Oracle

on NoSQL DBMSs [2, 14, 21, 31, 33]. Even in this case, variability and extensibility are not considered.

Finally, as already mentioned, some recent papers have proposed approaches to enable OLAP queries on schemaless data in presence of variety [12, 16]. In those papers, the focus is on how to enrich and query schemaless data, but conceptual modeling is not addressed.

To conclude, how to take into account variety at a conceptual level has not been investigated yet [22]. Indeed, to the best of our knowledge, only [34] proposed a conceptual model to design databases comprising complex data represented as graph and collections data, but variability and extensibility are not considered; besides, most importantly, that approach refers to transactional databases, whose features are very different from those of multidimensional data. On the other hand, multidimensional modeling in presence of variability and extensibility is becoming more and more important due to the increasing availability of schemaless data and to their relevance for the decision-making process.

3 MOTIVATING CASE STUDY: AGRICULTURAL ROBOTS MONITORING FOR AGRO-ECOLOGY FARMING

Variety and variability are among the main features of Big Data. Commonly, variety refers to non-relational data (semi-structured and unstructured data such as graph, images, etc.), while variability refers to the coexistence different representations of the same data entity in terms of name and type within the same data set [15].

To motivate the need for a conceptual multidimensional model that supports variety and variability, in this section we present a case study concerning the analysis the data produced by autonomous agricultural robots in smart farming.

Autonomous agricultural robots are unmanned ground vehicles equipped with sensors and actuators and capable of safely and autonomously performing one or more tasks while moving in a plot following a predefined trajectory. A trajectory is an ordered list of triples, each consisting of GPS coordinates, timestamp, and speed. Analyzing the actual trajectories followed by robots is very useful to adjust the reference trajectory, avoid recurrent problems caused by persistent obstacles in the fields, etc. Robot are powered by electronic engines. The cost for recharging robot batteries, called *trajectory cost*, is directly proportional to the working time of the robot. An analysis of the trajectory cost and its comparison with the performed work, also considering the total distance covered, are crucial for the farmer. Note that a farmer can either have a monthly subscription with an energy operator or use an on-demand energy recharge.

Usually, robots embed ROS (Robot Operating System), a software platform used to program the main functionalities of the robots. ROS uses a topic message system to exchange information among the physical components of each robot. The exchange of these messages is represented as a graph, whose analysis is useful to reveal recurrent problems. Noticeably, robots rely on different implementations to sense odometry values, thus the data they produce have different types and names. For example, the total distance can be represented using either an integer or a double, and the corresponding attribute name can be either total distance or distance. Besides, the ROS functions that collect odometry data are periodically updated, so it is realistic to think that additional measures will be made available for analyses in the future.

Finally, farmers currently ask to analyze the trajectories and the related costs by robot type and farm (a farm is a set of plots). However, new requirements may easily emerge in the future. For instance, farmers might find out that analyzing data by robot engine power is interesting. Also, should farms be grouped into cooperatives, analyzing data by cooperative will become relevant.

From what said above we can conclude that a variety/variability-aware multidimensional model should support:

(1) **complex multidimensional elements**, i.e., measures that are not simple numerical values and levels that are not simple categorical attributes;
(2) **variability of multidimensional elements**, in terms of both names and types;
(3) **extensible multidimensional elements**, i.e., elements whose structure can evolve in the future.

4 THE V-ICSOLAP UML PROFILE

A UML profile provides a generic extension mechanism for customizing UML models for particular domains and platforms. It is defined using stereotypes, tagged values definitions, and constraints applied to specific model elements, like Classes, Attributes, and Operations. Note that in, UML 2.0, tagged values are named *meta-class properties*. In this work, to avoid confusion with classical properties we will use the old terminology of tagged values.

Some UML profiles for DWs have been proposed in literature, as mentioned in the previous section ([1, 9, 28]). They are all based on the standard formalism represented by UML and therefore they come with the extensibility property offered by the UML profile mechanisms. The profile we have chosen to extend in this work is *ICSOLAP*, which already supports advanced multidimensional structures and has been successfully employed in several agro-environmental OLAP projects [9]. Remarkably, other UML profiles for DWs could be extended in quite a similar way.

ICSOLAP represents dimensions and hierarchies with package stereotypes. Levels are represented with the `AggLevel`

class stereotype. A level is composed of descriptive attributes (DescriptiveAttribute property stereotype). A fact is modeled using the Fact class stereotype and it contains measures (NumericalMeasure). Dimensions, hierarchies, and levels are specialized according to the types of the data they represent. ICSOLAP also defines the concept of BasicIndicator, which represents how measures are aggregated along dimensions.

V-ICSOLAP is the extension to ICSOLAP we propose to cope with variety and variability. It enables a formal and non-ambiguous conceptual design of facts characterized by variable and complex elements, and supports their extensibility. Noticeably, our extension adds few new elements to ICSOLAP, so as to let its readability intact. We do not present the entire meta-model, but only focus on the multidimensional elements of ICSOLAP that we have specialized. The new elements are shown in Figure 1. We recall that, in the UML notation, boxes represent classes (abstract classes in italics), solid lines are associations, solid lines with white triangular arrow are specializations, and solid lines with black diamonds are compositions. Finally, each stereotype is connected to the classifier it enhances using an *extension* (solid line with black arrow).

As a working example, in the remainder of the paper we will rely on the Trajectories fact, at the core of the case study presented in Section 3. Its measures are: Trajectory (the trajectory of the robot as composed of a set of GPS/timestamp/speed tuples; Trajectory cost (the cost for the battery energy used by the robot), Total duration (the time duration of the trajectory covered by the robot), Total distance (the total length of the robot trajectory), and Message (the graph of ROS topics). These measures are analyzed according to three main dimensions: Robot (the robot used for the work; robots are grouped by their type), Location (the plots and the farms), Time (a temporal dimension with day and year granularities). The instance of V-ICSOLAP representing the Trajectories fact is shown in Figure 2; in red, the multidimensional elements that rely on extensions to ICSOLAP.

The main new elements of V-ICSOLAP are DocumentFact, DocumentDimension, DocumentHierarchy, and DocumentAggLevel. These elements specialize, respectively, the Fact, Dimension, Hierarchy, and AggLevel ICSOLAP elements by introducing a tagged value that represents the possibility of extending the corresponding multidimensional element with additional components.

First of all, variability in schemaless data implies extensibility. DocumentFact includes two Boolean tagged values (DimensionExtensibility and MeasureExtensibility) that state whether a fact can be extended with new dimensions and measures, respectively. For example, fact Trajectories has DimensionExtensibility and MeasureExtensibility set to true.

A DocumentDimension can be extended by including additional hierarchies (tagged value HiearchyExtensibility), while a DocumentHierarchy can be extended by including new levels (tagged value LevelExtensibility). Finally, the possible presence of additional DescriptiveAttributes of a DocumentLevel is represented via the tagged value DescriptiveAttributeExtensibility. For example, the package hierarchy RobotH has HiearchyExtensibility set to true, meaning that new levels can be added to aggregate the Robot level.

Variability also occurs in terms of types and names of multidimensional elements. In particular, VariableTypeDescriptiveAttribute is a specialized stereotype of a DescriptiveAttribute that is characterized by several possible types. In the same way, the Measure property stereotype is extended (VariableTypeMeasure). Note that, since any data type may be chosen to be used in OLAP queries, it is important to specify type conversion methods (ConversionMethods operation stereotype). Indeed, conversion methods for complex data (one for each ordered couple of different types allowed) must be defined during design by decision makers, who are expert of the application domain. VariableTypeDescriptiveAttribute and VariableTypeMeasure present then an "Undefined type" [1]. An example of VariableTypeMeasure is Total distance, which comes with two conversions methods CastDistanceIntToDouble and CastDistanceDoubleToInt.

Finally, name variability is supported by V-ICSOLAP extension with new stereotypes for measures and level attributes: VariableNameMeasure and VariableNameDescriptiveAttribute, respectively. For example, Total duration has two names in the tagged value Names: Total duration and Duration. Figure 3 shows an OCL constraint defined in the context of the VariableNameMeasure stereotype to check that Names contains at least two items.

Note that variability in the structure of documents in terms of missing levels is already supported by ICSOLAP.

As to complex data modeling implied by variety, ICSOLAP already handles alphanumeric, spatial, and temporal types for measures and levels. However, in the context of Big Data new complex types can appear, therefore, in V-ICSOLAP we introduce some new complex types, namely, Graph and ComplexObject. A Graph class is a composition of Vertex and Arc (i.e. edge) classes stereotypes. These classes may have some fixed properties, but they can also be extended with additional ones; this is represented with the VertexExtensibility tagged value. An example of Graph is the Ros Graph Topic class. ComplexObject is a class extension meant to be used by ComplexMeasure and ComplexDescriptiveAttribute, which represent measures and levels whose type is not atomic but class. Two examples of ComplexObject are the Odometry and EnergyCost classes. EnergyCost is abstract and is specialized in two classes, namely, Subscription and CostUnit.

Our UML profile implemented with Papyrus is available as open-source project. [2]

5 ASSESSMENT OF V-ICSOLAP

In this section we provide a metrics-based validation of the quality of our UML profile based on the framework proposed in [3, 30][3]. This framework proposes to evaluate some quality metrics exclusively using the UML meta-models:

- *Reusability*: it measures to what extent the meta-model can be reused to create other meta-models for other usages.
- *Understandability*: it represents how well the meta-model is understood (i.e., readable) and instanced by end-users.
- *Functionality*: it is the overall capacity of a meta-model to support complete and diverse models.

[1] In order to keep the meta-model as light as possible, in this work we used a standard "Undefined type"; however, a new, specific abstract type (e.g., "VariableType") could easily be added to the profile.

[2] https://www6.inrae.fr/tools4bi/Design/A-UML-Profile-for-Variety-Aware-Data-Warehouse-with-Papyrus

[3] An empirical validation is out of scope for this paper; indeed, it would require a set of real case studies, which are not available to us at this time.

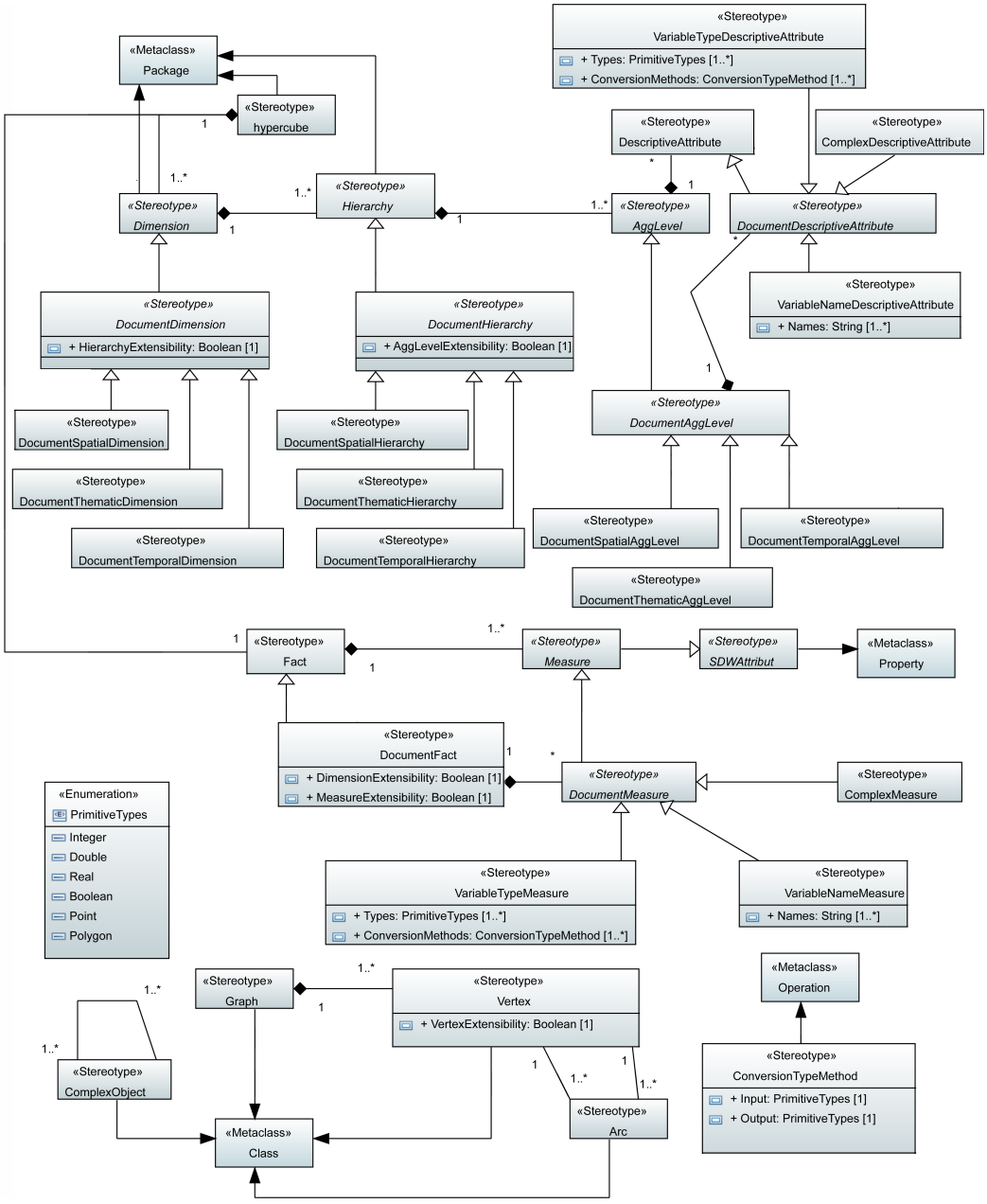

Figure 1: The V-ICSOLAP meta-model (only the elements that are not present in ICSOLAP are shown)

- *Well-structuredness*: it measures how well the meta-model supports the underlying architecture.
- *Extendibility*: it is the degree of ease in adding new modeling elements to the meta-model.

Our goal is to rank V-ICSOLAP and ICSOLAP, in terms of quality, against other meta-models in the literature. Thus, we proceed by first computing these metrics for both ICSOLAP and V-ICSOLAP based on the numbers of elements of the meta-model (such as meta-classes, properties, associations, etc.). Then, we evaluate ICSOLAP and V-ICSOLAP against the 2500 meta-models considered in [3, 30].

The results are shown in Figures 4 and 5. Figure 4 shows that V-ICSOLAP performs better than ICSOLAP as to reusability, extensibility, and functionality; it is roughly as well-structured as ICSOLAP, while it is worst for understandability —which is not surprising, considering that the domain it models is more complex than the one modeled by ICSOLAP. On the other hand, Figure 5 positions V-ICSOLAP and ICSOLAP with their percentile values amongst the 2500 meta-models considered in [3]. Both V-ICSOLAP and ICSOLAP appear to have a good ranking, especially in terms of understandability and well-structuredness, which provides a successful validation.

Finally, since V-ICSOLAP is very close to ICSOLAP and ICSOLAP has been effectively used in several real projects, we can argue that also V-ICSOLAP could be successfully applied in real complex situations.

6 PROOF-OF-CONCEPT

We have implemented V-ICSOLAP using Papyrus, an open-source UML tool based on Eclipse. Papyrus allows for defining UML

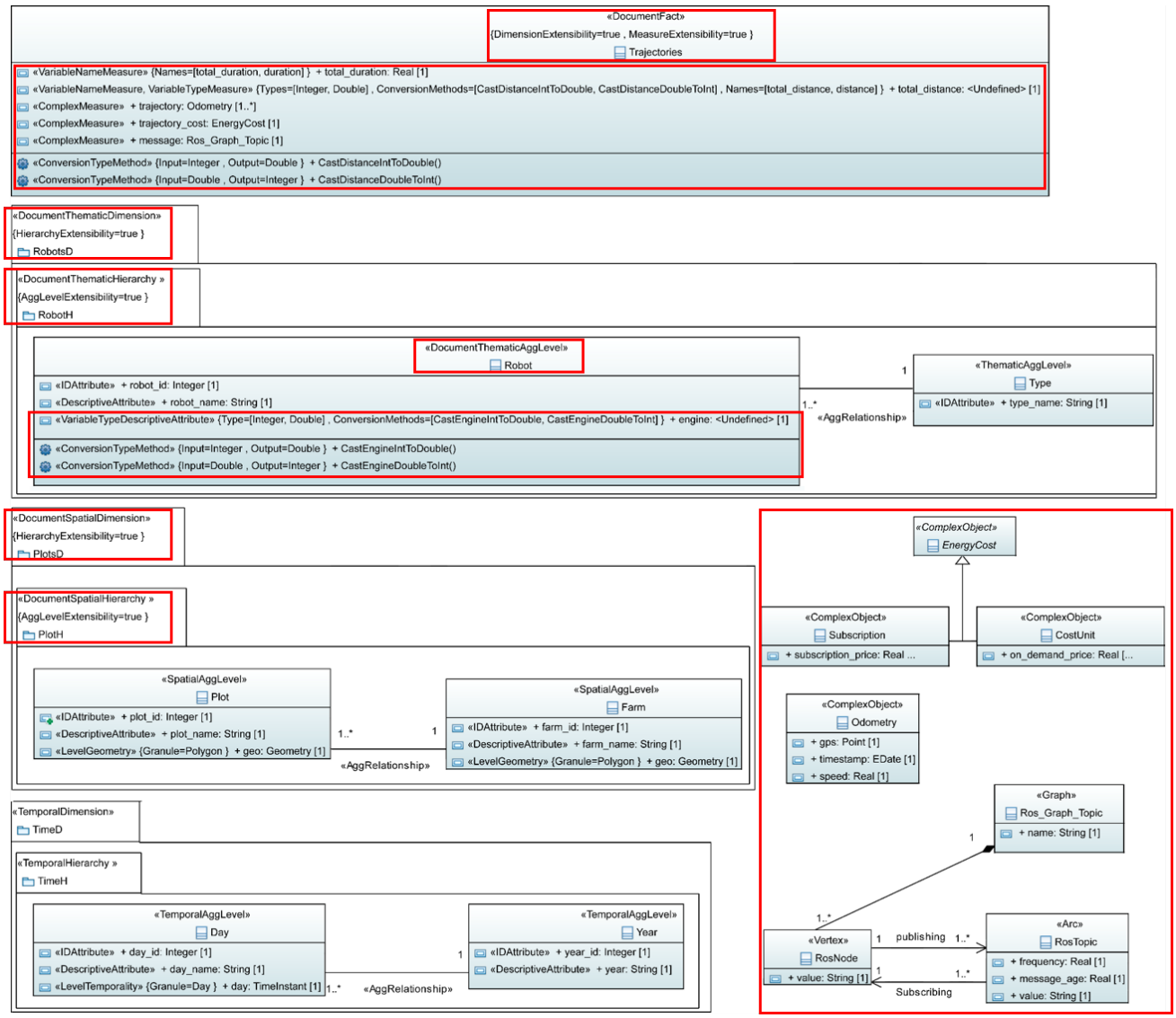

Figure 2: V-ICSOLAP class diagram modeling the case study DW

```
context VariableNameMeasure
inv AtLeastTwoItems:
self.Names->collect(self)->size()>1
```

Figure 3: Constraint on `VariableNameMeasure`

profiles by means of stereotypes and OCL constraints. Figure 6 shows how constraints work on Papyrus. The example refers to measure Total duration with `VariableNamesMeasure` stereotype; the number of defined `Names` is only one, hence, the constraint in Figure 3 is violated.

In this section we provide a proof-of-concept for V-ICSOLAP by showing how the class diagram for the trajectory case study can be faithfully translated into a logical schema on the PostgreSQL DBMS and efficiently queried.

6.1 Logical design

In this section we explain how the class diagram for the Trajectories fact, shown in Figure 2, can be implemented in a database. As

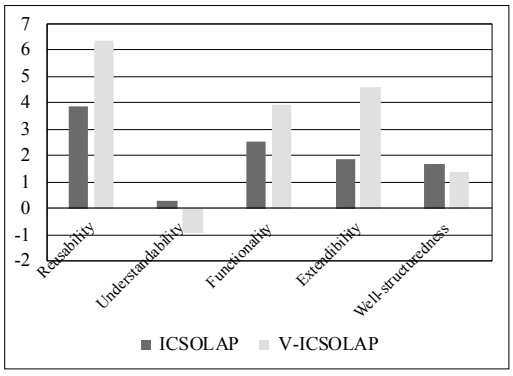

Figure 4: Comparison of V-ICSOLAP and ICSOLAP according to the framework of [30]

explained in the following, the translation of the class diagram into a logical schema has been carried out manually; although partially automating this translation towards some target logical

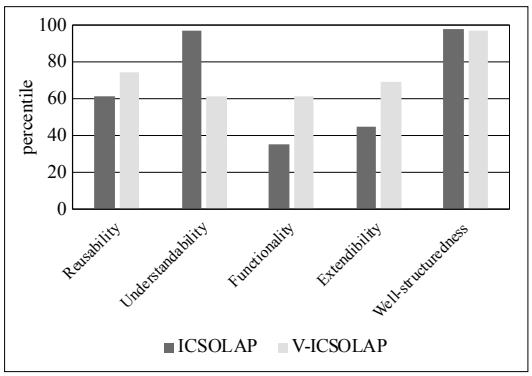

Figure 5: Comparison of V-ICSOLAP and ICSOLAP according to the framework of [30], in the context of all the metamodels considered in [3]

model is clearly possible given a set of design best practices, it is out of scope for this paper and is left for future work. We also show some OLAP queries on this database that address variability issues on the one hand, deal with complex data on the other.

A classical implementation relying on the relational model would not properly cope with the peculiar features of this fact, thus, we opted for the multi-model paradigm, which has been recently introduced for DBMSs to take advantage of different models. Specifically, we chose the PostgreSQL MMDBMS (plus the AgensGraph extension, bitnine.net/agensgraph/) since it handles relational, graph, and JSON data, which are the types of data used in our case study. Indeed, source data for the Trajectories fact come in different formats: (i) graph for ROS topic messages data, (ii) JSON for odometry data, and (iii) CSV with diverse structures for the remaining data.

Starting from the class diagram of Figure 2 and following the design guidelines defined in [7], we have obtained the multi-model logical schema shown in Figure 7. The core is a star schema [20] including one fact table and three dimension tables. The fact table (Fact_Trajectories) contains, besides the foreign keys referring the dimensions tables, four attributes of type JSON for the total duration, total distance, trajectory, and cost measures. All these measures are transformed from CSV into JSON to cope with their variability and complexity. The fact table also contains an attribute node_id which references the main node in the topic messages graph (measure message). The trajectory measure is represented as a JSON document with a vector of three required attributes.

JSON allows to handle the name and type variability of measures, levels, and descriptive attributes by means the `oneOf` JSON keyword, which forces each document to have only one of the listed attributes. An example of use of the `oneOf` keyword is measure total_distance for both name and type variability, since it can have either number or integer type.

To model extensibility, the idea is to add a JSON attribute to the table representing the fact or the dimension. An example is the JSON attribute measures_extens, added to the fact table to cope with the fact extensibility.

6.2 Query formulation

In this section we show some OLAP queries over this logical schema, and we highlight how the particular JSON keywords used to handle complex types, variability, and extensibility also influence the query definition. The goal is to show that querying a multi-model database created starting from a V-ICSOLAP class diagram is feasible with limited effort, and that an analyst can uniformly query all the data despite their variability. Clearly, this requires that the analyst knows the naming conventions in the different source documents; making this transparent to the user would require offering a layer of generic names that cover all variability, and is outside the scope of this paper.

Measure name and type variability are solved using in the select statement the `concat` operator and the `cast` operator, respectively. For instance, Figure 8 shows the SQL statement that returns the average duration per robot. When name variability concerns a level in an aggregated query, formulation becomes more complex. For example, assuming that the robot level is affected by name variability, a subquery solving name variability on the robot_name/robot level should be defined to be then used for the group by.

The `concat` and `cast` operators provide general solutions to name and type variability, respectively, for JSON data. In presence of complex measures, this is not enough and ad-hoc statements must be used to cope with the particularities of measure structures. As a first example, Figure 9.a shows the statement that computes the energy cost per day. We recall from Section 3 that farmers can either have a monthly subscription with an energy operator or use an on-demand energy recharge; the subscription price is not defined at the day, robot, and plot level. Following our multi-model approach, data can be loaded from sources into the fact table in their native format, so there is no need to apply a transformation to trajectory_cost measure to artificially disaggregate subscription costs. As a result, decision makers can choose at query time the transformation they prefer; in Figure 9.a, subscription costs per day are assumed to be 0.

Another example is the trajectory complex measure, for which we have implemented a query that finds the odometry data with maximum speed value per robot, farm and year, thus allowing farmers to understand the mechanical behaviour of their robots (Figure 9.b). This query is implemented with the `concat` operators previously mentioned. Moreover, this query uses the JSON operator `jsonarrayelementstext` that transforms the JSON array into a set of tuples, in order be able to use the `max` aggregation operator.

For the graph complex measure, we have implemented the graph intersection as shown in Figure 9.c. This query uses the Cypher language of AgensGraph combined to classical SQL operators. Interestingly, the join between the fact table and the graph nodes is obtained with the Cypher match operator.

6.3 Querying performance

In this section we provide some tests to assess the efficiency of OLAP queries over the logical schema of Figure 7. In particular, as shown in Table 2, we defined a set of OLAP queries that involve name and type variability for measures and levels, as well as complex measures. Some of these queries are the ones described in the previous section. Table 2 shows the variety/variability features involved in the query, the attributes used for grouping data (i.e., the GROUP BY clause), and the attributes used for selection (WHERE clause). In particular, $q1$ (Figure 8), $q2$, and $q3$ involve name and type variability for numerical measures, while $q4$ groups data by robot name —which presents a name variability. Queries $q5$, $q6$, and $q7$ use the *Cost* complex measure (we recall that *Cost* can be an on-demand cost, which is daily measured,

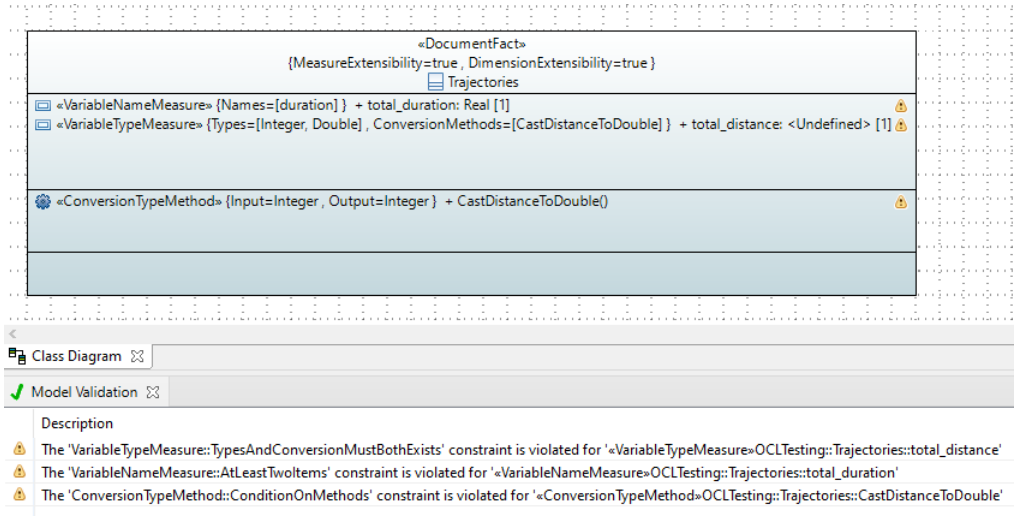

Figure 6: Example of constraint violation in Papyrus

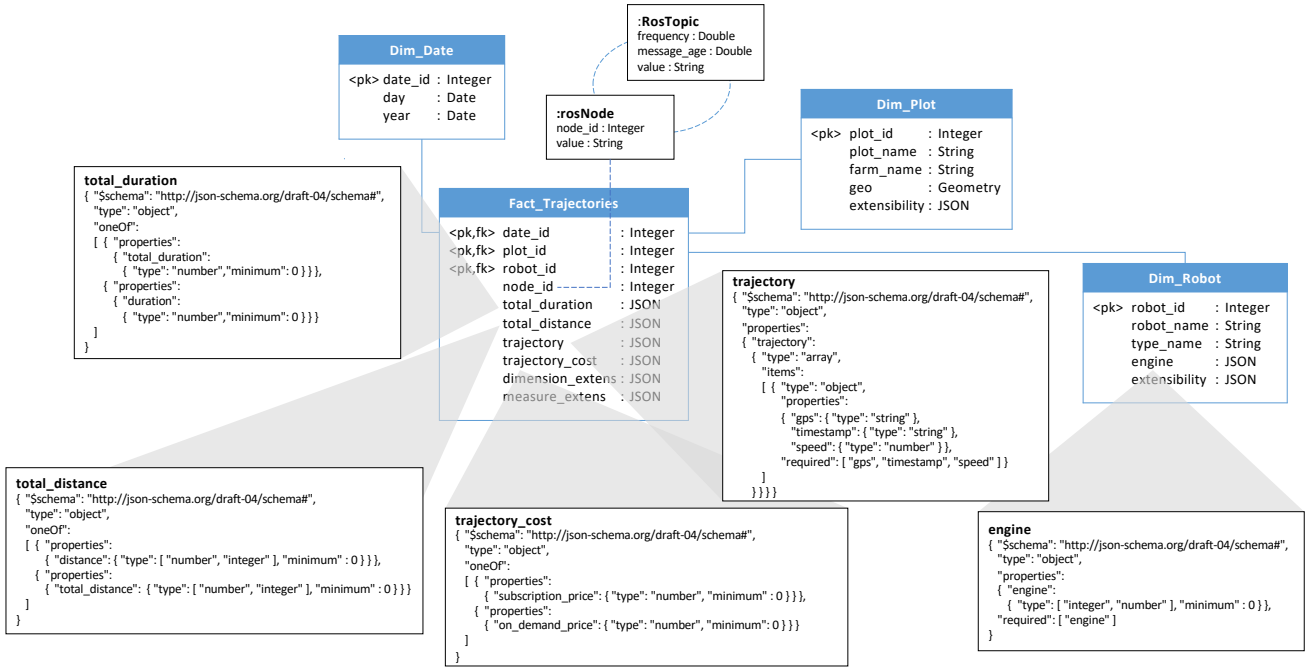

Figure 7: Multi-model logical schema for the Trajectories fact

```
with  robot as (
  select robot_id, CONCAT(robot_name->>'robot_name',robot_name->>'name' ) as "robot_name“
  from "Dim_Robot")
select robot_name, avg(CONCAT(total_duration->>'total_duration',total_duration->>'duration' )::float) as "total_duration"
from "Fact_Trajectories" f, "Dim_Plot"  ,robot
where robot.robot_id=f.robot_id  and farm_name='farm30'
group by robot_name
```

Figure 8: OLAP query in presence of measure and level variability

and a monthly cost associated to a subscription by the farmer). Specifically, $q5$ (Figure 9.a) uses only the on-demand cost without taking into account the subscription price. Query $q6$ aggregates data by day; then, a simple ratio by the number of days per month

```
with pricing_day as (
 SELECT  case
   when cast(f.trajectory_cost->>'on_demand_price'as float) is not null
   then (cast(f.trajectory_cost->>'on_demand_price' as float) )
   else 0 END as  "daily_pricing",dim_d.Day,dim_p.farm_name
 from "Fact_Trajectories" f, "Dim_Date" dim_d,"Dim_Plot"dim_p
 where f.date_id= dim_d.date_id and f.plot_id=dim_p.plot_id
)
select day,sum(daily_pricing)
from pricing_day
where farm_name = 'farm25'
group by day;
```

(a)

```
with gr as (
  select n,b,year,quarter
  from fact_trajectories,dim_date,
       (match (n:rosnode) match(b:rosnode)
        where n.value=b.value and n.id!=b.id return n,b )  as t
  where fact_trajectories.date_id=dim_date.date_id and  topic_message_id=((n->'id')::text)::int)
select * from gr where year=2021
```

(c)

```
with robot as (
   select robot_id, concat(robot_name->>'robot_name', robot_name->>'name') as "robot_name"
   from Dim_Robot),
max_speed as (
    select  date_id,robot.robot_id,plot_id,json_array_elements_text(trajectory)::json  as point_time_speed,robot_name
     from Fact_Trajectories,robot
     where robot.robot_id= Fact_Trajectories.robot_id and robot.robot_name='robot10')

select robot_name,year,farm_name,point_time_speed
from max_speed, Dim_Date,Dim_Plot
where Dim_Date.date_id=max_speed.date_id and Dim_Plot.plot_id =max_speed.plot_id
 and(((point_time_speed->'speed')::text)::int) in
    (select max((((point_time_speed->'speed')::text)::int))
     from max_speed,Dim_Date,Dim_Plot
     where Dim_Date.date_id=max_speed.date_id  and Dim_Plot.plot_id=max_speed.plot_id
    group by year,farm_name);
```

(b)

Figure 9: OLAP queries in presence of complex measures: (a) cost, (b) trajectory, and (c) graph

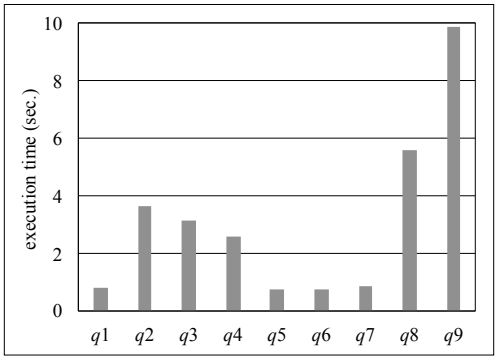

Figure 10: Execution time (in seconds) for the queries of Table 2

is used to translate the monthly cost of subscription to a daily one. Finally, $q7$ sums all (daily and monthly) costs per year. Query $q8$ (Figure 9.b) uses the complex measure representing the trajectory (set of Odometry objects), and finally $q9$ (Figure 9.c) involves the graph measure as previously described. Overall, these queries represents a realistic workload covering all the features that can be present in a variety/variability-aware DW.

Using the logical schema in Figure 7, we have simulated a fact table with 1.57 million tuples. Each trajectory contains 1440 points. Figure 10 shows the execution time of the different queries in the workload. The tests have been done using an i3 with 2CPU @ 2.20 GHz laptop with 8GB RAM and SSD running on Windows10 version 10H2. Note that queries that involve classical measures and levels with name and type variability have small computation time. Queries $q8$ and $q9$ yield worse performances since aggregation is more complex ($q8$ aggregates 1440 points and $q9$ 1000 trajectories), nevertheless they are still in line with the interactivity required by OLAP.

7 CONCLUSION AND FUTURE WORK

Big Data is characterized by the 6Vs, among which variety and variability are very relevant. Taking into account them in DW and OLAP systems is challenging, since multidimensional schemata are typically characterized by a rigid schema and simple data types. Shifting towards variety/variability-aware DWs is promising, as several recent works deal with new approaches for formulating OLAP queries on schemaless datasets or multi-model databases. In this paper, motivated by the need for a conceptual design step before the implementation of these complex DW and OLAP systems, we proposed a UML profile for multidimensional schemata taking into account name and type variability, extensibility, and complex data. We described our proposal using a real-case study issued from a smart farming application. We also showed a validation of the quality of the UML model, and we perform some tests to assess the implementation of our metamodel using a multi-model logical schema implemented with PostgreSQL.

We are currently working to define a set of best practices for the implementation of multi-model DWs taking time and space performance into account. This work will then be used to set up a complete design pipeline from our UML profile towards a PostgreSQL implementation. Another interesting direction for our future work consists in defining a classification of all possible complex data that can be defined using V-ICSOLAP. This will allow us to set a formal framework for an assessment of the modeling capabilities associated to our UML profile.

Table 2: Variety/variability-aware workload

Query	Description	Variety/Variability	Multidim. element	Type	Group By	Where	Measure
$q1$	Average total duration per robot for a given farm	Variability	Measure	Name and Type	Robot name	Farm	Total duration
$q2$	Minimum duration per robot for a given year	Variability	Measure	Name	Plot	Year	Total duration
$q3$	Average distance per plot	Variability	Measure	Name	Plot	–	Total distance
$q4$	Total number of trajectories per robot	Variability	Level	Name	Robot name	–	Robotid
$q5$	Total cost (only on-demand) per day for a given farm	Variety	Measure	Complex object	Day	Farm	Trajectory_cost
$q6$	Total cost per day for a given farm	Variety	Measure	Complex object	Day	Farm	Trajectory_cost
$q7$	Total cost per year and farm	Variety	Measure	Complex object	Year and Farm	Farm	Trajectory_cost
$q8$	Odometry with max speed per farm and year for given robot	Variety	Measure	Complex object	Year and Farm	Robot	Trajectory
$q9$	ROS nodes common to all ROS topic graphs for a given year	Variety	Measure	Graph	–	Year	rosNode, node_id

Acknowledgements. This work is supported by the French National Research Agency project 20-PCPA-0002 BEYOND, and French gov-ernment IDEX-ISITE initiative 16-IDEX-0001 (CAP 20-25)

REFERENCES

[1] Alberto Abelló, José Samos, and Fèlix Saltor. 2006. YAM^2: a multidimensional conceptual model extending UML. *Inf. Syst.* 31, 6 (2006), 541–567.

[2] Paolo Atzeni, Francesca Bugiotti, Luca Cabibbo, and Riccardo Torlone. 2020. Data modeling in the NoSQL world. *Comput. Stand. Interfaces* 67, 103149 (2020).

[3] Francesco Basciani, Juri Di Rocco, Davide Di Ruscio, Ludovico Iovino, and Alfonso Pierantonio. 2019. A tool-supported approach for assessing the quality of modeling artifacts. *Journal of Computer Languages* 51 (2019), 173–192.

[4] Yvan Bédard, Marie-Josée Proulx, Suzie Larrivée, and Eveline Berniee. 2002. Modeling Multiple Representations into Spatial Data Warehouses: a UML-Based Approach. In *Proc. Symposium on Geospatial Theory, Processing and Applications*. 1–6.

[5] Sandro Bimonte, Omar Boussaid, Michel Schneider, and Fabien Ruelle. 2019. Design and Implementation of Active Stream Data Warehouses. *Int. J. Data Warehous. Min.* 15, 2 (2019), 1–21.

[6] Sandro Bimonte, Omar Boussaid, Michel Schneider, and Fabien Ruelle. 2019. Design and Implementation of Active Stream Data Warehouses. *Int. Journal of Data Warehousing and Mining (IJDWM)* 15, 2 (2019), 1–21.

[7] Sandro Bimonte, Enrico Gallinucci, Patrick Marcel, and Stefano Rizzi. 2021. Data variety, come as you are in multi-model data warehouses. *Inf. Syst.* 104 (2021), 101734.

[8] Sandro Bimonte, Myoung-Ah Kang, and Juan Trujillo. 2013. Integration of Spatial Networks in Data Warehouses: A UML Profile. In *Proc. ICCSA*. 253–267.

[9] Kamal Boulil, Sandro Bimonte, and Francois Pinet. 2015. Conceptual model for spatial data cubes: A UML profile and its automatic implementation. *Computer Standards & Interfaces* 38 (2015), 113–132.

[10] Zouhaier Brahmia, Fabio Grandi, Safa Brahmia, and Rafik Bouaziz. 2021. A Graphical Conceptual Model for Conventional and Time-varying JSON Data. In *Proc. ANT*. 823–828.

[11] David Chaves and Elzbieta Malinowski. 2019. Document Data Modeling: A Conceptual Perspective. In *Proc. ADBIS Short Papers*. 19–27.

[12] Mohamed Lamine Chouder, Stefano Rizzi, and Rachid Chalal. 2019. EXODuS: Exploratory OLAP over Document Stores. *Inf. Syst.* 79 (2019), 44–57.

[13] Julia Couto, Olimar Teixeira Borges, Duncan D. Ruiz, Sabrina Marczak, and Rafael Prikladnicki. 2019. A Mapping Study about Data Lakes: An Improved Definition and Possible Architectures. In *Proc. SEKE*. Lisbon, Portugal, 453–578.

[14] Gwendal Daniel, Abel Gómez, and Jordi Cabot. 2019. UMLto[No]SQL: Mapping Conceptual Schemas to Heterogeneous Datastores. In *Proc. RCIS*. 1–13.

[15] Cheikh Kacfah Emani, Nadine Cullot, and Christophe Nicolle. 2015. Understandable Big Data: A survey. *Comput. Sci. Rev.* 17 (2015), 70–81.

[16] Enrico Gallinucci, Matteo Golfarelli, and Stefano Rizzi. 2019. Approximate OLAP of document-oriented databases: A variety-aware approach. *Inf. Syst.* 85 (2019), 114–130.

[17] Enrico Gallinucci, Matteo Golfarelli, Stefano Rizzi, Alberto Abelló, and Oscar Romero. 2018. Interactive multidimensional modeling of linked data for exploratory OLAP. *Inf. Syst.* 77 (2018), 86–104.

[18] Amine Ghrab, Oscar Romero, Sabri Skhiri, Alejandro A. Vaisman, and Esteban Zimányi. 2016. GRAD: On Graph Database Modeling. *CoRR* abs/1602.00503 (2016).

[19] Octavio Glorio, Jose-Norberto Mazón, Irene Garrigós, and Juan Trujillo. 2012. A personalization process for spatial data warehouse development. *Decis. Support Syst.* 52, 4 (2012), 884–898.

[20] Matteo Golfarelli and Stefano Rizzi. 2009. *Data Warehouse Design: Modern Principles and Methodologies*. McGraw-Hill, Inc.

[21] Shady Hamouda and Zurinahni Zainol. 2017. Document-Oriented Data Schema for Relational Database Migration to NoSQL. In *Proc. Innovate-Data*. 43–50.

[22] Irena Holubová, Pavel Contos, and Martin Svoboda. 2021. Multi-Model Data Modeling and Representation: State of the Art and Research Challenges. In *Proc. IDEAS*. Montreal, QC, Canada, 242–251.

[23] R. Kimball, L. Reeves, M. Ross, and W. Thornthwaite. 1998. *The Data Warehouse Lifecycle Toolkit*. John Wiley & Sons.

[24] Maha Ben Kraiem, Jamel Feki, Kaïs Khrouf, Franck Ravat, and Olivier Teste. 2015. Modeling and OLAPing social media: the case of Twitter. *Soc. Netw. Anal. Min.* 5, 1 (2015), 47:1–47:15.

[25] Olfa Layouni and Jalel Akaichi. 2016. A conceptual UML profile for modeling fuzzy trajectory data: An ambulance use case. In *Proc. AICCSA*. Agadir, Morocco, 1–6.

[26] Alexander Löser, Fabian Hueske, and Volker Markl. 2008. Situational Business Intelligence. In *Proc. BIRTE*, Malú Castellanos, Umeshwar Dayal, and Timos Sellis (Eds.). Auckland, New Zealand, 1–11.

[27] Jiaheng Lu and Irena Holubová. 2019. Multi-model Databases: A New Journey to Handle the Variety of Data. *ACM Comput. Surv.* 52, 3 (2019), 55:1–55:38.

[28] Sergio Luján-Mora, Juan Trujillo, and Il-Yeol Song. 2006. A UML profile for multidimensional modeling in data warehouses. *Data Knowl. Eng.* 59, 3 (2006), 725–769.

[29] Concepcion M. Gascueña and Rafael Guadalupe García. 2009. A Multidimensional Methodology with Support for Spatio-Temporal Multigranularity in the Conceptual and Logical Phases. In *Progressive Methods in Data Warehousing and Business Intelligence: Concepts and Competitive Analytics*. 194–230.

[30] Zhiyi Ma, Xiao He, and Chao Liu. 2013. Assessing the quality of metamodels. *Frontiers of Computer Science* 7, 4 (2013), 558–570.

[31] Jihane Mali, Faten Atigui, Ahmed Azough, and Nicolas Travers. 2020. ModelDrivenGuide: An Approach for Implementing NoSQL Schemas. In *Proc. DEXA*. 141–151.

[32] Elzbieta Malinowski and Esteban Zimányi. 2008. *Advanced Data Warehouse Design - From Conventional to Spatial and Temporal Applications*. Springer.

[33] Amal Sellami, Ahlem Nabli, and Faïez Gargouri. 2020. Graph NoSQL Data Warehouse Creation. In *Proc. iiWAS*. 34–38.

[34] Martin Svoboda, Pavel Contos, and Irena Holubová. 2021. Categorical Modeling of Multi-model Data: One Model to Rule Them All. In *Proc. MEDI*, J. Christian Attiogbé and Sadok Ben Yahia (Eds.). Tallinn, Estonia, 190–198.

[35] Matei Zaharia, Ali Ghodsi, Reynold Xin, and Michael Armbrust. 2021. Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics. In *Proc. CIDR*.

[36] Esteban Zimányi and Mohammed Minout. 2006. Implementing Conceptual Spatio-temporal Schemas in Object-Relational DBMSs. In *Proc. OTM Workshops*. Montpellier, France, 1648–1657.