Multidimensional Modeling Driven From a Domain Language

Multidimensional Modeling Driven From a Domain Language

Leandro Antonelli[1*], Sandro Bimonte[2†] and Stefano Rizzi[3†]

[1*]LIFIA - Facultad de Informática, UNLP, 50 and 120, La Plata, 1900, Buenos Aires, Argentina.
[2]UR TSCF, INRAE, 9 Avenue Blaise Pascal, Aubiere, 63178, Auvernia-Ródano-Alpes, France.
[3]DISI, University of Bologna, Viale Risorgimento 2, Bologna, 40136, Emilia-Romagna, Italy.

*Corresponding author(s). E-mail(s): leandro.antonelli@lifia.info.unlp.edu.ar;
Contributing authors: sandro.bimonte@inrae.fr; stefano.rizzi@unibo.it;
†These authors contributed equally to this work.

Abstract

Purpose: The multidimensional model is based on the concepts of facts (business phenomena to be analyzed), dimensions (coordinates for analyzing a fact), hierarchies (descriptions of each dimension at progressively coarser levels of aggregation), and measures (numerical attributes that quantify a fact), and it is commonly adopted for representing data to support the decision-making process. Though multidimensional modeling has been widely investigated, requirements elicitation is still an open issue mainly due to the poor knowledge end-users have of the multidimensional model on the one hand, to the lack of a domain language shared with designers on the other. In the direction of bridging this gap, this paper proposes an approach to obtain a multidimensional schema from the language of the domain captured through a Language Extended Lexicon (LEL). LELs have been introduced as structured glossaries to describe the language used in the application domain, aimed at facilitating requirements elicitation in software engineering. **Methods:** Our approach consists of two steps. In the first one, end-users apply a set of derivation rules to the LEL in order to obtain draft multidimensional schemata. The second step relies on

the interaction of end-users and designers to review and edit these draft multidimensional schemata so as to obtain the final ones. **Results:** The approach is validated via an experiment made on a case study, showing that end-users who apply our rules tend to produce multidimensional schemata that are more correct than those produced by end-users who work freely. **Conclusion:** Our rules provide a structured context where subjectivity has a smaller impact than in the case of designing with no guidelines, thus effectively supporting the collaboration between end-users and designers.

Keywords: Language Extended Lexicon, Multidimensional Modeling, Data Warehouse Design, Requirements Elicitation

1 Introduction

The *multidimensional model* is commonly adopted for representing data to support the decision-making process. It represents multidimensional data describing business phenomena to be analyzed using coordinates called *dimensions*. The metaphor normally adopted to illustrate multidimensional data is that of *cubes*, whose edges correspond to dimensions. For instance, a cube for analyzing car sales could have the sale date, the store, the car model, and the customer gender as dimensions. Analyzing such a cube means answering queries such as, for instance, how many cars of segment SUV where sold to female customers on each month of 2022 for each store region [1]. End-users who analyze cubes are normally the decision-makers in a company, such as CEOs, department managers, etc. In most cases, while end-users are business people with deep knowledge of their application domain, they do not have a significant ICT background.

Multidimensional modeling is the activity of designing multidimensional schemata[1], and it has a key role in enabling effective analyses of business data within different scenarios, namely, data warehouse design [1], situational analyses [2], and analysis of schemaless data [3]. Multidimensional schemata are designed based on the requirements expressed by the end-users who will analyze cubes [4]. Requirements elicitation typically relies on techniques such as interviews, questionnaires, user observation, workshops, brainstorming, role-playing, and prototyping [5]. Unfortunately, in practical cases, there are two major barriers to the adoption of these techniques. First of all, they ask that end-users have some understanding of the multidimensional model and good skills on online analytical processing (OLAP) —which is quite infrequent in organizations and companies. Secondly, their application is made complex and error-prone by the lack of a domain language shared by the end-users and

[1]Here we adopt the terminology commonly used in the database area, where the term *model* denotes a collection of concepts that can be used to describe a domain, while the term *schema* denotes an instance of a model, i.e., the description of a domain made according to a given model.

the designers. In this situation, end-users would really be interested in querying multidimensional data for decision-making, but their understanding of the multidimensional model is not sufficient to let them actively contribute to the design process. Besides, it is hard for designers to properly interpret and formalize requirements because they have little knowledge of the domain language spoken by end-users.

In the direction of making multidimensional modeling accessible to end-users with no knowledge of the multidimensional model and of bridging the terminological gap between end-users and designers, this paper proposes a novel approach that creates a multidimensional schema from the language of the domain captured through a *Language Extended Lexicon* (LEL). LELs were introduced in the 90's as a technique to capture and describe the language used in the application domain (Universe of Discourse) [6] [7] [8] [9]. A LEL is a glossary whose goal is to record the definition of terms that belong to a domain. LELs effectively capture and model the domain language because they conform to the mechanism used by the human brain to organize expert knowledge. There are some early reports about three significant characteristics of a LEL: it is easy to learn, it is easy to use, and it has good expressiveness [10]. Besides, a LEL can be used to obtain requirements [11]; thus, it can be considered as a first step in requirements elicitation.

Our approach consists of two steps. The first one relies on applying some rules to the LEL to obtain draft multidimensional schemata; this step is meant to be carried out by end-users, whose knowledge of the application domain lets them better understand the subtleties of the LEL and more effectively apply the rules. The second one relies on the interaction of end-users and designers to review and possibly edit these draft multidimensional schemata to obtain the final ones. Noticeably, the use of rules to obtain draft schemata from the LEL relieves the end-users and the designers from the effort of drawing schemata from scratch, as would be necessary in previous requirement-driven approaches to multidimensional modeling. Besides, the application of these rules can be automated to some extent by relying on Natural Language Processing (NLP) techniques [12].

The main difference between our approach and the previous requirement-driven ones, is that the latter lean on requirements that specifically concern multidimensional schemata and their querying, while the LEL describes the application domain in general and without any explicit reference to multidimensionality and user requirements. This makes our approach applicable even when end-users have no knowledge of the multidimensional model. Besides, there is a proposal to create the LEL in a collaborative way [13], hence our approach can cope with collaborative settings. This is important because multidimensional schemata are usually of interest to different business units inside an organization, and it is almost impossible to find one single expert who has detailed knowledge of the whole organization.

To validate our proposal, we made an experiment on a case study to answer the following research question: "Is the multidimensional schema produced using our approach more correct than the one produced without our approach?". The results indeed show that end-users who apply our rules to the LEL tend to produce schemata that are more correct than those produced by end-users who work freely.

The rest of the paper is organized in the following way. Section 2 describes some background needed to understand the proposal, and Section 3 discusses the related work. Section 4 describes the proposed approach relying on an example, while Section 5 provides a case study and a quantitative evaluation. Section 6 offers the conclusions.

2 Background

This section describes the main concepts behind our proposal. Specifically, we introduce the elements of the multidimensional model and LELs.

2.1 Multidimensional schemata

Multidimensional modeling has a key role for data analysis in different scenarios:

- **Data warehouse design.** A *data warehouse* (DW) is a database aimed at supporting decision-makers in analyzing useful data from heterogeneous sources [1]. The data in a DW are normally stored in multidimensional form and queried via OLAP tools; thus, multidimensional 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. The approach followed in DW systems is called *schema-on-write*, because multidimensional schemata are decided at design time and forced onto data at the time of writing them in the DW.
- **Situational analyses.** Schema-on-write approaches fall short 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. In this case a *schema-on-read* approach, which leaves data unchanged in their structure until they are accessed by the end-user, is preferred [2]. Though here the multidimensional schema is not decided at design time but at query time, multidimensional modeling is necessary to let end-users choose a useful perspective for analyzing data through OLAP tools.
- **Analysis of schemaless data.** *Data lakes* have been recently 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 [3]. Due to the evolving nature of schemaless data, adopting a schema-on-write approach would be complex,

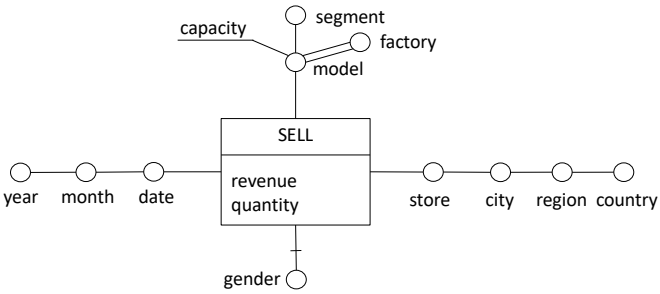

Fig. 1 The sales schema shown using the DFM notation

so even here a schema-on-read approach is preferred and multidimensional modeling becomes necessary to enable OLAP querying.

The multidimensional model relies on the concepts of fact, dimension, hierarchy, and measure. A *fact* is a relevant business phenomenon that decision-makers wish to monitor and analyze (e.g., car sales). *Dimensions* act as coordinates for identifying single occurrences of a fact (e.g., the date of a sale, the car model sold, and the store where it was sold). A dimension can be described at progressively coarser detail by a *hierarchy* of categorical *levels* (e.g., the store dimension includes levels city, region, and country); the levels in a hierarchy form a tree rooted in the dimension. Levels may have *properties*, i.e., (typically numerical) attributes that describe a level but should not be used for aggregation (e.g., the engine capacity of a model). Each fact occurrence is quantitatively described by some numerical *measures* (e.g., quantity of cars sold and the corresponding revenue). The possible values of levels are called *members* (e.g., Paris, Ile de France, France). A fact occurrence is related to one member for each dimension; so, for instance, it may state that 5 cars of model Mercedes-GLA were sold in Paris on May 4, 2020 to female clients.

A simple multidimensional schema for the car sales example is shown in Figure 1 using the Dimensional Fact Model (DFM) notation. The DFM is a conceptual model for multidimensional schemata [14]. It represents a fact using a box which, in its lower part, includes the list of measures. Levels are represented as circles, properties as horizontal lines; hierarchies are trees whose arcs normally model many-to-one associations (single line), but in some cases can also model many-to-many associations (*multiple arc*, shown by a double line). An arc is *optional* when its ending level is not always present. The arc from model to factory is *multiple* since one model can be manufactured by different factories, while the one to gender is *optional* since a client can be a company (Figure 1).

The metaphor normally adopted to illustrate multidimensional data is that of *cubes*, whose edges correspond to dimensions. Cubes are queried by slicing and dicing data and/or aggregating measures using aggregation functions (e.g., sum, average, min, max); the most used paradigm for querying cubes is *OLAP* (On-Line Analytical Processing), which provides intuitive operators such as

Table 1 LEL categories and their description

Category	Characteristics	Notion	Behavioral responses
Subject	Active elements (people or organization) that perform actions	Characteristics or condition that subject satisfies	Actions that subject performs
Object	Passive elements (resources, tools, data) on which subjects perform actions	Characteristics or attributes that object has	Actions that are performed on object
Verb	Actions that subjects perform on objects	Goal that the verb pursues	Steps needed to complete the action
State	Situations in which subjects, objects or even verbs can be located	Situation represented	Actions that must be performed to change into another state

roll-up (which aggregates data), *drill-down* (which disaggregates data), and *slice-and-dice* (which selects subsets of data).

2.2 Language Extended Lexicon

The Language Extended Lexicon (LEL) is a glossary that pursues the goal of recording the definition of terms that belong to a domain in order to "understand the language of a problem without worrying about the problem" [6] [7] [8] [9]. In order to develop an IT solution it is necessary to understand the goals, rules, and dynamics of the domain where the solution will be used. A prerequisite to understanding the domain consists in learning the language used in that domain: this is why building a LEL is so important. Many people inside an organization have some knowledge of the domain, for example experts, end-users, and customers. All of them have different and complementary points of view. Thus, the LEL must provide a unified and consistent representation of the language used by them. Although technical and specific domains use specific words, people tend to use synonyms. Moreover, hyponyms and hypernyms are also used. For example red is a color, and cherry could be a synonym. Blue is another color and blue and red are hyponyms while color is a hypernym. There are approaches to create the LEL collaboratively [13] that can effectively deal with the identification and clarification of the synonyms.

Terms (called *symbols* within the LEL) are defined through two attributes: *notion* and *behavioral responses*. Notion describes the symbol from inside, that is, intrinsic and substantial characteristics. Behavioral responses describe the symbol from outside, stating the relationships among the term being described and other terms. Each symbol of the LEL belongs to one of four categories: *Subject*, *Object*, *Verb*, and *State*. This categorization helps to organize the concepts of the domain and provides a template to describe the attributes. Table 1 shows each category with its characteristics and how to describe them.

Our running example refers to the car sales domain. Table 2 is an excerpt from the LEL (only notions are included for each symbol, since behavioral responses are not used by our approach) and shows, for the different categories, some symbols that will be used to exemplify the rules in Section 4.2.

Table 2 An excerpt of the LEL for car sales

Term / Expression	Notion
Verb: Sell a car	Operation in which a client pays a price to obtain a car on a date in a store.
Verb: Deliver a car	Operation in which a store prepares a car (physically and administratively) on a date to transfer its custody to a client.
Object: Price	Amount of money that the client must pay for the car.
Object: Car	Four-wheel vehicle that may have different packages and can be used either privately or commercially. A car has a model.
Object: Model	A car design that belongs to one segment. A model has an engine capacity and is manufactured in one or more factories.
Object: Segment	A category that groups different car models according to their size, use, and capacity.
Object: Factory	A place where cars are manufactured.
Object: Capacity	The engine displacement or size. It is the measurement of the total volume of the cylinders normally expressed in cubic centimeters or litres, e.g., 1800 cc.
Object: Store	Facility where the purchase has been done. A store is located in one city.
Object: City	The city where the purchase has been done. A city belongs to a region.
Object: Region	The region where the purchase has been done. A region belongs to a country.
Object: Country	The country where the purchase has been done.
Object: Date	The day when the purchase has been done.
Subject: Client	A person or organization. A client may be described by gender and age.
Subject: Gender	Male or female.

3 Related work

3.1 Multidimensional modeling

The approaches to multidimensional modeling are normally classified into four categories: data-driven, requirement-driven, mixed, and query-driven. Data-driven approaches design multidimensional schemata starting from the data sources (described for instance through Entity/Relationship diagrams, relational schemata, or XML schemata); user requirements impact on design by allowing the designer to select which chunks of data are relevant for decision-making and by determining their structuring according to the multidimensional model [15]. Requirement-driven approaches must be used when the source schemata are either not available at design time, or too complex, or poorly designed; they start from determining the information requirements of end-users, and how to map these requirements onto the available data sources is investigated only afterwards [16]. Mixed approaches combine data- and requirement-driven ones; in this case, both data source schemata and user requirements are used at the same time [17]. Finally, in query-driven approaches, a multidimensional schema is created starting from the set of OLAP queries the end-users are willing to formulate; these queries are specified using either SQL statements [18], MDX expressions [19], pivot tables [20], or query trees [21]. The reader is referred to [22] for a comprehensive survey of multidimensional modeling methods, and to [23] for a cost-benefit analysis of these methods.

Interestingly, some approaches introduce additional ingredients —besides source data, user requirements, and queries— in the modeling process. For instance, in [24] an Integrated-Process-Driven approach is proposed that integrates a data-driven, a requirement-driven, and a process-driven part, the latter taking into account the organizational processes that will use the multidimensional data. Giorgini et al. [25] describe a hybrid, goal-oriented approach that extends the Tropos methodology to let decision-makers specify their goals and derive multidimensional schemata from these goals.

An attempt to use NLP to support requirements elicitation is done in [26], which can be regarded as a query-driven approach since it starts from natural-language descriptions of the end-users analyses. NLP is also used in [27], where OLAP requirements are expressed in natural language via a specification template to be then used for deriving multidimensional schemata; to ensure the quality of the result, they use a data dictionary to remove ambiguities and resolve conflicts.

A mixed approach based on ontologies is described in [28]. Ontologies are used both to formalize users' requirements and to describe data sources; then a reconciliation step allows the multidimensional schema to be generated. Domain ontologies are also used in pure data-driven approaches as a starting point to discover multidimensional schemata [2, 15, 29] Noticeably, creating glossaries in a narrative way (as done with LELs) is recognized to be easier than creating ontologies [30–32].

Although our approach can be classified as requirement-driven, it is significantly different from the previously mentioned approaches. Indeed, while the latter lean on requirements that specifically concern multidimensional schemata and their querying, the LEL describes the application domain in general terms, without any explicit reference to multidimensionality and user requirements. This makes our approach applicable even when end-users have no knowledge of the multidimensional model.

3.2 LEL

LELs have been used as input of many different approaches aimed, for instance, at estimating the size of a software, at deriving descriptions of the domain, and at creating models of the software ranging from requirements to design.

As to estimation, Antonelli et al. proposed an approach to estimate the complexity of an application domain [33], while Centeno et al. proposed an approach to estimate the size of a software system [34].

As to deriving descriptions of a domain, different approaches were devised in the literature. Oliveira et al. [35] derive I* models, in particular they identify goals adopting a bottom-up approach. Breitman et al. [36] derive ontologies. Garrido et al. [37] derive models of mathematical problems. Barbosa et al. [38] consider the LEL as a shared communicative artifact during software design, so they propose an extension of the LEL to act as an interlingua that captures the shared understanding of both stakeholders and designers.

As to requirements, Cysneiros et al. [39] propose an approach to derive non-functional requirements. Antonelli et al. [11] derive functional requirements as use cases and user Stories. Mighetti et al [40] propose an approach to deal with global software development.

Also some approaches to deal with software architectures were proposed. Antonelli et al. [41] identify crosscutting concerns. Almentero et al. [42] obtain software modules. Mauco et al. [43] provide a set of heuristics which show how to derive types and functions, and how to structure them in a layered architecture.

Razafindramintsa et al. [44] provide an approach to software design using UML models, namely, class diagrams and state machines. Their proposal enriches the LEL description (notion and behavioural responses) with attributes and methods and they call this new LEL metamodel eLEL (extended LEL). Specifically, they propose an approach to derive class diagrams from an eLEL. A later work [45] derives also state machines. Another work [46] enriches the eLEL metamodel described in [44] to provide an additional description of attributes and methods (types and return types). Finally, Tarehy et al. [47] base their approach on [45] to add a repository with the goal of reusing components.

The LEL structures the knowledge of the domain in four types of elements: *subjects*, *objects*, *verbs*, and *states*, which can be mapped onto elements generally present in the artifacts used for software development: actors, entities, behaviour, and states. Thus, most of the previous works use the categorization of the domain proposed by a LEL to translate the knowledge captured by it to other artifacts, such as ontologies, goal-oriented models, functional requirements, and software models. The techniques adopted to this end are rules, heuristics, and algorithms. This is indeed the essence of the approach we propose, since we start from the elements of LELs to derive, by applying rules, elements of an artifact. However, ours is the first approach where the artifact derived is a multidimensional schema.

4 LEL-Driven Multidimensional Modeling

This section presents our proposal. It is organized in four subsections. The first one gives an overview of the approach, the second one describes the rules. The third and fourth subsections describe the two steps of our approach, namely, the strategy to apply the rules and how the resulting draft multidimensional schemata are reviewed.

4.1 The approach in a nutshell

The approach proposed, summarized in Figure 2, consists of a pipeline of two steps, where the LEL is the input and one or more multidimensional schemata are the output. Importantly, two distinct roles are involved: end-users and designer. End-users know the application domain well and they will access the

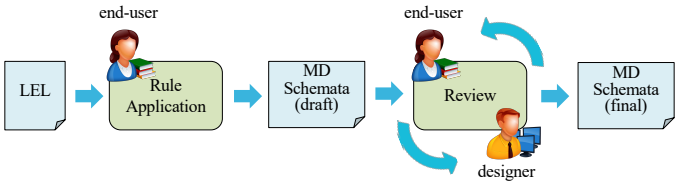

Fig. 2 The approach in a nutshell

data based on the multidimensional schemata; they can be analysts, decision-makers, data scientists, etc. Designers may have no or poor knowledge of the application domain but they are skilled in multidimensional modeling.

The steps of the approach are the following:

Step 1. **Rule application.** This step relies on the application of a set of derivation rules to a LEL describing the language of the domain in order to obtain elements to build one or more draft multidimensional schemata. For instance, facts are derived by analyzing verbs in the LEL. The rules are introduced and justified in Section 4.2. Though, as already mentioned, this step is meant to be carried out by end-users, in Section 6 we explain how the application of these rules could be automated, possibly by relying on NLP techniques.

Step 2. **Review.** This iterative step, described in Section 4.4, relies on the collaboration among designers and end-users, so that the former can adjust the multidimensional schemata according to the needs of the latter. For instance, some levels obtained by applying the rules may be deemed uninteresting by end-users for their analyses. Note that an editing step, where the schemata are manually adjusted to better fit the end-user's requirements, is part of all the existing methods for multidimensional modeling, even of those that automatically create a draft multidimensional schema starting from the schema of data sources (e.g., [14]).

Note that the creation of the LEL is outside the scope of this proposal; there are several works that explain how to create the LEL [13, 44, 48, 49]. It is also important to mention that the multidimensional schemata created through our approach are conceptual, i.e., implementation-independent; thus, as typically done in database design methods, they must be translated into logical schemata (e.g., star schemata) by applying the well-known best practices for logical design.

4.2 Derivation rules

This section introduces the rules used to derive multidimensional elements from a LEL in order to create one or more draft schemata. These rules, summarized in Table 3, provide a synthesis of different approaches to multidimensional modeling [26–28, 50–53]; they analyze symbols and their notions to derive different multidimensional elements: facts, measures, dimensions, etc. For example, Rules 2 and 3 relate a fact with its measures and dimensions,

Table 3 Rules summary

Rule	Description
1. Verbs give origin to facts	Let v be a verb of the LEL, then v should be defined as a fact.
2. Numerical objects and subjects of verbs give origin to measures	Let v be a verb defined as a fact according to Rule 1 and n be its notion. Let M be the set of objects and subjects in n that denote numerical attributes, then all the elements in M should be defined as measures of the fact corresponding to v.
3. Categorical objects and subjects of verbs give origin to dimensions	Let v be a verb defined as a fact according to Rule 1 and n be its notion. Let D be the set of objects and subjects in n that denote categorical attributes, then the elements on D should be defined as dimensions of the fact corresponding to v.
4. Categorical objects and subjects of objects or subjects give origin to levels	Let o be an object or subject defined as a dimension (according to Rule 3) or level (according to Rule 4) and n be its notion. Let L be the set of objects and subjects in n that have not been defined as dimensions, denote categorical attributes, and are related to o by aggregation semantics; then the elements in L should be defined as children levels of o.
5. Numerical objects and subjects of objects or subjects give origin to properties	Let o be an object or subject defined as a dimension or level according to Rules 3 or 4 and n be its notion. Let L be the set of objects and subjects in n that denote numerical attributes, then the elements in L should be defined as properties of o.
6. Plural objects and subjects give origin to multiple arcs	Let o be an object or subject defined as a dimension or level, n be its notion, and o' an object or subject in n defined as a child level of o. If o' is plural, then the arc from o to o' is a multiple one.
7. Expressions of possibility in objects and subjects determine optional arcs	Let o be an object or subject defined as a dimension or level, n be its notion, and o' an object or subject in n defined as a child level of o. If the verb used in n to relate o with o' suggests that some instances of o may not be associated to every instance of o', then the arc from o to o' is an optional one.

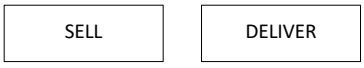

Fig. 3 DFM schemata of the facts derived from the LEL

respectively; they were formalized by observing that measures and dimensions provide, respectively, quantitative and qualitative descriptions of facts [26, 50]. Although the LEL is not as formal as an ontology in describing knowledge, it organizes the concepts of the domain in categories, namely, subjects, objects, and verbs. These categorization is used by the rules proposed in order to identify elements and relationships [28, 51]. Rules may also take into account some specific syntax and semantics expressed in the notion (e.g., Rule 4 relies on the identification of aggregation semantics to identify levels [27]).

The description of each rule is complemented with examples based on the car selling business and with a short justification.

Rule 1. Verbs give origin to facts.

Rule: Let v be a verb of the LEL, then v should be defined as a fact.

Justification: The verbs of the glossary describe the activities of the organization. A fact corresponds to an activity that occurs in the organization, and end-users are interested in monitoring and analyzing.

Example: The car sales glossary shows two verbs: *sell* (a car) and *deliver* (a car). Both these verbs should be considered as facts. Table 4 includes an excerpt from the LEL, Figure 3 shows the facts.

Table 4 Example of derivation of facts

Term / Expression	Notion
Verb: **Sell** a car	Operation in which a client pays a price to obtain a car on a date in a store.
Verb: **Deliver** a car	Operation in which a store prepares a car (physically and administratively) on a date to transfer its custody to a client.

Table 5 Example of derivation of measures

Term / Expression	Notion
Verb: Sell a car	Operation in which a client pays a **price** to obtain a car on a date in a store.
Object: **Price**	Amount of money that the client must pay for the car.

Fig. 4 DFM schema of the SELL fact enriched with measures

Rule 2. Numerical objects and subjects of verbs give origin to measures.

Rule: Let v be a verb defined as a fact according to Rule 1 and n be its notion. Let M be the set of objects and subjects in n that denote numerical attributes, then all the elements in M should be defined as measures of the fact corresponding to v.

Justification: Measures quantify occurrences of facts. Objects and subjects describe attributes or characteristics of activities. Numerical objects and subjects are candidates to be measures.

Example: The verb *sell* has the following notion: "Operation in which a client pays a price to obtain a car on a date in a store". This description shows one subject, *client*, and four objects: *price*, *car*, *date*, and *store*. Here, only price is numerical, so it is considered as a measure of fact SELL. Table 5 includes an excerpt from the LEL, Figure 4 shows the SELL fact enriched with its measure.

Rule 3. Categorical objects and subjects of verbs give origin to dimensions.

Rule: Let v be a verb defined as a fact according to Rule 1 and n be its notion. Let D be the set of objects and subjects in n that denote categorical attributes, then the elements on D should be defined as dimensions of the fact corresponding to v.

Justification: Dimensions are categorical attributes that act as coordinates for facts. Objects and subjects describe attributes or characteristics of activities. Non-numerical objects and subjects are candidates to be dimensions.

Example: The verb *sell* has the following notion: "Operation in which a client pays a price to obtain a car on a date in a store". This description shows one subject, *client*, and four objects: *price*, *car*, *date*, and *store*. Except for price, which is numerical, the other attributes are categorical and

Table 6 Example of derivation of dimensions

Term / Expression	Notion
Verb: Sell a car	Operation in which a **client** pays a price to obtain a **car** on some **date** and in some **store**.
Object: **Car**	Four-wheel vehicle that may have different packages and can be used either privately or commercially. A car has a model.
Object: **Store**	Facility where the purchase has been done. A store is located in one city.
Object: **Date**	The day when the purchase has been done.
Subject: **Client**	A person or organization. A client may be described by gender and age.

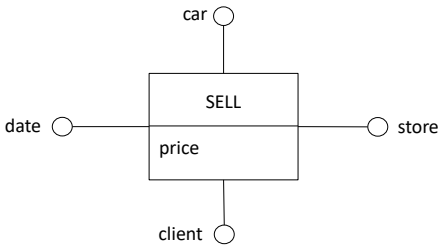

Fig. 5 DFM schema of the SELL fact enriched with dimensions

are considered as dimensions of fact SELL. Table 6 includes an excerpt from the LEL, Figure 5 shows the SELL fact enriched with its dimensions.

Rule 4. Categorical objects and subjects of objects or subjects give origin to levels.

Rule: Let o be an object or subject defined as a dimension (according to Rule 3) or level (according to Rule 4) and n be its notion. Let L be the set of objects and subjects in n that (i) have not been defined as dimensions, (i) denote categorical attributes, and (iii) are related to o by aggregation semantics; then the elements in L should be defined as children levels of o.

Justification: The different levels within a hierarchy are related by associations expressing aggregation. Objects and subjects in the notion of an object or subject may describe aggregation relationships using synonyms such as belongs to, is comprised by, and has a. Other relation semantics such as constitutes, is covered by, is incorporated in, involves, etc. can also give origin to aggregations.

Example: Car is a dimension, and its notion mentions object *model*. Since "a car has a model", part-of semantics is expressed; so model is a child level of car. In turn, the notion of *model* specifies that a model "belongs to one segment" and "is manufactured in one or more factories". Thus, segment and factory are children levels of model (see Table 7 and Figure 6).

Rule 5. Numerical objects and subjects of objects or subjects give origin to properties.

Rule: Let o be an object or subject defined as a dimension or level according to Rules 3 or 4 and n be its notion. Let L be the set of objects and subjects in n that denote numerical attributes, then the elements in L should be defined as properties of o.

Justification: Properties give further descriptions of a level. Notions state

Table 7 Example of derivation of levels

Term / Expression	Notion
Object: Car	Four-wheel vehicle that may have different packages and can be used either privately or commercially. A car has a **model**.
Object: **Model**	A car design that belongs to one **segment**. A model has an engine capacity and is manufactured in one or more **factories**.
Object: **Segment**	A category that groups different car models according to their size, use, and capacity.
Object: **Factory**	A place where cars are manufactured.
Subject: Client	A person or organization. A client may be described by **gender** and age.
Subject: **Gender**	Male or female.

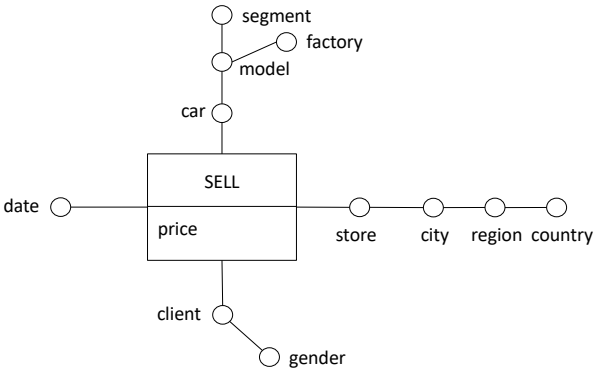

Fig. 6 DFM schema of the SELL fact enriched with levels

Table 8 Example of derivation of properties

Term / Expression	Notion
Object: Model	A car design that belongs to one segment. A model has an engine **capacity** and is manufactured in one or more factories.
Object: **Capacity**	The engine displacement or size. It is the measurement of the total volume of the cylinders normally expressed in cubic centimeters or litres, e.g., 1800 cc.

descriptions of the symbol, typically characteristics of a subject or object. It is common to use the verb "has", "possess", "owns" or a synonym.

Example: The notion of level model says that "a model has an engine capacity". Attribute *capacity* is numerical, so it is a property (see Table 8 and Figure 7).

Rule 6. Plural objects and subjects give origin to multiple arcs.

Rule: Let o be an object or subject defined as a dimension or level, n be its notion, and o' an object or subject in n defined as a child level of o. If o' is plural, then the arc from o to o' is a multiple one.

Justification: Natural language expresses relationship cardinalities in an accurate way. For example, in the expression "entity-1 belongs to one or more entity-2", entity-2 should be plural, thus it clearly defines the cardinality towards entity-2. Nevertheless, the reciprocal cardinality cannot be

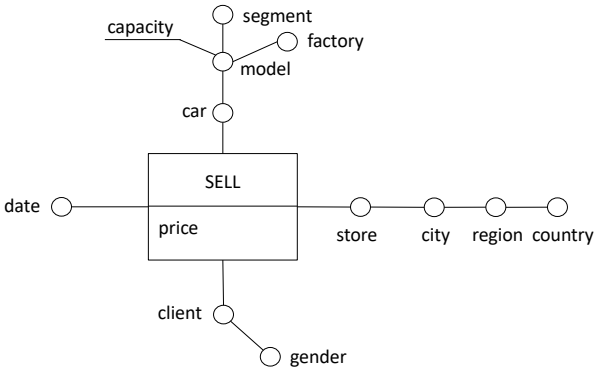

Fig. 7 DFM schema of the SELL fact enriched with properties

Table 9 Example of derivation of multiple arcs

Term / Expression	**Notion**
Object: Model	A car design that belongs to one segment. A model has an engine capacity and is manufactured in **one or more factories**.

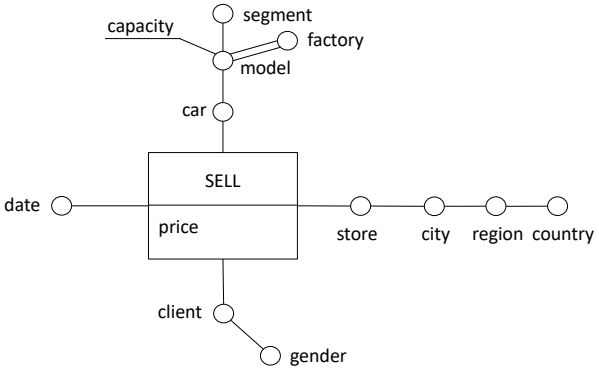

Fig. 8 DFM schema of the SELL fact enriched with multiple arcs

inferred, unless it is explicitly stated.

Example: Model and factory are objects defined as levels. The notion of *model* says that "a model is manufactured in one or more factories". Thus, the corresponding arc is multiple (see Table 9 and Figure 8).

Rule 7. Expressions of possibility in objects and subjects determine optional arcs.

Rule: Let o be an object or subject defined as a dimension or level, n be its notion, and o' an object or subject in n defined as a child level of o. If the verb used in n to relate o with o' suggests that some instances of o may not be associated to every instance of o', then the arc from o to o' is an optional one.

Justification: An optional arc expresses the fact that some members of a level are not associated to any member of a child level. Objects and

Table 10 Example of derivation of optional arcs

Term / Expression	Notion
Subject: Client	A person or organization. A client **may be described by gender and age**.

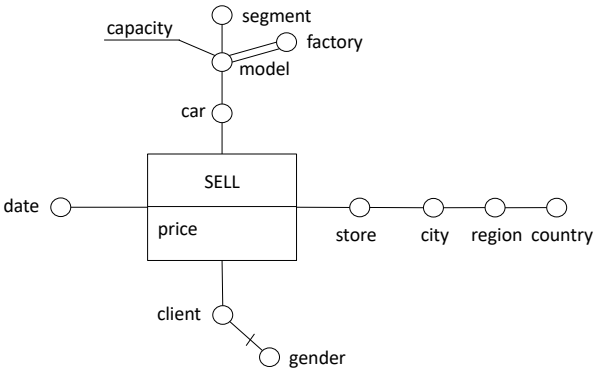

Fig. 9 DFM schemata of the SELL fact enriched with optional arcs

subjects in the notion of an object or subject may describe non-mandatory relationships using expressions of possibility such as "may".

Example: Client and gender are objects defined as levels. The notion of *client* says that "a client may be described by gender". Thus, the corresponding arc is optional (see Table 10 and Figure 9).

4.3 Step 1: Rule application

The procedure used to apply the rules described above can be outlined as in Algorithm 1, which takes the LEL glossary as input and provides a draft multidimensional schema as output (as in Figure 2). The algorithm starts by applying Rule 1 to find the facts (Line 1). Then, for each fact, it applies Rules 2 and 3 to find its measures and dimensions (Lines 4, 5). The LEL is then iteratively navigated by applying Rules 4 and 5 aimed at building hierarchies (Lines 9, 10); Rules 6 and 7 are then triggered to recognize multiple and optional arcs (Lines 12, 13). Note that, to enable a uniform application of Rule 4, dimensions are treated as levels. Besides, since properties cannot have children (i.e., they are always leaves in multidimensional schemata), Rules 4 and 5 are not further applied to them.

There are three specific situations that must be dealt with in applying Algorithm 1:

1. Let o' be a subject/object in the notion of o, where o has been defined as a level and o' as its child level by applying Rule 4. It may be the case that the notion of o' mentions o (for instance, the LEL may define a city saying that it is part of a region, and a region saying that it includes many cities); in this case the algorithm would loop trying to add o as a

Algorithm 1 Rule application

Require: a LEL
Ensure: set of facts F, sets of measures/dimensions $M_f, D_f \, \forall f \in F$, set of levels L
1: apply Rule 1 to the LEL, get set F of facts
2: $L \leftarrow \varnothing$
3: **for each** $f \in F$, corresponding to verb v: **do**
4: apply Rule 2 to v, get set M_f of measures, add them to f
5: apply Rule 3 to v, get set D_f of dimensions, add them to f
6: $L \leftarrow L \cup D_f$
7: **end for**
8: **repeat**
9: **for each** $l \in L$, corresponding to object/subject o: **do**
10: apply Rule 4 to o, get set C_l of levels, add them to l as children levels
11: apply Rule 5 to o, get set P_l of properties, add them to l as children levels
12: **for each** $l' \in C_l$, corresponding to object/subject o': **do**
13: apply Rule 6 to o and o', possibly change the arc from l to l' to multiple
14: apply Rule 7 to o and o', possibly change the arc from l to l' to optional
15: **end for**
16: $L :\leftarrow L \setminus \{l\} \cup C_l$
17: **end for**
18: **until** no new levels are added to L
19: **return** F, L, $\{M_f, f \in F\}$, $\{D_f, f \in F\}$

child level of o'. Hierarchies cannot have cycles in the multidimensional model (except in the case of *recursive hierarchies*, which are seldom used in practice so they are not considered here), so the algorithm detects the loop and breaks it without adding o as a child of o'.

2. It is often the case that the same object/subject o is mentioned in the notion of multiple objects/subjects. For instance, the LEL may define a customer saying that she lives in a city, and it also may define a store saying that it is located in a city. In the multidimensional model, this may give rise to a *shared hierarchy*, i.e., a portion of hierarchy that is reused twice or more. In this situation, the algorithm applies Rules 4 and 5 to o only once.
3. As a particular case of the previous situation, it may happen that some redundancies are present in the LEL so that a transitive arc is created within a hierarchy. For instance, as shown in Figure 10, the notion of symbol *date* may mention *month* and *year* (which are both added as children of dimension date), and the notion of *month* may in turn mention *year* (which is then added as a child of level month). The arc from date to *month* expresses a transitive functional dependency, so it is redundant and must be eliminated.

The result of applying Algorithm 1 to the LEL of Table 2 is shown in Figure 9 (with reference to the SELL fact only, the DELIVER fact is very similar). Precisely, after finding the facts and their measures and dimensions, Rules 4-7 are repeatedly applied to build a hierarchy for each dimension.

We complete this sections with some observations about the complexity of Algorithm 1. At most one among Rules 1 to 5 can be applied to a symbol in the LEL, so the complexity of the first part of the algorithm (Lines 1 to 10)

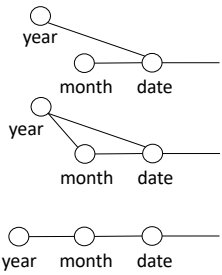

Fig. 10 Deleting transitive arcs in a hierarchy

is, in the worst case, given by the size of the LEL. On the other hand, Rules 6 and 7 may be applied to any object or subject in the LEL. Thus, an upper bound to the number of rules to be applied is given by three times the size of the LEL. This is clearly not critical, even more if you consider that a real LEL is usually no larger than a few hundreds of symbols.

4.4 Step 2: Review

To create multidimensional schemata, our approach does not rely on the schemata of data sources (as done in data-driven approaches to modeling), nor on the identification of the end-users' goals (as done in goal-driven approaches to modeling). It just requires that a LEL is available to describe the application domain. However, the draft schemata obtained by applying the rules will most probably need to be reviewed and edited to tightly suit the end-users' needs. Note that this refinement activity is also required in all the other approaches to multidimensional modeling devised in the literature (e.g., [14, 51]), and it is commonly performed with the joint participation of designers and end-users. The former have good knowledge of multidimensional modeling, while the latter are expert in the application domain.

The review step is carried out iteratively. The possible actions to be agreed between end-users and designers to edit the draft multidimensional schemata are listed below with some examples [54]:

#1 **Rename facts/measures/dimensions/levels/properties**. Renaming a multidimensional element may be beneficial when the name used in the draft schema is not perfectly clear to end-users and can be misunderstood. For instance, measure price in our working example could be renamed to revenue.

#2 **Drop facts/measures**. Dropping a fact may be necessary because not all verbs in the LEL describe processes that are deemed interesting for analysis. For instance, the LEL may include a verb *Login* to describe the authentication process of end-users; however, most probably analyzing end-users' logins will not be interesting for end-users. A measure in the draft schema may be dropped if it is not considered useful for analyses or if it can be computed from other measures (*derived measures*).

#3 **Drop dimensions/levels/properties**. As for measures, some dimensions, levels, and properties in the draft schema may be uninteresting for end-users, in which case they can be removed. However, deleting a dimension or a level requires some further explanation. As stated in [14], an uninteresting dimension or a level can be either *pruned* or *grafted*. In the first case, it is deleted from the schema together with all the subtree rooted in it. For instance, pruning level model from the schema in Figure 9 means removing model, factory, segment, and capacity. Pruning an entire subtree is quite uncommon; in most cases, a dimension or level is not interesting but its children are. Grafting a dimension d means removing d and having all its children (if any) become dimensions. For instance, grafting dimension car leads to making its child model become a dimension. Similarly, grafting a level l means removing l and having all its children (if any) become children of the father of l. For instance, grafting level region leads to making its child country become a child of city.

#4 **Add measures/dimensions/levels/properties**. Some useful multidimensional elements might not be mentioned in the LEL, in which case they should be added at this stage. For instance, assuming that a client can buy two or more cars at a time, a quantity measure could be added to the SELL fact, while two new levels month and year could be added as children of date.

#5 **Edit arcs**. Some editing of arcs may be required as well, to correct wrong interpretations of sentences in the LEL. This includes: (i) changing the father of a level; (ii) making an optional arc non-optional, or vice versa; (iii) making a single arc multiple, or vice versa.

#6 **Add levels by discretizing properties**. As already stated, properties are numerical descriptions of a level (e.g., the engine capacity of a car model). However, in some situations the end-users might prefer to discretize a property by defining ranges of values, to enable the use of these ranges for aggregation. For instance, capacities could be discretized into three ranges (less than 1500cc, between 1500cc and 2000cc, more than 2000cc), so the capacity property can be transformed into a capacity range level with three possible members.

#7 **Evaluate measure additivity**. A measure is said to be *additive* along a hierarchy if it can be safely summed when aggregating along that hierarchy. While this is true in most cases, some measures are *semi-additive* since they cannot be summed when aggregating along temporal hierarchies (e.g., a measure expressing the level of inventory), while others cannot be summed at all and are called *non-additive* (e.g., the exchange rate between two currencies). Detecting semi- and non-additive measures requires considering the semantics of measures, so it cannot be automated.

Interestingly, although there is no obvious automatism to trigger these actions, most of them can effectively be suggested in different ways:

- A first way is by an *early workload test* [55], which aims at verifying that the analysis queries end-users are interested in are actually supported by

the multidimensional schemata designed. Specifically, this implies checking, for each query, that the required measures have been included in the multidimensional schema and the required aggregation levels can be expressed as a valid group-by. This type of test can give precise indications to execute actions #1, #4, and #6. To some extent, it can also be used to trigger actions #2 and #3 (if a multidimensional element is used by no workload query, it could possibly be dropped).

- A second way is by a *hierarchy test* [55], which requires writing SQL queries to verify that the functional dependencies represented by hierarchies in the multidimensional schema (e.g., city → region) actually hold in the company data. This type of test can give precise indications to execute action #5.
- Additivity issues (action #7) can be detected during a *functional test of the front-end*, which selects a significant sample of queries and asks end-users to validate their results. Although this test can be conducted only at a late stage of design, correcting additivity errors has a small impact on the implementation so it does not significantly affect productivity.
- Finally, a set of metrics has been devised in the literature to assess the *quality* of a multidimensional schema. For instance, the *conformity factor* [55] measures to what extent dimensions are shared by different facts, and can be used to show that the designer failed to recognize the semantic and structural similarities between apparently different hierarchies or facts, thus triggering actions #2 and #3. In the same direction, by measuring the *similarity factor* one can infer that two facts are mainly overlapping, or that one of them is mostly included in the other, thus triggering action #2. Other useful metrics in this context are those introduced by [56], related to the *understandability* of a multidimensional schema, that may show that a schema is overly complex thus triggering actions #2 and #3.

In our working example, the draft schema in Figure 9 is transformed into the final schema of Figure 1 by applying the following actions:

1. Rename measure price into revenue for better end-users' understanding.
2. Drop dimensions client and car, whose aggregation level is considered to be too detailed.
3. Add measure quantity, which is useful for analyses.
4. Add levels month and year to enable interesting temporal aggregations.

Measure price is additive, so no further actions are required.

5 Validation

We have performed a validation of the proposed approach using a case study in the healthcare domain, whose (simplified) LEL is shown in Table 11. The research question to be answered is: *Do the rules based on LEL improve the correctness of the DFM schema produced?*

We have operated in two steps:

Table 11 The LEL for drug administration

Term / Expression	Notion	Result
Verb: Administer	Action performed by a doctor, that consists in dispensing at a certain date and hour a dose of a drug to deal with some disease of a patient in some seriousness in order to obtain some outcome.	Rule 1: fact Administer Rule 2: measure Dose Rule 3: dimensions Patient, Drug, Doctor, Disease, Outcome, Hour, Date, Seriousness
Object: Dose	Amount of a drug that is taken once, or regularly over a period of time.	
Subject: Patient	Person who is ill or hurt, characterized by gender and city.	Rule 4: levels Gender, City
Object: Drug	Medicine or other substance which has a physiological effect when ingested or otherwise introduced into the body according to its administration mode. The drug is also characterized by an elimination mode.	Rule 4: levels AdministrationMode, EliminationMode, PhysiologicalEffect
Subject: Doctor	A person with a medical degree whose job is to treat patients. The doctor works for a hospital and belongs to a Department.	Rule 4: level Department
Subject: Disease	Illness affecting humans.	
Object: Outcome	The result of a medical treatment.	
Object: Hour	One of the 24 parts that a day is divided into.	
Object: Date	A particular day of a month, in a particular year.	Rule 4: levels Month, Year
Object: Seriousness	Degree of risk of the life of a patient.	
Object: Administration mode	The way of giving a drug to somebody.	
Object: Elimination	The process of removing or getting rid of the drug completely.	
Object: Physiological effect	The expected result of administering a drug, it belongs to one family.	Rule 4: level Family
Object: Family	A group into which drugs that have similar characteristics are divided based on their physiological effect	
Subject: Department	A section of a hospital.	Rule 4: level Hospital
Subject: Hospital	An institution in which the sick or injured are given medical and surgical treatment. A hospital is located in a city.	Rule 4: level City
Object: City	A city belongs to a state.	Rule 4: level State
Object: State	A state belongs to a country.	Rule 4: level Country
Object: Country	An area of land that has or used to have its own government and laws.	
Object: Month	Any of the twelve periods of time into which a year is divided.	Rule 4: level Year
Object: Year	The period from January 1 to December 31.	
Object: Gender	A range of identities with reference to social and cultural differences.	

1. First, we have asked two of the authors of this paper to design a DFM schema starting from the LEL; the result obtained constitutes the ground truth for the next step.
2. Then, we have asked a set of 24 practitioners with no background on multidimensional modeling (who simulate end-users) to design a DFM schema starting from the same LEL, and we have compared the results (obtained with and without our approach) to the ground truth.

5.1 Setup

Two of the authors of this paper played the role of designers to provide the ground truth for the experiment by designing the DFM schema that correctly translates the knowledge captured by the LEL.

The other participants were 24 students of a degree course and a post-degree course of the Universidad Nacional de La Plata, Argentina. Most of the them had some practical experience in software development since, in Argentina, students generally begin to work in companies during the second year of their undergraduate studies. The age and the years of experience are very varied, since students can switch between studies and industry. It is important to mention that all the participants have volunteered for the activity.

The experiment aims at comparing the results of designing a DFM schema freely (control) against the results of designing a DFM schema using our approach (treatment). Thus, the 24 participants were randomly divided into two groups (control and treatment) and every participant performed the task in one way only (either freely or by applying the rules). The experiment was performed virtually, due to the pandemic restrictions in Argentina in 2021. The tasks of each group (control and treatment) were performed separately, on days and times agreed with the participants.

Both groups received an introduction about the DFM (30 minutes) and LELs (15 minutes), since both had to design a DFM schema using a LEL as input. It is important to mention that the experiment concerns the evaluation of the proposed approach (particularly, the rules) and not the creation of the LEL; thus, the same LEL was assigned to all participants. Only the treatment group received some additional training in the use of the rules proposed (20 minutes). Both groups had the same time (45 minutes) to design the DFM schema. Thus, the whole activity for one group took 110 minutes (65 minutes in training and 45 minutes in the requested task), while it took 90 minutes (45 minutes in training and 45 minutes in the requested task) for the other group.

We organized the experiment in four main steps; after each step, the participants should have created different elements of the DFM schema. The tasks provided specific scenarios in which the participants should have identified the elements. The steps were the following:

- Identification of facts
- Identification of measures (for one specific fact)
- Identification of dimensions (for one specific fact)
- Identification of levels (for some specific scenarios)

Each step above is related with one specific rule (Rules 1 to 4). Thus, the experiment only considers the rules aimed at identifying the most relevant elements, i.e., facts, measures, dimensions, and levels. There are three reasons to pick these elements: (i) they are those that most commonly appear in multidimensional schemata; (ii) their identification can be time-consuming; (iii) mistakes in the identification of these elements can have a disrupting

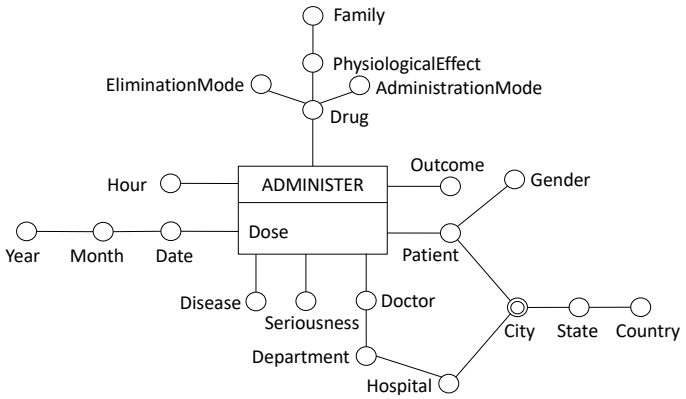

Fig. 11 Draft DFM schema of the ADMINISTER fact

impact on the final schema and would dramatically affect the identification of the remaining elements. Conversely, properties and optional arcs are excluded since they have a small impact on the final schema, while multiple arcs are excluded because they are seldom used in real multidimensional schemata. Finally, the reviewing step is not considered in the experiment since it has already been widely investigated in the literature (e.g., [14]).

5.2 Ground Truth

Having Algorithm 1 correctly applied by a designer to the LEL in Table 11 produces the draft multidimensional schema in Figure 11, which is used as ground truth. Some observations follow:

- The notion of *Drug* mentions its physiological effect, and vice versa. This loop is broken as suggested in bullet 1 of Section 4.3.
- Applying Rule 4 to subject *Hospital* leads to identifying child level City, which is also a child of Patient. As a result, as suggested in bullet 2 of Section 4.3, a shared hierarchy rooted in City is created (depicted in Figure 11 using a double circle).
- An arc from Date to Year would be created by applying Rule 4; according to bullet 3 of Section 4.3, this arc is transitive and so it is dropped.

5.3 Data Analysis

We evaluated the correctness of a DFM schema contrasting the one obtained via a free design (control) against the one obtained using our approach (treatment). Specifically, we compared the main multidimensional elements obtained by participants in the design activities in either way: control and treatment. We measured precision and recall to compare relevant elements identified in both ways against the ground truth. We classified the answers provided by the participants as true positive, false negative, true negative, and false positive. Table 12 summarizes these categories. Precision and recall are then defined as

Table 12 Answer categorization

	Identified by subject	**Omitted by subject**
Relevant element	True positive (correct identification of relevant element)	False negative (incorrect omission of relevant element)
Irrelevant element	False positive (incorrect identification of irrelevant element)	True negative (correct omission of irrelevant element)

Table 13 Precision and recall in identifying facts

	Group	**Min**	**Max**	**Average**	**Median**
Precision	Control	0.00	0.50	0.09	0.00
	Treatment	0.50	1.00	0.96	1.00
Recall	Control	0.00	1.00	0.42	0.00
	Treatment	1.00	1.00	1.00	1.00

follows:

$$Recall = \frac{TruePositives}{allRelevantElements} = \frac{TruePositives}{TruePositives + FalseNegatives}$$
$$Precision = \frac{TruePositives}{allIdentifiedElements} = \frac{TruePositives}{TruePositives + FalsePositives}$$

The information recorded for every participant was consolidated in a spreadsheet. Each participant was identified with an id and its type (control or treatment). Then, every concept identified during DFM schema design was recorded. The concepts were categorized according to whether they were true positives (correct), false negatives (omitted), or false positives (incorrect). The spreadsheet calculated the amount of answers for every type in order to obtain the number of true positive, false negative and false positive answers. With these numbers, the spreadsheet calculated recall and precision. This analysis was performed for every step of the experiment: identification of (i) facts, (ii) measures, (iii) dimensions, and (iv) levels.

We have performed a comparison of the distribution of the variables (precision and recall) for every step (facts, measures, dimension, and levels) and for every group (control and treatment). In order to do that, we calculated the minimum, maximum, average, and median values. The remainder of this section analyzes each step.

(i) **Identifying facts**. The domain described in the LEL had only one fact. Although this is a small amount, it is realistic. The precision of the control group is low (50% at maximum and 0.09% of average), while the precision of the treatment is high (0.96% of average). The recall of the control has a wide range (minimum of 0% and maximum of 100%), while the treatment is excellent (100% of average). The other values are summarized in Table 13.

Table 14 Precision and recall in identifying measures

	Group	Min	Max	Average	Median
Precision	Control	0.00	0.33	0.13	0.14
	Treatment	0.33	1.00	0.85	1.00
Recall	Control	0.00	1.00	0.58	1.00
	Treatment	1.00	1.00	1.00	1.00

Table 15 Precision and recall in identifying dimensions

	Group	Min	Max	Average	Median
Precision	Control	0.00	0.60	0.19	0.11
	Treatment	0.71	1.00	0.93	1.00
Recall	Control	0.00	0.50	0.21	0.17
	Treatment	0.50	1.00	0.80	0.80

(ii) **Identifying measures**. The domain description provided only one measure. The precision of the control group is low (33% at maximum and 0.13% of average) while the precision of the treatment is high (0.85% of average) although it has a wide range (minimum of 33% and maximum of 100%). The recall of the control has a wide range (minimum of 0% and maximum of 100%) with an average of 58%, while the treatment is excellent (100% of average). The other values are summarized in Table 14.

(iii) **Identifying dimensions**. The precision of the control group is really low (60% at maximum and 0.11% of median) while the precision of the treatment is high (0.93% of average and 100% of median) and it has a narrow range (minimum of 71% and maximum of 100%). The values of recall of the control are always below the values of recall of the treatment (maximum of the control 50% and minimum of the treatment 50%). Average and median of the control are 21% and 17%, respectively, while average and median of the treatment are 80% and 83%, respectively. The other values are summarized in Table 15.

(iv) **Identifying levels**. There were many levels as well as many distractors in the domain provided. Regarding precision, both groups had wide ranges (a length of 42% in the control group and a length of 33% in the treatment group). Nevertheless, the precision of the control is lower than the precision of the treatment considering average and median (66% average and 71% median of control, against 92% average and 92% median of treatment). The recall of the control has a wide range (minimum of 8% and maximum of 100%) with an average of 48%, while the treatment has a small range (minimum of 75% and maximum of 100%) with an average of 92%. The other values are summarized in Table 16.

Table 16 Precision and recall in identifying levels

	Group	Min	Max	Average	Median
Precision	Control	0.38	0.80	0.66	0.71
	Treatment	0.67	1.00	0.92	0.92
Recall	Control	0.08	1.00	0.48	0.42
	Treatment	0.75	1.00	0.92	0.92

5.4 Threats to validity

Wohlin et al. [57] group validity threats into four categories: conclusion, internal, construct, and external validity. The rest of this subsection analyzes different threats from each category.

Concerning the conclusion category, one possible threat is reliability of measures. Our evaluation uses precision and recall, two measures not biased with different points of view because the elements reported by participants either are or are not in the expected result. Another threat that belongs to this category is random heterogeneity of subjects. There is always heterogeneity in a study group. If the group is very heterogeneous, there is a risk that the variation due to individual differences is larger than the one due to the treatment. In our experiment, the participants are homogeneous considering that almost all of them have some experience in companies and all of them studied at a university.

The second category of threats to analyze is internal validity. Instrumentation is a threat that we were specifically concerned with, so we paid a lot of attention to the preparation of the artifacts for the experiment. We chose a realistic domain with a realistic description. Moreover, since nowadays homeworking is a common practice, the context in which the tasks were performed should not generate any threat. The maturation threat does not create any problem since the experiment duration was short and the experimental task, in particular, were very short. In this way, the participants would not get bored or tired from the experiment.

According to the construct validity category, we observed that the experiment did not suffer from such threats referred to as hypothesis guessing, evaluation apprehension, or experimenter expectations, because the participant only had to fill the template provided with elements belonging to a specification we provided.

Finally, Sjøberg et al. [58] state that many threats to external validity are caused by an artificial setting of the experiment. They mention the importance of realistic tasks and realistic subjects. Realistic tasks depend on the size, complexity, and duration of the tasks involved. Taking this into account, we set up an experiment that had the complexity of a simple but real situation. Specifically, we have based our decision of using a subset of rules instead of all the rules on [58], who claims that it is important to focus on the task more frequently performed. The time limit of 45 minutes was proportional to the size of the LEL; it was not longer than that since we wanted the task to be challenging for end-users. Although limiting the time could seem unrealistic,

note that in real settings end-users would probably not dedicate a long time to this task, which they might perceive as distracting them from their daily operations. Besides, although we validated the approach via a single domain, to effectively test the rules we edited the LEL so that several possible situations were covered. For example, Rules 2, 3, and 4 require to distinguish between numerical and categorical elements in the definition of the symbol, and there are plenty of them. Finally, the subjects' realism depends on how the subjects who perform the experimental tasks are selected. To cope with this threat, most participants we selected were practitioners with some experience in companies.

5.5 Discussion of the results

This section presents the conclusions we can draw with reference to the identification of the main multidimensional elements.

The conclusions we can draw are summarized below:

- **Identification of facts**. While identifying facts is quite simple for an expert in multidimensional design, it is not as easy for newly trained people. This was confirmed by our experiment: all participants had some background in the Entity/Relationship and Object-Oriented models, and those in the control group were biased by this background ending up by identifying many more facts than the ones actually present in the domain. Conversely, Rule 1 is very straightforward to follow, that is why the results of the treatment are almost perfect.
- **Identification of measures and dimensions**. This raises two challenges: that of detecting the relevant characteristics of the facts, and that of classifying them into measures (numerical attributes that can be used in mathematical operation to compute aggregate values) and dimensions (categorical values to be used for filtering and aggregating the data). The criteria and background needed to perform this task is not the simple knowledge that a software developer or an end-user can have: some experience in multidimensional design is required. Nevertheless, here the results of the control were better than the results of the control for identifying facts; we argue that this is because the concept of attribute of an entity was well known by all participants. The proposed rules help in automating this task; thus, not surprisingly, the results of the experiment suggest that participants who applied the rules performed better than the other participants. Specifically, Rule 2 is simple to follow but it needs some criteria from the end-user to distinguish numerical from categorical elements. This could raise an issue regarding time and dates, which include numbers but are not numbers. We believe this is the reason why the precision of the treatment was not as good as for the identification of facts. Rule 3 is complementary to Rule 2, since the same criteria to differentiate numerical from categorical elements should be used. Thus, the results of the treatment are related to those of measure identification.

- **Identification of levels**. This turned out to be the most complex activity both in the control and the treatment; indeed, it has the complexity of identifying dimension plus the one of sorting the levels according to their granularity (e.g., from Drug to PhysiologicalEffect to Family). Although this is the activity that produced the widest range of results, the experiment shows that better results are obtained by applying the rules. Indeed, as for the previous tasks, the rules provide a context where subjectivity has a smaller impact than in the case of performing the tasks with no guidelines. In some cases, the participants inverted the order of the levels; in others, they created a linear hierarchy while different branches were necessary (for example, between EliminationMode and AdministrationMode).

Overall, in the light of the above, we can answer the research question by stating that the correctness of multidimensional design is indeed improved by using the proposed rules.

6 Conclusions and future work

In this paper we have proposed an approach to obtain a multidimensional schema from the language of the domain captured through a LEL. The approach consists in a set of derivation rules that can be applied in a mostly straightforward way; thus, even end-users with no experience can follow them to obtain a schema. We have presented an experimental evaluation showing that end-users who apply our rules tend to produce schemata that are more correct than those produced by end-users who work freely.

The approach relies on a LEL that captures the domain knowledge; clearly, a good LEL provides a good multidimensional schema. The construction of a LEL asks for some extra effort, so the first contexts that will take advantage of our approach are those where LELs are already being used. In the future we plan to conduct further experiments aimed at comparing the effort for designing a DFM schema from scratch against the one for first building the LEL and then applying the proposed approach.

Another relevant direction for future work is to improve the definition of the rules to avoid false positives and negatives. Specifically, the main challenge in the application of the rules for measures and dimensions lie in correctly classifying numerical and categorical elements, which requires distinguishing between true numerical elements and categorical elements that contain numbers. Times and dates are examples of elements that contain numbers, but should be classified as categorical. As to the challenges about the identification of levels, the most critical issue is the one of putting the levels in the right order within the hierarchy. Indeed, each child level should correspond to a possible grouping of the members of its parent level. Thus, year is a child of month because it aggregates (contains) the months of each given year. Similarly, every month of a year contains a set of dates, so month is a child of date. This was not clearly explained in the rule definition; indeed, some participants wrongly built the time hierarchy without really considering this containment

relationship between levels. Thus, Rule 4 could be improved with a deeper description of the meaning of inter-level relationships.

Noticeably, while Algorithm 1 in itself can be easily coded, its full automation requires resorting to NLP tools [12, 26, 59] to check when the rules can indeed be applied. Though such issue is not in the scope of this paper, some preliminary hints are given below:

- Rule 1, which extracts facts, is easily implemented because it just collects all the symbols of the verb category, which is explicitly stated in the LEL.
- Similarly, Rules 2 and 3 deal with the subject and object categories, which are explicitly stated in the LEL as well. However, both rules rely on distinguishing whether a subject/object refers to numerical or categorical attributes. For example, *price* is a numerical attribute while *client* is a categorical one. One possible way to address this problem is by using a dictionary with keywords that refer to numerical attributes, such as measure, size, quantity, amount, length, height, depth, etc.; the symbols that include these keywords in their notion are considered to be numerical. Considering that definitions can include examples, another possibility is to check if the notion states any number.
- Rules 4 and 5 also need to distinguish between numerical and categorical elements. Moreover, these rules deal with aggregation relationships. Although the aggregation is implicit because the element is mentioned in the description, the use of a dictionary can provide more certainty. Thus, a dictionary can be used that considers expressions like "belongs to", "constitutes", "is covered by", "is incorporated in", "involves", and "has".
- Rule 6 needs to identify plural nouns (subjects or objects) —for example, "factories" in the following expression: "a car is manufactured in one or more factories". The Stanford NLP framework [60] provides a part-of-speech analysis, where a plural noun is tagged as "NNS" (for example "factories") while a singular noun is tagged as "NN" ("car").
- Finally, Rule 7 deals with expressions of possibility. Although simple analysis could be made using a glossary of modal auxiliaries (e.g., may, might, could, would, should), a more complete analysis would require some epistemic expressions [61].

NLP tools can be profitably used also to help build the LEL as well. We are currently assessing two different strategies to build the LEL: one is based on a chat bot that lists small pieces of text to build a LEL, while the other uses extensive documentation which allows reviewing the information many times when building a LEL. We are confident that the adoption of these tools will significantly contribute to improve the effectiveness of our approach and make it suitable to a larger number of contexts.

We close the paper by observing that LELs can be related to knowledge graphs. A LEL is composed by terms of category subjects and objects, which correspond to the nodes of a knowledge graph, while the terms of category verbs correspond to the arcs between the nodes of a knowledge graph.

Although some approaches were devised that deal with knowledge graphs and multidimensional modeling, for example for querying a knowledge graph [62] and building a knowledge graph using multidimensional modeling tools [63], we could not find approaches in the literature to build multidimensional schemata from a knowledge graph, nor to build knowledge graphs starting from a LEL. A promising direction to extend and improve our approach could be to take in input a combination of a LEL and a knowledge graph: the former would provide the knowledge about a specific domain and define its boundaries, while the latter could provide additional information for that domain, i.e., by explicitly representing inter-level relationships to be used for building hierarchies.

Conflict of interest

The authors declare that they have no conflict of interest.

References

[1] R. Kimball, M. Ross, The data warehouse toolkit: the complete guide to dimensional modeling, Wiley, 2002.

[2] E. Gallinucci, M. Golfarelli, S. Rizzi, A. Abelló, O. Romero, Interactive multidimensional modeling of linked data for exploratory OLAP, Information Systems 77 (2018) 86–104.

[3] E. Gallinucci, M. Golfarelli, S. Rizzi, Approximate OLAP of document-oriented databases: A variety-aware approach, Information Systems 85 (2019) 114–130.

[4] N. Prakash, D. Prakash, Data Warehouse Requirements Engineering, Springer Singapore, 2018.

[5] J. Dick, E. Hull, K. Jackson, Requirements Engineering, Springer International Publishing, 2017.

[6] J. Leite, A. Franco, A strategy for conceptual model acquisition, in: Proceedings of the IEEE International Symposium on Requirements Engineering, 1993, pp. 243–246.

[7] M. d. C. Leonardi, M. Ridao, M. V. Mauco, L. Felice, A natural language requirements engineering approach for mda, International Journal of Computer Science, Engineering and Applications (IJCSEA) 5 (1) (2015) 413–429.

[8] M. Urbieta, L. Antonelli, G. Rossi, J. C. Leite, The impact of using a domain language for an agile requirement management, Information and Software Technology 127 (2020) 106375.

[9] M. Urbieta, L. Antonelli, J. Guerra, G. Rossi, Tracing user stories and source code using the language extended lexicon, in: A. Rocha, H. Adeli, G. Dzemyda, F. Moreira (Eds.), Information Systems and Technologies, Springer International Publishing, Cham, 2022, pp. 413–429.

[10] L. M. Cysneiros, J. C. S. do Prado Leite, Using the language extended lexicon to support non-functional requirements elicitation, in: Proceedings of the Workshop em Engenharia de Requisitos, Buenos Aires, Argentina, 2001, pp. 139–153.

[11] L. Antonelli, G. Rossi, J. C. S. do Prado Leite, A. Oliveros, Deriving requirements specifications from the application domain language captured by language extended lexicon, in: Proceedings of the Workshop em Engenharia de Requisitos, Buenos Aires, Argentina, 2012, pp. 1–14.

[12] A. Ferrari, Natural language requirements processing: From research to practice, in: Proceedings of the IEEE/ACM 40th International Conference on Software Engineering, 2018, pp. 536–537.

[13] L. Antonelli, G. Rossi, A. Oliveros, A collaborative approach to describe the domain language through the language extended lexicon, J. Object Technol. 15 (3) (2016) 3:1–27.

[14] M. Golfarelli, D. Maio, S. Rizzi, The dimensional fact model: a conceptual model for data warehouses, International Journal of Cooperative Information Systems 07 (2-3) (1998) 215–247.

[15] O. Romero, A. Abelló, Data-driven multidimensional design for OLAP, in: J. B. Cushing, J. C. French, S. Bowers (Eds.), Proceedings of the Scientific and Statistical Database Management International Conference, Portland, OR, USA, 2011, pp. 594–595.

[16] P. Jovanovic, O. Romero, A. Simitsis, A. Abelló, D. Mayorova, A requirement-driven approach to the design and evolution of data warehouses, Inf. Syst. 44 (2014) 94–119.

[17] F. D. Tria, E. Lefons, F. Tangorra, Hybrid methodology for data warehouse conceptual design by UML schemas, Inf. Softw. Technol. 54 (4) (2012) 360–379.

[18] O. Romero, A. Abelló, Automatic validation of requirements to support multidimensional design, Data Knowl. Eng. 69 (9) (2010) 917–942.

[19] T. Niemi, J. Nummenmaa, P. Thanisch, Constructing OLAP cubes based on queries, in: Proceedings of the 4th ACM International Workshop on Data Warehousing and OLAP, Atlanta, Georgia, USA, 2001, pp. 9–15.

[20] S. Bimonte, L. Antonelli, S. Rizzi, Requirements-driven data warehouse design based on enhanced pivot tables, Requirements Engineering 26 (1) (2021) 43–65.

[21] R. Nair, C. Wilson, B. Srinivasan, A conceptual query-driven design framework for data warehouse, International Journal of Computer and Information Engineering 1 (1) (2007) 62–67.

[22] O. Romero, A. Abelló, A survey of multidimensional modeling methodologies, Int. J. Data Warehous. Min. 5 (2) (2009) 1–23.

[23] F. D. Tria, E. Lefons, F. Tangorra, Cost-benefit analysis of data warehouse design methodologies, Inf. Syst. 63 (2017) 47–62.

[24] C. Kaldeich, J. O. e. Sá, Data warehouse methodology: A process driven approach, in: Proceedings of the International Conference on Advanced Information Systems Engineering, 2004, pp. 536–549.

[25] P. Giorgini, S. Rizzi, M. Garzetti, GRAnD: A goal-oriented approach to requirement analysis in data warehouses, Decision Support Systems 45 (1) (2008) 4–21.

[26] E. Elamin, S. Alshomrani, J. Feki, SSReq: A method for designing star schemas from decisional requirements, in: Proceedings of the 2017 International Conference on Communication, Control, Computing and Electronics Engineering, 2017, pp. 1–7.

[27] F. Bargui, H. Ben-Abdallah, J. Feki, Multidimensional concept extraction and validation from OLAP requirements in NL, in: Proceedings of the 2009 International Conference on Natural Language Processing and Knowledge Engineering, 2009, pp. 1–8.

[28] M. Thenmozhi, P. Ezhilarasi, OntoMD: Ontology based multidimensional schema design approach, International Journal of Computer Science and Information Security 14 (2016) 8—16.

[29] M. Zekri, S. B. Yahia, I. Hilali-Jaghdam, A software prototype for multidimensional design of data warehouses using ontologies, in: Proceedings of Computational Collective Intelligence International Conference, Hendaye, France, 2019, pp. 273–284.

[30] R. Navigli, P. Velardi, From glossaries to ontologies: Extracting semantic structure from textual definitions, Frontiers in Artificial Intelligence and Applications 167 (2008) 71–87.

[31] L. Bozzato, M. Ferrari, A. Trombetta, Building a domain ontology from glossaries: A general methodology, in: Proceedings of the 5th Workshop

on Semantic Web Applications and Perspectives, Rome, Italy, 2008, pp. 1–10.

[32] G. Arnicans, D. Romans, U. Straujums, Semi-automatic generation of a software testing lightweight ontology from a glossary based on the ONTO6 methodology, in: A. Caplinskas, G. Dzemyda, A. Lupeikiene, O. Vasilecas (Eds.), Databases and Information Systems VII - Selected Papers from the Tenth International Baltic Conference, IOS Press, 2012, pp. 263–276.

[33] L. Antonelli, G. Rossi, J. C. S. do Prado Leite, A. Oliveros, Language extended lexicon points: Estimating the size of an application using its language, in: Proceedings of the 22nd International Requirements Engineering Conference, 2014, pp. 263–272.

[34] M. E. Centeno, M. Bertolami, A. Oliveros, Software size estimation in the early stages of development, in: Proceedings of the VI Workshop Ingeniería de Software, 2009, pp. 941–950.

[35] A. d. P. A. Oliveira, J. C. S. d. P. Leite, L. M. Cysneiros, C. Cappelli, Eliciting multi-agent systems intentionality: from language extended lexicon to i* models, in: Proceedings of the XXVI International Conference of the Chilean Society of Computer Science, 2007, pp. 40–49.

[36] K. Breitman, J. do Prado Leite, Ontology as a requirements engineering product, in: Proceedings of the 11th IEEE International Requirements Engineering Conference, 2003, pp. 309–319.

[37] A. Garrido, L. Antonelli, J. Martin, M. M. E. Alemany, J. Mula, Using LEL and scenarios to derive mathematical programming models. application in a fresh tomato packing problem, Comput. Electron. Agric. 170 (105242) (2020).

[38] S. Diniz Junqueira Barbosa, M. Selbach Silveira, Supporting a shared understanding of communication-oriented concerns in human-computer interaction: A lexicon-based approach, in: Proceedings of the International Conference on Engineering Human Computer Interaction and Interactive Systems, Hamburg, Germany, 2004, pp. 271–288.

[39] L. M. Cysneiros, J. C. S. do Prado Leite, Using UML to reflect non-functional requirements, in: Proceedings of the Conference of the Centre for Advanced Studies on Collaborative Research, Toronto, Canada, 2001, p. 2.

[40] J. P. Mighetti, G. D. S. Hadad, A requirements engineering process adapted to global software development, CLEI Electronic Journal 19 (2016) 181–209.

[41] L. Antonelli, G. Rossi, J. C. Leite, J. a. Araújo, Early identification of crosscutting concerns with the language extended lexicon, Requir. Eng. 20 (2) (2015) 139–161.

[42] E. Almentero, J. C. S. do Prado, C. Lucena, Towards software modularization from requirements, in: Proceedings of the 29th Annual ACM Symposium on Applied Computing, Gyeongju, Republic of Korea, 2014, pp. 1007–1012.

[43] M. Mauco, M. Leonard, D. Riesco, G. Montejano, N. Debnath, Formalising a derivation strategy for formal specifications from natural language requirements models, in: Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, 2005, pp. 646–651.

[44] J. L. Razafindramintsa, M. Thomas, J. P. Razafimandimby, Elaborate lexicon extended language with a lot of conceptual information, International Journal of Computer Science, Engineering and Applications 5 (6) (2015) 1—18.

[45] J. L. Razafindramintsa, J. Razafimandimby, Paul, M. Thomas, A. Becheru, Semantic aspect derivation of the Praxème methodology from the elaborate lexicon extended language, in: Proceedings of the 20th International Conference on System Theory, Control and Computing, 2016, pp. 842–847.

[46] R. M. Andrianjaka, R. J. Luc, T. Mahatody, S. Ilie, R. N. Raft, Restructuring extended lexical elaborate language, in: Proceedings of the 23rd International Conference on System Theory, Control and Computing, 2019, pp. 266–272.

[47] B. E. Tarehy, T. Mahatody, J. L. Razafindramintsa, J. P. Razafimandimby, Reuse environment based on elaborate lexicon extend language, in: Proceedings of the 18th International Carpathian Control Conference, 2017, pp. 310–315.

[48] G. Hadad, J. Doorn, Creating software system context glossaries, in: Encyclopedia of Information Science and Technology, Mehdi Khosrow-Pour DBA, 2009, pp. 789–794.

[49] R. Evans, The extended lexicon: language processing as lexical description, in: Proceedings of the International Conference Recent Advances in Natural Language Processing, 2013, pp. 270–276.

[50] A. Nabli, J. Feki, F. Gargouri, Automatic construction of multidimensional schema from OLAP requirements, in: Proceedings of the 3rd ACS/IEEE International Conference on Computer Systems and

Applications, 2005, pp. 1–7.

[51] O. Romero, A. Abelló, Automating multidimensional design from ontologies, in: Proceedings of the ACM Tenth International Workshop on Data Warehousing and OLAP, Lisbon, Portugal, 2007, pp. 1–8.

[52] C. Phipps, K. C. Davis, Automating data warehouse conceptual schema design and evaluation, in: Proceedings of the 4th Intl. Workshop on Design and Management of Data Warehouses, Toronto, Canada, 2002, pp. 23–32.

[53] A. Carmè, J. Mazón, S. Rizzi, A model-driven heuristic approach for detecting multidimensional facts in relational data sources, in: Proceedings of 12th International Conference on Data Warehousing and Knowledge Discovery, Bilbao, Spain, 2010, pp. 13–24.

[54] A. Sakka, S. Bimonte, S. Rizzi, L. Sautot, F. Pinet, M. Bertolotto, A. Besnard, N. Rouillier, A profile-aware methodological framework for collaborative multidimensional modeling, Data Knowl. Eng. 131-132 (2021) 101875.

[55] M. Golfarelli, S. Rizzi, Data warehouse testing: A prototype-based methodology, Inf. Softw. Technol. 53 (11) (2011) 1183–1198.

[56] M. Serrano, J. Trujillo, C. Calero, M. Piattini, Metrics for data warehouse conceptual models understandability, Information & Software Technology 49 (8) (2007) 851–870.

[57] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, A. Wessln, Experimentation in Software Engineering, Springer Publishing Company, Incorporated, 2012.

[58] D. Sjoberg, B. Anda, E. Arisholm, T. Dyba, M. Jorgensen, A. Karahasanovic, E. Koren, M. Vokac, Conducting realistic experiments in software engineering, in: Proceedings International Symposium on Empirical Software Engineering, 2002, pp. 17–26.

[59] L. Zhao, W. Alhoshan, A. Ferrari, K. J. Letsholo, M. A. Ajagbe, E.-V. Chioasca, R. T. Batista-Navarro, Natural language processing for requirements engineering: A systematic mapping study, ACM Computing Surveys 54 (3) (2021) 1–41.

[60] The Stanford natural language processing group, https://nlp.stanford.edu/software/, Accessed Aug. 18, 2017 [Online] (2022).
URL https://nlp.stanford.edu/software/

[61] H. Kilicoglu, S. Bergler, Recognizing speculative language in biomedical research articles: a linguistically motivated perspective, in: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing, 2008, pp. 46–53.

[62] H. Sack, C. G. Schuetz, L. Bozzato, B. Neumayr, M. Schrefl, L. Serafini, Knowledge graph olap, Semantic Web 12 (4) (2021) 649–683.

[63] L. Qingjie, X. Lingyu, Y. Jie, W. Lei, X. Yunlan, S. Suixiang, L. Yang, Research on domain knowledge graph based on the large scale online knowledge fragment, in: 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), 2014, pp. 312–315.