Textual Approach for Designing Database Conceptual Models: A Focus Group

Jonnathan Lopes[a], Maicon Bernardino[b], Fábio Basso[c] and Elder Rodrigues[d]
Federal University of Pampa, PPGES, LESSE, Av. Tiarajú, 810, Ibirapuitã, CEP 97546-550, Alegrete, RS, Brazil

Keywords: Domain Specific Language, Conceptual Data Model, Conceptual Model, Database Modeling.

Abstract: Different approaches to software development are based on at least three database models: conceptual, logical, and physical, requiring distinct abstraction levels to represent complementary concepts. The selection of a conceptual database modeling approach, among other things, depends on the problem domain, knowledge, and preferences of the developer. This paper presents a new domain-specific textual language devoted to the conceptual relational database modeling, *i.e.*, entity-relationship modeling. Furthermore, we present a preliminary evaluation conducted to analyze our proposed DSL, comparing two grammar versions in a focus group research method.

1 INTRODUCTION

Software Engineering (SE) is not feasible without persistence and data manipulation. According to Date (Date and Warden, 1990), a database is a collection of stored operational data, used by the application systems of some organization. For Elmasri and Navathe (Elmasri and Navathe, 2015), a database can be defined as an abstraction from the real world, also called the mini-world, since it represents aspects that, together, carry an implicit meaning. Databases can be considered as the most important organizational assets today. This is due to the fact that they do not only store trivial information but, for instance, also billing data, market research, and other aspects assisting in decision making.

In this scenario, it is notable that there is an increasing effort of academia to provide a good level of preparation for future professionals who will enter an increasingly demanding industry. Higher education institutions, *e.g.* universities, often approach the database curriculum with specific courses converging in their programs. The database teaching area is an essential part of computer professional training. The focus on database teaching is generally divided into four stages: (i) design and modeling, (ii) database management systems, (iii) comparative studies among these systems, and (iv) the application development (Al-Dmour, 2010; Connolly and Begg, 2006).

Assuming that there is a growing search for instruments supporting the teaching-learning process in academia, we present a study focusing on the first stage: design and modeling. In general, teaching database design and modeling includes the presentation of essential topics and the subsequent introduction of modeling tools using generally graphic approaches. While graphical modeling is the most popular approach, textual modeling can be an alternative to the teaching-learning process at a higher level (Dimitrieski et al., 2015). This is due to the lack of textual approaches for the first stage. Hence, this limitation can represent a gap in the teaching-learning process of database modeling.

Considering that logical and physical modeling occur mainly in text mode, an approach like the one adopted in the ERtext tool can be considered viable because it manages to maintain the same style workflow in all modeling stages. Nowadays, the declarative programming using Domain-Specific Languages (DSLs) (Kelly and Tolvanen, 2008) is the dominant paradigm of an extensive set of domains, *e.g.* databases or templating. With this approach, instead of hundreds of lines of verbose code it is possible to make clearly defined data schemes, providing an abstract representation, semantically simpler and meaningful. Built on a grammar perceived in the proposed study as "ease to use" and as "useful", the main aim

[a] https://orcid.org/0000-0001-9402-801X
[b] https://orcid.org/0000-0003-2776-8020
[c] https://orcid.org/0000-0003-4275-0638
[d] https://orcid.org/0000-0003-3431-2814

Lopes, J., Bernardino, M., Basso, F. and Rodrigues, E.
Textual Approach for Designing Database Conceptual Models: A Focus Group.
DOI: 10.5220/0010232801710178
In *Proceedings of the 9th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2021)*, pages 171-178
ISBN: 978-989-758-487-9

of this paper is to report the ERtext, a modeling tool based on a DSL for the database designing and modeling with the scope in conceptual level modeling.

This paper is organized as it follows. Section 2 presents the related work. Section 3 provides the design decisions and the architecture adopted. Section 4 describes an overview of the tool operation. Finally, Section 7 concludes this paper.

2 RELATED WORK

This section describes the most representative works for the object of this study: conceptual database modeling. After an extensive study composed of a Multivocal Literature Mapping (MLM) (DOI: doi.org/10.5281/zenodo.4064987 - in portuguese), we selected proposals and tools closely matching with our features from our DSL.

Dimitrieski *et al.* (Celikovic et al., 2014; Dimitrieski et al., 2015) presents a tool called System Modeling Tool (MIST). This tool uses a DSL called EERDSL, a language based on the improved Extended Entity-Relationship (EER) model. The EER includes all the concepts as extension from those introduced by the original ER model proposed by Chen (Chen, 1976). In addition, it includes the concepts of subclasses and superclasses (*Is-a*), along with the concepts of specialization and generalization. MIST presents a bidirectional (graphical and textual) modeling approach for the data model approach. From the results obtained, the idea of MIST was conceived. The purpose of the tool is to apply it both in the professional market and for teaching designing and modeling databases in the academic environment. MIST was developed with the aid of the Xtext framework for the notation of textual DSL and initially used the Eugene framework, a project that was discontinued, for its graphic version. As a result, Eugene was replaced by the Sirius framework. Besides, MIST also supports the generation of SQL code.

Dbdiagram.io (https://dbdiagram.io/) is a web-based open-source tool for drawing an Entity Relationship Diagram (ERD), with a textual approach implementing its own DSL. This DSL uses a model similar to the logic view, with a differential of fast learning curve and, in addition, an intuitive graphical representation. The presentation of the diagram elements can be freely organized by the user in real time. However, it is worth to note that all modeling is in fact performed in a textual manner. Moreover, the tool also offers automatic generation of SQL code.

Likewise, QuickDBD is a web-based tool adopting the same operational mode as dbdiagram.io, also implementing its own textual DSL for database modeling. However, it is a proprietary tool, *i.e.* paid for and with an explicit focus on the industry. Both tools are also very similar to the generation of graphical model representations and highlight several arguments for their adoption, such as the quick understanding of their DSL, the perspective of carrying out fluid works, the access of any platform and design sharing.

Finally, the web-based open-source tool RelaX (Relational Algebra Calculator) (Kessler et al., 2019) is developed in the academic environment aiming at teaching relational algebra through operations on relational databases. It has a textual approach, using a DSL called RelAlg, and including two operational perspectives: RelAlg and SQL commands. RelaX uses a model approach already at a physical level for operations, such as DDLs for data structures and DMLs for data manipulation *e.g.* queries. Despite its functionality, RelaX is not built as a database designing and modeling tool, but it intents for use restricted for teaching in the academic environment.

Several different tools have been put forward, the main difference from ERText is that the related work focuses in the logical or in physical model design. MIST on the other hand aims at the conceptual model, and since it was not possible to find the tool, or at least its code, for an analysis of usage, unlike Dbdiagram.io, QuickDBD and RelaX, we cannot properly compare MIST with ERText. This highlights another important difference, since our proposal is completely open-source, available for use, evaluation and learning. Among the mentioned tools, only RelaX has its code also available. This may indicate that the general applicability of our tool in the teaching-learning process of database conceptual modeling in the academic is possibly greater than these tools.

3 ANALYSIS AND DESIGN

This section presents the analysis and designing phases, describing the requirements raised for the DSL definition and its associated design decisions as well as the grammar and the architecture are shown.

3.1 Software Requirements

Our focus is on the teaching process, so it is essential to tag ERText as an open-source license, allowing the evolution and collaborative maintenance with the involvement of other developers.

In the following, we listed the Software Requirements (SR) that were defined based on the surveyed literature: **SR1.** DSL must be made available under an open-source license. **SR2.** The DSL should allow for the textual representation of conceptual data models. **SR3.** Conceptual data models should support the definition of fundamental domain concepts such as entities, attributes, relationships, and their cardinalities. **SR4.** Conceptual data models should support the definition of attributes, identifiers, generalization and specialization, self-relationships, and ternary relationships. **SR5.** DSL implementation must transform from the conceptual to the logical model displaying the result generated to the user.

In the following we described the Design Decisions (DD) supporting all the previously discussed SRs: **DD1.** The solution must adopt an open-source LW (Language Workbench) to aid the implementation of the textual DSL (SR1, SR2). Through the investigation conducted during this study, we select the LW Xtext for the development of our DSL. **DD2.** A DSL must provide a textual representation equivalent to the currently graphical ER model (SR3, SR4). For the SRs covered by this DD, we adopted a strategy to carry out an analysis on the tools identified in an MLM, as well as in the Heuser's reference book (Heuser, 2009). **DD3.** The solution must perform the transformation among the models (*e.g.* conceptual, logical, and physical) (RQ5).

3.2 The Language

In the current phase of the work, the language at the conceptual level is not fully finalized. There are topics related to the scope validation, as in the case of the treatment of unwanted cross-references and other restrictions inherent to the ER model that must be analyzed and then implemented. The grammar definition of the created DSL is displayed in Figure 1.

The `grammar` command specifies the DSL name, while the `with` statement declares an inheritance from another language. In the case of the proposed grammar, a standard Xtext grammar, called `Terminals`, will be used, which provides some predefined rules The `generate` command is the statement that produces the language's AST.

The first rule, called the input rule, defines what the general language structure looks like. Words and symbols in double or single quotes indicate reserved words. For example, the object `Entities` must be preceded by "`Entities {`". This object represents a kind of container as indicated through the assignment operator `+=`. It is an object that can contains other objects, in this case, one or more `Entity`. It is established that each DSL file must also be composed of a `Domain` and zero or more `Relation` objects.

```
grammar org.xtext.university.ertext.ERtext
with org.eclipse.xtext.common.Terminals
generate ERtext "http://www.xtext.org/unipampa/ertext/
      ERtext"
ERModel:
    domain=Domain ';'
    ('Entities' '{') entities+=Entity+ ('}' ';')
    ('Relationships' '{') relations+=Relation*
       ('}' ';');
Domain:
    'Domain' name=ID;
Attribute:
    name=ID type=DataType (isKey?='isIdentifier')?;
Entity:
    name=ID ('is' is+=[Entity])*
    ('{' attributes+=Attribute
    (',' attributes+=Attribute)*'}')?;
Relation:
    (name=ID)? ('[' leftEnding=RelationSide
    'relates'
    rightEnding=RelationSide ']')
    ('{' attributes+=Attribute
    (',' attributes+=Attribute)* '}')*;
RelationSide:
    cardinality=('(0:1)'|'(1:1)'|'(0:N)'|'(1:N)')
    target=[Entity] | target=[Relation];
enum DataType:
    INT='int'            | DOUBLE='double'      |
    MONEY='money'        | STRING='string'      |
    BOOLEAN='boolean'    | DATETIME='datetime'  |
    BLOB='file';
```

Figure 1: Grammar definition of the DSL.

For a better understanding, it should be made clear that multiplicity is indicated by `*` (zero or many), `+` (one or many), or `?` (zero or one). By not placing any of these operators, then one implicitly is expected to occur. Regarding assignments, when only one = is specified it means that the object on the left expects only one record. Therefore, for += then zero, one or more occurrences are expected.

The `Domain` object is preceded by a reserved word with the same name, followed by an identifier. The entity is defined by the `Entity` statement and an identifier name for this object. Defining an inheritance is optional using the reserved word `is`. After defining the name, a body of keys opens in which the entity attributes are specified. An entity must contain at least one attribute and, due to the possible existence of weak entity set, it does not need to be an identifier.

The compound rules are only performed due to the possibility of grouping expressions using parentheses, in addition to the possibility of using other rules through cross-references. The brackets between the `Entity` rule are used to indicate that you want to use only the `name` attribute that identifies the object. Entity attributes are defined by a name, inheriting the `ID` rule from `Terminals`, and optional attributes *e.g.* `isIdentifier` to symbolize primary keys.

The relationship is defined, already within the body of the `Relationships` block, with an optional declaration of its identification. Then, brackets are opened and two elements must be specified `RelationSide` as a reference to the attributes `leftEnding` and `rightEnding`. These attributes represent the sides of a relationship. These objects must be separated by the expression `relates`. The sides of the relation are defined in the rule *RelationSide*, composed of two attributes. The `cardinality` attribute is mandatory and uses numerals `(0:1), (1:1), (0:N), (1:N)`. The `target` attribute is likewise mandatory, accepting a cross-reference to a `Entity` or `Relation`. In the case of a `target` referencing a `Relation`, then a ternary relationship is highlighted. The attribute types are contained in an enumerated list called `DataType`, on the left is the representation in the model *ECore*. On the right is the reserved word that is used in the language. The conditional symbol `|` means the logical operator *OR* and serves to separate each definition `<key> = <value>` as an option within the list.

3.3 Tool Architecture

The solution architecture is shown in Figure 2 and was built with the help of Xtext, which helped to generate most of the editor's infrastructure, the parser and ECore model i.e., a representation in memory at runtime of the model created using DSL.

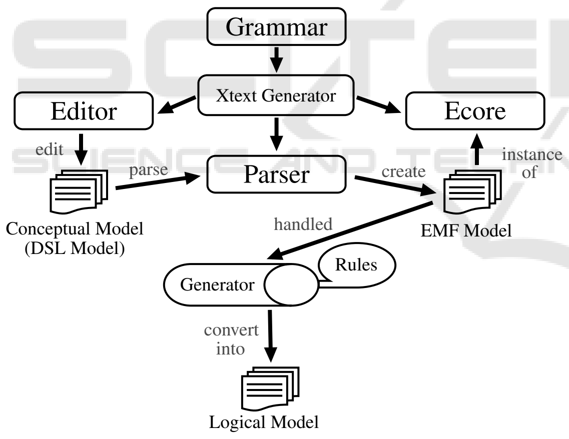


Figure 2: ERtext architecture.

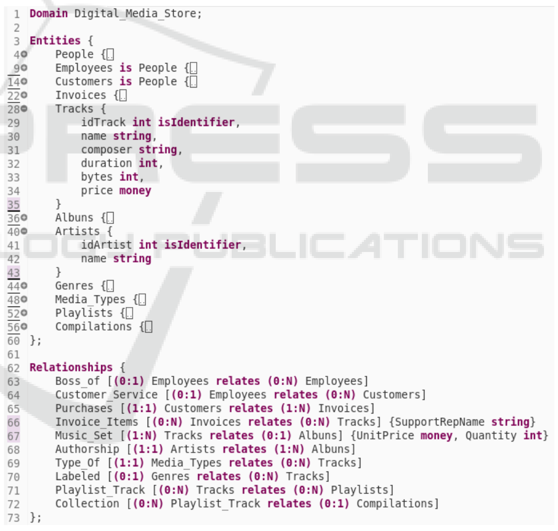
4 ERText

Our plugin can either be integrated with the Eclipse Rich Client Platform (RCP), or it can be used as a standalone application. The difference is that when used as an Eclipse plugin the editor can provide, in addition to the grammar features, support for other languages, *e.g.* Java, PHP. On the other hand, a standalone product provides the entire infrastructure focused solely on the developed language and can be distributed as an open-source tool as long as the EPL-2.0 software license guidelines are followed.

4.1 Operation

Figure 3 shows a screenshot of the ERText. This example is inspired on Chinook Sample Database (Repository: github.com/lerocha/chinook-database), and contains eleven (11) entities and ten (10) relationships, including generalization and specialization, a self-relationship and a ternary relationship. ERText is embedded with design features assisted by real-time validation including: a) syntax highlighting indicating syntax errors in writing time; b) code completion, and c) hovering, a feature that displays information about an item when the mouse cursor is placed over it.

In addition, although the definition of data types is not foreseen in the classic conceptual data model, for reasons related to the future intention to perform the generation of SQL commands, we decided to maintain this design decision in the grammar.

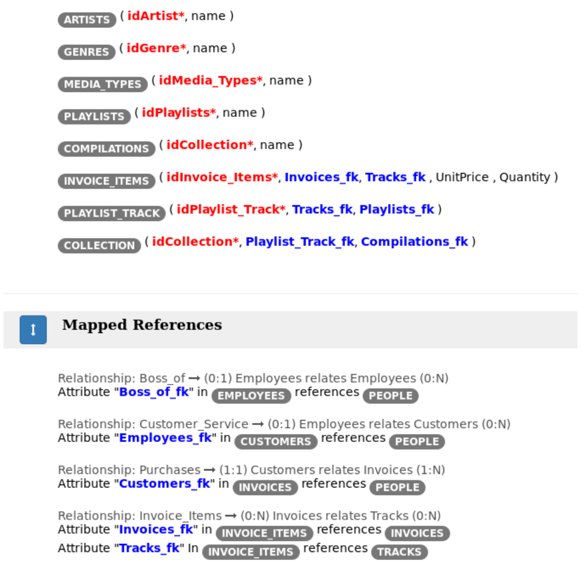


```
Domain Digital_Media_Store;

Entities {
    People {…
    Employees is People {…
    Customers is People {…
    Invoices {…
    Tracks {
        idTrack int isIdentifier,
        name string,
        composer string,
        duration int,
        bytes int,
        price money
    }
    Albuns {…
    Artists {
        idArtist int isIdentifier,
        name string
    }
    Genres {…
    Media_Types {…
    Playlists {…
    Compilations {…
};

Relationships {
    Boss_of [(0:1) Employees relates (0:N) Employees]
    Customer_Service [(0:1) Employees relates (0:N) Customers]
    Purchases [(1:1) Customers relates (1:N) Invoices]
    Invoice_Items [(0:N) Invoices relates (0:N) Tracks] {SupportRepName string}
    Music_Set [(1:N) Tracks relates (0:1) Albuns] {UnitPrice money, Quantity int}
    Authorship [(1:1) Artists relates (1:N) Albuns]
    Type_Of [(1:1) Media_Types relates (0:N) Tracks]
    Labeled [(0:1) Genres relates (0:N) Tracks]
    Playlist_Track [(0:N) Tracks relates (0:N) Playlists]
    Collection [(0:N) Playlist_Track relates (0:1) Compilations]
};
```

Figure 3: Snippet of the solution being used.

Files with this format were generated for a better view from the text-markings using a model transformation of type model-to-text. These markings can be rendered by any browser, or even within the environment, increasing the power of understanding by the user. The logical model mapping the conceptual data model, i.e., derived by the generator, is shown in Figure 4.

The model-to-text transformation was mapped to the save event. Considering the exemplified model as input, this transformation results in a new model composed of fourteen (14) entities already having their inferred referential integrity, *i.e.* the records that point



```
ARTISTS ( idArtist*, name )
GENRES ( idGenre*, name )
MEDIA_TYPES ( idMedia_Types*, name )
PLAYLISTS ( idPlaylists*, name )
COMPILATIONS ( idCollection*, name )
INVOICE_ITEMS ( idInvoice_Items*, Invoices_fk, Tracks_fk , UnitPrice , Quantity )
PLAYLIST_TRACK ( idPlaylist_Track*, Tracks_fk, Playlists_fk )
COLLECTION ( idCollection*, Playlist_Track_fk, Compilations_fk )

Mapped References

Relationship: Boss_of → (0:1) Employees relates Employees (0:N)
Attribute "Boss_of_fk" in EMPLOYEES references PEOPLE

Relationship: Customer_Service → (0:1) Employees relates Customers (0:N)
Attribute "Employees_fk" in CUSTOMERS references PEOPLE

Relationship: Purchases → (1:1) Customers relates Invoices (1:N)
Attribute "Customers_fk" in INVOICES references PEOPLE

Relationship: Invoice_Items → (0:N) Invoices relates Tracks (0:N)
Attribute "Invoices_fk" in INVOICE_ITEMS references INVOICES
Attribute "Tracks_fk" In INVOICE_ITEMS references TRACKS
```

Figure 4: Snippet of the created logical model.

to other records are already established.

In order to map and transform from a conceptual data model to a logical data representation of database, we adopted some assumptions proposed by Heuser (Heuser, 2009). The assumptions implemented so far can be summarized in: (i) column addition for 1:1 relationships; (ii) column addition for 1:N relationships, and; (iii) creation of own table for N:N relationships.

According to Heuser (Heuser, 2009), concerning generalization and specialization concepts, there are two alternatives that can be derived from a mapping for the transformation. The first one recommends the use of a single table for the entire hierarchy of entities, *i.e.* it advises merging the tables. The second one suggests the use of a table by modeled entity, as long as respecting referential integrity, *i.e.* the primary keys of the child entities must necessarily point to the primary key of the parent entity, creating a foreign keys references. We chose the second alternative as assumption for this design decision.

The generator of the logical model was developed with GPL Xtend, and currently has about four hundred lines of code to carry out the transformation from the conceptual data model to the logical data model.

5 PRELIMINARY EVALUATION

This section describes the preliminary evaluation conducted to analyze the proposed grammar, aiming to enhance it in a final version of the solution.

For that, it was established the use of a focus group, a qualitative research method that aims to generate feedback from a group of people in relation to a specific topic. This approach is widely used as an activity for market research in several areas, since it can play an important role in supporting exploratory research. Thus, starting from a previous investigation (Edmunds, 2000; Frisina, 2006; Tong et al., 2007) the process carried out in this step, expressed in Figure 5, was based on the guidelines established in the work of (Kontio et al., 2008), which cover the application of this method in the context of Software Engineering.

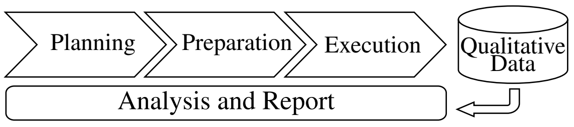


Figure 5: Focus Group Process (Kontio et al., 2008).

5.1 Planning

During the planning stage, a protocol was defined that should be followed. In this protocol, motivated by the problem of generating a definitive version of the proposed DSL grammar, the necessary documents were created for its execution: **(i)** Informed Consent Form (ICF); **(ii)** Glossary of Terms; **(iii)** Profile Questionnaire; **(iv)** Discussion Instruments 1, 2 and 3; **(v)** Models of the Grammars, and; **(vi)** Presentation Script. All models of the documents produced are available in a public repository (Repository: github.com/JonnathanRiquelmo/Focus-Group-Protocol - in portuguese).

5.2 Preparation

Typically evaluations using focus groups should consist of four (4) to six (6) individual groups to the scientific rigor to be considered high. The size of each group can vary from three (3) to twelve (12) elements, with the most common being a number between four (4) and eight (8) participants (Kontio et al., 2008).

For the present study, it was possible to run one (1) focus group for reasons of time and human resources. After the invitation, a total of thirteen (13) participants collaborated, all from the Software Engineering area. Of this total, three (3) participants were undergraduate students, nine (9) were master's students and one (1) doctoral student.

The Profile Questionnaire was then applied, in which it was possible to identify that there was a balanced level of knowledge among the participants. This was verified because everyone already had contact with DSL, having used this type of language, at least during graduation. It was also informed that the entire process would be recorded on audio, a fact that everyone agreed with it.

5.3 Execution

The focus group was held in the second half of 2019, on the premises of a higher education institution, and lasted for two (2) hours. Starting with the presentation script that should be followed, where there was the presentation of the focus group objective and the basic concepts involved, a request was made for the participants to read and sign the ICF. With the signed ICF, we continued the planned script.

For each discussion instrument available we waited until all the participants answered individually. Afterwards, there was a group discussion on the topic raised. The entire process was recorded in audio and had the support of the researchers involved in this study. A text transcription was made for the observations raised by the debate that followed, thus characterizing the brainstorming practices.

5.4 Results Analysis

According (Kontio et al., 2008), the analysis and interpretation stage of the generated data constitutes an important part of the qualitative research, considering the context, the behavior, and the perception of the subjects. For the data analysis stage the audio was analyzed in parallel to the taken notes. With these materials and the participant's responses to each discussion instrument, it was possible to evaluate the results of the executed focus group.

After the presentation of the script prepared for the focus group, the discussions on the three (3) instruments created for the dynamics began. The first instrument contained the following statement, associated with a Likert scale, composed of levels of agreement, and arranged from one (1) to five (5), where one (1) indicates totally disagree and five (5) totally agree: ***"Domain-specific languages with a textual approach can be applied in conceptual modeling as they can describe certain properties quickly and concisely. Therefore, these solutions can be used or even adapted in an effective way with respect to the representation of the domain they model."***

After all participants answered the instrument, a moment of discussion was opened among all. As they did not see the proposed DSL model, some doubts arose and the researchers involved sought to resolve all of them in order not to influence the following discussions. The debate continued with the participants raising possible advantages of a textual model. Some said they believed that this approach could be easier to understand, but that it would depend on the user. This assumption included two likely profile types: analysts and developers. The group came to the conclusion that the approach could be seen as more productive by users with a developer profile, but less profitable by those who had a more analytical bias due to their abstraction level in relation to graphic approaches. Figure 6 shows the responses distribution for the first discussion.

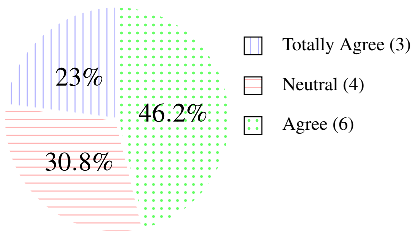


Figure 6: Results of Discussion 1 - Focus Group.

Next, the discussion of the second instrument was carried out. The activity consisted of the following question: ***"Considering that a conceptual data model must define at least the domain entities, their attributes and the number of possible occurrences (cardinality) of associations (relationship), as you would define a basic grammar (DSL) for your representation?"***

It is important to mention that the participants could talk freely during the entire realization of this instrument. After each one suggested its syntax, there was a debate and an information exchange on how to better structure the commands. The discussion was mainly focused on how to represent the relations of the ER model in a textual syntax.

The greatest difficulty observed is about how to define an order. Another point worth mentioning was the cardinality, where in general the nomenclature used in Chen's original diagram (*e.g.* 0,1) was followed. However, at the end of the instrument, there were very divergent opinions in relation to the ideal representation. Apparently, each had a different point of view, some choosing to include relationships within the entities, and others outside. There were also suggestions regarding possible keywords, such as `Element`, `ElementFather`, `ElementSource`, `Type`, and `Referential`. Still, six (6) participants suggested the use of semicolons (;) to separate the declarations of elements, and all thirteen (13) recommended the use of symbols such as parentheses, brackets and/or keys to group similar sets of elements.

After the end of proposed dynamic, the last experimental instrument is executed: An artifact composed by an example of each grammar version produced preliminary for this work. Based on both grammar versions, we asked the participants to choose between the options, thus indicating which one they evaluated as the most feasible for ER modeling. In addition, we also requested to indicate the pros and cons observed in each model. The second model was eventually chosen by the majority, as can be seen in Figure 7. However, at the end of the conditions we reached a consensus: the way of defining entities of the first model and the disposition of relationships for the second, especially the use of conventions in cardinalities, are the most suitable for application in teaching-learning process, thus indicating the need for a merger of both versions.

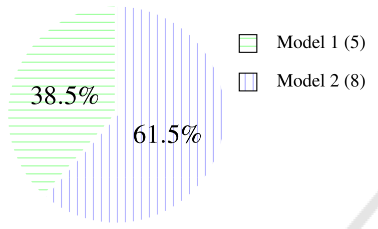


Figure 7: Results of Discussion 3 - Focus Group.

6 THREATS TO VALIDITY

For the list of possible threats, the topics and recommendations raised in the work of (Wohlin et al., 2012) were used. These threats followed a pattern and were divided into four (4) categories as follows.

Construct Validity: concerns the experimental design and social factors. (i) *Inadequate Preoperational Explication*: This threat is when the focus group may not have the objective of the artifacts sufficiently defined before they are translated into measures or treatments. (ii) *Interaction of different treatments*: If the subjects are involved in more than one study, the treatments of the different studies may interact and reverberate in the final results. All subjects did just this focus group at the time of realization.

Internal Validity: are influences that can affect independent variables in relation to causality, without the researcher's knowledge. (i) *History*: There is a risk that a specific time period will influence the experiment. In order to mitigate this threat the entire process was carried out upon the participants' notice of a period when everyone was not necessarily overwhelmed with academic activities *e.g.* tests and assignments. (ii) *Tests*: If the tests are repeated, the subjects may respond differently at different times, as they know-how the test is performed. There was no need to repeat the activities, since they were performed once by each participant. (iii) *Instrumentation*: This threat is related to artifacts used for the execution of the focus group, such as data collection forms, etc. If these are poorly designed, the experience is negatively affected. To combat this threat all artifacts were previously checked and validated at meetings among the researchers involved in this work.

External Validity: are conditions related to replication of the evaluation. (i) *Subjects*: The subjects selected for the focus group may not represent a significant group for the study area. In an attempt to mitigate this threat it was carried out with participants from the area of Software Engineering and Computer Science, inserted in the context of the use of conceptual modeling of databases. However, the fact that the sample is of only one focus group is a threat indicated in the literature. It was not possible to mitigate this fact. (ii) *Interaction of subjects with assessment artifacts*: This is the threat related to the application of the assessment artifacts with the subjects. Depending on the moment this can affect the results. If, *e.g.* a questionnaire is carried out a few days after the execution of the focus group, people tend to answer differently than they would do moments after the activities. All instruments were performed in the same session.

Conclusion Validity: are related to issues affecting the ability to infer a correct conclusion about the relationship between treatments and results of an evaluation. (i) *Reliability of Results*: This threats obtained has a direct impact on the validity of the focus group as a whole. As it is an assessment with a greater qualitative focus, this threat could not be mitigated due to the subjectivity inherent in the subjects' responses in such assessments. (ii) *Evaluation Environment*: The assessment should be carried out in a controlled environment, trying to avoid external influences. To mitigate this possible threat, participants were instructed that conversations outside the scope of the focus group could not take place during the entire duration of the activities, as well as leaving the environment or accessing electronic devices.

7 CONCLUSIONS

In this study, the requirements, the design decisions, the architecture, and the ERText tool were exposed. XText proved to be an LW capable of meeting the needs of the project. Using this framework, a DSL was defined and implemented as an integrated plugin in an Eclipse RCP, thus generating the ERText tool.

In this way, the modeling process with the newly created language gained native features such as code completion, formatting, validation based on the restrictions described in the grammar, and syntax highlighting. It is important to note that the plugin could only be tested due to the native integration provided by XText with EMF, a set of Eclipse features to represent models and generate code.

With the investigation of the scientific literature and the experience acquired mainly during the design stage of the textual DSL development, we can also mention the occurrence of a preliminary evaluation of a prototype with 13 participants. With the feedback obtained, it was possible to carry out the evolution of development and the definition of an experimental protocol to perform an empirical evaluation of the product developed.

Consequently, a controlled experiment (doi.org/10.5281/zenodo.4064991) was carried out with 27 participants, all students of different levels and belonging to SE area. This evaluation used two treatments, one with a graphical approach and the other with ERtext, with the groups of subjects randomly balanced. With the data collected it was possible to execute a quantitative and qualitative analysis of the tool's viability. The results obtained show evidence that, when performing modeling tasks with both approaches, there is less effort associated with the graphical approach. We believe this is due to the fact that the textual approach to conceptual modeling was not familiar to the subjects of the experiment. On the other hand, the performance was very similar regarding the quality of the models made in both tools, indicating some potential for competition of the proposal concerning to the graphical approach tools.

As future work, the solution is expected to generate SQL code for different technologies (*e.g.* MySQL, PostgreSQL), including as an improvement not only the generation of DDLs but also, for example, stored procedures of CRUD operations for each modeled entity. Besides, we intend to generate Object Relationship Mapping (ORM) input structures, for instance Hibernate and Entity Framework.

Finally, the project for this solution is publicly available under the EPL-2.0 license in the GitHub repository (Repository: github.com/ProjetoDSL/ERDSL), belonging to the Laboratory of Empirical Studies in Software Engineering (LESSE) research group of Unipampa.

REFERENCES

Al-Dmour, A. (2010). A cognitive apprenticeship based approach to teaching relational database analysis and design. *Journal of Information & Computational Science*, 7(12):2495–2502.

Celikovic, M., Dimitrieski, V., Aleksic, S., Ristic, S., and Lukovic, I. (2014). A DSL for EER data model specification. In *23rd Int. Conf. on Information Systems Development*, pages 290–297, Croatia. Springer.

Chen, P. P.-S. (1976). The entity-relationship model: Toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36.

Connolly, T. M. and Begg, C. E. (2006). A constructivist-based approach to teaching database analysis and design. *Information Systems Education*, 17(1):43–54.

Date, C. and Warden, A. (1990). *Relational database writings, 1985-1989*. Number v. 1 in Relational database / C.J. Date. Addison-Wesley Longman Publishing Co., Boston, MA, USA.

Dimitrieski, V., Čeliković, M., Aleksić, S., Ristić, S., Alargt, A., and Luković, I. (2015). Concepts and evaluation of the extended entity-relationship approach to database design in a multi-paradigm information system modeling tool. *Comput. Lang. Syst. Struct.*, 44(PC):299–318.

Edmunds, H. (2000). *The Focus Group Research Handbook*. McGraw-Hill Education.

Elmasri, R. and Navathe, S. B. (2015). *Fundamentals of Database Systems*. Pearson, 7th edition.

Frisina, A. (2006). Back-talk focus groups as a follow-up tool in qualitative migration research: The missing link? *Qualitative Social Research Forum*, 7.

Heuser, C. A. (2009). *Projeto de Banco de Dados*. Bookman, Porto Alegre, BR.

Kelly, S. and Tolvanen, J.-P. (2008). *Domain Specific Modeling: Enabling Full Code Generation*. IEEE Computer Society - John Wiley & Sons.

Kessler, J., Tschuggnall, M., and Specht, G. (2019). Relax: A webbased execution and learning tool for relational algebra. In *Proceedings of Datenbanksysteme für Business, Technologie und Web*, pages 503–506, Bonn, Germany. Gesellschaft für Informatik, Bonn.

Kontio, J., Bragge, J., and Lehtola, L. (2008). *The Focus Group Method as an Empirical Tool in Software Engineering*, pages 93–116. Springer London, London.

Tong, A., Sainsbury, P., and Craig, J. (2007). Consolidated criteria for reporting qualitative research (COREQ): a 32-item checklist for interviews and focus groups. *Int. Journal for Quality in Health Care*, 19(6):349–357.

Wohlin, C., Runeson, P., Hst, M., Ohlsson, M. C., Regnell, B., and Wessln, A. (2012). *Experimentation in Software Engineering*. Springer Publishing Company, Incorporated, London, England.