Derivation of Logical Aspects in Praxeme from ReLEL Models

Rapatsalahy Miary Andrianjaka[1], Razafimahatratra Hajarisena[1], Ilie Mihaela[2] Mahatody Thomas[1], Ilie Sorin[2] and Razafindrakoto Nicolas Raft[3]

[1]*Laboratory for Mathematical and Computer Applied to the Development Systems, University of Fianarantsoa, Fianarantsoa, Madagascar*

[2]*Dept. of Computers and Information Technology, University of Craiova, Craiova, Romania*

[3]*Laboratory of Multidisciplinary Applied Research, University of Antananarivo, Antananarivo, Madagascar*

Keywords: Service-Oriented-Architecture, Model-Driven-Architecture, Requirements Model, Praxeme Methodology, Universe of Discours, ATL, ReLEL, UML Modeling.

Abstract: The Praxeme methodology for enterprise architecture combines MDA (Model Driven Architecture) and SOA (Service Oriented Architecture) approaches by structuring the information system with a basic unit called the *logical service*. To capture the complexity of the system, it separates the facets of the company into a homogeneous set called *aspects*. The eLEL (elaborate Lexicon Extended Language) requirement model groups together the terms used by the company with their precise definition which makes it well suite to be integrated into the initial phase of Praxeme called the *intentional aspect*. From this aspect, eLEL can derive business logic services for the *logical aspect* of Praxeme. The logical aspect of Praxeme plays an important intermediary role between the enterprise and IT. It offers the possibility of describing the computer system in terms independent of technology. However, business logic services obtained from eLEL are not exploitable as skeletons of code of an object-oriented application, which is the next phase of Praxeme called the *software aspect*. For this reason we chose to use the ReLEL (Restructuring extended Lexical elaborate Language) requirement model for the initial phase of Praxeme, the *intentional aspect*. ReLEL is a terminology database and has very precise information about the conceptual representation of an information system. For this reason, we were able to create an automated derivation process using ATL (ATLAS Transformation Language) that uses the intentional aspect represented with ReLEL, to obtain the semantic and logical aspects of Praxeme. To validate our approach, we evaluated the performance of the two different methods on the same case study. The performance show our proposed approach is superior to eLEL, the most recent comparable requirements model. ReLEL offers 92% accuracy on the generated logic model in the *logical aspect* of the Praxeme methodology compared to eLEL which offers just 61.3%.

1 INTRODUCTION

The development of a company's information system is a complex task (Boussis & Nader, 2012). Among the greatest difficulty it encounters is describing the synergy of expertise and interactions that the company requires for normal operation (Vauquier, 2006). Therefore, the Praxeme methodology is essential to overcome this obstacle. Praxeme shares a common framework for accommodating skills and their articulation (Vauquier, 2006). The Praxeme methodology provides a framework of representation for the topology of the enterprise system. This system is categorized in seven aspects, each of which is represented by a particular UML model (Rumbaugh et al., 1999) and (Razafindramintsa et al., 2017). This paper deals in particulary with the three representative aspects of enterprise architecture : i) the intentional aspect, ii) semantic aspect (Rapatsalahy et al., 2020) as well as ii) the logical aspect of the Praxeme methodology (Razafindramintsa et al., 2017). The intentional aspect containes the detailed user requirements of the enterprise. (Biard et al., 2013) states that the intentional aspect of Praxeme must be expressed in a language close to natural language so that it is understood by all. Thus, there is the need for a lexicon called natural language oriented requirements model to represent it. Since the ReLEL requirement model is a terminology database and has very precise information about the conceptual representation of a system (Rapatsalahy et al., 2019,2020), the main contributions in this paper are based on four very distinct points. First, we instantiate the ReLEL requirement model in Praxeme to represent the intentional aspect of the methodology. Then we defined transformation rules to derive the semantic

Andrianjaka, R., Hajarisena, R., Mihaela, I., Thomas, M., Sorin, I. and Raft, R.
Derivation of Logical Aspects in Praxeme from ReLEL Models.
DOI: 10.5220/0010493004130420
In *Proceedings of the 16th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2021)*, pages 413-420
ISBN: 978-989-758-508-1

aspect of Praxeme from the ReLEL requirement model. Then we proposed a logical factory metamodel for the logical aspect of Praxeme. Finally, we have defined transformation rules to derive the logical aspect from the semantic aspect of Praxeme.

2 RELATED WORK

This section summarizes the research literature has focused on the use of the requirement model such as SBVR, eLEL and ReLEL in the Praxeme methodology (Biard et al., 2013), (Razafindramintsa et al., 2016), (Razafindramintsa et al., 2017) and (Rapatsalahy et al., 2020).

(Biard et al., 2013) present an approach to use the Praxeme methodology to transform the company. The authors begin with the pragmatic aspect in order to extract from the messages exchanged between the various actors the business objects and their data which constitute the semantic aspect of Praxeme. Then to etablish the business and organizational rules in the semantic and pragmatic aspects, they advocate the use of the SBVR to represent the intentional aspect of Praxeme.

(Razafindramintsa et al., 2016) have proposed the automatic derivation of the semantic aspect of Praxeme from the eLEL requirement model. The authors used the eLEL requirement model to represent the intentional aspect of Praxeme. eLEL is a lexicon rich in conceptual information proposed in (Razafindramintsa et al., 2015). They established rules for the derivation of the eLEL requirement model into a UML class diagram and transition state machine of the semantic aspect of the Praxeme methodology. Then to make automatic the articulation of the intentional aspect in semantics, they formalized the derivation rules with the ATL language (Jouault & Kurtev, 2005).

(Razafindramintsa et al., 2017) present an approach to automatically locate logical services from the eLEL model to reduce the time spent on SOA modeling and implementation as well as enterprise architecture. To do this, the authors derived the semantic and pragmatic aspect obtained in logic factory, data structure and logical workshop from the logical aspect of Praxeme.

(Rapatsalahy et al., 2020) aim at automatically generating software components from the intentional aspect of the Praxeme methodology represented by ReLEL. To do so, they have proposed rules to automatically translate the logical components obtained from ReLEL and the semantic model into software components while taking into account the choice of technology used in the technical aspect of Praxeme. ReLEL is an extension of the eLEL metamodel in terms of its conceptual representation (Rapatsalahy et al., 2019).

3 BACKGROUND

This section presents the three main concepts developed in this work. First, it describes the service architecture, the approach used by the Praxeme methodology to build a system. Second, it presents the ReLEL lexicon, a model that allows to represent the intentions of the enterprise within the intentional aspect of Praxeme. Finally, this section explains the concept of model transformation with MDA, which is used during articulating aspects of the Praxeme methodology.

3.1 Service Architecture

(Medvidovic & Taylor, 2010) argues that software architecture is at the heart of every system and thus plays a key role as a bridge between requirements and implementation. Therefore, using their reasoning, our approach would be a means to control the complexity of systems construction and evolution. However, the enterprise requires the software architect to manage a large amount of information and questions that are useful to describe different jobs and business (Vauquier, 2006). This suggests that using an enterprise architecture method would help. The main goal of enterprise architecture is to perfectly align an enterprise's information system with its strategy (Biard et al., 2013). For this purpose we use Praxeme, which is an enterprise architecture method that makes it possible to achieve this objective by grouping the facets of the enterprise in categories called "aspects". An aspect is a component of the system that is linked to a point of view, a type of concern, a specialization (Vauquier, 2006). The Praxeme methodology is meant to design software that is to be implemented using a Service Oriented Architecture (SOA) (Valipour et al., 2009). Consequently, it structures the system with the elementary unit called logical service. The logical services' role is to respond to the need for information, action or transformation (Vauquier, 2006). They are arranged in logical aggregates on three levels of aggregation, namely, logical machines, logical workshops (packages grouping strongly linked machines), and logical factories (packages corresponding to domains) (Fig. 1) (Vauquier, 2006). There are two categories of services depending on its location in either a business logic machine or an organizational logic machine. In order to solve the problem raised in this paper, we focus on the services located in the business logic machine because they are the preferable means to identify the software components (class, method) of Praxeme (Rapatsalahy et al., 2020).

Figure 1: Nesting of the logical aggregates (Vauquier, 2006).

3.2 Restructuring Extended Lexical Elaborate Language

ReLEL is a model expressed in natural language allowing to specify both the requirements and the conceptual level of a system (Rapatsalahy et al., 2019). It captures significant words or phrases in the UofD and then saves them as "symbols". Each symbol in the ReLEL lexicon belongs to one of the following categories: subject, object, verb or state. Symbols that are classified as subject, verb and object are the elements integrating requirements (Niu & Easterbrook, 2008). The conceptual linking of ReLEL symbols is done through the concept of circularity (Rapatsalahy et al., 2019). The three objectives of ReLEL are: (i) the unification of the language allowing the communication with stakeholders, (ii) the specification of requirements and (iii) the accurate representation of conceptual information corresponding to each term. The ReLEL metamodel is composed of nineteen classes (Rapatsalahy et al., 2019).

3.3 Notion of Model Transformation with MDA

MDA is a software development framework that is based on semi or automatic model transformation. The key principle of MDA consists in the use of models, namely the requirement model (CIM), the analysis and design model (PIM) and the code model (PSM) at the different phases of the software development cycle (Blanc & Salvatori, 2011). In this paper we transform the intentional aspect (at CIM level) of Praxeme into the semantic aspect (at PIM level) and then further transform the semantic aspect into the logical aspect (also at PIM level). We used the technique of exogenous transformation which means that the source and target model are derived from a different metamodel (Diaw et al., 2010), (see Fig. 4). The transformation process by the modeling approach is done through derivation rules that describe the correspondence between the symbols of the source model and those of the target model (Diaw et al., 2010).

4 THE DERIVATION PROCESS

The Praxeme methodology is based on the MDA standard in order to articulate the models corresponding to each aspect of the company. Fig. 2 represents the synoptic of the approach proposed in this article. The model transformation presented in Fig.2 belongs to the Model to Model group or M2M which generates target models such as the semantic 'red' and logical model 'yellow' from source models such as the intentional model 'black' as well as the semantic model 'red' (Jézéquel et al., 2012).

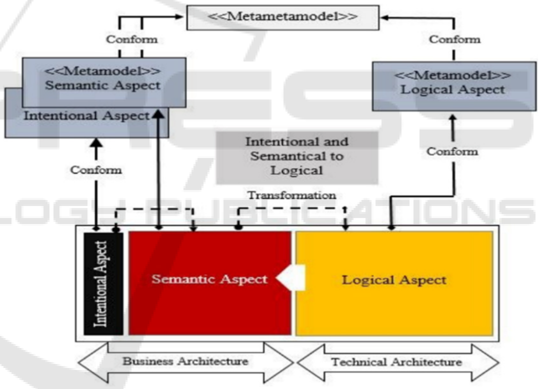

Figure 2: Derivation process of aspects of Praxeme in the context of MDA.

4.1 ReLEL Derivation into Semantic Aspect of Praxeme

It is important to group together in the intentional aspect all the terms used specifically in the company and/or in its field of activity, with their precise definitions (Boussis & Nader, 2012). (Rapatsalahy et al., 2020) shows that the ReLEL requirement model groups together the specific terms used by the company as well as their exact definitions, so we use ReLEL requirement model proposed by (Rapatsalahy et al., 2019) instead of the eLEL (Razafindramintsa et al., 2015) for the representation of the firm's intentions at the level of the intentional aspect of Praxeme. This approach facilitates and allows all level of abstraction changes (business, logical and technical) for each existing aspect of the Praxeme methodology. The semantics aspect houses the knowledge of the business of the enterprise while excluding the references related to the organization and means to control them. The UML class diagram and the state diagram are representative models of the semantic aspect of Praxeme (Vauquier, 2006). Our approach begins with the definition of the transformation rules (theoretical approach) of the ReLEL into a UML class diagram. The UML class diagram of the semantic aspect that we proposed as a target model is composed of a package, class, attribute, method and association. ***Rule 1*****:** A domain is a concept for storing ReLEL symbols classified subject and object. The domain corresponds to a UML package. ***Rule 2*****:** The subject classified ReLEL symbol corresponds to the actors in UofD. Actors can be individuals or part of an organization. The object classified ReLEL symbol represents passive and significant entities in the UofD. Each subject and/or object classified ReLEL symbols corresponds each to a UML class. ***Rule 3*****:** The attribute provides the characteristics of a ReLEL symbol. The attributes of each ReLEL symbol (subject or object) correspond to the attributes of each UML class obtained in rule 2. Then, we deduce the characteristic elements (definition, code, size) of each attribute. Rule 7 allows you to obtain the type of each attribute. ***Rule 4*****:** The method is the action that allows access to a ReLEL object. Each method of a ReLEL symbol (subject or object) corresponds to the operation of each UML class obtained in rule 2. ***Rule 5*****:** A parameter is none other than an already existing attribute which is manipulated by a method of a ReLEL object. The parameter of each method of the ReLEL symbol (subject or object) corresponds to the parameter of each operation of UML class obtained in rule 2. ***Rule 6*****:** The return value is the response returned by the method after handling a ReLEL object. The return value of each method belonging to a ReLEL symbol (subject or object) corresponds to the return value of each operation of a UML class obtained in rule 2. ***Rule 7*****:** The type of an attribute or the type of the return value of a method of a ReLEL object (subject or object) is either a classified ReLEL symbol (subject or object) or a simple type. The type (simple or symbol) of an attribute of a ReLEL object corresponds to the type (primitive or class) of an attribute of a UML class. The type (simple or symbol) of a return value of a method of a ReLEL object corresponds to the type (primitive or class) of the return value of an operation of a UML class. ***Rule 8*****:** The concept of circularity links two target and source ReLEL objects. The concept of the circularity of the ReLEL object corresponds to the association between two different UML class. ***Rule 9*** **:** The concept of number of elements created defines the minimum and maximum occurrence of an association between two symbols ReLEL. The concepts of the number of elements created corresponds to each UML class cardinality.

The rules for deriving the intentional aspect represented by ReLEL in a diagram of the UML state machine of the semantic aspect of Praxeme are as follows: ***Rule 1*****:** The state classified ReLEL symbol is characterized by attributes that contain values at different times during system execution. Each state type ReLEL symbol becomes state in the UML diagram of the state transition machine of the semantic aspect of Praxeme. ***Rule 2*****:** Circularity allows to link two target and source ReLEL objects classified as state. The target and source ReLEL objects classified state in the circularity corresponds to the transition of the event of the UML diagram of the state machine of transition of the semantic aspect of Praxeme. ***Rule 3*****:** A state is triggered by the event of another state. Each method belonging to a state type ReLEL symbol corresponds to the event of the UML diagram of the state transition machine of the semantic aspect of Praxeme.

4.2 Derivation from the Semantic Aspect to the Logical Aspect of Praxeme

The logical aspect of the Praxeme methodology is an asset for the opening of the system because it leads to the production of software component published as a service in the logical continuation of web service technologies. In this paper, our focus is particularly on the logical service named BLS. It is described in the logical aspect of Praxeme from the semantic modeling of the methodology. The BLS comes from the derivation of an operation belonging to a semantic class. It is located in the Business Logical Machine or BLM of the Praxeme logic model that derives from a semantic class as well. The BLM is encapsulated in the Logical Workshop or LW which does not correspond to any element in the upstream model but it results from the decision of structuring taken by the logical architect during the logical design. Thus, the access to the BLS is either directly between the BLM located in the same LW or through the interface services provided by the LW. The relationship between the BLM is only reflected in the usage relationship that is a functional dependency achieved at the time of BLS service call execution. Finally, the aggregates of the LW are stored in Logical Factory or LF.

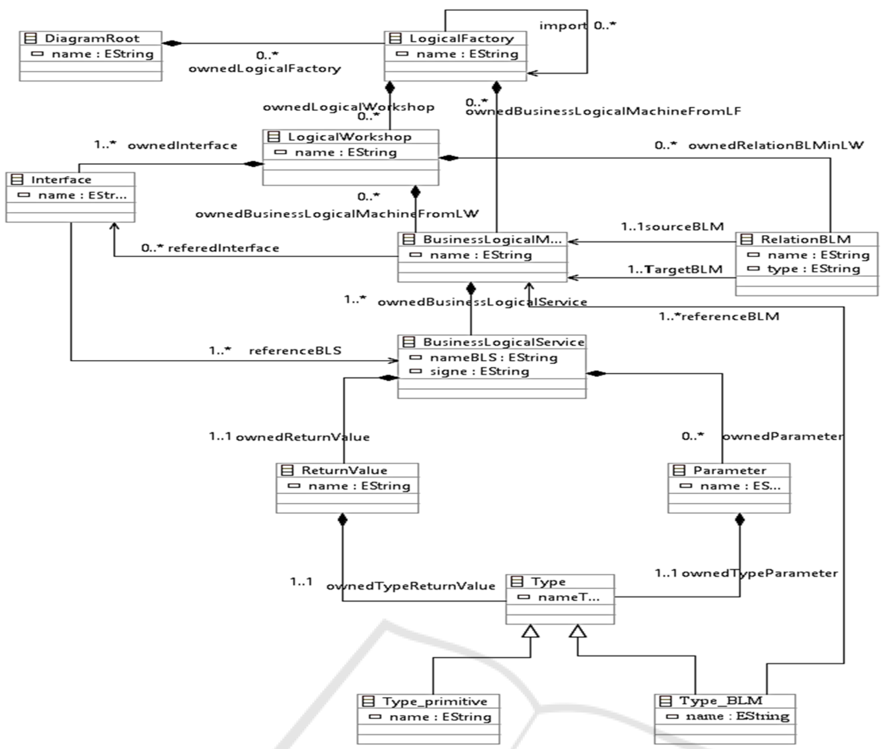

Figure 3: Logical Factory Metamodel proposed.

The derivation of the semantic aspect in logical aspect is constituted by seven rules including: ***Rule 1*****:** Each UML package of the semantic aspect corresponds to the LF of the logical aspect of Praxeme. ***Rule 2*****:** Each class of the UML class diagram of the semantic aspect corresponds to each BLM of the logical aspect of Praxeme. ***Rule 3*****:** Each UML class diagram operation of the semantic aspect corresponds to each BLS of the logical aspect of Praxeme. ***Rule 4*****:** Each parameter of a UML class diagram operation of the semantic aspect corresponds to each parameter of a BLS of the logical aspect of Praxeme. ***Rule 5*****:** Each return value of a UML class diagram operation of the semantic aspect corresponds to each BLS return value of the logical aspect of Praxeme. ***Rule 6*****:** Each primitive type of a parameter or return value of an operation of a UML class diagram of the semantic aspect corresponds to a primitive type of a parameter or return value of a BLS of the logical aspect of Praxeme. ***Rule 7*****:** Each class type of a parameter or return value of an operation of a UML class diagram of the semantic aspect corresponds to a BLM type of a parameter or a value of return of a BLS of the logical aspect to the Praxeme methodology.

4.3 Formalization of the Derivation Process with ATL

To define and execute the ATL transformation, we must define a source model and a target model as the ecore metamodel (Budinsky et al., 2004). In the ATL transformation from the intentional aspect into the semantic aspect, the metamodel ecore ReLEL (Rapatsalahy et al., 2019) represents the metamodel of the intentional aspect of Praxeme and is defined as a source model. Then we defined two ecore metamodels as the target model for the same ATL transformation, one for the UML class diagram and the other for the transition state machine diagram. The resulting derivation results in two target UML models representing in XMI (XML Metadata Interchange) format such as the class diagram model and the model of the transition state machine. In order to execute the ATL transformation of the semantic aspect rule into a logical one, we defined an ecore metamodel of LF (Fig. 3) for the logical aspect of Praxeme as target model (Fig. 4).

M CD: Model of the UML class diagram of the semantic aspect of Praxeme; ***MM CD:*** Metamodel of the UML class diagram of the semantic aspect of Praxeme; ***M LF:*** Model of logical factory of praxeme logical aspect; ***MM LF:*** Metamodel of the logical factory of the Praxeme logical aspect.

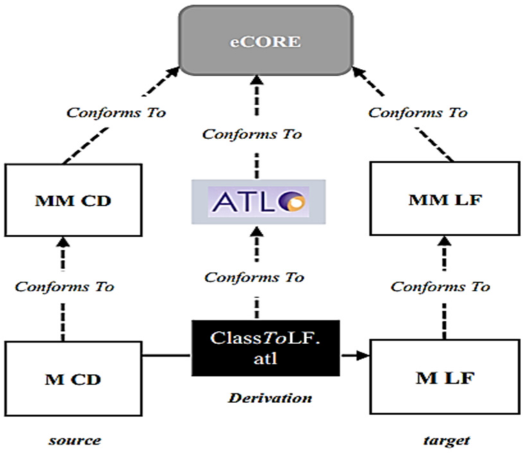

Figure 4: Derivation of the logical factory from a class diagram with ATL.

5 VALIDATION STRATEGY

For the experiment, we used the land travel booking process of a transport agency (Rapatsalahy et al., 2019). Only an excerpt from the need of the '*Transpost Malagasy*' agency from the UofD is illustrated in this paper.

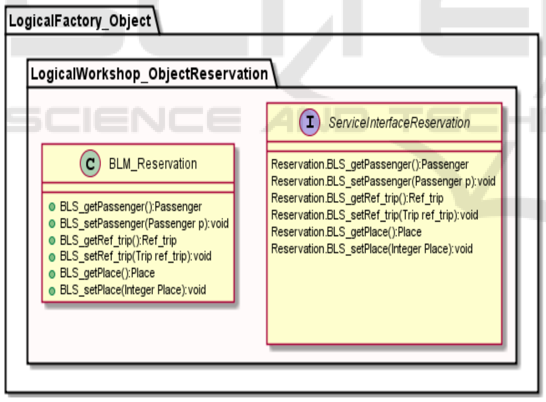

Figure 5: Represents the concrete logic model of the Praxeme methodology.

The agency proposes a trip in 'bush taxi' which connects two cities. The customer can make a reservation for a trip by registering the concerned passenger. The significant terms of UofD concerning the extract of the need of the agency previously are instantiated in the model ReLEL in the form of symbol classified by topology. We applied the derivation rules and the ATL transformation described in this paper on this case study. Finally, we automatically obtained the logic model of the logical aspect of Praxeme as result of the derivation represented by Fig. 5.

5.1 Comparison of the Logical Models Obtained by the eLEL Compared to the Approach Proposed in This Paper

We assessed the performance of our approach using two different methods to process the case study of the online booking from a land travel agency (Rapatsalahy et al., 2019). Method 1 consists in using the ReLEL requirement model in the intentional aspect of Praxeme and then deriving it in a semantic and logical aspect. Method 2 is identical except that we used the eLEL requirement model to represent the intentional aspect of Praxeme. Our approach consists firstly of counting the elements generated automatically and by refinement in the logical models of the two different methods. Tables 1 and 2 illustrate the elements generated automatically and by refinement in the logical models as well as their numbers from the two different methods. BLM is coherent set of BLS built around a strong notion of a semantic model. The total number of BLM generated automatically (each 8 in the Table I) for the two methods are equivalent since the eight semantic classes of the source model are all kept as they are during the derivation in BLM of the logical model but simply coated in the LW. The total number of BLS automatically generated in the logic model (target model) of the two methods 1 and 2 (each 57) comes from the 57 operations located on the eight semantic classes of the source model. The parameter is a data manipulated by the BLS having two types such as primitive and/or complex (BLM) (Rapatsalahy et al., 2019). The total number 20 (type primitive/BLM) belonging to the BLS parameters comes from the number 20 (type primitive /BLM) belonging to the parameters of the operations located on the eight semantic classes of the upstream model. The return value is a data returned by the BLS after its execution and has two types such as primitive and/or complex (BLM). The total number (16) of the primitive type belonging to the return values of the BLS comes from the 16 primitive types belonging to the return values of the operations located on the eight semantic classes of the upstream model. The 10 BLM types belonging to the BLS return values derive from the 10 class types belonging to the return values of the operations located on the eight semantic classes of the upstream model. Method 2 does not allow to generate a parameter or a return value because of the requirement model representative of the intention aspect of Praxeme (Rapatsalahy et al., 2019). The LW (each 3), the interface of the workshop (each 3) as well as the relationships in the logical architecture

Table 1: Elements generated automatically.

Approach	BLM	BLS	Parameter		Return value		Total
			Prim	BLM	Prim	BLM	
Method 1	8	57	20	20	16	10	31
Method 2	8	57	0	0	0	0	65

(each 10) are purely logical notions which result from a decision by the logical architect from where Table 2. Method 2 did not take into account in its logical factory metamodel the elements which have a purely logical notion namely LW, interface of the workshop, relationships in the logical architecture. This is what explains the absence of method 2 in the Table 2.

5.2 Evaluation of the Performance of the Proposed Approach

The metrics used are based on the complete state of the automatically generated model (Recall), with the accuracy of the generated information (Precision), the measurement of the number of information considered to be correct but not automatically generated by the Overspecification and the accurarcy of the proposed strategy (F-Measure). We use the parameters used in the table 1 and 2 to calculate the metrics adopted by CM-Builder's (Harmain & Gaizauskas, 2003). Thus in Table 3, we summarize the results obtained in order to measure the different metrics necessary for this evaluation. From Table 3, we obtained the metrics allowing to evaluate the completeness, the precision, the addition of the information considered important and the accuracy of the automatic generation of logical model of Praxeme. The analysis in Table 4 allows us to deduce that the method 1 proposed in this paper is the most suitable for obtaining BLS exploitable by developers (Rapatsalahy et al., 2020). Indeed the deduction is explained by the result of the metric 'F-Measure' which indicates that method 1 provides 94.2% of the exact logic model compared to method 2 only 61.3%. Therefore method 1 generates 89.1% of the full state of the automatically generated logic model and method 2 only 44.2% (Recall). Nevertheless, all information extracted from the two methods is correct, hence their precisions which each reach a value of 100% (Precision). Yet the overspecification of method 2 is of 55.7% because of the absence of conceptual elements in the logic model such as the parameter, type of parameter, the return value, the type of return value of BLM. In addition there are also the LW of the business stratum and their interfaces services to be manually added by the logical architects in the logic model for the two different methods. It should also be noted that the relationship between BLM for both methods (1 and 2) is still manually added by the logical architects. In spite of this, method 1 only requires the manual addition of 10.8% of the information considered correct but which is not generated automatically (Overspecification). Thus, the results of the performance evaluation are significant and support the approach proposed in this paper.

Table 2: Elements generated by refinement.

Approach	LW	Interface	Relation with BLM	total
Method 1	3	3	10	16

Table 3: Nature of information for measuring the metrics of the automatically generated logic model.

Nature of information	N_{key}	$N_{correct}$	$N_{incorrect}$	N_{extra}
Method 1	147	131	0	16
Method 2	147	65	0	82

Table 4: Performance evaluation.

Metrics	Equation	Method 1	Method 2
Recall (%)	$R=N_{correct}/N_{Key}$	89.1%	44.2%
Precision (%)	$P=N_{correct}/(N_{correct}+N_{incorrect})$	100%	100%
Overspecification (%)	$O=N_{Extra}/N_{Key}$	10.8%	55.7%
F-Measure (%)	$F\text{-Measure}=(2\times R\times P)/(R+P)$	94.2%	61.3%

6 CONCLUSION AND FUTURE WORK

In this paper, we have proposed the automatic derivation of the logical aspect of Praxeme from the requirement model noted ReLEL. Our goal is to make the *Business Logical Service* or BLS of the logical model of Praxeme exploitable by the developers. The importance of our study lies in being able to attribute to the logical aspect of Praxeme its intermediary role between purely business and technical aspects. To achieve the goal of the research, our approach consists of four steps, the first of which is to replace the eLEL requirement model which represents the intentional aspect of the Praxeme methodology by ReLEL (Rapatsalahy et al., 2020). And since the operation of the semantic class is the source of the BLS in the logic model, we then proposed to automatically derive the intentional aspect represented by ReLEL in the semantic aspect of Praxeme. Next we proposed a logical ecore metamodel to form the target model in the automatic derivation of the semantic aspect in the logical aspect of Praxeme. Finally, we have defined a transformation rule that allows to automatically derive the semantic aspect in logical aspect of Praxeme methodology. In order to validate our approach, we conducted a comparative study between the two different methods using the CM-Builder's metric and instantiating them in the same case study. The results of comparison concerning the accuracy of the proposed approach (F-Measure) allowed us to deduce that the logic model provided from our approach with a very high percentage 94.2% is composed of exploitable BLS by the developers (Rapatsalahy et al., 2020). Therefore, our approach easily allows the developer to translate BLM into software components (Rapatsalahy et al., 2020). The urbanization of the information system from the lexicon ReLEL is the research perspective of this present work.

REFERENCES

Rapatsalahy, A. M., Razafindramintsa, J. L., Mahatody, T., Ilie, S., & Raft, R. N. (2019). Restructuring extended Lexical elaborate Language. *2019 23rd International Conference on System Theory, Control and Computing (ICSTCC)*, 266-272.

Rapatsalahy, A. M., Razafimahatratra, H., Mahatody, T., Ilie, M., Ilie, S., & Raft, R. N. (2020). Automatic generation of software components of the Praxeme methodology from ReLEL. *2020 24th International Conference on System Theory, Control and Computing (ICSTCC)*, 843-849.

Biard, T., Bigand, M., & Bourey, J.-P. (2013). Explicitation et structuration des connaissances pour la transformation de l'entreprise : Les apports de la méthode Praxeme. *Congrès International de Génie Industriel*.

Blanc, X., & Salvatori, O. (2011). *MDA en action : Ingénierie logicielle guidée par les modèles*. Editions Eyrolles.

Boussis, A., & Nader, F. (2012). Urbanization of Information Systems with a Service Oriented Architecture according to the PRAXEME Approach. *Web and Information Technologies*, 102.

Budinsky, F., Ellersick, R., Steinberg, D., Grose, T. J., & Merks, E. (2004). *Eclipse modeling framework : A developer's guide*. Addison-Wesley Professional.

Diaw, S., Lbath, R., & Coulette, B. (2010). État de l'art sur le développement logiciel basé sur les transformations de modèles. *Technique et Science Informatiques*, *29*(4-5), 505-536.

Harmain, H. M., & Gaizauskas, R. (2003). Cm-builder : A natural language-based case tool for object-oriented analysis. *Automated Software Engineering*, *10*(2), 157-181.

Jouault, F., & Kurtev, I. (2005). Transforming models with ATL. *International Conference on Model Driven Engineering Languages and Systems*, 128-138.

Leite, J. do P., & Franco, A. P. M. (1993). A strategy for conceptual model acquisition. *[1993] Proceedings of the IEEE International Symposium on Requirements Engineering*, 243-246.

Razafindramintsa, J. L., Razafimandimby, J. P., Mahatody, T., & Becheru, A. (2016). Semantic aspect derivation of the Praxème methodology from the elaborate lexicon extended language. *2016 20th International Conference on System Theory, Control and Computing (ICSTCC)*, 842-847.

Medvidovic, N., & Taylor, R. N. (2010). Software architecture : Foundations, theory, and practice. *2010 ACM/IEEE 32nd International Conference on Software Engineering*, *2*, 471-472.

Niu, N., & Easterbrook, S. (2008). Extracting and modeling product line functional requirements. *2008 16th IEEE International Requirements Engineering Conference*, 155-164.

Razafindramintsa, J. L. (2015). Elaborate Lexicon Extended Language with a lot of conceptual informa-tion. *International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol*, *5*.

Razafindramintsa, J. L., Mahatody, T., Razafimandimby, J. P., & Simionescu, S. M. (2017). Logical services automatic location from eLEL. *2017 21st International Conference on System Theory, Control and Computing (ICSTCC)*, 849-854.

Rumbaugh, J., Jacobson, I., & Booch, G. (1999). The unified modeling language. *Reference manual*.

Valipour, M. H., AmirZafari, B., Maleki, K. N., & Daneshpour, N. (2009). A brief survey of software architecture concepts and service oriented architecture. *2009 2nd IEEE International Conference on Computer Science and Information Technology*, 34-38.

Vauquier, D. (2006). Modus la méthodologie PRAXEME Guide general. *Praxeme Institute*.

Jézéquel, J.-M., Combemale, B., & Vojtisek, D. (2012). *Ingénierie Dirigée par les Modèles : Des concepts à la pratique...* Ellipses.