


## SBVR: A Study on Model Transformations

Anderson Nobre Santos <sup>1\*</sup>, Paulo Caetano da Silva <sup>2</sup>

<sup>1</sup> Master, Computing Postgraduate Program, Universidade Salvador (UNIFACS), Salvador-BA, Brazil

<sup>2</sup> Ph.D, Computing Postgraduate Program, Universidade Salvador (UNIFACS), Salvador-BA, Brazil

\* **Corresponding Author:** [nobreanderson@gmail.com](mailto:nobreanderson@gmail.com)

**Citation:** Santos, E. N., & da Silva, P. C. (2024). SBVR: A Study on Model Transformations. *Journal of Information Systems Engineering and Management*, 9(4), 29182. <https://doi.org/10.55267/iadt.07.15406>

### ARTICLE INFO

Received: 19 Jun 2024

Accepted: 30 Aug 2024

### ABSTRACT

Semantics of Business Vocabulary and Business Rules (SBVR) is a standard established by the Object Management Group (OMG) that allows business rules and vocabularies to be written in a formal, structured language, which facilitates communication between business analysts or clients and system analysts and developers responsible for implementing those rules in a system, in addition to allowing automated transformation to other models, such as UML and BPMN. Therefore, it is possible to use the SBVR specification to integrate the Model Driven Architecture (MDA) approach in the initial stages of software development. With the MDA approach being integrated into the early stages of development, it is possible to use one of its main features, Model-To-Model Transformation (M2M), to reduce costs and time and increase productivity. Research involving M2M transformations using SBVR is few, which infers that advances can still be made in this area so, there is a need to carry out further research in this area, to develop new techniques and expand the models that can be generated from an SBVR model. The objective of this article is to present a systematic review of the literature, highlighting the main works in recent years in the area of model transformations with SBVR, in order to present what were the main advances in the area and what contributions can be made to the growth of the area.

**Keywords:** Semantics of Business Vocabulary and Business Rules, SBVR, Model-To-Model Transformation, Model Driven Architecture, MDA.

## INTRODUCTION

Business rules are one of the main pillars in building and maintaining computer systems. Through these rules, the purpose of a system, as well as its functionalities and restrictions, is defined. Therefore, the quality in defining business rules can positively or negatively influence the software development flow.

Usually, business analysts have no understanding of formal models or how to represent the flow of execution of a system in UML (Unified Modelling Language) models, BPMN (Business Process Model and Notation) etc. Consequently, system requirements tend to be written in natural language, which causes ambiguities and inconsistencies in system implementation. To solve this problem, OMG created 2008 the SBVR standard, which allows business rules to be written in a structured language, which can be understood both by people and by computers.

SBVR is a tool that facilitates communication between customers and software engineers, ensuring quality and productivity, reducing the likelihood of ambiguities and inconsistencies in the requirements definition stage. The automatic transformation to a variety of models for implementations of the most diverse types of applications can be a great benefit for organizations. According to Ramzan, Bajwa, Haq, and Naeem (2014), "If we transform natural language requirement specification into SBVR based requirements, we have an opportunity to easily

process these requirements by machine due to the formal logic of SBVR”.

In recent years, several studies have been carried out with a focus both on transforming other models to SBVR and transforming SBVR to other models. Much of the work aims to deal with the SBVR transformation for UML diagrams or to extract SBVR terms from UML diagrams. For example, Bonais, Nguyen, Pardede, and Rahayu (2014) seek to identify a subset to improve the robustness of the automatic transformation through SBVR for UML class diagrams. However, model transformation using SBVR is still an area with a great shortage of work and should be further explored.

SBVR is a relatively new technology that has great potential for introducing different techniques and tools for transforming the SBVR model into the most varied model. In view of this scenario, what gaps can be explored to contribute to the area and improve automatic model transformation techniques and tools using SBVR? This work aims to identify the tools and techniques available to transform models with SBVR, with the objective of verifying which were the main contributions and identifying ways to evolve these solutions and make contributions to the area. In this article, the main research on model transformations involving SBVR will be shown. In Section 2, a brief explanation of the main concepts of SBVR and M2M transformations will be presented. In Section 3, the methodology used to carry out the systematic literature review will be presented. In Section 4, the analysis of the works found in the systematic literature review will be carried out. In Section 5, the conclusion of the article is presented.

## LITERATURE REVIEW

### Model Driven Architecture

MDA is an approach created by OMG that provides guidance for structuring specifications that are expressed through models and architecture to support the physical and organizational life cycle of information technology systems (OMG, 2014). MDA allows you to deal with the complexity and interdependence of systems through models. The role of MDA is to separate details into those pertaining to business requirements and those pertaining to the technologies that implement them. This separation is achieved through the use of parameterized standards to define how a business-focused model can be transformed into a technological solution.

According to the official guide (OMG, 2014), a model is a representation of some aspect of a system based on a specific set of interests. A model can represent a business, a domain, software, hardware, an environment, and other aspects of a system. Various expressions can be included in a model, such as a UML diagram, entity-relationship diagrams, user interface images, etc.

A model needs to be expressed in a way that communicates information about the system it seeks to describe. Stakeholders must be able to correctly interpret what the model seeks to communicate, which requires language. Achieving this understanding implies that the structure, terms, notations, syntax, semantics, and information integrity rules in the model are well-defined and consistent. These elements that are used to express a model constitute a modeling language. Some well-known modeling languages are UML, SQL Schema, BPMN, OWL, etc. A model has to conform to a modeling language. Formal modeling languages have ways of determining whether a model conforms to the language. The best way to express a modeling language is through a model called a meta-model. The goal of the meta-model is to define a modeling language. Modeling tools help you create models that conform to the meta-model of the language being used.

Transforming models means generating different models or artifacts from a model based on a transformation pattern. Model transformation is one of the main attractions of the MDA approach and is a way to reduce costs and increase efficiency in projects. When we talk about transforming models, this means that the content remains the same, but the way in which the information is represented differs, thus making models of different levels of abstraction possible that represent different issues with related points of view. Therefore, model transformation can be used to cross levels of abstractions, going from a more abstract representation to producing a more concrete representation.

A fundamental goal of using templates is to improve communication between a team or a community. Using the MDA approach, it is possible to use models as communication vehicles, allowing the team to reach a common sense. The MDA approach assists this communication in three ways:

- Use well-defined terms, icons, and notations to help understand a specific area (OMG, 2014).
- Using semantic data as a basis for models, allowing management, versioning and sharing (OMG, 2014).
- Use of reusable model libraries, such as vocabularies and general rules, processes, business models and

design pattern architectures (OMG, 2014).

There are several ways to derive value from models: models as communication vehicles, derivation via automatic transformations, simulation and execution of models, structuring unstructured information (OMG, 2014). Deriving model implementations can be done from a fully or partially automated approach. Automation reduces the cost and time to realize a design, reduces the time and cost for changes and maintenance, and produces results that ensure consistency across all derived artifacts (OMG, 2014). Producing model artifacts is much more reliable and faster than implementing a set of processes and services for an organization. Derivation through automatic transformation requires a source model from an independent platform with some parameters specifying the way the source model should be interpreted, how the transformation will be performed and the target artifact for a given platform.

Models can drive simulation engines that help analyse and execute the designs captured by the model. Simulations help in understanding how a system will work as well as how to correctly validate the model. In some cases, models can be directly executed, serving as source code for high-level applications that implement processes, repositories and endpoints.

Documents, web pages, and diagrams contain vast amounts of unstructured data. Although MDA technologies do not have a direct way to extract unstructured information, it is possible to use MDA to define structured representations of these documents. Programs that extract information from unstructured texts can use MDA models as a repository.

For this work, the main focus will be on derivations via automatic transformations. Deriving artifacts and model implementations can be done partially or completely automated. Automating model transformations reduces time and cost in the initial stages of software development, system changes and maintenance, and ensures consistency of specifications.

### **Model-To-Model Transformation**

Model transformation is one of the main features of Model Driven Architecture. Transforming a model means translating a source metamodel to a target metamodel. There are several techniques and technologies to perform a model transformation, also dependent on its area of expertise.

The two most basic techniques to perform a model transformation are to use a programming language, e.g. Java, known as hard-coding technique, or to use an MTL (Model Transformation Language) that can be used by a MDE (Model Transformation Engine) (Skersys, Danenas, & Butleris, 2018). In the hard-coding technique, the developer manually inserts code instructions instead of using a template to adapt codes for different platforms. MTLs are languages that allow to transformation of models while maintaining the semantics of the system and enabling various forms of automation. Among the MTLs, the following stand out: ATL (Atlas Transformation Language), QVT (Query/View/Transformation) and XTend. Hard-coding techniques have disadvantages when compared with MTL-based applications. On the other hand, MTL-based implementations fail to provide user interactions at runtime (Skersys et al., 2018).

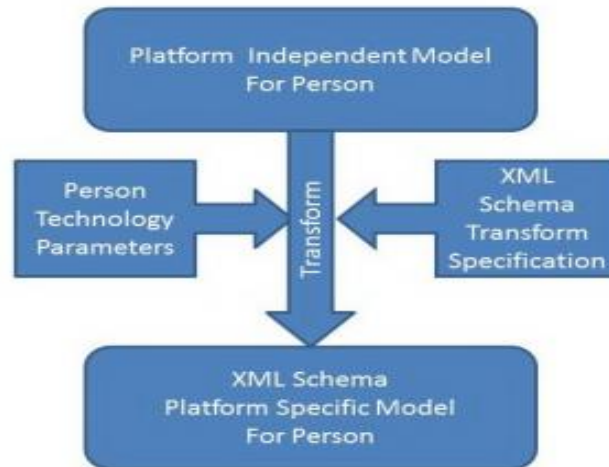
You can separate the types of model transformations considering the level of automation: manual, semi-automatic and automatic. In the manual approach, the transformation is done exclusively by a technician without the help of a dedicated application, that is, the systems analyst reads the specifications and, through them, creates, for example, a diagram in some modelling language. In the semi-automatic approach, a dedicated application is used to perform the transformation, but it will still need the technician's interference to complete the task. In the automatic approach, the transformation is exclusively performed by the application dedicated to model transformation, which can execute all the sets of rules necessary for the transformation without the interference of a technician.

By using the automatic approach in model transformation, there is the possibility of reducing costs, time and risks involved in software development and maintenance and increasing the quality of software processes. Despite this, not all models are suitable for transformation in an automated way. The models need to be sufficiently detailed and precise so that all business requirements for information systems can be expressed in the form of models (OMG, 2014).

The two main approaches to perform the automatic transformation of models are:

- Applying a transformation pattern to a model to produce technological artifacts such as XML Schema or Java code (OMG, 2014).
- An engine that directly executes the model on some technological platform, executes the model in the same way that source code is executed by a compiler (OMG, 2014).

A platform is always necessary for running an information system. There are two platform-related terms within the MDA: Platform Specified Model (PSM) and Platform Independent Model (PIM) (OMG, 2014). The first concept refers to a model of a system defined for implementation on a specific platform, while the second refers to a model that is independent of any platform. The MDA performs the automation of the transformation of a PIM model to PSM. A transformation specification, which may or may not be another model, specifies how a PIM is transformed into a PSM. **Figure 1** shows a representation of this process.



**Figure 1.** Diagram of the PIM to PSM Transformation (OMG, 2014)

Automatic model transformation has some advantages, some of which are:

- Customers are most likely to engage in model validation, improving communication and final product quality (OMG, 2014).
- Transform can be reused (OMG, 2014).
- Optimization can be performed to benefit all systems created by the transformation (OMG, 2014).
- Transformations may be subject to scrutiny from a cybersecurity and reliability perspective (OMG, 2014).
- Transformation can reduce coding time and cost (OMG, 2014).
- Because of the use of patterns, the system produced is very consistent (OMG, 2014).

### **Semantic of Business Vocabulary and Business Rules**

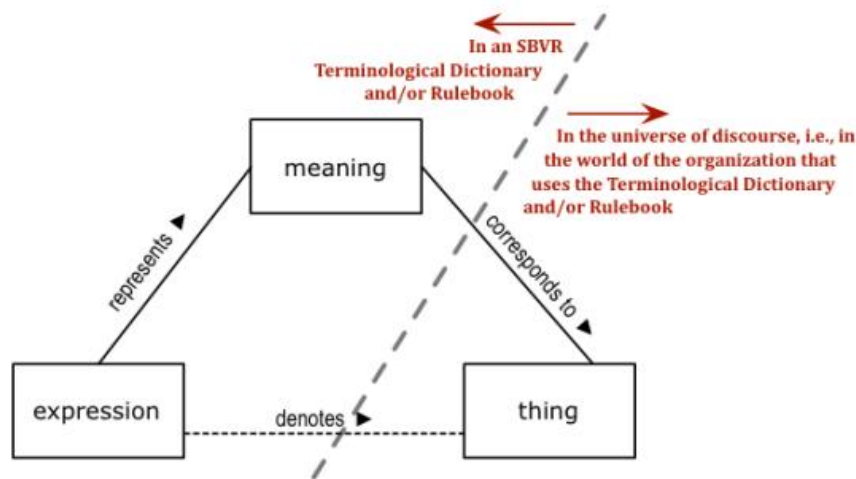
In 2008, OMG published the Semantic of Business Vocabulary and Rules (SBVR) standard, whose main objective is to facilitate communication between those responsible for defining business rules and those responsible for implementing and developing systems in the requirements definition stage. Generally, due to a lack of knowledge in graphical representations such as UML or BPMN, those responsible for defining business rules tend to write these rules in natural language, e.g., English, Portuguese, and in an unstructured way, this increases the risks of becoming ambiguous and inconsistent rules. In addition, performing transformation manually is a labor-intensive activity that costs a lot of time. SBVR facilitates communication between the different people responsible for defining and implementing a system while providing an automated means for machines to process requirements automatically. This was achieved by providing a means of expressing knowledge in a structured natural language, constrained by formal logic (Skersys et al., 2018).

The SBVR specification defines vocabularies and rules to document the semantics of business vocabularies and business rules, enabling the exchange of these rules and vocabularies between organizations or software systems. The specification is applicable to any type of activity or organization, providing an unambiguous, semantic way of defining meaning for language used by people in an industry, profession, discipline, field of study, or organization. SBVR was designed to be used in business applications, regardless of system. It can serve the following purposes:

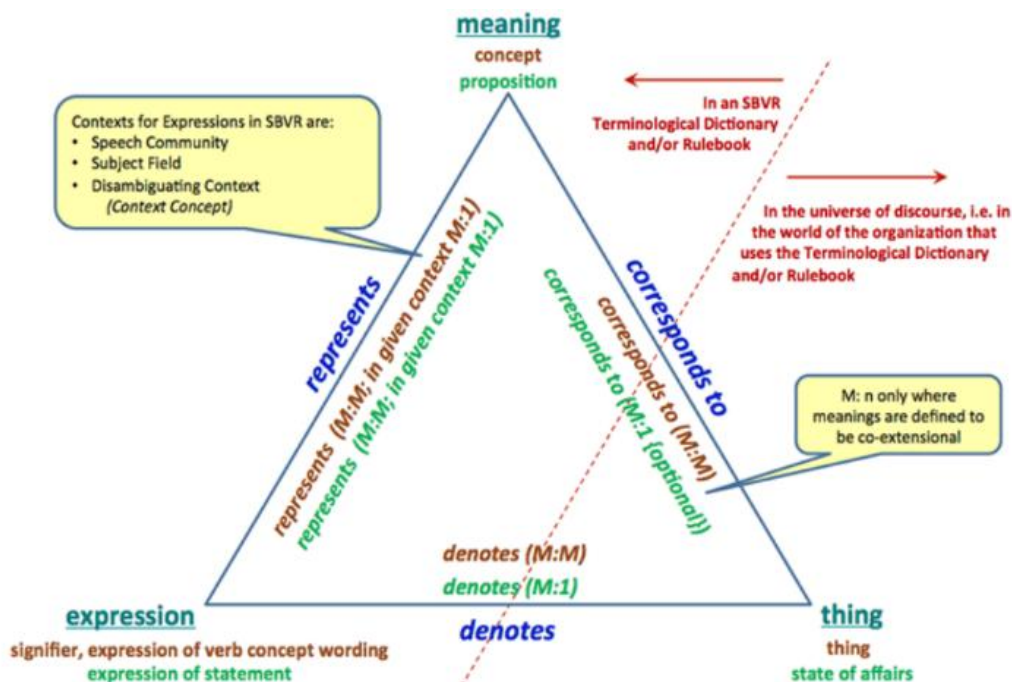
- Unambiguous definitions of the meaning of concepts and business rules, consistent across all terms, names, and representations that are used to represent them, and across the natural languages in which these representations are expressed (OMG, 2012).

- Expression of the meaning of business concepts and rules in verbiage used by business analysts, who may belong to different communities, so each expression is uniquely associated with a meaning in a given context (OMG, 2012).
- Transforming the meaning of concepts and business rules from the way expressed by humans to a form that is suitable for processing by tools and vice versa (OMG, 2012).
- Interpreting the meaning of concepts and business rules with the aim of discovering inconsistencies and gaps within the SBVR conceptual model using logic-based techniques (OMG, 2012).
- Application of the meaning of business concepts and rules to everyday situations in the business environment with the aim of reproducing decisions and identifying compliant and non-compliant business behaviours (OMG, 2012).
- Exchange the meaning of business concepts and rules between humans and tools as well as between tools without losing information about the essence of these meanings (OMG, 2012).

The SBVR has its theoretical basis in the so-called semiotic/semantic triangle. It is the basis for the linguistic architecture and for the separation between expressions and meanings in the SBVR architecture. **Figures 2 and 3** show the representation of the Semiotic/Semantic Triangle in a diagram and in SBVR terms, respectively.



**Figure 2.** Diagram Representing the Semiotic/Semantic Triangle (OMG, 2012)



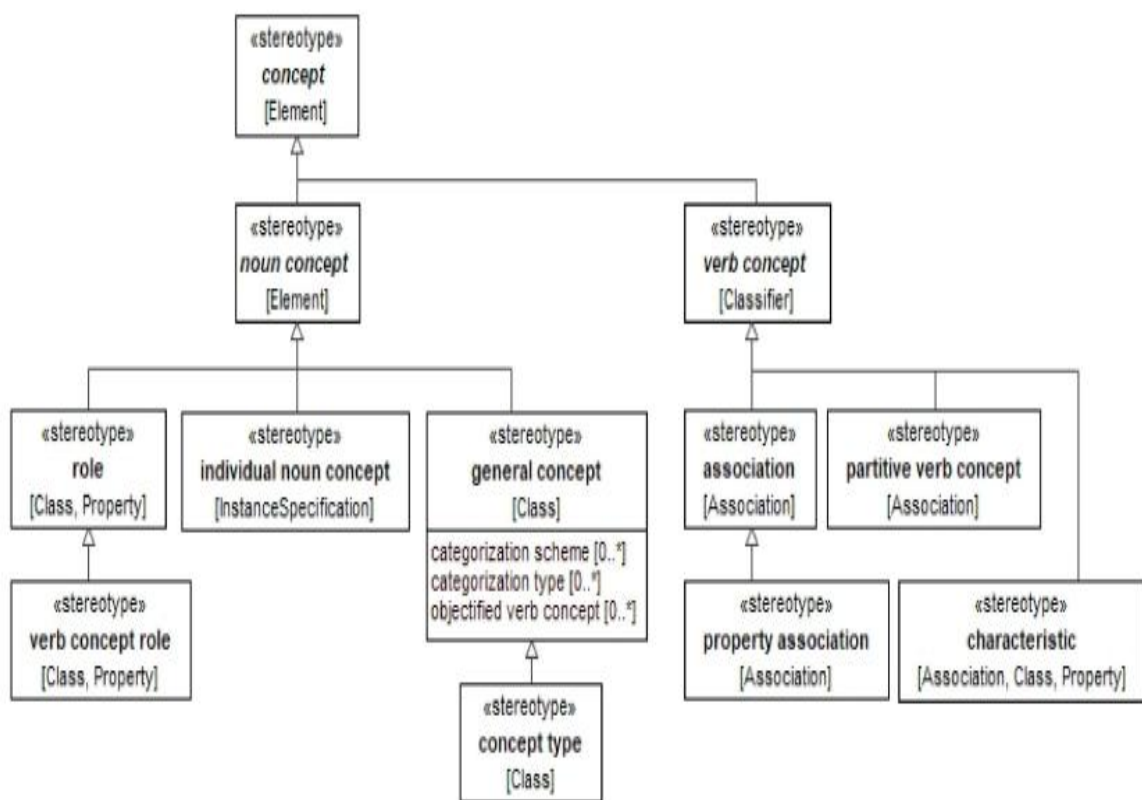
**Figure 3.** Semiotic/Semantic Triangle in SBVR Terms Triangle (OMG, 2012)

The Semiotic/Semantic Triangle is a graphical representation that describes the relationship between three main concepts. These concepts are:

- Meaning: What a word, sign or statement means (OMG, 2012).
- Thing: An object. Anything perceptible or conceivable (OMG, 2012).
- Expression: Something that is expressed or communicated but is considered independent of its interpretation (OMG, 2012).

The Semiotic/Semantic Triangle helps distinguish between business vocabulary, business meanings, and real-world objects. For example, the term "customer" may refer to an individual or organization that consumes a service. These concepts may refer to real-world entities involved in purchasing transactions.

SBVR has two main elements. The first, Business vocabulary, is the terminology used to represent pronouns and verbs that will be used to describe a certain domain. The other element, Business Rules, defines the constraints and behaviours of an organization's business model. **Figure 4** shows the SBVR Business Vocabulary represented in a UML diagram.

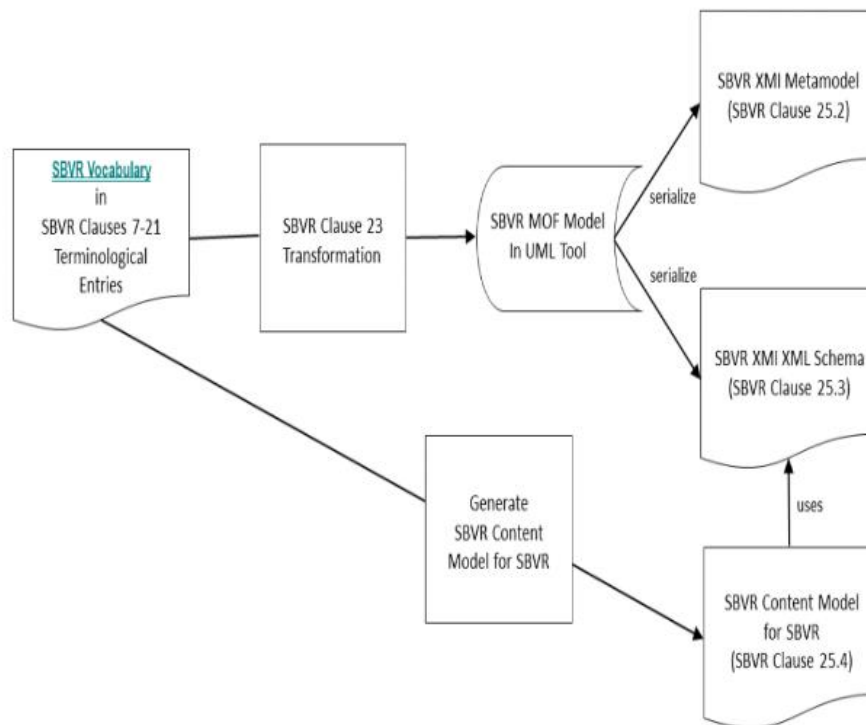


**Figure 4.** UML Representation of the SBVR Business Vocabulary (OMG, 2012)

The SBVR standard defines additional specifications for each concept to create the terminological dictionary, adding a more comprehensive meaning to the business domain (Haj, Jarrar, Balouki, & Gadir, 2021).

SBVR rules are formulated semantically by using logical formulations. Semantic formulations are used to control statements in English such as atomic formulations, logical operations, quantifications (Ramzan et al., 2014)

All machine-readable SBVR files are generated from natural language text in the SBVR terminological entries in the SBVR vocabulary. The result of the transformation is the MOF (Meta-Object Facility) meta-model. The SBVR MOF meta-model does not include definitions, rules, notes, examples or semantic formulations. Instead, it mirrors the SBVR namespaces for the SBVR vocabulary. **Figure 5** shows this transformation process.



**Figure 5.** Transformation Process (OMG, 2012)

SBVR is a complex pattern containing hundreds of elements representing aspects of a domain. Because of this, research dealing with transformations from SBVR to other models or vice versa tends to only deal with a subset of essential SBVR elements.

## METHODOLOGY

In this work, the methodology of systematic literature review (SLR) proposed by Kitchenham (Kitchenham, 2004) was used and a search was carried out in the repositories: IEEE, Science Direct, AIS, Scopus and Springer.

The SLR was based on the following research questions: What do the works currently available propose to transform models with SBVR? What are the pathways to improvement that these jobs provide? What are the tools and techniques available to carry out M2M transformations of the SBVR and what are the main deficits?

The following search string was used to perform searches in the repositories: (“Semantics of business vocabulary and rules OR SBVR) AND (M2M Transformation OR Model-to-Model Transformation OR Model Transformation OR Transformation”). The following inclusion, exclusion and quality criteria were applied:

- Inclusion: Articles published in the last ten years; Articles that address the theme “model-to-model transformation with SBVR”.

- Exclusion: Articles without SVBR and Model Transformation or Transformation among the keywords.

- Quality: Do the articles clearly and objectively present the solution for transforming the SBVR model? Does the solution presented have any gaps that allow for improvement and development? In the end, was the objective of the article achieved in a complete or partial way?

To search for the most up-to-date articles in the area, the inclusion criterion used was to search for articles published in the last 10 years on the subject. This criterion was used in all repositories. For the exclusion criteria, articles that deal only with the SBVR standard or only with M2M transformations were left out, since the objective is to analyze research that deals with both topics. For the quality criteria, it was analyzed whether the research presented its solution concisely, whether the solution presented any gaps that could be explored in future work and whether the research objective was fully or partially achieved.

The initial objective of this work was to carry out an SLR with only the transformation of models from SBVR to UML, but the search returned few results, perhaps due to the possibility that SBVR is not a widespread

standard, the research that deals with this area are few, showing the need for expand the search to work related to transformation to other models. Therefore, the scope of research for transforming models using SBVR has been expanded, regardless of which model is converted to. When performing the search and applying the inclusion, exclusion and quality criteria, 17 articles distributed in the following repositories were returned: 9 from IEEE, 2 from Science Direct, 1 from AIS, 4 from Scopus and 1 from Springer.

## SLR RESULTS

It is noted that the main contributions in this area in the last 10 years were from Skersys et al. The first article (Skersys, Danenas, & Butleris, 2014) explores the possibilities of extracting vocabularies and structured rules from specifications expressed through UML use case diagrams, discussing how to choose the level of automation. In this article, the semi-automatic approach is chosen, as the way of mapping the concepts and the algorithm for carrying out the extraction. Skersys et al. (2014) conclude that the possibility of extracting SBVR rules and vocabularies from use case diagrams provides some benefits such as quick definition of well-structured business rules and vocabularies and higher quality of the business model. Skersys et al. leave open solutions for future work such as two-way transformation between use case diagrams and SBVR and the development of a fully automatic approach involving use case diagrams and SBVR.

Continuing the work launched in 2014, Skersys et al. (2014) show how M2M transformation technologies were used to extract business rules and vocabularies from UML use case diagrams. Skersys et al. focus on two approaches: an automatic and a semi-automatic one. In both approaches, the implementation of a tool that aggregates a CASE system and four other types of systems is carried out entirely or partially by the authors (Skersys et al., 2018). The result showed that automatic approaches are more sensitive to errors, since inconsistently named components result in a higher proportion of errors at the time of transformation. The semi-automatic approach is less sensitive to errors, but the time to perform the transformation is longer. The authors cite the need for integration of more advanced language processing techniques to solve these problems.

Skersys, Danenas, and Butleris (2019) focus on the implementation aspects of the solution presented in the previous article (Skersys et al., 2018). The authors present the rules for performing model transformations, how to perform the extraction and explain in detail the tool that was developed for the article. The article does not present anything different from that presented in the solution of the previous article but details the implementation of the tool developed by the authors.

The next solution presented by Danenas, Skersys, and Butleris (2020) aims to address the need for more advanced language processing techniques to deal with the problems of automatic transformation, cited in the 2018 article (Skersys et al., 2018). In the article, the authors present a natural language processing component to improve solutions presented in previous articles. As a result, the solutions presented for extracting vocabularies and business rules from UML use case diagrams begin to recognize entities, verbs, pronouns in a better way than the previous ones. The authors develop algorithms for the different steps of natural language extraction using solutions implemented by the authors themselves and some external solutions. By validating the algorithms, the authors concluded that the use of improved natural language processing techniques provides certain benefits in addition to those already mentioned, such as accurate recognition of pronouns and other components. The solution also features greater accuracy in classifying entities.

In the most recent article published by Skersys, Danenas, Butleris, Ostreika, and Ceponis (2021), the authors expand the solution to encompass UML model transformations to SBVR with M2M techniques based on drag-and-drop (DnD) actions. The solution presented showed the same transformation power as the previous solutions, but with more flexibility for the user. The authors argue that one of the main characteristics of the DnD transformation technology is the model-oriented development and the complete customization of M2M transformations, guaranteeing extensibility, flexibility and usability to the user.

Another work that addresses SBVR-UML model transformations is that of Bonais et al. (2014), which seeks to identify a subset of SBVR components that express the main characteristics of a structural model and provides a framework to denote this subset and thus convert to the UML class diagram. The solution proves useful to improve the quality and robustness of computerized systems. Still on the SBVR-UML model transformation, another work that stands out is that of Iqbal and Bajwa (2016), in which the authors present the main challenges of model-to-model transformation from SBVR to UML and a method to carry out this transformation. This method consists of extracting the SBVR vocabulary, performing the semantic analysis of the specifications, mapping the terms and generating the UML activity diagram.



Ramzan et al. (2014) explore the possibility of model transformations using SBVR. They present a solution to transform models written in natural language to SBVR. The presented solution uses the transfer tool.

Bulbun and Shahzada (2016) use SBVR only to validate a traceability solution for transformations to the BPMN model. The authors use the SBVR to BPMN transformation as a case study. Although the article does not focus primarily on model transformation with SBVR, it offers some insights into the transformation of models from SBVR to BPMN. However, the future work proposed by the article does not correlate with model transformations with SBVR, but rather with the traceability of model transformations, regardless of the models used. Mickeviciute, Butleris, Gudas, and Karciauskas (2017) present a solution that goes in the opposite direction to Bulbun and Muhammad. Mickeviciute et al developed a plug-in for the MagicDraw CASE tool to enable the transformation to SBVR of business processes represented in BPMN 2.0. Abidin, Manaf, Moschoyiannis, and Jamaludin (2021) use SBVR to describe service choreographies and then transform the SBVR model to an Alloy model, thus being able to automatically generate and validate service choreographies. To perform the transformation, a tool called Alloy Analyzer is used, and a finite state machine is used to ensure that the transformation was performed correctly. The article leaves open the possibility of developing a tool that allows users to develop the SBVR model. Manaf, Antoniadis, and Moschoyiannis (2017) also address the transformation from SBVR to Alloy with the development of the SBVR2Alloy tool, which is proposed by the author to automatically generate and validate service choreographies. The work presented by Essebaa and Chantit (2016) does not focus on SBVR, emphasizing model transformations between the CIM (Computation Independent Model) and PIM (Platform Independent Model) layers of the MDA approach. SBVR is used in the work due to the need to address the business rules of a system, which are transformed into Object Constraint Language (OCL). The authors use the SBVR2OCL tool to perform this transformation. Although the work does not focus on model transformation with SBVR, it shows a path that can be extended with the transformation to the OCL model, and the total automation of model transformations from the CIM layer to the PIM layer.

Haj et al. (2021) develop an approach to automatically transform business rules written in natural language into an SBVR model. To do this, a natural language processor is used to extract an SBVR model that is understandable and includes the semantic formulations of each business rule together with the terminology dictionary of the extracted concepts.

Selway, Grossmann, Mayer, and Stumptner (2015) also discuss transforming natural language into an SBVR model, using MDE techniques, cognitive language and knowledge-based configurations. The authors seek to propose a method to perform semantic processing of a document and transform it into an SBVR model for rapid integration with MDE tools.

Roychoudhury, Sunkle, Kholkar, and Kulkarni (2017) also propose a semi-automatic approach to transform English specifications into SBVR through rules in SE (Structure English). Tangkawarow, Sarno, and Siahaan (2022) propose the ID2SBVR tool, whose objective is to extract SBVR operational rules from informal documents.

Skersys et al. have a good amount of work in the area. Their research deals with the extraction of SBVR rules and vocabularies from UML use case diagrams with a focus on semi-automatic and automatic approaches. Bonais et al. and Iqbal and Bajwa follow the opposite of Skersys et al. by transforming SBVR models into UML diagrams, in the case of Bonais et al. into the class diagram and in the case of Iqbal and Bajwa into the activity diagram. However, research in the area does not only involve the transformation of SBVR models into UML or vice-versa. Research such as that of Ramzan et al., Haj and Jarrar, Selway et al., Roychoudhury et al. and Tangkawarow et al. seek to transform informal texts or documents, written in natural language, into an SBVR model. In the works found, SBVR-UML and SBVR-NL transformations were the predominant topics, but there are articles that address the SBVR transformation to other models, such as the article by Bulbun and Shahzada, addressing the transformation of SBVR to BPMN. Najihah et al. transform an SBVR model to Alloy to validate service choreography. Essebaa and Chantit transform an SBVR model to OCL, but the focus is on the transformation of the CIM layer to the PIM layer. This literature review showed that there are several possibilities to be explored for transforming models that involve the SBVR standard.

## CONCLUSION

The area of model transformations using SBVR is an area that still needs to be further explored, given the scarcity of work and research in this domain. Due to the constant changes that software is submitted, SBVR is a tool that can be essential for better communication between customers and software engineers in the development and implementation of systems.

The automatic transformation of SBVR to other models, in addition to facilitating this communication, can guarantee the robustness and consistency of software solutions and reduce development time and increase productivity. There are works both to extract SBVR vocabularies and rules from models, and to transform SBVR rules and vocabularies to other models. The main approaches discussed in the works deal with the transformation of textual specifications to SBVR and SBVR to UML and vice versa. There are also solutions aimed at transforming SBVR to BPMN, OCL and Alloy, but the scarcity of research in the area illustrates the need to carry out more research in this domain, not only for the aforementioned models, but also in other modelling languages. No works were found that deal with the transformation for models such as Object-Role Modelling (ORM), Specification and Description Language (SDL), Architecture Description Language (ADL), among others. In the surveys found, it is noted that a good part of the work is dedicated to dealing with the transformation from SBVR to UML or vice versa. There are some works that deal with transformation from natural language to SBVR, however, the interest is the transformation of models from the CIM layer to PIM from the MDA approach, using SBVR to automatically generate a model of the PIM layer. This literature review found a good number of articles dealing with SBVR transformation for PIM layer models like UML and BPMN. Therefore, researching SBVR transformations for UML and BPMN is a promising area, with techniques and tools that can be improved or expanded. Thus, as future work, it is planned to propose a tool for automatic transformation of the SBVR business rule representation model into diagrams of modelling languages, UML and BPMN.

### **CONFLICT OF INTEREST**

There are no conflicts of interest.

## REFERENCES

- Abidin, N. N. Z., Manaf, N. A., Moschoyiannis, S., & Jamaludin, N. A. (2021, January). Deontic rule of rule-based service choreographies. In *2021 2nd International Conference on Computing and Data Science (CDS)* (pp. 510-515). Piscataway, NJ: IEEE.
- Bonais, M., Nguyen, K., Pardede, E., & Rahayu, W. (2014). A formalized transformation process for generating design models from business rules. *ACIS 2014 Proceedings*, 21.
- Bulbun, G. U. D., & Shahzada, H. M. A. (2016, August). BPMN process model checking using traceability. In *2016 Sixth International Conference on Innovative Computing Technology (INTECH)* (pp. 694-699). Piscataway, NJ: IEEE.
- Danenas, P., Skersys, T., & Butleris, R. (2020). Natural language processing-enhanced extraction of SBVR business vocabularies and business rules from UML use case diagrams. *Data & Knowledge Engineering*, 128, 101822.
- Essebaa, I., & Chantit, S. (2016, October). Toward an automatic approach to get PIM level from CIM level using QVT rules. In *2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA)* (pp. 1-6). Piscataway, NJ: IEEE.
- Haj, A., Jarrar, A., Balouki, Y., & Gadir, T. (2021). The semantic of business vocabulary and business rules: An automatic generation from textual statements. *IEEE Access*, 9, 56506-56522.
- Iqbal, U., & Bajwa, I. S. (2016, August). Generating UML activity diagram from SBVR rules. In *2016 Sixth International Conference on Innovative Computing Technology (INTECH)* (pp. 216-219). Piscataway, NJ: IEEE.
- Kitchenham, B. (2004). *Procedures for performing systematic reviews* (Technical Report TR/SE-0401). Retrieved from Keele University, Software Engineering Group website: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=29890a936639862f45cb9a987dd599dce9759bf5>
- Manaf, N. A., Antoniadis, A., & Moschoyiannis, S. (2017, November). SBVR2Alloy: An SBVR to alloy compiler. In *2017 IEEE 10th Conference on Service-Oriented Computing and Applications (SOCA)* (pp. 73-80). Piscataway, NJ: IEEE.
- Mickevičiute, E., Butleris, R., Gudas, S., & Karčiauskas, E. (2017). Transforming BPMN 2.0 business process model into SBVR business vocabulary and rules. *Information Technology and Control*, 46(3), 360-371.
- OMG. (2012). *Semantics of Business Vocabulary and Business Rules (SBVR) Version 1.1*. Retrieved from <https://www.omg.org/spec/SBVR/1.1/PDF>
- OMG. (2014). Model-Driven Architecture (MDA) Guide rev. 2.0. Retrieved from <https://www.omg.org/cgi-bin/doc?ormsc/14-06-01>
- Ramzan, S., Bajwa, I. S., Haq, I. U., & Naeem, M. A. (2014, September). A model transformation from NL to SBVR. In *Ninth International Conference on Digital Information Management (ICDIM 2014)* (pp. 220-225). Piscataway, NJ: IEEE.
- Roychoudhury, S., Sunkle, S., Kholkar, D., & Kulkarni, V. (2017, October). From natural language to SBVR model authoring using structured English for compliance checking. In *2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC)* (pp. 73-78). Piscataway, NJ: IEEE.
- Selway, M., Grossmann, G., Mayer, W., & Stumptner, M. (2015). Formalising natural language specifications using a cognitive linguistic/configuration based approach. *Information Systems*, 54, 191-208.
- Skersys, T., Danenas, P., & Butleris, R. (2014, May). Approach for semi-automatic extraction of business vocabularies and rules from use case diagrams. In *Enterprise Engineering Working Conference* (pp. 182-196). Cham, Switzerland: Springer.
- Skersys, T., Danenas, P., & Butleris, R. (2018). Extracting SBVR business vocabularies and business rules from UML use case diagrams. *Journal of Systems and Software*, 141, 111-130.
- Skersys, T., Danenas, P., & Butleris, R. (2019, July). Wizard-guided extraction of SBVR business vocabularies and rules from UML use case models: Practical implementation aspect. In *AIP Conference Proceedings* (Vol. 2116, No. 1). <https://doi.org/10.1063/1.5114359>
- Skersys, T., Danenas, P., Butleris, R., Ostreika, A., & Ceponis, J. (2021). Extracting SBVR business vocabularies from UML use case models using M2M transformations based on drag-and-drop actions. *Applied Sciences*, 11(14),

6464.

Tangkawarow, I., Sarno, R., & Siahaan, D. (2022). ID2SBVR: A method for extracting business vocabulary and rules from an informal document. *Big Data and Cognitive Computing*, 6(4), 119.