# A Model Transformation from NL to SBVR

Shabana Ramzan[1], Imran Sarwar Bajwa[1], Ikram Ul Haq[2], M. Asif Naeem[3]

[1] Department Of Computer Sciences & IT
The Islamia University of Bahawalpur, Pakistan

[2] School of Information Technology,
Deakin University, Australia

3 School of Computing and Mathematical Sciences
Auckland University of Technology, New Zealand

shabanaramzan@hotmail.com, imran.sarwar@iub.edu.pk, iulhaq@deakin.edu.au, mnaeem@aut.ac.nz

***Abstract*. In Requirement Engineering, requirements are usually written in sentences of natural language and natural languages are ambiguous and inconsistent, so the requirements written in natural languages also tend to be ambiguous. To avoid this problem of ambiguity we present an approach of model transformation to generate requirements based on SBVR (Semantics of Business Vocabulary and Business Rules). The information provided in source metamodel (NL) is automatically transformed into target metamodel (SBVR). SBVR metamodel can not only be processed by machine but also provides precise and reliable model for software design. The standard SBVR metamodel is already available but for natural language we proposed our own metamodel because there is no standard metamodel available for natural languages.**

***Keywords*: SBVR, Requirement specification, Model Transformation, Transformation Rules**

## I. INTRODUCTION

Software Development is based on the Requirement specification document provided by system analyst. Requirements specify in requirement specification document are usually expressed in natural language text, natural language text have multiple meanings so requirements also become ambiguous. As a result the software designed by using these requirements will not provide results according to the need of user. Ambiguity is also induced when requirements translating to code, so there is always demand of exact and precise requirements by development team.40-60% of software failure is due to the faults occurred at the requirements phase (Mitamura, 1995). To overcome the problems of ambiguities and inconsistencies, we presented an approach to transform NL based requirements into the Semantics of Business Vocabulary and Business Rules (SBVR) [1] standard of OMG. The presented approach transforms NL metamodel into SBVR metamodel. Requirements based on SBVR resolve the issue of ambiguity and provides machine processable requirements because of its formal logic. SBVR is highly beneficial to generate requirements and our study will diminish the gap between user/client and system analyst. SBVR based requirements lessen the semantics changes and reduce disasters in software industry caused by wrong communication between user and development team.

The remaining paper has following sections, section II presents related work, section III describes basic concepts involved in model transformation; section IV states the model transformation from NL to SBVR, section V illustrates the case study and followed by experiments and results section. The last section covers conclusions.

## II. RELATED WORK

Natural languages requirements are processed by different tools to control ambiguity problems of NL. CM builder is a case tool proposed by Harmain to create conceptual model from NL requirements using UML [4]. LOLITA proposed a case tool NL-OOPS which gives object model to improve the process of software development [3]. Different solutions are presented [10], [11], [12], [13], [14] to generate UML models from natural language requirements

In last few decades various controlled natural languages are used instead of natural language to communicate requirements, collected by system analyst for development team. Controlled natural languages are of two types, human oriented [8] and machine oriented [9]. The human oriented controlled natural language is PENG (Processable English) [5], ACE (Attempto Controlled English) [6], CPL (Computer Processable Language) [7], etc. Brillant used SBVR to represent natural language requirements into models that can be executed [15]. Umber presented a SR-elicitor tool to generate ambiguity less requirement document by using SBVR [16]. Firstly this tool analyzed NL text lexically, semantically and syntactically and then extract SBVR elements to generate SBVR based rules.

Bajwa presented SBVR2OCL prototype tool to automatically generate OCL constraints [17]. This approach facilitates the process of software development. Firstly transform NL text into SBVR and then generate OCL constraints from SBVR. For SBVR to OCL constraints transformation involved different steps, firstly objects

978-1-4799-5421-6/14/$31.00 © 2014 IEEE

oriented information is extracted from SBVR rule, and then generate OCL expression from this extracted information and finally mapped OCL syntax and semantics. Raj presented transformation technique to generate different UML diagrams from SBVR rules and vocabulary [18]. In the domain of model transformation presented work [19], [20], [21],[22] to transform different models into SBVR and as a reverse engineering generate SBVR model from other models.

The related work shows that the approach of model transformation has already been used to transform various models into other models, but no one tried to transform NL metamodel into SBVR metamodel. Our research presented the transformation of NL metamodel to SBVR metamodel.

## **III**. BASIC CONCEPTS

### *A. Semantics of Business vocabulary and Rules*

Semantics of business vocabulary and rules (SBVR) [1] is a standard established by object management group. If we transform natural language requirement specification into SBVR based requirements, we have an opportunity to easily process these requirements by machine due to formal logic of SBVR. Semantic models for business rules and vocabulary are developed by metamodel, which is defined by SBVR [1]. SBVR has two main constituents, SBVR rules and vocabulary. The detail description of SBVR constituents is given below.

*1) SBVR business vocabulary:* The main elements of SBVR vocabulary are concepts and fact types. Business people used SBVR vocabulary for their official writing. Concepts are of three types, Object Types and Individual Concepts and Verb Concepts. Common nouns of natural language text are represented as object types or general concepts and proper noun as individual concept. All helping verb and action verb are classified as verb concept. A proposition, a verb or a combination of verb and preposition is denoted as fact type [1].

*2) SBVR Business Rules:* These rules provide guidance about the actions taken for business and also define the structure of business. There are two types of rules

- *Definitional rules or structural rules:* The setup of organization is defined by these rules [1] e.g. **It is possible that each** customer can order for **more than one** meal per day.
- *Behavioural rules or operative rules:* The behaviour of an entity is characterized by these rules [1] e.g. **It is obligatory that each** truck can transport the tools to factory.

*3) Semantic Formulation of NL Text:* SBVR rules are semantically formulated by using logical formulations. A set of logical formulations are defined in SBVR document [1]. The semantic formulations are used to control English statements such as atomic Formulation, instantiation Formulation, logical Operations, quantifications, and modal formulation.

*4) SBVR Based Notation for NL Text:* In order to formalize natural language text according to some standard, one of the possible notations is struct`ure English, in annex C of SBVR 1.0 document [1]. We have used the following formatting rules of SBVR Structured English.

- Double underlined all individual concepts e.g. gold loan customer
- Underlined all noun concepts e.g. student
- SBVR keywords are bolded e.g. at least, it is possible, it is obligatory, some, each, etc.
- All verbs are italicized e.g. has , should be

In this paper, we have italicized the adjectives and possessive nouns like verb concepts but used different color to represent them.

### B. Model Transformation

The components required for model transformation from NL metamodel to SBVR metamodel are source metamodel (NL), transformation engine, transformation rule, transformation description. We used Sitra as a transformation engine. Transformation engine transform the source metamodel to target metamodel by using transformation rules. The complete sketch of model transformation is depicted in Figure 1.

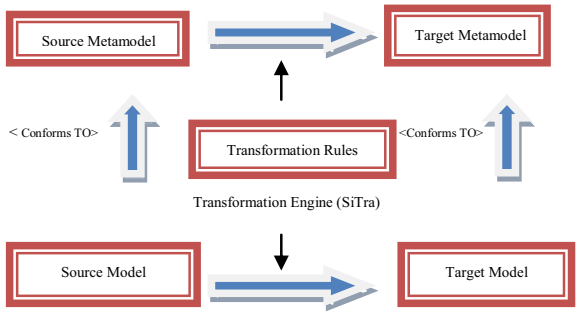


Figure 1: Model Transformation from NL to SBVR

MDA (Model Driven Architecture) is model based approach to develop software. Models and model transformation is our main concern. The MDA approach maps elements of source metamodel into related elements of target metamodel. This mapping is used to generate transformation rules. Transformation engine execute these rules to automatically generate target metamodel from source metamodel. . Model to model, model to text and text to model, are various types of model transformation. Model to model transformation transforms a model into another model. In our study, we employed the model to model transformation.

### C. Natural Language Metamodel

There is no standard metamodel for natural languages such as English. ProjectIT-RSL metamodel is natural language based metamodel proposed by Videria and Silva (2005). But this metamodel has some deficiencies, all concepts of natural language (such as English) are not described e.g. actor,

operation and entity are commonly used semantic role labels but not represented in the metamodel. However, in our research, a complete metamodel of English is required to transform natural language (such as English) to SBVR. We have complemented the ProjectIT-RSL natural language based metamodel to model transform natural language to SBVR. Figure 2 shows the newly added elements (orange colour) in ProjectIT-RSL metamodel and the metamodel is called English metamodel.

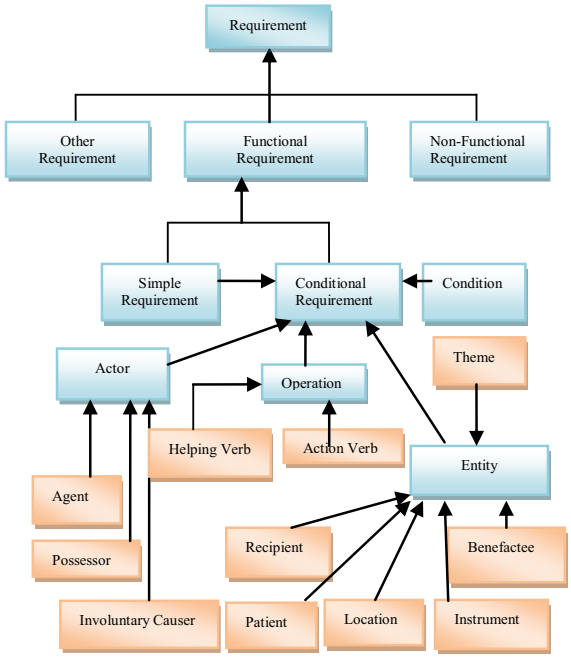


Figure 2: Proposed NL Metamodel

## IV. Transformation from NL TO SBVR

To translate natural language into SBVR, the approach will take NL requirements as input and then these requirements are processed to obtain SBVR elements. Requirements are linguistically analyzed in three steps, firstly requirements are POS (parts- of –speech) tagged. Then parser is used to obtain basic SBVR elements (such as individual concept, verb concepts, noun concepts, objects type, etc. For SBVR metamodel, elements of NL are mapped into elements of SBVR.

### *A. Processing Natural Language Specification*

*1) Lexical Analysis:* Various tasks are performed to accomplished lexical analysis In first step, the natural language text is read and tokenized to produce token. [2].These tokens are further processed by Stanford parts of speech (POS) [2] tagger v3.0 to identify POS tags.

*2. Syntactic Analysis:* The structure of text is determined by syntactic analysis.The text is syntactically analysed to generate parse. Figure 3 shows the parse tree of above example.

```
(ROOT
  (S
    (NP (NN Customer))
    (VP (MD should)
      (VP (VB have)
        (S
          (NP (NN business) (NN
account))
          (VP (TO to)
            (VP (VB get)
              (NP (NN credit) (NN
card)))))))
    (. .)))
```

Figure 3: Parse Tree Of NL Text

*3. Semantic Analysis:* Semantic analysis is performed to find out the meanings of sequence of words by uniting the meanings of all words. Semantic analysis provides appropriate text representation of parse tree. Semantic role labeling is carried out in this phase [14]. The agent, involuntary causer, theme, patient, and benefactee are desired semantic roles. SBVR elements are extracted with the help of these roles and forming fact types from the SBVR elements. SBVR elements (verb concepts, Noun concept, object type, individual concept, etc) are identified from the natural language (such as English) text.

### *B. NL to SBVR Mapping*

*1) Mapping SBVR Elements:* For NL to SBVR mapping, elements of natural language are mapped to corresponding SBVR elements. Table I shows the mapping of NL elements to SBVR elements.

Table I: Mapping Between NL and SBVR Metamodel Element

| NL Metamodel Elements | SBVR Metamodel Elements |
|---|---|
| **Nouns e.g., Agent \| Involuntary causer \| Theme \| Patient \| Benefactee** | Object Type |
| **Proper nouns e.g., Agent \| Involuntary Causer\| theme \| Benefactee \| Patient** | Individual Concept |
| **Helping Verb +Action Verb \| Action Verb** | Verb Concept |
| **Object Type\| Individual Concept + verb concept** | Unary fact Type |
| **Object Type\| Individual Concept + verb concept+ Object Type** | Binary Fact Type/Associative Fact Type |
| **Adjective Noun and Possessive Noun** | Characteristics |
| **Enumeration of Verb Concept or Noun Concept** | Quantification |
| **structures such as "is-part-of", "included-in" or "belong-to"** | Partitive Fact Type |
| **structures such as "is-category-of" or "is-type- of", "is-kind-of"** | Categorization Fact Type |

*2) Mapping SBVR Syntax:* Information required for the transformation of NL text to SBVR representation is

extracted from syntax rules. Table II shows SBVR syntax model,

Table II: SBVR Syntax Model

| Syntax Rule | | Syntactic Elements |
|---|---|---|
| SBVR rule | → | Modal Formulation + Fact Type |
| Modal Formulation | → | Necessity Formulation/Obligation Formulation/Permissibility Formulation/Possibility Formulation |
| Fact Type | → | Subject + Verb Concept + Object |
| Subject | → | Noun Concept |
| Object | → | Noun concept/Object Type + Characteristics |
| Verb Concept | → | Verb/Helping Verb + Action Verb |
| Noun Concept | → | Object Type/Individual Concept |
| Object Type | → | Object Type/Object Type + Logical Operators + Object Type / Quantification + Object Type |
| Individual Concept | → | Individual Concept/Individual Concept + Logical Operators + Individual Concept / Quantification + Individual Concept |

*3) Mapping SBVR Semantics:* In SBVR version 1.0, there are five types of logical formulations but our approach used three logical formulations, quantification, logical operations and model operations.

Table III: SBVR Quantification

| Logical Formulations | SBVR |
|---|---|
| at least one | existential quantification |
| exactly one | exactly-one quantification |
| each | universal quantification |
| at most one | at-most-one quantification |
| more than one | at-least-n quantification with $n = 2$ |
| at least $n$ | at-least-n quantification |
| exactly $n$ | exactly-n quantification |
| at most $n$ | at-most-n quantification |
| some | existential quantification |

Table IV: SBVR Logical Operations

| Logical Formulations | SBVR |
|---|---|
| it is not the case that $p$ | logical negation |
| if $p$ then $q$ | implication |
| $q$ if $p$ | implication |
| $p$ or $q$ | disjunction |
| $p$ if and only if $q$ | equivalence |
| $p$ and $q$ | conjunction |

Table V: SBVR Modal Operation

| Logical Formulations | SBVR |
|---|---|
| always | necessity formulation |
| must | obligation formulation |
| never | necessity formulation embedding a logical negation |
| must not | obligation formulation embedding a logical negation |
| may | permissibility formulation |

*C. Generating Transformation Rules*

The transformation of the source model into the target model is described by the transformation rules. There are two parts for each transformation rule, right hand side (RHS) for target pattern and left hand side (LHS) side for source pattern.

Rule 1 [SBVR-Rule (modal-formulation, fact-Type)]
= modal-formulation + fact-type

Rule 1.1 [Modal-Formulation (modal-formulation]
= modal-formulation

Rule 1.1.1 [Modal-Formulation (necessity-formulation)]
= "It is necessary that"

Rule 1.1.2 [Modal-Formulation (obligatory-formulation)]
= "It is obligatory that"

Rule1.1.3 [Modal-Formulation (permissibility-formulation)]
= "It is permissible that"

Rule 1.1.4 [Modal-Formulation (possibility-formulation)]
= "It is possible that"

Rule 2 Fact-Type (subject, verb-concept, Object)
= subject + verb-Concept + Object

Rule 2.1 Subject-part (subject)
= subject

Rule 2.1.1 Subject (noun-concept)
= noun concept

Rule 2.2 Verb-Concept (verb-concept)
= verb-concept

Rule 2.2.1 Verb-Concept (action-verb)
=action-verb

Rule 2.2.2 Verb-Concept (helping-verb, action-verb)
= helping-verb + action-verb

Rule 2.3 Object-part (object)
=object

Rule 2.3.1 Object (noun-concept)
=noun-concept

Rule 2.3.2 Object (Object–Type, Characteristics)
= object–type + characteristics

Rule 3 Noun-Concept (noun-concept)
= noun-concept

Rule 3.1Noun-Concept (object-type)
= object-type

Rule 3.1.1 Object-Type (object-type)
= object-type

Rule 3.1.2 Object-Type (object-type, logical-operators, Object-type)
= object-type + logical-operators + object-type

Rule 3.1.3 Object-Type (quantification, object type)
= quantification + object type

Rule 3.2 Noun-Concept (individual-concept)
= individual-concept

Rule 3.2.1 Individual-Concept (individual-concept)
= individual-concept

Rule 3.2.2 Individual-Concept (individual-concept, logical-operators, individual-concept)
=individual-concept + logical-operator + individual- concept

Rule 3.2.3 Individual-Concept (quantification, individual-concept)
= quantification + individual-concept

## V. CASE STUDY

We illustrate a case study of KeePass Password Safe[23] to show the performance of our approach in terms of accuracy and fulfillment of user need.. The problem statement of case study is

*KeePass consists of a database which contains data for one or more users. Each user's data are divided into groups and subgroups so that they are organized in a form that serves right the user. Every user has a unique Master Key which can be simple or composite and its combination opens uniquely the database. If lost there is no recovery. Groups and subgroups contain entries with usernames, passwords URLs etc that can be sent or copied to websites, application and accounts. There is also the ability for a onetime key creation to be used once in a transaction without the risk of reused by others for any reason.*

NL (English) text of the problem statement of case study is parsed lexically, semantically and syntactically to extract SBVR vocabulary.

Table VI: SBVR Vocabulary Extracted from NL Text

| Category | Count | Details |
|---|---|---|
| Object Types | 15 | keypass, user, database, data, masterkey, simple, composite, groups, subgroups, applications, webpage, accounts, onetimekey, form, transaction |
| Individual Concept | 00 | |
| Verb Concept | 09 | consists, contains, divided, organized, serves, sent, copied, reused, opens |
| Unary Fact Type | 02 | opens database, transactions reused |
| Associative Fact Types | 06 | database for user, form serves user, user has Masterkey, its open database, Group and Subgroup send or copy entries to websites, applications, and accounts, OneTimeKey used in a transaction |
| Characteristics | 03 | user name, password, url |
| Quantification | 02 | One, more |
| Categorization Fact Types | 02 | data is divided into groups and subgroups, MasterKey can be simple or composite base contains data |
| ` | 02 | KeyPass consists database, database contains data |

There are four requirements for the problem statement of the used case study, as shown in table VII:

Table VII: SBVR Based Software Requirements

| Category | Count | Details |
|---|---|---|
| Software Requirements | 04 | It is necessary that each user's data are *divided* into groups and subgroups so that they are *organized* in a form that *serves* right the user.<br><br>It is obligatory that every user *has* a unique Master Key which *can* be simple or composite and its combination *opens* uniquely the database. If lost there *is* no recovery.<br><br>It is necessary that Groups and subgroups *contain* entries with *usernames*, *passwords*, *URLs* etc that can be *sent* or *copied* to websites, application and accounts.<br><br>It is possibility that there *is* also the ability for a onetimekey creation to be used once in a transaction without the risk of *reused* by others for any reason. |

According to SBVR structured English underlined the object types e.g. groups, subgroups, accounts etc. italicized the verb concepts e.g. *has, can, contain* etc. the SBVR keywords are as e.g. It is necessary, It is obligatory etc. The purple color used for characteristics which are also italicized e.g. *usernames*, *passwords*, *etc.*

## VI. EXPERIMENTS AND RESULTS

In order to evaluate the performance of our presented approach, we have solved five case studies including the case studies discussed in section V. Table VIII shows the computed average recall, precision and F-value by using the results of all these case studies.

Average recall for these case studies is 83.22% and average precision is 87.13%, calculated average F-value is 85.14 % that is very supportive for future development. For all solved

case studies, the figure 4 shows the Recall, precision and F-Value.

| Input | $N_{test}$ | $N_{approved}$ | $N_{wrong}$ | $N_{omitted}$ | Rec % | Prec % | F-Value |
|---|---|---|---|---|---|---|---|
| **C1** | 63 | 51 | 09 | 03 | 80.95 | 85.00 | 82.93 |
| **C2** | 48 | 39 | 07 | 02 | 81.25 | 84.78 | 82.98 |
| **C3** | 39 | 30 | 08 | 01 | 76.92 | 78.95 | 77.92 |
| **C4** | 51 | 43 | 06 | 02 | 84.31 | 87.76 | 86.00 |
| **C5** | 50 | 44 | 03 | 03 | 88.00 | 93.61 | 90.72 |
| **C6** | 58 | 51 | 04 | 03 | 87.93 | 92.73 | 90.27 |
| | | | | **Average** | **83.22** | **87.13** | **85.14** |

Table VIII: Evaluation Results for Transformation

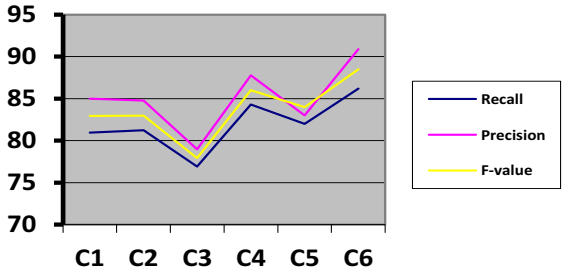


Figure 4: Recall, Precision and F-Value for set of case studies

## VII. CONCLUSIONS

The most important benefit of this research was to gain software requirements specification with minimum ambiguity by generating SBVR based requirement specification using model transformation approach. For model transformation required source metamodel (NL) and target metamodel (SBVR).Target metamodel is standard and available in SBVR 1.0 document.NL metamodel is not available, our research also proposed NL metamodel. This research transform NL metamodel to SBVR metamodel by using Sitra transformation engine. As a next step in future, we plan to perform our experiments on enterprise case studies to check the performance of our approach and evaluate the class of resulting SBVR based requirements specification.

## REFERENCES


[1] OMG. 2008. Semantics of Business vocabulary and Rules (SBVR), OMG Standard, v. 1.0.

[2] Toutanova. K., Manning, C.D. (2000). Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora. 63-70. Hong Kong.

[3] Mich, L. (1996). NL-OOPS: from natural language to object oriented requirements using the natural language processing system LOLITA. Natural Language Engineering. 2(2) p.167-181.

[4] Harmain, H. M., Gaizauskas, R. (2003). CM-Builder: A Natural Language-based CASE Tool For object Oriented Analysis. Journal of Automated Software Engineering, 10(2), 157-181.

[5] White, Colin and Rolf Schwitter. (2009). An Update on PENG Light. In: Proceedings ofALTA 2009, pp. 80–88.

[6] Fuchs, N.E., Kaljurand, K., and Kuhn, T. (2008). Attempto Controlled English for Knowledge Representation. In: Reasoning Web, LNCS, vol. 5224/2008:104–124.

[7] Clark, P., Harrison, P., Murray, W.R, Thompson, J. (2009).Naturalness vs. Predictability: A Key Debate in Controlled Languages. In: Proceedings 2009 Workshop on Controlled Natural Languages (CNL'09).

[8] Schwitter, R. 2010. Controlled Natural Languages for Knowledge Representation, Coling 2010: Poster Volume, Beijing, August 2010: 1113–1121

[9] Huijsen, W.O. 1998. Controlled Language –An introduction. In: Proceedings of CLAW 98:1–15.

[10] Ilieva M., Olga O. (2005) Automatic Transition of Natural Language Software requirements Specification into Formal Presentation. Springer LNCS Vol. 3513, pp.392--397 (2005).

[11] Whittle J., Jayaraman P., et al (2009). : MATA: A Unified Approach for Composing UML Aspect Models on Graph Transformation: Springer LNCS Vol. 5560, p. 191--237 (2009)

[12] Oliveira A., Seco N., Gomes P (2004). : A CBR Approach to Text to Class Diagram Translation, In TCBR Workshop at the 8th European Conference on Case-Based Reasoning. (2004).

[13] Bajwa I., Samad A., Mumtaz S (2009). : Object Oriented Software modeling Using NLP based Knowledge Extraction, European Journal of Scientific Research, 35(01), p.22—33.

[14] Bryant B., et al. 2008. From Natural Language Requirements to Executable Models of Software Components. In Workshop on S. E. for Embedded Systems pp.51-58

[15]Kouamou., G. E., Feuto Njonko, P. B. (2010). Cohérence devues dans la spécification des architectures logicielles.In proceeding of 10th African Conference on Research in Computer Science and Applied Mathematics (CARI'2010), Ivory Coast, October 18–21, 2010.

[16] Umber, A., Bajwa, I. S. (2012). A Step towards Ambiguity less Natural Language Software Requirements Specifications. *IJWA*, *4*(1), 12-21.

17] Bajwa I.S., Behzad Bordbar, Mark G. Lee, (2010), OCL Constraints Generation from NL Text, IEEE International EDOC conference 2010, Vitoria,Brazil,pp.204-213.

[18] Raj A., Prabharkar T., Hendryx S., (2008). Transformation of SBVR Business Design to UML Models., In ACM Conference on India software engineering, pp.29-38.

[19] Malik, S., Bajwa, I. S. (2013). Back to Origin: Transformation of Business Process Models to Business Rules. In *Business Process Management Workshops* (pp. 611-622). Springer Berlin Heidelberg.

[20] Nicolae, O., Wagner, G., (2008), Verbalising R2ML rules into SBVR, 10 International Symposium on Symbolic and Numeric Algorithms for Scientific Computing IEEE OMG: Object Constraint Language (OCL), OMG Standard, v. 2.0, (2006).

[21] Cabot J., et al., (2009), UML/OCL to SBVR specification: A challenging Transformation, Journal of Information systems doi:10.1016/j.is.2008.12.002.

[22]Afreen, H., Bajwa, I.S., Bordbar, B. (2011) SBVR2UML: A Challenging Transformation, Frontiers of Information Technology (FIT), 2011 9th International Conference, pp:33-38

[23] Bajwa, I.S., Bordbar, B., Lee, M.G. (2011) SBVR Business Rules Generation from Natural Language Specification. AAAI Spring Symposium 2011 - AI4BA, pp.541-545, San Francisco, USA.