

# Text Processing Techniques in Approaches for Automated Composition of Domain Models

Viktorija Gribermane<sup>1</sup> <sup>a</sup> and Erika Nazaruka<sup>2</sup> <sup>b</sup>

<sup>1</sup>*Institute of Applied Computer Systems, Riga Technical University, 1 Setas St., Riga, Latvia*

<sup>2</sup>*Department of Applied Computer Science, Riga Technical University, 1 Setas St., Riga, Latvia*

**Keywords:** Natural Text Processing, Knowledge Extraction, Unrestricted Language, Domain Model, Software Model.

**Abstract:** Text processing techniques are critical for automated analysis of domain documentation. Proper domain analysis may include analysis of a huge number of documents that may describe business procedures, policies, organizational structures, regulations, etc., as well as minutes of discussions with domain experts. The results of analysis are also presented as documents with or without supporting graphics, e.g., domain models. Automated composition of domain models (including software models) should decrease the time necessary for analysis and would provide traceability to the original documentation. The goal of this research is to understand how text processing techniques can be applied for composition of those models and what are the current trends in this field. The result of analysis of 15 approaches showed that Natural Language Processing features are just the starting point in document processing. The main difficulty is proper analysis of dependencies among words in sentences and among sentences themselves. The obtained results indicate two directions in identification of patterns of those dependencies and a complete diversity in their applications.


## 1 INTRODUCTION


Domain models are simplified representation of one or many aspects of the real phenomena (e.g., a system) which are created with a certain purpose. Composition of the domain models is a transformation of experts' knowledge about existing phenomena into some human understandable form, e.g., graphical, mathematical, or textual. Usually, this transformation is a mental work supported by guidelines that are aimed to make this work more systematic. However, the development of natural language processing (NLP) technologies opened an opportunity to use them for automated composition of domain models based on textual and graphical-textual descriptions of phenomena.

The automated composition of domain models is based on processing verbal descriptions that origin as a result of discussions with domain experts, tasks performed, or that are represented by existing documents. Besides, some knowledge may remain implicit. Therefore, validation of the composed domain models by the human expert is also required.

There are many research results on text processing as such, but the aim of this research is to find out what techniques are used mostly in the context of system/software domain model composition and their further use. For this purpose, we have selected and analyzed 15 research papers published from 2005 till 2020 and stored in IEEE and ACM publication bases. The main criterion of selection was the use of NLP features for processing textual descriptions of the problem domain structures and functionality that has finished in automated composition of the domain model. We skipped approaches those of using manual processing of text or manual composition of the domain models.

The following sections contain information on domain model characteristics that could be got from the verbal descriptions (section 2) and analysis of research works that uses NLP features at the beginning of the problem domain analysis (section 3). Conclusion summarizes main results obtained and describes validity of the research done.

<sup>a</sup>  <https://orcid.org/0000-0002-8368-9362>

<sup>b</sup>  <https://orcid.org/0000-0002-1731-989X>

## 2 COMPOSITION OF DOMAIN MODELS

The domain most often is understood as an application area or a field for which software is developed (Prieto-Diaz, 1990). Domains can be narrow like a user authentication or broad like a business. A domain analysis is a process of identification of information that can be used for software development with the purpose of making it reusable (Prieto-Diaz, 1990). If this information is available, then it will be faster and more effective for software engineer to decide whether to reuse components in the software system and understand the beginning idea of this designed component.

The goal of the domain analysis in the software development is to understand and specify the functional, behavioral and structural characteristics of a domain AS-IS (the problem domain: today's reality with existing business processes, existing objects, existing functionality, etc.) and/or a domain TO-BE (the solution domain: requirements to software to be built, information system planned, new business processes, modified functionality etc.) as well as conformity among them if the both domains are analyzed.

The structural characteristics of the domain can be modelled as concepts and their intra- and inter-relationships. The behavioral characteristics of the domain are expressed as activities, control flows, data flows and interactions with users or other systems. The functional characteristics of the domain are expressed as causal dependencies among activities or functional parts of the domain, or states of the software systems and transitions between them.

The domain models can be informal, semi-formal and formal. In order to use models as a source for further code generation, the models must be formal (Miller and Mukerji, 2001), i.e., readable by machines. Otherwise, only manual, or partially automated transformation into code is possible.

Composition of the domain model starts from gathering information. Usually, the obtained information is specified in the form of unstructured text (text in a formal writing style) and structured descriptions (user stories, use case specifications, scenarios, and other formats). Generally, descriptions are represented as a large number of documents, which consist of text with figures, tables, and multiple links to other documents. In practice, preparation of text and manual knowledge obtaining are too resource-consuming (Elstermann and Heuser, 2016). Therefore, the common case is to avoid the step of preparation of text or to automate or semi-automate

this process. Certainly, the automated process may require composition of a "transitional" model that can be used for organizing the knowledge gathered from the documentation and its further analysis.

The automated processing of structured text (especially if a language is controlled) is easier than processing of unstructured prose (Nazaruka, 2020). However, structuring the text also requires additional manual processing of information.

## 3 USE OF TEXT PROCESSING TECHNIQUES

Analysis of unstructured (uncontrolled) text requires not only application of basic NLP features such as tokenization, part-of-speech (POS) tagging, chunking and Name Entity Recognition (NER) but also analysis of dependencies between clauses, in complex noun phrases, in predicates and in verb phrases as well as one of the most difficult tasks, namely, discourse analysis (Nazaruka and Osis, 2018). The process is getting complicated with particularities of a natural language (Nazaruka et al., 2019). Thus, quality of the processing depends on the presence of needed knowledge, different structures of sentences and implicit synonyms.

In order to deal with the uncertainty of a natural language, machine learning or manual pre-processing of knowledge can be used (Nazaruka, 2019). Research on NLP techniques used for discovering causal dependencies in texts in prose showed two clear trends (Nazaruka, 2019). The first is increasing the accuracy of the results using ontology banks, machine learning and statistical inferring. The second is decreasing the cost of these activities. In case of construction of software models using machine learning or statistical inferring, the main challenge is a lack of corpuses and statistical datasets for potential problem domains. Nevertheless, this issue can be potentially solved by limitation of those source documents to specifications (requirements, scenario, etc.) having less variability in expressing causality.

The causal dependencies in text may be expressed implicitly and explicitly. The explicit representation means are causal links, causative verbs, resultative constructions, conditionals as well as causative adverbs, adjectives, and prepositions (Khoo et al., 2002). Though, the implicit causality usually is inferred by a reader associating information in the text with their background knowledge (Khoo et al., 2002; Ning et al., 2018; Solstad and Bott, 2017).

Theories attempting to reduce causal reasoning to a domain-general theory can be grouped as associative theories, logical theories, and probabilistic theories (Waldmann and Hagmayer, 2013). Logical theories seem to be more suitable to software development in processing verbally expressed information, since they model causal reasoning as a special case of deductive reasoning. Logical theories frequently analyze conditionals (if/when...then constructs) in the text. Although conditionals do not distinguish between causes and effects. Temporal priorities can be helpful in distinguishing them (Pearl, 2019; Solstad and Bott, 2017).

### 3.1 Domain Model Composition Approaches

The selected 15 approaches published from 2005 till 2020 and available in IEEE and ACM publication bases are presented here in brief and are chronologically ordered from the oldest to the more recent ones.

**Ilieva and Ormandjieva's Approach** (Ilieva & Ormandjieva, 2006) uses plain text descriptions of software requirements in unlimited natural language as a source of knowledge for constructing the domain TO-BE model with the aim to obtain requirements engineering models such as a Use Case Path model, a Hybrid Activity diagram model and a domain model. Tabular presentation and Semantic Networks are used as transition models. For extraction, the approach uses POS recognition and semantic analysis of text.

**Relative Extraction Methodology** (Krishnan & Samuel, 2010) uses user requirements or problem statements as a source of knowledge for constructing the domain TO-BE model and UML Class diagram as the target model. A dependency graph is used as the transition model. For the extraction POS recognition is used with Breadth First Search (BFS) and Depth First Search (DFS) algorithms for processing the graph as well as algorithmic structures for concept, value and action identification and class diagram generation. The obtained class diagram lacks advanced relationships like aggregation and dependency between classes, as well as multiplicities between the classes.

**DAA4BPM.** The Description Analysis Approach for Business Process Management (DAA4BPM) presented in (Friedrich et al., 2011) uses informal textual descriptions of processes (the domain AS-IS model) for Business Process Model and Notation (BPMN) models creation as a target model. The approach uses a dependency graph as a transition

model. The approach uses syntax parsing (factored model of Stanford Parser, action filtering by example indicators), semantic and flow analysis as well as a custom anaphora resolution algorithm.

**ReDSeeDS.** Requirements Driven Software Development System (ReDSeeDS) introduced in 2010-2012 by the international researchers' teams (Kalnins, 2010; Kalnins et al., 2011; Smialek & Straszak, 2012) uses domain vocabulary (the class model with links to WordNet entries) and semi-formal use case specifications in the Requirements Specification Language (RSL) as a source of knowledge. Scenarios are written in Controlled English. The target model is a platform independent model based on UML profile, which includes static structures as classes, components, and interfaces. The behavior is represented as a sequence diagram.

**AnModeler.** AnModeler is a tool for generating domain models from textual specifications (Thakur & Gupta, 2014). Semi-structured use case specifications with unrestricted plain text are used for composing UML Class and Sequence diagrams. The approach employs NLP, sentence structures and transformation rules.

**Nassar and Khamayseh's Approach** (Nassar & Khamayseh, 2015) targets the construction of activity diagrams from Arabic user requirements using NLP tools. User requirements must follow strict writing rules (formal short statements SVO and VSO "verb-subject-object"). The approach uses manually detected tag patterns to generate UML activity diagrams as the output model. As the result this approach allows extracting actions, domain objects without additional details and actors.

**IDM.** The Integrated Domain Modeling (IDM) approach uses semi-formal Use Case specifications as the domain TO-BE model (Slihte, 2015). The language is restricted since steps should be in the form of "subject verb object" (SVO). The approach produces a topological functioning model (TFM) as the target model, employing NLP features and ontology bank for this task. The approach allows determining actions, objects, actors, preconditions, and causal dependencies.

**Domain Model Extractor** (Arora et al., 2016) is capable of extracting UML Class diagrams as a target model from software requirements given in unrestricted natural language form. It uses NLP features and extraction rules. This approach allows extracting concepts, associations and generalizations, cardinalities, and attributes.

**DAA4BPM v.2.** The modified version of the DAA4BPM (Leopold et al., 2017). Knowledge is extracted from business process models given in

BPMN or Event-driven Process Chain (EPC) notations together with corresponding informal textual descriptions in plain text. Resource Description Framework (RDF) triplets are used as the target model. The approach uses syntax parsing (Stanford Parser) and semantic analysis (Stanford Parser, WordNet, and predicates).

**AGER.** The Automated E-R diagram Generation System (AGER) presented in (Ghosh et al., 2018) can generate E-R diagrams from plain text. User statements are given in VSO form. It employs NLP features to detect entities, attributes and relations.

**Kashmira and Sumathipala's Approach** (Kashmira & Sumathipala, 2018) is targeted at generation of Entity Relationship (E-R) diagrams from requirements specification using NLP. The source model is semi-structured use case specifications. The approach uses NLP, machine learning, ontology, and web-mining to achieve its goals. As the result this approach identifies entities, attributes, and relationships between entities/sub entities-attributes, entities-entities (association), entities-sub entities (generalization), attributes-attributes, cardinalities (One to One, One to Many, Many to Many).

**Shweta, Sanyal and Ghoshal's Approach** (Shweta et al., 2018) uses Use Case specifications in a semi-structured plain text form with keywords specified. The target model is UML Class diagrams. The approach uses NLP features and rules for extraction of classes, attributes, and methods. This approach allows extracting actions, domain objects and attributes and relationships between objects without additional information.

**AR2AA.** Automated Requirements to Assertions Analyzer (AR2AA) uses plain text Requirement specifications as a source of knowledge for constructing the domain TO-BE model (Anwar et al., 2020). The target model for the approach is a triplet <Requirements, Actions, Conditions>. The approach employs NLP for identification of nouns, verbs and adjectives, and rules to identify actions and conditions.

**DoMoRe.** Domain Modeling Recommender (DoMoRe) introduced in (Agt-Rickauer, 2020) is author's own implementation of the DoMoRe system that "generates context-sensitive modeling suggestions using SemNet Nad OntoConnector". The approach uses a large text corpus in plain text as input and Semantic Term Network as the target model. N-Grams constructs are used as an intermediate model. It employs Stanford NLP toolkit, syntactic POS patterns for 5-Grams, and statistics of term occurrence.

**Mirończuk's Approach or Fire Report Analysis** (Mirończuk, 2020) uses fire service reports as a source that is the combination of structured data and unrestricted text. The target model is a database with structured data as the records in it. The author employed classification by using supervised machine learning, creation of terms DB, manually created taxonomy, and extraction rules based on manually created patterns. As the result the author extracts concept values from text using predefined extraction patterns.

### 3.2 Characteristics to Compare

In order to determine what of these techniques and principles are used for composition of domain models which should be applied for software development the following characteristics are analyzed:

- **Source model** is a description of a domain, which is used as a source of knowledge about the domain's characteristics at the very beginning of analysis. It can be considered from two viewpoints:
  - Domain AS-IS model represents the problem domain that describes existing processes in the domain,
  - Domain TO-BE model represents the solution domain that that describes demanding processes as system or software requirements and improvements.
- **Mapping** indicates whether an approach has a support of providing formal conformity between domains AS-IS and TO-BE.
- **Source model type** indicates in which form of specification a source model is presented.
- **Source model format** indicates in which format (e.g., plain text, formatted text, hyperlink text, PDF, WORD, etc.) of specification a source model is presented.
- **Target model** represents the target specifications after knowledge extraction and transformation.
- **Transition model** is the transitional knowledge representation (specification) during transformation to the target model.
- **Knowledge extraction techniques** are methods or approaches used for getting knowledge from the source model.
- **Techniques for extraction of relationships** between elements are methods or approaches used for discovering relationships between elements of the source model.

- **Declarative knowledge extraction** indicates whether an approach analyzes the structural characteristics of the domain.
- **Procedural knowledge extraction** indicates whether an approach analyzes the functional and behavioral characteristics of the domain.
- **Synonym processing** indicates whether an approach processes anaphors, synonyms, and homonyms.

### 3.3 Comparison of Approaches

**Domains and Mappings Between Them.** Most of the 15 approaches are focused on analysis of specifications of the required solution, i.e., composing the model of the **domain TO BE**. Three of them DAA4BPM (Friedrich et al., 2011), its modification DAA4BPM v.2 (Leopold et al., 2017) and the Fire Report Analysis are dedicated for the **domain AS IS** analysis and do not consider the domain TO BE at all. The DoMoRe approach is a universal and can be applied for analysis of both the domains. Therefore, none of these approaches provides **conformity between models** of the domains just because none of them considers both domains.

**Source Model Formats.** Most of the considered approaches proceed the source documents in the plain text format (Table 1). Just two exceptions are presented, namely, the ReDSeeDS where hyperlinks are also acceptable and the DAA4BPM v.2 where the text is an informal textual description of processes that supplements BPMN (Business Process Model and Notation) or EPC (Event Process Chain) diagrams.

Source documents in approaches that analyze the domain AS IS – the DAA4BPM, its modification DAA4BPM v.2 and the Fire Report Analysis – are textual descriptions of processes such as policies, reports, forms, manuals, e-mail messages, etc. Reports may combine structured and unstructured data and text in the unrestricted language.

**Source Model Types.** Source documents on the domain TO BE are requirement specifications of different types: unstructured text and semi- or completely structured text in the form of use cases like in the IDM approach, AnModeler, and ReDSeeDS as well as domain vocabularies with hyperlinks to the ontology bank entries in the ReDSeeDS.

Table 1: Source documents type and format. Denotation: SVO – subject verb object, VSO – verb subject object.

Approaches	Source model type	Source model format
Ilieva and Ormandjieva	SR descriptions	Plain text
Relative Extraction Methodology	A large collection of sentences	Plain text
DAA4BPM	Informal textual descriptions: policies, reports, forms, manuals, e-mail messages etc.	Plain text
ReDSeeDS	Semi-formal equivalent of the domain class model with links to WordNet entries, use cases for scenarios in controlled language	Plain text with hyperlinks
Nassar and Khamaysch	Formal short statements SVO, VSO	Plain text
IDM	Use case specifications with steps in the form of SVO	Plain text
Domain Model Extractor	Unrestricted NL requirements	Plain text
AnModeler	Semi-structure unrestricted software requirement specification	Plain text
DAA4BPM v.2	BPMN or EPC diagram and its informal textual description	Plain text and BPMN / EPC
AGER	User statements SVO	Plain text
Kashmira and Sumathipala	Requirement specifications	Plain text
Shweta, Sanyal and Ghoshal	Semi-structured software requirement specification with keywords	Plain text
AR2AA	Textual Design Requirements	Plain text
DoMoRe	A large text corpus	Plain text
Mirończuk (Fire Report Analysis)	The combinations of structured data and unrestricted text	Plain text

**Target Models.** The target domain models are also quite diverse (Table 2) and can be grouped to UML or UML like models presented in 7 approaches, E-R diagrams presented in 2 approaches, BPMN models (just in one approach), Topological Functioning Models (in one approach) and others in 4 approaches. **Transitional Models.** The transition models are required in 4 approaches – Ilieva and Ormandjieva’s approach, Relative Extraction Methodology, DAA4BPM, DoMoRe – mainly for analysis of the semantical relationships between concepts (Table 2). **Knowledge Extracting Techniques.** Let us look at the knowledge extracting techniques used in the provided approaches (Table 3). The first step in the most approaches is parsing of sentences using standards NLP techniques. Thus, tokenization, chunking, POS recognition and dependencies identification are applied. The exception is ReDSeeDS that uses Keyword Based Analysis for the domain TO-BE model (RSL) since the language of the source model is Controlled English.

After syntax analysis, the semantic analysis is performed using dependencies between word pairs in the parsed sentences. In most of the approaches these dependencies are described in the form of lexical syntactical patterns. Their implementation differs in the approaches:

- Algorithmic structure for processing dependency graph as in the Relative Extraction Methodology as well as BFS and DFS traversals;
- Semantic analysis algorithm that uses ontologies and some predefined lists of “keywords”:
  - FrameNet and WordNet, lists of order indicators such as ConditionIndicators, ParalleIndicators, ExceptionIndicators, SequenceIndicators, as well as anaphora resolution algorithm in the DAA4BPM;
  - WordNet in the ReDSeeDS for domain vocabulary and in the Kashmira and Sumathipala’s approach, as well as together with predicates in the DAA4BPM v.2;
- Rules for detecting and extracting separate elements: in the “Domain Model Extractor” AnModeler, AGER (for entities, attributes, and relations), Shweta Sanyal and Ghoshal’s approach (for classes, attributes, relations, and methods), AR2AA (for actions and conditions), and Fire Report Analysis (for values for terms used in the database).

Table 2: Target model and transition model representation.

Approaches	Target model	Transition model
Ilieva and Ormandjieva	Use Case Path model, Hybrid Activity Diagram model, Domain model	Tabular presentation, Semantic Network
Relative Extraction Methodology	UML class diagram	Dependency graph
DAA4BPM	BPMN model	World Model
ReDSeeDS	PIM model based on UML profile: -Static structure as classes, components, and interfaces, -Draft behavior as sequence diagram.	N/U
Nassar and Khamayseh	UML Activity diagram	N/U
IDM	TFM	N/U
Domain Model Extractor	UML class diagram	N/U
AnModeler	UML class diagram, UML sequence diagram	N/U
DAA4BPM v.2	RDF triplets	N/U
AGER	E-R Diagram	N/U
Kashmira and Sumathipala	E-R Diagram	N/U
Shweta, Sanyal and Ghoshal	UML class diagram	N/U
AR2AA	Triplet <R, A, C>	N/U
DoMoRe	Semantic Term Network	N-Grams constructs
Mirończuk (Fire Report Analysis)	Database record with structured data	N/U

\* Donotation used: N/U – not used; TFM – Topological Functioning Model; UML – Unified Modeling Language; BPMN – Business Process Modeling and Notation; PIM – Platform Independent Model; E-R – Entity Relationships; R – requirements; A – actions; C – conditions; RDF – Resource Description Framework.

Classification of sentences according to the lexical syntactical patterns can be performed either manually as it is done in most of the approaches, or by using machine learning as in the Kashmira and Sumathipala's approach and the Fire Report Analysis. **Techniques for Extraction of Relationships.** The extraction of relationships implements the part of the techniques discussed above and is quite diverse in the approaches.

The triads, heuristics and mappings are used sequentially in the Ilieva and Ormandjieva's approach for automated extraction of relationships between elements:

- from text to tabular presentation using POS tagger and their own labeling;
- from tabular presentation to semantic network using relations (triads) and heuristic;
- from semantic network to three target models using mappings between semantic network and models.

The analysis of dependencies in the dependency graph is used in the Relative Extraction Methodology:

- from text to dependency graph using BFS traversal on the syntactic tree for concept and

value identification and DFS traversal on the Verb Phrase (VP) sub-tree of the syntactic tree for action identification;

- from the dependency graph to UML class diagram using BFS traversal on the dependency graph by searching a link between concept and its corresponding elements (value and action) for identification of classes with their attributes and methods.

The syntax parsing and semantic analysis are applied in the DAA4BPM (Friedrich et al., 2011) in the following way:

- Syntax parsing applies the factored model of Stanford Parser as well as action filtering by example indicators; Semantic analysis investigates meanings of words and phrases (synonyms, homonyms, etc.) searching the ontologies FrameNet and WordNet as well as lists of indicators of conditions (ConditionIndicators), parallel execution (ParallelIndicators), possible exceptions (ExceptionIndicators), and the sequential order (SequenceIndicators);

Table 3: Knowledge Extraction Techniques.

Approaches	Knowledge extraction techniques
Ilieva and Ormandjieva	NLP(POS), Semantic analysis of text
Relative Extraction Methodology	NLP(POS), BFS traversal, DFS traversal, algorithmic structures for processing the dependency graph
DAA4BPM	NLP: Syntax parsing (factored model of Stanford Parser, action filtering by example indicators), Semantic analysis (FrameNet and WordNet; lists of indicators ConditionIndicators, ParallelIndicators, ExceptionIndicators, SequenceIndicators), authors' original anaphora resolution algorithm
ReDSeeDS	Keyword Based Analysis for RSL; WordNet for domain vocabulary;
Nassar and Khamayseh	Manually detected tags patterns
IDM	NLP, ontology bank
Domain Model Extractor	NLP, extraction rules
AnModeler	NLP, sentence structure and transformation rules
DAA4BPM v.2	NLP: Syntax parsing (Stanford Parser), Semantic analysis (Stanford Parser, WordNet, and predicates)
AGER	NLP, Detection rules for Entities, Attributes and Relations
Kashmira and Sumathipala	NLP, machine learning, ontology, and web-mining
Shweta, Sanyal and Ghoshal	NLP, rules for extraction of classes, attributes, relation, and operations
AR2AA	NLP; Identification of Noun, Verb and Adjectives; Rules to identify Actions and Conditions
DoMoRe	Stanford NLP toolkit, Syntactic POS patterns for 5-Grams; Statistics of term occurrence
Mirończuk (Fire Report Analysis)	Classification by using supervised machine learning; creation of terms of the database; Manually created taxonomy; Extraction rules based on manually created patterns

- Anaphora resolution algorithm, i.e., identification of a grammatical substitute (such as pronouns “we”, “he”, “it”, certain articles “this”, “that” or a pro-verb) to refer to the denotation of a preceding word or group of words.

In the ReDSeeDS project (Kalnins, 2010; Kalnins et al., 2011; Smialek and Straszak, 2012) the authors use analysis of basic dependencies between words and keywords as well as synsets analysis by using ontology WordNet. The analysis is done in the step of transforming the model of the domain TO BE – RSL- into the platform independent one:

- each step of a use case in the Subject-Verb-Object (SVO) format;
- analysis of the SVO sentence type;
- analysis of keywords in SVO sentences;
- lexicographic term matching;
- WordNet-based similarity measures for comparing synsets (groups of synonyms that express the same concept).

The analysis of the SVO sentence type determines the direction of behavior described in the sentence by using the following rules:

- Actor – system sentence. An actor is the subject, and a system is the recipient of the SVO sentence;
- System – actor sentence. A system is the subject, and an actor is the recipient of the SVO sentence;
- System – system sentence. A system is the subject and the recipient of the SVO sentence.

The Nassar and Khamayseh’s approach transforms text in Arabic (Nassar and Khamayseh, 2015) as a sample is free to apply the algorithm which uses tags patterns and heuristic in order to create the UML activity diagrams

In the IDM approach (Slihte, 2015) the classic NLP features are used and for the outcome other techniques are applied: extraction rules for entities, descriptions of and dependencies between functional features based on the sequence of steps in certain scenario.

The approach using “Domain Model Extractor” (Arora et al., 2016) in order to extract relationships required for the UML class diagram apply the standard NLP features, extraction rules, which use determined dependencies for concepts, associations, generalization, cardinalities, and attributes.

The AnModeler approach (Thakur and Gupta, 2014, 2016a, 2016b) that uses the tool with the same name uses the Stanford NL Parser API to obtain type

dependencies (TDs) and parts of speech tags (POS-tags) of each sentence in the specification. The identified sentence structure and the semantic relationships between the words in the sentence obtained from the TDs and POS-tags are used by the tool to identify domain elements with domain objects, their attributes, operations, and interactions between the objects.

Analysis of individual sentences and extracted RSF triples are used in the modified DAA4BPM approach that we refer as DAA4BPM v.2 (Leopold et al., 2017). First, RDF specification is obtained from the informal textual specification in the following way:

- Stanford Parser is used to automatically transform a text into a list of individual sentences, identify the grammatical entities (the subject, object, predicate, and adverbial) and the relations between them;
- WordNet is used in conversion of all words into their base form;
- Then, RDF triples are used for specifying the obtained information;

The RDF specification is obtained from process models in the following way:

- Using extraction, linguistic analysis, normalization, and transformation of activity labels with its components (action, business object, additional information) into RDF;
- By extracting the inter-relations between activity records.

The inter-relations between activity records in the DAA4BPM v.2 are extracted in the following way:

- pair-wise relations are identified;
- relations are propagated;
- behavioral relations are transformed into RDF

In the AGER (Ghosh et al., 2018) the algorithm for detecting entities, attributes and relations is performed by using the entity, attribute and relation synonyms tables. However, to generate the E-R diagram the BFS algorithm is used.

In the Kashmira and Sumathipala’s approach (Kashmira and Sumathipala, 2018) supervised machine learning module (the “Weka” model), as well ontology and web-mining are used to identify entities, attributes, and relationships from the given text as well as to filter out the relevant attributes into extracted entities. In order to train the “Weka” model, the authors considered four classifier algorithms namely Random Forest, Naive Bayes, Decision Table, and SMO.



In the Shweta Sanyal and Ghoshal's approach (Shweta et al., 2018) Implementation Rules based on the universal dependencies between words (provided by the NLP tools) and authors' keywords are used for the sequential extraction of classes, attributes, methods, and relations.

In the AR2AA (Anwar et al., 2020) the rules are developed to identify actions and conditions in the textual requirements to the design. These rules analyze the POS tagging in the sentences and extracts all actions (verbs) and the corresponding conditions (nouns and adjectives) based on them.

Reducing N-grams to terms and counting their frequency of occurrence in text are techniques for extraction of relationships between elements in DoMoRe (Agt-Rickauer, 2020):

- N-Grams are a sequence of  $n$  consecutive words. It is used for derivation of semantic relationships between co-occurring words and terms.
- Redundancy and Paraphrases. The authors focus on identification of relationships on a conceptual level and use mentions of concepts that occur redundantly and paraphrased as well.
- Word Co-occurrences and Distributions. The authors derive the latent semantic relationship between domain-specific terms from frequently co-occurring words and phrases, as

well as identify technical terms with the help of collocations.

Mind maps and formal concept analysis (FCA) are used for manual extraction of the relationships in the Fire Report Analysis approach (Mirończuk, 2020).

**Declarative Knowledge Extraction** (Table 4). Since, all the target models include the concept of an object or a group of objects (a class) all the analyzed approaches support extraction of the declarative knowledge, i.e., identification of classes/objects, attributes and relations in pairs class-attribute, object-attribute, class-class, and object-object.

**Procedural Knowledge Extraction** (Table 4). The elements describing the dynamic characteristics of the domain are also extracted in all the considered approaches. The approaches, where the target model is an E-R diagram, express the procedural knowledge as an element *Relation* between entities – the AGER and the Kashmira and Sumathipala's approach. In their turn, in the Relative Extraction Methodology and the Shweta Sanyal and Ghoshal's approach – where the target model is just an UML Class diagram – the procedural knowledge is expressed as class operations.

**Synonym Processing** (Table 4). Most of the approaches provides automated processing of synonyms by using corresponding ontology banks.

Table 4: Extraction of the certain knowledge type and processing synonyms.

Approaches	Knowledge Type		Synonyms Processing
	Declarative	Procedural	
Ilieva and Ormandjieva	Y	Y	Y, belonging
Relative Extraction Methodology	Y	Y, class operations	N/I
DAA4BPM	Y	Y	Y
ReDSeeDS	Y	Y	Y
Nassar and Khamayseh	Y	Y	N/I
IDM	Y	Y	Y, theoretically
Domain Model Extractor	Y	Y	Y
AnModeler	Y	Y	Y
DAA4BPM v.2	Y	Y	Y
AGER	Y	Y, relations	Y
Kashmira and Sumathipala	Y	Y, relations	Y
Shweta, Sanyal and Ghoshal	Y	Y, class operations	N/I
AR2AA	Y	Y	N/I
DoMoRe	Y	N	Y, link to ontology bank
Mirończuk (Fire Report Analysis)	Y	Y	Y, manually

\*"Y"-yes; "N" – no; N/I – not indicated.

The exceptions are the Fire Report Analysis where the processing is done manually by the author, and the IDM approach where the authors described such an option theoretically. In the descriptions of four approaches – the Nassar and Khamayseh’s approach, the Relative Extraction Methodology, the Shweta Sanyal and Ghoshal’s approach and the Shweta Sanyal and Ghoshal’s approach – this question is not even discussed.

### 3.4 Discussion on Findings

As mentioned in the beginning of section 3, to minimize the uncertainty of natural languages manual pre-processing of text can be applied. The result of the review approves the same fact, just 6 of 15 considered approaches take as input unrestricted texts. Other approaches either use predefined structures of the sentences (SVO and VSO), or organize the content within some structure supplemented by pre-defined keywords. Most of approaches compose the domain model based on requirements specifications thus assuming further generation of application code or database schemes.

Another way how to deal with the semantical uncertainty is application of ontology banks, machine learning and statistical inferring. The presented approaches demonstrated this fact. Six of them use machine learning, ontology banks WordNet and FrameNet, statistics of term occurrence and even web-mining like in the Kashmira and Sumathipala’s approach. However, if the ontology banks were applied even in research published in 2011, then machine learning is the new thing; among the observed approaches the first mention is in 2018. Ontology banks are also applied for automated synsets analysis.

The target models are quite diverse. They are represented by such “traditional notations” of the domain models as UML (behavioral and structural diagrams), BPMN, E-R and such rare used as RDF triplets, semantic networks, database schemes and TFMs. Thus, this diversity approves that knowledge in all the dimensions of domain models could be automatically extracted and modeled. Besides that, it illustrates that both procedural and declarative knowledge can be processed in the automated way.

The applied knowledge extraction techniques are mostly algorithmic solutions that implements knowledge extracting rules based on keyword analysis, determined word dependencies and lexical semantic patterns. As an input these algorithmic solutions use outcomes of NLP of texts or a transitional model and as an output produce a target

model or a transitional model. A part of rules may share the same patterns. In this activity, machine learning models are used for the sentence classification task.

All the approaches have been validated by their authors. Most of them – manually. The main measurements were the accuracy and the completeness of the extracted knowledge. The results evaluated can be considered as satisfactory, and in a large degree depends on the quality of the patterns and extracting rules.

## 4 CONCLUSIONS

The research has provided the short overview of 15 approaches for automated domain model composition those of using automated knowledge extraction. The compared characteristics have been selected to understand the trends in application of the knowledge extraction techniques for analysis of documentation needed for composing the software domain model.

Main findings are the following:

- Still there is a small number of works that investigate processing of text in unrestricted languages;
- Approaches deal with behavioral, functional and structural dimensions of the problem domain that is approved by the target models used;
- It is not enough just to apply NLP feature for text: analysis of the NLP outcomes is required;
- The analysis of the NLP outcomes mostly is algorithmic and assumes a use of lexical semantic patterns based on dependencies and POS tags of words as well as keyword analysis for structures text and synsets analysis.
- Application of machine learning models is a new thing in the field.

The main manageable issues in the presented approaches relate to the quality of lexical semantic patterns and algorithmic processing of NLP outcomes. However, natural languages uncertainty remains and may affect the results in an unexpected way.

**Construct Validity:** Since various methods for extraction of domain models from text exist, the question about the format of the input information is open – should it be unrestricted or controlled language and what means do we have to process text in it. Thus, this research reviews existing results of other authors’ works that have been published from 2005 till 2020 by IEEE and ACM.

In order to evaluate the current state in the field, the characteristics of the source models, transition models and target models were evaluated, as well as methods or approaches used for procedural and declarative knowledge extraction including synonym analysis. The quality of extraction of knowledge is not analyzed since the diversity of input models and output models as well as a lack of precise information on implementation of extracting rules and algorithms do not allow to make objective conclusions about the degree of validity of the presented approaches.

**Internal Validity:** The presented research works are limited by publication period (15 years) and publication databases (IEEE and ACM). The search of related works has been done using a limited number of keywords: “Knowledge Extraction,” “Domain model extraction,” “Natural Language Processing.” The result of the search was filtered by relevance to the research question and the target model as well as manual processing and composing approaches have been ignored. Multiple publications of the same authors were reviewed, and the most complete publications were taken for the analysis.

**External Validity:** The presented results illustrate that despite having quite advanced NLP features, the main difficulty is processing of NLP outcomes. Language ambiguity, multiple possible constructs for expressing one and the same statements, multiple possible interpretations of a statement require huge work on processing text and does not guarantee that the results of application of any method will be the same or at least will have similar quality for all cases.

Future research directions can be related to investigation on how to obtain more complete set of lexical semantic patterns and extraction rules as well as to techniques used for discourse understanding in specifications written in unrestricted languages and having informal structure. Besides that, the important question is how one can discover the fact that some knowledge is missed and what this knowledge is.

## REFERENCES

- Agt-Rickauer, H. 2020. *Supporting Domain Modeling with Automated Knowledge Acquisition and Modeling Recommendation*, Technischen Universität Berlin.
- Anwar, M. W., Ahsan, I., Azam, F., Butt, W. H., and Rashid, M. 2020. A Natural Language Processing (NLP) Framework for Embedded Systems to Automatically Extract Verification Aspects from Textual Design Requirements. In *Proceedings of the 2020 12th International Conference on Computer and Automation Engineering*, 7–12. ACM. <https://doi.org/10.1145/3384613.3384619>
- Arora, C., Sabetzadeh, M., Briand, L., and Zimmer, F. 2016. Extracting domain models from natural-language requirements: Approach and Industrial Evaluation. In *Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems - MODELS '16*, 250–260. ACM/IEEE. <https://doi.org/10.1145/2976767.2976769>
- Asnina, E. 2006a. *Formalization of Problem Domain Modeling within Model Driven Architecture*, Riga Technical University.
- Asnina, E. 2006b. The Computation Independent Viewpoint: a Formal Method of Topological Functioning Model Constructing. In *Applied Computer Systems*, 26, 21–32. RTU Press.
- Donins, U. 2012. *Doctoral thesis “Topological Unified Modeling Language: Development and Application,”* Riga Technical University.
- Elstermann, M., and Heuser, T. 2016. Automatic Tool Support Possibilities for the Text-Based S-BPM Process Modelling Methodology. In *Proceedings of the 8th International Conference on Subject-Oriented Business Process Management - S-BPM '16*, 1–8. ACM. <https://doi.org/10.1145/2882879.2882882>
- Friedrich, F., Mendling, J., and Puhlmann, F. 2011. Process Model Generation from Natural Language Text. In *Proceedings of the 23rd International Conference on Advanced Information Systems Engineering (CAiSE 2011)*, 482–496. Springer-Verlag Berlin, Heidelberg. [https://doi.org/10.1007/978-3-642-21640-4\\_36](https://doi.org/10.1007/978-3-642-21640-4_36)
- Ghosh, S., Mukherjee, P., Chakraborty, B., and Bashar, R. 2018. Automated Generation of E-R Diagram from a Given Text in Natural Language. In *2018 International Conference on Machine Learning and Data Engineering (ICMLDE)*, 91–96. IEEE. <https://doi.org/10.1109/iCMLDE.2018.00026>
- Ilieva, M. G., and Ormandjieva, O. 2006. Models Derived from Automatically Analyzed Textual User Requirements. In *Fourth International Conference on Software Engineering Research, Management and Applications (SERA'06)*, 13–21. IEEE. <https://doi.org/10.1109/SERA.2006.51>
- Kalnins, A. 2010. A Model-Driven Path from Requirements to Code. *Computer Science and Information Technologies*, 756, 33–57. Horizon Research Publishing.
- Kalnins, A., Smialek, M., Kalnina, E., Celms, E., Nowakowski, W., and Straszak, T. 2011. Domain-Driven Reuse of Software Design Models. In J. Osis and E. Asnina (Eds.), *Model-Driven Domain Analysis and Software Development*, 177–200. IGI Global. <https://doi.org/10.4018/978-1-61692-874-2.ch009>
- Kashmira, P. G. T. H., and Sumathipala, S. 2018. Generating Entity Relationship Diagram from Requirement Specification based on NLP. In *2018 3rd International Conference on Information Technology Research (ICITR)*, 1–4. IEEE. <https://doi.org/10.1109/ICITR.2018.8736146>
- Khoo, C., Chan, S., and Niu, Y. 2002. The Many Facets of the Cause-Effect Relation. In R. Green, C. A. Bean, and

- S. H. Myaeng (Eds.), In *The Semantics of Relationships: An Interdisciplinary Perspective* (pp. 51–70). Springer Netherlands. [https://doi.org/10.1007/978-94-017-0073-3\\_4](https://doi.org/10.1007/978-94-017-0073-3_4)
- Krishnan, H., and Samuel, P. 2010. Relative Extraction Methodology for class diagram generation using dependency graph. In *2010 International conference on communication control and computing technologies*, 815–820. IEEE. <https://doi.org/10.1109/ICCCCT.2010.5670730>
- Leopold, H., van der Aa, H., Pittke, F., Raffel, M., Mendling, J., and Reijers, H. A. 2017. Searching textual and model-based process descriptions based on a unified data format. *Software & Systems Modeling*, 18(2), 1179–1194. <https://doi.org/10.1007/s10270-017-0649-y>
- Miller, J., and Mukerji, J. 2001. Model Driven Architecture (MDA). In *Architecture Board ORMSC*. <http://www.omg.org/cgi-bin/doc?ormsc/2001-07-01>
- Mironóczuk, M. M. 2020. Information Extraction System for Transforming Unstructured Text Data in Fire Reports into Structured Forms: A Polish Case Study. *Fire Technology*, 56(2), 545–581. Springer. <https://doi.org/10.1007/s10694-019-00891-z>
- Nassar, I. N., and Khamayseh, F. T. 2015. Constructing Activity Diagrams from Arabic User Requirements using Natural Language Processing Tool. In *2015 6th International Conference on Information and Communication Systems (ICICS)*, 50–54. IEEE. <https://doi.org/10.1109/IACS.2015.7103200>
- Nazaruka, E. 2020. Processing Use Case Scenarios and Text in a Formal Style as Inputs for TFM-based Transformations. *Baltic J. Modern Computing*, 8(1), 48–68. <https://doi.org/10.22364/bjmc.2020.8.1.03>
- Nazaruka, E. 2019. Identification of Causal Dependencies by using Natural Language Processing: A Survey. In E. Damian, G. Spanoudakis, and L. Maciaszek (Eds.), In *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering - Volume 1: MDI4SE*, 603–613. SciTePress. <https://doi.org/10.5220/0007842706030613>
- Nazaruka, E., and Osis, J. 2018. Determination of Natural Language Processing Tasks and Tools for Topological Functioning Modelling. In *Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering*, 501–512. SciTePress.
- Nazaruka, E., Osis, J., and Griberman, V. 2019. Extracting Core Elements of TFM Functional Characteristics from Stanford CoreNLP Application Outcomes. In E. Damian, G. Spanoudakis, and L. Maciaszek (Eds.), *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering - Volume 1: MDI4SE*, 591–602. SciTePress. <https://doi.org/10.5220/0007831605910602>
- Ning, Q., Feng, Z., Wu, H., and Roth, D. 2018. Joint Reasoning for Temporal and Causal Relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, 2278–2288. Association for Computational Linguistics.
- Osis, J., Asnina, E., and Grave, A. 2007. Formal computation independent model of the problem domain within the MDA. In J. Zendulka (Ed.), *Proceedings of the 10th International Conference on Information System Implementation and Modeling, Hradec nad Moravici, Czech Republic, April 23-25, 2007* (Vol. 252, pp. 47–54). Jan Stefan MARQ.
- Pearl, J. 2019. The Seven Tools of Causal Inference, with Reflections on Machine Learning. *Communications of Association for Computing Machinery*, 62(3), 54–60. ACM. <https://doi.org/10.1145/3241036>
- Prieto-Diaz, R. 1990. Domain Analysis: An Introduction. *Software Engineering Notes*, 15(2), 47–54. ACM SIGSOFT.
- Shweta, Sanyal, R., and Ghoshal, B. 2018. Automatic Extraction of Structural Model from Semi Structured Software Requirement Specification. In *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, 543–558. IEEE. <https://doi.org/10.1109/ICIS.2018.8466406>
- Slihte, A. 2015. *The Integrated Domain Modeling: an Approach & Toolset for Acquiring a Topological Functioning Model*. Riga Technical University.
- Smialek, M., and Straszak, T. 2012. Facilitating transition from requirements to code with the ReDSeeDS tool. In *2012 20th IEEE International Requirements Engineering Conference (RE)*, 321–322. IEEE. <https://doi.org/10.1109/RE.2012.6345825>
- Solstad, T., and Bott, O. 2017. Causality and causal reasoning in natural language. In M. R. Waldmann (Ed.), *The Oxford Handbook of Causal Reasoning*. Oxford University Press. <http://www.oxfordhandbooks.com/view/10.1093/oxfordhb/9780199399550.001.0001/oxfordhb-9780199399550-e-32>
- Thakur, J. S., and Gupta, A. 2014. Automatic generation of analysis class diagrams from use case specifications. In *ISEC '14: Proceedings of the 7th India Software Engineering Conference*, 1–6. ACM. <https://doi.org/10.1145/2590748.2590768>
- Thakur, J. S., and Gupta, A. 2016a. AnModeler: a tool for generating domain models from textual specifications. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering - ASE 2016*, 828–833. ACM/IEEE. <https://doi.org/10.1145/2970276.2970289>
- Thakur, J. S., and Gupta, A. 2016b. Identifying domain elements from textual specifications. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering - ASE 2016*, 566–577. ACM/IEEE. <https://doi.org/10.1145/2970276.2970323>
- Waldmann, M. R., and Hagmayer, Y. 2013. Causal reasoning. In D. Reisberg (Ed.), *Oxford Handbook of Cognitive Psychology*. Oxford University Press. <https://doi.org/10.1093/oxfordhb/9780195376746.013.0046>