CBASeRA: A Compiler Based Approach towards Semantic Requirements Analysis

Atifa Rafiq, Kashif Ayub and Muhammad Ilyas
Department of Computer Science and Information Technology, University of Sargodha
University Road, Sargodha, Pakistan

Abstract. In software development life cycle, requirements engineering plays a major role. It increases the overall success of software. To improve the whole process of requirements engineering, techniques, models and practices are available. Semantic based requirements analysis is one of the technique to improve the overall quality of software. A new methodology is proposed for semantic based requirement analysis which practices compiler based approach. Lexical Analyzer highlights ontologies from each requirement specification, described in natural language. During Ontologies Recognizer, the relationships and entities are extracted by acknowledging ontologies. Now by applying SQL commands on ontologies (entities and relationships), we form Requirements Knowledge Base. Tree Based Semantic Analyzer constructs a well-structured tree of entities and relationships. Semantic application on this tree presents the requirements in unambiguous form. In order to remove the ambiguities, the process of semantic based requirements analysis is described with a case study of a Pizza Shop’s SMS based Order Placing System.

Keywords: requirements, requirements analysis, ontologies, knowledge base, compiler

1. Introduction

Requirements Engineering (RE) becomes one of an important aspect in Software Development Life Cycle (SDLC). RE is the process of identifying the stakeholder’s needs. It is also used to document the requirements for the analysis, communication and development of software. RE consists of many activities like as: eliciting, modeling and analyzing, communicating, agreeing, and evolving requirements [20]. A lot of research work is performed in this area of software engineering. The American Standish Group’s professional research indicates that’s the 50 percent of software failure is almost depend on factors that are related to software requirements [1]. Many theories and practices are available for RE, but there is a big gap between these theories and practices which make a RE process weak [2]. So RE must be strong enough that they increase the overall quality of software.

Under the umbrella of RE, the corporation of semantic web and ontologies has given innovative dimensions. These areas facilitate the requirements elicitation and analysis. They allow communication and understanding between stakeholders throughout the SDLC [3]. Semantics are meanings, which used for requirements analysis and verification. Requirements can be describe in many form like it may be written in natural language (English), in mathematical form, or in semi-formal form (use cases, scenarios) [22, 23]. The ambiguity can be arises in semi-formal or natural language representation of requirements because for most of the participants, it is difficult to understand the requirements which are written in these form.

RE can also be supported with Knowledge Base (KB) systems. They are helpful for the requirements analysis and verification because they can explore large amount of application specific knowledge [14]. Informal requirements are gathers from customer in KB, Requirements Analysis and Knowledge Elicita-
and semantics. For example, first they find out the section gives a conclusion followed by future work. Completely support our proposed framework. The final section 3 the detailed proposed approach is provided.

While Section 4 discusses the case study which correlates to RE, Semantic RE, RKB and Compiler. In SQL commands are used to form a RKB from which generates the ontologies from highlighted information. are highlighted and then Ontologies Recognizer generates the semantics of sentences in natural language is CBA. In this technique the input string is organized as noun, prepositional, verb, adverb, adjective phrases etc. Then these phrases are analyzed during semantic analysis process to remove the ambiguities in words. Knowledge base is used to store all the possible meanings for words [16]. In order to extract information from such KB, we use Structured Query Language (SQL) for database manipulation. Here’s SQL commands are used to manipulate ontologies in Knowledge base.

New requirements analysis method is introduced in this paper which is CBA. Requirements are defined in the form of natural language statements. Now the first phase of proposed framework is Lexical Analysis through which the entities and relations are highlighted and then Ontologies Recognizer generates the ontologies from highlighted information. SQL commands are used to form a RKB from which Tree Based Semantic Analyzer construct a tree. Finally the Requirements Generator produces the furnished requirements by parsing a tree.

In this paper we present the Compiler Based Approach for semantic based RE. For requirements analysis, RKB and SQL commands are used. Section 2 describes context and background work that is related to RE, Semantic RE, RKB and Compiler. In section 3 the detailed proposed approach is provided. While Section 4 discusses the case study which completely support our proposed framework. The final section gives a conclusion followed by future work.

2. Related Work

Related work shows that many research works have been performed to support the RE process. Bashar Nuseibeh et al. presented a comprehensive roadmap to RE. RE activities are re-examined and associated with only core activities. They include the need for requirements elicitation, analysis and modeling [20]. Opdahl et al presented a review of research papers published at ten conductive annual workshops on RE. They performed the qualitative analysis for the evolution of RE activities, so the research interests are disclosed [21].

Semantic based RE and ontologies techniques are used to improve the RE process [18]. These assist in requirements elicitation and analysis. Fengdi Shu, et al. provides the method of requirements elicitation, which based on users individualities and context. This method encourages the user involvement in requirements elicitation and also improves the domain knowledge reuse [17]. Ontologies play an important role in software applications development [4]. These applications include natural language processing [5], databases [6], multimedia [7], data mining [8], and information retrieval systems [9].

Haruhiko et al. proposed an approach in which they apply the knowledge of domain ontology towards the requirement elicitation and analysis process. They map the software requirements description with the domain ontology. Their domain ontology system contains inference rules and thesaurus parts that are suitable for processing the semantics. It facilitates the requirements engineers for requirements specification analysis with respect to application domain semantics. They show three types of semantic processing with the help of case study. These semantic processing includes: identifying the inconsistent and incomplete requirements, quantifying the requirements specification through its meaning, and predicting the changes in requirements [18].

Requirements analysis and verification can be facilitated by the semantic based approaches. A new semantic approach based on domain methodology is presented for the analysis and the verification of requirement [10]. Semantic Wiki is one of the semantic based approaches which is used for RE. It is used explicitly to expose the relationship between requirements elements [11]. Yanwu Yang, et al presented an integrated (two level) framework for semantic based RE (shown in fig. 1). This two level framework is basis for requirements understanding and management. It is also act as eliciting, analyzing,
modeling, communication, and approving requirements. The lower level integrates the user ontology, enterprise ontology and domain ontology for the semantic representation of software requirements. User ontology is used for eliciting and modeling the requirements when user has no clear idea what they want. Enterprise ontology defines the rules, goals, resources and responsibilities with respect to business to hold high level requirements. Domain ontology plays key role, it assists the stakeholders to share background knowledge. The middle level of this framework consists of RE activities including modeling, analysis, communication and evolution. Requirements knowledge is acquired according to application domain. Requirements knowledge base is used to structured and store this knowledge. Now the requirements items can be analyzed. [19].

Semantic based composition is another idea used under the umbrella of formal semantic studies, which benefits the reasoning to identify the conflicts between requirements and also assist the meaningful mapping to derived architecture [12].

Haibo Hu, et al, proposed an approach, structural and formal semantic based on domain ontology and inference rule for analysis and verification of software requirements. They described requirements in natural languages. For requirements analysis, requirements descriptions are decomposed into atomic requirement item which represents with triplet \( D (C, R, AR, X) \) of semantics in domain ontology. Now the requirements are analyzed and verified according to provided framework (figure 2). The domain ontology which is described in this paper consists of semantic elements. These elements are represents as concepts and relationship between these concepts and rules of inference. They map the requirement elements to the domain ontology with the help of inference rules to analyze and verify the completeness, correctness, and consistency. In order to support their idea they used Predicate calculus notations for requirements elements and domain ontology representations. They introduced notations for mapping functions \( Fm(r) \) and inference rules \( D(c)/D(p) \) which describe the instance concept \( c \) or binary relation \( p \) [10].

The KB is another important aspect for analysis of requirements used in RE. In a Knowledge based approach, RAKES is implemented for requirements analysis. RAKES take informal requirements from user and produce formal specification. For input, system uses the requirements in natural language and produce output in FRORL. It also produces another kind of output like as side notes that is stored in knowledge base. A formal analysis is performed to analyze the requirements. This approach is totally concerned with analysis phase; however the information that is stored in knowledge base can also be useful during the entire software life cycle [13]. Overview of RAKES is shown in figure 3.

Fig. 1. An integrated framework for semantic requirements engineering [19].

Fig. 2. Framework of requirement analysis based-on ontology [12].
A KB approach is also used for software requirements elicitation of stakeholders. Unified model of dependability (UMD) framework exploit the knowledge. UMD represents the stakeholder’s requirements and countermeasures organize the system requirements [15]. R.Burlon et al. presented Analysis Assistant (AA) tool, which is knowledge based system focuses to help user in requirements analysis. AA utilize large amount of knowledge on different software domain. AA is the combination of artificial intelligence and software engineering techniques whose objective is to facilitate the starting phase of SDLC. It supports the intelligent environment for syntactical perspective of requirements as well as analysis methodology with respect to application domain. To represent the knowledge in AA, a Knowledge Representation System (KRS) is implemented. KRS provide the way to describe the Production System and Semantic Network. The main parts of KRS are Network Definition Language (NDL), Network Manipulation Language (NKL), and Query Language (QL). The analysis method of AA consist of set of phases in which the details of data and function is supported with the suitable techniques and languages. The framework (shown in figure 4) recognized the three dissimilar and consecutive phases. These phases are Analysis of Needs, Analysis of System Requirements and Definition of Software Requirements [14].

For semantic and syntactic analysis, another good approach is the usage of compilers, which have many applications used in natural language processing (shown in figure 5). It extracts the words and phrases (verbs, nouns, adverbs, adjectives …) from natural language statements. These words and phrases are analyzed to remove the ambiguities from statements. Knowledge Base is used to keep the record of application specific knowledge and general knowledge. It contains all possible meanings of sentences. Lexical analyzer check the syntax and semantic of each word in a statement and then store it in Knowledge Base. In this method the input strings are converted to SQL statements and then computer run these statements [16].
3. Proposed Framework

This paper proposed a new requirements analysis method which uses CBA. Here in this idea a string is an input, this string represents a requirement in simple natural language. Exactly one information string represents a requirement. The requirement string as input went to **Lexical Analyzer**. This analyzer reads the string and highlights the entities and relationships. Entities can be subjects or objects and relationships can be verbs. Now the highlighted information comes into **Ontologies Recognizer**, where Ontologies Jars are generated on the bases of highlighted information from input string. Now the actual string shrinks to core information related to some requirement. **Requirement Knowledge Base (RKB)** forms a Knowledge Base via SQL statement on the bases of relations between recognized ontologies. **Tree Based Semantic Analyzer** generates a well-structured tree of information from these ontologies, so the information gets some shape of representation. Now **Requirement Generator** generates requirements from constructed tree of ontologies and sends these requirements as its output.

3.1. **Lexical Analyzer**

Lexical Analyzer gets string as input. This string consists on a natural language sentence which completely describes a requirement. This requirement passes to Lexical Analyzer where this process re-reads the string completely and highlights some very basic information to it like entities and relationship etc. Normally these relations are of general type in these natural language sentences.

3.2. **Ontologies Recognizer**

Ontologies Recognizer recognizes the highlighted information (Entity1, Entity2, Relation1...) from a string and generates ontologies on the basis of this highlighted information, embeds some extra information like relationships between them. Now actual string breaks into small parts of information and shrinks as well.

3.3. **Requirements Knowledge Base (RKB)**

In this process, Ontologies Jar (O1, O2, O3..., On) form a RKB by using SQL commands. This KB can be of nested form, means multiple relationships can be describe in it. Like as:

Relation1 (Entity1, Relation2 (Entity2, Entity3)) etc.

Above example of KB is actually describing multiplicity of relationships between ontologies. RKB is shown in figure 7.

3.4. **Tree Based Semantic Analyzer**

Tree Based Semantic Analyzer gets the RKB as its input to construct a tree. Normally relationships between entities become even levels of tree and entities on odd levels. This is a valid scenario if we have bi-
nary relationships (as shown in fig. 8) but when there came multiplicity in relationships then this rule is no longer remain applicable. It increases the possibility to have relationships on either level. But one thing is sure either in binary relationship tree or in multiplicity relationships oriented tree, top node on level zero is always a relationship node. This starting node can’t be entity in any case. After generating a well-structured tree, now information of KB is in a well presentable form for any level of study. At this instant, any type of processing is applicable on this tree of information.

![Binary Tree Representation](Image)

Fig.8. Binary Tree Representation

### 3.5. Requirements Generator

Now requirement generator process is going to extract requirements via tree parsing. Tree has entities and relationships between these entities nodes. Requirement generator process is using same information to go through this tree and extract requirements to it. These requirements can be a string or in some other narrations for requirements. Here these requirements are the outputs from our Framework of RE.

### 4. Case Study

Following is the case study of a pizza shop’s SMS based Order Placing System (SMS OPS). We use natural language to describe the requirements. This case study describes implementation of our proposed framework toward semantic based requirement engineering. In this scenario user can do the following things with pizza shop’s SMS OPS.

**Requirements:**
- All customers can place order.
- All customers can place order from pizza shop’s menu.
- All customers can place order from pizza shop’s regular deals.
- Customers can register themselves.
- Register customers can place order from pizza shop’s menu.
- Register customers can place order from pizza shop’s member’s menu.
- Register customers can place order form pizza shop’s regular deals.
- Register customers can place order from pizza shop’s members’ deals.
- Customers who are registered and other customer will tell the spot for delivery.

Above are the English statements which describe the requirements for SMS OPS functionalities. These statements describe each and every requirement properly and separately. These requirement statements can be in any natural language. Now we are going to apply our framework on the above mentioned requirements to get the furnished requirements.

#### 4.1. Lexical Analyzer

Here this process reads all string one by one separately and highlights the relationships and entities from them. This highlight process uses the object, subject, verbs recognition pattern to highlight the words.

- All customers can place order.
- All customers can place order from pizza shop’s menu.
- All customers can place order from pizza shop’s regular deals.
- Customers can register themselves.
- Register customers can place order from pizza shop’s menu.
- Register customers can place order from pizza shop’s member’s menu.
- Register customers can place order from pizza shop’s regular deals.
- Register customers can place order from pizza shop’s members’ deals.
- Customers who are registered and other customer will tell the spot for delivery.

Highlighted words provide the meaning of each requirement. This process treats each requirement individually but later we merge them to get the results as a whole.
4.2. Ontologies Recognizer

Highlighted strings (output of Lexical Analyzer) are the inputs for this process. It recognizes the highlighted words and creates the ontologies. These ontologies also consist of some additional information which is attached with them like either the word is verb, object or subject. This additional information is helpful in constructing the tree. We get the ontologies from this process are given as: 
Here E denotes to Entity and R to Relationship.
– [Customers(E)] [Place order(R)].
– [Customers(E)] [place order(R)] [Menu (E)].
– [Customers(E)] [place order(R)] [Regular deals (E)].
– [Customers(E)] [Register(R)].
– [Register customers(E)] [Place order(R)][Menu (E)].
– [Register customers(E)] [Place order(R)] [Member’s menu (E)].
– [Register customers(E)] [Place order(R)] [Regular deals (E)].
– [Register customers(E)] [Place order(R)] [Members’ deals (E)].
– [Customer(E)] [Registered(R)] [Customer(E)] [Spot for delivery (E)].

4.3. RKB

Here this process creates the Requirement Knowledge Base (RKB) by using ontologies. SQL statements fills this database oriented RKB with requirements entities with respect to relationships between them. This RKB also helps to form final tree.

Our scenario based SQL statements are as:
– INSERT INTO RKB VALUES (customer, regular deals) WHERE customer=’register customer’;
– INSERT INTO RKB VALUES (customer, member deals) WHERE customer=’register customer’;
– INSERT INTO RKB VALUES (order, spot for delivery) WHERE customer=’customer’;

Real RKB forms through relationships taken from Ontologies Recognizer process’s output and entities from RKB database. Now real RKB looks like this:
– Place order(customer)
– Place order(customer, menu)
– Place order(customer, regular deal)
– Register customer(customer)
– Place order(register customer (customer),menu)
– Place order(register customer (customer),member menu)
– Place order(register customer (customer),regular deals)
– Place order(register customer (customer),member deals)
– A- Spot for delivery (place order (customer))
– B- Spot for delivery (place order (register customer (customer)))

9-A represents if there is a simple customer placing order and 9-B represents if there is a register customer placing order. They both have to tell their delivery spot in both scenarios.
This is the final representation of RKB which is further used for tree construction.

4.4. Tree Based Semantic Analyzer

Now this process takes reach statement from RKB as a separate input. It initially forms a tree for the individual requirement and at the end it forms a final tree as its output. Relationship between entities becomes parent nodes and left and right nodes are the entities associated with that relationship.

So tree representation of each requirement statement is illustrated as:
Fig. 9. Tree for requirement 1.

Fig. 10. Tree for requirement 2.

Fig. 11. Tree for requirement 3.

Fig. 12. Tree for requirement 4.

Fig. 13. Tree for requirement 5.

Fig. 14. Tree for requirement 6.

Fig. 15. Tree for requirement 7.
Final tree forms on bases of relationship. Here all nodes merges on the bases of similar relationship nodes and then removes the ambiguities like repeated or similar branches from these newly connected nodes of tree (shown in figure 18).

4.5. Requirements Generator

In this process, we get furnished form of requirements in any format from tree based semantic analyzer. By parsing a tree, final requirement can be either in simple English language statement or even can export complete tree of figure 18 by using some data structures like list, queue etc.
5. Conclusion and Future Work

Our proposed framework semantically analyzed the requirement strings to generate the final requirements in any form. This paper explores the idea for requirements analysis which follows compiler based approach. Requirements are in simple English language statements and ontologies are highlighted during Lexical Analyzer phase from them. SQL commands create Knowledge Base on the bases of ontologies (entities and their relationship). Now tree based semantic analyzer construct a tree of information from these ontologies and requirement generator generates final requirements.

In future, we will try to make verification on the basis of these parameters Correctness, Completeness and Consistency. These issues will be discussed on the output of frame work. We will also try to define output format by using some data structure like, will try to export final tree of Tree Based Semantic Analyzer as output by using link list data structure where each node will hold some information in its data portion which will describe either this is a relation node or Entity node and some addresses which will establish connection between these list nodes.

![Data Structure for output format](Image)

Fig. 19. Data Structure for output format

References
