Model-driven restricted-domain adaptation of question answering systems for business intelligence

Katia Vila
Dept. of Informatics
University of Matanzas
Matanzas, Cuba
kvila@dlsi.ua.es

Antonio Ferrández
Dept. of Software and Computing Systems
University of Alicante
Alicante, Spain
antonio@dlsi.ua.es

ABSTRACT
Business Intelligence (BI) applications no longer limit their analysis to structured databases, but they also need to obtain actionable information from unstructured sources (e.g. data from the Web, etc.). Interestingly, Question Answering (QA) systems are good candidates for these purposes, since they allow users to obtain concise answers to questions stated in natural language from a collection of text documents. Traditionally, QA systems include patterns for dealing with a large spectrum of general questions, namely open-domain question answering (ODQA). However, BI users should be aware of asking questions related to a specific activity of the business (e.g. healthcare, agricultural, transportation, etc.). Therefore, adapting ODQA systems to new restricted domains is an increasingly necessity for these systems to be precisely used in BI. Unfortunately, research addressing this topic has two main drawbacks: (i) patterns are manually tuned, which requires a huge effort in time and cost, and (ii) tuning of patterns is based on analyzing potential questions to be answered, which is not a realistic situation since, in restricted domains, questions are highly complex and difficult to be acquired. To overcome these drawbacks, this paper presents a novel approach based on model-driven development in order to use knowledge resources to automatically and effortlessly adapt patterns of ODQA systems to be useful for restricted-domain BI scenarios.

Categories and Subject Descriptors
I.2.7 [Artificial Intelligence]: Natural Language Processing; D.2.10 [Software Engineering]: Design

General Terms
Design

Keywords
question answering, metamodeling, model-driven development, knowledge organization systems

1.  INTRODUCTION
Traditionally, Business Intelligence (BI) applications have focused on providing mechanisms for storing, managing and analyzing huge amounts of structured data in support of the decision making process, e.g. data warehouses, OLAP (On-Line Analytical Processing) or data mining tools. However, in the last decade, the amount of data available as textual form on the Web has been growing rapidly, and the next generation of BI applications require mechanisms to search the Web for actionable information that improves the decision making process [10]. Interestingly, Question Answering (QA) systems are good candidates for these purposes, since they allow users to obtain concise answers to questions stated in natural language from a collection of text documents or corpus [11]. The development of QA systems has been stimulated by conferences such as TREC\(^1\), CLEF\(^2\) and NTCIR\(^3\), which have been major evaluation forums for QA systems. If the variety of approaches of QA systems are studied, it is observed that most of them present a common architecture (shown on the Question Answering part of Fig. 1). This architecture consists of three different sequential phases: (i) question analysis for analyzing and understanding the question by classifying it, and extracting the significant keywords; (ii) these keywords are used by an Information Retrieval (IR) system in order to select and retrieve the relevant passages or documents; and (iii) finding and extracting the expected answer by using natural language processing tools (such as POS tagger, syntactical parser, entity annotator, semantic role parser, etc.) to analyze this set of passages. If this architecture is analyzed, then it is obvious that the phases of question analysis and answer extraction are dependent on the knowledge from the domain which is usually included in patterns\(^4\).

QA systems can be classified in two types, depending on the application domain: Open-Domain Question Answering (ODQA) system and Restricted-Domain Question Answering (RDQA) system. While the former is concerned about a wide spectrum of questions and performs well in general questions, the latter properly adapts to answer questions related to certain area, thus obtaining more precise results in a specific topic. Interestingly, users of BI applications should

\(^1\)Text REtrieval Conference, http://trec.nist.gov/
\(^3\)NII Test Collection for IR, http://research.nii.ac.jp/ntcir/
\(^4\)Note that for the purpose of this paper, we will refer to all the possible strategies for detecting relationships between elements in both question and answer (e.g., logic forms, regular expressions, syntactical relations, dependency relations and so forth) as patterns.
be aware of asking questions related to the specific activity of the business (e.g., healthcare, agricultural, transportation, and so on). Therefore, RDQA should be used in BI applications to be able to acquire precise answers for questions related to a specific business activity, thus obtaining actionable information. Patterns for RDQA can be designed from scratch, which makes portability of the system highly complex, or from initial ODQA patterns, which may cause problems in the adaptation due to restricted-domain features [11]. Some of these problems of adaptation are: (i) QA patterns are manually tuned [12], which requires a huge effort in time and cost, and (ii) tuning of QA patterns is based on analyzing potential questions to be answered [9], which is not a realistic situation since, in restricted domains, questions are highly complex and difficult to be acquired.

To overcome these drawbacks and being able to design suitable QA systems for BI, we propose an approach to obtain existing patterns from an ODQA system and adapting them by using knowledge resources from a specific domain in an automatic manner. Our approach is based on model-driven software development [16] which has been proven useful for defining and managing several kinds of software models in an easy and structured way with a high degree of automatization. Our initial hypothesis is that the adaptation of ODQA systems to a new domain can be seen as a model-driven software development scenario, in such a way that existing QA patterns and knowledge from the domain are captured in a model which will guide the derivation of the new patterns for the restricted domain. The main benefit of our approach is that the adaptation of QA patterns is automatically done in a systematic, well-structured, and comprehensive manner, which significantly reduce the amount of manual labour required in adapting QA systems to a restricted domain for BI applications.

The remainder of this paper is structured as follows. Section 2 presents some related work. Afterwards, Sect. 3 describes our model-driven approach to use knowledge resources in the adaptation of QA patterns for restricted domains. Section 4 shows the implementation of our approach and its applicability by means of a set of experiments. Finally, Sect. 5 sketches out our conclusions and future work.

2. RELATED WORK

Related research in the BI field has been focused on integrating company-internal structured data with value-adding unstructured information from the Web [3]. Traditionally this integration has been performed through IR systems, such as the works presented in [14, 13]. The main drawback of these approaches is that IR systems only return unstructured information (whole documents or just pieces of text such as passages), in which the user has to further search for the requested information being a tedious and prone-to-fail task. Moreover, these unstructured pieces of information cannot be easily processed by BI applications. To overcome this drawback, some works (as [5]) propose the use of QA techniques. QA can be considered as an intelligent search system that firstly filters the corpora of documents by IR techniques, and later returns the answer of a query. Importantly, as BI users are concerned about obtaining actionable information about one specific business activity, then QA systems must be adapted for increasing their precision when answering specific-domain questions. Nowadays, the process of adapting existing QA systems for a specific domain is manually done by using linguistic resources [12, 15], thus being costly and prone-to-fail. Other approaches analyze potential questions to be answered [9], which is feasible for open domains, where repositories of questions are easily acquired from CLEF, TREC or from the Web, but difficult to apply in restricted domains since comprehensive enough training corpus are hard to find [11]. Our point of view is that patterns should be tuned by using different kinds of knowledge resources from a specific domain. These knowledge sources are also known as Knowledge Organization Systems (KOS)\(^7\); according to the level of detail or granularity of the knowledge they refer to, two kind of KOS exist: generic KOS (such as WordNet\(^6\)) or the more precise domain KOS (such as Agrrovoc thesaurus\(^7\)) for the agricultural domain). However, these KOS have their own formats and interfaces, which must be unified by the QA system, thus being a costly task [11]. Bearing these considerations in mind, we propose a model-driven approach to automatically adapt patterns in QA systems to a restricted domain from the collection of documents by integrating available KOS.

3. MODEL-DRIVEN ADAPTATION FOR QA PATTERNS

Our approach (see Fig. 1) consists of automatically adapting patterns for restricted-domain QA by integrating KOS within a nine-step model-driven process. In each step the models for the existing question and answer patterns are acquired and adapted by using a restricted domain model derived from the collection of documents and KOS from the restricted domain.

3.1 Metamodels for adapting QA patterns

Under the model-driven umbrella, and according to [8], “a model is a description of (part of) a system written in a well-defined language”, while “a well-defined language is a language with well-defined form (syntax), and meaning (semantics), which is suitable for automated interpretation by a computer”. Therefore, on one hand, a model must focus on those important parts of a system, thus abstracting away superfluous details. On the other hand, well-defined languages can be designed by means of metamodeling [2], which provides the foundation for creating models in a meaningful, precise and consistent manner. Our metamodels for restricted domains, question patterns, and answer patterns are described next.

3.1.1 Restricted domain metamodel

Our purpose is creating models that represent terms from the restricted-domain corpus and joining them with their corresponding concepts from the available KOS. To this aim we have defined the restricted domain metamodel which contains the adequate elements to create a variety of these models (see Fig. 2).

\[ \text{The core element in this metamodel is the } \text{RestrictedDomainModel} \text{ metaclass which is useful for creating a model}\]

\(^7\)Knowledge Organization Systems include a variety of schemes that organize, manage, and retrieve information. This term is intended to encompass all types of schemes for promoting knowledge management [7], e.g., dictionaries, thesaurus, or ontologies

\(^6\)http://wordnet.princeton.edu/

\(^7\)http://www.fao.org/agrovoc/
for a particular restricted domain. The CorpusTerm metaclass is useful for representing any of the terms appearing in a corpus. A metaattribute value is used to store the lemmatized value of each term. There are several lexical kind of corpus terms as adjectives, nouns or verbs, which are represented as several subclasses of the CorpusTerm, i.e. AdjectiveTerm, NounTerm or VerbTerm metaclasses. It is worth noting that syntactical relations between these terms (which can be easily provided by PoS tagger and syntactical parser when the corpus is processed) are valuable for being used in further steps of our approach. Specifically, the VerbTerm metaclass has relations to indicate which NounTerm can be seen as subject or as an object. Also the NounTerm can be related to an adjective or to other nouns. These relations are important to detect the multi-words which often appear in restricted domains (e.g. “calcium hydroxide” or “adrenal cortex hormones” in the chemical domain). Also, every kind of CorpusTerm has its own type (coming from several Enumerations as shown in Fig. 2). Finally, every CorpusTerm may also have some semantic information (SemanticLabel metaclass). This semantic information can be provided by open-domain tools when the corpus is being processed in the QA task, such as semantic role parser, Name Entity Recognizer (NER), temporal or numerical expressions recognizer, etc. The SemanticLabel metaclass indicates the name of the technique used to acquire the semantic information, the obtained value by applying these techniques and, also the probability of the certainty of this value.

Furthermore, Concept and Equivalence metaclasses allow the elements of this restricted-domain metamodel to be semantically enriched with concepts and relationships from several KOS. The Concept metaclass refers to an element from a particular KOS. Each of these elements has a value to represent it. Besides, each concept can be related to one or more concepts through relations of synonymy, hypernymy and hyponymy. Each concept may be related to more than one KOS for which the name is indicated and also an ID for the concept within this KOS. This metaclass has an isTop metaattribute that states if it is a top concept in that KOS. Equivalences between a term and a concept can be defined: the metaclass Equivalence represents an association between Concept and NounTerm.

### 3.1.2 Question pattern metamodel

Existing question patterns from ODQA systems must be represented into a model in order to be able to adapt them to the domain represented in a restricted domain model. To this aim we have defined the question pattern metamodel which contains the adequate elements to create a variety of these question pattern models (see Fig. 3). These models will define the system question typology, i.e. question types that the system will be able to answer, thus detecting the kind of expected answer and the keywords of the question. A pattern is represented as a Pattern metaclass in order to have several associated expressions (i.e. Expression and Association metaclasses) which represent a pattern. Moreover, a pattern is associated to an answer type (i.e. AnswerType metaclass).
metaclass), in such a way that the kind of expected answer is known when the classification of the question by choosing the pattern that best fits with the question. A metaclass **Expression** is used to consider every kind of expressions. For example, syntactical labels such as PP-preposition, PtDt-interrogative pronoun or determinant, VBC-verb head, SNS-simple noun phrase, SPP-simple preposition phrase and their values (e.g., an expression PtDt could have “which” as value). Expressions may have some related concepts (e.g., a SNS may have hyponyms of certain concepts within their expression). A metaclass **Association** relates expressions in order to know a sequential order. It has an antecedent and a consequent. An example could be association with name “PtDt-VBC” whose antecedent is an expression “PtDt” and a consequent “VBC”. An **AnswerType** metaclass refers to one or more concepts in order to determine the type of the answer. A metaclass **Concept** contains one or more IDs stored in the attribute that identifies different KOS where this concept is supposed to appear and also several verbs that frequently are associated to the concept.

### 3.1.3 Answer pattern metamodel

Answer patterns must be also represented into a model in order to be able to adapt them to the domain represented in a restricted domain model. The **Answer Pattern** metamodel contains the adequate elements to create a variety of these models (see Fig. 4).

![Answer pattern metamodel](image)

**Figure 4: Answer pattern metamodel.**

An answer pattern model (**AnswerPatternModel** metaclass) may contain one or several answers patterns as a **Pattern** metaclasses. Each of these patterns contains an answer type (**AnswerType** metaclass) which is related to one or several concepts from the restricted domain. These concepts are represented by means of a **Concept** metaclass in order to specify its value, the set of identifiers of that concept in different KOS (idSet) and the set of valid lexical types (typeSet) for this concept. These lexical types are represented by means of the **ELexicalType** enumeration.

### 3.2 Adapting QA patterns by using restricted-domain knowledge

Once our metamodels have been defined, every step of our approach for adapting the existing QA patterns by using restricted domain knowledge is described (see Fig. 1).

#### 3.2.1 Obtaining a restricted-domain model

As our adaptation approach is based on the most relevant terms appearing in the corpus, the first step consists of obtaining a **restricted domain model** (according to our aforementioned metamodel at Sect. 3.1.1) that contains all the available information from these terms previously extracted in the collection processing (see transformation T1 in Fig. 1). The transformation T1 has been defined to automatically obtain most relevant terms from the corpus and defining their corresponding elements in the restricted domain model. This transformation selects these terms based on two constraints: lexical (each term must be a noun, an adjective or a verb) and statistical (terms must have certain frequencies, e.g. relative frequency \( fr \) or \( tf-idf \) frequency [1]). It is worth noting that the threshold values for these frequencies may be modified depending on the specific domain. From each selected term, a class **CorpusTerm** is created (**AdjectiveTerm**, **VerbTerm** or **NounTerm**) with its corresponding lexical, syntactic and semantic information obtained from the corpus processing in the QA task, including the different kind of relationships between them. From now on, a running example based on the agricultural domain from the website of the Cuban Journal of Agricultural Science[^rcca] is used as a proof of concept of our approach, which can be used for acquiring from the Web actionable information for an agricultural, chemical or biological business. Sample relevant terms from our case study are: “chlorimuron”, “sulphonylureas”, “group”, “division” and “growth” as common nouns; “inhibit”, “control” and “belong” as main verbs; and “chemical” and “cellular” as qualifying adjectives. Also, the following syntactical information was extracted: “group” is related to the “chemical” adjective, “division” and “growth” nouns are related to the “cellular” adjective, while “chlorimuron” noun is the subject of the verbs “inhibit”, “control” and “belong”. This information is stored in the corresponding **CorpusTerm** classes (see Fig. 5).

#### 3.2.2 Enriching the restricted-domain model

The second step of our approach consists of adding semantic knowledge to the already defined elements of the restricted-domain model by means of concepts and relationships from different kinds of KOS in order to create an **enriched restricted domain model**. This enrichment step is done in the T2 transformation (see Fig. 1) which allows managing heterogeneous KOS (from a simple taxonomy until a complex ontology) by assuring integration and interoperability among them. The reason is that our metamodel is sound enough for specifying in a model those parts of KOS that will be useful in the following steps of our approach, thus abstracting away unnecessary details. Also, this transformation makes the system adaptable since if a new KOS

must be considered, then we are not obliged to change the whole QA system, but adapting T2 to the new KOS. In order to manage different kinds of KOS, elements of the available KOS that correspond to the elements of the target metamodel (i.e. restricted domain metamodel) are required to be defined. For example, hierarchical relationships of the restricted domain metamodel are found in the broader term and narrower term respectively if the KOS is a thesaurus; hypernyms and hyponyms if it is a lexical database as WordNet; subclass-of and instance-of if it is an ontology; functional dependencies if it is a relational database, etc. To do so, transformation T2 is composed of several extractors that are in charge of dealing with the specific storage format of the KOS (e.g. databases, text files, XML, OWL or RDF files, etc.), thus creating the necessary elements in the restricted-domain model.

Transformation T2 associates each corpus term previously detected with some concept from the domain KOS. First simple words of NounTerm are searched, and then multiwords by using its Related Adjectives and Related Nouns attributes. An Equivalence class is created for associating each new Concept class (including its corresponding KOS classes) to some existing NounTerm classes. The following step is to search for synonyms, hypernyms and hyponyms of the new Concept class in a domain KOS until a top concept is reached. Then, every top concept from the domain KOS is checked to be associated to some concept from a generic KOS (some disambiguation algorithm can be used at this stage), and if this association does not exist then hypernyms (and their synonyms) of this top concept are checked. For each concept belonging to a generic KOS, its hypernyms (and its synonyms) are added to the restricted-domain model until finding a top concept.

Following our running example, we choose to use the Agrovoc thesaurus as agricultural domain KOS, and WordNet as generic KOS. The NounTerm “chlorimuron” (see Fig. 5) in the restricted-domain model is checked to appear in Agrovoc. It is found, so a new concept “chlorimuron” is created and also a new equivalence between both. Then, KOS is navigated by its broader term relationships in order to find those concepts which are hypernyms. The concept “sulphonylurea” is then found, and also “sulphonamides” and “amides” as top concept in this domain KOS. Afterwards, this top concept is intended to be mapped to some concept in WordNet. In this case “amide” is found. Then, by recovering hypernymy relations, concepts “organic compound”, “chemical compound”, and concept “substance” as top concept in this generic KOS are found. All these new concepts are added to the restricted domain model to obtain an enriched one (see sample in Fig. 5).

3.2.3 Obtaining EAT taxonomies from the restricted-domain model

An Expected Answer Type (EAT) taxonomy is used to determine the semantic type of the answer, thus reducing the searching space over documents, while a right answer is obtained. The enriched restricted-domain model, previously obtained, is used to create an EAT taxonomy for the restricted domain by applying transformation T3 (see Fig. 1). Obviously, concepts in this taxonomy can be more or less refined, so a level of granularity should be selected in T3 by applying certain criteria over the enriched restricted-domain model (we refer reader to our previous work [17] for further details). We advocate for creating an EAT taxonomy from the terms in the restricted domain model (and not directly from the domain KOS) in order to assure that it contains those semantic classes more closely related to the domain. For example, if the domain of the corpus is fisheries but only an agricultural KOS is available (which also includes concepts and relationships from fisheries), then it is assured that the resulting enriched restricted model only contains those concepts from the domain KOS related to fisheries, ignoring the rest of the agricultural terms. Therefore, an EAT taxonomy derived from this restricted-domain model will have an adequate size, structure and recall for the actual domain. Finally, it is worth noting that the result of transformation T3 is a sub-model of the enriched restricted-domain model obtained in the transformation T2 (see Sect. 3.2.2).

3.2.4 Obtaining QA pattern models

To adapt existing patterns to a new domain, they have to be first acquired from the baseline ODQA system. Transformations T4 and T5 (see Fig. 1) are responsible for obtaining existing question and answer patterns in, respectively, a question pattern model and an answer pattern model by using the metamodels in Sect. 3.1.2 and 3.1.3. These transformations depend on the kind of the implementation of the system, therefore our approach can manage every kind of pattern by only updating transformation T4 and T5. Fig. 6(a) shows an example of question pattern from our previously developed ODQA system (AliQAn [15]) called “patternEO1” which is implemented in C++. Importantly, the answer type (e.g. “entity_object”) is defined through the TGroup class, while the pattern (e.g. “patternEO1”) is defined with the TPattern class. In AliQAn system the elements of the question (e.g. “which” as PtDt, “be” as Verb and SNS) and syntactical relations between them are defined at the TPattern class. Finally, the elements of SNS type are defined by the identifier of the concepts which are related with the answer type and their corresponding

Figure 5: Sample restricted domain model.
kind of semantic relation (i.e., L-Literal, S-Synonym, and H-Hyponym). The corresponding question pattern model is shown in Fig. 6(b), and it has the following expressions: PtDt (value: “which”), VBC (value: “be”) and SNS (whose related concepts are “musical_instrument”, “building”, etc.); as associations “PtDt-VBC” and “VBC-SNS”; and the answer type is “entity_object”. Lastly, it is worth noting that this pattern makes ODQA system able to identify questions of kind: “Which is the musical instrument that Beethoven was playing?”.

3.2.5 Adapting question answering pattern models

Transformation T6 (see Fig. 1) collects both the EAT taxonomy of the enriched restricted-domain model and models of existing question patterns, in order to generate new question pattern models specifically tuned for the restricted domain. Transformation T7 aims to derive adapted answer patterns model by using the same strategy, so due to space constraints, we focus on explaining transformation T6. The first step is to obtain the concepts from EAT taxonomy of the enriched restricted-domain model (previously obtained in Sect. 3.2.3) and represent them with the Concept meta-class of the adapted question pattern model, together with the required information to fill all the attributes of this class. The next step is to search for relationships between concepts which have been added to the adapted question pattern model (new concepts) and those existing concepts in the question pattern model (old concepts) which allow deciding if patterns of an old concept can be used to define patterns for a new concept. Assuming that every top-level concept of the generic KOS used by the ODQA system has patterns, we will create a new pattern derived of an existing one if the following conditions hold when comparing the old concept and the new concept: (i) they are equal, (ii) they have a common hypernym provided by the same KOS, i.e., they are siblings in the hierarchy, or (iii) they maintain a hyponym-hypernym relation, i.e., they are parent and child in the hierarchy. For example, recalling our case study, concept “sulphonylurea” has “substance” as hypernym top concept which is not among old concepts. However, “object” is an old concept with defined patterns and it has the same top concept hypernym than “substance” which is “physical entity”. Concepts “substance” and “object” are then siblings, so patterns from “object” can be reused to the new concept “substance”. In addition, the concept “substance” and the new concept “sulphonylurea” have a hypernym-hyponym relationship, therefore a new pattern for the “sulphonylurea” concept is also created from the old pattern previously explained for the “object” concept (i.e., “patternEO1” in Fig. 6(b)). The name of the new pattern is “patternSulphonylurea1” (see Fig. 7(a)), and it has the following expressions: PtDt (value: “which”), VBC (value: “be”) and SNS (whose related concepts are “sulphonylurea”, “chlorimuron”, “bensulfuron”, and “chlorsulfuron”); as associations they have “PtDt-VBC” and “VBC-SNS”; and the answer type is “substance_sulphonylurea”. Finally, it is worth to point out that a new kind of question can be answered: “Which is the sulphonylurea used for the weeds control in the soja crops?”
3.2.6 Generating new QA pattern code

Finally, transformation T8 and T9 automatically deploy the corresponding code for a specific QA system patterns. To do so, these transformations are based on the notion of customizable templates in order to capture rules for translating question and answer pattern models into corresponding code for different QA systems. As an example, an adapted question pattern “patternSulphonylurea” resulting for applying our approach is shown in Fig. 7(b).

4. IMPLEMENTATION AND RESULTS

Metamodels and transformations have been implemented by using the Eclipse Framework\(^\text{9}\). Eclipse is an open source project conceived as a modular platform able to be extended by plugins in order to add features to the development environment. In order to give support to modeling tasks, we have used facilities provided by Eclipse Modeling Framework (EMF)\(^\text{10}\) and Graphical Modeling Framework (GMF)\(^\text{11}\). For implementing transformations between models, ATL (Atlas Transformation Language)\(^\text{12}\) has been used. ATL is integrated with EMF, allowing the definition and execution of transformations between models by means of rules that matches elements in source and target models. For example, three rules have been defined for transformation T3 (see Sect. 3.2.3): RestrictedDomainmodel2EAT (for creating a target restricted domain model from the source one), Concept2Concept4EAT (for matching Concept classes in the source restricted-domain model that fulfill the granularity criteria, thus creating new Concept classes in the target model), and NewConcept4EAT (for creating new Concept classes as hyponyms of the Concepts classes created by the previous rule provided the granularity criteria holds). The concept2concept4EAT rule is shown next.

```
rule concept2concept4EAT {
  from c1:RD!Concept
  (c1.hasDescendentsWithMoreThan2Hyponyms())
  to c2:RD!Concept
  (value<-c1.value, hyponym<-c1.hyponym, hypernym<-c1.hypernym)
  do{
    for(i in c1.hyponym)
      if(i.hasDescendentsWithMoreThan2Hyponyms())
        c2.hyponym<-thisModule.newConcept4EAT(i);
        thisModule.resolveTemp(c1.getRestrictedDomainModel()()
          rm21).concepts<-c2.hyponym;
  }
}
```

This ATL rule has a from part with the pattern to be matched in the source restricted-domain model (a Concept class c1 without hypernyms and having more that two hyponyms, which is the granularity criteria). The to part of this ATL rule consists of several elements to be created in the target model (a Concept class c2 which is the counterpart of the aforementioned c1). Also, this rule contains a do part in order to execute the newConcept4EAT rule for each hyponym concept of c1 provided the granularity criteria.

Transformations that generate code have been implemented by using Acceleo\(^\text{13}\), which is integrated in EMF. The aim of Acceleo transformations are obtaining code from models by labeling metamodels. Each transformation is called module. Each module may contain one or several templates which are in charge of generating text or queries for extracting information from the source model. For example, transformation T6 is implemented by the Acceleo module generateQuestionPattern, an excerpt of which is defined as follows.

```
[module
generateQuestionPattern('http://questionpatternmetamodel/1.0/')]
[template public generateQuestionPattern
  (q : QuestionPatternModel)
  [for (a : AnswerType | q.questionTypology)][file
    (a.name.concat('._c'), false, 'UTF-8')]
  TGroup
  generating(a.name.toUpperCase()){
    list<TBS> listBlocks;
    TGroup [a.name]('['+a.name+']');
    [for (p : Pattern | q.hasPatterns)]
      if (p.answerType.name==a.name)
        TPattern [p.name/; [...]]
      }[/ts]
    }[/for]
}[/file] [/for] [/template]
```

This module contains a template that has an input parameter q of class QuestionPatternModel. Next, by using the file label, the name of the output file is indicated. In this file, the code of a function that returns an object of the class TGroup is created. Within this function, the code for generating the patterns included as elements of the Pattern class in the the adapted question pattern model is created (objects of TPattern class), provided the answer type of the pattern corresponds to the name of the question typology of the question pattern model.

**First Experiment.**

A previous study about our baseline ODQA system, named AliQAn [15], was carried out with 180 training questions over the RCCA corpus (i.e. 2024 articles from the RCCA journal), where it was detected that 73.3% of errors in the adaptation of the system was caused by (i) a poor and incorrect EAT taxonomy, and (ii) an incorrect classification of questions for absence or inefficiency of the question patterns. In addition, 17.2% of errors were due to failures in the answer patterns. The precision of the AliQAn system reached a 28.8% taking into account only the first retrieved answer and a 13.6% considering the three first answers. These results are very low as compared with the average (i.e. around 43% of precision) reached by AliQAn in the CLEF evaluation forum (i.e. open-domain scenario). Besides it attained an overall recall of 33%. Consequently, it is crucial to adapt AliQAn to restricted domain for increasing its precision and obtaining actionable information from the Web, thus being suitable for BI.

**Second Experiment.**

First step in this experiment consists of processing RCCA corpus with a POS tagger (MACO [6]) and a syntactical parser (SUPAR [4]), indexing it and computing frequencies for each term. As we consider the most relevant terms as those having \(fr > 25\) and \(tf-idf > 0.01\), 8696 relevant terms were obtained and specified in a restricted-domain model by means of transformation T1. Then, noun terms are used in transformation T2 for enriching the restricted-domain model by applying transformation T3 with the criteria of choosing those concepts with more than

---

\(\text{9}\) http://www.eclipse.org

\(\text{10}\) http://www.eclipse.org/emf

\(\text{11}\) http://www.eclipse.org/gmf

\(\text{12}\) http://www.eclipse.org/atl

\(\text{13}\) http://www.eclipse.org/acceleo
two hyponyms. The EAT taxonomy then contains roughly 10% of concepts from the restricted-domain model. Finally, the rest of transformations of our approach are executed (i.e. from T4 to T9), thus generating 325 new EAT concepts and their corresponding code for about 2600 question patterns and 325 answer patterns (one for each concept). These data show how much effort would require accomplishing restricted-domain adaptation manually and the benefit of applying our approach for supporting this process. Furthermore, it is worth noting that, by using our approach, the precision and recall are 58.3% and 75%, respectively. If we compared these values to the results of our first experiment, we can appreciate an increment in 29.5% of the precision and 42% of the recall. Therefore, there were 57.8% less errors for question patterns and 7.8% for answer patterns. These results show the effectiveness of our approach in the adaptation of AliQAn to the agricultural domain of the RCCA journal, and the suitability for being used in BI.

5. CONCLUSIONS

BI applications require QA systems for analyzing textual data from the Web, thus providing richer insights in the business by means of obtaining concise answer to questions stated in natural language. However, to be used in a wide range of business activities, QA systems must be adapted to restricted domains by integrating different knowledge. In this paper we have presented our approach for tackling the complex task of automatically adapting QA system to restricted domains in a systematic, well-structured, and comprehensive manner by using an innovative point of view borrowed from model-driven software development [16]. We have focused on explaining how to adapt patterns of an ODQA system to a new restricted domain by using different kinds of KOS. The main advantage of our approach is that a QA system can be automatically adapted to a restricted domain to increase precision in answering questions related to a specific business activity, thus allowing BI users to obtain actionable information from the Web. Our immediate future work consists of carrying out a more complete set of experiments to measure the effectiveness of our approach in more domains related to BI applications.

6. ACKNOWLEDGMENTS

This research has been partially funded by the Valencia Government under PROMETEO (Development of Intelligent and Interactive Techniques of Text Mining) project (PROMETEO/2009/119), and by the Spanish Government under Textmex 2.0 project (TIN2009-13391-C04-01). Therefore, there were 57.8% less errors for question patterns and 7.8% for answer patterns. These results show the effectiveness of our approach in the adaptation of AliQAn to the agricultural domain of the RCCA journal, and the suitability for being used in BI.

7. REFERENCES