Indian Journal of Science and Technology, Vol 8(29), DOI: 10.17485/ijst/2015/v8i29/86271, November 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Construction of Ontology for Software Requirements Elicitation

S. Murugesh[1]* and A. Jaya [2]

[1]B. S. Abdur Rahman University, Chennai – 600048, India; murugesh.here@gmail.com
[2]Department of Computer Applications, B. S. Abdur Rahman University, Chennai – 600048, India; jayavenkat2007@gmail.com

Abstract

Background/Objective: Elicitation of requirements from informal descriptions remains a major challenge to be accomplished in software industry. **Methods:** An important task in order to accomplish this goal is to construct an ontology consisting of set of concepts i.e. entities, attributes and relations based on the application domain of interest. The ontology constructed here represents the domain knowledge and requirements are the specialized subset of it. As standard description formalism the ontology is encoded using OWL DL, supported by Pellet reasoned to check the consistency of the components of the ontology. The populated ontology can be queried for matching words using SPARQL. **Findings:** In software development projects, voluminous unstructured text documents from different stakeholders are to be analysed and to be converted into structured requirements. This process of elicitation will be time consuming if it is to be performed manually. Domain specific ontology helps in automating the process of requirements elicitation, this article intends to construct such domain specific background ontology. The findings are elaborated in Section 3. **Improvements:** This article portrays the construction and use of domain specific background ontology containing the concepts and their relationships in the Automated Teller Machine (ATM) operations domain to guide the process of automation of elicitation of requirements from informal descriptions or unstructured text, which otherwise would be time consuming if carried out manually.

Keywords: ATM, Domain Ontology, Natural Language Processing, Software Requirements Elicitation, Unstructured Documents

1. Introduction

Ontology provides a representation vocabulary for the software process, eliminating the conceptual and terminological mismatches. This domain ontology will be used as a reference to guide the elicitation process[1]. This research article focuses on the construction of ontology for software requirements elicitation; Section 2 presents a general overview of domain ontology whereas Section 2.1 expands on the construction of the domain ontology for ATM operations. Section 3 provides the evaluation results of the constructed ATM operations ontology used for retrieving the concepts and finally conclusions and future directions of the research work. Ontologies are used to reconcile gaps in the knowledge and common understanding among stakeholders during requirements elicitation, and significantly improve the quality of elicited requirements. The approach was intended towards a particular application[1]. Requirements elicitation process starts out with a specification which is informal, opaque, and dominated by personal views, while the goal is to have a specification, which is formal, complete, and reflects the stakeholders' common view[2,3]. presents an approach to elicit and analyze domain requirements based on existing domain ontologies and requirements meta-model to elicit and define textual requirements[4]. Proposed GOORE, an approach to goal-oriented and ontology-driven requirements elicitation. GOORE represents the knowledge of a specific domain as ontology and uses this ontology for goal-oriented requirements analysis. This research works

*Author for correspondence

on application of knowledge engineering methodology to construct the ATM operations ontology.

2. Domain Ontology and Lexicon

Ontology is used to design the domain in order to preserve the semantics of the concepts and relations. Ontology contains the concepts in a specific domain and the relationships that exist among the concepts. Integrity and derivation rules can be applied on the ontology to retrieve the concept[5]. Core components of ontology are Classes (types of entities), Properties (designating relationship among entities) and Instances. Recent development in the ontology language is OWL i.e. Web Ontology Language from the World Wide Web Consortium (W3C). Ontology in software engineering refers to what exists in a system model. The software engineering concepts representing a specific domain are captured in the ontology. The relations and constraints can also be represented in the ontology. The ontology possesses knowledge which defines the structure of RDF data, the allowable classes, properties and their characteristics, individuals the RDF instance data i.e. specific thing, instance or concept[6]. The domain specific background ontology consists of two basic kinds of properties i.e. object properties which relate objects to other objects and data type properties which relate objects to datatype values.

This research work portrays the process of development of a domain ontology using Protégé 4.3 with OWL Plugin.

2.1 Construction of Ontology

A simple knowledge engineering methodology was applied to create the ontology. An iterative approach was employed, starting with the basic details in the first pass and then revise and refine the ontology at each step by filling in the details. The ontology is used as background source at the time of mapping with the extracted requirements to filter out infeasible requirements, in the process of extracting the software requirements of a system from informal descriptions specified in natural language.

A lexicon is a list of words along with how each word is used. The domain specific ontology is to be built with the following in mind, domain and scope; the domain of the ontology is to represent the Automatic Teller Machine operations. The function of an ATM is to support a computerized banking operation. The ATM will communicate with the server attached to the bank.

An Automatic Teller Machine (ATM) is a computerized and real-time system that provides the customers of the financial institution access to their accounts and perform various services like withdrawal, deposit, balance enquiry, account transfers etc, without intervening the routine functioning of the financial institution. To use an automatic teller machine, the customer must have an ATM card i.e. a plastic smartcard with a magnetic stripe that contains some security information about the client. To use the services of the ATM, a customer has to insert/swipe the ATM card and enter the Personal Identification Number (PIN). The ATM connects to the Bank's online banking server and authenticates the user and authorizes the transactions to be performed as requested by the customer.

The ATM is initialized with parameters like

TC – Total Cash in the ATM at the beginning of the day, ATM_Max – The maximum amount of withdrawal per day from the ATM, Trans_Max – The maximum withdrawal per transaction by the customer, ATM_Min – is the minimum cash that must be available in the ATM so that transactions can be carried out by the customer.

Important terms (lexicon) in the ontology were enumerated i.e. customer, bank, operator account etc. classes and class hierarchies were defined, for example, class Account consists of hierarchies like savings account, current account and checking account, then properties of classes- slots were defined like, atmlocation – country, state, city and street and define slot restrictions. Subsequently the slot-value type i.e. string, number, Boolean or enumerated that describes what type of value can fill in the slot were defined, for example, one of the option while displaying from which account withdrawal has to be carried out the "options" subclass has enumerated slot value type since the values are, savings, current, checking. Figure 1 visualizes the ATM operations ontology.

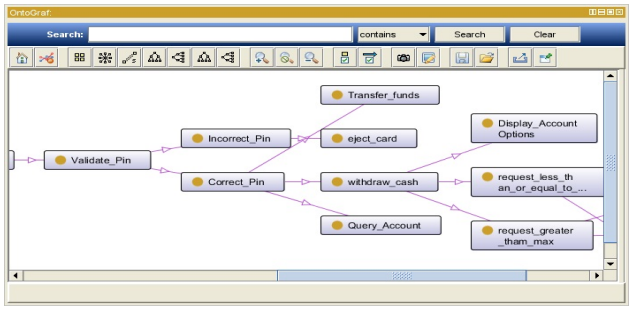

Figure 1. Visualisation of the ATM operations ontology.

The domain of software is a primary candidate for being formalized in ontology. Initially the document containing the software requirements expressed in natural language and in unstructured format is subject to preprocessing like morphological analysis i.e. tokenization, stemming and recognition of end of records followed by syntactic analysis i.e. part of speech tagging and partial parsing[7].

Domain ontology which contains the concepts and relationships for an automated teller machine network domain was built. In software engineering terminology the concepts are entities, tangible objects, classes or actors and relationships are the interaction between the system events with actors.

The Ontology development process emphasizes the activities that are carried out when building the ontology for extraction of requirements from informal natural language statements. It is essential to identify these activities in order to assure correctness and completeness. Figure 2 describes the role of domain ontology in software requirements elicitation.

Begin designing the ontology for ATM system, analyse the nouns and noun phrases, verbs and verb phrases to begin with that will help in identifying the Classes in the system. To identify attributes, look for descriptive words and phrases in the requirements statements, create attributes and assign them to classes, each attribute is to be given an attribute type, some attributes may have an initial value also there may be some classes that may end up without any attributes. The concepts were modelled close to the physical

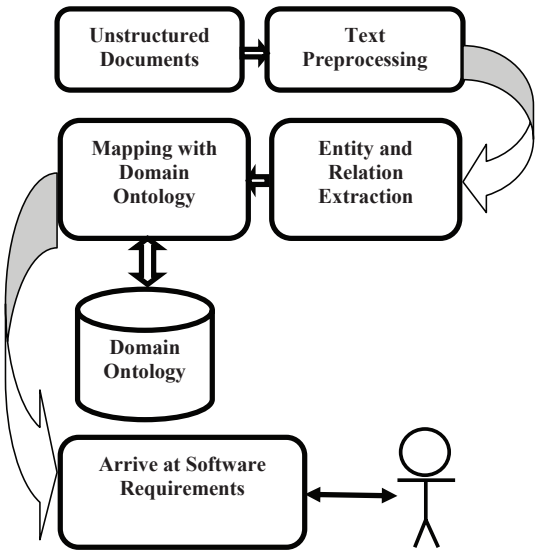

Figure 2. Role of domain ontology in Software Requirements Elicitation.

or logical entities. They were mostly nouns (objects) or verbs (relationships) in sentences of the domain. At initial stages of the design process, some classes may lack attributes, but they should not be removed, since at later stages attributes may creep in. To identify class operations, key verb and verb phrases in the requirements statements has to be examined. Each operation is to be given an operation name and a parameter list and return type i.e. operation Name (parameter 1, parameter 2, parameter 3, ……., parameterN) : return type, however some parameters may not have a return type. Each parameter has a parameter name and parameter type.

The developed ontology contains the domain entities, domain attributes, domain relationships as shown in Figure 3 and Figure 4 visualizes the withdraw cash operation.

Concept Classification or Concept Dictionary, the following actors were modelled as concepts in the domain ontology, they are

Customer: means one or more person or organization,

Bank: a financial institution that allows individuals to own account, issues debit/cash cards and authorizes them to access their accounts over the ATM network,

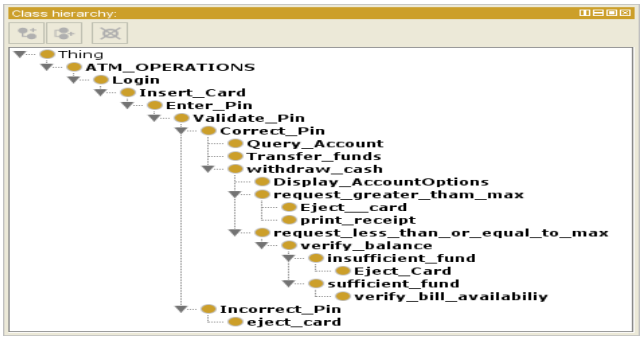

Figure 3. Screenshot of the Protege ontology development environment depicting part of the ATM operations.

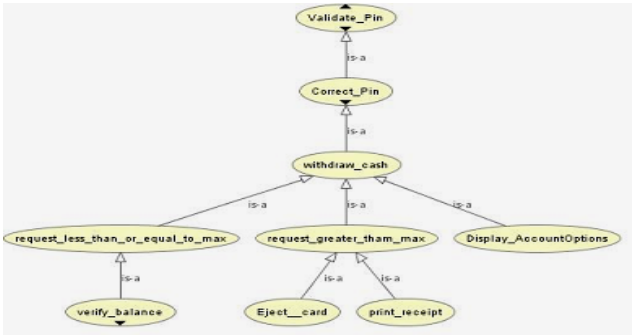

Figure 4. Visualisation of withdraw cash operation.

Operator: the individual or organization that owns the account and carries the transaction,

Account (Savings, Current Account): Account may be of various types like checking, savings against which transactions can be applied, a customer can hold more than one account in a bank,

ATM: is a station that allows customer to carry out their transactions using debit/cash cards with authentication,

Bank Computer: A computer that is owned by the financial institution that interfaces with the ATM network and the banks cashier system,

Cash Card/Debit Card: It is a card that is given to a customer of a bank that authorizes access to accounts owned by them and

Transaction: it is a request for operations like cash withdrawal, transfer, and request for cheque book etc.

Cash Dispenser: to keep track of cash on hand, starting with the initial amount. Report whether enough cash is available and to dispense cash.

Card Reader: To tell the ATM when the card is inserted and to read the information from the card.

Other candidate classes are Balance, BankAccount, Cash, Cheque, Ministatement, PIN, Statement, Printer, BankTeller and Receipt.

The candidate operations or verbs that were identified during initial phases were insertcard, enterpin, authenticate, withdraw, deposit cash, deposit cheques, view balance and print ministatement.

As far as the relationships or use-cases in software engineering terminology are concerned the following were modelled, Deposit transaction, withdrawal transaction, transfer of money from one account to another, balance inquiry transaction, verify/authorization transaction, abort transaction.

All important terms related to banking terminology were enumerated i.e. the list of all nouns and verbs, usually verbs will end up as relations, the relations were classified as subclass, disjoint decomposition, exhaustive decomposition, attributes were defined whenever necessary.

3. Evaluation Results

Evaluation was made with the objective to explore the performance of the ATM operations ontology in terms of accurateness and completeness in extracting actor and use case for ATM operations requirement document when compared to the same requirement identified manually.

The evaluation process used requirement specification document for ATM operations, the file containing set of 57 informal descriptions of requirements of ATM system. The requirements document submitted initially were itemized/tokenized as a list, then the extracted requirements were mapped with the concepts and relations in the ATM operations ontology. The size of the ontology is Number of concepts + Number of relations (including is-a) +Number of axioms divided by Number of concepts. Accordingly the size of the ATM operations ontology is 1.4.

The summary of the subset of actors and use cases identified using the developed ontology is given in Table 1.

For matches that were partial, some of the attribute values had to be modified. The reasons for the modification are

- Use of plural in the place of singular, eg. ATM machine reads ATM cards, i.e. in the place of card, cards were used.

Table 1. Guidance from the ATM operations ontology in identifying actors and use cases

Description	Identified manually	Identified using ontology
Actors	Bank	Bank
	Customer	Customer
	Operations	Operations
Use-case	Transfer of funds	Transfer
	Withdrawal of funds	Withdrawal
	Deposit	Deposit
	Verify Cash	
	Perform Inquiry transaction	Balance Inquiry
	Abort Transaction	
	Determine invalid PIN	

Table 2. Evaluation results statistics about the guidance from the ATM operations ontology

Entity	Count
Concepts	57
Relations	23
Verbs	22
Subject-Verb-Object	46

- Use of different determiner instead of the usual one i.e. instead of using 'the', 'a' is used.
- For some tokens there were no guidance received from the ontology, for example, use of numbers for the <number> attribute, words were used like, seven, nine etc which were not listed in the ATM operations domain ontology.

Apart from these, the extracted information was traditionally evaluated using Precision and Recall. Precision measures how many of the items that the system identified were correct, apart from whether the system failed to retrieve correct items. Higher precision implies what has been identified is correct.

Recall measures the number of correctly identified items as a percentage of the total number of correct items, i.e. how many of the items that should have been retrieved have been retrieved. Higher the recall rate the better the system is. The average number of relations extracted from the sentences containing abstract entities is 0.57 per document. In this group of extracted relations, precision of 52% and recall of 56% were recorded.

4. Conclusions

It could be very important for software developers to be able to make use of domain specific ontology during the process of requirements elicitation since it reduces the time and effort required to arrive at the requirements. While the system has been designed to handle ATM operations domain, it could be adapted to any domain by replacing the ontology, the corresponding ontology lexicon along with the relevant inference. Furthermore since there is a consistent use of standard OWL for the ontology, it is planned to use RDF for the actual data to be verbalized and SPARQL for modeling contextual conditions.

5. References

1. Omoronyia, Sindre G. A domain ontology building process for guiding requirements elicitation. Institute of Software Technology and Interactive Systems Vienna University of Technology Vienna, Austria, Requirements Engineering: Foundation for Software Quality Lecture Notes in Computer Science, Springer. 2010; 6182:188–202.
2. Pohl K. The three dimensions of requirements engineering a framework and its applications, Journal of Information Systems, Elsevier Science Ltd. Oxford, UK. 1994 Apr; 19(3):243–58.
3. Zhao W. An Ontology-Based Approach for Domain Requirements Elicitation and Analysis. In: Proceedings of the First International Multi-Symposiums on Computer and Computational Sciences. 2006; 2:364–71.
4. Kaiya H, Saeki M. GOORE: Goal-Oriented and Ontology Driven Requirements Elicitation Method. In: Hainaut J-L, Rundensteiner EA, Kirchberg M, Bertolotto M, Brochhausen M, Chen Y-PP, Cherfi SS-S, Doerr M, Han H, Hartmann S, Parsons J, Poels G, Rolland C, Trujillo J, Yu E, Zimányie E. (eds.) ER Workshops 2007. LNCS, Springer, Heidelberg. 2007; 4802:225–34.
5. Bures, Simko. Requirement specifications using natural languages, Technical Report; 2012 Dec.
6. Nahm UY. Text Mining with information extraction, Research Thesis; 2004. p. 1–42.
7. Gelbukh A. Natural Language Processing and its Applications Special Issue, Research in Computing Science, published by center for Computing Research of IPN. 2010 Mar; 46:3–105.