Goal Oriented Requirement Engineering

Shetty Balaji S. Bombade Balaji R.
Shri Guru Gobind Singhji Institute of Engineering and Technology, Nanded
India-431606

Abstract:- **The present paper proposes a methodology for goal oriented requirement engineering. It has three phases: 1.Gathering information from stakeholders results in a creation of 'Goal Graph'. An experienced analyst would select a single goal from multiple goals contained within a goal graph (Initial Goals) and distribute the same to all stakeholders. The method named "Natural Language Syntax and Semantics (NLSS)" would be employed to gather ideas from stakeholders. 2. These gathered ideas can be classified into two types of class viz. Functional and Non-functional. The 'Transductive Support Vector Machine (T-SVM)' technique helps classify corpus of ideas in to aforementioned classes and it also improves a speed of classification. 3. A different goal graph is then prepared for each classified ideas. The crosscutting table is then used to depict the dependency among these ideas.**
Said phases can be repeated for all other initial goals and the goal graphs.

Keywords: NLSS, TSVM, Goal, Idea.

I INTRODUCTION

A process of software requirements analysis consists of requirements elicitation and requirements description. Requirements elicitation is a phase where an analyst or expert facilitator collects information from the stakeholders, clarifies the problems and the needs of the customers and users (simply we use "stakeholder" from here), tries to find the best solutions, and makes its planning on what software system will be developed. A family of goal oriented analysis method is top down approach for refining and decomposing the needs of customers into more concrete goals that should be achieved for satisfying stakeholders' needs. The resulting artifact is a goal graph whose node represents identified goals.

To strengthen the goal oriented requirement engineering method we have divided it in different phases. In the first phase we collect the information from all the stakeholders. This information is outcome of general discussion of with the stakeholders. This information is considered as the initial goal, which is selected one by one for clarification and understanding it in detail. The selected goal converted in to specific format used in Natural Language Syntax and Semantics(NLSS)methodology.

This methodology provides special syntax and semantics format so that the discussion can be understood by analyst and the stakeholders easily. In this approach we have also tried to atomize the job of analyst by using Transductive Support Vector Machine (TSVM) [6] classification method. This method is used mainly for text classification by training the system. It has advantage of improving the classification output with less number of training examples. The ideas from the stakeholder after discussion are documented for classification. The classification output is the predefined classes, which are further used for regenerating the goal graph for requirement gathering.

II PREVIOUS WORK

Much more work is done for requirement engineering; The GREMSoC [5] methodology is to promote an approach to improve reusability, maintainability and comprehensibility of requirements specification by means of separation of concerns principle. But the Verification of catalogs to identify and specify concerns is time taking.

. The AGORA [6] goal graph method explains the requirement with the goal graph but it does not provide the methodology for decomposing the goals into sub goals. So the clarification of the goals and the detailed requirement gathering of each goal are restricted. They did not consider that goal elicitation should be the collaborative task done by a team of stakeholders who have knowledge of different domains.

The goal based requirement analysis method (GBRAM)[11] was developed as a response to the lack of goal identification techniques in other GORE (goal oriented requirement engineering) Methods. GBRAM provides set to heuristics for goal identification and their elaboration into a software document. This approach have disadvantage like no goal elicitation technique for gathering of requirements.

III OUR APPROACH

Our method starts with the collection of information from stake holders (stakeholders are the people that have direct or indirect influence in the project. They may be users, clients, managers, developers, etc.). Results with the creation of goal graph which expresses the high level motivations behind the intention. This is typically an incomplete sketch of the form of AND-OR graph called 'Goal Graph'. All the phases of approach are shown in

figure 1.
After this an analyst select a goal, from the gathered goals (from goal graph or collected initial goals),[2] which is most relevant and important of further discussion. With this selected goal we can collect the required information in details, which also provide the facility of verification of goal with all the stockholders.[1] This goal will be given to the group of stakeholders for further discussion by applying NLSS [8] methodology specified below . The details information gathering on the selected goal will provide additional information for that goal.

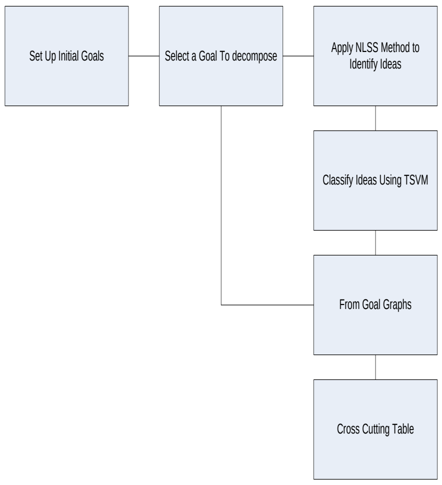
Set Up Initial Goals
Select a Goal To decompose
Apply NLSS Method to Identify Ideas
Classify Ideas Using TSVM
From Goal Graphs
Cross Cutting Table

Figure 1. Phases of the approach

NLSS METHODOLOGY

NLSS methodology has a specific syntax of writing requirements in a natural-language-like form, which is understandable to both the users and the analysts; such a syntax helps analysts to determine the questions of point (i) and additionally the answers to these questions; (ii) a clear analysis of the requirements in terms of system functionality, system entities and non-functional requirements. The questions to the stakeholder asked with this specific content on the functionality (e.g. the functional type of the requirement, prioritization, and common terminology of functions), the components (e.g. objects, people) and the non-functional requirements of the system; such questions are of close-end type to be easily understood and answered by the user. These questions comprise the results of the actual requirements analysis. According to NLSS methodology, every requirement must conform to a predefined syntactic form, which indicates the formal way of writing a requirement. The general pattern presented below for the syntax of a requirement (the second pattern is simply the decomposition of the first):

a. Requirement = [Subject][Verb][DirectObject][IndirectObject][Adverbial Complemnent]

b. Requirement = [Adjective, Noun][Verb][Adjective, Noun][Adjective, Noun][AdverbialComplemnent]

Every requirement sentence is written in the above syntax, which includes its functional part denoted by the verb, the people involved denoted by the subject and the indirect object, the relevant information objects denoted by the direct object, and the relevant non-functional part denoted by the adverbial complement; all these are written syntactically and semantically correctly in just one sentence. Analyst need to know and identify these roles to clarify and define what every entity does in the system, its properties and responsibilities. Furthermore, that helps the analyst to derive relevant questions to ask the stakeholder. The verb and adjectives help to analyze the answer from the user (called as 'ideas': answers for some specific goal), depending on the contents the answers can be classified in to functional and nonfunctional requirement classes. The ideas, which are stored in document and predefined classes, are input to the TSVM for further processing.

TSVM

TSVM classification is the automatic assignment of documents (ideas/answers from the stakeholder) to a fixed number of semantic categories. Each idea can be in multiple, exactly one, or no category at all. This is a supervised learning problem. To facilitate effective and efficient learning, each category is treated as a separate binary classification problem. Each such problem answers the question of whether or not a document should be assigned to a particular category.
Ideas, which typically are strings of characters, have to be transformed into a representation suitable for the learning algorithm and the classification task. The word stems work well as representation units and that for many tasks their ordering can be ignored without losing too much information. The word stem is derived from the occurrence form of a word by removing case and flections information. For example "computes", "computing", and "computer" are all mapped to the same stem "comput". This leads to an attribute-value representation of text. Each distinct word Wi corresponds to a feature with TF(Wi; x), the number of times word Wi occurs in the document x, as its value.

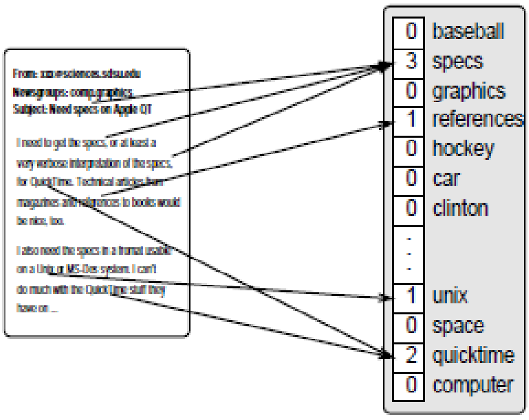
From: xxx@sciences.sdsu.edu
Newsgroups: comp.graphics
Subject: Need specs on Apple QT
I need to get the specs, or at least a very verbose interpretation of the specs, for QuickTime. Technical articles from magazines and references to books would be nice, too.
I also need the specs in a format usable on a Unix or MS-Dos system. I can't do much with the QuickTime stuff they have on ...
0 baseball
3 specs
0 graphics
1 references
0 hockey
0 car
0 clinton
1 unix
0 space
2 quicktime
0 computer

Figure 2. Representing text as a feature vector

Figure 2 shows an example feature vector for a particular document. Refining this basic representation, it has been shown that scaling the dimensions of the feature vector with their inverse document frequency IDF(Wi) leads to an improved performance. IDF(wi) can be calculated from the document frequency DF(Wi), which is the number of documents the word Wi occurs in.

$$\mathrm{IDF}(w_i)=\log(\ )$$

Here, n is total number of documents. Intuitively, the inverse document frequency of a word is low if it occurs in many documents and is highest if the word occurs in only one. To abstract from different document lengths, each document feature vector xi is normalized to unit length.

The setting of transductive inference was introduced by Vapnik (see for example [Vapnik, 1998]).
For a learning task

$$P(\ ,y)=p(y|)p();$$

The learner has given hypothesis space H of function h:X->{-1,1} . Sample Strain of n is training examples

$$(\ _1, y_1), (\ _2, y_2),\ldots\ldots\ldots\ldots, (\ _n, y_n);$$

And S_{test} examples

$$_1^*, _2^*, \ldots\ldots\ldots\ldots\ldots _k^*$$

The transductive learner L aims to selects a function hL = L(S_{train}, S_{test}) from H using Strain, S_{test} so that the expected number of erroneous predictions

$$R(L)= (h_L(_i^*),)dP(_1,y_1)\ldots dP(_k^*,$$

The training and the test classification examples collected as documents from the stakeholders considered as input x(S_{train}, S_{test}) with predefined classes yi (-1,1) and applied for training the support vector machine. The training and the test sample split the hypothesis space H into a finite number of equivalence classes. Two functions from H belong to the same equivalence class if they both classify the training and the test sample in the same way. The hyper plane will classify depending on the values of y which can be defined using transductive support vector machine, consider hyper planes:

$$h()=\mathrm{sign}\{.\ +b\};$$

For linearly separable problems this leads to the following optimization problem. Figure 3 shows the diagram for classification with support vector machine.

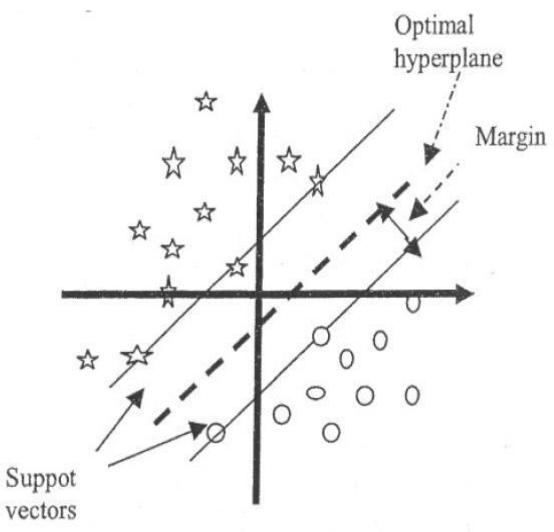

Figure 3. TSVM classification for separable case

For linearly separable case

Minimize over (

$$||||^2$$

subject to : $:y_i\,[._i+b]1;$

$$:[._j^*+b]1;$$

For non separable case

Minimize over
():

$$||||^2 + C + C^{**}$$

Subject to: : $:y_i\,[._i+b]1-\xi_i;$

$$:[._j^*+b]1-\xi_j^*;$$
$$:\xi_i$$
$$:\xi_j$$

C and C* are the parameters set by the user they allow the reading off margin size against misclassifying training examples or excluding test examples. The classification results in separating functional and non functional requirements. After classifying goals, construct the goal graphs for classified goals separately. It is necessary to identify among the goals which of them are crosscutting. A requirement is crosscutting if it affects other requirement in such a way that the crosscutting requirement needs to be applied in some points of other requirements specification. The next step is specifying how each crosscutting requirement affects the requirements it transverses.

The analyst can relate the sub goals of the FR (functional Requirement) graph and those of the NFR

(Non Functional Requirement) graph [3], so dependency among these goals can be represented using a table called as cross cutting table. The vertical axis of the table shows functional goals and the horizontal one is non-functional goals. Figure 4 shows the crosscutting dependency. As shown in the diagram the FR and NFR will be verified for finding the dependency.

All the procedure for forming the goal graphs from the initial goals can be repeated for other initial goals and for the constructed goal graphs by considering the constructed goal graphs as initial goals until the required information is gathered.

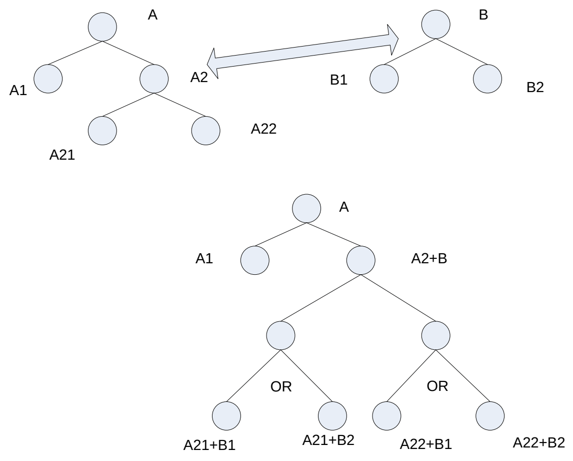

Figure 4. Dependency among FR and NFR

IV IMPLEMENTATION

The project implementation phases are shown in figure 5. It includes suffix stripping, word frequency count, and SVM classification. Words can appear many times with different suffixes (ex. Work, working, worked) in the requirement document. Suffix stripping algorithm converts every word to its root word by removing its suffixes. It implemented in C++. Once all the words are converted to its root form then the number of times each word appeared is to be counted, it is called word frequency database. This module is also implemented in C++. Directory of keywords is prepared by analyst It contains keywords and its corresponding labels. Label specifies functional or non-functional .This directory is applied to SVM(Support vector machine) for classification along with word frequency database, so that SVM can classify in their respective classes (functional and nonfunctional). SVM is implemented in MATLAB (MATLAB 7.5) the basic methodology of SVM is discussed in section IV. The suffix stripping and frequency count modules are compiled with the MEX compiler and attached with the SVM classification module in MATLAB.

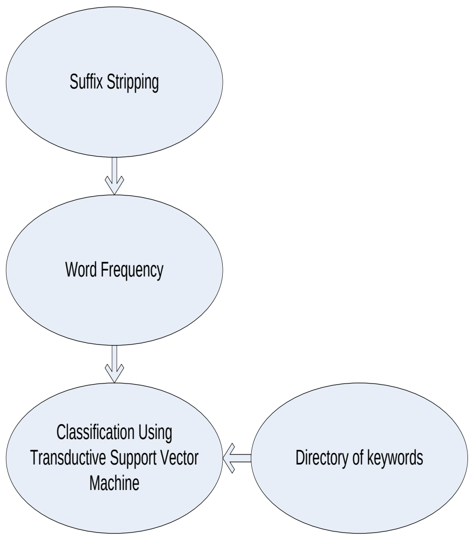

Figure 5. Phases of implemenation

V RESULTS

The requirement documents have been provided to an expert analyst to evaluate it manually using his knowledge, stakeholder’s knowledge and catalogue like NFR [9]. Same requirement documents are applied to our approach and the processing is done automatically. Table 1 shows the result of classification using manual and automatic method as well.

Table 1 Result of classification

No. of Req-uire-ments	No. of key - words	Manual classification		Using our method	
		Accu-racy (%)	Time (min)	Accu-racy (%)	Time (min)
14	6	100	30	57	1.578
40	10	100	50	82.7	1.890
62	18	100	75	90.32	1.998
100	24	100	110	100	2.12

The elevator system [10], a case study is used to create requirement documents in our experiments.

VI CONCLUSION

In this paper, we have proposed new goal oriented requirement engineering approach with natural language, to make understanding of the requirements easier, for analyst and the stakeholders. This approach also reduces the work of analyst by classifying the requirements into specific classes. With this method we have also discussed the method of clarifying and gathering information from the initial as well as sub-goals.

VII REFERENCES

[1] P. Checkland, J. Scholes, Soft Systems Methodology in Action, Jhon Wiley and Sons, 1990.
[2] K.Oshiro, K.Watahiki, M.Saeki., Goal-Oriented Idea Generation Method for Requirements Elicitation,,In Proceedings of 11th IEEE International Requirements Engineering Conference, pages: 363-364, 2003
[3] Kaiya,H.,Saeki,M.Weaving Multiple Viewpoint Specifications in Goal Oriented Requirement Analysis,11th Asia Pacific Software Engineering Conference, pages:418-427, 2004.
[4] Kazuya Ohshiro, Kenji Watahiki, Motoshi Saeki, Integrating Idea generation Method into a Goal Oriented analysis Method for Requirement Elicitation, In Proc.of the 12thAsiaPacificSoftwareEngineeringConference,pages:113-121,2005.
[5] H. Kaiya, H.Horai, M. Saeki, AGORA: Attributed Goal-Oriented Requirements Analysis Method, In Proc. of the 10th IEEE international Requirements Engineering Conference, pages:13-22,2002.
[6] Thorsten Joachims, "Transductive interface for text classification using support vector machine",
Proceedings of the Sixteenth International Conference on Machine Learning, Pages: 200 – 209, 1999.
[7] Georgiades, M.G.; Andreou, A.S.; Pattichis, C.S.,"A Requirements Engineering Methodology Based On Natural Language Syntax and Semantics", 13th IEEE International Conference on Requirements Engineering, Pages: 473 – 474, 2005.
[8] Shahzad Anwer, Naveed Ikram,"Goal-Oriented Requirement Engineering: A Critical Study of Techniques", 13Th Asia Pacific Software Engineering Conference, pages: 121-130,2006.
[9] Lawrence Chung, Brian A. Nixon, Dealing with non-functional requirements: three experimental studies of a process-oriented approach, 17th international conference on Software engineering, pages: 25 – 37, 1995.
[10] http://www.objectiver.com/ fileadmin/ download/, Respect-IT KAOS tutorials.2007
[11] Annie I. Antn, Goal-Based Requirements Analysis, 2nd International Conference on Requirements Engineering, page-136, 1996.