A NEW APPROACH TO REQUIREMENT ELICITATION BASED ON STAKEHOLDER RECOMMENDATION AND COLLABORATIVE FILTERING

Nilofar Mulla[1] and Sheetal Girase[2]

[1]Department of Information Technology, MIT Pune 38, Maharashtra, India
nilofar_mulla2005@yahoo.co.in
[2]Asst.Prof. Department of Information Technology, MIT Pune 38, Maharashtra, India
sheetal.girase@mitpune.edu.in

ABSTRACT

The customers' needs in a software project are identified in the process of Software requirements elicitation. For building a software system this process is considered as one of the most important parts. In this part it is decided precisely what will be built. A close interaction between developers and end-users of the system is needed by requirements' gathering. Meetings can be costly, inconvenient and infrequent if developers and end-users are in different organizations or different cities. The quality of the elicited requirements can greatly be impacted if there is a problem of communication. Requirement elicitation is a process difficult to scale to large software projects with many stakeholders which involves identifying and prioritizing requirements. A stakeholder is an individual or a group who can influence or be influenced by the success or failure of a project. Existing methods to identify and prioritize requirements do not scale well to large projects. Large projects tend to be beset by three problems: information overload, inadequate stakeholder input, and biased prioritization of requirements. Existing methods to identify and prioritize requirements do not scale well to large projects. Existing requirements prioritization methods require substantial efforts from the requirements engineers when there are many requirements. To address the problems Stakeholder recommender model will contain steps:-Identify the large project, Analysis of requirements, Identify and prioritize stakeholders, Predict requirements, Prioritize requirements. For making predictions, our approach will use one of the most well known algorithms that is k-Nearest Neighbor (kNN) algorithm. KNN is used to identify like-minded users with similar rating histories in order to predict ratings for unobserved users-item pairs. A unique subset of the community for each user is found out by KNN by identifying those with similar interests. To do so, every pair of user profile is compared to measure the degree of similarity. A neighbourhood is created for each user by selecting the k most similar users. The similarity between each pair of user profiles for users in the neighbourhood is used to compute predicted ratings. Finally, the predicted ratings for the items are sorted according to the predicted value, and the top-N items are proposed to the user as recommendations, where N is the number of items recommended to the user.

KEYWORDS

Requirements Elicitation, Stakeholder, k-Nearest Neighbour

1. INTRODUCTION

The most important activity in software project development is the requirements engineering. For a computer based system, activities involved in discovering, documenting, and maintaining a set of requirements are covered by requirements engineering. due to wrong requirements, numbers of

DOI : 10.5121/ijsea.2012.3305

consequences may arise like the system may be delivered late, system may be more costly than the original estimation, end-user and , customer will not be satisfied, system may be unreliable and there may be regular system defects. According to the survey conducted by ESPI in 1995 that about 40-60% of all defects found in a software project can be traced back to errors made during the requirements stage. According to another survey conducted by Standish Group Study, 1994 that 13.1% projects fail due to the incomplete requirements and 8.8% projects fail due to the rapidly changing in the requirements. There is a need to follow the best practices, tools, technologies, processess and methodologies for requirements engineering. Here we present these concepts to improve the requirements engineering phase.

In order to orient the paper it is necessary to have a definition of the scope of requirements engineering. Definition- Requirements engineering is the branch of systems engineering concerned with the real-world goals for, services provided by, and constraints on a large and complex software-intensive system. Requirements engineering is concerned with the relationship of these factors to precise specifications of system behaviour, and to their evolution over time and across system families." RE is the process of discovering and managing the purpose of the system for which it was projected. Poor user requirements increase risk of missing opportunity of meeting user's goals. Requirements Elicitation, Requirements Analysis & Negotiation, Requirements Documentation, Requirements Verification & Validation, and Requirements Management are five different phases of RE process. Conflicts between the views, perceptions, and goals of the stakeholders involved are highlighted by requirement elicitation process. Different viewpoints for requirement engineers are defined by the distribution of perspectives. On the bases of these view points and problems in the field of RE different frameworks are defined by researchers.

Before devoting increased effort and resources to requirements engineering it is essential for certain preconditions to be satisfied otherwise it will be dissipated by a generally disorganised development process. In other words it is important that developers do not run before they can walk!

The first step in the RE process is the elicitation of requirements. The important goals of requirements elicitation is to find out what problems need to be solved. It is defined as "the process of identifying needs and bridging the disparities among the involved communities for the purpose of defining and distilling requirements to meet the constraints of these communities". It is served as a front end to systems development. Requirements elicitation involves social, communicative issues as well as technical issues. With requirements elicitation, requirements analysts, developers, sponsors, funders, and end users are involved. The elicitation process is further decomposed as follows:

1. Identify the sources of requirements. Sources may be an end user, an interfacing system, or environmental factors.

2. Gather the wish list for each relevant party. Originally wish list contains ambiguities, inconsistencies, infeasible requirements, and untestable requirements. Also it is incomplete.

3. The wish list for each relevant party is documented and refined. All important activities and data are mentioned in wish list. The data is repeatedly analysed until it is consistent. Data in list is at high level. It is stated in user-specific terms.

4. The wish lists are integrated across various relevant parties. The conflicts between the viewpoints are resolved. One more important part of this process is consistency checking. Feasibility for wish lists or goals is checked.

5. The non-functional requirements like performance and reliability are determined. And these are stated in requirements document. These activities are common to most of the process definitions for requirements elicitation found in the literature.

The resulting product from the elicitation phase is a subset of the goals from the various parties which describe a number of possible solutions.

Existing requirements elicitation approaches have proven insufficient to record complete, consistent, and correct requirements. Studies conducted have shown that 40% of defects in software projects are due to incorrect recorded requirements. Eliciting clear, complete, and correct requirements is still a challenge and a difficult undertaking in requirements engineering. Crucial information related to the requirements is often ignored, and partially or not recorded at all during requirements elicitation. Engineers documenting the requirements may misinterpret, partially document, or omit important statements. Most of the existing requirements elicitation approaches are clearly lacking capabilities to support gathering complete and detailed requirements in a natural flow.

Our project proposes an open and inclusive method for requirements elicitation using social networks and collaborative filtering. An inherent feature in existing requirements elicitation methods is that they depend on a small number of experts such as the requirements engineers or the project team. These experts become a bottleneck in large-scale software projects where they have to process many requirements from many stakeholders. To remove the bottleneck, this work will shift the emphasis from requirements elicitation involving only the experts to a collaborative approach in which all stakeholders have a say.

2. RESEARCH METHODOLOGY

The study in the previous chapter highlighted the need for methods to identify and prioritise stakeholders and their requirements in large-scale software projects. It is necessary to show that a method that uses social networks and collaborative filtering can be used to elicit requirements in large-scale software projects.

The methodology in this work is divided into four parts:

- Select a software project and study the project to identify lists of stakeholders and requirements.
- Build social network which will help to identify and prioritise stakeholders.
- Develop a method that uses collaborative filtering to identify and prioritise requirements.
- Develop a software tool that supports the above method; apply it to real projects by having practitioners use it in their projects.

Stakeholders are asked to recommend other stakeholders. A social network is built with stakeholders as nodes and their recommendations as links. Various social network measures are applied to prioritise stakeholders. Social network analysis is the application of methods to understand the relationships among actors, and on the patterns and implications of the relationships. A social network is a structure that consists of actors and the relation(s) defined on them. Actors are discrete individuals, corporate, or collective social units, such as employees within a department, departments within a corporation, and private companies in a city. These actors are linked to one another by relational or social ties. The kind of ties is extensive, such as

evaluation of one person by another (e.g., friendship, liking, or respect), transfers of material resources (e.g., business transaction), association or affiliation (e.g., belonging to the same social club), and formal relations (e.g., authority).

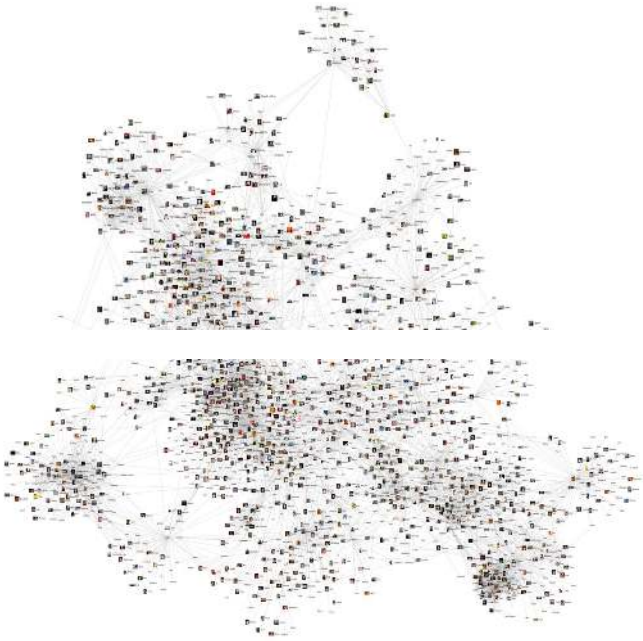


Figure 1: Large social network

A social network is often depicted as a graph in which the actors are represented as nodes and the relationships among the pairs of actors are represented by lines linking the corresponding nodes. Stakeholders selected initially are asked to assign a value or rating to each tie. These responses produce valued relations. Each of these stakeholders is asked to nominate others. Then, new stakeholders who are not part of the original list are similarly asked to nominate others. As the process continues, the group of stakeholders builds up like a snowball rolled down a hill, which results in a well-connected network.

3. STAKEHOLDER ANALYSIS

Requirements elicitation involves a wide range of people. These people include customers or clients who pay for the system, users who interact with the system to get their work done, developers who design, construct, and maintain the system, and policy makers who impose rules on the development and operation of the software system. They have diverse backgrounds, expertise, interests, and personal goals. Stakeholder analysis means identification and prioritisation of the individuals and groups that can influence or be influenced by the software project. These stakeholders are the source of requirements during requirements elicitation. Customers are not always the end-users of the product; end-users will make or break the software, affect or be affected by it, and decide on its usefulness. Modern requirements elicitation broadens its scope to involve stakeholders – any individuals or groups that can influence or be influenced by the success or failure of a software project.

The task of identifying stakeholders is far from straightforward. Information about stakeholders is not readily available and it is difficult to arrive at a complete list of stakeholders. Omitting stakeholders is reported as the most common mistake in development efforts. The majority of

developers face problems finding the right stakeholders with adequate time, interest, and knowledge for the project. All too often developers omit stakeholders and the omission significantly impacts project success.

2.1. Social Network Measures

The centrality of actors in their social networks is of great interest to social network analysts. Actors that are more central have a more favourable position in the network. For example, in a friendship network, an actor who is connected to many actors in the network is popular. In a business contact network, an actor that sits in between clusters of networks has high influence on the information that passes between the clusters. Degree centrality considers only direct connections between a node and its immediate nodes. Other social network measures, such as closeness centrality and betweenness centrality, consider the overall structure of the network. Closeness centrality considers an actor to have a central position if the distance of an actor to all others in the network is short. Betweenness centrality measures the extent to which a node sits between other nodes in the network. Our project is going to identify and prioritise stakeholders using the following steps (Figure 2).

Identify Roles Find stakeholder Get Recommendations Build network Prioritise stakeholders

Figure 2: Social networking steps

4. COLLABORATIVE FILTERING

With the Internet, the opinions of thousands can now be considered. Opinions from a large community of users can be gathered and filtered for information and patterns, a process known as collaborative filtering. The users' opinions on an item are expressed as ratings. Collaborative filtering mines patterns within these ratings in order to forecast each user's preference for unrated items. Collaborative filtering is used to support decision-making involving large amounts of information. For example, Amazon uses collaborative filtering to recommend books and Movie Lens uses it to recommend movies.

Collaborative filtering recommender systems produce recommendations for a given user on one or more items. In collaborative filtering, users are the individuals who provide ratings to a system and receive recommendations from the system. A rating is a numerical representation of a user's preference for an item. Profile is the set of ratings that a particular user has provided to the system.

The requirements are clustered using kNN algorithm and only relevant requirements are recommended to stakeholders to avoid information overload.

To summarize, collaborative filtering is a technique used to filter large sets of data for information and patterns. By collecting taste information from many users' predictions about the interests of a user are made. The underlying assumption is that users who have had similar taste in the past will share similar taste in the future.

The ratings from the stakeholders' profiles and the priority of the stakeholders and their roles are used to prioritise requirements. To calculate the importance of a requirement in a project, the influence of the stakeholder's role in the project is determined, and then the influence of the stakeholders in their roles is determined.
Stakeholder recommender model will contain following steps:-

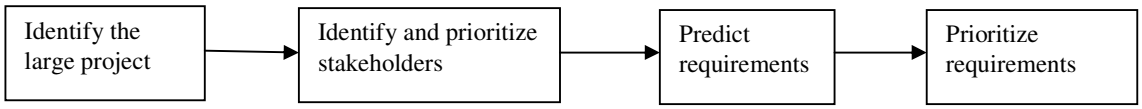


Figure 3: Stakeholder Recommender Model Steps

1. **Identify the large project:-**

There are a number of reasons and methods to size a project, leading to different views on what constitutes large-scale. The earliest measure of project size is lines of code (LOC), which counts the number of non-blank, non-comment lines in the text of a software program's source code. Another popular measure is function points (FP). Other measures for project size include man-hour, budget, and duration. In requirements elicitation, the number of stakeholders is used to size a project. From the requirements engineering perspective, a large software system is defined as "a software system that has a large and diversified community of users, and entails a variety of human, organisational, and automated activities, and various, sometimes conflicting, aspects of different parts of its environment". For this project, a large-scale software project is thus defined as a software project with a large and diverse community of stakeholders with different needs.

2. **Identify and prioritize stakeholders :-**

Stakeholders are asked to recommend other stakeholders to identify and prioritise stakeholders and builds social network which consists of stakeholders as nodes and their recommendations as links various social network measures can be used to prioritise stakeholders. Social network analysis is the application of methods to understand the relationships among actors, and on the patterns and implications of the relationships.

3. **Predict requirements:-**

Based on the stakeholders' profile, this step uses collaborative filtering to predict other requirements that each stakeholder needs or actively does not want.

K-nearest neighbour (KNN) algorithm

K Nearest Neighbour is very simple algorithm to understand. And it works incredibly well in practice. It is versatile. Its applications range from vision to proteins to computational geometry to graphs and so on. KNN is one of the top ten data mining algorithms.

KNN is a non parametric lazy learning algorithm. That is a pretty concise statement. A non parametric technique means it does not make assumptions on underlying data distribution. (eg. gaussian mixtures, linearly separable etc).

It is also a lazy algorithm. It does not use training data points for generalization. There is no explicit training phase or it is very minimal. The training phase is fast. The training data is needed during the testing phase.

The working is in contrast to other techniques like SVM where all non support vectors can be discarded without any problem. Lazy algorithms like KNN makes decision based on the entire training data set.
There is a non existent or minimal training phase but a costly testing phase - cost in terms of time and memory. In the worst case, all data points might take point in decision more time is needed. As all training data needs to be stored, memory required is more.

It is assumed by KNN that data is in a feature space. Data points are in a metric space. Data can be scalar data or multidimensional vector data.

Training data consists of vectors and class label associated with each vector. it will be either positive or negative classes. KNN works equally well with arbitrary number of classes.

Given is a single number "k". This number decides how many neighbors (where neighbor is defined based on the distance metric) influence the classification. This is usually a odd number if the number of classes is 2. If k=1, then the algorithm is simply called the nearest neighbor algorithm.

The k-nearest neighbour algorithm is sensitive to the local structure of the data. The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct.
Following are the important steps involved in the K-Means clustering algorithm.

1. Initially ‘k’ different clusters are created. And sample set is distributed randomly between the ‘k’ different clusters.

2. The distance between each of the sample within a given cluster to their respective cluster centroid is measured.

3. The cluster which records the shortest distance from a sample to the cluster (k ¢) centroid is find out and samples are moved to that cluster (k ¢).

A user decides number of clusters k for cluster analysis. The parameter k takes definite integer values with the lower bound of 1 and an upper bound that equals the total number of samples. The algorithm can be repeated number of times which will give an optimal clustering solution, every time starting with a random set of initial clusters.

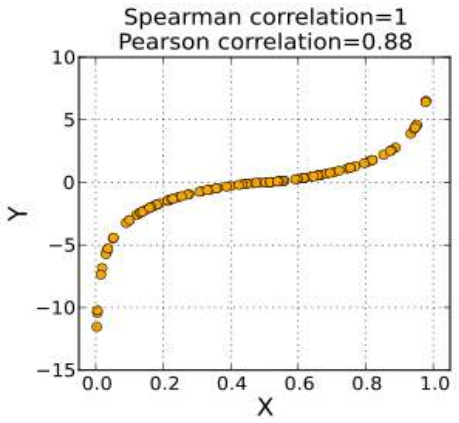
To measure similarity Spearman's rank correlation coefficient is used. It is a non-parametric measure of statistical dependence between two variables. It assesses how well the relationship between two variables can be described using a monotonic function. If there are no repeated data values, a perfect Spearman correlation of +1 or −1 occurs when each of the variables is a perfect monotone function of the other.

Correlation between two sequences of values is measured using Spearman Rank Correlation. The two sequences are ranked separately and then differences in rank are calculated at each position - i. The distance between sequences X = (X1, X2, etc.) and Y = (Y1, Y2, etc.) is computed using the formula:

$$1 - \frac{6 \sum_{i=1}^{n} (rank(X_i) - rank(Y_i))^2}{n(n^2 - 1)}$$

Here X_i = i th value of sequence X.

Y_i = i th value of sequence Y.



A Spearman correlation of 1 results when the two variables being compared are monotonically related, even if their relationship is not linear. In contrast, this does not give a perfect Pearson correlation.

4. Prioritize requirements

Once requirements are identified, there is need to prioritise them. Projects often have more requirements than time, resource, and budget allow for. A function can always be added and the user interface enhanced. Some requirements are critical for the success of the software system. Hence, requirements should be prioritised so that the ones that are most likely to achieve customer satisfaction can be selected for implementation. There are number of methods to prioritise requirements, such as numeral assignment technique, the cost-value approach, the value-oriented prioritisation method, the pair wise comparison approach, the Analytic Hierarchy Process (AHP) Value-oriented prioritisation method (VOP) , 100-point test, Hierarchical cumulative voting (HCV), the requirements triage method, the win-win approach, minimal spanning tree, the binary search tree (BST) . For projects with many requirements, we propose a method which uses data mining and machine learning techniques to support requirements prioritisation. Requirements are organized into different categories using clustering technique. The clusters are then prioritized. By automatically clustering the requirements into different categories, this method will reduce the number of manual prioritisations required from the requirements engineers.

4. ANALYSIS

Prioritisation is a very important activity in requirements engineering because it lays the foundation for release planning. However, it is also a difficult task since it requires domain knowledge and estimation skills in order to be successful. The inability to estimate

implementation effort and predict customer value may be one of the reasons why organisations use ad hoc methods when prioritising requirements. In practice, it is common that a larger number of requirements need to be prioritised. When the number of requirements grow, it is hard to get an overview. Therefore, visualisation is very important in order to share information.

Following are some of the measures to analyse the results:-

- The average time to conclude the prioritisations can be calculated.
- The ease of use can be considered.
- Consistency Ratio-The consistency ratio (CR) describes the amount of judgement errors that is imposed.

5. CONCLUSIONS

In large scale software projects, the requirement elicitation faces problems like information overload, inadequate stakeholder input, and biased prioritization of requirements. Stakeholder is an individual or a group who can influence or be influenced by the success or failure of a project. Stakeholders have to be identified as they are the source of requirements. Existing methods in stakeholder analysis are likely to either omit stakeholder roles or return "non-stakeholders". This approach uses social networks and collaborative filtering for requirement elicitation for large scale projects to identify and prioritize requirements. It elicits requirements from the stakeholders identified by social network. A highly complete set of requirements is identified compared to the existing method used, requiring less time from the requirements engineers and the stakeholders. The model also prioritizes the requirements accurately. It handles information overload by drawing stakeholders' attention to only the relevant requirements that they are unaware of. Its elicitation method, which provides stakeholders with a predefined list of requirements as well as allowing them to add new requirements, is rated by stakeholders as low difficulty and requiring little effort.

ACKNOWLEDGEMENTS

I take this opportunity to express my sincere gratitude and respect to my guide Prof. Sheetal P. Girase, Asst. Prof. Maharashtra Academy of Engineering & Educational Research's, Maharashtra Institute of Technology, Pune. who gave me assistance with their experienced knowledge and whatever required at all stages of my project. She has taken pain to go through the project and make necessary correction as and when needed.

REFERENCES

[1] Soo Ling Lim, and Anthony Finkelstein. "StakeRare: Using Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation", IEEE Transactions On Software Engineering 2011

[2] Lim, S.L., D. Quercia, and A. Finkelstein. "StakeNet: using social networks to analyze the stakeholders of large-scale software projects", in Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1. 2010.

[3] Castro-Herrera, C. J. Cleland-Huang, and B. Mobasher Enhancing stakeholder profile to improve recommendation in online requirements elicitation Proceedings of the 17th IEEE International Conference on Requirements Engineering. 2009: IEEE Computer Society. p. 37-46.

[4] Lim, S.L., "Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation. PhD thesis, 2010,University of New South Wales."

[5] Herrmann, A., M. Danev Requirement prioritization based on benefit and cost prediction- an agenda for future research Proceedings 16th IEEE International Conference on Requirement Engineering 2008.
[6] Azar J, R.K. Smith, D. Cordes, Value oriented requirements prioritization in small development organization,IEEE Software 2007
[7] Davis, A., O. Dieste, A. Hickey, N. Juristo, and A.M. Moreno. "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review", in Proceedings of the 14th IEEE International Conference on Requirements Engineering. 2006. p.
[8] Charette. Why software fails, IEEE Spectrum, 2005
[9] Alexander, I. and S. Robertson, Understanding project sociology by modeling stakeholders. IEEE Software, 2004. 21(1).
[10] Zhang, Q., Nishimura, T., "A Method of Evaluation for Scaling in the Analytical Hierarchy Process", Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Vol. 3, pp. 1888-1893, 1996.
[11] D. Leffingwell and D. Widrig, Managing Software Requirements – A Unified Approach, Addison Wesley, 2003.