A Recommender System for Requirements Elicitation in Large-Scale Software Projects

Carlos Castro-Herrera[+], Chuan Duan[+], Jane Cleland-Huang[+], Bamshad Mobasher[*]

Systems and Requirements Engineering Center [+], Center for Web Intelligence *
DePaul University
243 S. Wabash Ave, Chicago, IL 60604
{ccastroh, duanchuan, jhuang, mobasher}@cs.depaul.edu

ABSTRACT

In large and complex software projects, the knowledge needed to elicit requirements and specify the functional and behavioral properties can be dispersed across many thousands of stakeholders. Unfortunately traditional requirements engineering techniques, which were primarily designed to support face-to-face meetings, do not scale well to handle the needs of larger projects. We therefore propose a semi-automated requirements elicitation framework which uses data-mining techniques and recommender system technologies to facilitate stakeholder collaboration in a large-scale, distributed project. Our proposed recommender model is a hybrid one designed to manage the placement of stakeholders into highly focused discussion forums, where they can work collaboratively to generate requirements. In our approach, statements of need are first gathered from the project stakeholders; unsupervised clustering techniques are then used to identify cohesive and finely-grained themes and a users' profile is constructed according to the interests of the stakeholders in each of these themes. This profile feeds information to a collaborative recommender, which predicts stakeholders' interests in additional forums. The validity and effectiveness of the proposed recommendation framework is evaluated through a series of experiments using feature requests from three software systems.

Categories and Subject Descriptors

D.2.1 [**Requirements/Specifications**]: Elicitation methods (e.g., rapid prototyping, interviews, JAD), Tools; H.3.3 **[Information Search and Retrieval]:** Clustering, Information filtering.

General Terms

Algorithms, Measurement, Experimentation, Human Factors.

Keywords

Collaborative recommender systems, requirements clustering, large-scale software engineering.

1. INTRODUCTION

The software requirements process, including the tasks of eliciting, analyzing, and specifying the functional and behavioral properties of a system, represents one of the most critical phases of the software development lifecycle. Software requirements serve as a contractually binding specification, and guide the design, implementation, and testing efforts. In most projects, the requirements are proactively elicited from a broadly representative group of stakeholders through a carefully coordinated series of informal interviews, surveys, ethnographic studies, and organized brainstorming sessions. It is imperative to identify and include all of the relevant stakeholders in appropriate discussions, as failure to do so can lead to failed projects in which certain viewpoints are never explored, and entire groups of stakeholders are disenfranchised from the process.

Unfortunately, two increasingly prevalent software engineering practices have undermined the effectiveness of traditional requirements elicitation strategies and techniques. First, the growth of the global IT marketplace has meant that customers and developers are often geographically distributed, and in-person requirements meetings are not feasible on a regular basis. Secondly, the growth in size and complexity of software systems and the associated increase in the number of stakeholders, introduces significant problems in managing and coordinating the human-intensive requirements elicitation process in large projects. Furthermore, a recent report issued by the Software Engineering Institute entitled "Ultra-Large-Scale Systems: The Future Challenge of Software Engineering," outlines the emerging concept of Ultra-Large-Scale (ULS) systems for which the size and complexity is increased exponentially from most of the systems we build today [12]. It is therefore imperative to develop new techniques and tools that scale up to support effective requirements processes for large scale systems.

In previous work this problem was addressed by introducing a novel framework for facilitating the requirements elicitation process [4,5]. In this framework, statements of need are first gathered from the project stakeholders; unsupervised clustering techniques are then used to identify cohesive themes and user profiles are constructed according to the interests of the stakeholders in each of these themes. The profiles are used as input to a collaborative recommender, which recommends additional themes to stakeholders.

The framework, which is depicted in Figure 1, represents a philosophical shift from in-person, face-to-face activities to a computer supported environment that supports many of the new paradigms of distributed and large-scale development projects.

The need for this type of automated support is illustrated through examining the requirements features of open source projects. For example, in SugarCRM, a large open-source customer management system, users create new feature requests by browsing through a list of existing threads and determining whether to submit to an existing thread or to create a new one. An analysis of the resulting threads showed that many users created

SAC'09, March 8-12, 2009, Honolulu, Hawaii, U.S.A.

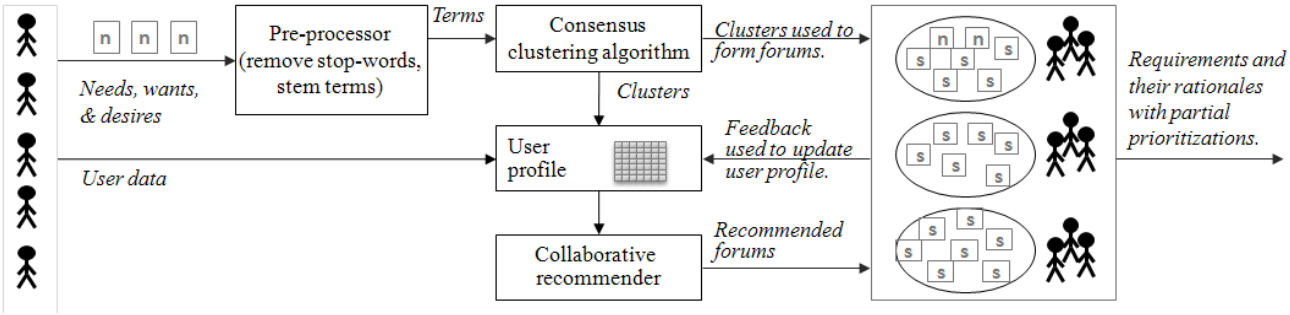


Figure 1. A Recommender Model for the Requirements Domain.

either a new thread for each feature request, or placed requests into one or two mega-threads. Neither of these approaches is ideal in an online requirements gathering tool, as the resulting threads can be redundant, isolated, and often too large to effectively support collaborative requirements activities.

This paper describes the architecture, algorithms, and processes of our proposed requirements framework and associated recommender system, motivating its design by examining the rather unique characteristics of the requirements domain in which it operates. The framework includes several primary components. First, a web-based collection tool is used to collect general comments and statements of need from participating stakeholders. Secondly, our framework utilizes data mining techniques to analyze these stakeholders' needs, identify dominant and cross-cutting topics, and dynamically generate a related set of highly focused and distinct forums. In order to manage the timely placement of stakeholders into forums, an initial user profile, which captures the stakeholders' interests, is generated. This profile drives initial forum recommendations based purely on each stakeholder's contributed needs, and then creates additional collaborative recommendations according to the interests of a neighborhood of similar stakeholders. Finally, a groupware environment facilitates distributed stakeholders working together in each forum to collaboratively transform their needs into sets of carefully articulated requirements. This is illustrated in Figure 2.

The potential benefits of this approach are significant and include the ability to capture a more complete set of requirements, explore options in greater depth, consider more perspectives, increase stakeholder buy-in to a project, emerge tradeoffs and conflicts earlier in the software development lifecycle, and scale up the process to manage many thousands of stakeholders.

The remainder of this paper is laid out as follows. Section 2 provides a brief analysis of the current state of practice and research in this area. Section 3 introduces the architecture of the proposed framework, while sections 4 and 5 describe the initial forum generation and use of recommender systems within the domain. Section 6 describes the experimental evaluation of the framework, while section 7 concludes with an analysis of the results and suggestions for future work.

2. RELATED WORK AND TOOLS

Many industrial requirements tools claim to support the kind of collaborative process that is needed to facilitate large-scale projects. For example, DOORS™ requirements management tool, which is one of the industry leaders in requirements software, is advertised as providing a collaborative requirements management environment; however, like other similar tools it simply provides a multi-user front-end for entering, updating, and viewing requirements, and for notifying stakeholders when changes of potential interest occur. Although multiple users can work together to construct a Software Requirements Specification the tool provides no real support for managing users' interactions or helping them to collaboratively author and prioritize requirements.

Several elicitation techniques are designed to support stakeholder collaboration [6]. For example, the Win-Win process provides a collaborative environment in which stakeholders brainstorm their needs, identify and prioritize stakeholders' win-conditions, identify issues, constraints, and options, and then finally negotiate agreements. IBIS enables stakeholders to collaboratively verbalize and explore key issues and related arguments in order to find and articulate solutions. The well-known Joint Application Development (JAD) gathers a broad range of stakeholders together in highly structured and focused meetings to gather information and specify requirements.

However, the centralized models adopted by these techniques can be challenging and time-consuming to implement in large projects in which knowledge is broadly distributed across multiple departmental or organizational boundaries. Tools are therefore needed to support a more decentralized process through automatically identifying stakeholders with common interests, even as new concepts dynamically emerge during the requirements gathering process, and also by placing stakeholders into non-redundant, focused, carefully managed, decentralized working groups where they can explore their needs and generate common requirements. Recommender systems provide potential solutions for accomplishing many of these goals.

3. RECOMMENDER ARCHITECTURE

There are several notable characteristics that differentiate the context of requirements recommender systems from those of more typical domains such as e-commerce websites or online news services, and these differences drive the unique design of our recommender system. The first characteristic relates to the nature of the forums that our proposed system must recommend. Each of these forums is represented by a set of underlying stakeholders' needs. Although other recommenders, such as those for news articles and jokes, are also designed to work with text-based items, our recommender must recommend a cluster of such items. Therefore the ultimate quality of the recommendation is

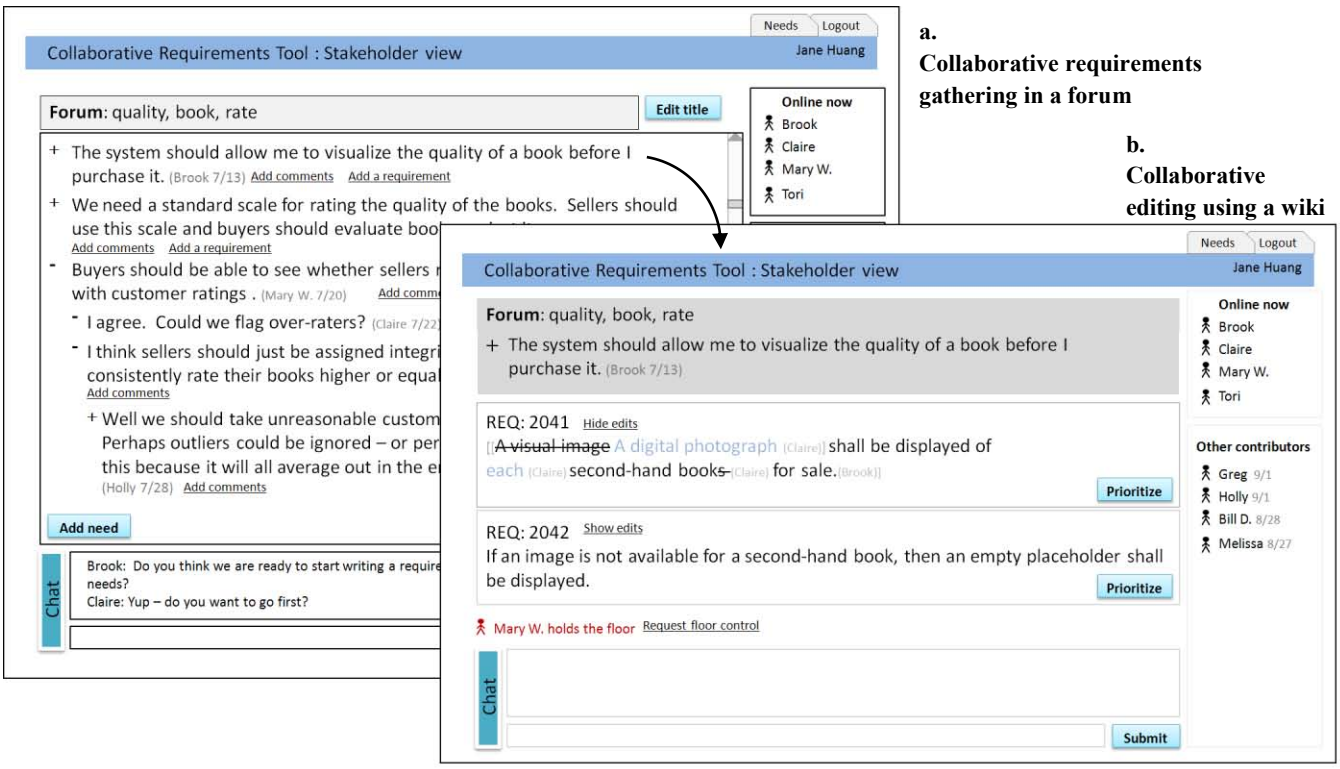


Figure 2. Recommendable forums in the requirements domain.

constrained by the quality of the generated clusters. Furthermore, because discussions in each forum will be centered around the underlying cluster of needs, and stakeholders will scrutinize the statements of need in each cluster, the granularity of the clusters must be significantly smaller than the granularity produced in most clustering domains, and the quality of the clusters must be sufficiently high as to build human confidence that the tool has performed an adequate clustering task.

Second, whereas in many recommender systems, the objective is to generate a few highly relevant recommendations; in the requirements domain the objective is to recommend all forums for which a stakeholder might hold an interest and therefore might need to contribute requirements. From a software engineering perspective, failure to include an important stakeholder in a relevant discussion could result in the construction of a suboptimal system that may fail to explore all pertinent viewpoints or to meet the needs of certain groups of stakeholders.

Finally, as users in this domain are often employees who are expected to actively participate in the requirements gathering process, a relatively high level of feedback can be anticipated from recommendations made. This feedback can be used to enhance and improve both the quality of the recommendable items (i.e. the forums), as well as the accuracy of future recommendations.

The overall architecture of our proposed framework is depicted in Figure 1. The recommender model includes a clustering component for constructing forums, a classification component for placing stakeholders into initial forums, a computer supported collaborative work (CSCW) component for hosting the forums, and a collaborative recommender for helping to place stakeholders into additional forums of potential interest. Each of these components is discussed in the following sections.

4. FORUM CREATION

In our proposed requirements framework, unsupervised clustering techniques are used to identify underlying themes and to construct a set of related forums. There are numerous algorithms that can be used to perform this task [8], including well known approaches such as hierarchical clustering [14], spherical K-means [7], and model-based probabilistic clustering. However, the quality of the generated clusters is highly sensitive to initial parameter configurations, which means that performance may differ from one clustering to another, and there are no guarantees that any individual clustering is an optimal one. To overcome this difficulty, a consensus clustering algorithm, which achieves a more robust clustering of needs, was adopted [16]. Our preliminary experiments have shown that this approach consistently generates clusters that are slightly better than the highest quality individual clustering because clustering mistakes made by any individual run of the algorithm tend to be outvoted by other members of the ensemble [8]. This is significant because it means that the process of forum creation is more robust, and significantly less likely to return a poor result. The consensus clustering algorithm includes the following steps:

a) Pre-processing. Stakeholders' needs are initially parsed to stem words to their root forms, remove common (stop) terms, and compute the *tf-idf* (term frequency, inverse document frequency) values for all terms. Intuitively, *tf-idf* weights terms more highly if they occur less frequently and are therefore expected to be more useful in expressing unique concepts in the domain. Each need is then represented as a vector, $x_i = \left(f_{i,j}\right)_{j=1}^{W}$ where $f_{i,j}$ is the weight

associated with term t_j, and W represents the total number of terms.

b) Granularity determination. The weighted need vectors χ are then used to determine the optimal number of clusters K heuristically using cover coefficient (CC) presented by Can [3]. CC estimates the optimal cluster number as a sum of degree representing the extent to which each vector differentiates itself from other vectors. Formally, it's defined as:

$$K = \sum_i \frac{1}{|x_i|} \sum_{j=1}^{W} \frac{f_{i,j}^2}{N_j}$$

where $|x_i|$ is the length of x_i, while N_j is the total number of occurrences of t_j. This metric has been demonstrated to be effective in Can's work and validated in our earlier work on requirements clustering [8].

c) Ensemble generation. A clustering ensemble of size R is generated through performing a sub-sampling clustering of χ. In each run of subsampling, a proportion α of the whole dataset is randomly extracted and then partitioned into K clusters using spherical K-means, followed by the classification of each remaining need into its most closely related cluster. Based on extensive experiments applied to several TREC document data sets, we determined that a quality ensemble for data sets with several thousands of data points could be generated by setting R to 100 and α to 0.5.

d) Construction of a co-association matrix. A co-association matrix $C := (c_{i,j})_{N \times N}$ was constructed where for each element $c_{i,j} = n_{ij}/R$, and where n_{ij} represents the number of times the artifact pair x_i, x_j is assigned to the same cluster over the entire ensemble of partitionings. The underlying assumption of using a co-association matrix is that vectors belonging to a "real" cluster are very likely to appear in the same cluster across multiple partitionings of the data.

e) Generation of consensus clustering. The average-link hierarchical agglomerative clustering algorithm (AHC) was applied to generate the final partitioning [11] from the co-association matrix, where the values in the matrix are used to represent the proximity between pairs of artifacts. Several choices exist for clustering a co-association matrix M, such as alternate variants of AHC, spectral clustering, and graph partitioning algorithms; however the average-link algorithm was chosen as it was demonstrated in our experiments to be very stable across ensembles of various sizes and characteristics.

In our recommender model, the results of the clustering were used to automatically populate an initial user profile, so that a user was assumed to be interested in all forums in which his or her needs had been assigned. In a non-traditional sense, these initial assignments of needs to forums could be considered recommendations to participate in the forum.

5. COLLABORATIVE RECOMMENDER

Placing stakeholders into forums based only on their prior contributions and stated interests misses the opportunity for more proactive recommendations. Collaborative recommendations can be made to the user to introduce serendipity into the process, through predicting items based on the known interests and opinions of other users [13]. These systems generate recommendations by identifying neighborhoods of users with similar interests, and then using these neighborhoods to predict the interest that a particular user might have in an item that he or she has not yet rated. Collaborative recommenders could be particularly useful in the requirements process, as they facilitate the cross-pollination of ideas and help to prevent the problem of stakeholders missing important discussions.

In our framework we use the standard user-based collaborative filtering approach, based on the k-nearest-neighbor algorithm, to predict the level of interest that a user might have in a forum for which the stakeholder has registered no prior interest, given a set of ratings (ranged over r, with $\bar{r}$ denoting an average rating). Specifically, the predicted interest of a user u on a forum f is computed as

$$pred(u, f) = \bar{r}_u + \frac{\sum_{n \in nbr(u)} userSim(u, n) \cdot (r_{ni} - \bar{r}_n)}{\sum_{n \in nbr(u)} userSim(u, n)}$$

where $n \in nbr(u)$ represents that n is a neighbor of u [13]. In a more typical recommender system, user ratings are obtained from direct or indirect user feedback such as user ratings of certain items, or through analyzing which items a user clicked on. However in our framework, these initial ratings are estimated by the membership scores of the stakeholders in the forums. The membership scores are represented by a matrix $M := (m_{i,j})_{S \times F}$, where S is the number of stakeholders and F the number of forums. This matrix is obtained from the product of $A := (a_{i,j})_{S \times N}$ times $B := (b_{i,j})_{N \times F}$ where A is the matrix of stakeholders by needs, B is the matrix of needs by forums, and N is the number of needs. Each $a_{i,j}$ indicates that a stakeholder i has expressed interest in need j; and each $b_{i,j}$ indicates that need i has been placed in forum j and the entry value is the distance of the need from the centroid of the forum. The resulting product of $A \times B$ is then normalized over the rows to obtain matrix M. This initial M can be further refined as the stakeholders provide feedback on the membership scores. Since recommendations can only meaningfully be made to those stakeholders who have sufficient interests registered in their user profile to support neighbor identification, we made the design choice to only compute prediction scores for stakeholders that belong to at least three or more forums.

Intuitively, the algorithm computes the average of ratings that the neighbors have given a forum, while taking into consideration the similarity of the neighbors and the fact that some users are more optimistic than others. User similarity, $userSim(u, n)$, between user u and the neighbor n can be computed in various ways. Our prototype used the Pearson correlation formula, in which $CR_{u,n}$ denotes the set of corated items between u and n [2]. It is computed as follows:

$$userSim(u, n) = \frac{\sum_{f \in CR_{u,n}} (r_{uf} - \bar{r}_u)(r_{nf} - \bar{r}_n)}{\sqrt{\sum_{f \in CR_{u,n}} (r_{uf} - \bar{r}_u)^2} \sqrt{\sum_{f \in CR_{u,n}} (r_{nf} - \bar{r}_n)^2}}$$

Table 1. Coupling and cohesion metrics for the clustered data sets

Data set	Student			SugarCRM			SecondLife		
Granularity (K)	29			60			50		
Metric	Cohesion		Coupling	Cohesion		Coupling	Cohesion		Coupling
	CH1	CH2	CP	CH1	CH2	CP	CH1	CH2	CP
Answer set	145.04	59.20	142.65	365.54	144.47	461.62	n/a	n/a	n/a
Dynamic clusters	164.03	75.78	124.04	424.38	183.04	372.68	471.17	181.82	415.34
Random clusters	114.10	35.92	180.27	283.30	81.05	554.53	289.62	67.45	665.38

Table 2. Sample needs in two SecondLife clusters

Cluster theme	A subset of needs taken from three clusters
permiss, object, item, transfer, script	I cannot edit objects that have been created by others when permission has been granted. we then took back permission and regranted it worked for the time I was logged int. Upon going back this morning its not working again. Cannot modify others objects despite permission
	The current way that permissions are handled on prims is very cumbersome. ESPECIALLY WHEN BUILDING AS A TEAM. I would like to propose the following: Prim Permissions - make this more intuitive!
ban, tp, list, land, estat	This feature proposes, that Linden Lab expand the current land ban list, block them from purchasing the land. This is a proactive measure to ensure residents, that land will not be purchased by landbots, griefer neighbors or otherwise undesirable persons. New Feature - Parcel - Ban List - option to block land buyers
	System generate messages because someone who owns a massive ammout of micro-plots has put me on their ban list on each one of them in the sim Ellingson where I own land. I do not enter their land, but the messages keep ;spamming; me, and I suspect it is done as a deliberate exploit of the system because I cannot turn them off.

This correlation metric generates numbers between 1 and -1, where users in perfect agreement score 1, and users in perfect disagreement score -1. As part of our empirical experiments we tested several alternatives to compute the similarities which will be discussed in further detail in section 6.2.2.

6. EXPERIMENTAL EVALUATION

The proposed requirements recommender model was evaluated through a series of experiments. The first one was designed to measure the extent to which the clustering algorithms established a quality set of forums and made adequate initial placements of stakeholders into forums, while the second series of experiments were designed to evaluate and compare the efficacy of several different collaborative recommender models. Before describing these experiments, the three datasets used throughout the remainder of this paper are introduced.

Student represents a small collection of 366 feature requests created by 36 graduate level students for an Amazon-like student web-portal system. A reference set, which created an "ideal" clustering, was developed by two of the researchers in the SAREC lab. The other two datasets were mined from the feature requests of two open source projects. Sugar is comprised of 1000 feature requests mined from SugarCRM, an open source customer relationship management system that supports campaign management, email marketing, lead management, marketing analysis, forecasting, quote management, case management and many other features. The feature requests were contributed by 523 different stakeholders over a two year period, and distributed across 309 threads. For the Sugar data, a reference set was constructed through reviewing and modifying the natural discussion threads created by the SugarCRM users. Modifications included merging singleton threads with other relevant ones, manual re-clustering of large mega-threads into smaller more cohesive ones, and then manually reassigning misfits to new

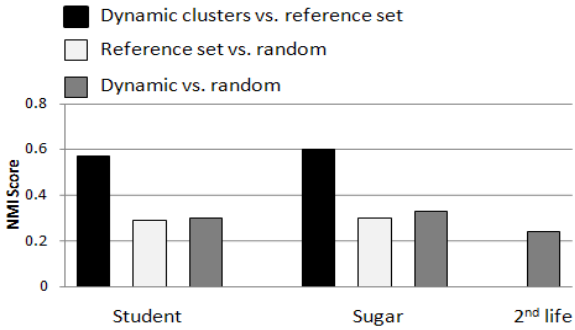


Figure 3. Similarity of clusterings measured using Normalized Mean Information (NMI).

clusters. The results were reviewed and agreed upon by a small team of DePaul researchers. Finally, the SecondLife data set represents the feature requests of Second Life users. Second life is an Internet-based virtual world game launched in 2003, developed by Linden Research, Inc. For the experiments described in this paper we used a subset of 1250 requests created by or commented on by 2120 stakeholders. There is currently no answer set available for this dataset, as the intrinsic categorization of feature requests in Second life is based upon rather general cross-cutting concerns such as *physics*, *scripting*, and *graphics*.

6.1 Forum Evaluation

In our framework, stakeholders' needs are initially organized into clusters which are used to anchor a set of discussion forums. The first experiment was designed to test the hypothesis that the quality of clusters automatically generated by the consensus clustering technique was sufficient to support the proposed recommender model. The analysis was based on the underlying assumption that high quality clusters were a necessary, albeit not

sufficient, property that would lead to satisfactory initial placements of stakeholders into forums.

Our experiments were therefore designed to measure the quality of the generated clusters using standard coupling (CP) and cohesion (CH1 and CH2) metrics described by Zhao [15]. CH1 measures the sum of similarity scores between each need and the centroid of the cluster that it has been placed in, while CH2 measures the sum of the average pairwise similarities between the needs assigned to each cluster, and weights the scores according to cluster size. Finally CP measures the degree to which the centroids of each of the clusters are dispersed from the centroid of the entire document space. CH1 and CP both range between 0 and N, where N represents the number of documents, while CH2 ranges between 0 and $N/2$. The objective of most clustering algorithms is to increase cohesion and decrease external coupling, while achieving desired levels of granularity. Based on the previously determined granularities for each dataset we compared the coupling and cohesion of the manually created answer set against the dynamically generated clusters, shown in Table 1. For control purposes we also generated a set of random clusters for each of the datasets. This was accomplished through creating k clusters, where k represented the same granularity used in the answer sets, and then randomly assigning each need to one of the k clusters. As a result, the random clusters were very evenly sized.

For both the Student and Sugar datasets, the automatically generated clusters were evaluated as more cohesive and less closely coupled than the manually created answer sets. One of the main explanations of these results is that the metrics are biased towards the automated approaches which focus on clustering around similar terms. In fact, given that the expert subjective opinion perceived the quality of the answer sets to be significantly higher than the automated clusters, these results demonstrate the well-known limitations of these standard metrics.

For the Student and Sugar datasets, for which an answer set was available, the similarity between the answer set and the generated cluster was measured using an information theoretical measure known as normalized mutual information (NMI). The assumption was that the answer set clustering provided a consistent and meaningful set of clusters to support the requirements recommender model, and that a high NMI score would indicate that many of the user defined partitions had been identified and recreated by the automated clusterer. In NMI, the level of agreement between two partitions P^a and P^b is expressed by the level of mutual information across the clusterings, computed as follows:

$$I(P^a, P^b) = \sum_{X \in P^a} \sum_{Y \in P^b} P(X,Y) \log \frac{P(X,Y)}{P(X)P(Y)}$$

$$= \sum_{i=1}^{k_a} \sum_{j=1}^{k_b} \frac{n_{ij}^{ab}}{n} \log \left(\frac{n_{ij}^{ab}}{n} \Big/ \frac{n_i^a}{n} * \frac{n_j^b}{n} \right)$$

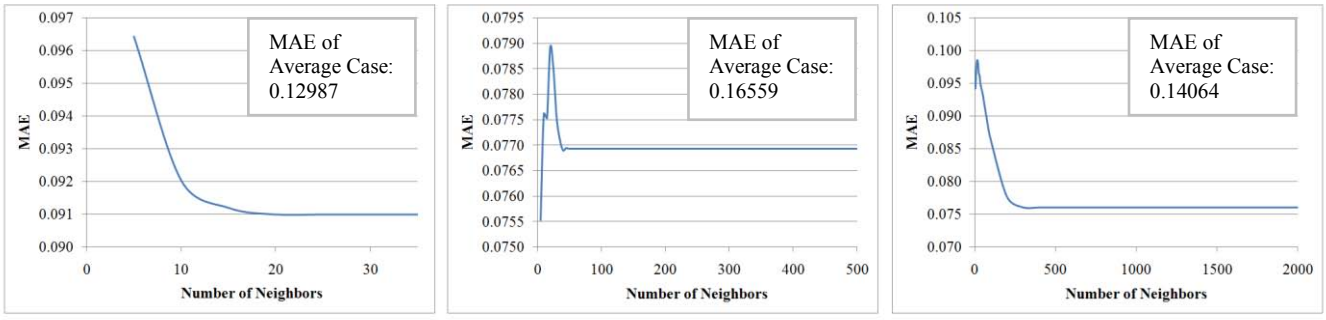
where k_a and k_b are the cluster numbers of the two partitions, n is the total number of artifacts, n_{ij}^{ab} is the number of shared artifacts in cluster a of clustering P^a and cluster b of clustering P^b, and the resulting score was normalized using an arithmetic average [14]. As depicted in Figure 3, the Student data set scored an NMI score of 0.57 and the Sugar of 0.60 suggesting that there was a significant level of agreement between the answer sets and the generated clusters, but also some clear differences. In each case, the NMI score comparing the automated clusters with the answer set, were significantly higher than those comparing the random case with the answer set.

Finally, because the clusters in our model are used to anchor discussion forums, and therefore will be scrutinized by human users, we performed a subjective assessment of several of the clusters. Table 2 provides a couple of examples showing the placement of a subset of requirements by the consensus clustering algorithm into three of the clusters generated for the SecondLife dataset. In general, most of the themes were found to be relatively well defined, although some outlying needs appeared to be misplaced, primarily because of undetected synonyms.

These results suggest that the consensus clustering technique delivered clusters which were sufficiently cohesive to support initial placement of stakeholders into forums and to provide an initial population of the user profiles.

6.2 Collaborative Recommendations

The second series of experiments were designed to determine the effectiveness of collaborative recommendations in the requirements domain. We designed an experiment to determine if the collaborative recommender was capable of recreating the initial users' profile, when known interests were systematically and temporarily removed. The underlying assumption was that the membership scores would provide enough information to form good neighborhoods of stakeholders that could feed the standard User Based Nearest Neighbor algorithm. Only stakeholders with three or more interests were included in the experiment. SecondLife had 273 stakeholders with three or more interests,



a. Student b. Sugar c. SecondLife

Figure 4. Results from leave-one-out cross validation experiment showing the effectiveness of the collaborative recommender at various sized neighborhoods, and including a comparison to an average case recommender.

while Sugar had 58 and Student only 33. In all three cases, the majority of stakeholders had fewer than ten registered interests.

6.2.1 Analysis of Basic Recommender

Two experiments were conducted. The first one was intended to gauge the overall performance of the algorithm as well as to determine the appropriate number of neighbors that were required for each one of the three datasets. The starting point of this experiment was the set of stakeholder profiles, represented as the matrix $M := (m_{i,j})_{S\times F}$, generated as the result of the consensus clustering and subsequent placement of stakeholders into forums as previously explained. A standard leave one out cross validation technique was applied, in which each $m_{i,j}$ was removed from the matrix and treated as the testing dataset. The remaining data points in the matrix were used as the training dataset to predict the value of $m_{i,j}$ using the recommender system algorithm previously explained. The differences between the predicted values and the known values were aggregated and the Mean Absolute Error MAE [10] metric was computed.

The results of this experiment are depicted in Figure 4 and show that each dataset stabilized at a different MAE and at a different number of neighbors. The Student dataset stabilized at approximately 10 neighbors with an MAE of 0.091, the Sugar dataset stabilized at approximately 40 neighbors with an MAE of 0.077 and the SecondLife dataset stabilized at approximately 200 neighbors with an MAE of 0.076. These MAE scores are quite promising, as they indicate a less than 10% error in the prediction score (which is between 0 and 1), and suggest that in fact collaborative recommenders can be usefully applied to recommend forums to stakeholders in the requirements elicitation domain. Note that at low numbers of stakeholders, the MAEs for the Sugar and the SecondLife dataset fluctuate a little erratically before they start to constantly decrease and eventually level off. This is probably due to the less predictable behavior that is product of small neighborhoods, Also notice that the X and Y axis scales are different for every graph.

6.2.2 Average Case Recommender

Predictions were also computed based on the average ratings for each item, in order to create a baseline for comparative purposes. A prediction was made to stakeholder *s* for forum *f* by computing the average ratings for all other stakeholders who had ratings for *f*. Again the differences between the predicted values and the known values were aggregated and the Mean Absolute Error MAE [10] metric was computed. The MAE scores for the average case are reported in Figure 4 for each of the three datasets. In all three cases, the nearest neighbor algorithm outperformed the average case recommender.

6.2.3 Variants of the Recommender Model

The final set of experiments was designed to compare several different techniques for computing the neighborhoods in comparison to the standard approach described in the previous section. The following variations were evaluated:

- **Terms versus forums**: The neighborhoods were computed using a lower granularity matrix $L := (l_{i,j})_{S\times T}$, where S represents the number of stakeholders and T represents the number of terms. In this matrix, each entry $l_{i,j}$ contains the number of times that a stakeholder i referenced the term j throughout all of the needs he or she expressed. Note that in the standard approach, the L matrix was used as the input to the clustering algorithm. In essence, by using this term-based matrix, the neighbors are calculated without considering the clustering and membership placement steps of the proposed framework. In this experiment, the previously used matrix of membership scores, $M := (m_{i,j})_{S\times F}$, was still used to drive the leave-one-out cross-validation experiment, however recommendations were based on the neighborhoods computed from L.

- **Similarity function**: In addition to using the basic Pearson similarity function presented in section 5, a modified version was also used. This modified version is computed as: $userSim'(u,n) = \frac{\min(|CR_{u,n}|,\gamma)}{\gamma} \times userSim(u,n)$[10]. Intuitively, this modified similarity function penalizes the similarity of two neighbors if the number of items that they have co-rated is small, indicating that there is low confidence in the correlation. For our experiments γ was set at 5; however future work will explore different values.

- **Dimensionality reduction**: An additional strategy that can be applied prior to computing the neighbors is to reduce the dimensionality of the ratings matrix. In particular, we performed Principal Component Analysis PCA [9] to reduce the dimensionality while preserving most of the variability in the data. PCA and other related techniques tend to eliminate superfluous noise that is inherent to the clusters of needs found in the requirements domain. For this experiment, the ratings matrices M and L were mean adjusted and PCA was applied to them in such a manner that 90% of the variability of the data was preserved.

The various combinations of these three factors for computing neighborhoods are depicted in Table 3. Note that the combination of the modified similarity function and PCA is not necessary, as PCA produces a fully dense matrix, so the size of the set of $CR_{u,n}$ will always be greater than γ. In order to evaluate these different approaches, the same leave one out technique that was previously

Table 3. MAE Results from various Collaborative Filtering Models

Approach number	Variations						Results - MAE		
	Data Source		Similarity Fn		PCA		Student	Sugar	SecondLife
	M (SxF)	L (SxT)	Pearson	Modified	Yes	No	Number of neighbors		
							10	40	200
0 (Standard)	1		1			1	0.092030	0.076908	0.077705
1		1	1			1	0.094023	0.069501	0.073725
2	1			1		1	0.086608	0.075608	0.072263
3		1		1		1	0.092821	0.071355	0.072814
4	1		1		1		0.084080	0.079106	0.074232
5		1	1		1		0.096214	0.072490	n/a

described was applied and the MAE computed. Approach 0, which was the approach used to determine the optimal number of neighbors, serves as a standard for comparing the other methods. The results of this experiment are reported in Table 3.

Several interesting observations can be inferred from these results. In general, applying the modified similarity function gives better results than the typical Pearson Correlation when the stakeholder × forum matrix M is used. This can be seen as the MAE values from approach 2 were consistently lower than those from experiment 0 in all three datasets. This is also true for the case when the matrix L is used (approach 1 and 3), except in the case of the Sugar data. Applying PCA did not provide consistent results. PCA in conjunction with the stakeholder × forum matrix M (approach 4) performed better for the Student and the SecondLife dataset; but it performed worse for the Sugar dataset. PCA on the finer grained stakeholder × terms (approach 5) performed worse for the Student dataset and better for the Sugar dataset. Further work is needed to determine the impact of using PCA across more varied datasets. In addition, our initial observations suggest that one of the reasons why the Sugar dataset performed differently than the other two datasets is because of the sparseness of the dataset. However, more work is needed to confirm these observations.

7. CONCLUSIONS

This paper has described a novel framework that utilizes recommender systems to facilitate the placement of stakeholders into cohesive and finely grained discussion forums. Although recommender systems have not, to the best of our knowledge, been used previously in the requirements domain, the experiments conducted in this paper, demonstrate the feasibility of the approach. Nevertheless future work is needed to investigate whether the approaches explored in this paper are the best ones for the requirements domain, or whether other types of recommender models might be more appropriate.

In addition to the domain characteristics described earlier, there are several additional unique features that could impact the implementation of recommender systems in the requirements domain. For example, the nature of the domain means that the community of users, although theoretically open, is in practice comprised of a more identifiable and known group of people. Many of the stakeholders who will use the system have clearly defined roles and responsibilities, which creates the opportunity for using role-based knowledge to improve recommendations. Similarly, because many stakeholders are involved in the requirements process as part of their job, we can expect a higher than average level of feedback, and future recommender models could be designed to take advantage of this feedback.

Although this work represents an initial investigation into the use of data mining techniques and recommender systems in the requirements domain, the initial results suggest the usefulness of these techniques for facilitating requirements elicitation in large and distributed projects.

8. ACKNOWLEDGMENTS

The work described in this paper was partially funded by NSF grants CCR- 0306303, CCR-0447594, and IIS-0430303.

9. REFERENCES

[1] Basu, C., Hirsh, H., & Cohen, W. Recommendation as Classification: Using Social and Content-Based Information in Recommendation. National Conference on Artificial Intelligence, (Madison, WI, 1998), 714-720.

[2] Breese, J. S., Heckerman, D., & Kadie. C. Empirical analysis of Predictive Algorithms for Collaborative Filtering, Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, (Madison, WI, 1998), 43–52.

[3] Can, F. and Ozkarahan, E. A. 1990. Concepts and Effectiveness of the Cover-coefficient-based Clustering Methodology for Text Databases. ACM Trans. Database Syst. 15, 4 (Dec. 1990), 483-517.

[4] Castro Herrera, C., Duan, C., Cleland-Huang, J. and Mobasher, B. Using Data Mining and Recommender Systems to Facilitate Large-Scale, Open, and Inclusive Requirements Elicitation Processes, IEEE Conf. on Requirements Eng., (Barcelona, Spain, Sept. 2008), 165-168.

[5] Cleland-Huang, J. and Mobasher, B. Using Data Mining and Recommender Systems to Scale up the Requirements Process, ACM Intn'l Workshop on Ultra-Large Software Systems, (Leipzig, Germany, May, 2008), 3-6.

[6] Davis, A., Dieste, O., Hickey, A., Juristo, N., & Moreno, A. Effectiveness of Requirements Elicitation Techniques, IEEE Intn'l Requirements Engineering Conf., (Minneapolis, MN, Sept. 2006), 179-188.

[7] Dhillon, I. S. and Modha, D. S. 2001. Concept decompositions for large sparse text data using clustering. Machine Learning, 42, ½, (Jan. 2001), 143-175.

[8] Duan, C., Clustering and its Application in Requirements Engineering, Technical Report #08-001, School of Computing, (DePaul University, February, 2008).

[9] Hair, J.F., Black, B., Babin, B., Anderson, R.E. and Tatham, R. L., Multivariate Data Analysis, 5th ed. Upper Saddle River, NJ: Prentice Hall, 1998.

[10] Herlocker, J., Konstan, J., Borchers, A., and Riedl, J. An algorithmic framework for performing collaborative filtering. 22nd ACM Conf. on Research and Development in Info.Retr. (SIGIR'99),(Berkeley, CA, Aug. 1999), 203-237.

[11] Jain, A. K., Murty, M. N., and Flynn, P. J. 1999. Data Clustering: A Review. ACM Comp. Surveys, Vol 31, No. 3, 264-323.

[12] Northrop, L., Feiler, P., Gabriel, R., Goodenought, J., Linger, R., Longstaff, T., Kazman, R., Klein, M., Schmidt, D., Sullivan, K., Wallnau, K., Ultra-Large-Scale Systems: The Software Challenge of the Future, Tech. Report, Software Eng. Institute., (June 2006).

[13] Schafer, J. B., Frankowski, D., & Shilad, S., Collaborative Filtering Recommender Systems. In P. Brusilovsky, A. Kobsa, & W. Nejdl, The Adaptive Web: Methods and Strategies of Web Personalization. New York: Springer-Verlag. 2007.

[14] Strehl, A. and Ghosh, J. 2003. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn.* Res. 3 (Mar. 2003), 583-617.

[15] Zhao, Y, and Karypis, G. 2001. Evaluation of hierarchical clustering algorithms for document datasets. In Proceedings of the Intn'l Conf. on information and Knowledge Management, (McLean, Virginia, Nov 4-9, 2002), 515-524.

[16] Zhong, S. and Ghosh, J. 2003. A unified framework for model-based clustering. J. Mach. Learn. Res. 4 (Dec. 2003), 1001-1037.