Community Rating Service and User Buddy Supporting Advices in Community Portals

Martin Vasko[1], Uwe Zdun[1], Schahram Dustdar[1], Andreas Blumauer[2], Andreas Koller[2] and Walter Praszl[3]
[1]Distributed Systems Group, Vienna University of Technology, Austria
{m.vasko,zdun,dustdar}@infosys.tuwien.ac.at
[2]punkt.netServices, Vienna, Austria
{koller,blumauer}@punkt.at
[3]Special Interest Magazines, Vienna, Austria
w.praszl@simskultur.net

Abstract: Many community portals allow users to search for events, such as concerts, festivals or other things of interest and to rate them. Especially in the culture domain the users' impressions of events is based on many factors, such as quality, personal interests, etc. Such factors can be represented using an ontology. The ratings provided by the users of community portals are often highly biased by personal opinions, and hence not all information provided by users is useful for all other users. But it can be observed that users with similar interests provide similar opinions. This paper introduces a community rating approach based on this observation. Our concept introduces for each user a user buddy, representing the part of the community with similar opinions as those of the user. The buddy uses a community rating service as a basis to give advices to users, such as recommendations for events or help in searching the portal. Our approach gathers opinions using a domain ontology, but it is not dependent on a specific ontology.
Key Words: Community Rating, Ontologies, Web Services, Community Portals
Category: H.3, H.3.1, H.3.5

1 Introduction

Many community portals [Schuemmer and Lukosch 2007] for diverse domains are existing on the Web. The users of the community portals usually provide information about events or things of interest to other users in the community. In many cases the relevant information is hard to find. A simple reason for this is the mass of information provided in community portals: it is hard for the user to filter out the useful information. Unfortunately, automatically filtering the information is difficult, too, because of the diversity of opinions in online user communities.

Consider the culture domain as a typical example. In this domain, cultural events, such as concerts or festivals, are advertised and rated on community portals. Large community portals in this domain usually have many users with diverse interests. Even people going to the same kind of event often have different

preferences. Consider the example of a festival: For some users only the quality of the music counts, for others additional attractions or the quality of the camping site are as important. This paper deals with the question: If thousands of opinions for such events are provided in a community portal, how can a user retrieve the useful information, given the user's personal preferences, and how can the community portal help the user to retrieve the information?

Structuring the various factors in a user's opinion is – in the context of the Semantic Web [Berners-Lee 1999] – done using ontologies. Ontologies unify diverse understandings by introducing a central perception hierarchy for different knowledge domains. However, there is the problem that in many domains which use community portals, such as the culture domain, no well accepted standard ontology exists for the whole domain, but only partial ontologies. In addition, a generic approach should not be dependent on one domain ontology, but be open for any domain. Also, over time, the domain ontology must be adapted.

In our approach users can provide opinions about events or things of interest. Other users can provide a rating about a given opinion. We assume that users with similar interests and background provide the most useful information for a user. Hence by collecting and evaluating the user's ratings about other users' opinions, we can provide a *community rating* that shows the relevance of a user's opinion for another user. The community rating is transitive in the sense that it not only considers the ratings of a user about other users' opinions, but also the ratings of the other users about yet other users' opinions, and so on.

A central *community rating service* calculates the community ratings between the user and all other users. This way, a *user buddy* is dynamically constructed, which is an abstraction representing the user's specific view on the community. The user buddy is then used to calculate advices for the users of the community portal. Note that our approach uses a custom ontology as the basis to describe all the factors relevant for user ratings. The general approach hence is open to be used with any ontology. We use a service-oriented architecture for the community service in order to deal with heterogeneous platforms and technologies that are usually used for community portals and semantic web ontologies.

Our approach is exemplified for a community portal for cultural events [SCG 2008]. Our community portal offers an *event buddy* using the community rating service to give advice for future events. The community ratings are calculated based on ratings given for user opinions on (mostly past) events. The culture portal uses a culture ontology as a basis for all user opinions, which is maintained by the culture experts who run the portal.

This paper is organized as follows: An overview and a motivating example are provided in Section 2. The community rating service, the user buddy, and the system architecture are described in Section 3. Our approach is compared to existing approaches in Section 4. Finally Section 5 concludes the paper.

2 Motivating Example

In this section we give an overview of our approach from the user's perspective using a motivating example for a situation in which a community rating can be applied. This example is resolved later in this paper. Consider users provide opinions for events, attractions, or things of interest via the community portal. A user opinion consists in our approach of a preselected set of elements from an ontology. The ontology is maintained by experts from the domain, and the experts also select the elements from the ontology which can be rated by users. For instance, in our culture portal 4-8 ontology elements are selected to be rated per culture event, and in addition the user can provide free text. Users are automatically asked to provide their opinion after they visited events (i.e., if they bought a ticket via the portal), as well as future events in which they are interested. They are motivated to participate using lotteries for free tickets.

When a user views another user's (anonymous) opinion, the user is asked to give a rating on that opinion. This rating should be provided in a simple and easy-to-use fashion. The user is simply asked: "How do you rate this user opinion?" The user can answer on a scale ranging from "very helpful" to "not helpful at all", and as a result values ranging from 1.0 to 0.0 are produced. This way, incrementally a graph of ratings about other users' opinions is built up. This graph is the basis for calculating the user buddy. New users must first train their buddy – to get connected to the community graph. A number of random user opinions are shown to the user, and the user must give a rating for them.

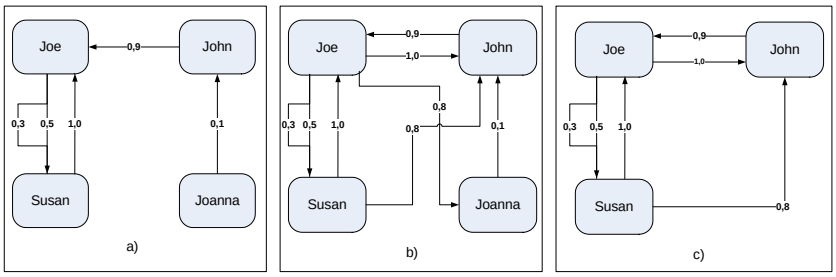

Figure 1: Sample User Ratings

Three rating scenarios are illustrated in Figure 1. In first scenario (a) four users have provided ratings. For instance, user *Susan* has provided one rating on an opinion by *Joe*, and the opinion was very helpful to her (1.0). *Joe*, in turn, has provided two ratings about opinions by *Susan*, with the values 0.3 and 0.5. The complete ratings derived from Scenario (a) are: $< John \overset{0,9}{\rightarrow} Joe >, <$

$Susan \overset{1,0}{\rightarrow} Joe >, < Joe \overset{0,3}{\rightarrow} Susan >, < Joe \overset{0,5}{\rightarrow} Susan >, < Joanna \overset{0,1}{\rightarrow} John >$. Scenario (b) shows the addition of more ratings over time. Finally, Sceanrio (c) illustrates the removal of a user and the consequences for the ratings. Based on such rating graphs we can derive the *community rating* to produce the user buddy that represents the user's view on the community.

3 Community Rating Service

In this section we present the details of the community rating service and introduce the prototype implementation details.

3.1 Application Logic of the Community Rating Service

Algorithm 1 illustrates the main logic of the community rating service. The algorithm recursively calculates the community rating between two users $from$ and to for a depth of $levels$ through the graph. All users for which the user $from$ has given one or more ratings are considered. If there is a direct rating between the user and to, the direct rating is considered with a factor $DirectRatingFactor$. Otherwise the community rating is calculated recursively, and added with a factor of 1. If no rating has been added, or if the $levels$ are exceeded, -1 is returned to indicate the stop of the recursion. This way all direct and transitive ratings from user $from$ to to up to the depth $levels$ are added. Finally they are weighted by the $number$ of ratings that have been added. We use the function $getAverageRating()$ to obtain a rating between two users, because each user might have n ratings for another user. Please note there are two ways to fine-tune this algorithm:

- $levels$ determines the depth of the search for ratings through the graph. If the graph is only loosely populated, the number of levels can be increased to obtain better results. If the performance decreases because of the size of the graph, the number of levels can be decreased.
- $directRatingFactor$ determines the importance of a user's own judgment compared to ratings made by others.

3.2 Example Resolved

Figure 2 illustrates the community ratings calculated for the examples from Figure 1. The user ratings from Scenario (a) in Figure 1 result in the community ratings depicted in Scenario (a) in Figure 2. Scenarios (b) and (c) in Figure 1 illustrate the adding of ratings and the removing of users. Scenario (b) in Figure 2 illustrates the resulting community ratings.

Algorithm 1 CommunityRating

```
Require: from : User, to : User, levels : Integer
Ensure: directRatingFactor
  number := 0
  ratingSum := 0
  addedRating := false
  if levels == 0 then
    return -1
  end if
  for all u : User in getUserRatings(from) do
    if u == to then
      addedRating := true
      number += directRatingFactor
      ratingSum += directRatingFactor * getAverageRating(from, u)
    else
      communityRating := CommunityRating(u, to, levels - 1)
      if communityRating != -1 then
        addedRating := true
        number += 1
        ratingSum += 1 * getAverageRating(from, u) * communityRating
      end if
    end if
  end for
  if addedRating == false then
    return -1
  end if

  return ratingSum / number
```

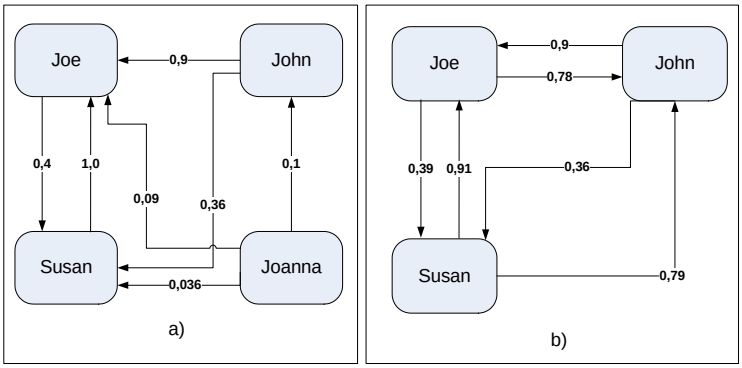

Figure 2: Sample Community Ratings

3.3 User Buddy

To use the community ratings in order to give advices to users, we developed a virtual *User Buddy* concept. The aim of this abstraction is to track individual user preferences and to aggregate interesting events, attractions, or things of interest according to the community rating. The buddy uses the community rating service to give advice to users, such as recommendations or helping users to search the portal. By actively providing user ratings about other user's opinions, users teach the buddy their personal preferences.

From a user perspective, trustworthy rating mechanisms will only be accepted

if they (1) help to improve ranking and filtering of information and (2) if they do not rely on methods which need intensive training efforts accomplished by the user. In our approach the buddy learns quickly, even when single users do not put much efforts on the buddy training since our approach relies on the overall community behavior. That is, the user can benefit from the training of the buddy that other users have performed.

In the culture portal case study, we provide an *event buddy* to give advices for cultural events. The event buddy aggregates the community ratings for each user. When the user searches for a cultural event, hence the opinions of those users that the user has directly or indirectly given a high community rating, are considered. Three exemplary usage scenarios are:

- The event buddy can be asked to provide a list of recommended events in a time frame. This is done by calculating a list of users (number can be configured) with the highest community ratings. Then the events with a positive opinion (i.e. over a certain threshold) that take place in the specified time frame are recommended.

- The event buddy can provide opinions on a specific event sorted by their relevance for the user, based on the community ratings.

- The event buddy can tell the user the relevance of an opinion for him. Consider user *John* in Scenario (c) of Figure 1 would like to see a theater play, and the user *Susan* has given a very positive opinion on the play. Even though *John* has never rated *Susan* himself, the event buddy can give the advice that this positive opinion has to be considered with care, because *Susan* has only a low community rating, meaning that users (in this case *Joe*) that have similar opinions like *John* have given low ratings for *Susan*.

3.4 Service-Based Integration

Figure 3 illustrates an abstract overview of the service environment of the culture portal. The heterogeneous environment in the project motivated us to implement the algorithm as a Web Service based on Apache Axis [Axis 2008]. Additional components (Thesaurus Server, Triple Store, CMS functionality etc.) are integrated by the use of Web Services as well.

As can be seen in the previous section, the community rating service has no direct dependencies to information of the Web Portal, such as the ontologies used for rating the user opinions. The only input dependencies are users (addition, removal) and user ratings (addition, removal), and the only output dependencies are the community ratings for the user buddy. Hence, it makes sense to enable the integration of the community rating service into different system architectures which is achieved using Web Services.

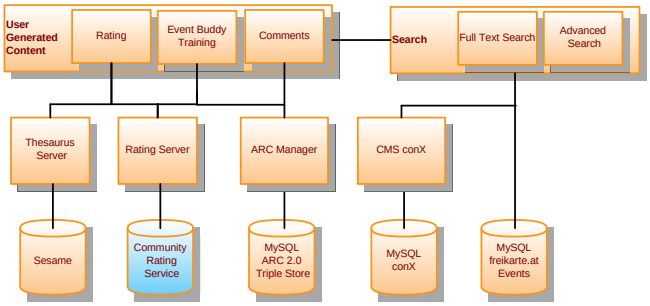

Figure 3: System Architecture

4 Related Work

[Staab et al. 2000] introduce a community Web Portal using semantic approaches to structure and unify diverse information. Their motivation to use ontologies for capturing knowledge in a generic way and to explicitly specify shared concepts corresponds to the motivation for this work.

[Galizia et al. 2007] introduce a trust based methodology for Web Service selection. Their work introduces a Web Service Trust Ontology (WSTO) based on Web Service Modeling Ontology (WSMO [Fensel et al. 2006]). Their approach matches classes of Web Services with participant trust profiles.

Ranking user content is quite popular in the field of semantic computing.

An approach to rank ontologies is presented by [Tartir and Arpinar 2007]. They introduce OntoQA, a tool to evaluate and rank ontologies based on a catalog of metrics. OntoQA provides a tuneable ranking approach by allowing the user to bias the preference for certain ontologies.

[Massa and Bhattacharjee 2004] provide an experimental analysis of a community Web Portal based on a recommender system incorporating trust. The authors argue, that classical collaborative filtering approaches consider only a small portion of the user base whereas trust-aware mechanisms build on a high rate of the whole user community. [Zhdanova and Fensel 2005] identify the creation of semantic web content and a community-driven ontology management as approaches to overcome the limitations of current community Web Portals.

[Schuemmer and Lukosch 2007] describe software design patterns in groupware systems. The *Buddy List* pattern descibes personalized user lists. This description matches to the group of users identified by the *community rating* algorithm. The *Expert Finder* pattern describes the problem to identify a user having expertise on a special artifact in the platform. This pattern can be mapped to the problem of identifying users in the community sharing the same preferences.

5 Conclusion and Future Work

The introduced *community rating service* provides an approach to relate and weigh diverse opinions of community portal users. The approach can work with arbitrary ontologies for defining the rating of opinions on events, attractions, and other things of interest, but it is not dependent on the ontology used. Our approach provides individual users with an individualized view onto the communities' opinions. As part of the Web platform a *user buddy* is introduced, which uses the community rating service to provide advices, such as recommendations or help in searching the portal, to the users. By "teaching" individual event preferences, the user is able to sharpen the accuracy of recommendations. The combination of the community rating service and the user buddy significantly improves the recommendation functionality in community portals. Future work covers the application of the algorithm on bigger communities by the integration of further cultural event platforms in the SCG prototype [SCG 2008].

Acknowledgment

This work is supported by a BMVIT/FFG grant for the FIT-IT project "Semantic Culture Guide" [SCG 2008].

References

[Berners-Lee 1999] Berners-Lee, Tim. Weaving the Web, Orion Business Books, London, 1999

[Fensel et al. 2006] Fensel, Dieter; Lausen, Holger; Polleres, Axel; Brujin, Jos de. Enabling Semantic Web Services: Web Service Modeling Ontology. Springer, 2006

[Galizia et al. 2007] Galizia, Stefania; Gugliotta, Alessio; Domingue, John. A Trust Based Methodology for Web Service Selection. International Conference on Semantic Computing(ICSC), pp.193-200, September 2007

[Massa and Bhattacharjee 2004] Massa, Paolo; Bhattacharjee, Bobby. Using trust in recommender systems: an experimental analysis. iTrust, pp. 221-235, Springer, 2004

[Schuemmer and Lukosch 2007] Schuemmer, Till; Lukosch, Stephan. Patterns for computer-mediated interaction. Wiley Series in Software Design Patterns, 2007

[Staab et al. 2000] Staab, Stefan; et al. Semantic community Web portals. Int. Journal of Computer Networking and Telecommunication, pp. 473-491, June 2000

[Tartir and Arpinar 2007] Tatir, Samir; Arpinar, I. Budak. Ontology Evaluation and Ranking using OntoQA. Int. Conf. on Semantic Computing(ICSC), 2007

[Weerawarana et al. 2005] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, D.F. Ferguson. Web Services Platform Architecture, Prentice Hall, 2005

[Zhdanova and Fensel 2005] Zhdanova, Anna; Fensel, Dieter. Limitations of Community Web Portals: A Classmates Case Study. Int. Conf. on Web Intelligence, 2005

[Axis 2008] Axis2 Version 1.3, Apache Software Foundation, http://ws.apache.org/axis2/, Last Access: 2008

[SCG 2008] Web Portal Prototype, Semantic Culture Guide Project, http://www.semantic-culture-guide.net/, Last Access: 2008

[WSDL 2008] Web Service Description Language, W3C, http://www.w3.org/TR/wsdl, Last Access: 2008