A Quality of Service Management Framework Based on User Expectations

Vikas Deora, J. Shao, W. Alex Gray, and Nick J. Fiddian

School of Computer Science
Cardiff University
Cardiff, UK
{v.deora,j.shao,w.a.gray,n.j.fiddian}@cs.cf.ac.uk

Abstract. The ability to gauge the quality of a service is critical if we are to achieve the service oriented computing paradigm. Many techniques have been proposed and most of them attempt to calculate the quality of a service by collecting quality ratings from the users of the service, then combining them in one way or another. We argue that collecting quality ratings alone from the users is not sufficient for deriving a reliable or accurate quality measure for a service. This is because different users often have different expectations on the quality of a service and their ratings tend to be closely related to their expectations, i.e. how their expectations are met. In this paper, we propose a quality of service management framework based on user expectations. That is, we collect expectations as well as ratings from the users of a service, then calculate the quality of the service only at the time a request for the service is made and only using the ratings that have similar expectations. We give examples to show that our approach can result in a more accurate and meaningful measure for quality of service.

1 Introduction

There is a growing interest in service oriented computing (SOC) in recent years [1,2,3,4]. Central to SOC is the notion of *service* which can broadly be considered as a software component that represents some computational or business capability. By allowing services to be advertised declaratively, discovered dynamically and invoked remotely, SOC makes it possible for users to locate, select and execute services without having to know how and where they are implemented. This new computing paradigm offers great potential for agent-based e-commerce applications [5,6,7]. For example, vendors may wish to identify suitable partners from time to time to form a virtual organisation [8] so that they together can compete better in the market, and consumers would always want to select the services that best serve their interests. All such "match-making" tasks can potentially be performed by the agents automatically in an SOC environment.

In this paper, we consider the problem of quality of service (QoS) management in SOC. This is an important problem to consider because, just like in any other business environment, it is possible to have several service providers (SP)

M.E. Orlowska et al. (Eds.): ICSOC 2003, LNCS 2910, pp. 104–114, 2003.

offering the same service but with different qualities in an SOC environment. It is essential, therefore, that an agent should select a service that meets not only the required capability with the lowest possible price, but also the quality requirement. Various methods for modelling, calculating and monitoring QoS have been proposed in the literature [9,10,11,12], especially for web services and multi-agent systems [13,14,15]. A common approach is to collect quality ratings from the users of a service and then aggregate them in one way or another to derive the quality of the service. The following example explains this.

Suppose that we have three SPs who offer a multimedia news service to PDA or mobile phone users. Suppose also that there are six users (or their agents) who have used the services, and each of them has been asked to rate the quality of the service he or she has used in terms of news update frequency. Table 1 below shows the quality ratings collected from the users, where ratings are expressed as real numbers in $[0, 1]$ with 0 representing the most unsatisfactory quality and 1 the most satisfactory.

Table 1. Collected Quality Ratings

Users	SP1 update frequency	SP2 update frequency	SP3 update frequency
A1	0.3		0.3
A2	0.8	0.9	
A3	0.3		1.0
A4		0.8	
A5	0.5		0.1
A6	0.6	0.3	
Aggregate rating	0.50	0.67	0.47

For simplicity of presentation, we assume that the aggregate quality rating for each SP is derived by combining the individual ratings using a simple arithmetic average. So according to Table 1, `SP2` offers the best service with respect to news update frequency.

While various methods may be employed to aggregate the collected ratings more rationally, for example, using a weighted average so that the reputation or trust of the user may be taken into account [16,14], this approach to quality rating calculation suffers from two fundamental weaknesses:

– First, users are invited to rate a service in "absolute" terms, e.g. 0.3 or 0.8 out of 1.0 in our example. Such quality ratings may not be very meaningful or can even be misleading in some cases, because the context within which the ratings are derived is not known. For example, A1 rated SP1 low perhaps because SP1's news update was not frequent enough for him or her, but this does not necessarily mean that the same frequency is not good enough quality for someone else, e.g. for A2.

– Second, the aggregate quality rating for a service is derived "statically" using all the ratings collected from the users. This does not take into account the fact that some of the ratings may not be relevant to a particular quality assessment request. For example, if the request was to assess the quality of SP1, SP2 and SP3 in terms of their ability to update news at least 4 times a day, then A6's rating should not be included in the quality calculation if A6 had expected a minimum of 8 updates per day from SP2.

In this paper, we address the above two problems by introducing a new model for collecting and monitoring QoS "relatively". That is, we attempt to collect from service users QoS ratings as well as their expectations on QoS, so that we can measure QoS in relative terms, i.e., how well a delivered service meets users' expectations. Based on user expectations, we also propose to calculate the quality of a service dynamically at the time a request for QoS assessment is made, and use only the ratings that have similar expectations. We show that our approach can result in a more accurate and meaningful measure for quality of service.

The rest of the paper is organised as follows. Section 2 introduces our framework for managing QoS information. Section 3 discusses how we calculate QoS based on user expectations and gives examples to show how our approach works. Related work is considered in Section 4, and finally Section 5 presents conclusions and discusses future work.

2 The QoS Model

To develop a QoS management model, it is essential to understand what the term *quality* actually entails. Unfortunately, what defines quality is vague, and different views exist in different studies and from different perspectives [10,11, 17,12]. The following three views are, however, most common.

– Quality as Functionality. This view considers quality in terms of the amount of functionality that a service can offer to its users. For example, if SP1 allows you to select different positions of cameras from which you may watch a football game, and if this functionality is not provided by SP2 or SP3, then SP1 can be considered as offering a better quality than SP2 and SP3 do.
– Quality as Conformance. This view sees quality as being synonymous with meeting specifications. For example, if SP1 specified in its service agreement that it would provide 1 Mb/s bandwidth for its news service and SP1 did provide users with 1 Mb/s bandwidth (or more) at all times in its operation, then SP1 is usually considered as offering good quality of service.
– Quality as Reputation. This view links quality to users' perception of a service in general. It is worth noting that this perception is typically built over the time of the service's existence. For example, the BBC news service is generally considered to be offering good quality to its users, due to its reputation built over many years as a news service provider.

These different views of quality require QoS to be monitored and measured differently. Quality as functionality characterizes the design of a service and can only be measured by comparing the service against other services offering similar functionalities. Quality as conformance, on the other hand, can be monitored for each service individually, and usually requires the user's experience of the service in order to measure the "promise" against the "delivery". Finally, reputation can be regarded as a reference to a service's consistency over time in offering both functionality and conformance qualities, and can therefore be measured through the other two types of quality over time.

While it is possible to establish all three types of quality for a service in an SOC environment, it is perhaps most interesting and relevant to understand how quality as conformance may be monitored. In this paper, therefore, we adopt the conformance view and define QoS to be a degree of satisfaction that the user has experienced after using a service. More specifically,

Definition 1. *let S be a service and $A_1, A_2, \ldots, A_n$ be a set of attributes that describe S and upon which we wish to monitor the quality for S. Assume that for each A_i, A_i^a is the advertised quality (or the quality that the service provider promised to offer), and A_i^d is the delivered quality (or the actual quality that the service provider delivered). Then the QoS for A_i is given by*

$$Q_{A_i} = f(A_i^a, A_i^d)$$

where f is a function that calculates the conformance between A_i^a and A_i^d.

The above definition captures the notion of conformance generically, but does not specify how A_i^a and A_i^d may be obtained and Q_{A_i} may be calculated. In practice, it may not be realistic to expect every A_i to have an A_i^a value specified by the service provider, and its A_i^d value monitored and Q_{A_i} calculated by the system automatically. Often, we need user feedback to help assess the quality of a service.

While the need for involving users in QoS assessment is well recognised, existing methods tend to collect quality ratings only from the users. This is inadequate if we wish to measure quality as conformance according to Definition 1. In this paper, we propose to collect "fuller" ratings from the users and then to use such ratings to assess QoS.

Definition 2. *Let U be a user and A be an attribute of a service S. A quality rating on A by U is a triple*

$$\langle E_u(A), P_u(A), R_u(A) \rangle$$

where $E_u(A)$ represents the quality that U expects from A, $P_u(A)$ the actual quality of A perceived or experienced by U after using S, and $R_u(A)$ the quality rating that U gives to A.

Collecting $\langle E_u(A), P_u(A), R_u(A) \rangle$ from users can perhaps be considered as a way of "materialising" the conformance calculation function introduced in

Definition 1. Instead of relying on the system for monitoring A_i^d and calculating Q_{A_i}, we obtain $P_u(A_i)$ and $R_u(A_i)$ from the user. The use of $E_u(A_i)$ represents a shift from using SP advertised values to user expectations on quality in QoS calculation. This is significant. While some correlation between A_i^a and $E_u(A_i)$ can be expected - users are likely to be influenced by advertisement in forming their expectations, expectations are not solely based on advertisement. Other factors, such as the user's past experience with the service, the price the user is paying for the service, or the recommendation by a friend for the service can all influence the user in forming his or her expectation on the quality of the service. Thus, by including user expectations as part of user rating on a service, we can hope to interpret such ratings more accurately and meaningfully.

To explain how the proposed QoS model works, consider the example we gave in the Introduction again. Suppose that we still ask the six users to rate the new services in terms of update frequency, *fr*, but this time use the expectation model we introduced here. Assuming that we represent $E_u(\text{fr})$, $P_u(\text{fr})$ and $R_u(\text{fr})$ all as real numbers in $[0, 1]$, Table 2 below shows the ratings collected from the users.

Table 2. Expectation based Quality Ratings

Users	SP1 $\langle E(\text{fr}), P(\text{fr}), R(\text{fr})\rangle$	SP2 $\langle E(\text{fr}), P(\text{fr}), R(\text{fr})\rangle$	SP3 $\langle E(\text{fr}), P(\text{fr}), R(\text{fr})\rangle$
A1	$< 0.9, 0.7, 0.3 >$		$< 0.7, 0.5, 0.3 >$
A2	$< 0.4, 0.4, 0.8 >$	$< 0.5, 0.5, 0.9 >$	
A3	$< 0.8, 0.6, 0.3 >$		$< 0.4, 0.5, 1.0 >$
A4		$< 0.6, 0.6, 0.8 >$	
A5	$< 0.9, 0.7, 0.5 >$		$< 0.9, 0.5, 0.1 >$
A6	$< 0.9, 0.7, 0.6 >$	$< 0.7, 0.5, 0.3 >$	

How a user arrived at a particular rating may never be known to us, but it is interesting to speculate what the ratings shown in Table 2 might suggest. The majority of the users of SP1 and SP3 seem to have high expectations (probably as the result of some effective recommendations or advertising effort), but do not seem to get what they expect (perhaps due to the unexpected level of business that SP1 and SP3 have got themselves into). SP2, on the other hand, is the opposite: users do not have high expectations but are generally satisfied with the service. In the following section, we show how this difference in expectation is taken into account when assessing QoS for services.

3 Collection and Calculation of QoS Ratings

In this section, we consider how the QoS ratings may be collected from the users and be used in the calculation of QoS by the Quality Assessment (QA)

agent that we are currently constructing as part of the CONOISE project (`www.conoise.org`). The basic system architecture is outlined in Figure 1 below.

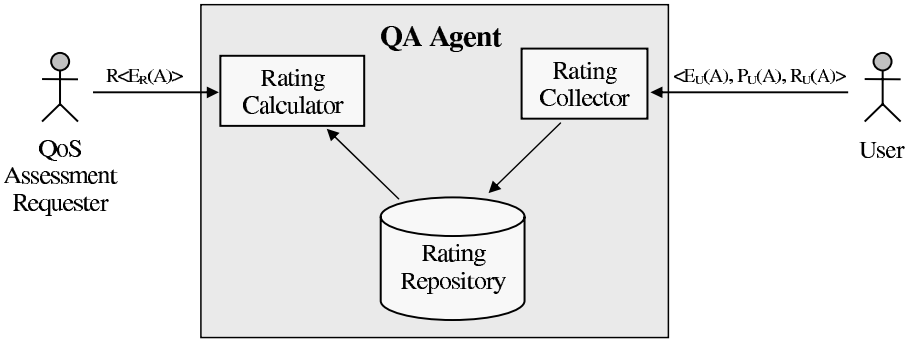

Fig. 1. The system architecture of the QA agent

The QA agent consists of two main components. The Rating Collector is responsible for soliciting quality ratings from the users. In this paper, we have assumed that the users are willing, when asked, to return quality ratings on the services they have used and their ratings can be trusted. We have also assumed that all three elements of a user rating, $E_u(A)$, $P_u(A)$ and $R_u(A)$, are expressed as real numbers in $[0, 1]$. These simplification assumptions have resulted in a fairly straightforward process for collecting ratings from the users, but they may not be particularly realistic for many practical applications. To relax these assumptions, it is possible to incorporate some more advanced techniques into our QA agent, so that issues such as how to collect quality ratings from users or agents whom we can not fully trust may be addressed [18,16,19,14].

The Rating Calculator is responsible for calculating QoS from the collected ratings. To aggregate individual ratings into an overall assessment of quality for a given service S, two calculations are necessary:

1. combining individual ratings for each A_i of S into an aggregate rating for A_i, and
2. combining the ratings for individual A_i's into an overall rating for S.

Currently, we treat all quality attributes of a service to be of equal importance and the overall rating for S is derived by a simple average of the individual ratings for its attributes. But it is possible to consider a weighted average so that the fact that some attributes are more significant than others may be taken into account [20].

How to combine individual ratings for each A_i into an aggregate one, however, needs some explanation. In contrast to many existing methods which simply aggregate *all* the collected ratings on A_i, our approach is to selectively aggregate only the ratings that have similar expectations. That is, we allow a quality

assessment request R to specify a quality expectation, $E_R(A_i)$, on A_i, and derive an aggregate quality rating for A_i by using only the ratings in the Rating Repository that have similar expectations to $E_R(A_i)$. More specifically,

- If R does not specify any quality expectation on A_i, then $Q(A_i)$, the quality rating for A_i, is

$$Q(A_i) = \sum_{j=1}^{k} w_j \times R_j(A_i)$$

  where $R_j(A_i)$ is user j's rating on A_i and w is a weight. This is equivalent to the majority of existing approaches to quality calculation: the overall rating for A_i is a weighted sum of individual ratings, and the weights are used to allow factors such as reputation or trust to be taken into account [16,14].
- If R specifies a quality expectation $E_R(A_i) = \alpha \in [0, 1]$ on A_i, i.e. the quality expectation on A_i is α, then

$$Q(A_i) = \sum_{j=1}^{m} w_j \times R'_j(A_i)$$

  where $R'_j(A_i)$ is the element of the rating $\langle E'_j(A), P'_j(A), R'_j(A)\rangle$ in the Rating Repository whose corresponding expectation element $E'_j(A_i)$ is *similar* to $E_R(A_i) = \alpha$. In this paper, we use a simple criteria for determining whether the two are similar: $E'_j(A_i)$ and $E_R(A_i) = \alpha$ are compatible if $|E'_j(A_i) - \alpha| \leq \delta$, where δ is a constant. However, more complex forms of similarity test are possible, for example, by specifying quality expectations as "ranges" over $[0, 1]$ (instead of *points*) and by allowing fuzzy matching between $E'_j(A_i)$ and $E_R(A_i) = \alpha$.

By aggregating individual quality ratings dynamically at the time when a QoS assessment request is made and by comparing the raters' and the requester's expectations on qualities, our approach is able to calculate QoS in "context", that is, to use the ratings which are actually relevant to the context within which the QoS assessment request is made.

Now consider our example and Table 2 again. Suppose that the QA agent has been asked to assess QoS for all three SPs in terms of news update frequency, given $E_R(\text{fr}) = \text{unspecified}$, $E_R(\text{fr}) = 0.5$ and $E_R(\text{fr}) = 0.8$, respectively. Assuming that we have $\delta = 0.1$, the result of calculation by the QA agent is given in Table 3.

As can be seen from Table 3, the quality ratings for SPs can vary with respect to expectations. For example, when the expectation is $E_R(\text{fr}) = 0.5$, SP3 emerges to be the best service provider, whereas when the expectation is changed to $E_R(\text{fr}) = 0.8$, we have SP1 as the winner. This is in contrast to conventional approaches to quality calculation that do not consider user expectations (equivalent to setting $E_R(\text{fr}) = \text{unspecified}$, resulting in SP2 being the best provider for all cases), our method gives a more meaningful rating for a service on a case-by-case basis.

Table 3. Calculated quality ratings for SPs

Expectation	SP1 aggregate rating	SP2 aggregate rating	SP3 aggregate rating
$E_R(\text{fr}) = \text{unspecified}$	0.50	0.67	0.47
$E_R(\text{fr}) = 0.5$	0.80	0.85	1.00
$E_R(\text{fr}) = 0.8$	0.43	0.30	0.20

Finally, it is worth mentioning that although $P_u(A_i)$, the quality perceived by the user, is not used in quality calculation in this paper, it can play an important role in deriving more accurate quality assessment. For example, by monitoring the relationship between $R_u(A_i)$ (the verdict) and $|E_u(A_i) - P_u(A_i)|$ (the basis for the verdict) over a period of time with sufficient rating data, we can determine whether a particular user has been harsh or lenient in rating the services. By factoring such knowledge into quality calculations, we can deliver more accurate QoS assessment of services.

4 Related Work

There exist a large number of proposals in the literature for managing QoS as reputation or trust for service providers. These approaches, e.g. [21,16,19,14], typically seek to establish the quality of a service by gathering ratings from the users who have used the service, but do not consider the context within which the ratings are derived. The need for having some contextual information alongside the ratings themselves has been identified only recently. For example, Maximilien and Singh [20] suggest that some attributes be used to describe the ratings and users be allowed to define their preferences on the importance of individual ratings. While this work is similar to the method proposed here in principle, it does not include user expectations as part of the context, and thus does not help to identify the reasons behind the ratings given by users.

Our approach to include expectations as part of QoS management was inspired by work done in the area of marketing, where user expectations are commonly collected as a way of understanding quality. For example, the SERVQUAL system [12] uses the difference between what consumers expect and what they perceive to determine product/service quality, so that quality may be improved to meet users' expectations better. Our approach however differs from marketing-based ones in that we do not directly use expectations to calculate the actual quality of a product or service, but as a basis for determining which ratings are relevant to a given QoS assessment request.

The idea of using similar expectations in assessing QoS is also akin to the concept of *collaborative filtering* [22]. In collaborative filtering, users are categorised into several groups and the users in the same group are considered to have the same "likes" and "dislikes", or expectations. This common expectation

is then used to determine or predict if a given item (e.g. a book) would be of any interest to a particular user. We also use similar expectations to determine the quality of a service for a given quality assessment request in our QoS management framework, but there is a fundamental difference between our work and that in collaborative filtering: we do not attempt to classify users (raters and requesters). The groups of users who have similar expectations are formed dynamically in our model, based on the given QoS assessment request and the criteria for similarity measures.

It is worth noting that quality management has also been considered in other research areas [9,10,17,23,24]. For example, Mecella et al. [17] have designed a broker which can select the best available data from a number of different service providers based on a specified set of data quality dimensions such as accuracy, completeness, currency and consistency. In [9,10], QoS management for network resources has been considered. These studies aim to identify and establish suitable quality matrices for specific services in specific application areas, so that the quality of these services can be meaningfully gauged. In contrast, the quality management model proposed in this paper assumes the availability of appropriate quality attributes (or matrices), and is designed to address the problem of how qualities of similar services may be compared based on user ratings and expectations.

5 Conclusion

In this paper, we have introduced a user expectation based framework for modelling and calculating QoS in an SOC environment. This framework is founded on the following basic observation: if A rates the quality of S as x, then this rating is only meaningful to B if A and B have similar expectations on S. So instead of aggregating all the quality ratings collected from the users of services, we propose to calculate QoS dynamically at the time a QoS assessment request is made, and use only the ratings that have similar expectations to that of the request. This, as we have shown in the paper, can lead to more accurate and meaningful rating in QoS assessment.

The work reported in this paper is still at an early stage, and a number of issues still need to be investigated. First, there is a need to consider how user expectation may be best represented. The current real-number based representation is rather limited, and does not allow the user to specify, for example, the minimum and maximum expectations. Second, better techniques for matching expectations need to be developed, particularly for expectations that are not expressed as simple real numbers. We envisage that some reasoning mechanisms will be required. Finally, there is a need to consider what happens when there are no matching expectations when assessing QoS, or the so-called "cold start" problem [25]. This is particularly interesting to consider in our quality model, as tightening or relaxing similarity measures in the model can have a direct impact on matchable expectations.

Acknowledgments. This work is supported by British Telecommunications plc, and we wish to thank the members of the CONOISE project team for their constructive comments on this work.

References

1. Casati, F., Shan, M.C.: Definition, execution, analysis, and optimization of composite e-services. Data Engineering Bulletin **4** (2001) 29–34
2. Leymann, F.: Web services: distributed applications without limits. In: Proceedings of 10th Conference on Database Systems for Business. (2003) 2–23
3. Piccinelli, G., Stammers, E.: From e-processes to e-networks: an e-service-oriented approach. In: Proceedings of 3rd International Conference on Internet Computing. (2002) 549–553
4. Rust, R.T., Kannan, P.K.: E-service: a new paradigm for business in the electronic environment. Communications of the ACM **46** (2003) 36–42
5. He, M., Jennings, N.R., Leung, H.: On agent-mediated electronic commerce. IEEE Trans on Knowledge and Data Engineering **15** (2003) 985–1003
6. Jennings, N.R., Faratin, P., Norman, T.J., O'Brien, P., Odgers, B.: Autonomous agents for business process management. International Journal of Applied Artificial Intelligence **14** (2000) 145–189
7. Papazoglou, M.P.: Agent-oriented technology in support of e-business. Communications of the ACM **44** (2001) 71–77
8. Petersen, S.A., Gruninger, M.: An agent-based model to support the formation of virtual enterprises. In: International ICSC Symposium on Mobile Agents and Multi-agents in Virtual Organisations and E-Commerce. (2000)
9. Aurrecoechea, C., Campbell, A.T., Hauw, L.: A survey of qos architectures. Multimedia Systems Journal **6** (1998) 138–151
10. Chalmers, D., Sloman, M.: A survey of quality of service in mobile computing environments. IEEE Communications Surveys and Tutorials **2** (1999) 2–10
11. Lee, Y.W., Strong, D.M., Khan, B.K., Wang, R.Y.: Aimq: a methodology for information quality assessment. Information and Management **40** (2002) 133–146
12. Parasuraman, A., Zeithaml, V.A., Berry, L.L.: Reassessment of expectations as a comparison standard in measuring service quality: implications for future research. Journal of Marketing **58** (1994) 201–230
13. Trzec, K., Huljenic, D.: Intelligent agents for qos management. In: Proceedings of First International Conference on Autonomous Agents and Multi Agent Systems. (2002) 1405–1412
14. Yu, B., Singh, M.: An evidential model of distributed reputation management. In: Proceedings of First International Conference on Autonomous Agents and Multi Agent Systems. (2002) 294–301
15. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J.: Quality driven web service composition. In: Proceedings of Twelfth International Conference on World Wide Web. (2003) 411–421
16. Mui, L., Mohtashemi, M., Halberstadt, A.: A computational model of trust and reputation. In: Proceedings of 35th Hawaii International Conference on System Sciences. (2002) 2423–2431
17. Mecella, M., Scannapieco, M., Virgillito, A., Baldoni, R., Catarci, T., Batini, C.: Managing data quality in cooperative information systems. In: Proceedings of 10th International Conference on Cooperative Information Systems. (2002) 486–502

18. Braynov, S., Sandholm, T.: Incentive compatible mechanism for trust revelation. In: Proceedings of First International Conference on Autonomous Agents and Multi Agent Systems. (2002) 310–311
19. Schillo, M., Funk, P., Rovatsos, M.: Using trust for detecting deceitful agents in artificial societies. Applied Artificial Intelligence, Special Issue on Trust, Deception and Fraud in Agent Societies (2000) 825–848
20. Maximilien, E.M., Singh, M.: Conceptual model of web service reputation. SIGMOD Record, ACM Special Interest Group on Management of Data (2002) 36–41
21. Maes, P., Guttman, R.H., Moukas, A.G.: Agents that buy and sell. Communications of the ACM **42** (1999) 81–91
22. Herlocker, J., Konstan, J., Riedl, J.: Explaining collaborative filtering recommendations. In: Proceedings of ACM 2000 Conference on Computer Supported Cooperative Work. (2000) 241–250
23. Burgess, M., Gray, W.A., Fiddian, N.: Establishing a taxonomy of quality for use in information filtering. In: Proceedings of 19th British National Conference on Databases. (2002) 103–113
24. Wang, R.Y., Strong, D.M.: Beyond accuracy: what data quality means to data consumers. Journal of Management Information Systems **12** (1996) 5–34
25. Schein, A., Popescul, A., Ungar, L., Pennock, D.: Methods and metrics for cold-start recommendations. In: Proceedings of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. (2002) 253–260