Knowledge Pump: Community-centered Collaborative Filtering

Natalie Glance, Damián Arregui and Manfred Dardenne

Xerox Research Centre Europe, Grenoble Laboratory

October 27, 1997

Abstract

This article proposes an information technology system we call the Knowledge Pump for connecting and supporting electronic repositories and networked communities. At the time of writing, we have a working prototype that we'll be ready to deploy soon within a first group of users. Our first goal is to achieve proof-of-principle: to show that community-centered collaborative recommendation can indeed support knowledge sharing and improve community awareness and development.

1 Introduction

This article proposes an information technology system we call the Knowledge Pump for connecting and supporting electronic repositories and networked communities. Our objectives are two-fold. The first is to facilitate getting the right information to the right people in a timely fashion. The second is to map community networks and repository content. These goals are complementary because the community and repository maps help channel the flow of information while the patterns inferred from information flow help refine the maps.

The aspect of Knowledge Pump on which we focus primarily here is its distribution capability. In particular, our first goal is to help communities, defined by their common interests and practices, more effectively and more efficiently share knowledge, be it in the form of must-read documents or new ways to get work done. We introduce a technique we call community-centered collaborative filtering. This technique combines statistical algorithms and heuristic rules with a community bias to guide the distribution of information based on explicit and implicit recommendations.

In the next section, we describe our first implementation of the Pump, and in Section 3 we conclude with a summary and outlook. A more complete elaboration of the system can be found in (1).

2 Implementation

In this section we present our first implementation of the Knowledge Pump. At the time of writing, we have a working prototype that we'll be ready to deploy soon within a first group of users.

2.1 Technologies and architecture

We had a few, basic initial design requirements: portability, ease of use and immediate value. Portability means one code set, all platforms, and suggested building something riding on top of the Web for a first implementation. Effectively, this pointed to Java, since HTML and scripting languages alone are too limiting. Portability also means not touching the browser: no plug-ins, for example, and no browser-specific capabilities, like cookies. In addition to that, we brought up an Apache HTTP server to access the system HTML pages and Java applets, and an mSQL shareware database engine to keep the whole system persistant data about users, documents, etc.

The Pump is implemented as a client-server system. The client is written in Java and runs off a Web browser. It talks to the Knowledge Pump server, also written in Java, which is responsible for a number of functions. The KP server provides an interface to system administration, periodically runs the community-centered collaborative filtering algorithm and builds the "What's recommended?" pages for each user. The pages are then saved and delivered to the user via the HTTP server. The database is accessed via yet another server.

In choosing to connect first to the Web, we've joined a well-populated playing field of WWW filtering efforts. However, we doubt that collaborative filtering over such a large domain can work well in an organizational setting: too many pages and too few reviewers. Our real goal is to connect to repositories which have Web interfaces. We're designing a plug-and-play interface to the Pump so that with minor modifications, any repository (Web front-end or no) can connect transparently with the Knowledge Pump. In the meantime, the Pump connects with any repository with a Web front-end in a less transparent, but still useful way. Currently, we are testing the plug-and-play principle by connecting the Pump with an electronic repository of scanned journal articles, (2).

2.2 Functionality: document management and recommendation

The user's interface to the Pump, shown in Figure 1 is through a small palette of functions. The *bookmark* and *search* functions are basic document management capabilities provided by the Pump. *Bookmark* allows the user to save a pointer to any Web page. (S)he rates the document on a five-point scale from "irrelevant" to "one of my favorites," and optionally types in some comments. The user also classifies the documents into any of a number of the listed communities. Finally, the user can save the pointer as "private" – for his/her eyes only – or as "public."

Figure 1: The user's palette of controls.

Complementary to the bookmark function is the *search* function, which allows the user to search for bookmarks classified into any number of domains according to date, title, author, rating, reviewer and private vs. public classification. The Pump delivers the search results as an HTML document to the Web browser. For each pointer satisfying the search criteria, the page includes the predicted or actual rating and provides a hyperlink to the comments associated with the pointer.

The *profile* function allows the user to enter or modify his/her personal profile. Here the user selects his/her "advisors" from the list of Knowledge Pumpers. Advisors refer to people whose judgement the user particularly respects – this list is used by the community-centered collaborative filtering mechanism described further in Section 2.3. The user also selects any number of domains of interests from the hierarchy of "communities." Recommendations by the Pump to the user will be sorted according to this identified set of domains.

The *What's recommended?* function brings up the Pump's most recent personalized list of recommendations sorted by category as an HTML document in the Web browser. An example recommendations page is shown in Figure 2. If the user keeps the recommendation page open in the Web browser, the Pump periodically and automatically updates it. In the current implementation, each recommendation takes the form of a pointer to a URL. The Pump only recommends items which the user has not seen before (not to the Pump's knowledge, at least) and for each item, displays the Pump's prediction of the user's interest as a number of stars, lists the names of all reviewers, and provides a link to their comments. The user can prune the recommendations page by deleting entries and can also review items directly from the page.

The panel to the right of the recommendations page contains a set of *gauges*, as we call them. These gauges reflect the activity level of the Pump. There is a gauge displayed for each community to which the user belongs. The INFLOW half of the dial indicates how many recommendations are flowing in per person per week for the community. In black is the community average; in red is the individual inflow. The OUTFLOW part of the dial indicates how many recommended links are being followed per person per week. Once again, black represents the community average; red, the individual outflow. The gauges give feedback on how the average level of activity in a community fluctuates over time and give users a feel for how their level of participation compares with community averages.

2.3 Community-centered collaborative filtering

The Knowledge Pump uses what we call community-centered collaborative filtering to predict a user's level of interest for unread items in each of the users' domains of interest. This mechanism combines elements of social and content-based filtering.

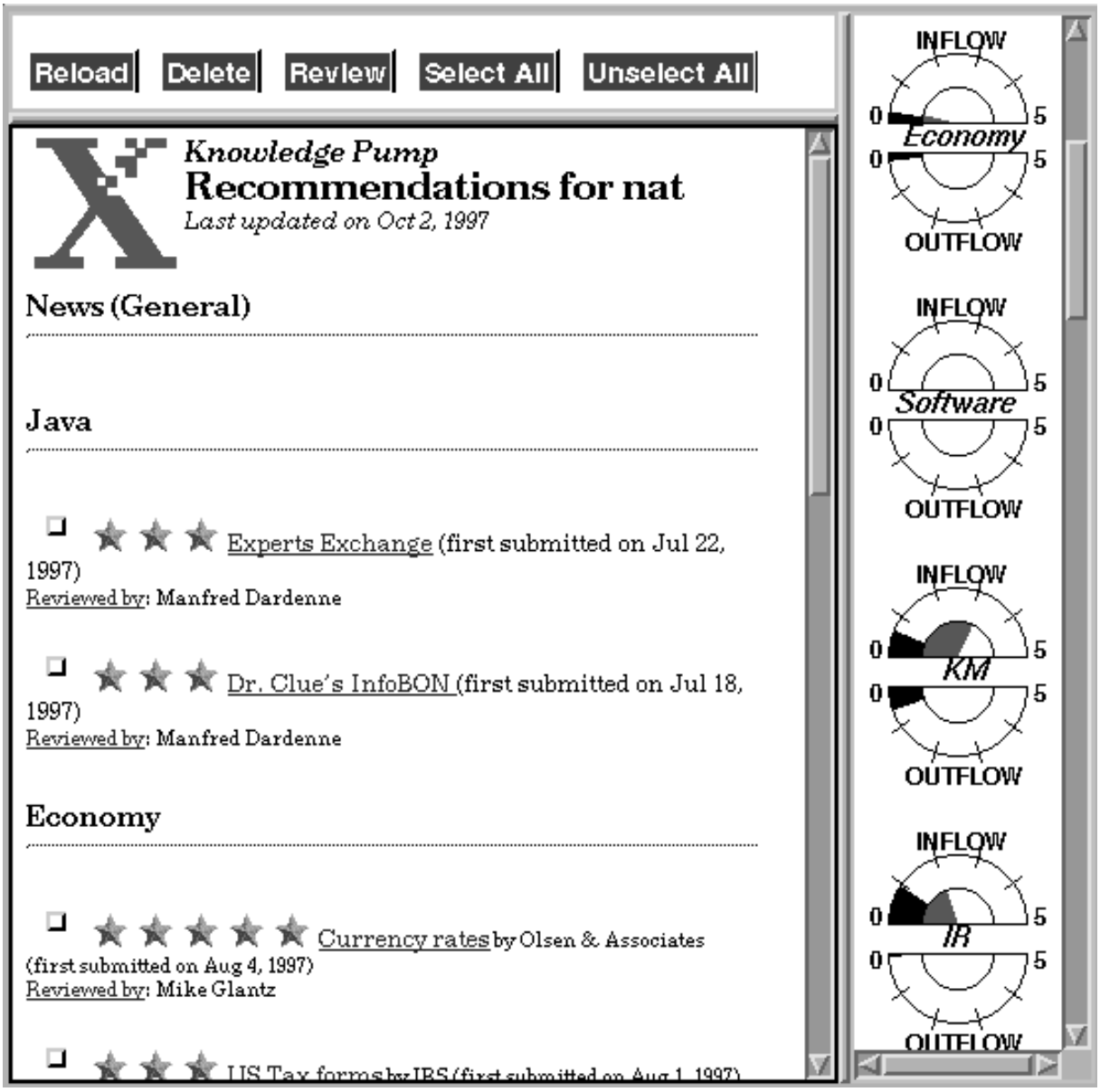

Figure 2: An example of "What's recommended?" by the Pump.

For the moment we rely on recommenders to classify items into a commonly agreed upon classification scheme. This could be complemented down-the-line by automatic categorization via statistical classification algorithms.

The second layer of social filtering – matching items to people by first matching people to each other – lies on top of the initial classification by domain. It's important to filter by content first and by social relationships second because similarities among people tend to vary greatly across different domains. For example, the authors of this article have similar rankings of the most influential knowledge management gurus, but wildly different opinions concerning the best guitar players alive. Social filtering over all domains at once tends to wash out the differences in people's similarities to each other.

Social filtering via automated collaborative filtering is based on the premise that information concerning personal relationships are not necessary. In principle, we agree, because once an automated collaborative filter has collected enough information about its users, it can work very well. In practice, however, automated collaborative filters suffer from the cold-start problem: without large amounts of usage data, they work very poorly, which discourages the usage that would overcome this lack.

In contrast, in community-centered collaborative filtering, the collaborative filter is bootstrapped by the partial view of the social network constructed from user-input lists of "advisors" – people whose opinion users particularly trust. Bootstrapping the system in this way allows the collaborative filter to perform well from the start, weighting

higher the opinions of his/her most trusted contacts when predicting the user's opinion on items. Over time, as more usage data is collected, the weight given to automated (statistical) portion of the collaborative filter can be increased relative to the weight given to advisors' ratings.

Statistical algorithms can then mine the usage data to automatically refine the Pump's view of the social network and visualize it for the users. This sets up a feedback loop between users and the collaborative filter: on their end, users collectively (re-)describe the social network; on its end, the Pump automatically refines and visualizes the social and community maps from usage data.

Item #	Alice	Bob	Chris	Dave
1	5	?	3	4
2	?	1	5	2
3	?	4	2	?
4	0	?	1	?
5	?	?	3	3

Figure 3: A sample user-item matrix of ratings.

From a mathematical standpoint, collaborative filtering within a given domain can be viewed as matrix filling, where the rows of the matrix are items recommended into the domain, the columns are the people who have reviewed an item in the domain, and the cells contain the ratings submitted. An example is shown in Figure 3.

The prediction algorithm used by the Pump is a weighted sum of three components:

- the average population-wide rating;
- the average over advisors' ratings;
- the correlation-weighted sum of all ratings.

The first two components are straight-forward and are very important when ratings are very sparse, for example, when the system is first deployed. The second component uses the elements of the social network revealed from user-input lists of advisors.

The third component is a standard automated collaborative filter (see (3), for example), which can be implemented in any of a number of ways. Our implementation first calculates person-person correlations from previous recommendations. These correlations indicate how much two reviewers tend to agree with each other on the items they both rated.

The collaborative filter automatically weighs more heavily the ratings of users that historically tend to agree with the user in question and discounts the ratings of those that tend to disagree. It is most effective when the user-item matrix is densely filled.

Currently, we use heuristics to combine the three components into one prediction. The heuristics take into account how long the system has been in place and the confidence level of each of the three elements. The confidence level is simply an ad-hoc estimate based on the density of ratings in the respective three populations. Once we have a user base established, we'll be able to test the effectiveness of our approach and of the confidence level estimates and refine them for future use.

One last element of community-centered collaborative filtering is related to the user interface: users see the names of the people who have recommended an item and can read the publicly-available comments. This makes the boundaries between communities more permeable. A user can classify an item into any domain, regardless of whether (s)he considers him/herself as a member of that community. Thus, members of a community can receive recommendations from people outside the community. Over time, the person can explicitly join the community by changing his/her profile or may become a de facto participant in the minds of its members.

3 Summary and outlook

As we discussed earlier in this article, we believe that the key to sucessful knowledge sharing is focusing on the community. We've implemented a first version of the Knowledge Pump that attempts to leverage community

currency in the form of reputation, trust and reciprocity to create incentives for sharing recommendations. At the heart of the Pump is a recommendation distribution mechanism we call community-centered collaborative filtering. This mechanism matches items to people by first matching people to each other, giving extra weight to trusted advisors.

The implementation of the Knowledge Pump as described in the previous section is a work in progress. Our first goal is to achieve proof-of-principle: to show that community-centered collaborative recommendation can indeed support knowledge sharing and improve community awareness and development. This first prototype is intended to provide the minimal set of functionality sufficient to make it acceptable for use within an environment of early adopters.

However, understanding the environments in which the Pump could be used will be vital in order to tailor its functionalities and create incentives for use. For something like the Knowledge Pump to successfully support the flow and use of knowledge in organizations, it will have to become a seamless part of the way people do their work. The social aspects of use are perhaps the most fascinating and the most challenging.

4 Acknowledgments

The authors thank Daniele Pagani and Dan Holtshouse for inspiring this work and for their continued support. We are also grateful to Stefania Castellani, Antonietta Grasso, David Hull, Francois Pacull, Remo Pareschi and our other colleagues at XRCE for their suggestions and feedback.

References

[1] N. Glance, D. Arregui, and M. Dardenne, "Knowledge pump: Supporting the flow and use of knowledge," in *Information Technology for Knowledge Management* (U. Borghoff and R. Pareschi, eds.), ch. 3, Springer-Verlag, 1998.

[2] URL. Calliope: http://www.xrce.xerox.com/ats/digilib/calliope/.

[3] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the Conference on Computer-Supported Cooperative Work*, (Chapel Hill, NC), pp. 175–186, ACM, 1994.