Recommenders Everywhere: The WikiLens Community-Maintained Recommender System

Dan Frankowski, Shyong K. (Tony) Lam, Shilad Sen, F. Maxwell Harper,
Scott Yilek, Michael Cassano, John Riedl

GroupLens Research, University of Minnesota
4-192 EE/CS Building, 200 Union Street Southeast
Minneapolis, Minnesota, 55455 USA

{dfrankow,lam,ssen,harper,syilek,mcassano,riedl}@cs.umn.edu

Abstract

Suppose you have a passion for items of a certain type, and you wish to start a recommender system around those items. You want a system like Amazon or Epinions, but for cookie recipes, local theater, or microbrew beer. How can you set up your recommender system without assembling complicated algorithms, large software infrastructure, a large community of contributors, or even a full catalog of items?

WikiLens is open source software that enables anyone, anywhere to start a *community-maintained recommender* around any type of item. We introduce five principles for community-maintained recommenders that address the two key issues: (1) community contribution of items and associated information; and (2) finding items of interest. Since all recommender communities start small, we look at feasibility and utility in the *small world*, one with few users, few items, few ratings. We describe the features of WikiLens, which are based on our principles, and give lessons learned from two years of experience running wikilens.org.

Categories and Subject Descriptors

H.4.3 [**Information Systems Applications**]: Communications Applications]---*information browsers*. K.4.3 [**Computers and Society**]: Organizational Impacts---*computer-supported collaborative work*.

General Terms

Design; Experimentation; Human Factors.

Keywords

community-maintained, member-maintained, recommender systems.

1. Introduction

As of November 2006, many top sites on the web are built from or prominently feature user-contributed content. For example, MySpace, eBay, Amazon, YouTube, craigslist, and Wikipedia comprise six of Alexa's top ten sites for the United States by web traffic. Some of these sites didn't exist even 5 years ago. The rapid growth of user-contributed content is possible due to inexpensive storage, inexpensive network connections, increasingly powerful computers, and people eager to share content with others.

As of November 2006, the Internet Movie Database[1] has over 880,000 movie or TV show episodes in its database, and Wikipedia has over 1,400,000 articles in English alone. Even Beeradvocate.com, a more specialized site, lists over 32,000 beers. You could drink a different beer every night for the rest of your life! There is far more information online than any person could ever process, which makes finding information of interest difficult. Researchers call this problem *information overload*, and recommender systems are a popular solution to the problem. *Recommender systems* (or *recommenders*) [1] suggest items of interest based on a user's preferences, behaviors, and information about the items themselves.

However, recommenders are rarer than other types of community sites, like forums or blogs. PhpBB[2], popular online forum software, was downloaded from SourceForge over 9 million times, and Technorati.com tracks almost 60 million blogs. By contrast, there are only dozens of online recommenders for books, movies, music, web sites, and so on. Wikipedia's "Collaborative Filtering" page lists 27 commercial web sites, 26 non-commercial sites[3]. As further evidence of rarity, there are no systems or hosted services to support recommenders. Rather, there are algorithms [1][16] or libraries[4] or web services[5] from which you may assemble your own site. We are drowning in information, but thirsting for recommenders to help navigate it.

WikiSym'07 October 21–23, 2007, Montréal, Québec, Canada.

[1] http://www.imdb.com

[2] http://phpbb.sourceforge.net

[3] Many recommender systems are based on *collaborative filtering* algorithms that produce recommendations using the assumption that similar users have similar tastes.

[4] See http://en.wikipedia.org/wiki/Collaborative_Filtering

[5] For example, http://www.easyutil.com

Taking a cue from successful web sites with user-contributed content, we propose recommenders everywhere using a *community-maintained recommender* that allows users to contribute content as well as the information necessary to recommend that content. Such a system would support the activities of *recommender communities*, groups of people who come together to recommend things.

One new research challenge for the recommenders everywhere project is creating *small world recommenders.* Most online communities are small, and are based on groups of people who get to know each other over time. By contrast, the best-known recommender algorithms evolved from large e-commerce communities, so these algorithms rely on statistical predictions based on ratings from a largely anonymous community. Small world recommenders have four key aspects: (1) providing value to users even with very little preference data per item, (2) aggregating user preferences into recommendations, (3) allowing users to see specific individuals' preferences for items, and (4) depending on users to understand the relationship of their preferences to those of other users. Small-world recommenders help a user overcome the scarcity of preference data. They supplement aggregate recommendations with a user's understanding of other users and their preferences.

A community-maintained recommender has similarities to discretionary databases [19] or public document repositories [15], although it would also have evaluation information like ratings or reviews as well as documents. Thorn and Connelly [19] predict that "discretionary information will be chronically undersupplied." However, systems like Wikipedia, YouTube, and MySpace belie that prediction. Other social theories try to explain this success. For example, *critical mass theory* [14][15] says that a small number of contributors can make a big difference, especially in the face of increasing marginal returns (when contributions count for more and more). Theorizing which community-maintained recommenders have increasing marginal returns is interesting, but beyond our scope.

This paper explores the design and implementation of WikiLens, a community-maintained recommender that communities could use as a hosted service or install for themselves. We propose a set of principles for supporting a recommender community (section 2), describe the design of the WikiLens system to support those principles (section 4), and report lessons we learned from our experiences in running WikiLens installations (section 5). We focus mostly on wikilens.org, a public website, but also have some lessons from two smaller, private installations.

Community-maintained recommenders are desirable and may be feasible. Now let us explore some proposed principles with a parable.

2. Supporting Recommender Communities

Suppose you have a passion for beer. You are probably not the only one. In fact, you wish to participate in an online community about beer. First, you seek an existing web site and find Beeradvocate.com. However, they don't have your favorite brew (served only in your neighborhood pub), and they won't let you add beers until you've reviewed at least 20. At some large community sites, especially commercial ones, the barriers are even higher: at Epinions.com you have to become a Category Lead, selected by Epinions from a nomination process that includes evidence of enthusiasm and passion, and even phone interviews. At least they allow adding items eventually. At Amazon, you can add reviews, but cannot add new items. There are other communities that let you add items right away, but not about beer: YouTube (videos), flickr (pictures), digg (news articles), last.fm (music), StumbleUpon (web sites).

So you decide to start your own community. *How should this community work to be effective at feeding and harnessing the community passion for finding items of interest?* The rest of this section proposes answers to this question in the form of organizing principles. Inspired by the success of sites with user-contributed content, we postulate principle #1:

ADD: *Members should be able to add items immediately.*

Adding beers has more than simple moral satisfaction. Anderson [3] says that people like many more things than just the most popular. He calls the little-known things the "long tail" and argues that technologies or businesses that help people find those things are valuable and revolutionary: "the market for books that are not even sold in the average bookstore is larger than the market for those that are."

Moreover, Cosley et al. [6] found that members who could see the results of their contributions immediately did more work than those who saw results only pending review. Thus, we propose that members be able to see their additions right away.

So you, our intrepid beer lover, could try a forum or a blog. There are plenty of software packages or free services. However, you want a community with in-depth information about beers. Adding beers as forum threads or blog posts is unsatisfying. The basic organization is discussion-based, not beer-based. Hosted online community software like Yahoo or Google Groups has similar disadvantages. This inspires principle #2:

DEEP CHANGE: *Members should be able to uniquely identify items, and define and redefine their attributes and organization.*

Each beer should be a uniquely identifiable entity in the system, and also allow associated details (call them *fields*) such as style, brewer, or alcohol content. Furthermore, for usability's sake, fields should be viewed and edited in a

visually consistent manner, using widgets people expect like appropriately sized text boxes, radio buttons, drop-downs, and so on. It might also be desirable to be able to more closely link items to discussions as explored in [8], but here we focus on the item database.

Our beer lover may be interested in any sort of item or even multiple sorts—say pubs and beer magazines. Thus, the system should support adding new categories of items, changing item details, and adding fields to items.

Content Management Systems (CMSs) try to address such problems in a structured way, and some (like Drupal) even have recently added rating and recommendation modules. However, CMSs are most often used for the few to broadcast to the many. Few people can add items. Even fewer people can change structure (say, by giving items different fields, or adding new categories). An alternative, Google Base[6], allows posting items with arbitrary fields, but lacks features to allow a community to collaborate on posts, and its listings are ephemeral.

However, our beer lover wishes to unleash community creativity, allowing members to produce interesting and unexpected applications. Many people's first inclination is that information on a site is better controlled by a privileged few. Indeed, surveys in [6] indicate people prefer expert oversight of contributions. However, that same study also showed that despite peoples' preferences, member oversight produced similar quality and quantity. Since there are often more community members than experts, allowing members to do more work gets more done. Further, the quality of the resulting database is high as long as other members can review that work.

Allowing members to have control over the site may allow the high contributors to step forward and start work without delay. Finally, allowing members to make deep changes to the site, like adding item fields or adding categories, may allow the site to be more reflective of community desires, and hence produce higher satisfaction and commitment.

Some of the most successful community-maintained sites reflect the DEEP CHANGE principle. For instance, Wikipedia's community adds items to categories, changes uniform display templates, and even builds bots and tools to assist users.

On the opposite end of the spectrum, users may wish to try a little bit before devoting a lot of effort to a system. Principle #3:

MICRO-CONTRIBUTE: *Members should be able to make small contributions.*

Many people may be willing to make small contributions, especially ratings. Netflix[7] (a movie rental website) has a dataset with thousands of movies, but over one billion ratings[8]. Perhaps this is because people find rating fun. In a survey[9] of 357 MovieLens[10] users, 193 (54%) said one of their top 3 reasons to rate movies (of 8 possibilities) was because it is fun. Ratings also support important tasks, such recommending or evaluating items.

Micro-contribution may also motivate casual contributors, who can be very important to the community. Often, a few people contribute a lot, and a lot of people contribute a little bit. The top 1,009 reviewers at Amazon produce a disproportionately high 257,773 reviews (with a median of 148 reviews per reviewer), but that is still a small fraction of the total of 3.4 million reviews (with a median of 1 review per reviewer) [15]. It is an open question whether more Wikipedia contributions are made by large contributors or small ones. Jimbo Wales points out that a small core community make most of the edits, but Aaron Swartz claims occasional users may contribute more content overall[11]. The debate hinges on the size of contributions, which Swartz argues are often much larger for the occasional contributor. Moreover, *legitimate peripheral participation* (small starter tasks) may be a path for a casual contributor to become a heavy one [4].

Our beer lover could try a wiki, which has many of these characteristics. Each beer could be a page, and some wikis, like MediaWiki, support templates that have visually consistent details, although not familiar editing widgets. In related research, Völkel et al. [21] propose ways to add arbitrary attributes and relations to a wiki, but again not using familiar editing widgets or consistent visual display. Supporting micro-contribution requires simple, direct interfaces for adding information.

However, as members contribute there will rapidly be many beers, resulting in information overload. Remember, Beeradvocate.com has over 32,000. How will members find beers they will like from the plethora of available beers? Principle #4 is

FIND: *Members should be able to find items that interest them.*

A member is "interested in" a beer if they are convinced by available information to wish to learn more about it, perhaps by reading about it on the manufacturer's web

[6] http://base.google.com

[7] http://netflix.com

[8] http://www.netflixprize.com/faq

[9] http://grouplens.org/data/mlsurvey0604.html

[10] http://movielens.org. A movie recommender system run by GroupLens Research.

[11] http://www.aaronsw.com/weblog/whowriteswikipedia.

page, studying the label in the shop, or drinking it. FIND is *information filtering*: selecting items of interest from a larger set of possibilities. Even a small-world recommender can contribute to successful information filtering; in that case, the "larger set" is in the world.

Malone [13] summarizes three approaches to information filtering based on interviews of office workers: *cognitive* (content-based), *economic* (cost/benefit-based), and *social* (other-people-based). We may wish a system to support all of them.

Cognitive filtering may be served by tools operating on factual details or descriptions. Organization (such as categories or tags) is likely useful. Economic filtering is when a user tries to estimate cost and benefit of consumption from indirect clues, like message length or sender. In a recommender, cost and benefit might be estimated using clues such as popularity or even the actual monetary cost.

What about social filtering? Before going online, our beer lover learned about beer from his friends. Our intuition agrees: we often seek recommendations from those we know. Principle #5:

SEE OTHERS: *Members should be able to see each other and their contributions.*

Several researchers theorize that social information is important in our context. Erickson et al. [9] suggest that "social translucence (systems supporting visibility, awareness, and accountability) is "a fundamental requirement for supporting all types of communication and collaboration." A community-maintained recommender would be a collaborative system. Seeing others might motivate contributions or social conventions. Schafer et al. [17] speculate that showing individual ratings or reviews may be particularly helpful in small communities.

These five principles serve two high-level goals. (1) ADD, DEEP CHANGE, and MICRO-CONTRIBUTE are about contribution: entering information into the system, either for oneself (memory, self-expression), or for others (buddies, the community, the world); (2) FIND and SEE OTHERS are about exploration: extracting information in the system for some useful purpose, like finding new things, making decisions, or remembering.

3. Why A New System?

The preceding parable showed our beer lover encountered systems that don't support recommendations, or are not for beer, or are editorially controlled. There are libraries or web services to assemble your own recommender, but they do not address user interactions and system design. Someone who wishes to lead a special-interest, possibly small community probably wants an off-the-shelf system.

Well-known tools or hosted services propelled the popularity of blogs and wikis. Where is the phpBB or Wikipedia for recommenders? We see none yet. Moreover, we have seen little research into building tools for collaboratively maintained repositories with recommendations (e.g., [10]), although there has been study of recommending work in wikis [7].

4. System Design

WikiLens is an open-source recommender system we built to investigate our principles in practice. We modified PhpWiki, a popular open source wiki software package written in PHP. A *wiki* is online software invented by Ward Cunningham in 1995 [22] that displays pages and allows users to edit any page or add new pages, and see their changes immediately. Users edit *wiki markup*, a simple text markup language that is translated to HTML when it is displayed. Wiki markup in PhpWiki can include plugins written in PHP, which may perform operations on the wiki database and produce arbitrary output. This output is not directly editable by users. All versions of each page are saved so users can recover from mistakes or vandalism. Viégas showed that a common type of vandalism in Wikipedia was often reverted within minutes [20]. We chose to modify a wiki because it supports user contribution in a simple and robust way. We chose PhpWiki because it is open source and fairly popular (150-200 downloads per day).

In WikiLens, *an item is represented by a wiki page*. The wiki page collects all information in the system about the item (e.g., name, details, ratings, comments), as we describe below.

Let us examine WikiLens features added to PhpWiki, organized by our principles.

ADD. In WikiLens, any user may immediately add or edit pages, hence items. Users can also *import* items in categories for which we have written an importer plugin: Book, Album, Restaurant. To import, a user finds the item in a different system (say, Amazon or Chefmoz), pastes its URL into WikiLens, and clicks an "Add Book" or "Add Restaurant" button. WikiLens creates the appropriate page using information from the provided URL.

DEEP CHANGE. An item may optionally have a *category* (say, Restaurant), which dictates its *structured data fields* (or *fields*). A category is simply a wiki page which is in the "Category" category. Thus, categories may be created by any user. Also, a category may have a fields definition page (Figure 1).

```
* ADDRESS
  * Name: Address
* CITY
  * Name: City
  * display_as_subcategory: true
* ZIP
  * Widget: textbox(size=5,maxlength=10)
  * Name: Zip code
* CUISINE
  * Name: Cuisine
  * Widget: checkbox
  * Options
    * American
    * Bar / Pub
    … (some lines omitted)
```

Figure 1. Fields definition page for a Restaurant category.

Defining fields for a category has several effects. First, field names and values (possibly blank) are displayed on each item page in the category (Figure 2). Second, field names and edit widgets are displayed when editing an item page (Figure 3). Third, item importers (written in PHP) may refer to the fields, and fill them where possible. For example, Amazon provides book author, and our Book importer can put that information in the Book field 'Author'.

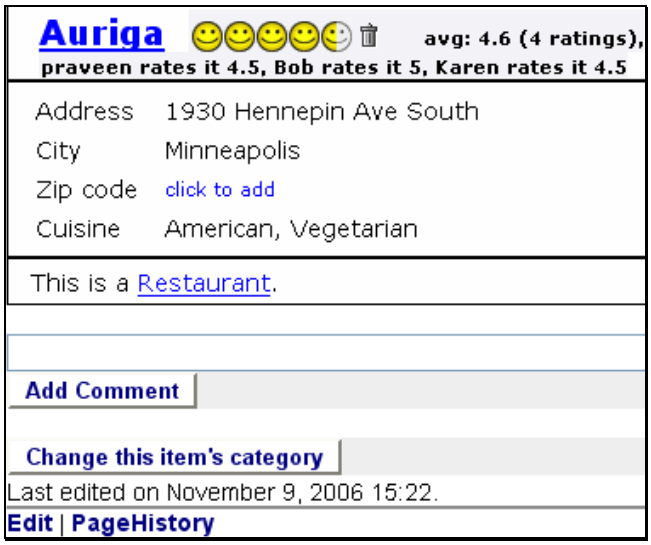

Figure 2. Display of Auriga, a page with fields from the Restaurant category. Zip code is blank.

Field definitions are written in a simple indentation-sensitive language. Each field has a name, at the leftmost indentation level. Field properties are indented one level. Possible properties are:

- Name: the display name of the field
- Widget: the widget used to display the field: 'textbox', 'textarea', 'radio', 'checkbox', 'dropdown', 'select', 'hidden', or 'listitems'. The default Widget is a textbox.
- Options: possible values for restricted-choice widgets (radio, checkbox, dropdown, select).
- `display_as_subcategory`: displays simple filtering tools (see the category page filtering description below)

The widgets correspond directly to common HTML widgets, except for 'listitems' (see the lists description below).

If a category has no fields definition page associated with it, its items have no fields.

We built special editors to change an item category or to change which page is a category's field definitions page. That is, that information is not displayed in the normal wiki text editor. We started with it in specially formatted wiki text, but were concerned that people would accidentally erase or change it without realizing the implications.

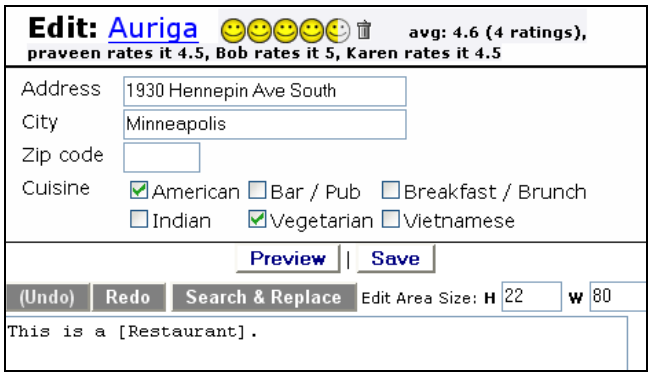

Figure 3. Editing Auriga, a page with fields from the Restaurant category.

This raises a general question: which information should be in the wiki text, and which in specialized editors? We decided to make special editors when it protects users from making big mistakes that are hard to recover from. An item's category may be recovered since it is saved with each page version—simply revert the page. However, if you change an item's category, the item will no longer show up in that category, which may make it hard to find. Similarly, if you change the fields definition page associated with a category, all of the items in that category will show different (or possibly no) fields. By this principle, it might be best to have a special editor for the fields definition page itself as well (e.g., "Are you sure you wish to hide 4,037 field values with this change?"), but for expediency we didn't build it.

Specialized editors move the system closer to a CMS in some ways, but still with the wiki semantics of allowing members to edit most things and have vandalism or error protection (recent changes and being able to revert to previous versions), and still with the semantics that most changes are "editing a page," possibly through a special editor.

Finally, item category, category field definitions, and item fields may be edited by any member. They are in the version data of the appropriate page, hence may also be reverted to undo mistakes.

MICRO-CONTRIBUTE. We expected some features to be used by almost every member (e.g., ratings and finding

things), some only by the more dedicated (e.g., page editing), and some only by those knowledgeable about the system itself (e.g., category editing). We called these 100% features, 10% features, and 1% features, respectively, referring to the likely fraction of members that would use each feature. We felt designing a feature for a wider audience required more thought and engineering to make it easy, so we tried to guess the audience for a feature before building, and work harder at the features with a wider audience.

Inspired by [11], to allow another type of micro-contribution, we put an "Add Comment" widget at the bottom of a page (Figure 2): type in a comment, click the "Add Comment" button (or hit return), and it inserts the comment at the bottom of the page text.

FIND. One can imagine many mechanisms for information filtering. We believe it has two parts: 1) *selecting* items to evaluate, which may include browsing, filtering (removing possibilities), and ordering (focusing attention on certain possibilities); 2) *evaluating* the items that were selected to verify that they are indeed interesting.

A good way of selecting or evaluating items is with user ratings. Items are pages, and users rate pages on a scale of one-half to five smiley faces (Figure 2). Recall that users also provide the items to be rated. Sometimes a user wished to create a list of options on a single page (a *poll*) and have others rate each option. To support this, we also allow wiki markup (Figure 4) to place ratings widgets into the middle of a page (Figure 5). We call this "Ratemania." Often the user's options did not require rigorous fields, so it seemed onerous to require a page for each option. Thus, a ratings widget can refer to a page, or to an abstract identifier that does not refer to any particular page.

```
<?plugin RateIt caption="highrider"
urn="grouplens:printernames:highrider" stats="true"
expandable="true" ?>
```

Figure 4. Markup for Ratemania ratings widget in Figure 5.

Users may also have buddies. As in a typical social network system, a user A requests another user B to be a buddy, and B may accept the request, after which each user is a buddy of the other. They can see ratings of their buddies under certain circumstances, such as in a Ratemania widget in its expanded state (Figure 5), or a page ratings widget (Figure 2).

Surowiecki [18] says that an important feature of harnessing collective wisdom is that people should contribute information independent of each other. Otherwise people can get sucked into an *information cascade*, where they agree with each other instead of rendering independent judgments, and information is lost. Cosley et al. [5] supports this point, showing that seeing predictions can affect rating behavior. Thus, the `expandable` option for a ratings widget hides buddy ratings or statistics until the rater clicks "more", preferably after they've rated.

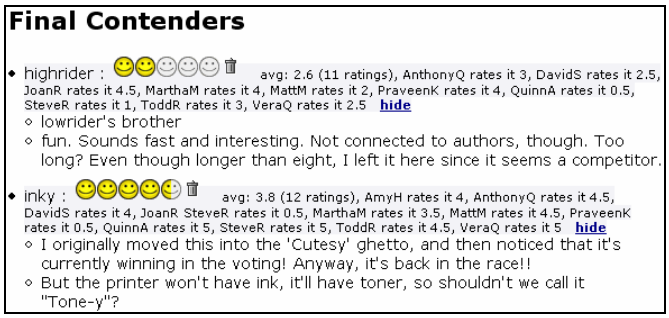

Figure 5. Ratemania ratings widgets for printer names, expanded to show extra information.

Each category page is a hub of different activities. A category page (e.g., "Book") usually has a CategoryPage plugin on it that shows (1) items in the category (Figure 6), sorted by prediction value, (2) some filtering links (hiding or showing rated or unrated items or values of fields marked `display_as_subcategory` true, such as City in Figure 1), (3) directions on how to search for and add items, and (4) directions on how to set the Category fields definition page, and other category maintenance activities. There are also usually directions on how to import an item, if importing is possible for the category.

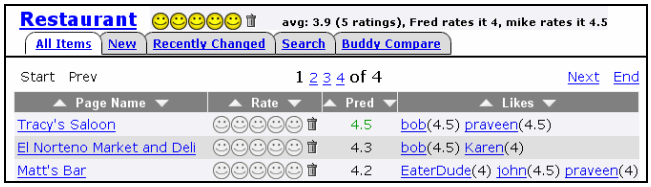

Figure 6. Top of the Restaurant category page, with recommendations, ratings widgets, buddy likes.

We tried several prediction algorithms that included information about buddy ratings. We wanted an algorithm with three properties: (1) predicted values on the same scale as the ratings, (2) higher predicted values if buddy ratings were higher, and (3) higher predicted values if items were rated by many buddies. The primary challenge is that most items have only a few buddy ratings. An algorithm that used only buddy ratings had noisy predicted values. Another algorithm that assumed an average rating for a buddy's missing rating yielded predictions near average for users with many buddies.

The algorithm we chose for predicted values is based on buddies' ratings, but includes some influence from the community average. Precisely, a prediction p for a subject user s and item i is 0 if the item has been rated fewer than 3 times, or

$$p = \bar{r}_s + \frac{|B_i| \left(\frac{\sum_{b \in B_i} w_b s_b (r_{b,i} - \bar{r}_b)}{\sum_{b \in B_i} |w_b s_b|} - P_b \right) + 0.5 \frac{\sum_{u \in U_i} r_{u,i}}{|U_i|}}{|B_i| + 0.5}$$

where

- B_i is the set of buddies that rated the item i
- U_i is the set of users that rated the item i
- $r_{u,i}$ is user u's rating of item i
- $\bar{r}_u$ is the average rating of a user u
- w_b is the Pearson correlation between the buddy b and the user who is receiving the prediction
- s_b is a significance weighting $\max(C_b/50,1)$ where C_b is the number of co-rated items between the user and the buddy
- $P_b = 1/2^{|B_i|-1}$ if the user has more than two buddies, otherwise 0

While this looks fearsome, the sum on the left is simply a traditional user-based k-nearest-neighbor prediction with significance weighting, the summation on the right is simply the item average, P_b slightly penalizes a prediction if a user has many buddies few of whom have rated the item, and the two sums are weighted and averaged together.

Finally, the "likes" column in Figure 6 shows buddies that rated the item above their own average, along with their ratings.

Another way of finding items of interest is by looking at user pages. Each user has a page where you can see their ratings, buddies, lists, and profile information, unless they have marked such information as private in their preferences.

There are three ways to search based on content: an open search box that searches page titles, a parameterized search within a category that allows the user to specify particular fields of interest (Figure 8), and one-click filtering by fields marked `display_as_subcategory` (Figure 1) or by category on the user pages. Clicking on a value filters the category page items to only those that have that value.

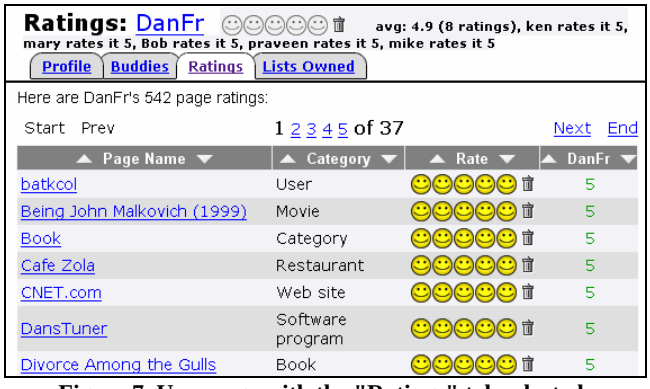

Figure 7. User page with the "Ratings" tab selected.

Figure 8. Parameterized search, generated from fields.

WikiLens also supports user-created lists of items. At its simplest, a list could just be a normal wiki page with either bulleted or term-definition lists. However, pages in the List category have special structure. A field of "listitems" type comes up in the page editor with a bit of extra help: a widget to add items to a list (Figure 9). There is also a spot on every page to add that page to an arbitrary list (Figure 10). WikiLens displays on each page any lists of which that page is a member.

A list is an explicit (*forward*) specification of a group of items all in one place. A *backward* grouping is editing each item to declare its membership in a group, much like categories. One can also imagine backward grouping with tags, which would allow an item to be in multiple groups.

One key question is whether items should be forward grouped through lists, or backward grouped through categories. We hypothesize that forward might be more useful for small groups or when you are declaring the whole group at once, while backward might be more useful for large groups or if the group is declared as you browse items.

WikiLens also has features supporting evaluation of interest in items the user selects: item details, prediction values, average ratings, buddy ratings, comment text, and also the normal wiki page text, which users may edit freely.

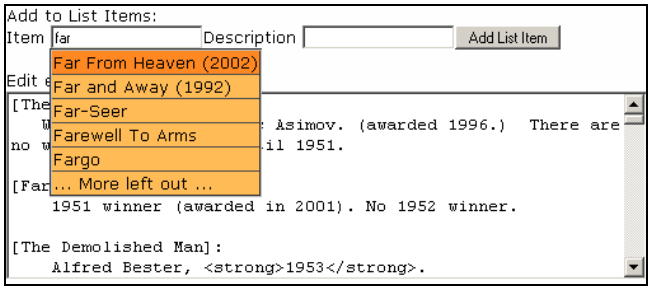

Figure 9. Adding items to a page in the List category.

Figure 10. The list widget (left) shows this page is on the "Hugo Award" List. It also allows the user to add the page to a list, with auto-completion of the user's existing lists (right).

SEE OTHERS. There are several wiki mechanisms for seeing others. A "recent changes" link shows pages that have been added or edited. Each page has the history of who edited it. There are also mechanisms we added: user pages, with profiles and ratings (Figure 7); and buddies, who allow you to see their ratings.

People can also affect each others' contributions. WikiLens' wiki features enable changing page content, including item field values, other users' comments, and the category of which a page is a member. Other users' ratings are not changeable, as they are intended to be owned by a particular user. Comments are also perhaps owned in this way, but it was expedient to place them in page text, and there is wiki precedent for this. We have not had problems with comments being maliciously edited.

Although we described features by the main principle they supported, each feature supports multiple principles (Table 1).

5. Experiences

In this section, we describe our experiences with the WikiLens software on wikilens.org, a public semi-anonymous web site, and two installations for private groups who knew each other (our research group GroupLens, and a book club).

Table 1. How features support design principles.

Feature	Principles
Item is a wiki page	All: ADD, DEEP CHANGE, MICRO-CONTRIBUTE, FIND, SEE OTHERS
User-maintained structured data	DEEP CHANGE, FIND, MICRO-CONTRIBUTE
Import	ADD
Ratings	FIND, MICRO-CONTRIBUTE, SEE OTHERS
Comments	FIND, MICRO-CONTRIBUTE, SEE OTHERS
Lists	FIND, MICRO-CONTRIBUTE, SEE OTHERS
Category page	ADD, FIND, MICRO-CONTRIBUTE, SEE OTHERS
Open search, Parametric search, Filtered browse	FIND
Recent changes	SEE OTHERS
Recommendations	FIND, SEE OTHERS
Buddies	FIND, SEE OTHERS
User pages	FIND, SEE OTHERS

The usage statistics we describe are from wikilens.org, gathered from April 13, 2004 to October 22, 2006 (about 31 months), with more detailed usage logging from May 3, 2005 onward (about 18 months). The amount of contribution (ratings and pages) was heavily influenced by individual users, hence it varies widely, and it is hard to say whether the trend was more or less usage over time. The site was open to anonymous contributions (adding items, even rating items) until January 13, 2006, when we required registration in order to combat wiki spam attacks. Anonymous users show up as IP addresses (e.g., 128.101.35.68) in the logs. We cannot tell how many anonymous users are behind an IP address, so we report the number of distinct IP addresses ("IPs") in addition to the number of distinct logged-in visitors in Table 2. We do report the number of items, ratings, and comments made by anonymous users ("Anon.") in column 2. There is also a lot of anonymous browsing, but we do not report it because we do not know how much of it is bots like search engine web crawlers.

Table 2. wikilens.org basic contribution statistics.

Statistic	All (Anon.)	GroupLens	Paper Authors	# distinct users
# users	231 (+36 IPs)	26	7	*n/a*
# items	4430 (84)	812 (19%)	594 (14%)	99 (+29 IPs)
# ratings	17271 (18)	2427 (14%)	1386 (8%)	199 (+7 IPs)
# comments	791 (51)	459 (58%)	375 (47%)	46 (+47 IPs)

There are several social factors that may have a significant influence on the results. Wikilens.org was started by GroupLens members, so we call out in Table 2 the fraction of contributions by anyone associated with GroupLens, and by the authors of this paper. Table 2 shows that GroupLens had a considerable fraction of contributions but by no means a majority, except that the authors contributed many of the comments.

Furthermore, we occasionally recruited in MovieLens forums or in response to emails about MovieLens. Members complain when movies they wish to rate are not present in the system, and we would respond that they could try WikiLens.

We now discuss lessons we learned, organized by our principles: ADD, DEEP CHANGE, MICRO-CONTRIBUTE, FIND, SEE OTHERS.

ADD.

Lesson #1: Users will add items

About half of users (99 of 231, plus 29 IPs, Table 2) added at least one item. Some users added many, which we describe below.

Lesson #2: Import tools made it easier for users to add items, but were not needed to broaden the community of contributors.

Some of us were excited to build import tools, since it lowers the cost of getting items with complete details (e.g., a book with author, link to Amazon, and so on).

Table 3. Contribution in top five categories of wikilens.org (not counting the User category).

Category	# items	# distinct contrib-utors	# distinct contributors in top 25 most-rated	Had importer
Movie	1967	23 (3 IPs)	7	No
Book	730	50 (7 IPs)	16	Yes
Album	673	25 (3 IPs)	8	Yes
Restaurant	114	24 (2 IPs)	12	Yes
Web site	93	30 (3 IPs)	9	No

However, the largest category (Movie) had no import tool (Table 3). The next two (Book and Album) had import tools. Book had a larger number of distinct contributors than Movie, while Album (with fewer items) had a similar number of contributors to Movie. From this, we can't tell if the import tool broadened the community of contributors or not.

Lesson #3: Broadening the community of contributors is useful.

A broad community of contributors is desirable. For example, it may make a community more robust to members leaving.

It may also make the recommender more reflective of the community's interests. We had one user (call him MovieMaven) who added a considerable portion of all movies (1,357 of 1,967). However, MovieMaven did not add all of the popular movies. He added only three of the top 25 most-rated movies in WikiLens. Similarly in the Book category the top contributor (the first author) added only 3 of the top 25. Table 3 shows that in general there were many distinct contributors in the top 25 most-rated items of each category.

We were also surprised that a broader community may converge more rapidly than we expected to interests outside of WikiLens. Popular movies in MovieLens are often also in WikiLens: 95 of the MovieLens top 100; 399 of the top 500; 693 of the top 1,000. Since MovieLens has thousands of active users from all over the world, it is a reasonable proxy for movie popularity. Presumably, on average these popular movies are more well-known and interesting for browsers. MovieMaven did not add all of the movies by this external measure of popularity either: 36 of the MovieLens top 100; 239 of the top 500; 468 of the top 1,000.

Lesson #4: One empowered user can make a big difference.

This is perhaps no surprise to those familiar with volunteer efforts. MovieMaven says:

> *"I really love the opportunity to add whatever you'd like in the film category, which is the main reason I'll keep the site on my radar in the future. It makes the site unique among its kind, at least as far as I know."*

It turns out MovieMaven is not only a WikiLens user, but also a MovieLens user. MovieMaven added 1,357 movies in WikiLens, of which 1,320 (97%) were already in MovieLens! We confirmed via usage statistics and email that these movies were added manually without the use of scripts. Most of the times between MovieMaven's movie adds were 10-300 seconds. Conservatively estimating 30 seconds per add, this is at least 12 hours of work. MovieMaven says:

> *"I did it the old fashioned way, line by line, allowing myself to become a bit too obsessed by the whole thing! It didn't seem to take too long regardless, because I knew most of the info already.."*

Apparently the 3% of movies not in MovieLens was sufficiently motivating to seek another recommender in which to participate. In fact, it is a common complaint on MovieLens by heavy users that some of their favorite movies are not present and the maintainers of MovieLens do not add enough movies, which is part of what motivated this research.

It seems likely that the "Movie" category became the biggest because one user dedicated considerable energy to it, although we cannot entirely rule out the possibility that other users from MovieLens would have picked up the slack.

Lesson #5: Items in the "long tail" can attract attention

Our top "search keyphrase"[12] reported by our web log analyzer is "thottbot.com". Thottbot.com is a site with supporting details for World of Warcraft, a very popular video game, and also one of the "Web site" items on WikiLens. We also attracted the most comments (13, mostly anonymous) to the "Michelob Golden Draft Light" beer page discussing how it is only available in certain places, people

[12] a phrase used in a search engine to get to WikiLens

miss it, love it, hate it. Both of these items show up on the first page of Google search results at present.

Neither of these items is in the most popular category, or the most popular in their respective categories (Thottbot.com is #28 of 94, Michelob is #13 of 64). This is another example of the argument in [3] that there is real interest in the many little-known items in the "long tail".

DEEP CHANGE.

Lesson #6. Users understand and change categories and fields.

Table 4 shows that the top categories on WikiLens have fields. Those fields were mostly set up by the authors, although not in the case of the most popular category, Movie. We avoided the "Movie" category because it seemed redundant to duplicate the more popular MovieLens. However, a casual user of both systems created the Movie category and many of its fields, and then another user (not MovieMaven) added a couple of fields ("Genre" and "Starring") and field values (genre choices).

Table 4. Fields in top five categories.

Category	Movie	Book	Album	Restaurant	Web site
# visible fields	8	7	6	12	2
% field values filled in	65 %	84 %	80 %	71 %	48 %
% items with some fields filled in	99 %	97 %	97 %	94 %	49 %
% items imported	0 %	78 %	53 %	24 %	0 %
Fields created by non-authors	yes	no	no	no	no
Fields edited by non-authors	yes	no	no	yes	no

There are other examples of users using categories and fields. There is a "SuggestedCategory" category where people can rate ideas they'd like to see as categories. The first author made the "TV Show" category when a user strongly prompted him to promote it from SuggestedCategory. It then had some fields and possible options added by users. Now it has 16 distinct contributors of items, 57 items, and 350 ratings. Also, "New York Restaurant" shares many fields with "Restaurant" but with "cross street" added, which is important in Manhattan, where a primary user of that category lived.

Lesson #7. Users fill in field values.

Table 4 also shows that fields are often filled in, both by importing and entering manually. The "Web site" category has an unusually low number of field values filled in. In this case, fields were added after many of the items in the category, and migration of the field data from page text to fields has not yet been done.

Lesson #8. Users can define new applications with flexible tools.

In a private book club, we created "Proposed Book" and "Read Book" categories that share fields and importers, since an important task of the book club is to choose a book (from "Proposed Book"), then read it (dropping it into "Read Book"). Also, the group evolved to choose their next book on a "Next Book" page, where a few books are hand-selected by members for the next book chooser to examine. The page is free-form text with Ratemania widgets so ratings can be gathered and opinions seen by the chooser, and the "Proposed Book" category keeps data on which books have not been chosen but have high ratings.

Another example from wikilens.org: the "Recipe" category has the problem that many recipes are copyrighted, and should not be reproduced. A user who wanted this category solved the problem (perhaps unintentionally) by simply making one of the fields be "URL" and referring to recipes on other web sites. (To be fair, the "Recipe" category is still quite small.)

MICRO-CONTRIBUTE.

Lesson #9. WikiLens supports a range of contribution, and the easiest things are participated in widely.

Table 2 shows that 199 of 231 registered users made at least one rating, 99 added an item (+29 IPs), 46 made comments (+47 IPs), whereas 16 users (only 6 not associated with GroupLens) made changes to category field definitions.

Lesson #10: Small communities need spam protection.

As noted above, an easy and obvious "Add Comment" interface attracted outside contribution. However, wikilens.org also attracted unwanted contribution. In late 2005, spammers discovered the site, and started editing dozens of pages per day in clearly automated ways to put links to their own sites, presumably to increase their perceived importance to search engines. While Wikipedia successfully defends against vandals through a large and vigilant community, WikiLens is smaller and more vulnerable. Even requiring creation of a login did not help—the spammers automatically created users. (This was more impressive because we had customized the login screen to our own site.) We solved this problem by requiring answering a question about a picture to create a login (also called a CAPTCHA [2]), and rating at least one page before editing.

However, our solution may have been too drastic, because we also stopped anonymous comments. (Up to that point 51 comments had been added anonymously.) Perhaps we should follow the lead of popular blogs that pose a CAPTCHA to anonymous users after entering a comment. This entices the user to add a comment, then once already invested, prove that they are a human.

FIND.

Lesson #11. Category pages were hubs of browsing.

The top 10 pages browsed by logged-in users in our detailed usage logging were: Movie (6288 times), RecentChanges (4998), TitleSearch (4684), Book (3234), Album (3198), HomePage (2331), Artist (881), AccountCreated (784), Beer (733), Web site (648). Six of those top 10 were category pages. Note that this result is somewhat confounded by the fact that category pages encompass several non-browsing activities as well, such as adding or searching for an item in that category.

We further investigated browsing behavior with a survey we administered to all WikiLens users November 10-15, 2006 with the goal of measuring their perceptions of the system. The survey got 37 responses, of which 54% considered themselves current users of the site. We administered questions on a five point scale, "strongly agree" (1), "agree" (2), "neutral" (3), "disagree" (4), and "strongly disagree" (5) along with open comment boxes.

Users most strongly agreed that they use WikiLens to 'find new items to learn more about' (81% agreed or strongly agreed, average 2.11). Users most strongly agreed that they find items in WikiLens by 'a category page (e.g., "Movie" shows all movies, ordered by prediction)' (65% agreed or strongly agreed, average 2.1). Users most strongly agreed that they evaluate items based on 'prediction value on a category page' (67% agreed or strongly agreed, average 1.9).

Users mostly search with the search box prominently displayed on each page, which searches page titles. They performed 5,324 searches over 18 months: 88% page title searches, 5% parametric searches, 4% full page text searches, and 2% fuzzy title searches.

Lesson #12. Traditional collaborative filtering is possible even in small datasets.

While we did not implement traditional collaborative filtering in WikiLens, we decided it would be interesting to simulate it after the fact to see if it might be useful in the small world.

We compared the recommendation quality of two models: First, an average model that predicted the average item rating for each item. All ratings fed in and out of the model were user-average-adjusted ratings, i.e., (user rating - average user rating). Second, a standard item-based model using cosine similarity [16].

To reduce prediction noise, we pruned the dataset to items and users with at least five ratings, which reduced it to 156 users and 1,035 items. To evaluate recommendation accuracy, we trained a prediction model on 80% of the users, and withheld 20% for measurement. For each withheld user, we hid 20% of the user's ratings (selecting only relatively high ratings), fed the other 80% into the recommendation model, and generated a recommendation list without regard to category. We measured *recall*, the fraction of the hidden 20% of the ratings included in recommendation lists of size 1, 5, 10, and 100. We repeated the process 300 times.

Table 5 shows the results. Recall was higher for the cosine model, especially for short recommendation lists. A chi-squared test on the number of recalled hidden ratings versus missed hidden ratings show the differences between models to be significant (χ^2=211, 422, 134, 49; p <= 0.001 for all sizes).

Table 5. Recommendation simulation results.

Model	Recall1	Recall5	Recall10	Recall100
Average	0.00185	0.010	0.028	0.215
Cosine	0.00674	0.024	0.039	0.230

SEE OTHERS.

Lesson #13. Buddies were mostly used by pre-existing social groups.

On wikilens.org, 44 users had at least one buddy, 21 from GroupLens Research (call it the "lab"), and 22 from outside GroupLens, and one test user. Most users had no buddies. There were 184 lab-lab relations, 34 lab-nonlab relations, and 30 nonlab-nonlab relations. Thus, the average number of buddies for a lab member was 8.8, and for a non-lab member with at least one buddy it was 2.8. A user in the lab was much more likely to have many buddies, and the buddies were also likely to be in the lab. Similarly, in the two smaller private WikiLens instances people had many buddies in order to see each others' ratings.

Some of us hoped users on wikilens.org would bring new users into the system as buddies to see their recommendations or ratings, but this did not happen.

People did view others' ratings somewhat: 36 logged-in users (9 from GroupLens) viewed another user's ratings page 1,491 times (567 from GroupLens users). However, according to our survey, users tended to respond least positively to the social features of the site. For example: "I use WikiLens to influence others" (33% agree or strongly agree), "I use buddies' ratings to evaluate items" (36%), "I add things for particular people" (21%). This indicates that most users did not feel social ties, but used the system alone. User did seem to feel a connection to the community as a whole – they claimed to add and rate items for the good of the community (86%).

Lesson #14. If you can get buddies, seeing their ratings may be valuable.

Our two smaller, private WikiLens installations with pre-existing social groups often used others' ratings in "instant poll" Ratemania to help choose research papers or books to read, printer names, and more. Identified ratings helped

intuitively answer such questions as "is one item here a clear winner?" or "does this choice make someone very unhappy?"

6. Possible Improvements

Our experiences suggest many ideas for improvements.

Improving recommendations.

81% of users surveyed agreed or strongly agreed that they use WikiLens to 'find new items to learn more about'. WikiLens recommendations are averages, modified by buddies' opinions. These social recommendations proved valuable in small groups who knew each other before joining the system, but not in larger groups who did not know each other in advance. This may have occurred because we did not design features to help people get to know one another. Future work might benefit from explicit design to encourage the development of relationships. Because of the lack of social relationships, many users did not get personalized recommendations. Perhaps the social recommender should be augmented with content filtering techniques that work independent of the number of buddy relationships, or even mixing in some amount of traditional collaborative filtering algorithms.

Improving content organization and manipulation.

Many survey respondents agreed or strongly agreed that they use WikiLens to 'keep track of items (e.g., movies) I like or dislike' (64%, average 2.54), yet organizing the repository has several major challenges. First, some categories have natural hierarchical structure. At present wikilens.org has five restaurant categories, one for each locality of a user who wished to start one (Boston, New York, Bay Area, Minneapolis, and Chicago). One category per locality does not scale. How should these categories share with each other appropriately, while preserving local character? Second, supporting DEEP CHANGE while providing satisfactory performance is challenging. How should user changes to category structure propagate through the system? Should the changes immediately lead to reorganization, which would slow their initial execution, or should the changes be interpreted at search time, which would slow every search operation? Third, there are several interesting extensibility features that could be explored, including allowing users to build or modify item importers[13], or to plug in custom recommendation algorithms. Few users would take advantage of these features, but they might have far-reaching effects.

Improving usability, sociability, and incentives to contribute.

During development, we did informal usability testing. However, WikiLens is a real, complex system with multiple goals (contributing and finding various types of content), and could be made easier and more obvious to use. For example, some people make accounts and do not rate or edit anything. This is perhaps a sign that they do not know or care what to do next. One user said,

> *"I got frustrated with the interface. There were also too many things to rate. Rating other users seemed a bit too 'meta' too me."*

On the other hand, it may not appeal to everyone. Another user said, "*I tried it shortly for [curiosity] but did not yet have a concrete case where I would need to use it.*"

Furthermore, a stronger social community (e.g., more social interaction or even mentoring) supported by social features might foster more contribution. One user said,

> *"i had a lot of intention to contribute more, I just haven't had the motivation to :)"*

One might look at research on using social theory to motivate contribution [12].

Improving ease of installation

There are several WikiLens installations with varying degrees of activity and success. However, WikiLens is not trivial to install. One potentially successful application is very small communities where everyone knows each other and would like simple ways to share their opinions about items. Such communities are more likely if installation is easy.

Improving underlying technology for performance and ease of development.

We chose PhpWiki as a popular, extensible, mature platform upon which to develop. However, developing in PHP has challenges. It is hard to write code that is fast and space-efficient, especially across requests. This has been a constant issue even at the small scale of wikilens.org (thousands of items), and hampers experimentation. Possible strategies to address this include using PHP's shared-memory caches (rumored unreliable) or distributed caches like memcached[14] (which would increase the complexity of installation and operations). Also, PHP more easily allows coding errors like misspelled variable names or changed function interfaces. Several of us have been tempted to switch to or rewrite a system in a compiled language like Java for performance and type checking, while others of us are repulsed by this same suggestion. Since WikiLens to date has required many hours of development, these are important issues to grapple with.

7. Conclusion

We have argued that recommenders everywhere would be valuable, and that they are feasible. We have explored five key principles to guide their development. We described WikiLens, our open-source system with features that support those principles, and shared our experiences with it. We cannot draw incontrovertible conclusions based on only a few deployments, but we found support for many of the

[13] http://citeulike.org allows this

[14] http://www.danga.com/memcached

principles, especially those enabling user contribution and deep changes.

In our experience, many users will contribute a little, some a lot. A broad community adds many of the most popular items, and also gems of interest from the long tail. Given power, users understand, add, and change categories and item fields, and can solve problems in unexpected ways or even create new applications. Users use and like an interface to recommend interesting items, and are interested in their buddies' opinions.

However, we also learned of limitations to our approach. WikiLens users could use improved recommendations, usability, sociability, incentives to contribute, performance, and ease of installation, while its developers would appreciate an easier platform upon which to develop.

We encourage other researchers to join us in achieving the vision of community-maintained recommenders. To that end, WikiLens is available as open source software at http://www.wikilens.org/wikilens-src.tgz, and we make datasets of the non-private parts of the current repository available to other researchers upon request. Our hope is that recommenders may flourish everywhere to help us find our way through an overwhelming world of information.

8. Acknowledgements

Thanks to Joe Konstan for design discussions; Dan Cosley, Melissa Skeans, and Kurt Wilms for helpful conversations and support; and WikiLens users for passion. This work is funded by National Science Foundation grants IIS 03-24851 and IIS 05-34420.

9. References

[1] G. Adomavicius, A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Trans. on Knowledge and Data Engineering. Vol. 17, Issue 6. (2005) pp. 734–749.

[2] L. von Ahn, M. Blum, N. Hopper, J. Langford. CAPTCHA: Using hard AI problems for security. Eurocrypt, 2003.

[3] C. Anderson. The Long Tail. Wired 12.10, Oct. 2004. http://www.wired.com/wired/archive/12.10/tail.html.

[4] S. Bryant, A. Forte, A. Bruckman. Becoming Wikipedian: transformation of participation in a collaborative online encyclopedia. GROUP 2005, pp. 1-10.

[5] D. Cosley, S. K. Lam, I. Albert, J. Konstan, J. Riedl. Is Seeing Believing? How Recommender Systems Influence Users' Opinions. CHI 2003, pp. 585-592.

[6] D. Cosley, D. Frankowski, S. Kiesler, L. Terveen, J. Riedl. How Oversight Improves Member-Maintained Communities. CHI 2005, Portland, OR, 2005, pp. 11-20.

[7] D. Cosley, D. Frankowski, L. Terveen, J. Riedl. SuggestBot: Using Intelligent Task Routing to Help People Find Work in Wikipedia. Intelligent User Interfaces (IUI) 2007.

[8] S. Drenner, M. Harper, D. Frankowski, J. Riedl, L. Terveen. Insert Movie Reference Here: A System to Bridge Conversation and Item-Oriented Web Sites. CHI, 2006.

[9] T. Erickson, W. Kellogg. Social translucence: an approach to designing systems that support social processes. ACM Transactions on Computer-Human Interaction (TOCHI). Vol. 7, Issue 1 (March 2000), pp. 59-83.

[10] J. Goecks, D. Cosley. NuggetMine: Intelligent Groupware for Opportunistically Sharing Information Nuggets. In Proceedings of the 2002 Int'l. Conf. on Intelligent User Interfaces (IUI 2002), pp. 106-113.

[11] M. Guzdial, J. Rick, B. Kerimbaev. Recognizing and Supporting Roles in CSCW. CSCW 2000.

[12] K. Ling, G. Beenen, P. Ludford, X. Wang, K. Chang, X. Li, et al. Using social psychology to motivate contributions to online communities. Journal of Computer Mediated Communication, 2005.

[13] T. Malone, K. Grant, F. Turbak, S. Brobst, M. Cohen. Intelligent Information-Sharing Systems. Communications of the ACM, Vol. 30, No. 5, May 1987, 390-402.

[14] P. Oliver, G. Marwell, R. Teixeira. A Theory of the Critical Mass. I. Interdependence, Group Heterogeneity, and the Production of Collective Action. The American Journal of Sociology, Vol. 91, No. 3. (1985).

[15] N. Peddibhotla, M. Subramani, M. Contributing to Public Document Repositories: A Critical Mass Theory Perspective. Working Paper 2006, Carlson School of Management, University of Minnesota. http://misrc.umn.edu/workshops/2006/spring/naren.pdf.

[16] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. World Wide Web Conference (WWW10), May 2001.

[17] J. Schafer, J. Konstan, J. Riedl. Electronic Commerce Recommender Applications. Journal of Data Mining and Knowledge Discovery. (2000). vol. 5 nos. 1/2, pp. 115-152.

[18] J. Surowiecki. The Wisdom of Crowds. Doubleday, 2004.

[19] B. Thorn, T. Connolly. Discretionary Data Bases: A Theory and Some Experimental Findings. Communication Research, Vol. 14, No. 5. (1 October 1987), pp. 512-528.

[20] F. B. Viégas, M. Wattenberg, K. Dave. Studying cooperation and conflict between authors with history flow visualizations. SIGCHI, pages 575–582, Vienna, Austria, 2004. ACM Press.

[21] M. Völkel, M. Krötzsch, D. Vrandecic, H. Haller, R. Studer. Semantic Wikipedia. World Wide Web Conference, 2006.

[22] Wiki Wiki Web: http://c2.com/cgi/wiki?WikiWikiWeb.