Creating Metadata in a Social Portal

Project Report
Student Vacation Project
Information Ecology Research Group
Distributed Systems Technology Centre Pty Ltd

Student:
Trond Abelseth
tabelseth@dstc.monash.edu.au

Supervisors:
Dr Nigel Ward – nigel@dstc.edu.au
Gregor McEwan – mcewan@dstc.edu.au
ABSTRACT
A faction of the Information Ecology project at DSTC is currently constructing a
social portal called Scuttlebutt. This social portal, a personal space of information
organised into channels, allows for sharing and recommending of information,
predominantly web pages to other users in the social network.

This report addresses the issues involved with extending the metadata in Scuttlebutt;
the opportunities and the limitations, user interface and interaction issues as well as
presentation mediums. A developed prototype is presented to show the feasibility of
certain kinds of metadata in a social portal.

INTRODUCTION

Background
The Information Ecology project at DSTC is currently constructing a social portal
called Scuttlebutt. The notion of a social portal refers to a personal space of
information sharing where information is presented through channels. This sharing of
information occurs in form of recommendations, predominantly web references
(Mansfield, Ward et. al. 2001).

The idea behind the social portal stems from the interest to investigate portal-style
presentation of information from social networks. This takes the portal notion a bit
further than broadcast-style portal such as MyYahoo! or MyNetscape, which merely
puts all information of interest to a user in one place, targeted by demographics.
Rather than relying only on large demographics to target users, the social portal is
utilising the social context to recommend information.

Motivation
Scuttlebutt is a web-based social portal where users recommend web pages to others
in the network of users who subscribes to the channel where the web page was
recommended. Users also create and customise channels they get recommended from
other users in the social network. If a receiver of a recommended channel wishes to
subscribe to it, the user will receive all recommendations to that channel. The
channels can be placed anywhere in the portal page depending on the user’s
preferences.

User satisfaction is a critical success factor for any interactive software application,
particularly when the usage of it is voluntary (i.e. the user decides if he/she wants to
use it, and will only use it if it proves helpful. As opposed to an airline booking
system which an employee is forced to use). Further, in a social portal the
effectiveness of an individual recommendation can be jeopardised by the lack of a
good description and sufficiently detailed information about the recommendation.
This research project seeks to address issues around the problem of providing fully,
and properly, described recommendations. It will present possible solutions with
respect to what descriptive data (metadata) is useful and how it is gathered as well as
how the metadata interacts with both the sender and receiver.
A constructive and useful interaction between the metadata-creating software and the user of the portal is imperative to the success of the metadata. There are two aspects to this issue; the interaction of the user receiving the recommendation, and the interaction the metadata-creating software has with the sender. The sender of the recommendation necessarily wants to submit the richest, yet most precisely described recommendation in order for the receiver to instantly identify the topic and mentally place it somewhere along the ‘boring-to-interesting’ spectrum. In other words, put it into context.

Definitions
This project seeks to identify how ‘metadata’ can enhance the usefulness of recommendations in a social portal. Metadata is in its broadest meaning defined as ‘data about data’. In the context of this project we use this definition quite liberally and usually refer to any data or information which serve to enrich or further describe a particular object.

**Research Questions**
The research questions outline what the research effort aims to achieve. These questions have been formulated as follows:

1. How does metadata-creating software interact with the user?

Metadata-creating software in this context is an autonomous or semi-autonomous application which attempts to get as much ‘knowledge’ as possible about a recommendation (web link) from a wide range of sources (the web link itself, previous interactions, Internet searches etc.). And in turn present this knowledge or information to the user. This question pertains to potential user experiences with this software component. What options are there? What kinds of user actions are desirable/optimal? And is the sender ultimately responsible for a recommendations’ content and the metadata it might present to the receiver?

2. What metadata or ‘descriptive information’ is useful, and how is this metadata created/extracted?

The second question deals with the number of ways the extra descriptive information (metadata) can be extracted and created from information sources like the document itself, its location, its classification in independent sources or even its recommender etc. Further, can data from these information sources be automatically created or is human intervention necessary?
**OVERVIEW OF SCUTTLEBUTT**

The Social Portal

Scuttlebutt is a web-based social portal, constructed to study how groups work and to get a better understanding of social networks. The social portal aims to show traditional portal information with social information. Traditional portal information meaning newswire services, share market information, weather updates etc. Whereas social information pertains to information from colleagues and friends, that is targeted content from people who know you and your interests (Mansfield, Ward et. al, 2001). The social portal consequently aims to contain all information of interest, customised to the users interest and preference both in terms of content and layout.

The current implementation of a social portal, Scuttlebutt, focuses on the social aspects of this portal experience, and lets the users recommend web pages to groups or friends. Every user receives all the information in one place, the web portal, organised into channels on a particular topic. The organisation or layout of the channels is customisable by the user. Additionally, any user can post comments about a recommendation and thus trigger a discussion. At any time any user can check the discussion thread.

What Scuttlebutt Does

The Scuttlebutt web portal is organised into channels which a user subscribes to. A channel typically covers a theme or topic of interest to a user, and is thus created either by yourself or by other users asking you to subscribe. Each channel contains a number of recommendations which have been posted by the owner of that channel. Whenever a user is making a recommendation in a channel, the subscribers gets notified by receiving an email containing the recommendation accompanied with a brief description (if the sender has included one). The recommendation is automatically added to the portal channel.

There are two ways a user can make a recommendation in the Scuttlebutt universe; by making a recommendation while in the portal, or using the ScuttleButton (explained in the next section). Making a recommendation requires the sender to fill in fields in a form; the title, the URL and a description. Doing this from the portal page currently provides limited aid in terms of entering the fields automatically. Automatic entry would make the task of recommending faster. Manual entry can be a tedious task, even if it's only a copy-and-paste affair, and potentially deters users from actually making the recommendation. The ScuttleButton feature aids in this process.

The ScuttleButton Feature

The ScuttleButton is an extra feature in the system which functions as a link in the browser to facilitate easy access to the recommendation system. It means the user does not necessarily need to have their Scuttlebutt portal open to recommend a web page. The ScuttleButton is a link button added to the browser toolbar (in Netscape or Internet Explorer). When the user presses the button a Scuttlebutt pop-up is displayed, ready to recommend the current page the user is visiting. The feature automatically inserts the URL and extracts the title from the web page to be included in the
recommendation. The task of making recommendations can then be completed with a couple of clicks.

Current Metadata in Scuttlebutt

This study is concerned with metadata for the recommendations in Scuttlebutt, and metadata from the viewpoint of a recommendation. Currently, a recommendation object has, or can have, the following metadata:

- URL: the unique identifier for a web reference
- Recommender: the user who recommends the object (referred to as the sender)
- Title: title of the web-page (or a title the sender chooses)
- Date/Time: when the recommendation took place
- Description: a longer description of the recommendation

This metadata is voluntary and can be omitted by the sender, except Recommender and Date/Time which are generated and logged automatically by the system. The Title field is also mandatory.

Comments about a recommendation can be posted by users. The ‘Comment’ element in the recommendation system can be regarded as metadata to the recommendation object. This kind of metadata is based on the opinion of a receiver of that particular channel and is helpful in creating the context for the page. A comment is naturally added after the recommendation has been posted.

Problem Statement

In the recommendation process it is difficult to attach good metadata. Good, or quality, metadata is descriptive information that can be regarded as useful for the receiver. And it is hard to know of what type this metadata is, and from where it can be extracted. Even with the aid of the ScuttleButton pop-up feature, the recommendation’s metadata is still limited and it is up to the recommender to make the effort of inserting extra descriptions and information about the recommendation.

Having more metadata included as a part of the recommendation can be very helpful for the receiver, and subsequent sections will discuss this and related aspects in depth. Questions which will be debated are how much metadata to include? What is too much, and how do we find the optimal amount of metadata? How are user experiences affected, for both sender and receiver, and what are possible solutions?

RESEARCH APPROACH

The project took form as a proof-of-concept investigation. The first period set the scene and involved formulating research questions, exploring and reading relevant literature and collaborative exchange of ideas, a brainstorm, among group members, to identify alternatives and directions for the research project.

A list of ideas were compiled and a project plan was written, elaborating on all aspects of each idea, including sender and receiver interaction, and technical
opportunities and limitations. One proposed solution was implemented and integrated into an existing version of Scuttlebutt. Subsequently, lightweight user tests were carried out in order to evaluate the usefulness of the concept.

**OPTIONS FOR METADATA EXTENSIONS IN SCUTTLEBUTT**

Metadata Ideas in Scuttlebutt

The purpose of metadata is to provide supplementary information about a data object. The value of metadata in a computerised system depends on the data it’s describing and how important the metadata is in the functioning of the system. Scuttlebutt, as a web page recommendation system can ‘get away with’ only core metadata as described in previous sections. Other metadata is not critical to Scuttlebutt’s basic functions. However, the value of extending the metadata types and means of obtaining them can prove useful for the functionality and usability of the system. Receivers of recommendations can with metadata more quickly put the page into context, and instantly know what the topic is. Automatic extraction of metadata lessens the burden on the sender to manually create the metadata.

Core to this discussion is the issue of what makes up a recommendation. And how important is extra information about it. The task of recommending something, in our case web links to an interest group, involves social context. Scuttlebutt is a recommendation system in a social context; recommendations to a group of interested subscribers. The research group believes it is important to include both the social context and the document context, to create the overall dimension in which the recommendation was made. The prototype created in this project, documented in a later chapter, tries to create a context for the recommendation or web page, by identifying in which category the page is referenced in a search engine directory, and highlight this to the receiver.

Also central to this discussion is the issue of metadata quality. The value of extended metadata in Scuttlebutt recommendations can be great if the quality of that data is high. Quality metadata is thought of as being relevant to the recommendation and hence potentially interesting to the user. But how do we know the metadata is relevant or interesting to the receiver? This will be discussed in a subsequent section where the issues of extraction and user intervention are dealt with. There are many sources and ways of creating metadata and the next section will go through the types of metadata, as well as their source, followed by a discussion on how and when this metadata could be extracted and created.

**Metadata Types and Sources**

Outlined below is a list of potential metadata types and the source of that metadata which can be of use in order to create a context around the recommendation. These specific types and sources touches on the document itself, its context as well as its usage after the recommendation has been posted.

From the document:
- Manual entry of keywords by sender
- Automatic keyword extraction from the text
- Automatic meta-tag extraction (keywords or description fields)
  Evaluate them and include most appropriate – if they exist at all in the web page

References to the document in search engines (contextual):
- Location of a recommendation in a directory listing like Google, Yahoo! or Dmoz.org (to put the recommendation into a broader context)
  - Parse searches based on title, base URL, meta-tags etc.
- Identify newsgroups relating to the recommendation (automated search of related Usenet groups) to see where this topic has been discussed.

Within the Scuttlebutt universe and usage related metadata:
- List channels, recommendations, comments which contain the same words as the keywords in the new recommendation
- Number of people subscribing to the channel where this recommendation is posted to
- What channels this user has recommended to you before (similar to a history function)
- Introduce a ‘title’ for the comments posted and have that listed with the recommendation link (in the web portal)

This summary lists different types of metadata, from the document, its context, and the usage of it in Scuttlebutt. Metadata can be more interesting when it can further illustrate or create context around a recommendation and potentially also ignite discussion about the topic. In a social portal this kind of metadata would be particularly beneficial.

When and How to Add Metadata?
This discussion is divided into two areas: pre-recommendation and post-recommendation generation of metadata. With pre-recommendation we mean metadata which is generated before (or at the same time) the recommendation takes place. In practice this implies that the sender at his or her discretion has the chance to incorporate metadata about the recommendation before it is submitted. Post-recommendation generation represents the metadata that can be created after the recommendation has taken place (and the receivers have actually received the recommendation.)

The two distinct approaches, pre-recommendation and post-recommendation, pose different challenges to both the metadata itself (what metadata it can be – research question 1) and how users (senders and receivers) interact with interface that provides it (research question 2). The following two sections address the research questions with respect to each approach.
Pre-Recommendation Processing

Pre-recommendation processing of metadata implies adding it to the recommendation before submitting it to the receivers. In practical terms this means the metadata adding occurs as the sender is actually making the recommendation. With this approach the sender includes what is appropriate, and what he or she identifies as beneficial and relevant in order to get the message across. The quality aspect of the metadata is also potentially greater this way, a particularly important issue since the metadata is a part of how the senders represent themselves through the channel.

Quality issues with regard to metadata are of great concern in any system. Pre-recommendation generation is arguably a more ‘safe’ option to ensure quality since the sender can exclude and include metadata at the time of making a recommendation, and therefore have more control over it. Further, with this approach the sender can make sure the intentions behind the recommendation is clear.

The major drawback from this procedure however is that all metadata needs to be processed and presented as the sender makes the recommendation. This process is constrained both by time, type of metadata available and the interface presented to the user (the interface issue is discussed later).

Post-Recommendation Processing

Metadata which is added to the recommendation after it has been posted is regarded as post-recommendation processing of metadata. The post-recommendation approach involves automatically attaching metadata to the recommendation. It does not seem feasible to prompt users to verify a metadata entry after the recommendation has been posted, hence the post-recommendation process needs to be automatic. The automatic entry alternative can be considered as ‘unsafe’ because there is a chance unintended metadata is added, misrepresenting what the sender wanted to recommend.

Any metadata extracted at the pre-recommendation stage can also easily be accessed at a later stage by the metadata-creating software. In post-recommendation mode there are also a host of other metadata options available, including usage data; how many people accessed the related channels and documents information etc. This metadata requires a certain amount of usage before it is available. Hence, technically speaking, the access to metadata after the recommendation has taken place is potentially greater than before. But still there is a problem with quality and verification of the metadata. Further, the extra metadata available after the recommendation has taken place, might not be as relevant for the receiver.

Metadata in an Inconsistent Web

Extracting metadata on the World Wide Web, dealing with a vast array of public sites, is a process of handling data in an inherently inconsistent environment. Analysing web pages to look for valuable and defining content, to be used as metadata, is proving very difficult since every web page has different ways of presenting and organising their data. Metadata standards exist, one of them being the Dublin Core standard (dublincore.org, 2002), but the uptake ratio of these standards seems to be very low, based on the very informal surveys we’ve made. Making software which attempts to work similarly on every page being recommended is therefore a
challenging task. There are too many options on what metadata can be included, and because of the great discrepancies between web pages, it is difficult to single out quality metadata. The chance of including misleading or undesirable metadata is therefore high.

User Interaction
Any user in Scuttlebutt can be both a sender and a receiver. For the purpose of the discussion the two roles are divided in order to focus on the user experiences attached to each function.

The Sender
The sender has at present two interaction options when posting a recommendation; from the Scuttlebutt channel page or via the ScuttleButton. These options have been described previously. The question here is how much and what should be presented to the user while making a recommendation. There are many options on what can be included, most of which was mentioned in the section on potential metadata. The options, dealing with document-related metadata like keywords, descriptions etcetera are difficult to single out because of the great discrepancies between web pages. And there is therefore a great chance of including misleading or undesirable metadata. So, in terms of fulfilling the goal of efficient interaction design, we need to identify what metadata could be useful and what could lead to confusion. Further, there is the issue of presenting too much metadata. We want to avoid a situation where the sender needs to spend considerable time choosing and editing metadata while making a recommendation.

First of all, we want to support the sender filling out the fields, and thereby presenting options, to make the recommendation task as simple and quick as possible. One of the implemented solutions addresses this and provides a button that automatically fills the fields, and the sender can choose to keep the suggested values or edit them. This prototype feature, with test results and evaluation, is discussed in a later section.

The other approach to add metadata, post-recommendation, can have quite undesirable effects on usability issues in terms of the sender’s role. The quality aspect has been discussed and found to be difficult, if not impossible, to ensure with this approach, with the result of possible misrepresentations of the sender’s intentions. And equally important, from a usability perspective, is the fact that the sender with this approach does not know what metadata has been added, unless prompted about it. Hence post-recommendation seem to be an infeasible option due to both quality and usability, unless it is usage related metadata which is continually updated.

There is also the matter of tailorability and changes to the metadata after it has been created. The sender could be able to edit the metadata attached to recommendations. However, this could also lead to confusion on the receiver side

The Receiver
The receiver gets notified of a recommendation through chosen mediums, which currently is through email, the web-portal or Tickertape, a synchronous chat tool. The
Idiosyncrasies of these mediums will be discussed in the next section. This section will deal with how a receiver views the recommendation, which is primarily in the web-portal. In the portal the recommendation comes with attached metadata, including additions made by the sender in the pre-recommendation stage.

The receiver faces similar interface issues to the sender, but with a somewhat different perspective. Both see the same recommendation and the same metadata, but the receiver does not have to choose the optional metadata, it is merely displayed. Hence, the most important matter seems to be the amount of metadata presented and the way it is displayed. Too much could be cluttering and confusing, as would be the problem if it is poorly displayed. The reason for including a particular metadata option is also important, as this is always not obvious to the receiver, even if it is clear from the sender’s perspective. If it is unclear why the metadata is there it will have little significance to how the receiver sees the recommendation.

There is also the issue of tailorability with regard to the metadata. Should metadata be editable after being made? Supposedly a receiver cannot remove current metadata, but rather extend it and add more information.

**Presentation Mediums**

Presentations mediums are interfaces for both viewing and notification of recommendations. As previously mentioned, there are currently three ways to get notified in Scuttlebutt; web-portal, email and Tickertape, the synchronous chat tool. Which media the recommendation is presented through, the receiver interface, determines how the receiver experiences the recommendation in terms of richness of that information. Different mediums bring different user interaction issues, for instance in the case of email and a web-portal. Potential extensions are mobile devices such as mobile phones (SMS) and PDA’s. The different aspects of these devices and their typically limited screen space are beyond the scope of this report. This section will go through each medium and comment on their particularities in terms of user interaction.

The web-portal serve as the hub for all recommendations, organised into channels, and where the latest recommendation appears on top in its channel column. In terms of viewing recommendations the portal is ideal, as it allows for rich and tailorable descriptions. But the user will have to be logged in to get notifications of new recommendations, and this is probably not very practical for most users.

Email is the primary medium for notifying receivers, and can potentially contain most types of metadata attached to a recommendation. But the email is most likely sent out only once, at the time the recommendation takes place, and it would be undesirable to add or change metadata through emails. The web-portal is the place for this. Currently emails present the recommendation in its simplest form, the URL and a description, without any extra metadata attached.

The Tickertape communication tool, as it is designed, accommodates for recommending an URL together with a description. This description can be a further explanation of what the page contains, or simply be who it is targeted towards. The
communication tool itself obviously keeps track of sender information, time, and which chat channel it is sent to (chat channels do not correspond to Scuttlebutt recommendation channels). However, there is little room for extra information such as lists of keywords, related pages et cetera. This information could alternatively be posted in the corresponding web portal, if recommendations made in the chat tool are to be logged there. The question that then arises is whether such logs are feasible. If you just want to remind someone about a recommendation by sending it again in chat, is it then logged for a second time? The current Scuttlebutt system keeps these functions separate, so that recommendation sent only from the portal are being logged. The chat tool only serves as a display and notification of recommendations made through the web-portal.

**Prototype Development – Directory Listing**

**Justification**

As a part of this proof-of-concept project one of the metadata ideas was to be implemented for us to see how the functionality would work in practice. This process also spurred new ideas as a result of the development work as well as the testing and evaluation.

This prototype shows how we can put a recommended web page into context by identifying the categories where it is referenced in a directory listing (search engines). For instance, if a page from a conference is recommended, the categories from the directory can be:

- Computers: Computer Science: Conferences: 2001
- Computers: Computer Science: Distributed Computing: Conferences

*(example based on actual tests performed)*

This way some context is created in terms of location in a directory listing and related information about the topic can be found. The search engine used is Dmoz.org, the Open Source Directory for the web, feeding off a large community of human editors of the directory. Many of the more famous search engines such as Google, Lycos, Netscape Search and AllTheWeb, are all powered by the Open Directory.

Choosing this option as one to investigate further enabled us to study what metadata can be useful and also realise how difficult it can be to get quality data out of a web of inconsistency. As discussed previously a web page you recommend might not even have a title, and is unlikely to have metadata in HTML meta-tags.

**Functionality**

This prototype utilises various information in the recommended web document to make searches on the Open Directory, including title, a mixture of different components of the URL, keywords plus others. The result of each search is a list of all categories, or more precisely category hierarchies, where this web page is referenced.
The sender commits to most of the interaction relating to the assessment and adding of this metadata while the recommendation is made. As previously mentioned, there are two ways to make recommendations, using the ScuttleButton pop-up (figure 1) or posting directly from the channel in the portal page (figure 2).
The ScuttleButton Interface

Figure 1
“Make Recommendation” Interface

These two options needed somewhat different strategies in terms of functionality, although the layout and display is quite similar. Where the ScuttleButton automatically includes a list of suggested categories from dmoz.org where the page is referenced, the sender needs to press a button in the “Make Recommendation” interface after entering the URL to get the categories displayed. But as mentioned, the two approaches do have similar interfaces. Each line with a category has a checkbox for the sender to choose whether the category is deemed relevant or not.

The receiver will see the attached categories as comments immediately below the recommendation in the portal page (figure 3). The actual categories can be viewed by clicking on the “Read Comments” link.
Viewing Attached Categories

Figure 3
Integration with Existing Scuttlebutt Implementation

The prototype was integrated into an existing version of Scuttlebutt to construct a more realistic user environment, and to give other members of the group the opportunity to try the new functionality. It also provided the opportunity to make some informal tests on the prototype’s performance.

The integration process raised relatively few obstacles and only minor changes were made to the existing Scuttlebutt code to accommodate the new classes. Further, there were no changes made to the database structure of Scuttlebutt. Dmoz categories were treated as Comments, a special form of metadata, and easily displayed attached to a recommendation. This was done due to both time constraints and the wish to minimise changes to the current implementation of Scuttlebutt

Components

The final version of the prototype which was integrated with an existing version of Scuttlebutt contained four classes.

DmozCategory
- Opens an URL and extracts the values of given html tags and searches dmoz.org search engine and then places the results, ie. the categories, found in a dictionary
- Uses LinkParser, MetaParser and TitleParser

LinkParser
- HTML parser that detects link tags <a href> and places them in a list
- Subclass of HTMLParser

MetaParser
- HTML parser that detects META tags and places them in a dictionary
- Subclass of HTMLParser

TitleParser
- HTML parser that detects the Title tag and returns a string
- Subclass of HTMLParser

Main data structure used by DmozCategory class:
- Each value element in the key:value pair in the dictionary is a list. A list of tuples. eg. {key:[(element, value)]}

Testing

Informal and lightweight user testing was performed on current Scuttlebutt users. These users were from within the wider Scuttlebutt development team group, in other
words they are all expert users. Most were still external to this particular metadata project. Five people participated in the procedure.

The testing was done through informal telephone interviews where the user was performing set tasks and answering accompanied questions (Details about the procedure and questions found in Appendix 2).

Findings

This section will be divided into the topics that were identified as recurring themes from the user test results. These topics include user interface and design concerns, usefulness of the prototype and first reactions to categories as metadata. The roles of both senders and receivers are highlighted.

The users initially took the role as senders of recommendations, and there was a mixed reaction to what these dmoz categories were and why they were there. The interface did not properly explain this obvious to the sender. This was particularly potent when the suggested categories displayed were unrelated and irrelevant. Once the first barrier was broken, users started taking to the idea.

In terms of design, most users found the ScuttleButton to be better designed and “easier to understand” than the other option. This, it was alleged, was due to the button “Find More Info” in the “Make Recommendation” option. Little information was given through the interface to explain what this action did.

As receivers, all users appreciated the categories presented to them in the three recommendations which were evaluated. They were described as helpful, and that the additional related information could spur further interest and exploration in the topic. It was also indicated that the value of this kind of metadata (related categories) is possibly more constructive in a “topical” channel. In other words a channel which deals with specific topics, for instance “Extreme Programming” or “Research in Software Development”, as opposed to channels like “Funny News Stories” or “Interesting Inventions” which probably contains discrete and mostly unrelated recommendations. Such channels based on specific topics can make better use of metadata like categories since it can point to new, but related directions.

The user interface, from a receiver’s perspective, was constrained by the fact that the prototype used the existing Comments functionality in Scuttlebutt, a design decision made due to time limitation. All test subjects remarked on the pitfalls of this, the unintuitive design and the possible confusion this leads to. Also noted was the fact that it took too much screen space, with irrelevant information such as date and time, as well as sender was included (sender is logged as the system user “Scuttlebutt”).

All users responded positively to that this kind of metadata could add value to a recommendation. Criticism was made about unclear layout and some irrelevant category suggestions, but most believed it could add value.
Implications
There seems to lie a real challenge in the interface to fully and clearly explain the presence of related categories as metadata. And this challenge appears to comprise the realms of both senders and receivers. Further, the obstacle is at its greatest with new users of the system. However, if this can be resolved successfully, most users seem to appreciate the benefits of search engine categories as metadata.

CONCLUSION

Summary of Report
There are a host of opportunities for metadata in a web-based recommendation system. The limitation is in the quality and the relevance of the metadata to the receiver. Because of these limitations, automatic creation of metadata can have undesirable outcomes, as these can result in misleading information. Therefore it is important to let the sender take control over what information is attached to the recommendation.

The prototype development highlighted some of the difficulties, but also opportunities, of extending recommendation metadata to include references to other information sources. User acceptance is a concern, particularly with new users, which again is a challenge within the realms of user interface design. The users need to know why categories can be metadata to a recommendation. The result of user tests does however show that the approach could be useful.

Suggestions for Future Work
This discussion has only involved recommendations of websites, the main current functionality of Scuttlebutt, and neglected other items that can be recommended in a social network. The metadata options and opportunities, dealing with arbitrary objects, is a considerable challenge not investigated in this project. Future work could be extended to include these other kinds of recommendations.

In the context of this prototype and the present version of Scuttlebutt there are many options to explore and areas to assess. All the ideas mentioned in this paper have yet to be tested practically. In terms of extending the directory listing prototype, there are potentially improved ways of parsing searches, as well as options for ranking the suggested categories on relevance to the channel.
REFERENCES

World Wide Web References:

Dublin Core Metadata Standard
http://www.dublincore.org

DSTC Information Ecology Group – Social Information

Conference Papers and Technical Documents

APPENDIX 1 – PROJECT TIMELINE AND MILESTONES

The Fact Enrichment Agent project will run for 8 weeks, culminating in a seminar to present the findings. This section will firstly outline an activity plan for each week, and secondly identify milestones along the eight weeks.

Activities:
Broad outline of activities broken down to weeks:

Week 1: Familiarising myself with Extreme Programming, the Pliant approach and learn Python (Developing a small prototype in the process).

Week 2: Learn the PostgreSQL DBMS and continue to learn Python, with particular reference to the interfacing between Python and PostgreSQL (extend prototype).

Week 3: Complete prototype, discuss further steps in the project. Investigate possible ways to solve problem, identify which will be explored in detail (ie. Implemented) Complete initial list of possibilities of metadata and interaction. Decide implementation for the project and start programming.

Week 4: Continue implementing solution. Outline user interaction issues specific to the solution. Start to plan the report.

Week 5: Complete solution. Review. Further extensions on functionality?

Week 6: Complete extensions. Start integrating with Scuttlebutt.

Week 7: Complete integration with Scuttlebutt and plan user testing. Write report.

Week 8: Complete user testing and finish off report. Prepare presentation. Present report and findings in a seminar. Post-mortem.

Timeline and Milestones:
The outlined milestones represent major steps in the course of the project.

[---------------------------] [---------------------------] [---------------------------] [---------------------------] [---------------------------] [---------------------------] [---------------------------] [---------------------------]
       2    4    6    8

[------------|------------|------------|------------|------------|------------|------------|------------]  
complete prototype]  
project plan]  
decide implementation]  
complete impl prototype]  
review]  
integration with SB]  
testing and evaluation]  
report]  
seminar]
APPENDIX 2 – USER TESTING QUESTIONNAIRE

Medium: Telephone Interview

What: Informal user testing of a new semi-automatic metadata creator for Scuttlebutt, a social portal. Test subjects taken from within the wider development group.

Why: To identify the usefulness of the feature and potential improvements to its implementation.

Protocol: Test subjects (from here on called users) have agreed to use the prototype system while taking part in a telephone interview. Users will perform tasks in a sequential order and answer questions relevant to the task.

Below is a brief description on how the interview will proceed.

Preparation:
- Log out from your existing Scuttlebutt account.
- The user will need to log on to the prototype version of Scuttlebutt (located at: http://internal.dstc.monash.edu.au/cgi-bin/trond/SB1/cgi/fundament.cgi/) Please make sure you have a valid user account for this version before the interview.
- Get the ScuttleButton for this SB version (to be used temporarily for the duration of the user testing)

The interview:
- Then the user will need to send recommendations. The recommendations can be from any site you find interesting (from you bookmarks or an article in an online magazine etc.). Use both “Make Recommendation” and “ScuttleButton” to make recommendations, one from each interface. Question 1 and 2 will be asked.
- Answer questions relating to design issues in both solutions (Question 3 and 4).
- Go to the channel page and take on the role as the receiver. Go to the channel called “User Testing” and click on the three recommendations in that channel. Then answer questions on how helpful the dmoz metadata is, as well as on design issues (ques.5, 6)
- Wrap-up questions (ques.7, 8)
1. Sending a recommendation, how did you react to the extra information (dmoz.org categories)? Was it confusing or helpful to you? Why?

2. When making a recommendation you want the receiver to understand what the recommendation is all about. Do you feel dmoz categories add value to the recommendation you just made? If yes, how? If no, why not?

3. Using the “Make Recommendation”, did you notice the ”Find More Info” button? How did you react to the presented information? What did you think of the design?

4. Using the “ScuttleButton”, what do you think of the page design?

5. As a receiver of the recommendations, did you find them confusing or helpful? Why? What did you think of the design?

6. How would you improve the dmoz category listing feature?

7. Other comments?