Interest Based Selection of User Generated Content for Rich Communication Services

Matthias Strobbe^a, Olivier Van Laere^a, Samuel Dauwe^a, Bart Dhoedt^a,
Filip De Turck^a, Piet Demeester^a, Christof van Nimwegen^b, Jeroen
Vanattenhoven^b

^a*Ghent University - IBBT, Department of Information Technology, Gaston
Crommenlaan 8, bus 201, 9050 Ghent, Belgium*

^b*Centre for User Experience Research - IBBT / Katholieke Universiteit Leuven,
Parkstraat 45, bus 3605, 3000 Leuven, Belgium*

Abstract

The last few years, we have witnessed an exponential growth in available content, much of which is user generated (e.g. pictures, videos, blogs, reviews, etc.). The downside of this overwhelming amount of content is that it becomes increasingly difficult for users to identify the content they really need, resulting into considerable research efforts concerning personalised search and content retrieval.

On the other hand, this enormous amount of content raises new possibilities: existing services can be enriched using this content, provided that the content items used match the user's personal interests. Ideally, these interests should be obtained in an automatic, transparent way for an optimal user experience.

In this paper two models representing user profiles are presented, both based on keywords and with the goal to enrich real-time communication services. The first model consists of a light-weight keyword tree which is very fast, while the second approach is based on a keyword ontology containing extra temporal relationships to capture more details of the user's behavior, however exhibiting lower performance. The profile models are supplemented with a set of algorithms, allowing to learn user interests and retrieving content from personal content repositories.

In order to evaluate the performance, an enhanced instant messaging communication service was designed. Through simulations the two models are assessed in terms of real-time behavior and extensibility. User evaluations

allow to estimate the added value of the approach taken. The experiments conducted indicate that the algorithms succeed in retrieving content matching the user's interests and both models exhibit a linear scaling behavior. The algorithms perform clearly better in finding content matching several user interests when benefiting from the extra temporal information in the ontology based model.

Key words: user profiling, communication services, user generated content, context awareness

1. Introduction

Typical Internet users are nowadays overwhelmed by the ever-increasing amount of (multimedia) content available on the World Wide Web, waiting to be consumed by millions of households connected to the Internet. However, in addition to pure content consumption, the rise of the Web 2.0 paradigm facilitated the creation of content and its publication to the Web: social networks popped up everywhere (like MySpace¹ or Facebook²), many people started maintaining a weblog (commonly shorthanded to 'blogging'), and users began to upload their personal pictures to online photo galleries such as Flickr³ or Panoramio⁴ or videos to YouTube⁵, social bookmarks can be posted on Delicious⁶, etc. In 2008 there were 82.5 million content creators in the US alone (Verna, 2009), and nearly 116 million content consumers. By 2013, these numbers are forecasted to rise to 114.5 million and 154.8 million respectively. In (Kelly, 2005) the author states that a simple extrapolation of the amount of content generated by users over the last years, suggests that in the near future, everyone alive will (on average) write a song, author a book, make a video, craft a weblog, and code a program and more content will be uploaded to the Web than downloaded.

Because of the rapidly and continuously changing content landscape, users are experiencing difficulties in searching and identifying content matching their interests. In an effort to meet this challenge, new intelligent services

¹<http://www.myspace.com>

²<http://www.facebook.com/>

³<http://www.flickr.com/>

⁴<http://www.panoramio.com/>

⁵<http://www.youtube.com/>

⁶<http://www.delicious.com>

are being developed that help users in exploring this content and finding the content they need. While doing so, these services could be used to enrich existing services with this content and as such cross-fertilize between content exploration and enhanced service experience. An example of such a user-generated content based application is *unortkataster* (Academy of Media Arts, 2008), an application developed within the IST-FP6 Citizen Media research project where users can mark 'bad places' and annotate such a place with pictures, videos, comments, votes, etc. (a field trial for this application was conducted in the city of Cologne (Germany)). Another example application is *Tag Galaxy* (Wood, 2008), improving the exploration of Flickr photos by presenting tags and related tags as a virtual planetary system. Clicking on any of the 'planets' reveals a globe built up of all the photos tagged with that particular set of tags. Such tags attached to user-generated content, describing the content, are a key characteristic for social websites and the sets of all these tags are called folksonomies (Vander Wal, 2007), to distinct this approach from the more formal taxonomies.

In order to successfully match these tags with known users interests (thereby allowing to present content tuned to the individual preferences of users), building an interest profile is of prime importance. This process should ideally be non-intrusive for the end-user, and could e.g. make use of user input and output to the service or application.

In this paper we extend our previous work as described in (Strobbe et al., 2008a) and (Van Laere et al., 2008). In those papers we describe a user profile model using a keyword tree, where the keywords represent specific topics of interest. The tags attached to user generated content are used to capture the user interests. As the expressiveness of such a model is limited, in this paper we present a new model besides the keyword tree based model, that uses the more complex structure of an ontology, allowing to model additional relationships between the keywords. By using semantic knowledge combined with a number of rules processed by a reasoner, a more intelligent model is obtained where the relationships between the keywords model the behavior of the user. Examples of such relationships are: Which keywords does the user often use together? If the user is talking about topic X, what will he talk about next?

Along with the user profile models a set of algorithms is presented that learn the user interests in an automatic way, by analyzing user input and feedback. These algorithms use the resulting profile information for retrieving content and assisting the user input analysis process (e.g. when real-time

speech recognition is used). We show that the developed algorithms benefit from the extra information in the keyword ontology. As content needs to be shown in real-time for communication services, also the performance of both keyword models is evaluated.

The models and algorithms are illustrated with an enhanced instant messaging service. As the users are talking, pictures from a web hosted personal content system (in this case Flickr was used) are shown based on the topic of the conversation and the interests of the users. A user evaluation learned that the developed service is suited for use in a social context (communication with friends and family) and that the shown pictures are considered as an added value.

The remainder of the paper is organized as follows: section 2 discusses related work. In section 3 the user profile models and associated algorithms are described. The enhanced communication service is presented in section 4. In section 5 we present simulation, prototype evaluation and user evaluation results. Finally, section 6 discusses future work and section 7 states the conclusions.

2. Related Work

Acquiring an accurate user profile is not an easy task, as users are not reluctant to complete (extensive) forms or give explicit feedback (MyYahoo!, 2009), (Pazzani et al., 1996) and often forget to mention essential details. Moreover, user interests may change over time, which is hard to cover by filling in a form once. Ideally a user profile should be obtained in an automatic non-intrusive way.

A lot of research has been carried out on creating user profiles based on the browse and search history of a user aiming at improving web search results. In (Liu et al., 2004) a user profile consists of a set of categories and each category consists of a set of keywords with attached weight values. The directory structure of the Open Directory Project (ODP⁷) is used as basis. Also in (Daoud et al., 2009) ODP is used for building a graph based user profile to improve search results during a search session. In (Gauch et al., 2003) a user profile is a weighted concept hierarchy based on a reference ontology. This reference ontology is also based on web directories. A similar approach is

⁷<http://www.dmoz.org/>

followed in (Sieg et al., 2007). In (Kim and Chan, 2003) a hierarchic user profile is built up using clustering techniques. The words in visited web pages are clustered and subsequently the pages are associated with possibly several clusters. In (Stamou and Ntoulas, 2009) a topical ontology is built up based on WordNet (Fellbaum, 1998), SUMO (Pease et al., 2002) and ODP. Web pages are classified into the set of predefined topics of the ontology.

In literature several ways are proposed to personalize web search by modifying Google’s PageRank algorithm to take personal interests into account. (Richardson and Domingos, 2002) uses a more intelligent surfer model, i.e. a model guided by a probabilistic model of the relevance of a page to a query, however not making use of personal profiles. (Haveliwala, 2002) proposes a topic-sensitive pagerank algorithm using the topic of query keywords. The use of personal data analysis (bookmarks, emails, browsing patterns, etc.) is mentioned but is not further addressed. (Qiu and Cho, 2006) combines this topic-sensitive pagerank with a framework that learns the user interests based on the user’s past click history.

All these research efforts have in common that the web pages visited by the user and explicit search queries are used as input. The research presented in this paper focuses on implicit input provided by communicating with other users and the consumption and exploration of user generated content (especially pictures).

Considering user generated content, user profiles are often used in recommender systems, of which a well known example is Amazon⁸. Such systems were originally (Resnick and Varian, 1997) designed such that people could insert recommendations into a system, which are then in turn used to make recommendations to other people. Nowadays, we are moving into a direction in where recommendations are made based on the content and other knowledge sources, e.g. explicit functional knowledge about how certain object features meet user needs.

(Adomavicius and Tuzhilin, 2005) and (Burke, 2007) provide in-depth overviews of recommender techniques. Combination of these techniques (collaborative, content-based, demographic and knowledge-based) leads to the creation of hybrid strategies, combining strengths of the different practices, in an effort to eliminate (some) weaknesses of each of them. (Pazzani and Billsus, 2007) provides an overview of content-based recommendation sys-

⁸<http://www.amazon.com/>

tems, and discusses user profiles consisting of a user model and historical information. The learning aspects of modeling user preferences is discussed, along with techniques for classifying content.

In (Middleton et al., 2004) an ontology containing research topics is used for creating user profiles for a recommender system for online research papers. The user’s web browsing is used as input and user feedback is given explicitly by changing a combo-box to ‘interested’ or ‘not interested’.

As mentioned in the introduction, a common feature of user generated content is the annotation with metadata (tags). In this paper the tags attached to the pictures on Flickr are used as input for the user interests models. Another approach is described in (Diederich and Iofciu, 2006) where users have to indicate the most relevant objects in a system after which they build up a profile based on the tags attached to the objects. Yet another approach to create a user profile used in (Chirita et al., 2007), is analyzing the user’s Personal Information Repository, i.e., the collection of personal text documents, emails, cached web pages, etc.

An example of an application where user generated content is used to enrich the user experience can be found in (Kang et al., 2008). A visitor’s path and timestamp are recorded on a badge during an aquarium visit. Pictures taken during the tour can be uploaded afterwards, synchronized with the path and shared with other users. (Baladron et al., 2008) gives an overview of a number of projects that support the integration of user-generated content in a world of pervasive communications.

The research presented in this paper is complementary to the above approaches by focussing on user-generated content, using the attached metadata (tags) for the exploration of the content. Implicit user input is used by analyzing the user’s communication instead of monitoring explicit search queries. Moreover temporal relationships are used to follow the user’s current interest and take real-time aspects into account.

3. User Interests Matching

In this section the designed models for representing the user’s interests are described. Next, the algorithms for learning the interests and selecting matching content are presented.

(a) Keyword Tree

(b) Keyword Ontology

Figure 1: Examples of a keyword tree and a keyword ontology

Figure 2: Formal representation of the keyword models - the *comesNext* and *usedTogether* relationships are only part of the keyword ontology

3.1. User Profile Models

3.1.1. Keyword Tree

The first user interest representation consists a light-weight model where keywords are represented in a tree structure. An example tree is shown in figure 1(a). Each branch represents a certain concept/category. Lower level keywords represent subcategories and specific interests. Keywords may appear in multiple branches of the tree. Each node in the tree is characterized by a (normalized) weight value. This weight value represents the user specific importance of the keyword.

A more formal representation is shown in figure 2. Each node $n_{i,j}$ has an index i , a parent node j and a weight value $w_{i,j}$. The sum of all keywords on any level in the tree is normalized to 1:

$$\sum_i w_{i,j} = 1 \quad (1)$$

These weight values are adapted when input or feedback is received from the user and are used to select appropriate keywords to search for content.

3.1.2. Keyword Ontology

A tree model is limited as only the hierarchical relationship between keywords is modeled. Nowadays ontologies are often used to describe the knowledge about a certain domain of interest, and they exhibit the added benefit

of being reusable for other applications. On the other hand they show a lower performance, especially when a lot of reasoning is to be performed.

An ontology was designed based on the keyword tree, modeling a number of additional relationships between the keywords to make the model more expressive and thereby more useful for applications.

Extra relationships. In the designed ontology the hierarchical classification of keywords of the tree model is preserved but two extra temporal relationships are added. An example ontology is shown in figure 1(b).

comesNext This relation indicates that when keyword i is used, often keyword j (belonging to another branch) follows within a short time frame. This relationship could be used to predict which subset of the keyword tree will be used in the near future by an application. This might be important when for example only a limited set of the keywords can be stored in memory, due to memory or real-time constraints, or to prefetch relevant content.

A formal representation is shown on figure 2. Each instance of this relationship $c_{i,j}$ connects node $n_{i,x}$ with node $n_{j,y}$ and has an associated weight value $w_{i,j}^c$ between 0 and 1 indicating the strength of the relationship. To distinguish strong relationships from weak relationships, the sum of these weight values is normalized to 1 for all relationships originating from a keyword i :

$$\sum_j w_{i,j}^c = 1 \quad (2)$$

usedTogether This relation indicates that certain keywords (belonging to the same branch) are often used together. This relationship might be useful when selecting additional keywords (besides the recognized one) for a personalized content search (query expansion). Using a keyword tree typically high valued keywords that are related to the recognized keyword (parent, children, siblings), are used for this purpose. This relationship allows keywords that are often used together with the input keyword, but are not closely related, to be used in addition to the related keywords.

Each relationship $u_{i,j}$ connects node $n_{i,x}$ with node $n_{j,y}$ and has also an associated weight value $w_{i,j}^u$ between 0 and 1 indicating the strength of the relationship. The sum of these weight values is normalized to 1 for all relationships originating from a certain keyword i :

$$\sum_j w_{i,j}^u = 1 \quad (3)$$

Ontology Design. Figure 3 shows the designed ontology. The *KeywordTreeNode* concept is at the centre of the ontology. Such a *KeywordTreeNode* models a keyword inside the hierarchy, connects it with a *User* and contains a weight value indicating the importance of the keyword for the user. Note that the represented *relative weight values* are defined as the actual weight of a keyword times the number of children on the level of the keyword:

$$w_{i,j}^r = w_{i,j} * \sum_i n_{i,j} \quad (4)$$

This allows to compare weight values of nodes in the tree in a correct way: a node with value 0.25 having 8 siblings, has a higher importance than a node with weight 0.3 having only one sibling (which means its sibling has a weight of 0.7). The first node has an relative weight value of 2 whereas the relative weight value of the second node is only 0.6. Obviously, the average relative weight value of a node is always 1.

The ontology also contains the *ComesNextProbability* and *UsedTogetherProbability* concepts representing the strenghts of the newly defined relationships. Using rules (see section 3.2) the actual *comesNext* and *usedTogether* relationships are inferred if the associated probability concepts have sufficiently high weight values.

Note that those *ComesNextProbability* and *UsedTogetherProbability* concepts are only added to the ontology at run-time. All possible relationships between all *KeywordTreeNodes* could be added in advance, with default weight values, but this would result in a very large ontology with a poor performance. With the adopted approach the ontology of a user will become larger over time when it gets trained but as a user will never use certain keywords and combinations of keywords it will not get as overloaded as with all relationships present from the start.

As an example, consider a keyword model consisting of 1000 keywords divided in 10 categories of 100 keywords each. If every keyword is connected to every other keyword within the same category via a *UsedTogetherProbability* concept and with all keywords in the other categories via *ComesNextProbability* concepts, the ontology would contain 99000 *UsedTogetherProbabilities* and 900000 *ComesNextProbabilities* or almost one million extra concepts and

Figure 3: Keyword Ontology

associated properties in total. Suppose that in reality, a user uses half of the keywords in the model and that on average a keyword is connected with 20 other keywords via a *UsedTogetherProbability* concept and with 50 keywords via a *ComesNextProbability* concept, then the model will only contain 35000 extra concepts.

3.2. Rule Reasoning

The ontology based model allows to infer new information using a rule reasoner. A rule is defined for each of the newly defined relationships using the probability information in the ontology to infer when such a relationship is strong enough to be used by applications.

As an illustration the following rule will infer a *comesNext* relationship when there exist a *ComesNextProbability* with a relative weight value above average (larger than 1). We used Jena2 rule format as described in⁹.

```
[comesNextRelation:
(?s comesNext ?d)
<-
(?x rdf:type ComesNextProbability)
(?x hasRelativeWeightValue ?y)
greaterThan(?y '1.0')
(?x hasSourceKeywordTreeNode ?s)
(?x hasDestinationKeywordTreeNode ?d)]
```

Using rules allows to easily change the thresholds that define the minimal strenghts for the relationships. The defined rules are currently relatively straightforward, but could be more complex for a larger ontology, e.g. also containing context information about the user (e.g. location, presence status, etc.). Rules could then infer the specific situation of the user (e.g. being 'at work' or being 'at home') and infer certain relationships only in particular situations. Examples of rules inferring the situation of the user can be found in earlier work (Strobbe et al., 2008b).

3.3. Algorithms

In this section algorithms are presented for adapting the weight values in the interest models based on user input and feedback, for learning the

⁹<http://jena.sourceforge.net/inference/index.html>

Figure 4: Overview of the interactions between the algorithms

user interests, for using the profile information for retrieving content and assisting the user input analysis process. Figure 4 shows a general overview of the interactions between the implemented algorithms.

When a user communicates with another user his input is analyzed by a *Recognizer*. Based on the available set of keywords in this component, words will be recognized and their associated weight values will be updated. Based on the recognized keywords, the *Keyword Selection for Content Search Algorithm* will construct a set of relevant search terms for the communication at hand and retrieve matching content from the content repository. The retrieved content items are ranked by the *Content Ranking Algorithm* based on the goodness of the match between the keywords used for the search and the attached tags. The ranked results (URLs) are then sent to the client of the user. When the user clicks on a result feedback is provided to the *Feedback Analysis Algorithm* which updates the weight values of the attached tags that are part of the keyword model accordingly. Finally, the *Keyword Selection Algorithm* will identify a subset of keywords from the model to be used in the recognizer in case of real-time or memory constraints.

3.3.1. User Input Algorithm

User input can take several forms depending on the actual application. When using a search engine the user will type search terms or could use his voice and a speech recognizer to provide the search terms. In the case of a communication application (instant messaging, VoIP, ...) a user gives input in an implicit way by just talking to another user.

It is assumed that the recognized keywords are an indication of the interests of the user. So the weight value of a recognized keyword is increased and the weight values of the siblings are lowered.

To increase the weight value of a keyword equation (5) is used, with a and b being parameters and where $w_{i,j}$ represents the current weight value of recognized keyword $n_{i,j}$.

$$w'_{i,j} = \log_e(a * w_{i,j} + 1)/b \quad (5)$$

Using a saturating logarithmic function allows to forget the past as the function is steep for small values and flat for high values. In this way, the model captures that user interests may change over time.

As the maximum weight value a keyword can get is 1, it is enforced that the function has a fixpoint for $w_{i,j} = 1$ (i.e. $w'_{i,j}(w_{i,j} = 1) = 1$), by adjusting the parameters a and b , leaving one degree of freedom to determine the steepness of the function. The steepness allows to attach more importance to for example user feedback than user input. (In the experiments a value of 20 for parameter a is used, which corresponds to a rather low steepness.)

The weight values of the siblings ($w_{s,j}$) are renormalized such that the sum of the weights of siblings always equals 1:

$$w'_{s,j} = \frac{(1 - w'_{i,j})}{\sum_{\substack{s \\ s \neq i}} w_{s,j}} w_{s,j} \quad (6)$$

In the case of the ontology based keyword model, also the *ComesNextProbabilities* and *UsedTogetherProbabilities* are adapted by consulting the recently recognized keywords within a certain time frame. This is detailed in pseudocode in algorithm 1

Input: Set of the 10 most recently recognized keywords R_c , Subset of R_c containing the 3 most recently recognized keywords R_u , Most recently recognized keyword k_r

```

foreach  $k_i \in R_c$  do
  if  $k_i.topLevelCategory \neq k_r.topLevelCategory$  then
    // comesNext relationships are only created between keywords
    // belonging to different top-level categories
    updateComesNextProbability( $k_r, k_i$ );
  end
end
foreach  $k_i \in R_u$  do
  if  $k_i.topLevelCategory = k_r.topLevelCategory$  then
    // usedTogether relationships are only created between
    // keywords belonging to the same top-level categories
    updateUsedTogetherProbability( $k_r, k_i$ );
  end
end

```

Algorithm 1: Adaptation of *ComesNextProbabilities* and *UsedTogetherProbabilities*

Similar logarithmic and normalization functions as for the keyword weight value are used, but with different values for the parameters (indicated by a_c, b_c and a_u, b_u respectively):

$$w_{i,j}^{c'} = \log_e(a_c * w_{i,j}^c + 1)/b_c, w_{i,s}^{c'} = \frac{(1 - w_{i,j}^{c'})}{\sum_{\substack{s \\ s \neq j}} w_{i,s}^c} w_{i,s}^c \quad (7)$$

$$w_{i,j}^{u'} = \log_e(a_u * w_{i,j}^u + 1)/b_u, w_{i,s}^{u'} = \frac{(1 - w_{i,j}^{u'})}{\sum_{\substack{s \\ s \neq j}} w_{i,s}^u} w_{i,s}^u \quad (8)$$

The steepness varies with the distance between the recognized keyword and the keywords within the time frame. The smaller the distance between the two keywords the larger the steepness as this indicates a stronger relationship.

3.3.2. Keyword Selection for Content Search Algorithm

Based on the recognized keyword, and the ones related to it, a selection of search terms is made, including the recognized keyword. Related keywords are accounted for if their relative weight value exceeds a certain threshold value, indicating that the keyword is of interest to the user. For the tree based keyword model these related keywords consist of the parent keyword and sibling or child keywords. In the case of the ontology based model also keywords that are often used together (via the *usedTogether* relationship) are added. Apart from the recognized keyword the c best keywords from this set (having the highest relative weight values) are then used to search content.

3.3.3. Content Ranking Algorithm

After a search, the results are ranked. Ranking is based on the matching between the tags attached to the content and the keywords used for the search. The more tags, attached to a particular piece of content, the higher a result is ranked. When two content items have an equal number of matching tags the items are ranked based on the weight values of these matching keywords. This is illustrated in pseudocode in algorithm 2.

Input: Set of content items C , Set of search keywords S

Output: Ranked set of content items C

foreach $c_i \in C$ **do**

 | matchingTags[i] $\leftarrow$
 | countMatchingTagsWithSearchTerms($c_i.getTags()$, S);

end

// Sort according to number of matching tags and if equal according

// to relative weight values of matching tags

sort(C , matchingTags[], S) ;

Algorithm 2: Content Ranking Algorithm

3.3.4. User Feedback Algorithm

When a number of results are returned to the user, he will typically choose the content he prefers to see in more detail. A number of tags attached to the content will match the keywords of the keyword model. So, the weight values of these matching keywords will be increased and the siblings lowered. Again, the same formulas as for the User Input Algorithm (3.3.1) are used, but with different parameter values (a_f, b_f) to have bigger increases as user feedback

is very valuable information that tells more about someone's interests than a generic search term:

$$w'_{i,j} = \log_e(a_f * w_{i,j} + 1)/b_f, w'_{s,j} = \frac{(1 - w'_{i,j})}{\sum_{\substack{s \\ s \neq i}} w_{s,j}} w_{s,j} \quad (9)$$

The specific values of the parameters can depend on the kind of user feedback. An enriched communication application could not only offer an option to click on a picture but also the possibility to recommend a picture the other user.

3.3.5. Keyword Selection Algorithm

The Keyword Selection Algorithm identifies a relevant subset (the 'current keyword set') of the keyword model. This set keeps track of the keywords that can be recognized for a specified user. As content has to be shown in real-time during a communication session, it might not be possible to analyze a conversation for all keywords present in someone's keyword model. E.g. when speech recognition is used a lot of memory and processing power is needed to recognize keywords in real-time for a large vocabulary. If a server needs to handle lots of users it might be necessary to use a smaller vocabulary.

The algorithm starts by providing a number of initial keywords consisting of high-level keywords from all branches in the model. Depending on recognized keywords, related keywords are added to the subset and keywords with low weight values not belonging to the current topic of the conversation are removed.

For the keyword tree based model these related keywords consist of the children of the recognized keyword. Using an ontology as model the algorithm takes advantage of the extra defined *usedTogether* relationships. As these relationships connect the recognized keyword with keywords that have a high chance of being used in the very near future, these keywords are also added to the current keyword set.

When a lot of keywords of a certain top-level category are recognized, this might result in a current keyword set containing few keywords from other categories, resulting in a system that is not very adaptive to topic changes. To counter this, the top-level keywords are always present in the current keyword set and the algorithm assumes the topic of the conversation

has changed when no keywords are recognized for a predefined period of time and recovers in that case to a general keyword set, consisting of high-level keywords from all categories in the model. In case of an ontology based user profile, also the *comesNext* relationships originating from keywords belonging to the last recognized topic t are analyzed. The top-level categories of the keywords that make up the end points of these relationships are inspected. More keywords from categories that follow topic t most often are used to fill up the general keyword set than from other categories. In the case of the keyword tree model equal numbers of keywords from all branches are used for the general keyword set.

4. Use Case details

The goal of this use case is to provide users of an instant messaging client with content that is an added value to their conversation, i.e. photographs about the topic they are discussing that match at the same time with their personal interests. This enriches the conversation and will possibly influence it.

This section describes the use case scenario (4.1), the Context Aware Service Platform (CASP) that was used for managing the keyword models (4.2) and some implementation details (4.3).

4.1. Use Case Scenario

Figure 5: Use case overview: two users chat with each other using a web based enriched chat client

Figure 5 gives an overview of the presented use case. Two users chat with each other using a web based chat client. The contents of their conversation are redirected to a text analyzer, which is able to recognize a certain set of keywords, based on the topic of the conversation and the weight values of the keywords in the models.

The keywords are provided by CASP, where the user profiles are stored and updated. When a keyword is recognized, CASP is notified and the keyword model of the user is updated. The weight value associated with the recognized keyword is increased and the weight values of sibling keywords are lowered according to the formulas of section 3.3.1. When an ontology is used as user profile model the text analyzer is consulted for past recognized keywords to adjust the *comesNext* and *usedTogether* probabilities in the model, ensuring autonomous updating of the user profiles.

At the same time pictures are looked up in a content store (e.g. Flickr) based on the recognized keyword and related keywords with high weight values.

The resulting pictures are sent to the instant messaging clients and added in a rotating carousel of pictures. A user can click on any picture in the rotating list to see a larger version. In that case CASP is notified, as this provides user feedback, and the keyword model for that user is updated.

The keywords that make up the keyword models of the users need to be sufficiently distinguishing for the topic of the conversation. This allows the application to follow the conversation and to detect the current topic and context. Having an idea about the topic of the conversation allows the application to retrieve better matching pictures from Flickr.

Figure 6: Screenshot of the user interface - trained model

Figure 6 provides a screenshot of the user interface of the developed use

case showing the online buddies on the right, the chat conversation in the middle, a carousel of 10 pictures at the bottom left and the picture the user most recently clicked on the left. This figure also illustrates the effect of a trained model on the provided pictures. The users start by talking about cycling and then about the Champs Elysées in Paris. As a result pictures about the traditional last stage of the Tour de France on the Champs Elysées are shown. In the case of an untrained model the typical travel guide pictures of the Champs Elysées are shown with the Arc de Triomphe, traffic by night, etc.

4.2. CASP

Figure 7: Overview of the CASP architecture - components used by the presented use case are indicated with rectangles.

For the retrieval and interpretation of the user interests the Context Aware Service Platform as described in (Strobbe et al., 2006) and (Strobbe et al., 2007) was used. This platform, inspired by the work of (Gu et al., 2005), takes care of the aggregation and abstraction of different kinds of context information. The information is gathered in a knowledge base, typically using ontologies as data model. A rule based reasoner allows to analyze and validate the information and to infer new information. The information in the knowledge base is offered to services and third parties.

Figure 7 gives an overview of the architecture of CASP consisting of different layers with for each layer a number of developed components. The components used by the presented use case are indicated with rectangles. Most other components are used by the desk sharing office use case as described in (Strobbe et al., 2007) where focus is mainly on location and presence information.

The *persistence layer* ensures persistence of context information, i.e. the storage of the user’s profile between two chat sessions. The *device layer* includes the chat clients that deliver user input and feedback. The *context gathering layer* takes care of the acquisition of specific context information. Here this information consists of the recognized keywords delivered by the Text Recognizer component and user feedback. The *context framework layer* is responsible for the aggregation of the context information according to the developed context model and the derivation of implicit information by reasoning. Information coming from the context gathering layer is translated to OWL (W3C, 2004) by the context providers and gathered in a knowledge base. Derivation of extra knowledge or validation of the information in the knowledge base is done by providing rules. These are executed by a reasoner using a rule engine providing a forward chaining, a backward chaining and a hybrid execution model. Finally in the *application layer* the information from the knowledge base is used by the algorithms responsible for selecting keywords for the content search, ranking the content and assisting the user input analysis. The resulting content items (URLs) are then offered to the chat clients via a web service.

4.3. Implementation

4.3.1. Keyword Model

In order to construct a keyword model to represent the knowledge of the user interests, different approaches can be followed. As mentioned in section 2, web directories like the Open Directory Project are used to construct a hierarchical structure of categories, possibly combined with other sources like WordNet. An example can be found in (Stamou and Ntoulas, 2009).

For the purpose of the prototype, another approach was adopted, by selecting 4 categories of interests: music, travel, photography and sports. Google’s Adwords¹⁰ tool was used to generate keywords about those cate-

¹⁰<https://adwords.google.com/select/KeywordToolExternal>

gories. A number of the returned keywords were used again as new search queries to create the hierarchical structure of the tree (e.g. when classical is returned by a query for music, a new query to search for keywords connected to classical was performed). To improve the quality of the keywords the resulting keywords were filtered by hand removing ambiguous words (like 'Nice' (French town) versus 'nice' (the adjective)) and duplicates. This method resulted in a keyword tree containing 796 nodes.

4.3.2. *OWL*

The developed ontology is implemented in OWL (Web Ontology Language (W3C, 2004)), an ontology language proposed by W3C, as a vocabulary extension of RDF. OWL allows automated processing of terms and relationships between terms in vocabularies, by representing the meaning of those terms. Domain knowledge can be accurately described by means of classification, modeling dependencies and restrictions on these dependencies. Other ontologies can be imported, encouraging reuse.

4.3.3. *Jena2*

For the implementation of the knowledge base and reasoner the Jena2 Semantic Web Toolkit¹¹ was used. This Java library offers an OWL API and a rule-based inference engine.

4.3.4. *Flickr Web Service*

Images are retrieved from Flickr using an XML-RPC¹² based approach. A first implementation used FlickrJ¹³, a Java implementation of the Flickr API¹⁴, but this implementation proved to be rather slow. The average retrieval time for 5 photos was around 5 seconds. With the XML-RPC based approach retrieval times are 3 to 6 times faster.

4.3.5. *JEE application server*

The communication between the chat clients was implemented using a Glassfish JEE server¹⁵. Direct Web Remoting¹⁶ was used for the interaction

¹¹<http://jena.sourceforge.net/>

¹²<http://www.xmlrpc.com/>

¹³<http://sourceforge.net/projects/flickrj>

¹⁴<http://www.flickr.com/services/api>

¹⁵<https://glassfish.dev.java.net>

¹⁶<http://directwebremoting.org/dwr/index.html>

between the javascript in the web based chat client and the Java code on the server. This allows interaction in two ways: browser calling server and server calling browser.

5. Evaluation details

5.1. Comparison of the algorithms

To verify the correctness of the algorithms and compare them for the keyword tree and ontology based representations of the user interests a number of simulations were performed. For that purpose arbitrary keyword models were generated and populated with code words. The construction of the trees was bound to a specified minimum and maximum number of nodes on each level, but the exact number was randomly chosen between these boundaries. The size of the model was specified and fixed, depending on the experiment. The code words are unique string identifiers generated as all possible combinations of the letters of the alphabet (e.g. 'aa', 'ab', ...).

5.1.1. User Input & Feedback Algorithm

Figure 8: Comparison of keyword weights for three types of keywords: a keyword that represents a real interest of the user, a sibling of such an interest and a random keyword. A value of 40 was used for parameter a

As the algorithms take user input and feedback into account they learn the specific interests of a user. Especially user feedback is very valuable information. In this series of simulations, the impact of this user feedback on the weight values of the keywords is verified.

For this simulation a keyword model was used consisting of 500 nodes, and random conversations consisting of 1000 words were simulated. Each result is the average value of 25 runs of the experiment. The weight values are relative weight values (being the actual weight times the number of children on the level of the keyword as explained in section 3.1.2). Three types of keywords were tracked: First, the weight of a keyword that is of interest to the user was characterized. The user gives feedback when content containing this keyword as tag, is presented to him. The second keyword is located on the same level as the interest of the user but receives no feedback and the third keyword is a randomly chosen one, with the constraint that it is not competing with any interests.

As can be seen in figure 8 the random keyword stays around the average relative weight value of 1 as it just undergoes its recognitions, in a similar way as all other keywords. The keyword that represents an actual interest of the user gets a very high weight value as a result of the user feedback. As the sum of the absolute weight values of sibling keywords is always 1, the sibling keyword gets a very low weight value.

5.1.2. Keyword Selection Algorithm

In this series of simulations we evaluate the adaptivity of the Keyword Selection Algorithm which selects a relevant subset of keywords from the keyword model.

As the algorithms are primarily intended to enhance communication sessions, it is particularly important that the shown content closely follows the topic of the conversation for a good user experience. When a user's keyword model contains the most popular keywords for the defined categories and enough resources are available to analyze the conversation for all keywords in the model, keywords will be often recognized and the content will nicely track the conversation. In situations where only a subset of the keyword model can be used, e.g. because a server handling a lot of users, can only recognize a limited number of words in real-time due to memory or CPU constraints, the selection of this subset has to be adaptive to topic changes.

Conversations were simulated by generating random keywords from the keyword model. Consecutive keywords come from the same main category

and every 250 keywords a category switch occurs to simulate a change in the conversation topic.

When testing the keyword ontology a set of 1 to 5 *usedTogether* keywords was predefined for every keyword. During the simulations there was a 50% chance that a keyword from this set was generated as next keyword. A similar set of 1 or 2 *comesNext* keywords was predefined for every keyword. This set was consulted to define the new topic in the case of a topic switch. As the ontology needs to be trained before these relationships will be present, conversations consisted of a large number of keywords (10000). Only for a trained model better results from the ontology model than from the tree model are expected.

Figure 9: Comparison of the performance of the Keyword Selection Algorithm for tree and ontology based models. The bars show the average values and the whiskers the average maximum and minimum values over different current keyword list sizes.

Figure 9 shows the results for different sized keyword trees. Every result is the average over different current keyword list sizes (from 10 to 100 keywords) and 10 different runs. Also the maximum and minimum values (averaged over different current keyword list sizes) are shown. The results of the algorithm show considerable improvement in comparison with a randomly filled current

keyword list: in the latter case a recognition rate of $\frac{|C|}{|K|}$ is expected with C the set containing the keywords from the current keyword set and K the set containing all keywords from the keyword model.

The improvement is due to the fact that the current keyword set dynamically follows the topic(s) of the conversation, whereas a randomly chosen current keyword set never changes.

The ontology keyword model performs better than the tree model as the current keyword set will contain some *usedTogether* keywords after a while which have a higher probability to be generated. After a number of topic switches also some *comesNext* relationships will be present in the ontology resulting in a current keyword set containing more keywords from such *comesNext* categories when a recovery to a general current keyword set occurs. This happens when no keywords are recognized during a certain period of time and this happens often after a topic switch.

Observing the average standard deviation for the number of recognized keywords for the tests with different sized keyword models and current keyword lists a value of 92.33 was obtained for the keyword tree and 208.06 for the keyword ontology. This larger standard deviation for the ontology is due to the more dynamic nature of the model. During the tests some *comesNext* and *usedTogether* relationships will appear in the ontology. As these relationships are based on the random generated keywords, the number of these relationships will vary from run to run and consequently the results of the Keyword Selection Algorithm that uses these relationships, will have a larger spread.

When real-time speech recognition is used for analyzing the conversation instead of an instant messaging client, an estimate of the number of unrecognized keywords can be extracted from information about conversations (number of words generally spoken in a certain time interval) and the time the conversation is going on.

5.1.3. Keyword Selection For Content Search Algorithm

To test the selection of keywords that are used to search for content in a content repository and test the influence of user feedback, a content store containing 20000 content items each having 10 tags selected from the same top-level category was simulated. The user interests were generated by randomly selecting 5 keywords in each category. If content is retrieved from the content store with a tag matching one of these interests, user feedback is given. Three extra keywords (besides the recognized keyword) are used for

the content search (parameter c of the algorithm).

For the ontology keyword model it is important that there are a number of *usedTogether* relationships present in the model as the Keyword Selection For Content Search algorithm checks this relationships to determine the set of search keywords. So the simulated conversations are again quite long (5000 keywords). For each randomly generated keyword 25 content items were retrieved from the content store. For each retrieved content item it was checked if one or more of the tags match with the defined user interests.

Users will typically use the keywords they are really interested in, often during a conversation. This was simulated by defining an interest probability that indicates the chance a simulated keyword is a predefined user interest. Tests for different interest probabilities ranging from 0% till 100% were performed.

Figure 10: Comparison of the number of content items that match user interests

Figure 10 shows the percentage of content items that match at least one or several user interests for both keyword models and for different values of the interest probability. The ontology model performs slightly better for the number of content items matching at least one interest. The difference is not pronounced as the simulated user interests are often generated. In such cases the retrieved content items will always contain this interest as

a tag. Differences are larger inspecting content items matching more than one interest. By simulating the frequent occurrence of user interests a lot of *usedTogether* relationships will appear between these keywords (especially for high interest probabilities). These *usedTogether* keywords are taken into account by the algorithm resulting in a lot of content items matching several user interests.

5.2. Performance evaluation

The performance of the algorithms was evaluated by measuring the processing time of words that are recognized as keywords. When a word is recognized almost all algorithms are put into action as can be seen on figure 4. The weight value of the recognized keyword is increased, in the case of a keyword ontology model *UsedTogetherProbabilities* and *ComesNextProbabilities* are created and/or adapted, keywords are selected to search for content, the content is fetched from Flickr, the retrieved content is ranked and possibly a new current keyword list is determined. Fetching the content from Flickr takes most time by far, so we measured the time of the Flickr call separately.

Figure 11: Average processing times for recognized keywords for different sized keyword trees (without Flickr call)

The measurements were performed on a rack mount PC with a dual core AMD Opteron processor (2 GHz) and 4 GB RAM. Measurements were

performed on simulated conversations that were sufficiently long (10000 keywords) in the case of an ontology keyword model to take into account that the model is trained during the test.

Figure 12: Average processing times for recognized keywords for different sized keyword ontologies (without Flickr call)

On average Flickr access time was 2276 ms. Figures 11 and 12 show the average processing times without Flickr access for different sized keyword models. The graphs show a linear trend between processing times and the sizes of the models, but processing times are a lot bigger for the ontology model than for the tree model. Performance of ontologies still is a known issue and further research is needed to improve this, as the reasoning step is slow. On the other hand the ontology contains more information than the tree model. This aspect combined with its formal characteristics makes this approach better reusable for other applications and services.

As explained in section 3.1.2, a particular characteristic of the ontology model is the addition of *ComesNextProbability* and *UsedTogetherProbability* concepts at runtime. So, a trained ontology will contain a lot more concepts and relationships than an untrained version and processing times will be higher. This can be seen on figure 12 which also contains the average processing times for the first 100 and last 100 keywords of the simulated conversations.

Figure 13 shows the memory usage of the application during a simulated conversation of 10000 keywords between 2 users. Results are similar to the

Figure 13: Memory usage for 2 users and a keyword model containing 1000 keywords

processing times. Memory usage is higher for the ontology model than for the tree model and increases during the conversation, as the model gets trained. Memory usage for the tree model is almost constant. Figure 13 gives also an illustration of the default garbage collection behavior of Java. Objects that are no longer in use are automatically freed by the garbage collector that is part of the Java Virtual Machine. Garbage collection may happen at any time, typically when a program is idle or when there is insufficient memory available on the heap. In this case the zigzag form of the graph indicates garbage collection occurred at regular intervals.

5.3. User Evaluation

To evaluate the use case a user test with 15 participants was conducted. The set-up was a little different as the one presented in section 4. The users communicated via VoIP and could see each other over a webcam. Pictures were shown in a carousel and users could enlarge them by simply pressing a button on a remote control.

Users talked with a researcher (located in another room) for about half an hour and were asked to fill in a questionnaire to evaluate their experiences with the application. A second researcher listened to the conversation and provided the recognized keywords into the system (Wizard of Oz testing (Kelley, 1983)).

As half an hour is quite short to train the user profile for a user, participants were asked to indicate their specific preferences relating to four topics

Table 1: Main questions of the user evaluation questionnaire

Question	Average value
Would you use this system to talk to people you know about subjects that are of interest to you?	3.9
Do you think that the fact that the users see the same pictures during the conversation can have an added value?	3.7
This application is suited for communication in a social context, for example with friends or acquaintances	4.5
Would you like to use your own pictures in a conversation with people you know?	4.5
How much time did you look at the windows containing your chat partner, and how much time did you look at the pictures?	61.7% (chat partner) 38.3% (pictures)
Can you indicate how many pictures did or did not match the topics of your conversation?	28.2% (not related) 22.6% (don't know) 49.2% (related)

(photography, travel, sports, music) in advance.

Table 1 shows a number of the questions of the questionnaire. Most of the answers in the questionnaire took the form of a 5-scale Likert (Likert, 1932). To interpret the results of these questions, the average score on each question was examined.

The questionnaire results show that the shown pictures are considered as an added value although the actual video-chat remains the main feature. About half the time, the users were convinced that the shown pictures were related to the conversation. The users indicated that they would like to use their own pictures instead of the randomly selected pictures from the Internet which were used during the test. Finally, we see that the users like the system, especially for communicating with family and friends (4.5/5).

6. Future Work

At this moment the models contain dynamically changing weight values to represent the importance of a certain keyword for a certain user. However the keyword hierarchies in the models are statically defined. It requires a lot of work to build such a model. The models built for test purposes were quite small, only containing keywords in four categories (music, sports, travel and photography), but it would require a lot of effort to extend this to all possible categories. Also (Stamou and Ntoulas, 2009) indicate they needed nearly three months to build and validate their topic ontology. It would be interesting if the system could automatically add new keywords to the model that are often found attached to content consumed by the user. And in the same way, keywords that are never used could be removed. A possible solution could be to add an extra relationship *possibleNewKeyword* in the ontology between keywords already in the model and tags attached to retrieved content items that are not present in the keyword model. When such tags turn up regularly they could be promoted to real keywords, possibly as siblings of the keywords they appear together with most.

Another approach could be to cluster all the tags of social websites (Begelman et al., 2006) and put only the identifying keywords of these clusters in the keyword models. This would make the keyword models considerably smaller. Flickr, the content store we used for the implementation of the use case knows this concept of clusters¹⁷.

User interests also are dependent on the context. When communicating with a colleague at work you are typically interested in other things than when you are chatting with friends at home. As a consequence, it might be interesting to attach several weight values to the keywords in the model matching with different contexts to improve the quality of the retrieved pictures. Of course some context detection system is needed for this to work. This could be as simple as classifying your buddies in 'colleagues', 'friends', 'family', etc. but could also be based on location, time of the day, day of the week (weekend or not) or presence status. CASP, the context framework described earlier in this article is well suited for this task. More information can be found in (Strobbe et al., 2006), (Strobbe et al., 2007) and (Strobbe et al., 2008b) .

¹⁷<http://blog.flickr.net/en/2005/08/01/the-new-new-things>

7. Conclusions

In this paper a novel way to enrich applications with user generated content was presented. By taking the interests of the user into account the user experience is increased. We proposed and compared two ways to model the user's interests: a light-weight keyword tree and a more complex keyword ontology. The ontology contains extra temporal relationships to capture the typical sequence of keywords and topics used by the user.

Also several algorithms were presented for learning the user interests in an automatic way, by analyzing input and feedback from the user. The resulting profile information is used for retrieving content from Flickr to enrich applications. Applicability of the presented approach was illustrated with an enhanced instant messaging service where pictures are shown related to the conversation and the particular interests of the users.

Through simulations it was shown that the developed algorithms succeed in learning the user interests, following a conversation topic and retrieving content matching the interests, and that they can benefit from the extra information in the keyword ontology. For applications with hard real-time constraints the keyword tree might be better suited as the performance of ontologies is still a bottleneck.

A user study showed that users are enthusiastic about the use case and are convinced that it is well suited for use in a social context: communicating with family, friends and acquaintances. They find the shown pictures an added value and indicated that the service could be further improved by automatically using their own pictures. There are opportunities for automatic tagging services to relieve the user from the time consuming task of tagging all of his pictures.

Acknowledgement

The research presented in this paper is partially funded by the IST-FP6 Citizen Media project.

Matthias Strobbe is a research assistant of the Fund for Scientific Research - Flanders (F.W.O.-Vlaanderen).

References

Academy of Media Arts, 2008. online.
URL <http://unortkataster.de/>

- Adomavicius, G., Tuzhilin, A., June 2005. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *Transactions on Knowledge and Data Engineering* 17 (6), 734–749.
- Baladron, Carlos, Aguiar, J., Carro, B., Sanchez-Esguevillas, A. N., Baldauf, M., Fröhlich, Hlich, P., Musialski, P., Falcarin, P., Rocha, O. R., Costabello, L., Goix, L. W., Cadenas, A., Paganelli, F., Parlanti, D., Giuli, D., Pimentel, M., a, Cattelan, R., Melo, E., Teixeira, C., Raibulet, C., 2008. Integrating user-generated content and pervasive communications. *Pervasive Computing, IEEE* 7 (4), 58–61.
- Begelman, G., Keller, P., Smadja, F., 2006. Automated tag clustering: Improving search and exploration in the tag space.
URL http://www.pui.ch/phred/automated_tag_clustering/
- Burke, R., 2007. The Adaptive Web. Vol. 4321 of *Lecture notes in Computer Science*. Springer Berlin / Heidelberg, Ch. Hybrid Web Recommender Systems, pp. 377–408.
- Chirita, P., Firan, C. S., Nejdl, W., 2007. Personalized query expansion for the web. In: *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, pp. 7–14.
- Daoud, M., Tamine-Lechani, L., Boughanem, M., Chebaro, B., 2009. A session based personalized search using an ontological user profile. In: Shin, S. Y., Ossowski, S. (Eds.), *SAC*. ACM, pp. 1732–1736.
- Diederich, J., Iofciu, T., 2006. Finding communities of practice from user profiles based on folksonomies. In: *Proceedings of the 1st International Workshop on Building Technology Enhanced Learning solutions for Communities of Practice (TEL-CoPs'06)*, co-located with the First European Conference on Technology-Enhanced Learning.
- Fellbaum, May 1998. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press.
- Gauch, S., Chaffee, J., Pretschner, A., 2003. Ontology-based personalized search and browsing. *Web Intelli. and Agent Sys.* 1 (3-4), 219–234.

- Gu, T., Pung, H. K., Zhang, D. Q., 2005. A service-oriented middleware for building context-aware services. *Journal of Network and Computer Applications (JNCA)* 28(1), 1–18.
- Haveliwala, T. H., May 2002. Topic-sensitive pagerank. In: *Proceedings of the Eleventh International World Wide Web Conference*. Honolulu, Hawaii.
- Kang, Y., Stasko, J. T., Luther, K., Ravi, A., Xu, Y., 2008. Revisitour: Enriching the tourism experience with user-generated content. Springer, pp. 59–69.
- Kelley, J. F., 1983. An empirical methodology for writing user-friendly natural language computer applications. In: *CHI '83: Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, pp. 193–196.
- Kelly, K., 2005. online.
URL <http://www.wired.com/wired/archive/13.08/tech.html>
- Kim, H. R., Chan, P. K., January 2003. Learning implicit user interest hierarchy for context in personalization. In: *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*. ACM Press, New York, NY, USA, pp. 101–108.
- Likert, R., 1932. A technique for the measurement of attitudes. *Archives of Psychology* 22 (140), 1–55.
- Liu, F., Yu, C., Meng, W., 2004. Personalized web search for improving retrieval effectiveness. *IEEE Transactions on Knowledge and Data Engineering* 16 (1), 28–40.
- Middleton, S. E., Shadbolt, N. R., De Roure, D. C., 2004. Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.* 22 (1), 54–88.
- MyYahoo!, 2009. online.
URL <http://my.yahoo.com/>
- Pazzani, M. J., Billsus, D., 2007. Content-based Recommendation Systems. *Lecture Notes in Computer Science* 4321/2007, 325–341.

- Pazzani, M. J., Muramatsu, J., Billsus, D., 1996. Syskill & webert: Identifying interesting web sites. In: 13th National Conference on Artificial Intelligence. Vol. 1. Portland, OR, US, pp. 54–61.
- Pease, A., Niles, I., Li, J., 2002. The suggested upper merged ontology: A large ontology for the semantic web and its applications. In: In Working Notes of the AAAI-2002 Workshop on Ontologies and the Semantic Web. p. 2002.
- Qiu, F., Cho, J., 2006. Automatic identification of user interest for personalized search. In: WWW '06: Proceedings of the 15th international conference on World Wide Web. ACM Press, New York, NY, USA, pp. 727–736.
- Resnick, P., Varian, H. R., March 1997. Recommender systems. Communications of the ACM 40 (3), 56–58.
- Richardson, M., Domingos, P., 2002. The intelligent surfer: Probabilistic combination of link and content information in pagerank. In: In Advances in Neural Information Processing Systems 14. MIT Press, pp. 1441–1448.
- Sieg, A., Mobasher, B., Burke, R., 2007. Web search personalization with ontological user profiles. In: CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. ACM, New York, NY, USA, pp. 525–534.
- Stamou, S., Ntoulas, A., 2009. Search personalization through query and page topical analysis. User Modeling and User-Adapted Interaction 19 (1-2), 5–33.
- Strobbe, M., De Jans, G., Hollez, J., Goeminne, N., Dhoedt, B., De Turck, F., Demeester, P., Pollet, T., Janssens, N., June 2006. Design of an open context-aware platform enabling desk sharing office services. In: Proceedings (on CD-ROM) of PSC2006, the 2006 International Conference on Pervasive Systems & Computing (part of the 2006 World Congress in Computer Science).
- Strobbe, M., Hollez, J., De Jans, G., Van Laere, O., Nelis, J., De Turck, F., Dhoedt, B., Demeester, P., Janssens, N., Pollet, T., May 2007. Design of casp: an open enabling platform for context aware office and city services.

- In: Proceedings of MUCS 2007, 4th International Workshop on Managing Ubiquitous Communications and Services.
- Strobbe, M., Van Laere, O., Dauwe, S., De Turck, F., Dhoedt, B., Demeester, P., 2008a. Efficient management of user interests for personalized communication services. In: Proceedings of the 5th International IEEE Workshop on Management of Ubiquitous Communications and Services, held as part of IEEE NOMS 2008. pp. 1–8.
- Strobbe, M., Van Laere, O., Volckaert, B., De Turck, F., Dhoedt, B., Demeester, P., 2008b. Context based selection of user generated content. In: Arabnia, H. R., Marsh, A. (Eds.), SWWS. CSREA Press, pp. 100–106.
- Van Laere, O., Strobbe, M., Dauwe, S., Dhoedt, B., De Turck, F., Demeester, P., Verde, O., Hülken, F., 2008. Interest Based Selection of User Generated Content for Rich Multimedia Services. In: Proceedings of the 9th International Workshop on Image Analysis for Multimedia Interactive Services 36–40
- Verna, P., 2009. online.
URL <http://www.emarketer.com/Article.aspx?R=1006914>
- W3C, 2004. OWL Web Ontology Language Overview. [online],
<http://www.w3.org/TR/owl-features>.
- Wood, S., 2008. online.
URL <http://taggalaxy.de/>
- Vander Wal, Thomas, 2007. online.
URL <http://www.vanderwal.net/folksonomy.html>