Webograph: A Selective Publication Model for Bloggers

Vincent Oria[+], Phan Quang Trung Tien[±], Jean-Marc Saglio[§]

[+]Department of Computer Science, New Jersey Institute of Technology, USA

[§]Ecole Nationale Supérieure des Télécommunications, Paris, France

[±]LABRI, Bordeaux, France

vincent.oria@njit.edu, phanquan@labri.fr, saglio@enst.fr

ABSTRACT

We believe that the important architectural features for the next generation web publishing systems will be: semantic classification and selective access to information. As publishing on the Web is becoming the leading publishing channel, the publishing systems should allow the publisher to more effectively control the access to the information that is being published. The current "share to all or share with friends only" model is limited even when the access is restricted with a password, and is very far from the way people share information in real life: What is said in private is not always what is said in public and what is referred to as public can further be categorized. The Webograph project explores a restrictive publishing model, called the "*iceberg visibility model*", based on dynamic views defined on classes of RSS feeds and reader groups.

Keywords

Semantic web, Peer-to-Peer Systems, Really Simple Syndication (RSS), Views.

1. INTRODUCTION

The World Wide Web through tools like XHTML or XML content edition, RDF description, RSS/Atom publication and aggregation has enabled the creation of several *social networks* of different types and dimensions: networks of newspaper publishers and their interactive readers; networks of domain experts (e.g. marketing, finance, science, etc..) and scattered librarians or information agencies; networks of bloggers collecting and sharing information; networks of administrations, organizations and active citizens who spend a large amount of their time referring to same web resources; and networks of specialized networks!

Several different *content management systems* are proposed to help implement these networks with more or less functionalities. But the common features of these systems are: "(1) Uniform Resource Identification (URI), (2) public directories, thesauri or ontologies and (3) RSS/Atom channels!". Actually sharing in all these networks means (1) unique references, (2) global interpretation and (3) communication channels between agents. All *heavy content system* providers on the Web present a detailed list of RSS/Atom content channels to readers who want to selectively subscribe and read through their personalized reading portals (like iGoogle[1] or Netvibes[2]) or directly through their email client (like Thunderbird[3]). At the opposite end *light content systems* and many *blogs* offer to syndicate (republish content of) others and reclassify them under local interpretations.

In the way the World Wide Web is currently perceived, everything is public. Search engines index billion of pages in order to make them known and easily accessible. Web sites owners are doing their best to be well ranked on the pages returned by popular search engines. This model works fine for some businesses that need to be known to exist. But the real world is far more complex: a large amount of valuable information or useful knowledge is neither directly accessible by anonymous readers nor by web crawlers. Often portals and databases limit access to only registered users and/or well-formed queries because of economic or social reasons. Hence, the real World Wide Web is already divided between a "surface Web" and a "deep web" as defined in [Berg01]. The surface Web is open to the public while the "deep Web" allows interchanges between identified partners only.

---

[1] www.google.com/ig

[2] www.netvibes.com/

[3] http://www.mozilla.com/en-US/thunderbird/

Even between identified partners, all information or knowledge is not always shared. In real life, a person gradually reveals himself/herself to a partner as he/she gets more and more comfortable with that partner. So on the Web, the amount of information shared should be adjustable according to reciprocal trust which could vary in the lifetime of partners. That means that different partners can get different information from the same site depending on the degrees of trustiness assigned to them by the site they are visiting. Our work encompasses database, portal and weblog technologies for P2P networks and addresses the requirement of trust-based selective publishing.

With the emergence of Really Simple Syndication (RSS), and various news readers/ feed readers, it has become easy for the readers to subscribe to the RSS feeds or the semantic web resources he/she wants to access. There exist also a large number of web publishing systems but none of them have the feature of selective publishing to selective readers. They usually use the method of share to all or share to none. In our current implementation of the publishing system, the publisher can create different views on the hierarchy of topics, and allow selected groups of users to access these views. The readers in a group have the permission to access the views the group have access to. Another noticeable trend of practice in this area is the *restrictive publishing policy*: according to "who you are" a server will disclose to you only the part of its content you are entitled to read and/or copy. The “deep” or “invisible” Web is implemented either by a portal (one has to login with a password) or by a personal identity number/code. They are needed in order to be able to subscribe to a RSS/Atom channel. The server manages the client identities and links them to the contents.
The rest of the paper is organized as follows: Section 2 discusses issues related to personalized storage spaces, Section 3 presents the “Iceberg model” used to implement the selective publishing model, Section 4 presents an example of application, Section 5 describes successive prototypes and Section 6 presents the related work.

2. A PERSONALIZED CONTENT BASE

The data management is a peer-to-peer model. Each peer has a personal memory space on which he/she is the only author and administrator. Let us call it "Personal Content Base" (PCB) as in [TSP04]. This content aims at storing the authors write-ups (personal notes, e-mail, blog articles, annotated excerpts of web pages, etc...). The diversity of the write-ups and their diverse formats is certainly source of difficulties in term of management but gives more flexibility to the authors. The write-ups are seen by the peers only when the author decides to make them public.

2.1. Hierarchic Organization of Topics

The write-ups can be grouped in folders and the folders can be semantically organized based on an ontology. The ontology can be complex but in the case of simple publication application a tree of topics is enough as in Figure 1. We will use the term topic to refer to a folder and say that a write-up belongs to a topic when the write-up is in that topic. This simple structure is also adequate to manage different levels of confidentiality for the topics, sub-topics as well as the files they contain. Giving access to a topic gives also the right to read all the write-ups that belong to the topic. This rule allows to attach a RSS/XML feed to a topic.

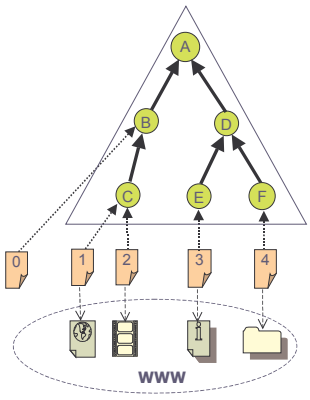

Figure 1 A Personal Content Base

2.2. Linking Personal Content Bases

Although a writer owns and manages a PCB, it is possible for him/her to refer to a topic of another peer on which he/she has the right to access and reference the write-ups. We distinguish two types of links: Link between topics and link between write-ups:

- Inclusion: A topic in a writer PCB can include the topic in another PCB. Although we are concerned about maintaining confidentiality, there is no need of duplicating the topics. If a writer finds that another peer has an interesting topic, he/she can decide to link the topic of this peer with one his/her topics in the same way a Web page can refer to an external Web site. This assumes the writer the permission to access the posts. The original creator gets full credit for the write-ups.
- Reference: A writer can reference a write-up of another peer to comments it, answer it or to simply reference and use it as a source of information in his/her own write-up.

With these two links inter-PCB, the PCBs form a system multi-base in which each peer can freely incorporate in his local space the topics or the write-ups of others (Figure 2).

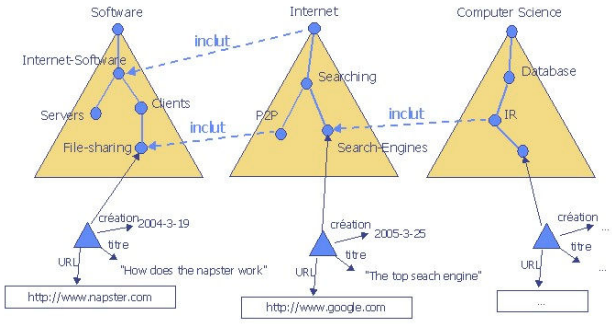

Figure 2: A multi-Personal Content Base

3. THE « ICEBERG VISIBILITY » MODEL

The iceberg model proposes to classify objects of the same type according to a visibility scheme where the uppermost object class is seen by everybody whilst the objects in the leave class nodes are viewed by hand-picked readers.. The main differences between the iceberg model and an object-oriented model are:

- All the objects in the Iceberg Model have the same data structure. So the hierarchy is more about defining semantic groups.
- The definition of the deep extent of a class: In the Iceberg model, the more you go towards the leaf nodes, the more objects you see.

The Iceberg model is a classification model of objects of the same type (or class). We need more explanation here after all the meanings given to this term by the object model. All the objects in the Iceberg model refer to objects with the same properties: Same programmatic interface, same implementation of the programmatic interface and same internal machine representation. This makes the *type* in the Iceberg model simpler: just extent management.

Like object models [LOS 98], the Iceberg model is a rooted hierarchy of classes with a super-class/ sub-class relationship. Since all the objects have the same properties and a class like a semantic cluster. The main difference with object models lies in the definition of the deep extent of a class. Unlike object models, in the Iceberg model, the more you go down in the class tree, the more objects you see. In the publishing application for example, the idea is to make the objects in the root class public and make the objects in the specific classes accessible to only users that have the right to. This also means that different users will have different views on the objects published.

3.1 Concept Definitions

Publishers create their own topics/posts privately. They create, on their topics base, views for external visibility.

Definition 1: Class Path:

Given a class *C*, which is a node in the class hierarchy, the path defined by *C* is the ordered list of classes *path(C)=[C, ..., R]* from C to the root *R*. The path of a class includes all the ancestors to the root.

Definition 2: Deep extent of a class:

Let denote *by Shallow_extent(C),* the shallow extent of a class C and *deep_extent (C)* its deep extent. Then $Deep\_extent(C)=\cup_i Shallow\_extent(C_i),\ C_i \in Path(C)$.

A class, which is a node in the class hierarchy, defines a path that includes all the ancestors to the root. A deep extent of a class includes all the objects in all the classes along the path defined by the class.

Definition 3: Shear of a set of classes:

Given a subset $X=\{C_1,...,C_n\}$ of classes in the class hierarchy, $Shear\ (X) = \cup_i Path(C_i),\ C_i \in X$.
If the classes in *X* are the most specific classes we would like a user to reach, then the shear defines a sub-tree composed of all the classes accessible by that user. Figure 3 illustrates the concept: The light blues circles represent the most specific classes and the doted line represent the shear defined by the selected classes.

Lemma 1: Shear Union: From the definition of a share we can easily prove the following equality that if X_1 and X_2 are subsets of classes in the class hierarchy then:

- $Shear(X_1 \cup X_2) = Shear(X_1) \cup Shear(X_2)$

Proof (sketch): Straightforward from the definition of path and shears. There are basically 2 cases:

- $\forall C_i \in X_1\ \forall C_j \in X_2$ $Path(C_i) \cap Path(C_j) = \varnothing$: In this case, there is no common path between the shears and the resulting shear contains all the paths from both shears.
- $\exists C_i \in X_1\ \exists C_j \in X_2$ $Path(C_i) \cap Path(C_j) \neq \varnothing$: In this case, there are some common paths and since the longest path of $\{Path(C_i), Path(C_j)\}$ contains the shortest one, the 2 shears add up to form a larger shear.

The shear union property can be used to define more complex shears and the Shear notion can be used to define dynamic views.. The publisher can define views by means of *shears*, to present to readers or group of readers only the topics (class) that the publisher wants them to see. With the Iceberg model, when a topic is visible its ancestors are too.
This model is suitable to implement various security and privacy preserving publishing applications. The publisher can store the most secure/private information towards the leaves. The privacy of the information keeps on increasing as you go towards the leaves. The publisher can define views consisting of topics, which are deep in the hierarchy using this model, and give the groups that are more confidential and interactive the permission to this view.

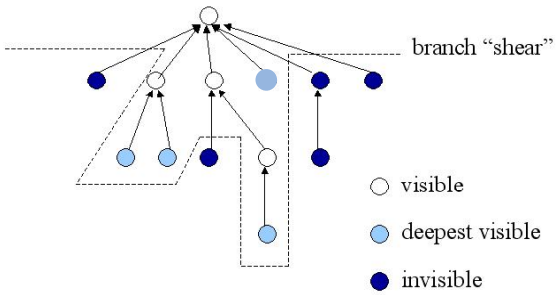

Figure 3. An example of shear

The Iceberg model proposes a progressive disclosure of topics. The publisher can disclose more information to a reader as he starts having more confidence or has more interaction with the reader.

An important property of shears is that the union of shears is a shear. So the final view for a group which has the permission to access different shears is a shear, and similarly the reader who is present in different groups has his final view, which is the union of the views for the groups he is a member of, is also a shear.

The following sections show how the Iceberg model is used to implement the topic hierarchy, the user group hierarchy and the topics individual users are allowed to view. From a given node, all the nodes all the way to the root are visible in the Iceberg visibility model. Hence the model is suitable to organize topics from general to specific, from public to private and user groups from less trusted groups to most trusted ones. In the user group the more trusted users should be in the leave nodes and the most private posts should be at leave nodes of the topic hierarchy. With the Iceberg model, the same content can be adapted to the needs of specific groups including disabled users especially since the user and the topics are defined by the Web content provider. This can be used deny access to some information to some users: When a reader is granted access to a topic node, he/she has access to all the nodes on the path to the root but not to the descendent topics. The same user group can be allowed to view several topic nodes and the same user can be assigned to several user groups. So the topics a user is allowed to see have to be computed for each user as a sub-graph that contains all the paths to the root from all the nodes the user has been granted permission to view. User groups and access privilege to topics nodes have to be granted accordingly in order to maintain trust and privacy.

3.2 Topics, Reader Groups and Views

The class as defined in the Iceberg model is used to implement the user group hierarchy and the topic hierarchy while the shear is used to dynamically implement the view a reader has of a PCB depending on the groups he/she belongs to. Because a view is defined as a shear, the view a reader has is also a shear defined as the union of the views associated to the different groups the reader belongs to.

3.3 Managing Trustiness

Each peer is entirely in charge of assigning trustiness rate to the others.

The more a person feels comfortable with a peer the more the person will reveal himself/herself. The level of confidence or trust is expressed through the user hierarchy: The less trusted users are at the root of the user hierarchy and most trusted at the leaf nodes. Our model of safety for each PKB is inspired by the model of Biba [CFMS95] with the "discretionary policy" of type "hierarchy of objects" in which the visibility of an object means the visibility of all its ancestors. The aim is to transform the definition of the list of the visible objects into a choice of cut (i.e. a choice of level of visibility for each branch of the hierarchy). This level is adjustable dynamically according to the confidence given to the reader.

In practice, the owner of a PCB creates in advance as many user groups as he/she intends to have and for each user group, defines the view on the content. Only one view is allowed for a group of readers but a reader can belong to many groups. When a new user requests to be part of the readers the owner has to figure out the groups the new reader should belong to. When a user belongs to many groups, the view he/she has is the combination of all the views of the groups he/she belongs to. As shown in Figure 3, a publisher creates his/her own topics and user groups in his/her PCB. He/she creates on his/her PCB the views for the user groups. Then he/she registers readers as they come and assigns them to various groups (Figure 4). The final view of the reader will be the union of the views of all the groups he/she is assigned to.

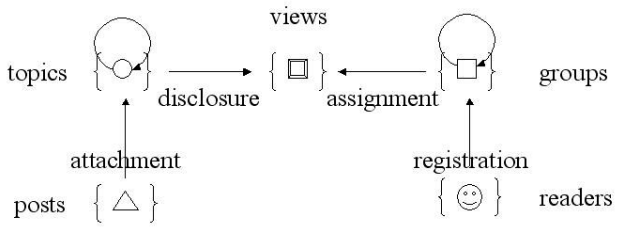

Figure 4. Topic, Views and User Assignment to Reader Groups

4. THE WEBOGRAPH ON A TYPICAL TEST SCENARIO

In the publishing application we have *topics* that contain write-ups (e.g. *posts* the content units of blogs) and *users* that access the topics.

4.1 Hierarchy of topics

Publisher *Saglio* has the hierarchy of topics in Figure 5. The topic hierarchy for a publisher is rooted. (Figure 5). The topic *Top* is a default topic created which will be used to create the default view, and this will be explained shortly in the Views section. The number that is present in the brackets beside each topic is the total number of posts that are accessible from this topic up till the root.

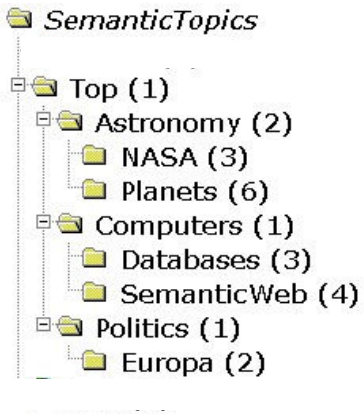

Figure 5. A sample hierarchy of topics of a publisher

4.2 Hierarchy of Reader Groups

Publisher *Saglio* has the hierarchy of reader groups in Figure 6.

Figure 6. A sample hierarchy of groups of a publisher

The user group Hierarchy is also rooted as shown in Figure 6. When a publisher registers, his/her default reader group is *Everyone*. A default reader *guest* is registered and the group *Everyone* is assigned to the reader *guest*. Every new reader created is by default assigned the group *Everyone*. As the name suggests, every reader of the publisher is a member of the group *Everyone*. The number that is present in the brackets beside every group is the number of readers present in that particular group. In the above example, there are six readers, which are number beside *Everyone*, and there are three readers in the group *Astro*, among them a single in subgroup *Amateurs*, where as there is one reader in the group *Citizens,* etc... Note that a reader in a particular group will be a member of all the ascendant groups of that group and that a reader may belong to several disjoint groups.

4.3 Creation of a view

The publisher *Saglio* has the views in Figure 7.

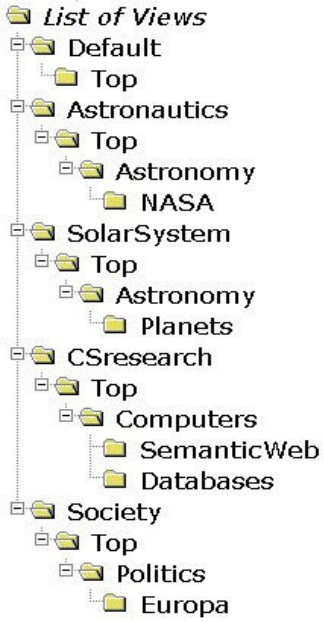

Figure 7. A sample of the views of a publisher

The publisher *Saglio* in this example, has defined four views. The first view *Default* is created by default when the publisher is registered and consists of the default created topic *Default*. Note that all these views are defined using the topics hierarchy of the publisher (Figure 7). The publisher can edit his/her views whenever needed.

4.4 Assigning views to groups

When a publisher has views and user groups defined, he/she can give the groups the permission to access the views he/she wants them to access. Remember that by definition when a group of user is given permission to access a view, all the descendants of that particular group (sub-groups) have also the permission to access the particular view.

In our example, group *Astro* is given view *Astronautics*, subgroup *Amateurs* is given also view *SolarSystem*, group *Citizens* received view *Society* and group *ENST* the view *CSresearch*. The final view for a group is the union of all the views it can access (Figure 8).

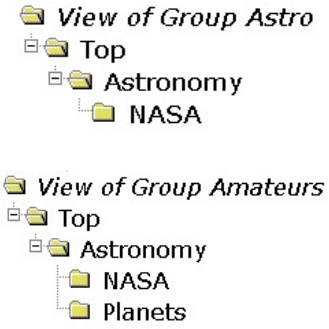

Figure 8. Samples of the final views for groups

Note the view dynamic composition for group *Amateurs*.

4.5 Assigning readers to groups

A reader can be assigned to several groups. If a reader is assigned to a group, then he/she will be automatically assigned to all the ancestral groups of the present group.

For example reader *Oria* as been assigned to both groups *Citizens* and *ENST* and reader *Hebrail* to both groups *ENST* and *Astro*.

The final view for a reader is the union of all the views for the groups it belongs to.

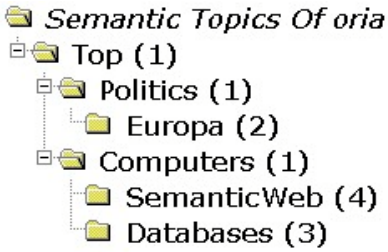

Figure 9. A sample of the final view of reader Oria

The reader *Oria* belongs to the groups *Citizens* and *ENST*. His resulting final view is a union of views for the two groups (Figure 9).

The reader can subscribe to the RSS feeds of various topics in his view. The RSS file is dynamically generated for each reader and because the permalink is also dynamically generated for each reader, privacy is maintained. A reader can neither subscribe to a RSS feed, nor access the post of a topic that doesn't exist in his/her view.

5. ICEBERG IMPLEMENTATIONS AND PROTOTYPES

The "iceberg trust model" presented in this paper has been progressively specified along with prototype implementation since year 2005:

- A first prototype in "LAMP technology" (Linux system, Apache HTTP server, MySQL and PHP) - technology which yields light and easy to install software - was delivered [http://www.infres.enst.fr/webograph] at the beginning of year 2006 for further research. Even if it allows very suggestive demonstrations of scenarii, like the one described in the former paragraph, it still contains a few bugs, partly due to XUL library limitations.
- A second prototype, also in LAMP technology, but using, instead of XUL, AJAX for user interface development and run times, was developed in years 2006-2007 [http://webographe.projet.free.fr/]

thanks to a join effort of the French “Groupe des Ecoles des Télécommunications”. Some bugs were fixed and and some limitations of the previous prototype encompassed. But the propotype was in remains in “French only” for local promotional reasons. It allowed "real life" social networking experiments.

These prototypes were called “webographs” – respectively v1.1 and v1.2 – to focus on the family of tools used by people – “webographists” i.e. web librarians – annotating the web for their personal/professional needs and exchanges.

As early as in 2003, a few researchers [PAB+03] working on "semantic web" and "community of practice" issues designed a so called "web-of-people". It was, to our knowledge, one of the earlier intuitions of what is nowadays known as Web2.0. It laid the basis of a "state of understanding" between researchers about what could be a tool kit for social network practices like sharing bookmarks, annotated excerpts or photos, shorts articles, etc.. stored in personal content bases of each peer in a P2P network.

During following years they implemented various prototypes to experiment communication protocols that are more appropriate for content sharing via web resource referencing. They have been closely concerned with a dynamic disclosure policy based on views management in association with diffusion lists management. Several theoretical issues dealing with *peer to peer querying* and *transitivity of grants* have been set and discussed [TSP 04] [SST 05]. But first prototypes developed then remain only usable for research experiments.

6. Conclusion

The work presented in this paper can be categorized into personalization and more specifically into sociological filtering. Personalization is often used for Web systems that filter the content to be presented based on some knowledge on the user. Sociological filtering is defined in [MGT+87] as filtering that works by supporting the personal and organizational interrelationships of individuals in a community. Sociological systems integrate user models with the knowledge needed for filtering the information. Sociological filtering can further be divided into properties-based and collaborative-based filtering systems. Properties-based filtering uses properties defined on the user (e.g. age, area of interest) while collaborative-based filtering makes decisions on what to be presented to a user based on the recommendations from “similar” users. Recommender systems are representatives of sociological filtering systems. A more complete survey on information filtering can be found in [HSS 01].

Recommender systems suggest documents and services that users might be interested in viewing and have been shown to very effectively guide users in selecting appropriate documents for their tasks. By recommending useful next steps for an information search, recommendation systems have a huge potential for reducing information overload and it is expected that recommendation systems will be an integral part of the future applications [Wil 02]. The two most common technologies for recommender systems are collaborative filtering (CF) and content-based filtering (CB). The former generates recommendations based on users’ evaluations and preferences while the latter provides recommendations based on the similarities of document content [LI 02].

In personalization filtering and recommender systems, the assumption is again that all in the Web is public and the aim is to reduce the amount of information to be presented to the user. In the case of this paper, the publisher is the one filtering what the readers can see.. Filtering what readers are allowed to view is not a novel concept either It is known as defense applications where resources are only visible to people having the appropriate security clearance (login and password and privileges). This is the classical security system implemented by companies and administrations to secure their databases. The Iceberg model proposed here differs from defensive applications in that it aims to filter content according to type rather than according to content. Another major difference is that the Iceberg models allows everybody to have access to some kind of information whereas in defense applications only named personnel designated as needing access are allowed to view the database content. The Iceberg model allows filtering users according to different levels of trust. It will involve the need to set up different "ontologies" of user types as well.

One does not have to pick between the 2 approaches: filtering the content of the Web for a reader depending on the reader's wishes and filtering what the reader should see based on the publisher's whishes. Both approaches are needed and are complementary although more attention had been paid to the first approach mainly because of the "all or nothing" publishing model of the Web.

With both v1.1 and v1.2 webograph prototypes the reader can subscribe to the posts of just one topic using one RSS feed. In future we would like to allow the reader to subscribe to the posts of one or more topics using a single RSS feed by passing a few filters to the RSS feeds.

The other features we would like to implement in the future versions will help:

- Allow the publisher to merge two or more topics into a single topic while creating an external view.
- Allow the publisher to divide a topic into two or more topics, where the division is made depending either on the content or attributes of the posts in the topic. The publisher can create filters on the content or attributes of the posts.

7. References

[Berg01] M. K. Bergman "The Deep Web: Surfacing Hidden Value". The Journal of Electronic Publishing **7** (1), August 2001.

[BMO+89] R. Bretl, D. Maier, A. Otis, J. Penney, B. Schuchardt, J. Stein, E. H. Williams and M. Williams, The GemStone data management system," In Object-Oriented Concepts; Databases Appl., Eds. W. Kim and F. H. Lochovsky (Morgan Kaufmann, 1989).

[CFMS95] S. Castano, M. Fugini, G. Martella, P. Samarati : Database Security. ACM Press, Addison-Wesley, 1995.

[HHL 04] Tony Hammond, Timo Hannay, and Ben Lund : The Role of RSS in Science Publishing, Syndication and Annotation on the Web. In D-Lib Magazine December 2004, Volume 10 Number 12, ISSN 1082-9873, http://www.dlib.org/

[HSS 01] U. Hanani, B. Shapira, P. Shoval: Information Filtering: Overview of Issues, Research and Systems. User Model. User-Adapt. Interact. 11(3): 203-259 (2001)

[LI 02] Z. Li and I. Im: "Recommender Systems: A Framework and Research Issues," Americas Conference on Information Systems (AMCIS), Dallas TX, 2002.

[LLOW91] C. Lamb, G. Landis, J. Orenstein, and D. Weinreb, The objectstore database system," *Commun. ACM* 34(10), 19-20 (1991)

[LOS 98] Y. Leontiev, M. T. Ozsu, and D. Szafron. On separation between interface, implementation and representation in object DBMSs. In Proceedings of Technology of Object-Oriented Languages and Systems 26th International Conference (TOOLS USA98), pages 155—167, Santa Barbara, August 1998

[LRFV92] C. Lcluse, P. Richard, and Fernando Vlez, Building an Object-Oriented Database System," The Story of O2; an Object-Oriented Data Model (Morgan Kaufmann, 1992)chap. 2, pp. 77-97.

[MGT+87] T. Malone, K. Grant, F. Turbak, S. Brobs and M. Cohen: Intelligent information sharing systems. Communication of ACM 43(8), 35-39

[PAB+03] M. Plu, L., Agosto P., Bellec, W. Van De Velde: The Web of People: a dual view on the WWW. In Proc. of the 12th International World Wide Web Conference, WWW'03 Budapest, 2003.

[SST 05] Saglio, J.M., Scholl, M.., Ta, T.A.: Efficient Query Processing in P2P Networks of Taxonomy-based Sources. In Workshop Data Integration and the Semantic Web, CAiSE'05 Porto, 2005.

[TSP 04] Ta, T.A., Saglio, J.M., Plu, M.: An architecture based on semantic weblogs for exploring the Web of People. In Workshop Application of Semantic Web Technologies to Web Communities, ECAI'04 Valencia, 2004.

[Wil 02] J. F. Williams: Hot technologies with a purpose," *Library Journal*, 2002, page 51, 2002.