Profile-based retrieval on the World Wide Web

B. van Gils, H.A. Proper, P. van Bommel E.D. Schabell
basvg@acm.org, e.proper@acm.org, pvb@cs.kun.nl, erics@cs.kun.nl

Abstract

In this article we present a novel architecture for Information Retrieval on the Web called Vimes. This architecture is based on a broader definition of *relevance*. This broader definition lies in the fact that there is more then just topical relevance. Documents (or: resources) must also confirm to other constraints with regard to form, format and things like price and quality.

1. Introduction

In today's ``information society'' information plays an increasingly important role. The trick is to get the right information at the right time and in an appropriate format for a given goal. Finding the right information has been researched extensively in the IR-field. As recently as the 1970's people tried to devise computer programs to assist them in their search for information. These computerized searches started with searching in homogeneous document collections such as in the STAIRS-project [21]. The search process became more elaborate with the growing use of the Web. It led to the introduction of search engines such as GOOGLE which not only indexes (hyper)text, but also images, PDF-documents and interactive databases such as Citeseer [8]. In other words, search engines attempt to retrieve relevant *resources*, rather than documents alone.

The importance of the timing aspect is particularly obvious when investment decisions are involved, such as on the stock market. Getting some information late could have huge (financial) consequences. Implementing a strategy for getting information in time often depends on many things such as choosing the right partner/supplier: some news sites are `faster' then others in picking up news.

The third aspect mentioned deals with formats in the broad sense. It refers to ``file format'' (e.g. PDF, or HTML) as well as ``structural format'' (e.g. ``abstract'', or ``photograph''). The file format issue has been around since the early days of computing. Since people use different tools for jobs such as text processing a need for conversion tools between the file formats arose. Many of these conversions are available today. This is not (yet) the case for the latter issue, even though attempts have been made. A good example of this type of software is a computer program that generates abstracts for expository text (see e.g. [2]).

It is apparent that these factors vary for different users of IR-systems. For some users it is ok if certain financial records arrive slightly late, whereas for others it might have unpleasant consequences, some people would prefer an abstract of a (large) report over its full text, etc. In other words, each of these factors can be seen as a *characteristic* of a searcher. Loosely defined, a *profile* is the collection of all characteristics of a searcher that are relevant for the retrieval process.

The goal of this paper is to present a broader definition of what *relevance* is and to show how this can be used in IR. This broader notion of relevance is based on the mentioned issues and will be presented in Section 4. To this end, we briefly present a model for information supply in Section 3. This model (based on [14, 15]) allows us to introduce transformatons which are essential to the introduction of our prototype retrieval architecture in Section 5. Section 2 introduces the profiles which are used in our architecture (called Vimes).

2. Profiles

Already in [19] it was determined that information retrieval systems can be personalized for users by means of profiles. For years a lot of research has been invested in the area of user profiles. Often, these profiles are used to enhance the query by capturing the user's notions of query terms (see e.g. [7, 19, 20]). However, profiles can be used more extensively. For example, in [16] profiles are used for access control. We define that:

Profile

A (user) profile consists of a set of preferences with regard to behavior of a search engine as well constraints on the results it presents to the user.

To illustrate this definition, the following list are the items that make up a particular user-profile:

preferences

I prefer a maximum of 25 results per page, and by selecting a relevant resource (clicking on the link) will open a new window.

constraints

I prefer HTML and PDF formats and refuse the Microsoft DOC-format. Furthermore, the size of the resource should not exceed 25Mb.

Using this definition, there are two areas in the retrieval process where profiles can be used. Firstly, they can be used for *post-processing* the results of the ranking process. For example, an resource that was found to be topically relevant can be converted to the proper format (See Section 4). Furthermore, profiles can be used to make sure that the retrieval engine operates according to the user's wishes.

In the previous section we explained what profiles are and what they can be used for. In this section we present a *possible* format for storing these profiles, whereas in the next section we explain how/where they are stored exactly.

Since we want the profiles to be re-used across (Web) search engines, the format should be an *open standard*. More specifically, we want our format to be machine understandable and interoperable. The eXtensible Markup Language (XML, see e.g. [5]) is particularly well suited for this task (see e.g. [24]). The following XML-fragment is an example of what a profile could look like:

```
<? xml version="1.0" ?>
<!------------------------------------------------------------- -->
<!-- A profile has an owner, identified by his/herEmail-address. -->
<!-- Furthermore, a check-sum is included for security purposes. -->
<!-- This profile stores 3 characteristics.                      -->
<!------------------------------------------------------------- -->

<!-- define the owner of the profile -->
<profile owner="Basvan Gils" email="bas.vangils@cs.kun.nl"cs="2768A493">

    <!-- 1stcharacteristic: how many results per page? -->
    <characteristic type="results">
        <page> 25 </page>
    </characteristic>

    <!-- 2nd charactersitic: the max. size in Mb -->
    <characteristic type="max_size">
        <mb> 5 </mb>
    </characteristic>

    <!-- 3rd characteristic: preferred file-types -->
    <characteristic type="file_type">
        <type nr="1"> HTML </type>
        <type nr="2"> PDF </type>
        <type nr="3"> PS </type>
    </characteristic>
</profile>
```

Note that this excerpt is intended to illustrate our ideas. Defining a formal DTD for profiles is part of future research.

3. The model

Our model of information supply is based on the distinction between data and information. The entities found on the Web, which can be identified by means of a URI [3], are *data resources*. These data resources *can* be information, if and only if they are relevant with regard to a given information need. Also, we presume that many data resources can, at least partially, convey the same information. Hence, we define *information resources* to be the abstract entities that make up information supply. Each information resource has at least one data resource associated to it. Consider for example the situation in which we have two data resources: the painting Mona Lisa, and a very detailed description of this painting. Both adhere to the same information resource in the sense that a person seeking for information on `the Mona Lisa' will consider both to be relevant.

In a way, the data resources *implement* the information resources; a notion similar to that in [12] where `facts' in the document subspace are considered to be `proof' for hypotheses in the knowledge subspace. Note that each data resource may inplement the information resource in a different way. We define a *representation type* to indicate exactly how a data resource implements the information resource it is asociated to. Examples of representations are full-content, abstract, keyword-list, extract, audio-only, etc.

Many different types of data resources can be distinguised on the Web today, such as documents in different formats (HTML, PDF, etc), databases, interactive Web-services, etc. Hence, each data resource has a *data resource type* Furthermore, data resources may have several attributes such as a price or a measurement for its quality. Such attributes can be defined in terms of an *attribute type* and the actual value that a data resource has for this given attribute type.

Also, *values* can be attributed to data resources. For example, the value ``640x480'' can be used to denote the resolution of an image, or €20 the price of a data resource. We model this by defining that combinations of data resources and values associated to these data resources have an attribute type.

Last but not least, data resources can be interrelated. The most prominent example of this interrelatedness on the Web is the notion of *hyperlinks* [6, 9], but other types of relations between data resources exist as well. Examples are: an image may be part of a webpage, a scientific article may refer to other articles, etc.

Figure 1 shows the General Model for Information Supply, which is based on the following verbalisation:

- Information Resources have at least one Data Resource associated to them;
- A Representation denotes the unique combination of an Information Resource and a Data Resource
- Representations have at least one Representation Type
- Data Resources have at least one Data Resource Type
- Data Resources are related via Relations with a source and a destination.
- Relations have at least one Relation Type.
- Data Resources may have attributed values
- Attributes have at least one Attribute Type

The fact we have several *types* in our model indicated heterogeneity. The fact that many different data resources types exist refers to the file-format discussion from Section 1, where as the heterogeneity refers refers to the structural format. The following section explains how these affect the definition of relevance.

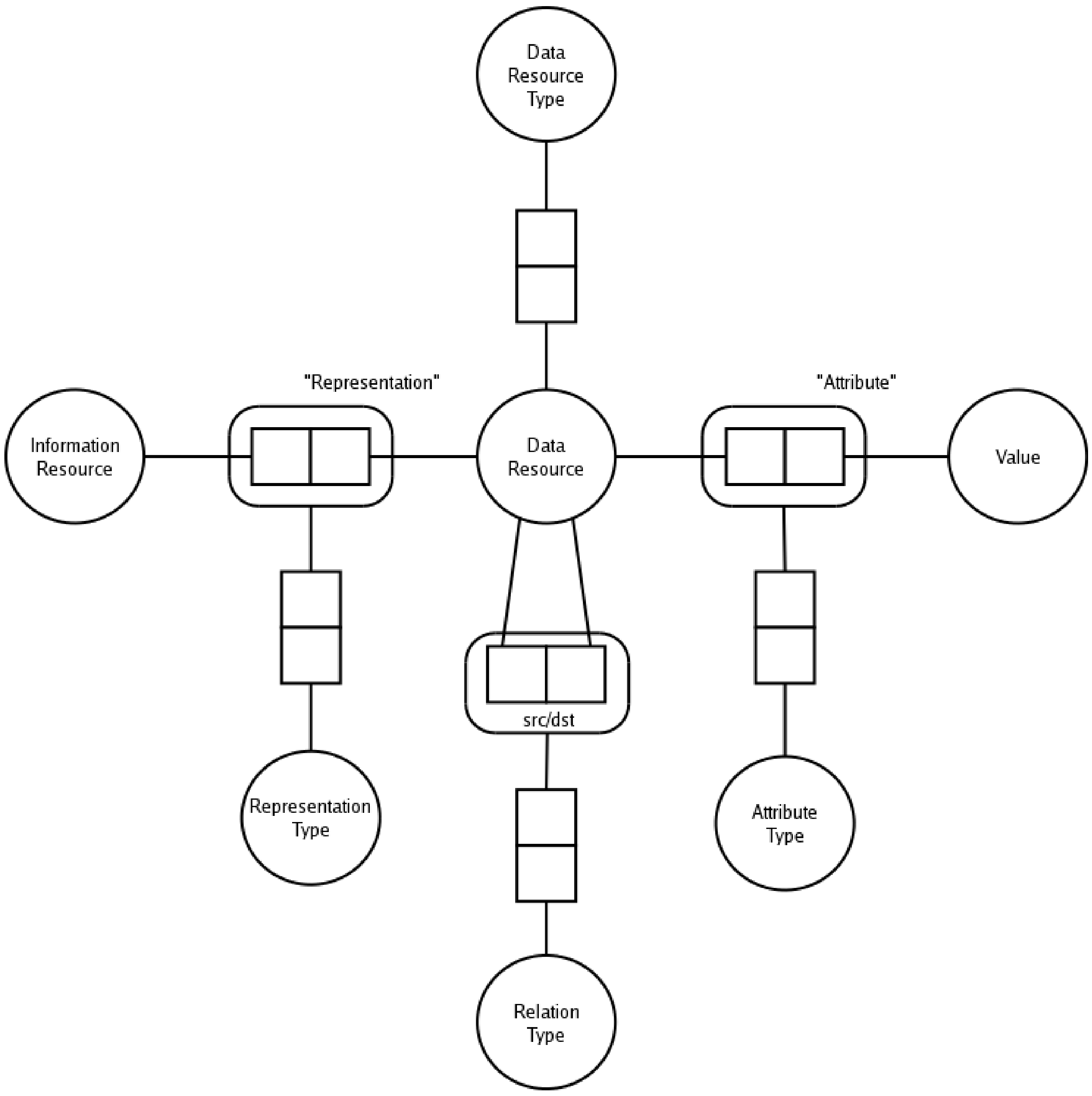

Figure 1. A general model for information supply

4. Relevance

One of the basic functions of any information retrieval (IR) system is *relevance ranking*: the (characterizations of) resources are ranked such that the resources that are ``most relevant" are listed first, and the ones that are least relevant are listed last. In [11] an overview is given of metrics that are used to determine the relevancy of a Web-document with regard to a query. Furthermore, it is pointed out that relevancy involves more than *topical relevance*; other attributes of resources (such as its quality and price) are important as well.

Apart from topical relevance, which is the `traditional' way of measuring relevance, we define that other constraints must be met as well. Examples of such constraints are its format (as explained in the previous section), but also price, quality, etc. It may very well be that a searcher is willing to pay a certain amount of money in order to get his hands on a high-quality resource! Hence, we define relevance as follows:

Relevance

Resources are relevant with regard to a query if and only if this resource meets all the criteria that a searcher poses on it. These criteria can be formulated in either the query, or the user-profile.

This definition resembles the notion of functional versus non-functional requirements in Software Engineering [23]. It is now well accepted that non functional requirements and functional requirements are equally important to any software engineering project (see e.g. [1, 10] for a discussion on the importance of non functional requirements).

This modified view of relevance has an impact on *precision* and *recall*, for it is `less easy' for a document to be relevant with regard to a query. For example, it may be that a resource must be converted to another format before it is really relevant. In Section 5 we explain how a retrieval system can exploit this new notion of relevance in order to achieve `better retrieval'.

5. Architecture

In the previous sections we explained our notion of formats, profiles and relevance. These notions are essential for the architecture of Vimes, which we will introduce here. The architecture uses many elements that stem from previous research such as brokers, agents, semantic web components and web services.

5.1 Components

The first component is the *profile repository*. This repository stores the characteristics of users so that they are available at all times. This implies that they can/should be used for all the queries, regardless of the search engine that is used. In order to achieve this we make use of an open standard (XML) as outlined briefly in the previous section. An additional advantage of using such an open standard lies in the fact that it will make life easier for developers in the sense that they can more easilly integrate repositories. In the end, users should benefit from this all: they only need to specify their preferences and constraints once in a single profile (over which they have full control) and all search engines that are able to make use of profiles can re-use that single profile[1].

The second component in the Vimes architecture is the *transformation broker*. The basic functionality of this broker is simple: try to transform a given resource into a different form or format. To this end, the broker must have access to a number of transformations, and must be smart enough to be able to "ccompose" other transformations from them. For example, if there is no 1-step fransformation available from the LaTeX-format to the DOC-format, it may be possible to *compose* this transformation from two other transformations such as latex2rtf and rtf2doc.

With this transformation broker we hope to cater for a broader notion of relevance, as explained in the previous section. The broker will be a networked service, encapsulating functionality of all available transformation tools on the network and provide for multiple methods of transport. For example, a request to the transformation broker includes the form desired is PDF and the resource document is a postscript document. This particular conversion can be achieved by a transformation broker on the network that provides the tool *ps2pdf*. For similar reasons as before, we choose open standards for transport such as ftp and http. An additional benefit is that other parties can more easily participate and contribute by submitting transformation routines to the broker.

The broker component will be the main interface for users seeking information. It will interact with the user-profile repositories and search engines on the Web. Essential to our architecture is the broker's ability to interact with not only the well known web search engines (Yahoo, Google, AltaVista, Excite, etc.), but also with such enabling technologies as static agents, mobile agents, web services and services using the Semantic Web or Resource Description Framework (see e.g. [4, 13, 17, 18]). Our broker component will also provide interaction with different forms of user-profile repositories, both local and remote. This will allow interaction with other profile systems on the Web (see e.g. [20] for an agent-based approach along these lines). This leads to the following architectural diagram, with the components in the shaded area making up the system.

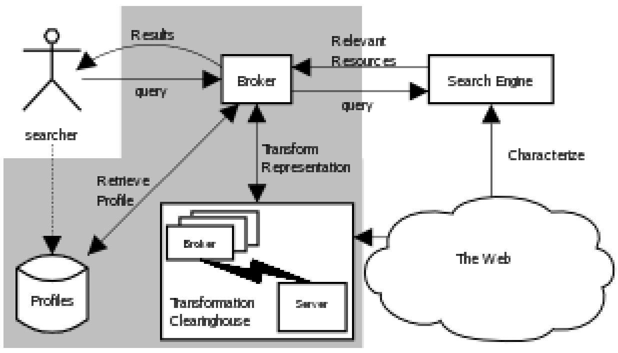

Figure 2. A general model for information supply

Please note that the components will be loosely coupled so that they (especially the profile repository and the transformation broker) can also be accessed by other systems via the Web.

To summarize: the Vimes architecture provides a single interface for users who want to search the Web. It makes use of loosely-coupled components which are available as services over the Web. The main innovation of Vimes is that it uses a broader notion of relevance in the sense that it is capable of doing "more" than just topical relevance.

5.2 Example session

In this section we describe what the retrieval process could look like, based on the architecture as defined in the previous section. The first thing to be done is that the user creates a profile, preferably via an intuitive Web-interface, after which it can securely be stored in the repository. The second step is to browse to the broker, which functions as the main interface for the rest of the process. The user identifies himself (either automatically via e.g. a cookie, or more explicitly via a login-screen) after which the relevant profile is retrieved from the repository.

When the profile is retrieved, the user can enter his query into the system. Two things can happen at this point: either the broker decides to reformulate the query based on the user-profile, or it leaves the query untouched. Subsequently, the query is submitted to one of the search engines. This can be one of the well known web search engines, but others are possible such as an agent, a web service or other external services as described above. Based on the user's profile, the broker may decide to post-process discovered resources. The returned list of discovered resources would then be transformed using the transformation broker. If this is indeed the case, the resources are processed and ranked again before they are presented to the user.

6. Conclusion

In this position paper we introduced a novel retrieval architecture called which is based on a broad notion of relevance and profiles as a means to store user preferences that are (semi) constant.

This broader notion of relevance is derived from a model for information supply (Section 3). The foundation for both this model lies in the *heterogeneity* of information supply: there are many different kinds of resources, several resources may (partially) convey the same information, may have attributes such as price, quality, etc.

The traditional notion of relevance, which ``only'' considers topical relevance, can then be extended such that a resource is relevant with regard to a query if and only if all constraints that were posed on it by a searcher are met. These constraints may include things like price and quality, but also structural form and format.

Vimes is intended to be a broker that assists users in querying the Web. There are three important components in this architecture. The *profile repository* stores the profiles of all users in an open format such as XML. There are still some open issues in this area, such as specifying a language for storing the profiles, enforcing that they are stored securely, etc.

The second component is the *transformation broker*, which enables us to perform transformations on resources found on the Web. With these transformations we hope to be able to transform resources into formats that are both convenient and desired by individual users. For example, we can transform a HTML document into PDF, or generate an abstract of a report that is too long according to a user's profile. We are currently working on a system that performs these transformations by setting up a Conversion Clearinghouse that is web accessible, allowing users to search through our available conversions.

Last but not least, the *broker* in the architecture is the user-interface. It interacts with the two other components, as well as with search engines on the Web. Much work remains to be done in this area. For example, we need to figure out what the interface will look like, which message-standards are going to be used to interface with the other components, etc.

This article intends to provide insight into our novel way of thinking about retrieval. It outlines our proposed architecture without giving a full specification. Finally, we have presented a road-map for our research.

References

1. Barrett, M. L. (2002). Putting non-functional requirements to good use. *The Journal of Computing in Small Colleges*, 18(2):271-277.
2. Barzilay, R. and Elhadad, M. (1997). Using lexical chains for text summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS'97)*, Madrid, Spain.
3. Berners-Lee, T. (1994). Universal Resource Identifiers in WWW. Technical Report RFC1630, IETF Network Working Group,`http://www.ietf.org/rfc/rfc1630.txt`. last checked: 13-aug-2002.
4. Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities.*Scientific American*. 284(5):34-43
5. Bray, T., Paoli, J., Sperberg-McQueen, C. M., and Maler, E. (2000). Extensible markup language (XML) 1.0 (second edition). Technical report, World Wide Web Consortium,`http://www.w3.org/TR/REC-xml`. last checked: 19-may-2003.
6. Bush, V. (1945). As We May Think. *The Atlantic Monthly, 176(1):101-108.*
7. Chen, P.-M. and Kuo, F.-C. (2000). An information retrieval system based on a user-profile. *The Journal of Systems and Software*, 54(1):3-8.
8. Citeseer (1997). *NEC ResearchIndex Citeseer*. `http://citeseer.nj.nec.com`. Last checked: 19-may-2003.
9. Conklin, J. (1987). Hypertext: An Introduction and Survey. *IEEE Computer*, 20(9):17-41.
10. Cysneiros, L. M. and do Prado Leite, J. C. S. (2002). Non-functional requirements: from elicitation to modelling languages. In *Proceedings of the 24th international conference on Software engineering*, pages 699-700, Orlando, Florida. ACM Press. ISBN: 1-58113-472-X.
11. Dhyani, D., Ng, W. K., and Bhowmick, S. S. (2002). A survey of web metrics. *ACM Computing Surveys (CSUR)*, 34(4):469-503. ISSN:0460-0300.
12. Feng, Hoppenbrouwers, J. (2001). Towards knowledge-based digital libraries. *SIGMOD Record 30*, 1:41-46.
13. Fünfrocken, S. and Mattern, F. (1999). Mobile agents as an architectural concept for internet-based distributed applications - the wasp project approach. In Steinmetz, editor, *Proceedings of the KiVS'99 ("Kommunikation in Verteilten Systemen")*, pages 32-43. Springer-Verlag.
14. Gils, B. v., Proper, H. A., and Bommel, P. v. (2003a). A conceptual model of information supply. *(submitted to) International Journal: Universal Access in the Information Society.*
15. Gils, B. v., Proper, H. A., and Bommel, P. v. (2003b). Towards a general theory for information supply. In Stephanidis, C., editor, *Proceedings of the 10th International Conference on Human-Compu ter Interaction*, pages 720-724, Crete, Greece.
16. Gligor, V. (1996). Characteristics of role-based access control. In *Proceedings of the first ACM Workshop on Role-based access control*, Gaithersburg, Maryland, United States. ACM Press. ISBN: 0-89791-759-6.
17. Google (2003). *Google Web API's*. Google, `http://www.google.com/apis`. last checked: 16-May-2003.
18. Miller, E., Swick, R., and Brickley, D. (2003). *Resource Description Framework (RDF)*. World Wide Web Consortium, `http://www.w3.org/rdf`. last checked: 16-May-2003.
19. Myaeng, S. H. and Korfhage, R. R. (1986). Towards an intelligent and personalized retrieval system. In *Proceedings of the ACM SIGART international symposium on Methodologies for intelligent systems*, pages 121-129, Knoxville, Tennessee, United States. ACM Press. ISBN:0-89791-206-3.
20. Pierra, S., Kacan, C., and Probst, W. (2000). An agent-based approach for integrating user profiles into a knowledge management process. *Knowledge-Based Systems*, 13(5):307 - 314.

21. Salton, G. and McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill New York, NY.
22. Schabell, E. D. (2002). Resource access in generic information retrieval systems. Master's thesis, Vrije Universiteit, Amsterdam, Netherlands.
23. Sommerville, I. (1989). *Software Engineering.* Addison-Wesley, Reading, Massachusetts.
24. Suryanarayana, L. and Hjelm, J. (2002). Profiles for the situated web. In *Proceedings of the eleventh international conference on the World Wide Web*, pages 200-209, New York, NY, USA. ACM Press. ISBN:1-58113-449-5.

Footnotes

... profile[1]
: Open issues that we need to work out still are the management of these profiles: where to store them? how to achieve an acceptable level of security?

... broker[2]
: This transformation broker flowed out of the earlier work done on resource access for generic information retrieval [22].