Directory Management for Knowledge Sharing in Enterprises

● Takanori Ugai ● Kazuo Misue ● Kunio Matsui

(Manuscript received June 17, 2003)

Web servers are becoming very common in large organizations. While they seem to promote and support knowledge sharing, the lack of control is causing many problems, for example, the decentralization of information and redundancy of categories and contents. We are developing a system to support directory maintenance. In this paper, we describe a unified framework for document directory maintenance and a system we are developing that decreases directory maintenance work for editors.

1. Introduction

Ten years or more have passed since the appearance of the WWW (World Wide Web). Now, every organization has a Web server in each of its sections and each organization sends information using their own original format. To manage large amounts of electronic documents, software must evolve so that it becomes easy to operate Web servers and hardware such as large-volume disks and high-performance CPUs. Knowledge management and knowledge sharing in organizations must also evolve.

While knowledge sharing seems to be promoted and supported in organizations, the lack of a well-controlled centralized organizer causes many problems, for example, information is decentralized and there are redundant categories and contents. Web directories such as Yahoo[1)] and Web search engines such as Google[2)] are popular systems that help people find Web pages. The Web directory would be a better solution to the above problems, and it performs a key role in knowledge sharing for two reasons:

1) Information is arranged according to a consistent category system.
2) The related information is collected in a category or near category in the directory tree.

In other words, editors of directories must perform the following tasks in order to manage the information according to an organized hierarchical category and keep the quality of directories high:

1) Maintain the category structure: The tree structure is under the control of a unified point of view, a name is applied to each category, and the number of contents in each category is kept at the appropriate level.
2) Maintain the registered information: URLs (Uniform Resource Locators) that are profitable and useful for users are chosen from the Internet and intranets, they are allocated in suitable categories, and the information is kept up to date.

In addition to the above two tasks, the editors must also perform the following for a Web-based directory.

3) Maintain HTML (Hypertext Markup Language) files: One category is described with one consistent set of syntactically valid HTML files.

These tasks must be performed continuous-

ly so that the Web directory functions effectively for knowledge sharing. However, the editors of many directories allow their directories to contain obsolete information, because it takes them a lot of time to fix dead links and update the information when a large amount of it has been accumulated. In addition, many editors do not sufficiently understand the users' purpose of use, and the information in the directory becomes different from what the user requires.

As mentioned above, it takes a lot of time to construct, maintain, and manage a large directory. Yahoo! employs many specialists to maintain its directory. The Open Directory Project[3] has a large number of volunteers (57 245 as of June 12, 2003) participating as editors to maintain the directory. In an intranet portal, a directory has only one or two editors in most cases. Even when several sections maintain a directory jointly, one editor is provided from each section. Most editors maintain a directory for just a couple of hours a day, because they have other work to do.

The problem to be solved here is how to improve the efficiency of directory maintenance management and also reduce the cost. Spreadsheets are widely used to maintain categories, and URLs and HTML editing programs are used to make HTML files for each category but they are not effective as support tools. The Open Directory Project has developed some tools that extract titles and content and also detect invalid links automatically to reduce the editor's work. However, it is hard to handle the requests for checks, classification, and registration because the number of requests has become large.

There are some potential tools and techniques other than human power that might be useful for directory maintenance. However, it is difficult to combine and tune these alternatives according to our purpose, so a unified environment is required.

We will now describe a unified management framework and a supporting system called FOD (Framework of Directory) that we are developing to support document directory maintenance that overcomes these problems. FOD has the following functions:

1) Functions that enable editing from a Web browser and generation of HTML for a category from data in a DB (database) using templates.
2) Functions for presenting log analysis results.
3) Functions for checking and updating dead links.
4) A function that makes it easy to create original screens by using templates and Web services.

These functions reduce the editor's maintenance work.

In this paper, we describe the functions that FOD provides and the system's elemental technologies. In addition, we describe the effects of introducing FOD to some in-house sites based on interviews with their editors.

The directory management tasks are listed in Section 2. The functions of FOD are described in Section 3, and the effects of using FOD are described in Section 4. We applied FOD to five in-house directories and found that it reduced the maintenance cost by half. The final section presents a conclusion.

2. Tasks of document directory management

After analyzing how the editors worked and interviewing them, we found they did the following:[4]

1) Maintenance of registered information
   - Selection: The editors find useful URLs on the Internet and intranets. The number of useful URLs is much smaller than the total number, so they are hard to find.
   - Classification: The editors allocate URLs in appropriate categories. The directory should be built from a unified point of view, and the allocation strategy should be consistent. It is difficult to keep the allocation strategy consistent.

- Information description: The editors add a title and remarks.
- Update: The editors check whether the URLs have changed and look for dead links. If required, the editors modify information. Usually, a directory has many URLs, so it takes a long time to check all of them.
- Deletion: The editors delete useless URLs.

2) Maintenance of category structure
- Construction: Tree structures should be built from a united viewpoint, and a name is applied to each category. It is not easy to keep a consistent viewpoint.
- Uniting and subdivision: The category is kept to a suitable size.
- Deletion: Unused categories are deleted.

3) Maintenance of HTML files
- Description: Using an editing program, the editors write a set of HTML files that describe each category.
- Consistent link building: The editors make related links between the HTML files that represent the categories. There is no automatic tool for this work.
- Consistent design: The editors create each HTML file using a consistent design and a valid syntax. Authoring tools[5] may give some help in this task.
- Making a "what's new" list and a sitemap: When the editors add a URL, they modify several files and make them consistent.

3. FOD (Framework of Directory): directory management system

Figure 1 shows the system structure of the FOD directory management system. The system stores the category and page data in an RDB (Relational Database) and uses JSP (Java Server Page) as a template engine. FOD provides several functions. In Section 3.1, we describe three kinds of basic functions that support HTML file editing, maintenance of registered information, and maintenance of the category structure. In the following sections we describe two advanced functions: automatic URL recommendation and log analysis.

3.1 Basic functions

FOD provides the following basic functions:

1) Support for editing HTML files
- Form-based editing and editorial help: The Web interface and forms can be used to maintain a category. The editor just fills in the forms, which are defined in advance by the system administrator. The title is automatically taken from the HTML file when a URL is added, the HTML file is analyzed, and the explanation is extracted automatically from the description META-tag. When a URL is added, the system checks and warns if the same URL is registered in other categories to prevent unintentional duplication of registrations. This function prevents spam activity, for example, a user trying to register a URL to multiple categories. Also, the same URLs are registered in multiple categories and the information about them is updated synchronously.
- Automatic HTML generation: Because of the DB and template system, the editor

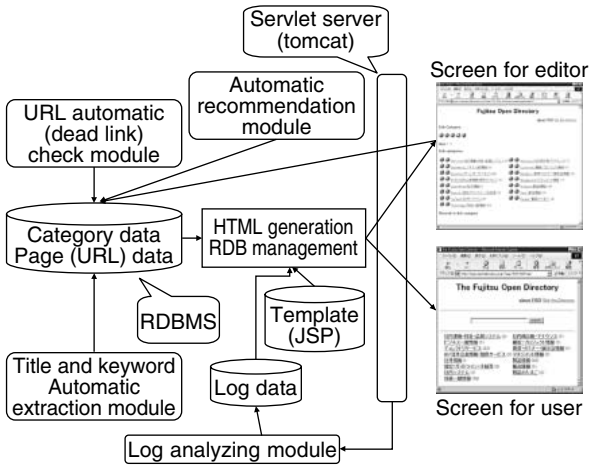

Figure 1
FOD system structure.

does not need to handle HTML files. They are automatically created from the data stored in the database using the template function. Moreover, the editor does not need to transfer HTML files to the server, so special software for transferring HTML files to the server is unnecessary. Because HTML is generated by the system, the screen of each category can be kept to a consistent design. The template can be customized flexibly. The system provides a function for making a what's new list of renewed categories and URLs, and a site map is generated. A Microsoft Explorer style interface and an upper and lower division type interface are provided for the users (**Figures 2** and **3**).

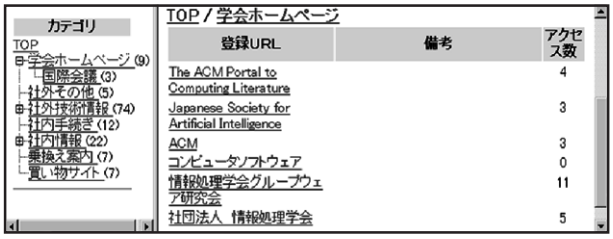

Figure 2
Explorer style interface.

2) Support for maintaining registered information[4)]
   - Selection: This consists of importance degree judgment, entrance judgment, and user demand judgment. Entrance judgment finds the URLs that point to the context of manuals and the guidance pages of homepages. The system accepts requests for URL addition and modification from users other than the editor. Automatic importance degree judgment and entrance judgment are based on link analysis.
   - Classification: Category allocation is based on automatic classification.
   - Appending information: The name of the organization that provides the contents and the indicator that shows whether authentication and authorization are required to access the contents are extracted automatically. Automatic acquisition is based on information extraction techniques. The editor can check and approve the appended information.
   - Update: The URLs in the directory are accessed and checked, and invalid URLs that point, for example, to files that have been

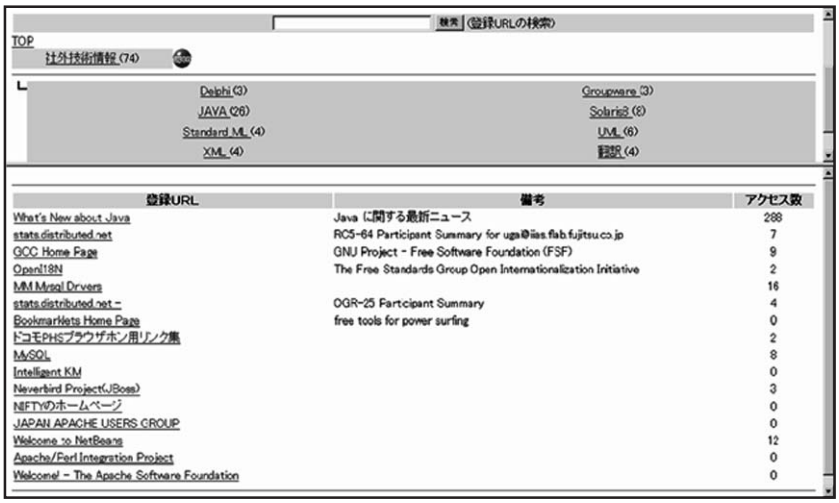

Figure 3
Upper and lower division type interface.

deleted or moved to other servers, are detected regularly and automatically. We have built original heuristic rules for automatic detection.
- Deletion: Deletion candidate judgment is based on log analysis.

3) Support for maintaining the category structure.
- Deletion: Deletion candidate judgment is also based on log analysis.

3.2 Automatic URL recommendation

The automatic URL recommendation function collects useful URLs and allocates them to categories to reduce the maintenance work of the directory's editor.

If the document pointed to by a URL is appropriate for the category, the editor adds supplementary information in remarks and registers the URL to the directory. The editor handles the automatically generated requests in the same way they are handled by the users.

We have implemented two types of systems:

1) URLs are automatically collected by a Web robot, and they are sorted using an algorithm that improves the PageRank of Google. The URLs are allocated to categories using an automatic classification technique according to the similarity of the contents. In this technique, even if a new Web page becomes useful, it is not collected. The URLs are not allocated correctly in the category classified by meta-information, for example, floor design and member lists of the sections.[6),7)] This is because the documents in the category have the name of the organization and members but there are few common words among the documents.
2) The system has another automatic recommendation function based on cooperative filtering[8)-11)] of the bookmarks that the users preserve in the server and the document directory. Compared with the technique described in the above paragraph, this technique can recommend effective URLs to the categories whose contents have few common words, for example, an organization's top page.[7),12)] The categories are hard to recommend using a content-based classification.

3.3 Log analysis

FOD provides three types of log analysis results from the access log and keyword search log:

1) Access frequency by category: This type lists the number of accesses to each category. This type enables the editor to determine which category should be enhanced or abolished.
2) Access frequency by URL and URLs that are not accessed: This type lists the number of accesses to each URL and the URLs that have not been accessed in a particular period, for example, three months. This type enables the editor to determine which kind of URL should be enhanced or abolished.
3) Frequency ranking of retrieval key word: This type shows the key word, the number of categories, and the URLs hit by the word in order of key-word frequency. This type enables editors to read the following information from the ranking and take the appropriate action:
   - For key words with a high use frequency, the corresponding document is expected to be registered. The editor can expand the corresponding category and relocate the corresponding URL to an accessible place.
   - The system provides a list of words that are used frequently by users but are not found in documents in the directory. The list contains the information that the directory's users expect but the directory did not provide. The editor can make the appropriate category, register the URLs to point to the appropriate document, and add an explanation to the URLs.

4. Effectiveness of FOD

We will now describe some examples of FOD's effectiveness. In Section 4.1 we present the sta-

tistics of five in-house directories that are being driven by FOD. In Section 4.2, we describe the effectiveness of the automatic invalid link checker on two of these directories. Then, in Section 4.3, we present some example results of using FOD's automatic URL recommendation function.

4.1 FOD usage

We are using FOD to drive five in-house directories. **Table 1** shows the statistics of the directories. The access numbers in the table show the number of page views per day. FOD reduced the number of editors needed to maintain directory A from four to two. In the case of directory E, which had only one editor, FOD reduced the editing time from about four hours every day to just a few hours a week.

Table 1
Statistics of directories.

ID	Categories	URLs	Users	Accesses
A	325	1488	30000	13000
B	129	953	7500	49000
C	106	399	2000	1000
D	50	227	550	1200
E	9	88	500	400

Table 2
Ratio of invalid links.

Directory	Status	Check result
A	Before FOD	32% URLs are invalid.
	After FOD	0.6% URLs become invalid every week.
C	Before FOD	28% URLs are invalid.
	After FOD	2.5% URLs become invalid every week.

4.2 Checking invalid links

Table 2 shows the ratio of invalid links in directory A and C. The second column shows the status before and after FOD was used. Before FOD was used, the editors of directories A and C left invalid links. When FOD was used, the system checked every link once a week and the editors of the directories checked and fixed the links. It is not realistic to check all of the URLs in directory A by hand. Directory C has 399 URLs, which is relatively smaller than the number in directory A. The pre-FOD ratio of invalid links in directory C was 28%, which means 112 URLs. The editors left them invalid. This means that before FOD, about one in three accesses to the URLs in directories A and C failed. The system reduced the rate of invalid links in directories to less than 5%.

4.3 Automatic URL recommendation

Automatic URL recommendation automatically looks for URLs on the Internet and intranets and allocates them to categories. In two examples, we allocated 3172 URLs from 122 users' bookmarks. We chose five typical categories found in the directories of enterprises. The Java category has 26 URLs. All contents are written in Japanese, and half of them are standard documents such as syntax definitions and API documents. This category represents the categories that have a set of standard documents and the technical documents in the enterprise. The category labeled XML has only three URLs, and these are technical documents. This category represents the categories that have a small number of URLs. The PERL category has 24 URLs. All contents are written in Japanese, and most of them include tips and know-how for making perl programs. This category represents the categories that have a set of know-how documents in the enterprise environment. The Procedure category of directory A has 136 URLs; these are written in Japanese and form a set of company procedure documents, for example, payment procedure documents and personnel procedure documents. There are many such categories in the large number of enterprise portals that exist. We assumed that content-based classification would not work well for the Procedure category of directory A because each content is written in a different vocabulary and there are few common words. The Contact category of di-

rectory A has 36 URLs. The contents of this category are written in Japanese and are lists of persons responsible for answering questions about their company's products. We assumed that content-based classification would not work well for this category either because there are only a few common words.

Table 3 shows the results of allocation and the precision. The precision is defined as follows:

$$\frac{N_1}{N_2}, \qquad (1)$$

where N_1 is the number of URLs that the editors accepted as suitable for the directory and N_2 is the number of URLs that the system allocated to the directory.

The URLs are allocated to categories that automatically classify according to the similarity of the contents. URLs are not allocated correctly for categories classified in the organization to which the corresponding document belongs, for example, the procedure and contact section categories.

Table 4 shows the recommendation results of a function that performs cooperative filtering of the user's bookmarks and directory A. The results show that the system recommends useful URLs to categories that are classified according to meta-information such as enterprise section information.

There is much room for improvement, but the two techniques work well together. We believe that it is effective to use the threshold of similarity level and the number of recommendations that become parameters of the recommendation algorithm according to the operation phase. When the directory has a small number of URLs and the editors want to register many, the threshold of similarity should be low and the number of recommendations should be large. On the other hand, when the directory becomes stable and the editors do not want many recommendations, the threshold of similarity should be high.

Table 3
Recommendation by content-based classification.

Category name	Recommendation	Correct	Precision
Java	5	2	0.40
XML	5	4	0.80
PERL	5	3	0.60
Procedure	5	2	0.40
Contact	4	0	0.00

Table 4
Recommendation by social filtering.

Category name	Recommendation	Correct	Precision
Java	5	3	0.60
XML	5	5	1.00
PERL	5	4	0.80
Procedure	5	2	0.40
Contact	4	3	0.75

5. Conclusion

In this paper, we listed the tasks for directory management and investigated the problems associated with these tasks. Then, we introduced a directory management system we are developing called FOD and described the functions of FOD that solve these problems. Next, we described the types of log analysis that FOD produces. Lastly, we described the effectiveness of using FOD on five in-house directories and presented some example results of using FOD's automatic URL recommendation function.

From interviews with editors, we identified the following requirements for FOD:

1) The editors require guidance about the kind of categories to make, which names to give the categories, and where to locate the categories in the directory.
2) FOD provides the analysis results and category data in the form of RDF,[13] and the editors can transform the statistics data and category data using their own screen designs. A document conversion processor and an XML database to handle this data are required.
3) The system currently provides log analysis results that enable editors to perform the tasks described in Section 3.3. The editors

want the log analysis results to recommend actions, for example, delete certain URLs and categories and reallocate certain URLs to other categories.

References

1) Yahoo!
http://www.yahoo.co.jp/
2) Google.
http://www.google.com/
3) Open Directory Project.
http://dmoz.org/
4) T. Ugai: Document directory management system for Intranet. (in Japanese), Information Processing Society of Japan, 99-GW-40, 1999.
5) HTML Lint.
http://validator.w3.org/
6) T. Ugai: Automatic collection and classification for document directory management. (in Japanese), Japanese Society for Artificial Intelligence, the 62nd national meeting, 2001.
7) H. Tsuda, T. Ugai, and K. Misue: Link-based Acquisition of Web Metadata for Domain-specific Directories. PKAW2000, 2000, p.317-324.
8) T. Ugai and K. Misue: Interaction between a Large Directory and Bookmark. (in Japanese), INTERACTION2003.
9) J. Rucker and M. J. Polanco: SiteSeer: Personalized Navigation for the Web. *Communication of the ACM*, **40**, 3, p.73-75 (1997).
10) M. Balabanovic and Y. Shoham: Fab: Content-based Collaborative Recommendation. *Communication of the ACM*, **40**, 3, p.66-72 (1997).
11) B. Sarwar, G. Karypis, J. Konstan, and J. Riedl: Item-based Collaborative Filtering Recommendation Algorithms. Proceedings of the 10th Intl. World Wide Web Conference (WWW10), 2001, p.285-295.
12) K. Misue and T. Ugai: Maintenance Support of Corporate Directories with Social-filtering. HCII2003.
13) RDF.
http://www.w3.org/RDF/

Takanori Ugai received the B.S. and M.S. degrees in Information Science from the Tokyo Institute of Technology, Tokyo, Japan in 1990 and 1992, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1992, where he has been engaged in research of formal description techniques, enterprise security, and Web document processing. He is a member of the Information Processing Society of Japan, the Japan Society for Software Science and Technology, the Japanese Society for Artificial Intelligence, and the Association for Computing Machinery.

Kazuo Misue received the B.S. and M.S. degrees in Information Science from the Tokyo University of Science, Tokyo, Japan in 1984 and 1986, respectively. He received the Ph.D. in Engineering from the University of Tokyo, Tokyo, Japan in 1997. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1986, where he has been engaged in research of diagrammatic thinking support, visual text mining, and Web document processing. He is a member of the Information Processing Society of Japan, the Japan Society for Software Science and Technology, the Japanese Society for Artificial Intelligence, and the Association for Computing Machinery.

Kunio Matsui received the B.S. degree in Information Engineering from Shizuoka University, Shizuoka, Japan in 1980. He received the Ph.D. in Engineering from the Tokyo Institute of Technology, Tokyo, Japan in 2003. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1980, where he has been engaged in research and development of natural language processing systems and information retrieval systems. He is a member of the Information Processing Society of Japan.