# HTML Document Analysis for Information Extraction

**Radek Burget**, PhD student,
Department of Information Systems, FIT,
Supervisor: Doc. Ing. Jaroslav Zendulka, CSc.

*The today's World Wide Web contains a vast amount of information stored in HTML documents. However, the HTML language primarily describes the look of the documents and it doesn't contain facilities for the description of contained data structure. In this paper we propose a model of a Web site that describes logical structure of contained data. Furthermore, we propose methods for creating such a model by analyzing the look and the structure of HTML documents.*

## 1 Introduction

Today's Web consists above all of a huge amount of documents written in Hypertext Markup Language (HTML). These documents contain alltogether a vast amount of data, but the used HTML language provides very poor facilities for describing the structure and semantics of this data. It only allows description of several basic constructions which are important for the text layout like headings, tables, lists (ordered or unordered) and above all links (i.e. references to other documents). Furthermore the authors of the Web pages often don't care about the semantics of particular HTML constructions. The reason is that the authors of Web pages are mostly interested in the final look of the page only.

The knowledge of contained data structure is however essential for information extraction from the Web content. It allows to locate relevant information in structured parts of the page and to categorize this information. Although the data structure is usualy not specified by means of the HTML language, most of the Web pages (or at least some parts of them) usualy have clear structure which allows an easy orientation for a human reader. In other words, the Web page usually contains significant information about its structure but this information is hidden in the look of the page: in the properties of the text (e.g. text font or color) and the layout of the text.

In this paper, we propose a hiererchical model of a Web site that represents its logical structure and relations between contained data abstracting from the physical organization and we describe the process of HTML analysis in order to create this model.

# 2 A Web Site

In this paper with the notion of *Web site*, we understand a Web page or the set of Web pages (single HTML documents) which concern one topic. For example, we can find many Web sites dedicated to software projects, company Web sites, personal Web sites, etc.

Exact meaning of this notion always depends on the point of view. We can consider a Web site dedicated to a particular software project. However, this site can form a part of some company Web site etc. Anyhow, a Web site always has to have strictly specified bounds. In other words, we have to be able to say for each Web page if it forms a part of a particular Web site or it doesn't.

When creating a logical model of a Web site, we have to abstract from its physical organization. The resulting model has to be independent on the means that are used for the data presentation at the page. For example, structured data in a personal page (like *name*, *position*, *age*, etc.) can be presented using a table or using a list. The logical structure of the data is however the same in both cases.

Furthermore, the data can be split to several HTML documents that are connected using links (references). Let's imagine a Web site dedicated to Ernest Hemingway. The site consists of three HTML documents connected with links. We can see the link structure in the figure 1. The first document is the *main page* and it contains references to another two pages called *Personal data* and *Bibliography*. These pages contain a link back to the main page.

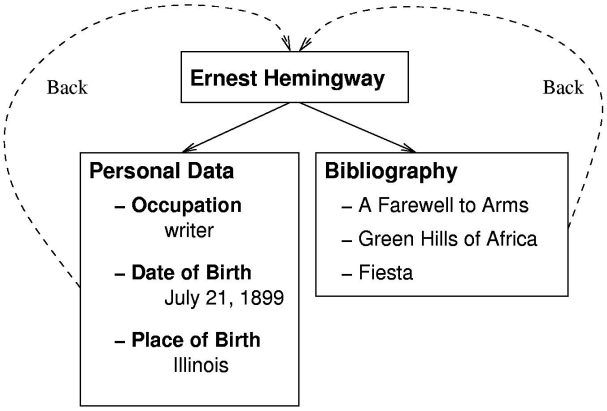


Fig. 1: A graph of the link structure

The logical structure of the data is showed in the figure 2. We can see that the links from the main page are important for the logical data structure because they form a subtree of the logical model tree by adding the logical structure of destination pages to the model. However, the links back (the dashed lines) are not important from the view of logical data structure.

We can see that we have to distinguish between the page references that are *significant* for the logical data structure and references that only serve for more effective navigation in the structure.

The process of the model creation consists of two major parts:

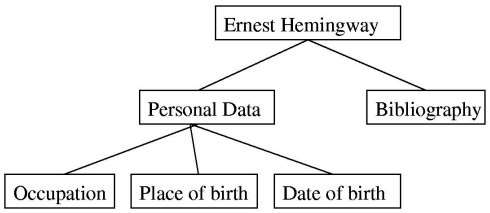


Fig. 2: Logical data structure

1. **Page-level analysis.** In this phase we analyze the code of individual pages. First, we analyze the structure of headings. Secondly, we analyze the HTML constructions contained in the page, especially *definition lists* and *tables*.

2. **Link structure analysis.** In this phase we decide which references contained in the documents are significant for the logical structure and we create a tree node for each of these links.

We start with the page-level analysis of the main page. During this analysis, we create top level nodes of the model tree. If there was some node created that corresponds with some significant reference to another page, we repeat the analysis for the referenced page. This will produce a tree model of the referenced page which is added to the model of the whole site as a subtree of the corresponding link node. The links that point out of the site bounds are ignored (treated as simple text). Note that each HTML page can be analyzed only once. If there are more references to the same page, only the first of them is considered.

As a result we obtain a tree model of the site which describes the relations between contained data.

# 3 HTML Code Representation

For the representation of the HTML documents during the analysis we propose a hierarchical model that represents contained constructions simultaneously at different levels of abstraction. This model consists of *Page Elements*. A page element represents any HTML structure from the simple ones (parts of text, links) to complex ones (lists, tables). The elements are organized to a hierarchical tree structure. The root node of the tree represents the dokument as whole. Its child nodes describe the document at highest possible level of abstraction. Any of these nodes can be either a leaf node of the tree or it can be a root node of a subtree. In this case, the subtree contains page elements that can be used as an alternative representation of corresponding part of HTML code.

We have defined several types of page elements. A *Text Element* represents a continuous part of the text with constant attributes. Actually, it's always a text between two nearest HTML tags. A *Link Element* represents a link defined using the `<A>` tag. This element has always a subtree which describes the same part of the HTML code, ignoring the fact that this part is marked as a link. A *Table Element* is

a representation of a table and its contents. A subtree of this node always contains elements describing the individual table cell contents.

# 4 Analyzing the Link Structure

The task of link structure analysis is to find all the references (links) in a Web page that are important for the logical structure of the Web site. The references that are not significant are then treated as plain text.

Our method is based on the fact that the structured Web sites usually contain somethink like a menu or table of contents – a set of links that lead to various sections of the Web site. We call such a structure a *guide-post*. We presume that all local references within the guide-post are significant.

For discovering a guide-posts within the page, we use regularity analysis. We attempt to find some regular sequence of page elements that contains exactly one Link Element. When we find such a sequence with more than certain number of repetitions, we have found a guide-post.

# 5 Chapter Structure Analysis

The HTML language contains a facility for definition of chapter headings. There are six header levels: **H1** to **H6**. During the chapter structure analysis, we add the tree of contained headers to our logical Web site model according to their level.

# 6 Table Analysis

A table is one of the most frequently used construction in the Web pages. A table can serve to present the tabular data but very often it is only used as a hidden instrument for achieving desired page layout. In the first case we would like to add the structure of the table to our logical Web site model. In the opposite case we ignore the table and we process its contents as separate page elements.

When analyzing the table, as a first step we decide if the data is organized in rows or in columns. When no header cells are specified we expect that the header cells are in the first row or the first column and we try to prove this assupmtion by analyzing the cell styles. The header cells should have different style (text style, background, etc.) than the rest of the cells. When we find a header we analyze each column or row recursively as a separate subtable.

If we fail to find any header fields (marked in the code or estimated) in the whole table we say that the table has no structure and we treat it as individual page elements.

# 7 Other constructions

Another HTML construction that is very suitable for structure analysis are *Definition Lists* that explecitly define the relation between a term and its definition. Each of the list items forms one parent – child pair in the model tree.

# 8 Conclusions

We have implemented an experimental system using the algorithms stated above and first test results show that this is a promising way for creating a logical model of a Web site. But the experiments have also shown many problems that have to be solved.

Firstly, the algorithm for discovering regular areas in the link structure gives good results for surprising amount of various Web sites but it gets confused with any small irregularity in the HTML code (e.g. a highlighted word). Some more sophisticated method should be developed that whould distinguish between significant and not significant irregularities in the structure.

There is also a need to analyze some commonly used patterns in the plain text elements like **Term: definition** (e.g. Position: assistant) and similar ones. The use of Text Elements could help this analysis because it allows to locate the boundaries of such expressions.

# Acknowledgement

This work has been supported by the long term grant project of Ministry of Education No. J22/98:262200012 "Research of information and control systems".

# Reference

[1] George Chang, Marcus J. Healey, James A. M. McHugh, Jason T. L. Wang: *Mining the World Wide Web: an information search approach*, Kluwer Academic Publishers, 2001

[2] Paolo Atzeni, Giansalvatore Mecca, Paolo Merialdo: *Semistructured and Structured Data in the Web: Going Back and Forth.* In Proceedings of ACM SIGMOD Workshop on Management of Semi-structured Data (1997)

[3] Serge Abiteboul: *Querying semi-structured data.* In Sixth International Conference on Data Base Theory (ICDT'97), Delphi (Greece), 1997

[4] *HTML 4.01 specification*, The World Wide Web Consortium, 1999 `http://www.w3c.org/TR/html401`.