

A Gateway From HTML to XML, Review and Expectations

Friedrich Dimmel*

15th April 2005

Abstract

Using the H2X¹-Gateway from Tao Fu and Mengchi Liu, it's possible to convert any HTML web site to a structured XML document. The clue with their algorithms is that you do not have to adapt the program for different web sites, the program will run in each environment. This article shows, how this approach can work, it shows, where to find its limitations and explains the program's features.

1 Introduction

There are different ways of getting out information from already existing web pages in HTML format. First we can just filter out all HTML tags to get plain text. With this approach we will need human "help" to sort out relevant data and bring it into the right order.

Another way of retrieving information is to use pattern recognition for listings and tables in web sites. That means, if we review the output from e.g. dynamically created overviews or product lists from a web shop, we can assume that each product in this listing may have the same layout format. So we can use this information to group each tables columns together.

The approach of the authors of this paper is quite similar to this one. They heavily use pattern recognition out of layout markups. That means the software checks for bold or emphasised texts and uses them as relevant nodes in an XML document. It finds out, which text next to those elements can be important and calculates their relevance.

*Registration No.: 0302230; Classification No.: 534; Study: Software and Information Engineering;

¹H2X: HTML2XML

With all those methods it's possible to write down XML documents out of existing HTML sites. In this paper I'll mention the layout-based retrieval approach.

2 System overview

The H2X Information Extraction program uses the HTML's DOM² interface. The DOM is defined as a number of element nodes where attributes and text nodes are defined to. To separate the information units in the HTML DOM, the authors use *records* and *delimiters* as terms to describe the meaning of text blocks. The term *record* means meaningful text values of HTML nodes which consist of characters used in a certain natural language and the attributes *href* and *alt* (other attributes are generally used as layout-describing elements). *Delimiters* are the opposite, text blocks without making meaningful sense in the document's context, and attributes like *align*, *height*, *width*, ... which are only used for layout purposes.

To deepen the meanings of texts in an HTML structure, we've to differentiate between *content blocks* and *topics*. *Content blocks* can be explained as similar items on the same level in a DOM tree, e.g. sub-categories below a certain category. *Topics*, however, are regarded as a subject above other records. It's assumed that any content block in the HTML document has a default topic, if no default topic could be found, the software selects a record to be the default topic.

3 Work flow

The H2X program works in two stages. The first one transforms the HTML DOM into a Transitional DOM (T-DOM). The T-DOM is a program generated middle-tier model, which allows to select nodes in the XML tree more easily. Therefore the nodes are classified into records and delimiters as mentioned above. The *classification* is as easy to understand as logical in its design: bold written records must be more important than non-bold fonts. In a next step (*Content Partition*) the records are grouped into content blocks, this can be done using delimiters as boundaries. In the *Topic Aggregation* step the software searches for topics (as above) and transforming the data into a hierarchical relation tree.

The second stage uses extraction patterns to recognize the structure and importance of sub-nodes and transforms the T-DOM into the XML-DOM. These patterns can e.g. consist of regular expressions for finding similar text patterns.

²DOM: Document Object Model, <http://www.w3.org/DOM>

How does content partition work?

HTML has been designed as a markup language (HyperText Markup Language) so we can say, its purpose was to bring structure in text documents and highlight important passages. So it's a pretty good idea to use exactly this feature to recognize, which parts of the document may be more important than others or which parts belong together and so on. Furthermore we've to find boundaries between text blocks and remove them correctly to get out an XML structure. We've to separate explicit boundaries, which are HTML tags like `<hr>` (horizontal line), `<table>` for displaying a table structure, ... from implicit boundaries such as `
` which is just a line break. We've to remove all this tags and take care, how much whitespace was intended to be between the text blocks, so the software can decide, how related such blocks are.

Topic Aggregation

The algorithm of the software clusters consistent records or blocks into topics. Topic nodes are normally emphasized such as bold fonts, indented paragraphs or something similar. Another way to get out topics nodes is to check for their frequency such as in lists where we always can see something like: “**ArticleNo:** 123232, **ArticleNo:** 123456, ...”. The algorithm finds out, that “ArticleNo:” has to be a topic.

There also hierarchies in HTML documents which have to be flattened by the algorithm into a single-dimensional space. Therefore we can use static clustering rules for topic node detection, e.g. using the header tags (`<h1>` ... `<h6>`). Or we use dynamic clustering which searches for similar nodes differed by their attributes and uses a *Frequent Structure Mining* formula to get out its frequency of occurrence.

To bring multi-dimensional tables down to one dimension, the H2X Gateway uses a table normalization algorithm which merges multi-column or multi-row cells so that they will keep their sense. After that, the table headings will be written into each line before each data field and marked up. After that, each dataset is one-dimensional.

4 Limitations

The software only works for the most standalone HTML pages with only little CSS markups. Sometimes web designers don't use the original `` tags or headings like `<h1>`. They define their own CSS classes and use e.g. `` tags which cannot be distinguished as important or less important from the algorithm.

Another problem is, when web designers use `<div>` tags (which of course should be done, when setting up the layout of a website correctly) because—like above—using stylesheets or even a script language like JavaScript could have a big influ-

ence on what is visible on the browser window for the user. It's possible that the algorithm finds text blocks in the source code, which do not belong to the topics visible on the screen.

The algorithm uses regular expressions in the end of its turn. For example the english terms "Blvd." or "St." for "Boulevard" or "Street" which are specific for the English language and useless or can even worsen results in other languages (e.g. "St." means "Sankt" in German, which means "Saint" in English).

5 Conclusion

The algorithm is a great approach to use layout elements from an existing web page to transfer its structure into an XML document. This is a good feature, because with an well-formed XML document you can do nearly anything. The idea for using exactly this layout tags is as simple as genius. The authors write about the paper's difference to any other approaches because other's don't use the layout as it looks like, they use heavily too much accuracy in analyzing structures of source code. This paper uses more viewer-centric pattern recognition.