From Image to SGML/XML Representation: One Method

Yalin Wang[†] **Ihsin T. Phillips** [‡] **Robert Haralick**[†]

[†] Department of Electrical Engineering
University of Washington Seattle, WA 98195 U.S.A.

[‡] Department of Computer Science/Software Engineering
Seattle University, Seattle, WA 98122 U.S.A.

{ylwang, yun, haralick@george.ee.washington.edu}

Abstract

In this paper, we propose a system that converts an OCR output of paper documents to a well partitioned SGML/XML representation. The system consists of a commercial OCR, a document layout analysis module and a markup module. The DAFS file structure is used to represent the document hierarchy. We give the background of the DAFS file structure, a brief overview of our document structure analysis (DSA) and markup modules. An example of our system is also shown.

1 Introduction

For the last few decades, the document layout analysis has been researched extensively. However, because of the lack of tools, researchers were not able to translate or view or share their results conveniently. With the advance of web technology, Standard Generalized Markup Language (SGML,ISO 8879)/Extensible Markup Language (XML) enable the researchers to fully explore the results of their document analysis systems. As a markup language, XML has numerous benefits and applications [1]. In the field of document analysis, XML can be used as a tool for transmitting and rendering the recognition results. Using the XML representation, the recognition results also can be used in information retrieval easily.

In this paper, we propose a system that converts an OCR output of paper documents to a well partitioned SGML/XML representation, say, Structured Document Format (SDF), which is a Document Type Definition (DTD) written in XML and reliably identifies the basic structural of the paper documents. The system architecture is given in Figure 1. The system consists of a commercial OCR, a document layout analysis module, and a SDF markup module.

We assume that given a paper document, the OCR produces, at the minimum, a set of characters (say glyphs) where each character is associated with its ASCII value and the bounding box.

The document layout analysis (DSA) module takes the output of the OCR, and produces a document hierarchy in the form of DAFS (Document Analysis File Structures) [2]. The produced hierarchical structure captures the physical structure and the logical meaning of the input document. The top of the hierarchical structure presents the entire page, and the bottom of the structure includes all glyphs on the document. Entities in the hierarchy are labeled and are associated with a set of attributes describing the nature of the entities. For example, the character set on a textual document would reside at the bottom of the hierarchy; each character would be labeled as a "glyph", and the attributes for the glyph may be the ASCII value, the font style, and the position of the character. The next level up may be words, then, text-lines, text-zones, text-blocks, and so on to the entire page.

The markup module converts the structured DAFS hierarchy into SDF file according to the SDF syntax specifications.

In the following sections, we give the background of the DAFS file structure (in Section 2), a brief overview of our DSA module (in Section 3), our markup module (in Section 4) and one example of our system(in Section 5).

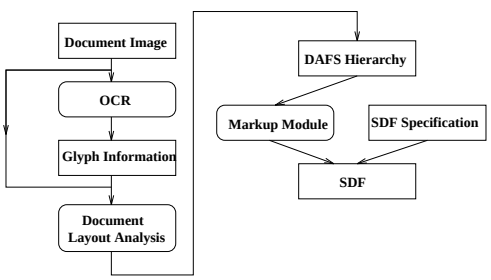

Figure 1: The System Architecture

2 DAFS File Hierarchy

The Document Attribute Format Specification (DAFS) [2] is a file format specification suitable for document decomposition applications, such as the document layout analysis, OCR and the logical analysis. The DAFS utilizes several existing document interchange standards, such as the standard generalized markup language (SGML, ISO 8879), the standard universal character set (UNICODE, ISO 10606), and the CCITT group IV standard for bi-level image compression.

DAFS provides a format for breaking down documents into entities, e.g. glyphs and words, defining entity boundaries and attributes, and labeling their content (text values and/or bitmap images). The DAFS entities are conveniently defined as the objects of interest within a document such as a paragraph, or text line, or word.

DAFS permits the creation of "parent", "child" and "sibling" relationship among entities, forming the basis for specifying any hierarchical relationships desired. As an example, consider a paragraph entity made up of text lines, and the text lines composed of words. The words are the child entities of each text line, while the paragraph is the parent entity of each text line. The other text lines in the same paragraph forms a given text line's siblings (see Figure 2).

In addition, DAFS permits the creation of an unlimited number of user defined properties. A property is used to describe or classify an entity and its contents, and exists only in association with the entity to which it refers.

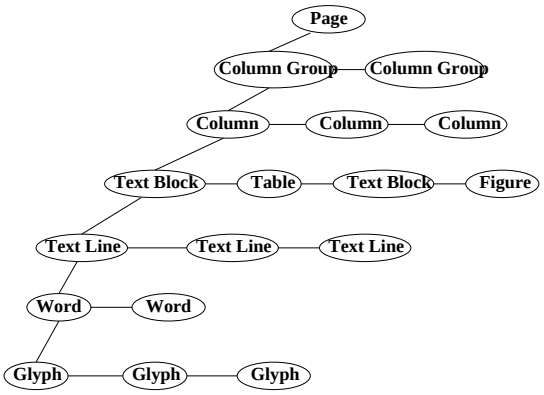

Figure 2: An example of a DAFS hierarchical structure.

3 The Document Structure Analysis Module

The input to the Document Structure Analysis module (DSA) is the glyph information, the location and the ASCII value of the glyph, from the OCR module (or other source). From the glyph information, the DSA module first constructs a one-level DAFS structure where the entities are glyphs. The output of DSA is a hierarchical DAFS structure. The hierarchical structure (from top-to-bottom) consists of column-groups, columns, zones (paragraphs, tables, figures, etc.), text lines, words, and glyphs (see Figure 2 for an example.)

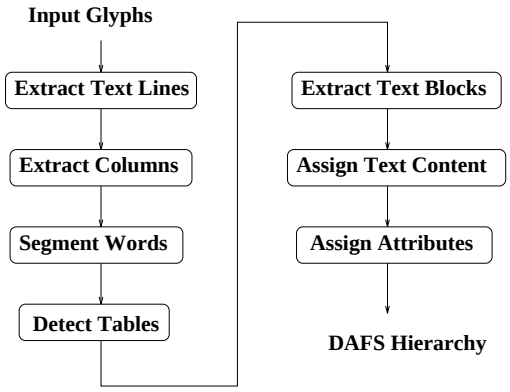

Figure 3: The DSA system architecture

There are seven sub-modules within the DSA module. (see Figure 3.) The DSA works as follows. The spatial configuration of glyph bounding boxes are analyzed to extract text lines ([4], [5]). The extracted text-lines are further segmented into words. From the extracted text-line, the text-columns and text-regions are formed to extract text-blocks (i.e. paragraphs). Within each text region, the neighboring text-lines are merged to form text-block (paragraph) based upon the changes in the inter-text-line spacing and the text-line justification [6]. Each tabular structure is further decomposed into row and column items for further analysis. For each text line, words and word spaces are formed into a text string. The formatting attributes are assigned to each text string. The result of this module is represented as a hierarchical document structure, a DAFS structure (see Figure 2.), which is comparable to the SDF syntax.

4 Markup Module

The input to the markup module is a hierarchical DAFS structure and the output of this module is a well partitioned SGML/XML representation, an SDF file. The elements defined within an SDF file are page, column-group, column, text-block, table, figure, and text-line. The SDF hierarchy is similar to that of DAFS hierarchy (see Figure 2). A column-group can have one or more column elements. A column element can have one or more text-block el-

ements and non-text elements (such as tables, figures, etc.). A text-block element can have one or more text-line elements. The markup module directly translates the input DAFS structure to an SDF file according to the rules of the SDF definition.

Within a DAFS hierarchy, besides the content, each entity is associated with a list of attributes. The attributes we use are similar to those used in the UW databases [3]. We add a few formatting attributes to text-line elements.

The translation process works as follows. We visit all nodes in the input DAFS hierarchy in a depth-first fashion. At each node, we translate the contents and the attributes of the node into an SDF element statement, according to the node type and the SDF's syntax rules. For example, if the input DAFS hierarchy has a column-group entity that contains two columns. And within one of the columns there is a table, a figure, and two text-blocks, and the other column has five text-blocks, the translation result of the SDF file would have 14 elements – one page, one column-group, two columns, one table, one figure and eight text-block elements. (Here, we do not count the text-line elements.)

Within an SDF, the content of a text-block is stored as a sequence of text-line elements. A text-line element is represented as a character string with the formatting attributes attached to each character in the string. An SDF statement for a table element is a sequence of matrix elements, caption element and legend element. And each matrix element can have one or more cell elements. The caption and legend elements are defined as text-blocks. A figure element in an SDF is given by the name of the bit-map file of the figure.

5 Illustration

An example has been constructed to illustrate our proposed system. We compiled the LaTeX file of this paper and use our in-house software tool to convert the DVI file to obtain a TIFF image and the information for each character in the images. The information includes the ASCII, the location and font attributes. The character information of the second page of this paper is used as the input to our system, instead of the result of an OCR system. Currently, we do not have figure zone detection in our DSA module, we marked the figure zone in the second page manually to construct our example. Also, since our SDF is still under construction, we use HTML to do the markup of our DSA result. The result of the markup module is shown in Figure 4.

References

[1] Ken Sall, "XML: Structuring Data for the Web: An Introduction", *http://wdvl.com/Authoring/Languages/XML/Intro*

[2] "DAFS Document Attribute Format Specification", RAF Technology, Inc., Redmond, Washington, 1994.

[3] Ihsin T. Phillips, "Users' Reference Manual", *CD-ROM, UW-III Document Image Database-III*, 1995.

[4] Jisheng Liang, Ihsin T. Phillips, and Robert.M. Haralick, "A Unified Approach for Document Structure Analysis and Its Application to Text-line Extraction", *Proceedings of the 1999 Symposium on Document Image Understanding Technology (SDIUT99)*, pp32-41, Annapolis, Maryland, April 14-16, 1999.

[5] J. Liang, Ihsin T. Phillips, and Robert M. Haralick, "A Statistically Based, Highly Accurate Text-line Segmentation Method", has been accepted and to be published in *Proceedings of the 5rd International Conference on Document Analysis and Recognition (ICDAR99)*, Bangalore, India, September 20-22, 1999,

[6] Jisheng Liang, Ihsin T. Phillips, and Robert M. Haralick, "Consistent Partition and Labeling of Text Blocks", has been submitted to *the Journal of Pattern Analysis and Applications*, May 12, 1999.

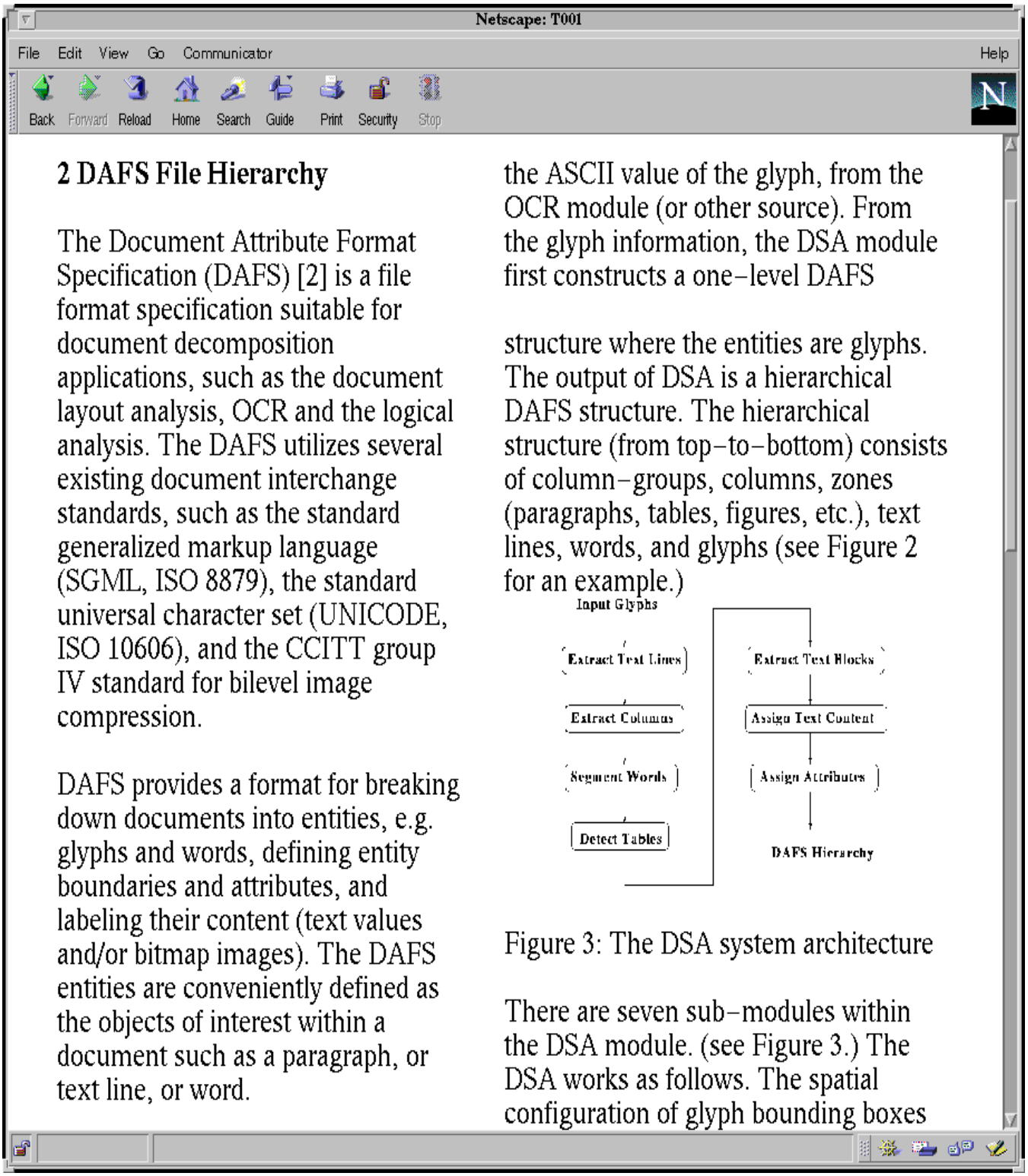

Figure 4: The converted HTML file of the second page of this paper