

Steven J. Simske
Hewlett-Packard Company, 700 71st Avenue, Greeley CO 80634
Phone: 970-350-4920; Fax: 970-350-5388; e-mail: Steven_Simske@hp.com

The Use of XML and XML-Data to Provide Document Understanding at the Physical, Logical and Presentational Levels

The current array of SGML-derived encoding schemes (markup languages), such as HTML, XML and XML schema languages (such as DDML, SOX, DCD, XML-data, and other XML DTD's [Document Type Definitions, which allow translation, or parsing, of the XML document]), as well as embeddable scripted languages like JavaScript, provide a platform-independent manner in which to reconstruct (format) documents. In its presentational level, the internet is document-centric (inasmuch as multimedia insertions can be viewed as elements on the "document"). However, the manner in which the information travels from creation to presentation can vary widely, depending on the document, the application and the designer. This extended abstract intends to show how XML can be used to help break down the problem of document understanding into a reasonable set of component stages, allowing for a common set of physical, logical and presentational schemas while supporting extended schemas. This approach, it is hoped, will reward and encourage both innovation and the ability to comparatively test distinct document understanding packages for their ability to achieve default behavior.

The Physical, Logical & Presentational Levels of Document Understanding

Document layout interpretation is a critical part of an overall document understanding technology, that herein will be viewed as comprised of three largely autonomous levels: (1) the physical, (2) the logical, and (3) the presentational. The second level (logical) is generally associated with document layout interpretation, and as such will be the primary focus herein, but a coherent understanding of each of these levels will be beneficial.

The physical level deals with zoning the document into pages, regions on pages, and in some cases (such as text columns, tables, etc.) composite or associated regions. This level is thus often designated the zoning, segmentation, region-forming, geometric analysis, or page analysis level. In general, the physical level is the first stage of document understanding, providing as its output segments (regions) on the document that have been (a) defined geometrically, (b) typed for data storage, compression and other treatment purposes, and (c) possibly prepared for the logical stage through the use of specific region-type processors (e.g. optical character recognition [OCR] for text, palette definition for photos, and raster to vector conversion for line art or drawings). Often, this level of document understanding will suffice for the application user (e.g. someone who merely wishes to scan a photo into a publishing application, an OCR'ed copy of a document into a database, or a specific drawing into a word processor file).

The logical level is, in general, required to provide useful automaticity to a document understanding package. This level is also referred to as the structural, the meta-data, the contextual or the meta-informational level. It is the stage at which information about information (meta-information) is generated. For example, the "text" generated at the physical level may be labeled here as "title", "author", "page number" or "URL". "Drawings" may be labeled as "logos", "photos" as "portrait" or "scenery". At a broader level, document types ("business letter", "XYZ Journal article", etc.) are identified. The information at this level can be attached to the individual regions formed at the physical level (e.g. "URL region type") or to the document itself (e.g. "business letter").

The presentation level, traditionally handled by the SGML derivative HTML (often with scripting such as that provided by JavaScript embedded into the HTML), provides the final understanding of the document. At this level, also known as the rendering, destination, or demonstration level, the various parts of the document are presented for viewing. The destination, be it a graphic display monitor, a printer, an application (such as a page editor, word processor, etc.), should be able to render a version of the document

similar in appearance (and content!) to the original document. A variety of means are provided for this, including formats such as PDF that focus on presentation (but are not generally editable) and various printer drivers.

This breakdown of the process of document understanding represents a particular vision for task flow from the creation to rendering (e.g. scanner to document understanding package to screen destination). Other task flows, however, are readily understood in this context. Documents can be stored in their region + meta-information form, then loaded into a browser for viewing. Similarly analyzed documents can be directly sent to a printer without viewing. The full path is discussed here only for completeness.

XML, XML-Data and Document Understanding

Extensible Markup Language (XML), fully described at <http://www.w3.org/TR/REC-xml>, is a SGML subset that provides several features that improve on HTML. Among these are tag flexibility, which allows for meta-information creation and interpretation; and schema extension (the XML-Data specification for schema extension will be used in the examples herein), which opens up XML to inheritance and to the extension of a subclass. Tag flexibility is useful for increasing the diversity of objects formed at the physical level of document understanding; for increasing the amount, richness and utility of meta-information at the logical level of document understanding; and for providing destination-specific instructions at the presentation level that can themselves be largely device-independent. Schema extension provides the means for specific applications and devices to differentiate the quality of their digital document solutions, be it in the fields of digital capturing (e.g. scanning, digital photography, etc.), document creation (e.g. word processing, publishing, photo editing, etc.), digital document understanding, and/or digital rendering.

The advantages of tag flexibility are well-known and are likely the primary impetus behind the rapid acceptance of XML. It eases data extraction, search engine productivity, data exchange, and meta-information packaging and interpretation. Schema extension, on the other hand, offers a similar increase in productivity and capability to that offered by OO (object-oriented) computer languages when compared to non-OO (structured) languages. Through schema extension, increased capability can be seamlessly added to a document understanding package, while predictable and acceptable performance can be achieved through the use of “public” or “default” XML behavior. A brief albeit cogent example of the power brought by schema extension in each of the three levels is provided in the following section.

Examples

At the physical level, a document may be zoned by a document analysis, OCR or other segmentation package. The output of this package will typically be a list of regions, typed and usually ordered (e.g. for text flow). A generic XML description of such a region may be as follows:

```
<Region>
  <BoundingBox>
    <xmin> 100 </xmin>
    <xspan> 1200 </xspan>
    <ymin> 100 </ymin>
    <yspan> 100 </yspan>
  </BoundingBox>
  <Type> Text </Type>
  <!-- other default region attributes would be included here -- >
</Region>
```

which describes a text region starting at (x,y) = (100,100) and spanning 1200 pixels in the x-direction and 100 in the y-direction (e.g. it may be a large font title if the resolution is 300 ppi). For purpose of illustration, suppose that the “font size” of the Text is not part of the default description of the <Region>. Using XML schema extension, this information can be added to the region definition:

```
<relationType id="FontSize" extends="http://generic_region.org/#FontSize">
  <pcdata/>
</relationType>
<elementType id="Region" extends="http://generic_region.org/#Region">
  <relation href="#FontSize"/>
</elementType>
```

A similar default schema/extended schema combination can be used for the logical and presentation levels of document understanding. The following table gives an example system, incorporating default and extended schemas for each of the physical, logical and presentational levels. It is not comprehensive and may be in parts arbitrary, but is meant to convey the concept of default/extended schemas in a simple fashion. Note that, as in the example above, many extended schema elements may only apply to certain elementTypes.

Level	Default Schema Elements	Extended Schema Elements
Physical	Region types Bounding box Polygonal boundaries Region order Style sheet association* Skew Orientation	Composite relationships Background regions Additional region types Histograms Font size Z order Threshold value
Logical	Text block associations Logical description of regions Language Source & document type Purposing	Task-specific meta-data User-specific routing User-specific storage/retrieval Learning algorithms
Presentational	ASCII Bitmap Location-specific layout Relational layout	Bezier curves Combined motifs Ornamental motifs

* See <http://www.w3.org/TR/1999/xml-stylesheet-19990629/>

At the physical level, it seems reasonable to make elements that all regions require part of the default schema. This includes boundaries (polygonal and rectangular), region order, association with regions in a style sheet, skew and orientation. Elements which only certain types of regions require, or else novel region types not part of the default set, can be made part of the extended schema set. This provides proprietary ways of handling the physical document understanding, while encouraging solutions to support a minimum specification given by the default schema.

At the logical level, it makes sense to assign all of the descriptive terminology for defining the document languages and for describing regions and their locations to the default set. Less obvious is how to handle source and document typing or generic purposing, but assuming some standardization of these occur, they would conveniently be absorbed by the default schema. Task

and user specific logic, as well as proprietary learning algorithms, would reside in the extended schema, allowing innovation while ensuring minimal support overhead across applications.

Finally, at the presentation level, the current breadth of HTML and other display specifications (termed location-specific and relational layout in the table) should clearly reside in the default schema. Other, proprietary, means of enhancing displays (e.g., the use of Bezier curves, ornamental and combined [e.g. Bezier + bitmap] motifs) will be possible through schema extension.

Implementation

A system comprised of a combination of default and extended schemas for physical, logical and presentational levels should be carefully designed by the larger document understanding community. This short overview merely shows a possible pathway to this system. Where possible, existing standards (XML and HTML being the most obvious) should be incorporated. Importantly, a clear separation of physical, logical and presentational levels (or “domains”) makes possible proprietary solutions (which can be written in any high-level language, compiled and communicate with the XML document understanding schema via, for example, DOM) for any of these three domains to intercommunicate via the XML schemas for each level.

Advantages

The use of a reasonable, cogent default schema allows for ready comparison of differing algorithms for their abilities to provide document understanding within the range of a ground-truthable test. This “common ground” for testing will be of some use in terms of comparing systems for their minimal suitability, while the extended schema allows for enhanced, proprietary functionality that encourages innovation. The separation of physical, logical and presentational domains allows for ease of intercommunication among disparate algorithms or solutions. It also allows for the use of subsets of these three levels, as shown in the following table:

Paired Level Subset	Sample Applications
Physical + Logical	Document management (storage/retrieval)
Physical + Presentational	Generic markup (e.g., HTML)
Logical + Presentational	Data mining, search engine

The advent of XML suggests that the internet is evolving in a way analogous to the way computer languages have from structured to OO to visual languages. The larger document understanding community stands to benefit from a common scheme for achieving minimal default behavior while allowing proprietary solutions to the separate domains.

Acknowledgements

The author gratefully acknowledges all of the various committees responsible for creating the DCD, DDML, HTML, PDF, RTF, SGML, SOX, XML, and XML-data formats and standards. Information on these can be obtained on the internet, with a good starting point being <http://www.w3.org>. The author wishes to thank John Burns, Bob Chalstrom, Julie Dawe, Rick Lesser and Virgil Russon for helpful feedback, and to thank three anonymous reviewers for their pertinent suggestions and directions for the revised submission.