Digital Documents

Anastasios Dimou
Independent Researcher, Athens, Greece

Apostolos Syropoulos
https://orcid.org/0000-0002-9625-1482
Greek Molecular Computing Group, Greece

INTRODUCTION

Nowadays documents (e.g., books, articles, papers, etc.) are created electronically and they are printed electronically. Modern printers are programmable and so documents are uploaded to them as *programs* in some *page description language* (like PostScript, PDF, and PCL). Currently, PDF is the de facto industrial standard for printable documents so all documents are finally converted to PDF.

Definition A digital document is one that is created exclusively with digital means (e.g., computer programs), can be read with digital means (e.g., electronic book readers), and can be transferred to paper by purely digital means (e.g., fonts are digital and the printing machine transfers the digital document to paper).

Documents are created using either sophisticated GUIs or using an "advanced" text editor. Programs like MS Word and LibreOffice Writer are sophisticated GUIs that are used to create digital documents. On the other hand, one can create the same documents "manually" by using a simple text editor and/or some CLI tools. For example, it is entirely possible to create a document in the Open Document Format for Office Applications (ODF) with such tools and of course most people create their LaTeX documents using simple or advanced text editors like NotePad++.

The various file formats that are used to electronically encode documents based on the idea of *markup*. To understand what markup is consider a book that submitted for publication. Typically, a copy editor will annotate the text with various types of instructions. This is markup. In case, the markup is entered electronically, then, clearly, it is called *electronic* markup. *Specific* markup is a form of markup that tells the formatted to do each point. *Generic* markup adds information about the logical components of a document. In order to add markup to a text we need a precisely defined notation. Such a notation is called a *markup language*. The basic elements of such a language are called tags.

A markup language defines *tags* that group text and affect the way specific parts of a document will be rendered. In certain cases it is possible to define *macros*, that is, new tags that have the combined functionality of *primitive* tags. There are several markup languages that are widely used today. These include TeX, DocBook, and HTML and macro packages like LaTeX. However, in a way all these markup languages are descendants of SGML.

In what follows we will give a brief overview of SGML and XML. Then, we will discuss page description languages in some details, and we will continue with an overview of some very popular markup languages.

DOI: 10.4018/978-1-7998-3479-3.ch077

BACKGROUND

The *Standard Generalized Markup Language* (SGML) is the International Organization for Standardization (ISO) standard for document description [see (ISO-Standard, 1986) and (van Herwijnen, 1990)]. SGML is actually a meta-language that allows people to create markup languages. Typically, an SGML document consists of a file that contains marked-up data, the SGML declaration, and the document type definition (DTD). The SGML declaration specifies which characters and delimiters are used. For example, it is common to use the characters <, >, and /> as delimiters. And this why it is so common to see tags like <intro> and </intro>. Also, the The SGML declaration specifies which character set (Unicode or other) is used. Figure 2 shows an SGML Declaration for HTML 3.2. The DTD specifies how one can markup a class of documents. In particular, it defines the structure of a document and it is written in SGML. In general, no two different document classes have the same structure. Figure 1 shows a very simple DTD.

Figure 1. A very simple DTD from (Syropoulos, Tsolomitis, & Sofroniou, 2003).

```
<!ELEMENT first_name (#PCDATA)>
<!ELEMENT last_name (#PCDATA)>
<!ELEMENT profession (#PCDATA)>
<!ELEMENT name
(first_name, last_name)>
<!ELEMENT person
(name, profession*)>
<!ELEMENT scientists (person*)>
```

Today there are a few projects that are directly based on SGML. The *Text Encoding Initiative* (TEI) is such a project whose aim was to create software-independent methods for encoding humanities data in electronic form (TEI: Text Encoding Initiative). Docbook is another markup language based on SGML that was designed for technical documentation (DocBook Version 5.2, 2019). However, the current version of DocBook is an XML application.

The eXtensible Markup Language (XML) [see (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 2008) and (Harold & Means, 2009)] is a subset of SGML. The goal of XML is "to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML." XML has been used to design the file formats of many office-productivity tools including the Office Open XML format (Standard ECMA-376, 2016) and the Open Document Format for Office Applications (OASIS Standards, 2011). The Open Document Format is also the ISO/IEC 26300:2006 standard.

An XML document consists of a DTD and a data file. For example, if one uses a DTD like the one shown in figure 1, then she can create a valid XML document like the one shown in figure 3. Note that the various <something> and </something> pairs are called *elements.* One of the differences between SGML and XML is that the later uses Unicode's UTF-8 encoding as the default character encoding. In addition, since XML is a subset of SGML, it does not "understand" a number of SGML declarations, constructs, and entity declarations.