Using XML as a reading enabler for visually impaired persons

Bert Paepen and Jan Engelen

Katholieke Universiteit Leuven
Research Group on Document Architectures
Kasteelpark Arenberg 10
B-3001 Leuven (Belgium)
bert.paepen@docarch.be, jan.engelen@docarch.be

Abstract. Reading large texts like books, newspapers and magazines, is still a hard task for visually impaired persons. Accessibility software such as screen readers and screen magnifiers are a valuable aid, but they give the reader only small parts of information at a time. Since XML (Extensible Mark-up Language) can add structure to information, it can be used to help visually impaired readers cope with large amounts of information. A special XML Viewer application can give the reader additional navigation and search possibilities, enabling a personalised and fast reading experience for visually impaired persons.

1 Problem area

The emerging boom of the Internet has meant a revolution comparable to the invention of the television. Suddenly electronic information originating from the other side of the world has become accessible at the click of a button. Especially for visually impaired persons, this has created an enormous breakthrough. Thanks to the widespread existence of electronic information, visually impaired persons can now do things by themselves that they could not do before: buy groceries or CD's without the help of others, read the newspaper every morning, read their favourite magazine every week, etc.

Still, handling information using screen readers, text magnifiers, speech synthesizers or other accessibility aids remains a cumbersome task. These applications and devices typically offer information in very small chunks, for example line by line (screen readers) or on-screen area by area (magnifiers). As a consequence, visually impaired persons have to handle electronic information in parts; "seeing" the big picture mostly is impossible. For web sites this is not a big problem, since they tend to chop up their information in smaller parts using various ways of navigation. For large texts such as books, newspapers and journals, a much bigger problem exists. Large amounts of text are hardly ever presented in a structured way. They can contain chapters and paragraphs, like this paper does, but that does not mean that their structure is *offered* to the reader in a structured way. This paper shows how large texts can be structured so that it is much easier for visually impaired persons to read them.

2 XML

XML (Extensible Mark-up Language) is a *meta-mark-up language*: a mark-up language that can be used to create other mark-up languages. What is mark-up? That's easy: mark-up is a way of adding extra information to text. This extra information is called *meta-data* and mark-up is indicated using tags. So: XML uses mark-up tags to add structure and meaning to information. An example:

```
<section>
   <title>XML</title>
   <p> XML is a <em>meta-mark-up language</em>: it is a
mark-up language that can be used to create other mark-
up languages. </p>
</section>
```

The example above is an XML way of representing the information of the previous paragraph. Tags are marked with < and > signs. Parts of information start with a start tag and end with an end tag (which uses a slash). With XML you can create your own tags to structure your information in a meaningful way. HTML (Hypertext Mark-up Language) is also a mark-up language and thus comparable to XML, but there is an important difference. While HTML is mainly used to add *layout* to text, XML is used to add *structure* and *meaning*. In the example above, the term 'meta-mark-up language' is emphasised using the tag. This tag adds extra meaning to the term, saying that there is a special emphasis on it. HTML would put the <i> tag instead, saying that the term needs to be displayed in italic. The difference is subtle, but very important. If you use tags such as and <title> you know that these parts of the text have a special meaning. A lot can be done with this extra information.

Since its inception in 1996 and its birth as a W3C[1] recommendation in 1998, XML has proved to be a very powerful *enabler*. XML by itself does not do anything; it only provides a way of structuring, storing and exchanging electronic information. XML is only about tags; it is up to the developer to decide what to do with them. One of the application areas of XML is in helping to find your way in large amounts of information, like a digital library or (why not) the entire Shakespeare collection.

If you add meaningful tags to text parts, searching these parts can become more efficient. Keywords can summarise a text part, allowing search algorithms to maintain the focus on these keywords only. Taking this one step further, keywords can be extracted from documents to form a separate XML index of your documents, allowing quick search over this index. This way, XML can help overcome a major problem threatening the future success of the Internet: there is so much information out there that it is impossible for people to cope with it.

3 XML benefits for visually impaired persons

Albeit on a different level, the same is true for visually impaired persons. Because of their impairment, it is difficult for them to cope with large electronic texts such as

[1] World Wide Web Consortium

books, magazines and newspapers – texts that are much easier to cope with if you do not have a visual impairment. People reading a printed newspaper typically have a quick look at the large headlines, allowing them to browse through the paper in a couple of minutes. In this example, formatting helps people cope with large amounts of information.

Not only the amount of text can be an obstacle for visually impaired persons, but also the way this text is presented. If a blind person receives a digital copy of the same newspaper without special treatment, such browsing is impossible. He or she would have to read the newspaper line by line, not being able to see the difference between big and small headlines and not being able to skip uninteresting parts.

Furthermore, each type of reading impairment (blindness, dyslexia, colour-blindness, etc.) brings along different problems in electronic reading. Blind people do not see text formatting, colour-blind people have difficulties with certain colour combinations and partially sighted people need very big text formatting and vivid colours, depending on their specific type of impairment.

XML is very well suited in situations like this, where one information source needs to be displayed in various output forms *(single source, multi target)*. XML brings an important principle to the accessibility world: *the separation of information from its presentation form*. If you maintain this separation, it becomes very easy to create different presentation forms for different kinds of impairment. If personalised presentation of information is often seen as a gimmick, for impaired persons it is a *necessity*.

Making information more accessible thus means restructuring both the information and the way it is presented to the reader. In order to make large amounts of information accessible to visually impaired persons, they need to be offered in a structured way, so that disadvantages of line by line viewing are diminished. Further, navigation and search must be enabled in a user-friendly way. Information must be presented in manageable chunks and the way it is presented must be very flexible and user-adaptable.

4 The XML Viewer

Using these principles as a starting point, the Research Group on Document Architectures (DocArch[2]) of the Katholieke Universiteit Leuven constructed an "XML Viewer" together with the "Federatie van Nederlandse Blindenbibliotheken" (FNB-The Netherlands). The XML Viewer is an application enabling visually impaired readers to read texts of any kind (books, articles, magazines, newspapers, etc.) in an easy-to-cope way.

The XML Viewer offers information in a structured way, based on the document structure (this is mostly what a table of contents presents). Information overload is avoided by presenting the structure of a text level by level. The reader can *navigate* through this structure using shortcuts or buttons, moving to a next or previous item on the same level or moving up to a higher level in the document structure.

[2] http://www.docarch.be

An XML document can be regarded as an information tree. In a book, a chapter is the parent of several sections, which have at their turn several paragraphs as their children. Only the smallest parts of this document tree is presented to the reader as a whole. In the case of a newspaper, this is one article. In the case of a book, this can be a paragraph or a section, depending on the structure the content provider gave it. This way, information transfer uses *small chunks* at a time. Navigation allows fast browsing through these chunks.

A strict separation of text and layout is maintained: texts are stored in XML and for layout, a *style sheet* is used, containing instructions on how to present the text on the screen. These style sheets can be personalised, so that readers can use the layout that is suited best for them. *Personalised layout* is a key issue for many visually impaired readers. Readers can change the background colour, text colours, sizes and fonts separately for each type of information.

Additional features have been added in order to permit efficient document reading. *Bookmarks* can be used to fasten the reader's navigation experience. A reader can assign a bookmark to a part of the text, giving it a meaningful name. Later he or she can quickly jump to that particular part by referring to its bookmark name. An automatic bookmark is created when the application is closed, allowing a fast reference to the last position. Furthermore, *search* through the entire text is possible.

Several possibilities that paper documents have are imitated by the viewer. Readers can for example add *personal notes* to a certain part of the text, just as the reader of a book can write notes in the margin. Also, *page numbers* can be included, allowing cross-reference between the electronic and the printed version of a document. Both functions are interesting for visually impaired students using the XML Viewer on a portable computer and screen reader in a classroom situation, where a teacher refers to page numbers and makes remarks on certain parts of a study book.

A number of user settings make the application more flexible in use. Readers can choose a language for the user interface, change document locations, choose whether or not to show printed page numbers and change the name of the automatic bookmark.

Because content providers are not very keen to distributing their copyrighted material in an electronic form, a number of *security features* have been built in the viewer. Texts are encrypted in such a way that they cannot be read by other applications besides the XML Viewer. Once opened, a document cannot be printed or copied to an other application without the content provider's consent. Content providers often refuse to make their material available without these kinds of copyright protection.

5 Technical approach

A standard Microsoft Windows application was chosen as operating environment. The application was developed in Microsoft Visual Basic 6.0, allowing fast development and easy plug-in facilities.

5.1 Architecture

Several e-readers offer a combined solution, containing both navigation and text-to-speech or Braille output in one single application. The FNB-DocArch XML Viewer does not follow this approach. Instead a user-oriented architecture has been chosen. Since every person has his own preferences and special needs, letting her/him make the decision on presentation and optimum output channel is essential.

This principle is reflected in the modular architecture and design of the application. If different system functions are performed by different parts of the XML Viewer and other computer software and hardware, a very personal user experience is made possible. The functions of the XML Viewer are mainly restricted to user navigation and document display. Handling of the XML document is handled by the MSXML[3] parser (see below). The accessibility software and hardware are strictly separated from the viewer, ensuring that the user can keep his personal settings and preferences.

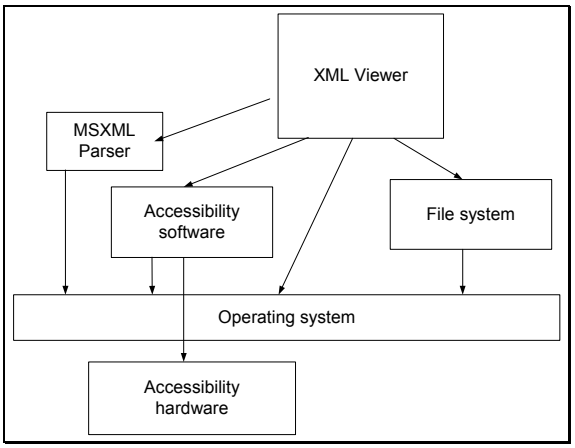

Fig. 1. Architecture

Because of this approach the XML Viewer is only an *enabler* for visually impaired persons in reading electronic information. Depending on their own choice, readers can use a screen magnifier, screen reader, voice synthesiser or any other device to 'finish the job' for them.

5.2 System design

In order to obtain a flexible application, the design of the system is modular (see figure below). When a user opens a document, the Loader uses the FileHandler to get the document from the file system. Then the Decryptor decrypts the document and the Unzipper unpacks the different parts of the zip file. Through the MainModule the documents is passed on to the Tree Builder (see below) and further to the Browser.

[3] Microsoft XML Parser

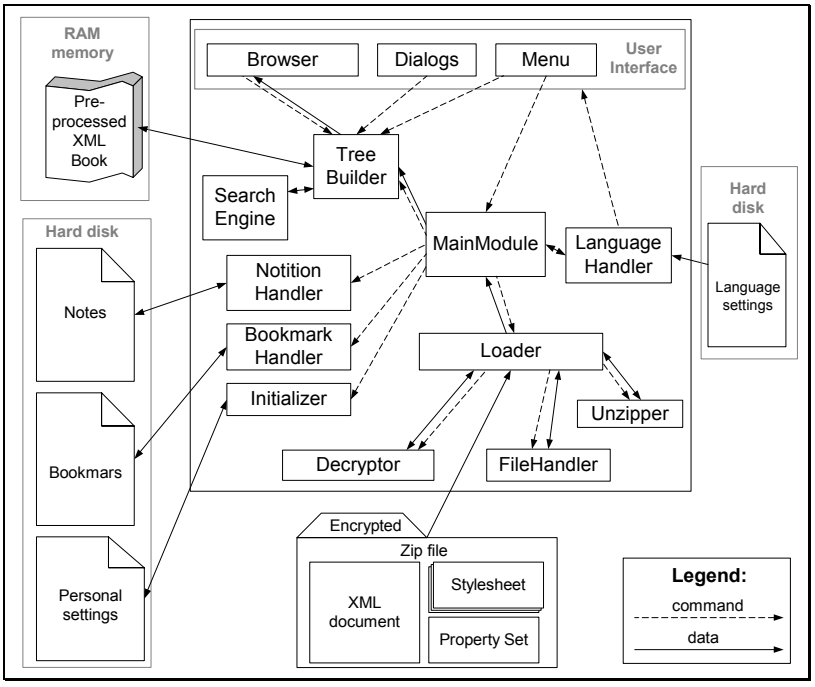

Fig. 2. System design

A document that can be opened by the XML Viewer has the extension .exd[4]. This is an encrypted zip file containing three kinds of files: the XML document, one or more style sheets and a "property set". Style sheets are used to translate the XML tags into layout. When a document is opened, the reader can switch dynamically between the available style sheets. Furthermore, the reader can adapt existing style sheets to his own preferences and he can create new style sheets.

Since XML is extensible, content providers are free to create their own XML vocabulary[5]. In order to allow this flexibility, a property set file is used with the XML file. In the property set a small part of the used XML vocabulary is defined. This way, the viewer can read almost every kind of XML document. The only restriction is that it is structured more or less like a book, with nested sections that have titles.

5.3 XML processing

Before the content of a document can be displayed in the viewer, the XML needs to be processed. This processing is handled by the MSXML Parser, which reads the document into a tree structure in the computer's memory. As long as the document is

[4] Encrypted XML Document
[5] Set of tags that can be used by an XML document.

open, this structure remains in memory, allowing quick navigation through the tree nodes. The Tree Builder module handles all navigation actions performed on the XML tree using DOM[6].

5.4 Rendering

One of the most complicated tasks to be performed by the viewer turned out to be the rendering of text – in other words the on-screen display – in an accessible way. During development, the Browser module went through a number of important changes due to these difficulties.

In a first stage, the Browser module used the Microsoft Internet Explorer component for displaying the document structure and contents on the screen. The advantage of this component is that it can "understand" XML by itself. If an XML document together with a style sheet are sent to the browser, it takes care of the entire rendering and display process. A major drawback turned out to be the component's accessibility to screen readers. Internet Explorer is a browser, meaning that it does not provide a cursor like editors do. As a consequence the screen reader software cannot position itself to the spot where the user wants to start reading[7].

A further problem was the interference of the Explorer component installed for the XML Viewer and the full Internet Explorer application already present on the computer. For the viewer, at least version 5 of this browser was required, meaning that users who were running previous versions would have been forced to upgrade. While for most persons this should not be a big problem, for visually impaired persons changing a well-functioning configuration is not that obvious: there is always a risk that your accessibility software is not well suited for the new configuration.

These problems led to a thorough study of existing browsers and editors in order to find a component that is suited for the job. It should be able to render the XML using a style sheet (browser) and it should show the text on screen like an editor does (using a cursor), without allowing the user to edit the text. A number of existing editors and browsers were examined, such as XMetal, Opera, Stilo and XML Spy. Although these applications all have their unmistakable value, the conclusion was that what we were looking for, a combination of an editor and a browser, did not exist. In a second phase of this study the possibility of re-using existing browser architectures, such as Amaya and Mozilla, was examined. Since these browsers are based on open source software projects, it is possible to re-use some parts for the XML Viewer. The problem is that these browsers are not available as components usable in a Visual Basic application. Using them would mean rebuilding the entire XML Viewer application within one of the two browser architectures (instead of the other way around).

---

[6] Document Object Model: an API for processing XML using a tree-like memory structure.

[7] This is not true for *all* screenreaders.

5.5 RTF

Finally an approach was chosen using standard Windows components without needing additional browser components for rendering. In the final approach, the navigation through the document tree takes place in a list box, while the display of the node content happens in a so-called “RTF box”. RTF[8] is a layout-oriented language that contains special tags that determine how the text must be displayed.

A major advantage of this approach is its accessibility. The RTF box has a cursor, allowing positioning by a screen reader. In spite of this, several disadvantages do remain. The RTF box is purely text oriented, meaning that it will not support any kind of multimedia[9]. Furthermore, there is no direct link between a text part on the screen and its corresponding node in the XML tree. This makes it hard to implement some additional functionality.

6 Conclusion

In this paper we have demonstrated that producing documents in XML format has many advantages for visually impaired readers. As such this is not new because our opinion is shared by several users and user groups including many national organisations and the international Daisy and eBook consortia. But we found out that actually no good XML reader, offering a minimal set of extra features besides pure reading, did exist. The market of XML-aware browsers and the available browser components was studied carefully in this respect before accepting this conclusion.

Therefore, in collaboration with FNB, the Dutch Federation of Libraries for the Blind, a special XML viewer was developed by the DocArch group. In the beginning of 2002, it has been offered as standard XML viewer for the large collection of documents available through the Dutch Web portal "Anderslezen".

Due to the great need for feature-rich XML viewing, it can be anticipated that several new developments, including several commercial ones, are upcoming. Making sure that there will be enough electronic documents available in XML remains therefore of utmost importance.

7 Acknowledgement

The development of a basic XML viewer prototype by the DocArch group was made possible through a grant of the Cera foundation. The work on the current XML viewer was funded partially by the Dutch Federation of Libraries for the Blind.

---

[8] Rich Text Format

[9] Some support is possible, for example hyperlinks and pictures, but these are not standard RTF functionality.