An Audio View of (LA)TEX Documents — Part II

T. V. Raman*
Digital Equipment Corporation
Cambridge Research Lab
One Kendall Square, Building 650
Cambridge, MA 02139, USA
Email: raman@crl.dec.com
URL: http://www.cs.cornell.edu/Info/People/raman/raman.html

Abstract

ASTER — Audio System For Technical Readings — is a computing system that produces audio renderings from the *same* (LA)TEX source used to produce the printed document. Raman (1992) described our preliminary work on this project. At the time, correct handling of user-defined (LA)TEX macros was described as one of the key issues in building a fully extensible audio rendering system. ASTER (Raman, 1994) has now been fully implemented. This paper reports on the approach used in ASTER to handle user-defined macros.

The approach used not only makes ASTER fully extensible; it points out a unique advantage of (LA)TEX — the ability of the author to encode semantic meaning into the markup by extending the document model in ways appropriate to the specific document instance that is being encoded.

Introduction

ASTER — Audio System For Technical Readings — is a computing system that aurally renders electronic documents marked up in the (LA)TEX family of markup languages (Raman, 1994). ASTER uses the structural markup present in the electronic source to advantage in producing high-quality, interactive audio renderings. This paper focuses on a specific aspect of the problem; namely that of flexibly rendering the extended document logical structure encapsulated in a (LA)TEX document.

One primary advantage of (LA)TEX is the flexibility it provides the author in defining logical structures that are specific to a particular document instance. In this sense, the class of logical structures that can be encapsulated in a (LA)TEX document is extensible. (LA)TEX macros allow an author to abstract away the layout details. At the same time, they provide a powerful mechanism for defining new constructs that are not already present in the document style (DTD in SGML parlance) in use. As a consequence, when introducing a new piece of mathematical notation, an author can first define a new (LA)TEX macro that produces a desired layout, and then use this newly defined construct throughout the document.

The flexibility of the (LA)TEX macro facility initially proved a major stumbling block in building a fully extensible audio rendering system. A system that attempts to produce aural renderings by *mapping* the builtin (LA)TEX commands to an equivalent aural representation faces the severe shortcoming of not being able to render documents that contain user-defined macros. At the same time, it is impossible to translate such user-defined (LA)TEX macros into a suitable aural representation. This is because TEX in its full glory is a Turing-complete programming language, and saying "we can translate a general TEX macro to audio" is equivalent to saying that "Given a TEX program, we can predict the result". Being able to achieve the above without actually running TEX on the program (document fragment) would amount to being able to solve the Halting problem!

In the rest of this paper, we describe the solution used in ASTER to circumvent this difficulty. The solution we used in fact turns the presence of user-definable (LA)TEX macros into an advantage. Such user-defined constructs allow ASTER to glean even more information about the document logical structure than would be possible if the document were

* *Now at:* Adobe Systems, Advanced Technology Group, 1585 Charleston Road, Mountain View, CA 94039-7900; Email: raman@adobe.com

encoded using only the built-in (LA)TEX operators; as a consequence, the audio renderings produced are also significantly better.

Document Models in ASTER

ASTER produces audio renderings by first extracting the document logical structure. In this model, all forms of rendering, i.e., visual, aural, etc. are regarded as a projection of the structure present in the information being conveyed onto the medium being used to communicate the information. Thus, typesetting a document requires visual formatting — projecting the information structure onto a two-dimensional visual tablet; aural rendering requires presenting the structure using various features of the auditory display.

The recognizer used in ASTER extracts logical structure present in documents encoded in the (LA)TEX family of languages. An important feature of this recognizer is that it works on the entire gamut of encodings, ranging from plain ASCII documents, i.e., no explicit markup, up to documents containing completely unambiguous encodings of the logical structure.

The basic document model used in ASTER is the attributed tree. Each hierarchical level of the document is modeled as a node in this tree. Each node can have content, children and attributes. Using object-oriented terminology, each different kind of node of the tree is called an *object* and represents a document element. Thus, "chapter", "section", "paragraph", and "sentence" are all objects. If a document contained five sections, its representation in ASTER would have five instances of object "section". This object-oriented terminology is used because ASTER actually uses CLOS objects in this fashion. The use of an object-oriented language was instrumental in allowing us to develop and implement the ideas in ASTER incrementally and effectively.

This attributed tree structure is augmented to represent mathematical content; we call this augmented representation the *quasi-prefix form*, (see figure 1 above). Expressions that are completely unambiguous, e.g., $x + y$, are captured in their prefix form. In addition to linearizing the underlying tree structure, mathematical notation uses *visual attributes* such as superscripts and subscripts, whose interpretation is context-dependent. We extend the prefix form to capture such visual attributes — hence the name *quasi*-prefix.

The next section describes how this model is extended to encapsulate the use of user-defined constructs in (LA)TEX.

Extended Logical Structure

The (LA)TEX facility can be used to extend the document logical structure by defining new constructs. Thus, an author preparing a manuscript on inference logic might define

```
\newcommand{\inference}[2]{{#1\over#2}}
```

and write

```
\inference{x}{y}
```

and use this construct throughout the document.

Notice that defining the `\inference` as shown above and using it to encode inference statements is distinct from and more powerful than just using the TEX builtin operator `\over` throughout the document. A commonly mentioned advantage in this context is that using the newly defined construct `\inference` will permit the author to easily change the notation used to denote *inference*. Notice, that this is in fact the same as saying that

> If distinct elements in a document instance are marked up using distinct constructs, then it is possible to recognize and process these elements in a multiplicity of ways.

In ASTER, the (LA)TEX facility of defining a second `\inference` macro that produces a different layout for *inference* can be generalized to the notion of different *audio renderings* for *inference*.

As explained above ("Document models"), ASTER achieves its aural renderings by building a rich internal representation of the document content. In this representation, each document element[1] E is represented by an instance of object O_E. ASTER provides a predefined type O_E for each of the builtin constructs in (LA)TEX. Thus, we could represent the use of `\inference` defined above in terms of object O_{over}. However, notice that this would mean losing valuable information. When building up the internal representation, the additional semantic information provided by the author's use of the `\inference` construct is very useful. In addition, expanding all (LA)TEX macros results in a pure layout representation, which is not appropriate for producing aural renderings (Raman, 1992). If we were to represent instances of `\inference` in terms of O_{over}, ASTER would be forced to render `\inference` the same as the `\over` construct. Though the author in this particular example may have chosen to use the same visual rendering for inferences that is normally used for fractions, the same may not carry over well to the aural domain.

[1] We use the term *element* loosely to mean a logical unit of the document.

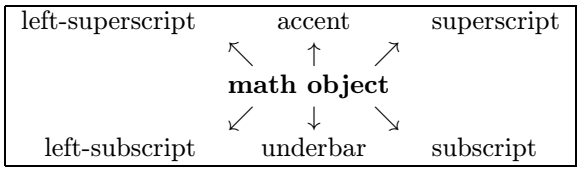

Figure 1: A math object with attributes. Each of the attributes themselves contain math objects.

Representing Extended Logical Structure AsTeR solves the problem of representing and rendering the extended logical structure arising from user-definable macros by considering each macro definition as introducing a new object type. Instances of a macro M, are represented by instances of object O_M. Thus, in the example shown above, the definition of the construct `\inference` introduces a new object type $O_{\text{inference}}$. The (LA)TeX macro consists of two parts; a declaration, and a series of TeX commands that the macro expands into. The macro expansion is nothing but a visual rendering rule that specifies how TeX should display instances of the object represented by the macro.

AsTeR provides an equivalent mechanism for extending the class of logical structures that are recognized. Once AsTeR has been told about a user-defined macro, audio rendering rules for the new object type introduced by this macro can be defined in AFL (Audio Formatting Language). Notice that such audio rendering rules have to be defined by the user, just as the (LA)TeX macro is defined by hand. It is not possible in general to translate the TeX macro into a set of audio rendering rules. This is because the TeX macro is capable of performing any arbitrary computation permitted by the operators present in the TeX language (Knuth, 1984) — a Turing-complete programming language.

Rendering Information

AsTeR renders information by applying *rendering rules* to the internal representation described above ("Document models"). The system of rendering rules used in AsTeR and the language in which they are written (AFL — Audio Formatting Language) are described in detail in (Raman, 1994). In a sense, AFL is to audio formatting as Postscript is to visual formatting, although AFL is a much smaller language.

Here, we show a small example of such a rendering rule for a user-defined macro. In the following, we use CLOS generic function read-aloud. For the present, let us assume that function read-aloud executes the necessary actions to render its argument.

After extending AsTeR to process the (LA)TeX macro `\inference` shown above ("Logical structure"), we can define

```
(defmethod read-aloud((inference inference))
  "Sample rendering for object inference."
  (read-aloud (argument 1 inference))
  (read-aloud "implies")
  (read-aloud (argument 2 inference)))
```

Given $\frac{A}{B}$, this produces "A implies B".

If we wished to produce a rendering that inverts the order in which the arguments to macro `\inference` are rendered, we would define:

```
(defmethod read-aloud((inference inference))
  "Renders inference with arguments reversed."
  (read-aloud "We know that ")
  (read-aloud (argument 2 inference))
  (read-aloud "because")
  (read-aloud (argument 1 inference)))
```

which produces "We know B because A".

Switching between these two rendering rules has the effect of inverting a proof-tree! Notice that writing a new rendering rule for an object O_E has the same effect as redefining the (LA)TeX macro that corresponds to E.

AsTeR makes it easy to write several rendering rules for the same object and also allows rendering rules to be partitioned into rendering *styles*. Such *styles* can be thought of as being analogous to LaTeX styles, but with one important difference. Due to the non-interactive nature of traditional paper documents, a paper is typically typeset in a given style. It is not possible for the reader to change the style in which the document is typeset. Typically, we do not feel the shortcoming of not being able to change the way a mathematical expression is rendered when reading a printed paper because the eye is capable of reading the various parts of an expression in any order that is convenient. However, when listening to an aural presentation, the listener does not have this flexibility. In other words, an active reader peruses a printed paper, a passive display, whereas in the case of audio, these roles are reversed — the aural display scrolls *actively* past a passive listener.

AsTeR overcomes these difficulties by being a fully interactive system. It is possible for the listener to interrupt the rendering, change the rendering style in use, and listen to the document. In an interactive session with AsTeR, switching between rendering styles (a collection of rendering rules for different objects) and invoking individual rendering rules can be done with a few keystrokes, making it easy for a listener to obtain many different views of a document. This facility enables *active* listening.

AsTeR derives its power from representing document content as objects and by allowing multiple user-defined rendering rules for individual object types. These rules can cause any number of audio events (ranging from speaking a simple phrase, to playing a digitized sound). The pitch of the voice, the physical head-size of the virtual speaker, the volume, and many other parameters can be changed by rendering rules, making it easy to create sound cues to help display structure. In fact, the design of AsTeR does not restrict the system to producing purely aural renderings; there is nothing to preclude us from defining renderings that produce truly multimodal output; i.e., renderings where the traditional visual rendering is augmented with aural feedback. We conjecture that such multimodal renderings may prove very useful for persons with learning impairments.

To give an example of a multimodal rendering, the logo for AsTeR is

and is produced by (LA)TEX macro `\asterlogo`. After appropriately extending AsTeR to recognize this macro, we can define an audio rendering rule for object *asterlogo* that produces a bark when rendering instances of this macro. Thus, the same piece of markup `\asterlogo` produces the picture of Aster[2] when rendered visually, and an appropriate sound[3] when rendered aurally.

This feature was exploited to advantage when producing the audio formatted version of the author's thesis. The dedication page of the thesis contains a large picture of Aster, and the audio formatted version[4] contains a verbal description of the picture, accompanied by the sound of Aster panting in the background. You can listen to this example on the WWW — visit the AsTeR home page by following the link to the AsTeR demonstration from my home page[5] and clicking on the picture of Aster.

Several ideas come together to make all this possible. First, logical structure is of paramount importance — not its display on any one particular medium. The more a document makes structure explicit, the better the document can be displayed on (projected onto) several different mediums.

Next, the use of (LA)TEX macros to encode structure makes it possible to have a system like AsTeR, in which the internal structure can be extended to fit a document. This allows the encoding of the structure in a flexible, uniform, and consistent representation such as an attributed tree, with the addition of the quasi-prefix form for dealing with mathematics.

Finally, providing different rendering rules and styles and a flexible way to switch among them makes it possible to obtain multiple views of a document in an interactive fashion.

Conclusion

The approach used in AsTeR to exploit the additional semantic information present in the electronic encoding in the form of user-defined constructs points to an important feature of markup systems like (LA)TEX that is currently missing to a certain extent in systems like SGML. When AsTeR as at its inception, I firmly believed that one should use a semantic-oriented DTD to encode a document in order to be able to produce high-quality audio renderings. I still believe this; however the work on AsTeR does point out one shortcoming with the fixed document DTD model. Given that mathematical and technical notation is being invented all the time, a fixed DTD forces the author to encode new constructs using *only* primitives that are provided by the DTD. As a consequence, authors end up using a presentation-oriented encoding even though the DTD in use is one that is semantically oriented.

To make this concrete, consider the case of the *inference* construct described above ("Logical structure"). If the document were being encoded using a fixed non-extensible DTD that only provides a *fraction* element, the author would be forced to encode *inference* using this element.

Since in general it is not possible to define an all-encompassing DTD that covers every possible kind of math notation (those currently known and those

[2] Aster is my guide-dog.

[3] The bark is that of a generic dog, Aster is too well trained to bark, and could not therefore be recorded.

[4] An audio formatted version of the thesis produced by AsTeR (about 6 hours) is being distributed by RFB — Recordings For The Blind — as the first fully computer-generated talking book.

[5] `http://www.research.digital.com/CRL/personal/raman/raman.html`

yet to be discovered) extensibility of the DTD as provided by (LA)TEX is of vital importance.

Another good example of this facility in (LA)TEX being put to good use is the HyperTEX system — an extension to TEX that allows the user to view his legacy (LA)TEX documents as online hypertext. Conceptually, we can think of `\ref` and `\label` as being object types; traditionally, these cause specific marks to appear on paper when rendered visually by TEX; to a system like HyperTEX these turn into *active* links that a user can follow interactively.

The ability to produce multiple renderings of the same object provided by ASTER was introduced in the context of aural presentations. However, such multiple presentations become equally relevant when interactively perusing online documents visually. For instance, when reading a document that presents a complex proof, a user may wish to have the same proof displayed as an outline in one window, and as a proof-tree in another (Lamport, 1993). In the case of paper documents, the user has to use her imagination to achieve such multiple views — though she is aided in this by the visual notation. In the interactive scenario presented by electronic documents, the previewer can provide some additional functionality to aid in this process.

References

D. E. Knuth. *The TEXbook*, volume A of *Computers and Typesetting*. Addison-Wesley, Reading, Massachusetts, 1984.

L. Lamport. "How to write a proof". Technical Report 94, DEC Systems Research Center, Palo Alto, CA, 1993. To appear in *American Mathematical Monthly*.

T. V. Raman. "An audio view of (LA)TEX documents". *TUGboat* **13**(3), 372–379, 1992.

T. V. Raman. *Audio System for Technical Readings*. Ph.D. thesis, Cornell University, 1994.