CMU-ITC-89-082

1st World Electronic Media Symposium
October 1989
Geneva

EXPERIENCES INTERCHANGING MULTIMEDIA DOCUMENTS USING ODA

Mark Sherman, J. Akkerhuis, A. Marks, J. Rosenberg, M. Zheng
Carnegie Mellon University
(USA)

IX . 5

Abstract

Through the EXPRES project, Carnegie Mellon University has been investigating the use of Office Document Architecture as an intermediate representation when translating documents from one multimedia system to another. This paper describes our environment and our needs for exchanging multimedia information. A brief sample of interchanged documents between Andrew, Diamond, Interleaf and troff are shown, followed by a discussion of different levels of document translations: imaging fidelity, structural fidelity and editing fidelity. We conclude that ODA is useful basis for promoting multimedia interoperability, but that further work is needed to maintain editing fidelity.

1. Introduction

Multimedia systems are becoming increasingly popular as ways to produce documents and exchange electronic mail containing multifont text, raster images, geometric graphics, voice, and other media. Unfortunately, different systems use different external (file) formats which makes interchange of multimedia electronic information difficult. The exchange of images alone, such as by fax transmission, is only a short term solution. Users need to manipulate the multimedia information on the receiving system as well as image it. Therefore, some mechanism is needed to promote the exchange of media content, document structure and editing control of documents. One candidate is the use of a common, intermediate representation. In this paper, we discuss our need for exchanging multimedia information and how we used the ISO standard Office Document Architecture.

2. Multimedia Environment at Carnegie Mellon University

2.1. Andrew Project

Carnegie Mellon University (CMU) has been pursuing a large educational software project called the Andrew Project[1]. The Andrew project is a joint effort of IBM and CMU, and is designed to support multimedia document preparation, electronic messaging and educational software construction on the CMU campus. The Andrew project is based in the Information Technology Center, a research center within the School of Computer Science. Approximately 40 people work on the Andrew project.

The Andrew hardware environment currently consists of a large integrated campus network, with approximately 300 high-function workstations (IBM RT PCs), several hundred Macintoshes and IBM PCs, and a dozen PostScript printers. The system includes a distributed file system that provides the individual workstations with the appearance of a large, monolithic Unix file system.

2.2. Multimedia Facilities in Andrew

The primary application software provided by Andrew is the Andrew Toolkit[2] and an associated set of applications. The Andrew Toolkit is a subroutine library for high-function workstations that can be used by application programs to manipulate multimedia documents.

The Andrew Toolkit contains support for creating and modifying media objects, for displaying these objects on a screen and for producing hard copy. Currently, the Andrew Toolkit supports multifont text, hierarchical line drawings, equations, spread sheets, raster images, hypertext links and simple animated line drawings.

One of the great strengths of the Andrew Toolkit is its extensibility: new media types can be added by users without requiring recompilation or relinking of the subroutine library or applications. Instead, the object code for dealing with a multimedia object is loaded dynamically on demand.

The primary multimedia applications provided by Andrew are a multimedia editor, an interface to the C shell (allows cutting and pasting between windows), a graphical shell, a messaging system (mail and bulletin boards)[3], a help system, and a graphical console for monitoring the state of the workstation.

Because the messaging system is built using the Andrew Toolkit, it allows users to create, send and receive multimedia mail. Figure 1 shows a screen image of the message system running with a multimedia announcement from a bulletin board.

Currently the messaging system at Carnegie Mellon University supports several thousand daily users with access to nearly two thousand bulletin boards in addition to personal mail. The system receives a new message every 20 seconds and accumulates over 10 gigabytes of bulletin board messages each year. Although most mail is internal, a substantial fraction of the message flow travels between CMU and other sites. Therefore, the need for document exchange for the messaging system is clear.

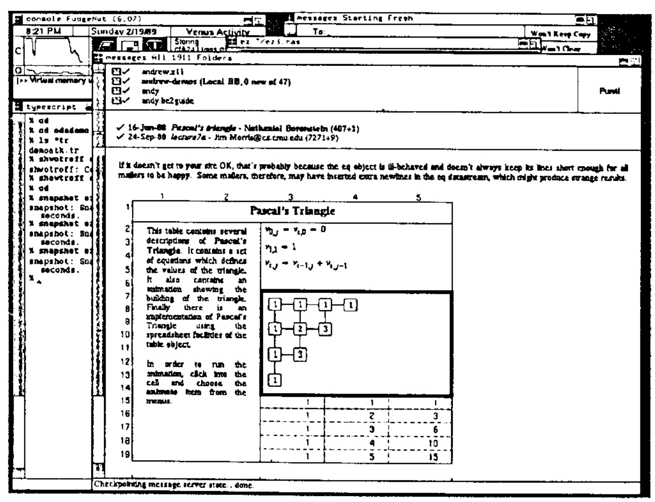

Figure 1: A Multimedia Mail Message

The Andrew Toolkit and messaging system has been widely distributed. There are over 100 sites in Europe, Asia and North America exchanging multimedia mail using this system. One source of users outside of CMU is the EXPRES project.

2.3. EXPRES Project

The EXPRES (*EXP*erimental *R*esearch in *E*lectronic *S*ubmission) project is sponsored by the US National Science Foundation (NSF) to promote the electronic interchange of multimedia documents among the scientific research community. The research community in the United States prepares much of its written material electronically. These papers, reports and proposals are multimedia documents, filled with tables, charts, equations and other information.

The community has also established a collection of computer networks that link together researchers in various fields. However, there is no convenient way for researchers to exchange their reports electronically. Different groups use different hardware and software. They resort to the least common denominators for exchanging information: simple character text and paper. With the increase of computer facilities, the NSF wants to encourage the use of multimedia electronic collaboration. Thus, the NSF started the EXPRES project.

The Andrew project at CMU is a natural environment in which to further the goals of EXPRES. With its installed base of multimedia editors and the message system, it had a substantial amount of scientific materials being generated and exchanged electronically. Of course, Andrew is not the only computer system used at CMU. Faculty and students use the native software on Macintoshes, IBM PCs, and high functional workstations to create multimedia documents. Thus, CMU has a substantial internal need for systems to interoperate as well.

The NSF funded two institutions to explore EXPRES: CMU and the Center for Information Technology Integration at the University of Michigan. These schools would help install multimedia systems in universities around the country, and collaborate in ways to exchange information among the different systems. In addition, the NSF encouraged other groups to collaborate in the EXPRES project. The McDonnell Douglas's Aerospace Information Systems Company, the US National Institute of Standards and Technology (formerly National Bureau of Standards) and the University College of London were among our informal collaborators. Each group had a substantial expertise in a different multimedia system. The goal was to exchange multimedia documents among the groups.

3. ODA as an Interchange Medium

Early in the EXPRES project, the collaborators decided that a common intermediate format was an appropriate mechanism for exchanging documents. The representation had to meet the following criteria:

1) It should be publicly available and controlled. No manufacturer's representation is acceptable.

2) It should have a predefined semantics for the both logical and layout descriptions of a document.

3) It has to represent multimedia documents, that is, documents with multifont text, raster graphics, geometric graphics, equations and tables.

4) It has to be extendable to include new media types, such as animations or video.

The Office Document Architecture (ODA) standard[4] is the only one that meets all of these criteria. The EXPRES project has focused on the use of ODA for exchanging processable multimedia documents between heterogeneous systems.

3.1. Capabilities of ODA

Document Structures

The Office Document Architecture (ODA) is an international standard designed to facilitate the interchange of multimedia documents. One of the important factors in our choice of ODA is that it includes a complete semantics for specifying the layout of a document. We feel that interchanging only the logical structure of a document would not be sufficient for effective interchange, but that users would insist on the ability to specify the appearance of the document.

ODA defines a document architecture, several content architectures and two datastream formats. The *document architecture* is the means by which the structure of a document, irrespective of its content, is represented. In general, an ODA document is represented using two sets of structures. The *logical structure* is based on the meaning of various divisions of the document. For example, the logical structure of a document might consist of chapters, sections and paragraphs. In the *layout structure*, the document is structured on the basis of presentation. For example, the layout structure of a document might consist of pages and, within the pages, frames and blocks that define headers, footers and paragraphs.

In addition, each structure may exist in two forms: *generic* and *specific*. A generic structure may be thought of as a template or macro that allows structure information to be collected and referenced. For example, the *generic logical structure* of a document might indicate that the document consists of a title, followed by one or more sections, followed by a set of references. Correspondingly, a *generic layout structure* for the same document might indicate that the title is a block that appears two inches from the top of the first page and is centered, and each footer contains a right justified page number.

If the generic structures of a document can be thought of as macros, then the specific structures represent invocations of those macros. The *specific logical structure* is, thus, the actual structure of a document. For example, the specific logical structure might show that a particular document consists of a title, five sections and a set of seven references. There is a *specific layout structure*, corresponding to the generic layout structure, but it is used only for the representation of a final form document (one that may be imaged). Since we are concerned only with editable documents, our investigations do not require any specific layout structures.

Medium Structures

Each medium in a document is described by a *content architecture*. Each content architecture defines its own internal structure, which may consist of logical and layout structures. There are currently three content architectures defined within ODA. *Character content architecture* defines the presentation and processing of characters and allows the specification of graphic character sets, multiple fonts, ligatures and formatting directives such as indentation and justification. *Raster graphics content architecture* defines pictorial information represented by an array of picture elements. *Geometric graphics content architecture* defines representations of picture description information such as arcs and lines.

In addition to the three predefined content architectures, one can create *private content architectures*. A private content architecture can be used to extend the media available in the ODA standard. This is important to us since our constituency uses equations and tables heavily.

Datastreams

A *datastream* is an out-of-memory representation for a document that is suitable for storage in a file or transmission over a network. The ODA standard defines two datastream formats. The binary format is known as the Office Document Interchange Format (ODIF). It is an ASN.1 encoding of an ODA document. The other datastream representation, the Office Document Language (ODL), is a clear text representation that defines a tag set that conforms to the Standard Generalized Markup Language (SGML) standard. However, this does not imply that there is a direct relationship between an ODA document and the equivalent document marked up using SGML.

Although the binary ODIF representation of a document is cryptic and unreadable by a human, it is also much easier to parse and unparse than ODL. Most ODA implementations of which we were aware are using ODIF and not ODL.

Attributes

The logical and layout structures of documents are represented in ODA as graphs, the nodes of which are known as *constituents*. Each constituent has a set of *attribute-value* pairs. Attributes have values that control the presentation and layout of the document. For example, the value of the attribute "Separation" at a constituent will control the distance between blocks of text when the document is displayed or imaged.

Figures 2 and 3 provide the abstract and imaged interpretations of a two page ODA document. In figure 2 only the specific logical and specific layout structures are used.

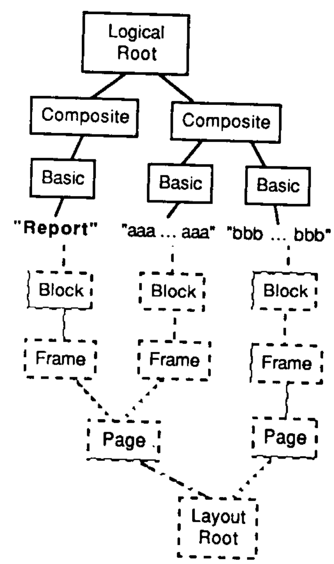

Figure 2: An Abstract ODA Document

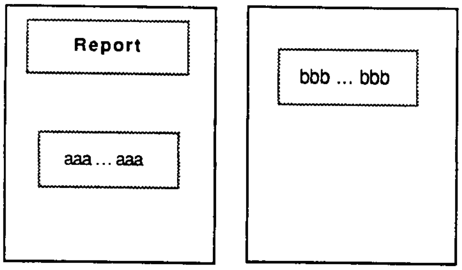

Figure 3: Imaged ODA Document

3.2. Implementing ODA

ODA is a rather large and complex standard. The main body of the standard is about 600 pages, and its references to other standards encompass about another 600 pages. To implement properly the various data and algorithms specified by the standard, all applications that process ODA documents must address a common set of issues:

1) Defining data types for representing the abstract description (constituents and attribute-value pairs) of an ODA document and routines for manipulating this description;

2) Routines for converting between the internal representation of an ODA document and a datastream;

3) Routines for performing high-level actions on the document as defined by ODA (for example, ODA defines a complex inheritance scheme for determining default attribute values at specific points in the structure).

To provide an environment for investigating ODA, we have constructed a tool kit that addresses the three concerns above. The ODA Tool Kit[5] provides developers with a ready environment in which to explore ODA applications.

The tool kit is a C subroutine library that has been designed to be extremely portable. In particular, we expect that the Tool Kit can be installed with a small amount of effort on virtually any machine-operating system combination provided that a viable C compiler is available. The Tool Kit currently runs on a variety of BSD- and System V-based Unix systems, DEC's VMS operating system, Microsoft's MS-DOS system and the Macintosh's MPW development system. We and others have used the tool kit to build translators that convert between native multimedia system formats and the ODA format.

Besides saving the time required for each group to write code to support ODA, use of the Tool Kit had some additional benefits. Using common code minimized the opportunities for each group to assign different interpretations to parts of ODA. While using different implementations may be a good way to detect incorrect interpretations, the time frame in which we had to work and the complexity of ODA made the use of common code imperative. In addition, using the Tool Kit in the early stages of its development ensured that the same functionality was available to all translators. This allowed us to begin interchanging documents at an early stage without concern about a mismatch in the degree of implementation of each project.

4. Experience with Translators

4.1. Description of Translators

The various groups started on a large collection of translators. A total of six translators were written using the CMU ODA Tool Kit. Converters to ODA were written for

1) Andrew Toolkit
2) Diamond[6]

Converters from ODA were written for

1) Andrew Toolkit
2) Diamond
3) Interleaf
4) troff

The translators varied in size from about one thousand lines of C code to about ten thousand lines. The variation was due in part to the number of features that the translator writer wanted to preserve and in part to the amount of support available for the native format. The translators required between two and five man-months each. By comparison, the common Tool Kit that they used contains about 80,000 lines of C code and required about 20 man-months. We believe that the common set of support libraries significantly reduced the amount of time needed to build a translator.

4.2. Scope of Translators

The goal of the EXPRES experiment was to see whether ODA was a viable medium for exchanging multimedia documents. Therefore, a specific set of features were selected for translation:

1) Font information, including family, face codes and size;
2) Character adjustments (superscript, subscript, underlining);
3) Paragraph indentation and margins;
4) Paragraph justification/alignment (centering, right justified, left justified);
5) Bitmapped raster images;
6) Style sheet information;
7) Document structure.

These features were selected to provide a minimally useful set of exchanged information that also would exercise the translators' abilities to exchange editable documents. Other features would have been interesting as an investigation of ODA's ability to exchange information, such as tables of contents, but the native systems being used and the time limitations prohibited investigation of those features.

4.3. Example Translations

We assembled a variety of hardware and multimedia systems to show how multimedia documents could be exchanged[7]. A screen snapshot from the Andrew system of a typical document is shown in figure 4.

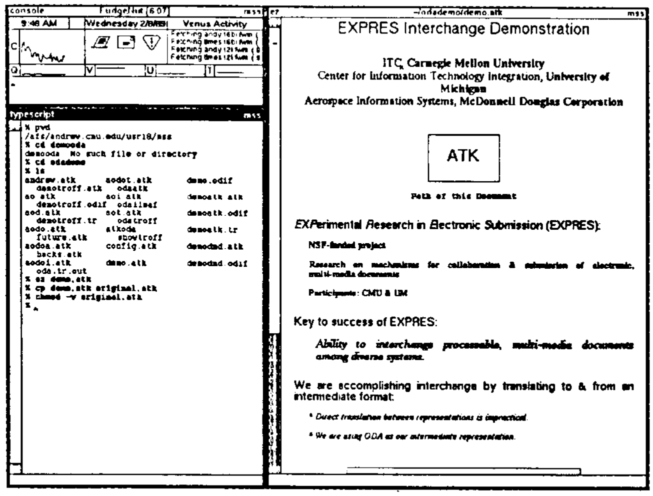

Figure 4: Sample Multimedia Document on Andrew

This document illustrates the features listed previously: a variety of fonts, margins, indentations, alignments and some raster images (only one raster showing the "ATK" path is visible in figure 4). The effects of translating into ODA and then into the Diamond format are illustrated by the snapshot from the Diamond system in figure 5.

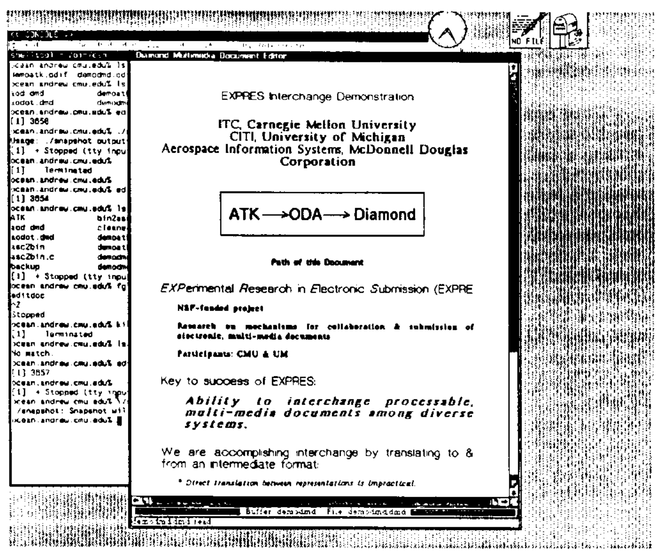

Figure 5: Translated Document on Diamond

The document was also translated from ODA into Interleaf format. The result is shown in figure 6.

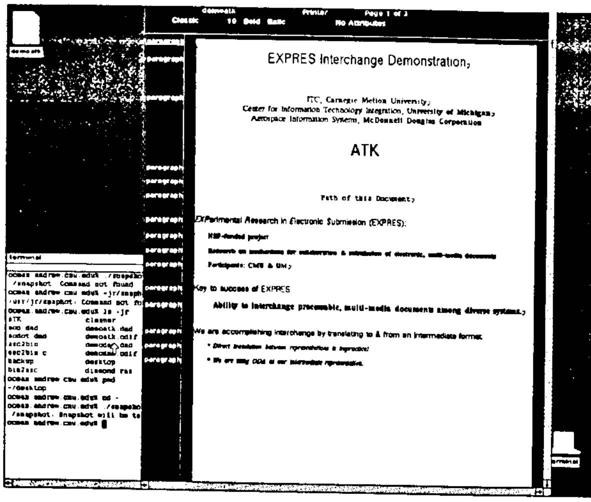

Figure 6: Translated Document on Interleaf

A fourth translation that was demonstrated converted the document from ODA into troff. The image that results from running this translator and troff on the NeXT computer is shown in figure 7.

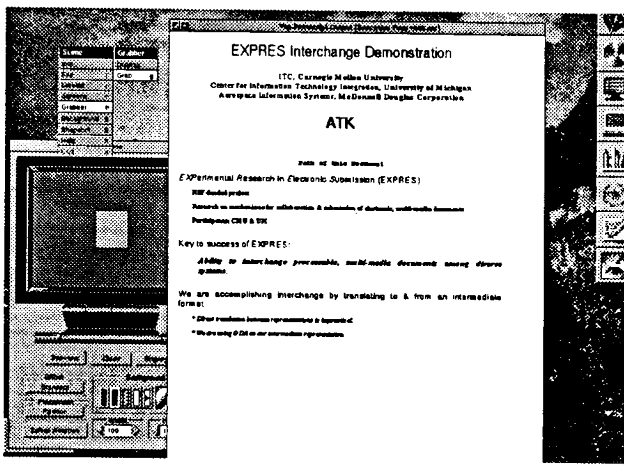

Figure 7: Translated Document run through troff

A close examination of these snapshots reveals a number of differences between the various systems. These differences can be grouped into two categories: missing functionality and detail mismatch.

When translating from one system to another using ODA, there are three possible sources of missing functionality. First, either of the native systems could be missing a feature. For example, some of the snapshots illustrate a font being substituted for a missing font. Second, the intermediate form, i.e., ODA, could be missing a feature. For example, ODA has no way to describe the relative margins that are specified by Andrew, e.g., move the margin in by another 1 cm. Since this omission reduces only editing fidelity, its effects are visible only when changes are made to a document. The appearance of margins in the figures, i.e., imaging fidelity, is unaffected. Third, the document model specified by a document application profile (DAP) could omit some feature, such as font information. A DAP is a sanctioned subset of ODA that defines document structures, such as a chapter or section. We were using the largest subset available, which in the United States is called the NIST Implementors Agreement[8] (and is aligned with the European Q/113 from EWOS). The NIST DAP does allow font information to be specified, but many other DAPs do not.

A detail mismatch occurs when the document models being converted include a concept but have slightly different interpretations of that concept. For example, systems differ in their interpretation and measurement of "line spacing". One detail mismatch illustrated several times in the snapshots is display artifacts. Some systems place boxes around raster images, some do not. Some systems add extra white space to the left margin during screen display, others do not. Although the information being specified by the systems is being faithfully translated into and from ODA, the appearance on the screen differs.

5. Nature of Translating Multimedia Documents

These discrepancies led us to consider the quality of the translations and what might be achieved. Based on our experiences building translators, we believe that there are several levels of fidelity that one might strive for when using ODA for multimedia document interchange: imaging fidelity, structural fidelity and editing fidelity.

5.1. Imaging Fidelity

Imagining fidelity refers to the way that a document is imaged, either on a screen or paper. The use of ODA does not provide perfect imaging fidelity, although the specific layout structure is intended to approximate it. The problem is that various parts of ODA allow for implementation-specific interpretations of imaging rules. Different systems using different font tables, for example, could place line breaks (and hence page breaks) in different locations. We did not attempt imaging fidelity for our translations. Our concern was focused on editable documents, and any editing change would alter the image. Therefore spending effort on imaging fidelity seemed futile when the recipient was intended to change the image anyway. We believe that there are times when imaging fidelity is important, but we also believe that a page description language, such as PostScript, provides an appropriate vehicle for interchange.

5.2. Structural Fidelity

Structural fidelity refers to the structure of a document as it appears to the user of an editing system. For example, documents can be collections of sections and chapters. Some document processing systems support manipulation of sections and chapters. Therefore, any translation scheme should be able to communicate the structural information in a document. The ODA standard provides relatively little to assist with interchange of structure. Although the standard does provide descriptions of abstract logical structure, there is no distinction between, say, a chapter, section, chapter, footnote or reference. These distinctions are made through the use of a DAP, discussed earlier. Therefore, successful interchange of document structure using ODA requires the selection of an appropriate DAP.

5.3. Editing Fidelity

Editing fidelity refers to the way that a document can be edited on a system. Our primary concern with editing fidelity centered around the use of style sheets (sometimes called property sheets, font deltas or simply styles). In order for a document to edited in a consistent way as it is moved from system to system, the logical editing operations defined by the document's style sheets must also be translated and interchanged. Unfortunately, the style model provided in ODA is quite simple compared with the style sheet systems of the editors we were using. Therefore, we had to invent encodings of style information within ODA so that the information could be extracted. Although we were partially successful, we believe that a great deal more work is needed in this area.

6. Conclusions

Our experiences in translating multimedia documents between ODA and other formats have been positive. We aimed for maintaining the editing fidelity of documents, and we believe that we achieved most of our goals. However, we also feel that attempts at concentrating on imaging fidelity can hurt editing fidelity. Although keeping the exact appearance of a document as it moves from system to system may be useful, a document that is to be processed must also contain the editing and structural information that was used to create it. We believe that this higher level of fidelity is needed for widespread interoperability of diverse multimedia systems.

7. Acknowledgements

This material is based on work funded by the National Science Foundation's EXPRES Project under grant number ASC-8617695. Any opinions, findings and conclusions or recommendations in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the US Government.

References

(1) The Andrew Project: Selected Papers from the USENIX Technical Conference, Technical Report, Information Technology Center, Carnegie Mellon University, April 1988.

(2) Andrew J. Palay, Wilfred J. Hansen, Mark Sherman, Maria G. Wadlow, Thomas P. Neuendorffer, Zalman Stern, Miles Bader and Thom Peters, "The Andrew Toolkit - An Overview", Proceedings of the USENIX Winter Conference, February 9-12, 1988, pg. 9-21.

(3) Nathaniel Borenstein, Craig Everhart, Jonathan Rosenberg and Adam Stoller, "A Multi-media Message System for Andrew", Proceedings of the USENIX Winter Conference, February 9-12, 1988, pg. 37-42.

(4) International Organization for Standardization (ISO), (ISO 8613) Office Document Architecture (ODA) and Interchange Format, June 1988.

(5) Jonathan Rosenberg, Ann Marks, Mark Sherman, Paul Crumley and Maria Wadlow, The CMU ODA Tool Kit: Site Installation Guide & Application Programmer's Interface, Technical Report CMU-ITC-071, Information Technology Center, Carnegie Mellon University, March 1988.

(6) Robert H. Thomas, Harry C. Forsdick, Terrence R. Crowley, Richard W. Schaaf, Raymond S. Tomlinson, Virginia M. Travers, "Diamond: A Multimedia Message System Built on a Distributed Architecture", IEEE Computer, Vol. 18, No. 12, December 1985, pg. 65-78.

(7) Jonathan Rosenberg, Mark S. Sherman, Ann Marks and Frank Giuffrida, "Translating Among Processable Multi-media Document Formats Using ODA," Proceedings of the ACM Conference on Document Processing Systems, December 5-9, 1988, Sante Fe, New Mexico, ACM, New York, pg. 61-70.

(8) National Institute of Standards and Technology, Stable Implementation Agreements for Open Systems Interconnection Protocols, Version 2, Edition 1, December 1988, NIST Special Publication 500-162, US Department of Commerce, pg. 16-1 to 16-149.

Biographies

Dr. Mark Sherman is a system designer of the Information Technology Center at Carnegie Mellon University. He graduated from the Massachusetts Institute of Technology with S.B. degrees in 1977 and from the Carnegie Mellon University with a Ph.D. in 1983. Mr. Jaap Akkerhuis is a system designer of the Information Technology Center at Carnegie Mellon University. He graduated from the Hogere Technische School with M.Ing. degree in 1975. Dr. Ann Marks is a system designer of the Information Technology Center at Carnegie Mellon University. She graduated from the Cornell University with a BS degree in 1976, a M.Eng. (Elec.) degree in 1977 and a Ph.D. in 1980. Dr. Jonathan Rosenberg is associate director of the Information Technology Center at Carnegie Mellon University. He graduated from the University of Maryland with a B.S. degree in 1977 and from the Carnegie Mellon University with a Ph.D. in 1983. Mr. Michael Zheng was a visiting systems programmer of the Information Technology Center at Carnegie Mellon University. He graduated from the Tsinghua University with a B.S. degree in 1987 and from the Marquette University with a M.S. in 1989. The authors are members of the CMU EXPRES project investigating the use of ODA for exchanging multimedia documents. Their book on ODA and multimedia document translation will be available in early 1990.