The Open Document Architecture: From standardization to the market

by H. Fanderl
K. Fischer
J. Kämper

The Open Document Architecture (ODA) was developed in the mid-1980s by several standardization bodies. It is now a stable set of international standards for the interchange of compound documents consisting of text, image, and graphic content. Since 1985 the standardization process has been accompanied by European industrial cooperation projects in order to get early experience with the standard and to develop technologies implementing the standard. IBM's European Networking Center has participated in the projects and has prototyped enhancements to OfficeVision™ platforms to allow the interchange of ODA documents between OfficeVision applications and applications running on other vendor platforms. In this paper, the ODA technology is described, experiences of interworking in heterogeneous environments are given, and the role of cooperating with project partners is outlined.

Computer networks are used more and more to support the collaboration of geographically dispersed professional workers. In IBM, as in other large enterprises, many employees use electronic mail, access forums and archives, and interchange electronic documents in their day-to-day work. Increasingly, this kind of communication is being used not only inside enterprises but also in interenterprise relationships. It has already been in use for many years in the academic community.

Along with becoming more widespread, computer-supported collaboration is applied to increasingly complex situations, for example, in publishing and engineering applications. In order to cope with both the widespread nature of the collaboration and the complexity involved, common architectures are needed in two areas:

- Communication protocols: Here we find IBM's Systems Network Architecture (SNA) and the nonproprietary architectures, Transmission Control Protocol/Internet Protocol (TCP/IP) and Open Systems Interconnection (OSI), established in the market.
- Information architectures: Here we find architectures for text established. The character-encoding specifications EBCDIC and ASCII suffice for normal notes in using electronic mail. IBM's Revisable Form Text Document Content Architecture (RFT:DCA) is widely accepted in office applications.

To capture the functionality of advanced applications that require the integration of different content types such as graphics and of advanced structuring concepts such as multiple columns and different layout streams, the Open Document

Architecture (ODA) was developed. It is an internationally standardized information architecture for compound documents consisting of text, images, and geometric graphics. ODA provides a common architecture for sharing information when collaborating via computer networks. The standard is part of IBM's strategy for the interchange of revisable documents in the office arena.

To encourage the use of ODA, the Commission of the European Communities launched in parallel to the finalization of the standard a series of cooperation projects named Piloting the Open Document Architecture (PODA) under the umbrella of the ESPRIT (European Strategic Programme for Research and Development in Information Technology) program. As a first objective, the projects jointly developed technology for building ODA support for existing applications. The projects have focused on establishing and publicly demonstrating interoperability of existing office solutions. The development of ODA support for the different products and the interworking tests among the different vendors have contributed to further development of the standard and have made possible early discussion of ODA with the user community.

The IBM European Networking Center (ENC) in Heidelberg, Germany, participated in two of the projects, shared a common implementation technique with project partners, developed a prototype that is based on the OfficeVision* family of products, and demonstrated this prototype in several public interworking trials. Results of this work are described in this paper.

An investigation of the requirement toward open document architectures is given in the first section. The ODA standard itself and the functional profiles that define the interworking in a heterogeneous environment are described in the subsequent section. The third section describes the environment and the main goals of the PODA projects. The method to implement ODA products, the approach to achieve interworking, and the OfficeVision ODA prototype are described last.

The need for an open document architecture

In recent years there has been a rapid development of computer-based systems that support communication among people through the capability of the systems to exchange electronic information of different types. Besides the actual exchange of information, these systems also support the storage, processing, and presentation of the information.

In the context of computer communication, a document can be defined as a structured amount of information that is meant for human perception and can be interchanged among systems as a unit (compare with Reference 1). This paper may be considered as an example of this notion of a document. It consists of text and figures. These content portions are arranged on pages. The actual arrangement of content corresponds to structure information that defines the document layout. Besides the layout information, the paper contains information that defines its logical structure. It is divided into segments that are marked by headings, and it contains notes and references that are numbered. Thus, it is a structured amount of information, and, of course, this information is meant for human perception. During its preparation the paper has been exchanged in its source format between the authors and the editorial staff by means of a network.

A document is a structured amount of information intended for human perception and can be interchanged among systems as a unit.

One of the key reasons for introducing the notion of a document is the interchange of compound entities of information among different systems. In order to limit the scope, only information that can be interchanged as a unit among systems is considered to be a document. Thus, for example, the data streams exchanged among systems during a telephone conference over a computer network are not considered to be a document.

The notion of a document as introduced above applies to many application areas such as editing,

computer-aided design, computer-aided publishing, and tutoring. It is not restricted to types of information that can be presented on paper such as text or graphics; documents may also contain information types that need to be presented in real time, such as audio and video.

A document architecture defines the concepts for integrating content portions of different information types with control information for the processing of that information into one entity, namely a *document*.

In order to allow documents to be interchanged among systems as a unit, document architectures comprise a language for the representation of structural information and content portions in a sequential data stream. Each system that shares a document with other systems in a computer network can parse the data stream and make the structural information and the content portions accessible to local applications. Local processing of a document often means editing, displaying, or filing. When sharing the document among different systems, the expectation is that such processing is performed in a consistent way on all systems independent of their local architecture. When sharing a document among different authors, the expectation is that the logical structure of the document and the editing controls such as margin specifications or font specifications are processable on all systems in a consistent way. When distributing a document to different readers, the expectation is that the document is displayed on all systems in a consistent way. When filing a document in order to allow others to retrieve it, the expectation is that all systems share a set of attributes in order to retrieve the document and, of course, to also share a language for the sequential representation of the document. A document architecture therefore comprises a model for processing documents. Such a model is not meant to standardize the local applications. It is a means to specify how the constituents of a document are supposed to be processed by the applications. For example, a model for document formatting defines how attributes such as line spacing or automatic footnote numbering are to be interpreted by a document formatter.

The notion of openness in open document architecture can be interpreted in a variety of ways. In this paper it means that the architecture is owned by the international standards bodies and thus its definition and further development is under public control.

In the following subsections, we investigate what major requirements should be fulfilled by document architectures. For this discussion three scenarios—document interchange, document storing, and document synthesis—are considered in more detail.

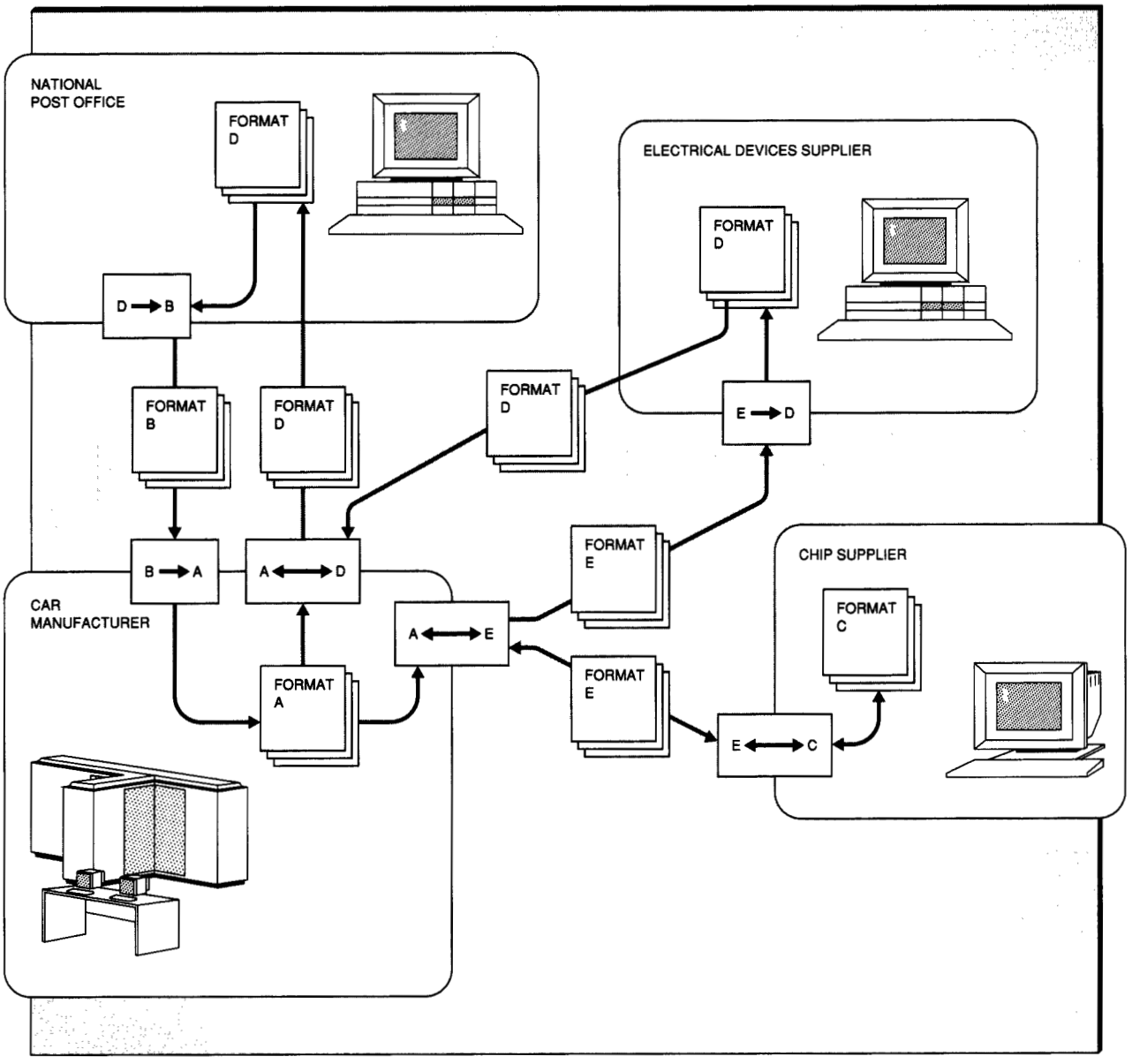
Open document interchange. The scenario in Figure 1 shows the interchange of documents among a motor-car manufacturer, the national post office, and two suppliers. The interchange is used in developing an offering, or proposal, to supply vans for the post office. During this business process many documents must be exchanged, e.g., meeting minutes, draft and final versions of the proposal, and technical design documents. After being exchanged, the documents are handled by the receivers in different ways, for instance, a receiver may want to present the document on a printer or a screen, forward it to some colleague, or change the document content using an editor.

In such scenarios, mainly proprietary document architectures are being used today to exchange documents between applications on different systems. Well-known examples of such architectures are IBM's RFT:DCA[2,3] format, the RTF (Rich Text Format) architecture from Microsoft Corporation, and the CDA** (Compound Document Architecture) architecture from Digital Equipment Corporation (DEC**).

These proprietary architectures have been made public. Thus, all software suppliers can implement applications that support these architectures. For example, the RFT:DCA data stream format is not only supported by IBM's word processors, but also by many applications of various vendors. Moreover, gateways have been developed that translate document descriptions between the different proprietary architectures. In this way it is already possible to interchange documents between a variety of applications. However, in complex scenarios this approach is unsatisfactory for at least two reasons:

1. As indicated by Figure 1, the conversions will be performed by different tools. Due to the lack of commonly accepted rules for the design and the quality of document conversion tools, the results may be quite different. The motor-

Figure 1 Heterogeneous document interchange



car manufacturer of our example cannot exchange documents with all partners in a uniform way with the same interchange quality. Thus, the task of negotiating the proposal becomes difficult because the partners have widely varying views of the documents.

2. Normally quite a large number of different document formats are in the network. Today that means that either the sender or the receiver of a document has to decide which converter is to be used. If the sender does the conversion he or she has to know which type of document processor the receiver would use; this is something that the sender normally does not want to deal with. If the receiver does the conversion, he or she has at least to be able to

identify the interchange format with which a document complies. Of course, there are pragmatic ways to indicate the format. However, there is no standardized way, and therefore the receiver has to manually deal with this problem.

Both problems can be solved when a standardized document architecture and interchange format are used by all systems in the network. Applications convert document descriptions between this intermediate format and the local document formats. Figure 2 addresses the same application scenario as Figure 1, but it is based on use of a uniform interchange format.

Of course, this approach of having a standardized interchange format does not guarantee that different converters use this format in a consistent way. Interoperability problems with respect to document layout and document processing behavior remain possible. For this reason, agreements are required that concern the use of the intermediate format and the design of converters adapting local systems to this format. Figure 2 reflects this fact. The shaded area indicates that an open document architecture must describe an intermediate document format as well as have commonly agreed-upon rules for converter design and document interchange.

A new application can share the scenario of Figure 2 in providing support for only one document format by following the commonly agreed-upon rules for document interchange and without any additional negotiations with the partners of the scenario. Thus, this approach is *open* for new communication partners. As an additional advantage, the described approach also reduces the number of required converters.

Open document stores. Well-known examples of document stores are archives and public libraries. In the scenario of Figure 1, the motor-car manufacturer and the two suppliers could establish a common library for the technical documentation of all van attachments. Such a common document base would be beneficial in working out the proposal for the national post office.

Accessing remote document stores and sending documents to some partner in a network are different sides of the same coin. If someone sends a document, the communication process is activated by the originator of the information being mailed. If someone accesses remote document stores, the recipient of the documents triggers their exchange.

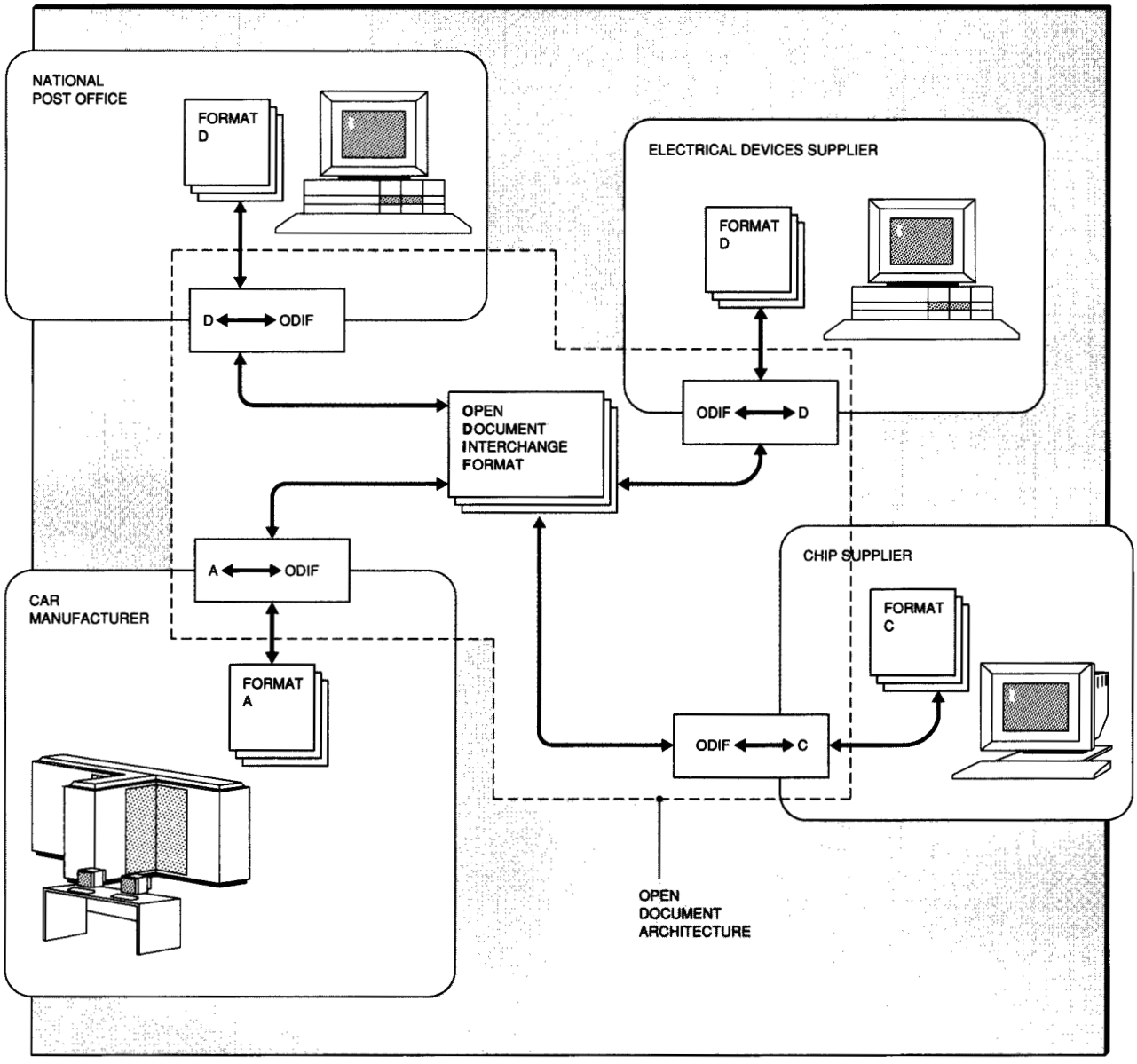
Involving document stores presents another reason for using standardized document architectures: the lifetime of documents. In the future, applications should be capable of handling documents that were created several years or even several decades ago.

In order to enable document interchange between applications of today and tomorrow, document architectures should fulfill two requirements. First, they should be highly stable. Since the formats of word processors tend to change with each new version of the software, they are normally not adequate for that application scenario. In a standardized architecture as in some proprietary ones, the stability is guaranteed through the objectives and stability of the architecture owner, namely the standardization body. Second, the architecture should be extensible for or adaptable to future applications. This flexibility can be achieved by defining general concepts in the architecture that hopefully are then applicable also to future applications. Tailoring for specific sets of applications can then be done through subsetting and through specifying how the concepts are to be applied to represent typical features of the applications.

Currently, existing document processing systems are being enhanced in many ways, e.g., multimedia and hypertext (compare, for example, Reference 4 and Reference 5). Without any framework, these different ways will result in advanced applications with no satisfactory capability to interchange information. Standardized document architectures aim at providing a framework for important developments in document processing applications as far as they are foreseeable today. Based on such a framework future versions of existing applications can improve their capability to interchange information with other applications. It is a key issue for open document stores as well as for *open* document interchange.

Open document synthesis. Besides being important in the interchange of information among different nodes in a network, document architectures are also important in integrating different applications on one system. Document processing applications are required to be capable of integrating content portions generated by other ap-

Figure 2 Open document interchange



plications into one document. As an example, consider a user preparing a business graphic from entries of a database and inserting this business graphic into a document. A document architecture supporting this integration is *open* with respect to input from different applications such as database systems and spreadsheet programs.

Different enhancements of document applications that will be developed in the near future result in new types of document content. Multimedia applications, for example, are characterized by the integration of continuous content types like audio and video. For this integration, new types of structure information, e.g., for syn-

chronization, are required. Hypertext applications allow the definition of arbitrary references between different parts of the same document or between parts of different documents. For many applications, especially in a networked environment, it will be advantageous to automatically follow links to information that is generated by or accessible through other applications. In order to support such function, standards for the representation of links and for denotation of parts of documents have to be defined in conjunction with document architectures. These mechanisms for referencing are another example of structure information that will be used by future document applications.

In the following section, the major concepts of the Open Document Architecture are introduced, and how these concepts may help to achieve the goals described in this section is discussed.

The Open Document Architecture

The need for an open document architecture has been recognized by the International Organization for Standardization (ISO) and the Comité Consultatif International Télégraphique et Téléphonique (International Telegraph and Telephone Consultative Committee, or CCITT) and has led to the definition of a set of international standards. Among those standards the Open Document Architecture (ODA)[1] and the Standard Generalized Markup Language (SGML)[6] are the most general ones in terms of the requirements that have been explained in the previous section.

ODA defines interchange formats, concepts to represent the structure of the information in a document, and the meaning of a set of formatting parameters. The architecture aims to allow so-called *blind document interchange*. This term means that a document can be interchanged among two systems such that revisability and layout are preserved just based on the knowledge that both systems comply to the international standards.

SGML is a general language that represents the structure and content of documents. It does not provide a definition of the meaning of formatting parameters. Therefore, it is not suited for blind document interchange to the same extent as ODA. However, since it is not limited to a certain set of formatting parameters, it can be used for the interchange of documents that require formatting or for other types of document processing that could not be covered with the fixed set of ODA attributes. For a comprehensive explanation of SGML see Goldfarb.[7]

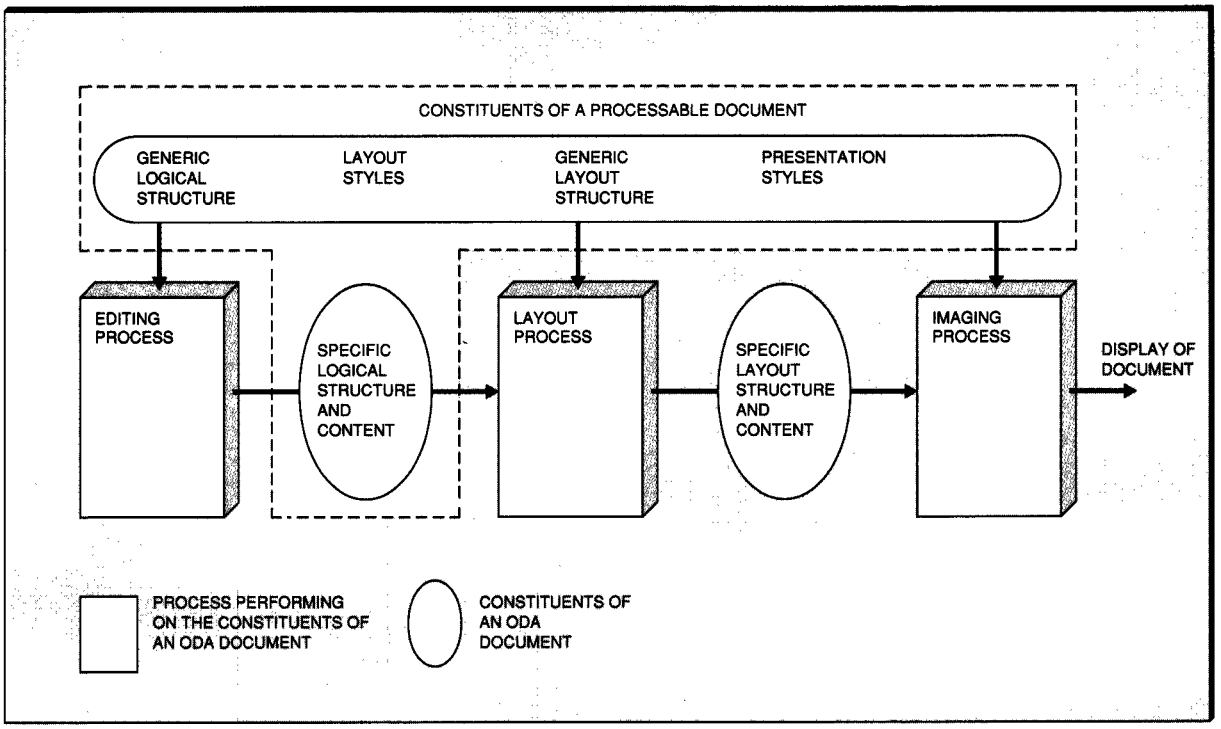
In this section, a brief overview of the main concepts of ODA is given, along with a discussion of how far ODA meets the requirements established in the previous section. For a detailed introduction of ODA concepts, see Rosenberg et al.[8] or Hunter et al.[9]

ODA processing model. ODA is essentially a model for formatting and presenting compound documents consisting of text portions, geometric graphics, and raster graphics. ODA distinguishes three so-called document architecture forms: formatted documents, processable documents, and formatted processable documents. Documents in any of those forms can be subject to document interchange. However, their purpose with regard to local processing is different. Processable documents are meant for processing by a human-controlled editing session or by automatic processes that change the content or structure of a document. The presentation of a processable document on different systems may yield different layouts because ODA does not specify all parameters of the formatting process. Formatted documents are meant for presentation. The interchange of formatted documents ensures that the same layout can be presented on all systems. Formatted processable documents preserve the exact layout and are to be processed in the same way as processable documents.

In addition to the content and the logical structure, processable documents may contain attributes that specify the formatting of the document. In order to define the meaning of those attributes, ODA establishes a document processing model. This model specifies how formatted documents may be generated from a processable document. In the sequel, the processing model given in Figure 3 is used to explain the different constituents of an ODA document. There are three document processing steps: the editing process, the layout process, and the presentation (imaging) process.

The main parts of a processable document deal with the document content, the logical document

Figure 3 The ODA processing model



structure, and rules for document layout. The logical structure describes how a document is divided into logical objects like chapters, sections, paragraphs, lists, footnotes, and so on. It consists of two substructures, a generic logical structure and a specific logical structure.

The generic logical structure represents document properties that are shared by a class of documents, and the specific logical structure represents parts of the document description that are specific to one document. For example, the generic logical structure for the document classified as a business letter may describe general properties such as its division into the following parts: addressor details, addressee details, subject, greeting, and body and the assignment of such formatting information as fonts to each of those parts. The specific structure comprises the actual content and structure of a letter that, for example, is interchanged between the motor-car manufacturer and the national post office in the scenario of Figure 1. In the document editing process the specific logical structure and the document content are generated or changed. In this process, the generic logical structure can have different roles. The structure could control the editing process in the sense that only specific structures complying with the generic structure are generated. Although this is what the ODA processing model implies, this possibility is not being used in most of today's applications of ODA. The other possibility is to represent in the generic structure fixed properties of the editor used to generate the document. This latter possibility is being used when existing word processor formats are converted to ODA data streams.

In addition to the logical structures and the document content, a processable document contains rules that direct the layout process. These rules are described by a generic layout structure and by layout and presentation styles. The generic layout structure defines properties of layout objects that are common to a class of documents, e.g., pages, header and footer areas on a page, and columns.

Layout styles aggregate information that controls how content is to be assigned to such objects during the layout process. For instance, a layout style may specify that all objects referencing this style have to be laid out on a new page. Presentation styles aggregate information that concerns the formatting of content, such as fonts and line spacing.

According to the ODA processing model, the layout process transforms a processable document into a formatted or a formatted processable document. It specifies what specific layout structures may be generated from the constituents of a processable document. Examples for constituents of the specific layout structure are page sets, pages, and frames.[10]

A formatted document can be presented on a device such as a printer or a display screen. According to the ODA processing model, this output is done by the document presentation process that is directed by the specific layout structure and, in some cases, by the generic layout structure and presentation styles.

A formatted processable document contains the specific layout structure in addition to the constituents of a processable document. If such a document is subject to an editing process, the layout process must be repeated in order to create a new specific layout structure before the presentation process can take place. If the same formatted processable document is edited with the layout process being repeated on different systems, the resulting layout structure will in general not be the same: The information comprising a processable document does not and is not meant to specify all aspects of the layout process.

In addition to the document constituents mentioned so far (specific and generic logical structure, specific and generic layout structure, layout styles, presentation styles, content portions), an ODA document includes a document profile. It contains information concerning the whole document. For instance, the document originator, the document creation date, and some copyright information can be specified by the profile. The profile attributes are primarily intended for applications that have functions for the administration, the exchange, and the archiving of documents.

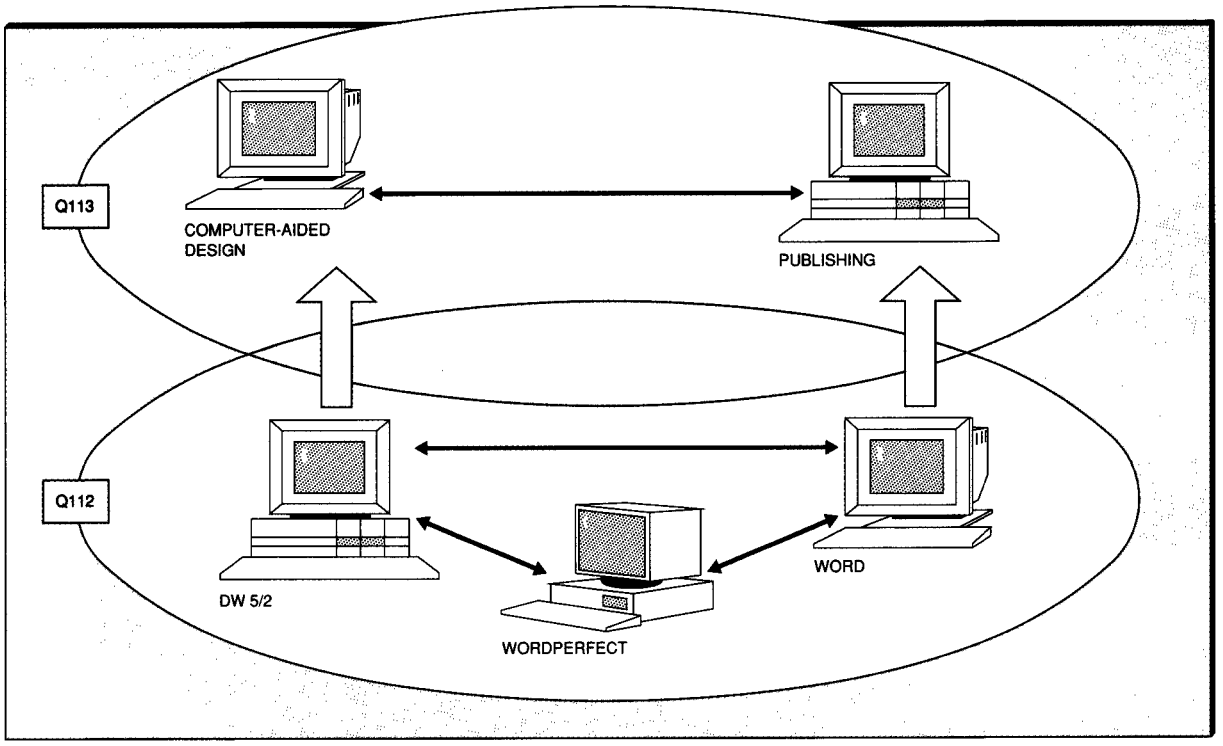
The following subsections discuss how ODA meets the requirements for an open document architecture as derived in the previous section.

Open document interchange. So far ODA has been described in terms of its constituents and their meaning for document processing applications. For interchanging the constituents of a document among systems, two different interchange formats can be used. The Open Document Interchange Format (ODIF) employs the Abstract Syntax Notation One (ASN.1)[11] international standard and the basic encoding rules for ASN.1[12] in order to define a bit stream representation of a document. The other format employs SGML. As mentioned before, SGML is a general language that represents structured information. In this case, it has been used to define a representation of the ODA constituents, which is named *Open Document Language* (ODL). In contrast to ODIF, an ODL data stream is human-readable because SGML is a "cleartext" markup language.

In the previous section we stated that an open document architecture should be open to represent features of future document processing systems. To achieve this goal, ODA does avoid standardizing features of applications in which changes may be expected. With regard to the layout structure, the assumption in ODA is that no essential new features are going to show up. Therefore, the standard introduces the usual concepts of page sets, pages, and frames. Of course, this binds the standard to a more or less paper-oriented output model. With regard to the logical structure, the ODA approach is to try to be open to as many potential new features of document processing applications as possible. The standard therefore defines structural components and attributes that are capable of representing a variety of features. For example, ODA does not define an attribute for automatic page numbering. For that feature it defines concepts that are also capable of representing footnote numbering, chapter numbering, etc.

Because of this richness and generality of concepts regarding the logical structure, on one side the set of features of applications that can be supported by the standard is not fixed, and on the other side a specific feature of an application can be represented using different concepts of ODA. Therefore, at least for processable and formatted processable documents, open document interchange is not possible based on the ODA standard alone.

Figure 4 Migration from Q112 to Q113 applications



To allow open document interchange, a set of ODA-based standards, so-called *Document Application Profiles* (DAPs), have been and are being issued. Each of the DAPs specifies open document interchange within a certain class of applications, through the definition of a set of features that are to be preserved with regard to document layout and processing behavior and its representation in terms of ODA constituents. Each DAP has a globally registered identifier, which is being interchanged as part of the document profile. It allows the receiving system to determine whether it is capable of processing the functionality of the ODA document. The concept of DAPs aims at providing a certain level of consistency in all compliant applications. In the scenario of Figure 2 this fact has been indicated by including the converters within the area of ODA.

Currently DAPs on three levels have been issued for different classes of applications:

- Level 1 addresses the functionality of revisable form teletext services, which would be offered by the public network operators. It enables the interchange of formatted as well as of processable documents. The content types are limited to text.
- Level 2 captures a kernel of the functionality of modern, workstation-based word processors such as Word** and WordPerfect** or the editors of the DisplayWrite* family. The content types include text, raster graphics, and geometric graphics.
- Level 3 addresses the capabilities of advanced document applications such as computer-aided publishing. It supports the same content types as Level 2; however, it allows more advanced formatting features.

The three levels are in a subset relationship in terms of functionality and in terms of allowable data streams. Therefore, in addition to allowing applications that belong to one level to interchange documents, the DAP hierarchy also allows documents to be transferred from applications of a lower level to the ones of a higher level (see Figure 4). In addition to this hierarchy of DAPs,

other DAPs, e.g., image-processing applications, are currently under development. If the interchange of documents from higher-level DAPs to lower-level DAPs or between different DAP hierarchies is required, the fact that all DAPs are based on the same document architecture makes the design of converters relatively easy.

The development of DAPs according to the above-mentioned levels of functionality has been carried forward by different organizations in Europe, North America, and Japan. The different developments have been harmonized and are currently being issued by ISO.[13–15] The European development has led to European Standards named Q111 and Q112[16,17] that have been in place since 1990. Q112 is the standard that is currently the basis of most of the commercially available implementations. For a more detailed introduction to the DAP hierarchy and its development, see Spiceley.[18]

Open document stores. The ODA interchange formats can also be used for the storage of documents. The information contained within the profile supports retrieval operations. For instance, the ODA document profile may contain key words, the document title, or the author's name. Retrieval systems can support the selection of documents within some document store on the basis of such attributes.

To allow applications to access documents in a remote document store, a document architecture, interchange formats, a model of the document store, and standardized access protocols are necessary. As discussed above, ODA provides the required document architecture as well as interchange formats. Access protocols and a sensible storage model are defined by another standard that is called *Document Filing and Retrieval* (DFR).[19] In the next section we describe a prototype implementation of a remote document store that is based on ODA and DFR.

Open document synthesis. The ODA standard is divided into parts that describe the actual document architecture and that specify content architectures. These parts are linked by an interface: the ODA layout process determines location and dimension of frames, and the content presentation processes specify how to lay out the document content within these frames.

Because of this interface concept, it is possible to include content portions that are meant for processing by special applications (e.g., spreadsheet programs or database applications) into an ODA document. New content architectures can be added to the architecture, provided that they fit the mentioned interface and can be referenced from the ODA document by a registered identifier. Also, work is underway within the standardization bodies to include new content architectures in the standard itself (see Bormann and Bormann[20]).

Currently, the ODA standard provides content architectures for text portions, geometric graphics, and raster graphics. For instance, the integration of business graphics would be possible by just adding a new content architecture for this content type. The creation and processing of business graphics should be described by the new content architecture; the integration of business graphics into documents can be done by the interface concept ODA provides today. To enhance ODA with respect to continuous media like audio and video is a more difficult task. For such enhancements, concepts for time synchronization would have to be included that affect the content architectures as well as the overall document architecture.

ODA-related ESPRIT projects

For the purpose of promoting the ODA standard as a widely accepted document architecture, the Commission of the European Communities supported a series of ODA-related projects in their ESPRIT program. Two of them will be addressed in more detail within this section: the PODA-2 and the PODA-SAX project. The main objective of both projects is to develop a base technology that allows ODA support to be built for existing and future office systems. Through participation in trade shows and other promotional activities the projects aimed to convince users that ODA is a workable solution for document interchange in heterogeneous environments. The difference between the projects is in the variety of applications that are subject to ODA-based interworking: PODA-2 has focused on the mailing of documents among different vendors' office system products; PODA-SAX also addresses accessing document stores, printing documents, and linking data from spreadsheet applications into documents.

Because the PODA-2 project is now finished, we are able to give an overview of the results of that

project. The PODA-SAX project is still running, thus a survey of the objectives of the project is given.

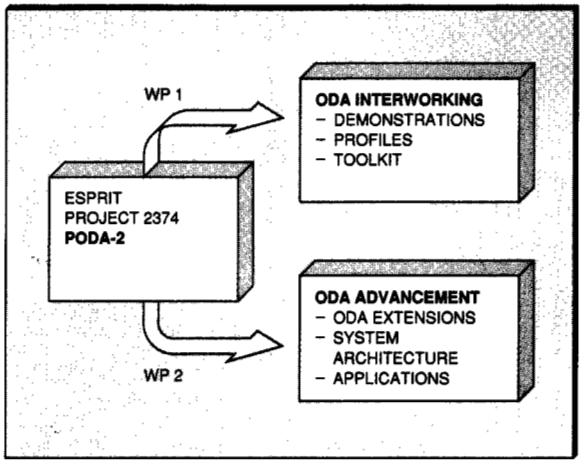
PODA-2 project. The PODA-2 project (Piloting the Open Document Architecture-2, ESPRIT Project 2374) started in January 1989 as the follow-on project to PODA (ESPRIT project 1024) and had a duration of 30 months.[21] The partners within PODA-2 were British Telecom, Bull SA, IBM, ICL PLC, Siemens-Nixdorf Informationssysteme AG, Alcatel TITN Answare, Océ, and, as an associated partner, the University College London.

Seven of these partners had already worked together in the PODA project with the major objective of evaluating the ODA standard in terms of implementation issues. At the beginning of the PODA project the standard was in the draft proposal phase at ISO. Valuable feedback from early prototypes was delivered from the PODA project members to standardization bodies for incorporation into the text of the final standard.

At the beginning of PODA-2, the ODA standard was stable and in the publication process of ISO. Therefore, the main emphasis of this project was the development of an ODA implementation technology and of prototypes that enabled interworking in a multivendor environment, as well as the advancement of the ODA standard itself. Advancing the standard and standard-based applications was done mainly through the development of *Document Application Profiles*, which are key items for the practical use of ODA (see the earlier section entitled "The Open Document Architecture") in terms of interoperability. It was also done by contributing to the standardization bodies those enhancements to ODA that enable additional document features such as data in documents and security mechanisms.

The project was divided into two separate fields of work called *work packages* (see Figure 5). Work Package 1 (WP 1) dealt with the interworking of ODA products and prototypes. These items were shown in public demonstrations during the project duration. The basis of most of the systems at the demonstrations was a set of tools developed by some partners and shared among nearly all of the PODA-2 partners. Such sharing of project results is a concept of the ESPRIT program, which fosters precompetitive development among industrial partners. In this case, the sharing of the

Figure 5 PODA-2 overview

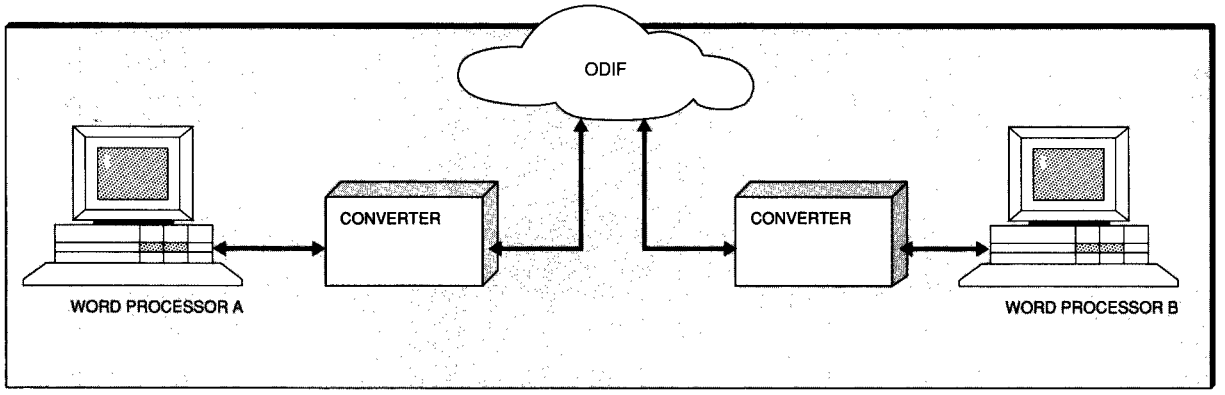


toolkit allowed a reduction in the effort of prototyping applications, and it eased the establishment of interoperability among the applications. As the basis for interoperability in the demonstrations, Document Application Profiles have been agreed on and developed. Work Package 2 dealt with extensions to the ODA standard carried out in close relationship with the standardization bodies. In this work package, an ODA system architecture was developed, defining the relationship between the ODA document structure and the processes, services, and applications using and handling ODA. A number of applications were designed and prototyped to implement this architecture, such as document-storing facilities and hypermedia document access.

In the following subsections we consider two of the major activities of the PODA-2 project in more detail.

PODA-2 interworking demonstrations. The focus of this activity of the project was the interchange of processable form documents among today's word processors using a mailing component suitable for interchanging ODA documents. When off-the-shelf word processors are used, such as Word, WordPerfect, or DisplayWrite, which employ internal document representations with different structural elements than those of ODA, a conversion process has to be carried out for sending and receiving documents as depicted in Figure

Figure 6 Document interchange scenario



6. In this figure, word processors A and B might employ different proprietary document architectures. For the document interchange from word processor A to B, the document is converted from the format of A to ODIF, transferred as ODIF data, and after being received at the system where word processor B is running, converted from ODIF to the document representation of B. The sending system has not had to care about what word processor (or document representation) is used at the receiving side. This method is denoted as blind document interchange. Word processors that are capable of processing ODA structures and attributes without conversion, so-called ODA native systems, have also been investigated.

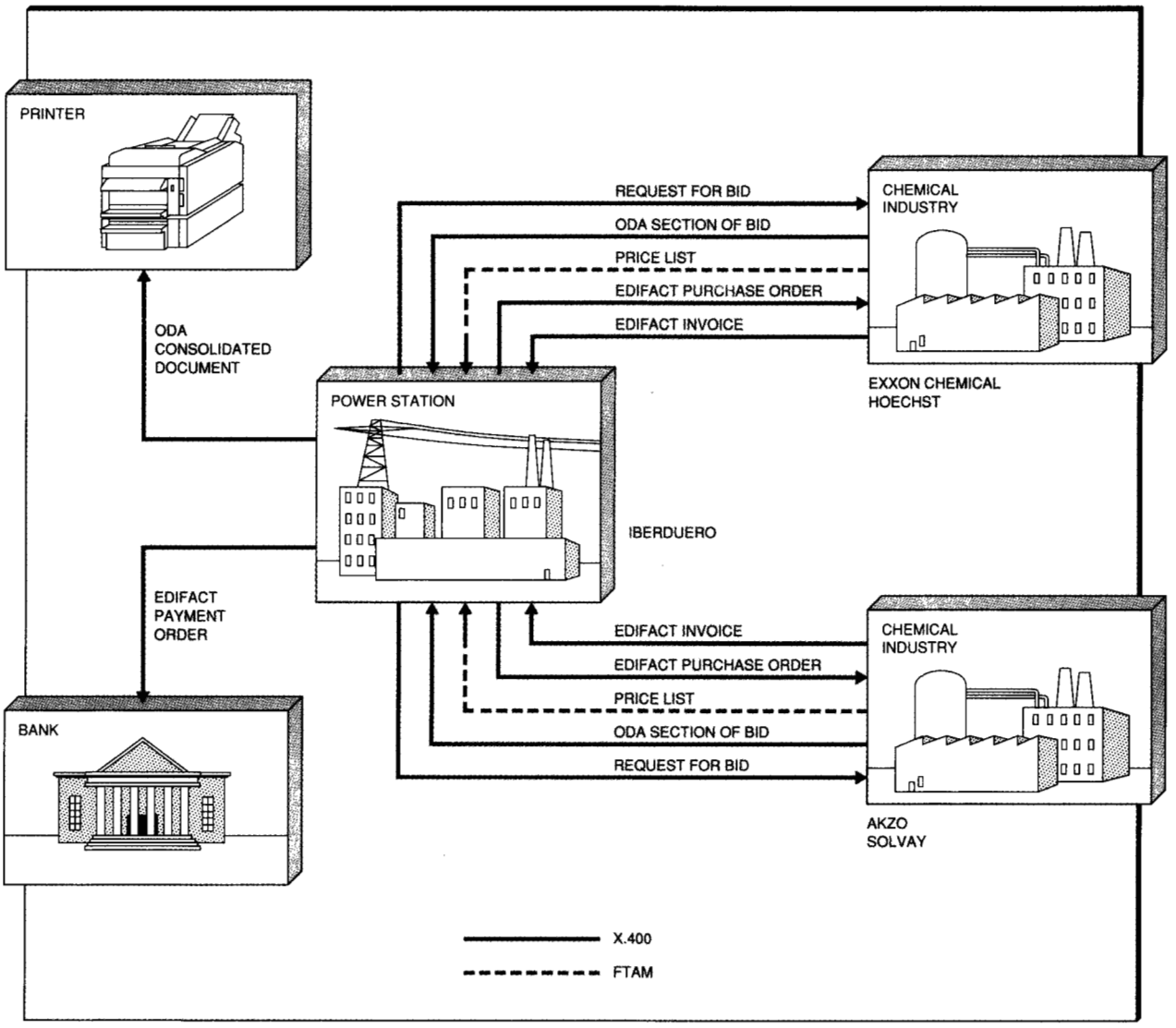
These conversion processes are mapping the specific document formats to and from the standardized interchange format. Although this process has to be different for each individual word processor format, a number of common parts are in this module, such as analyzing and producing ODIF streams or accessing specific ODA-defined document features. This commonality enables the reuse of software components and is addressed in more detail in the discussion of the PODA-2 toolkit development later in this section. The disadvantage of the converter approach is the inevitable loss of information resulting from the different formats employed at the conversion process. The advantage, however, is that the users can work with well-known editors and interchange documents using ODA.

In the demonstrations of the project, ODIF data streams were interchanged through an appropriate mailing system. These systems not only have to support the conveyance of ODIF, they also have to comply to an internationally agreed-upon standard in order to fit the requirements of open document interchange. The decision in the PODA-2 project was the adoption of the X.400 standard.[22] Mailing notes and messages in an open environment is today an integral part of many office systems. However, if the interchange of documents conforming to the ODA standard should become a feature of such office systems, an ODA converter and enhancements of the mailing components have to be integrated.

In order to demonstrate interoperability among ODA-based implementations from different vendors, the PODA-2 project participated in a series of public trade shows. The most important of them were the CeBIT '89 and '90 fairs in Hannover, Germany, and the ODA Symposium 1990 in Paris.

In CeBIT '90 the PODA-2 project demonstrated document interchange at the EurOSInet[23] booth. Besides the PODA-2 partners, Apple Computer, Inc., Digital Equipment Corporation, Philips, Rank Xerox, and Unisys participated in this demonstration. The main objective of this demonstration was to show that ODA, together with other open-system standards such as X.400, EDIFACT,[24] and FTAM[25] have the potential to bring real benefits to daily business work. Therefore, the partners demonstrated a business scenario to the visitors at the booth. They were playing roles interchanging notes and documents, producing and consolidating proposal documents, and exchanging price lists and purchase orders. The sce-

Figure 7 The Consortium Model



narios were designed to show ODA as a facility of open systems which is integrated with OSI facilities like file transfer or directory. The ODA systems at the fair included two editors with ODA as the internal document format, eleven converters, and two printing services. ODA support for well-known editors such as Word and DisplayWrite, as well as for familiar office systems such as ICL's Officepower** or IBM's OfficeVision, were shown.

The Consortium Model scenario (see Figure 7) will be explained in more detail. In this scenario, a group of chemical companies (Akzo, Exxon Chemical, Hoechst, and Solvay) jointly created a proposal to supply chemicals to the company Iberduero. All providers sent their input parts as ODA documents over X.400 to the partner who played the role of Iberduero. After converting these parts to a proprietary format, this partner consolidated them in a single document and sent

it to a printing service as an ODA document. Other OSI protocols were also in use, such as FTAM services to provide Iberduero with price lists and EDIFACT for handling purchase orders.

ODA documents that were interchanged within this scenario contained text portions, geometric graphics, and raster graphics. The documents were made up of running headers containing a company logo as a raster graphic and a footer area with automatically generated page numbers. The logical structure contained a hierarchy of passages, sections, and paragraphs as well as an automatic section-numbering scheme. The text portions were portrayed by the use of multiple fonts, including proportional ones. Other document features were also shown, such as tables in which different types of tabulation stops were in use.

A contribution to the development of a Document Application Profile was a major achievement of the project: Interchange of documents among different word processors in an open environment requires not only support by these systems, it also requires a common understanding of the standard by all participating system implementers. Therefore, a subset of the ODA standard has to be agreed upon in order to define restrictions to the concepts provided by the standard and to specify allowed ranges of values in the standard-defined attributes, such as page formats or character sets. The definition and execution of the interworking tests prior to the PODA-2 project demonstrations made major contributions to the development of the Q112 document application profile that was mentioned earlier. Actually, for each of the demonstrations, interworking agreements were formally defined, which later, together with other work, led to the definition of Q112.

Other topics not addressed in the standard in 1990 were defined as so-called EurOSInet agreements, such as the conveyance of ODIF streams in X.400 systems as well as the description of fonts in the documents. Such agreements enable early prototype experience before the appropriate standard has been adopted. The agreements have been made available to the standardization bodies. As an example, the method to convey ODA through X.400 has been adopted in international standards.

Extensive tests had been carried out between all PODA-2 partners. During those tests a series of test documents were created and a test procedure pursued. It was agreed to restrict the interchange of documents to processable ones; therefore, it was more important to preserve the logical document content and the editing behavior (e.g., automatic footnote numbering) than to preserve the exact layout. However, in accepting ODA, preservation of the image of the document to a wide degree is inevitable. Here it becomes clear that document interchange in an open environment demands more than pure data stream conformance; it requires the emergence of a common understanding and a common interpretation of the features of the standard. Different capabilities of the involved editors result in different varieties of the resulting ODA documents or, worse, can lead to interworking problems. Consider as an example an editor that is not able to handle multicolumn text. To ensure interoperability, the system has to provide a *fallback* solution that defines how to replace this unsupported feature by another one. In the case of multicolumn layout a possible fallback could be the sequential layout of all columns. Such problems encountered in the preparation phase led to the definition of the capabilities of the involved ODA systems. These definitions, which are called GSS/RSS (Generator Support Statement/Receiver Support Statement), have been forwarded to the standardization bodies.

The demonstrations showed that interworking in an open environment based on ODA and on well-defined profiles is possible and brings real benefits to the partners in realistic business scenarios. ODA provided the document architecture and interchange format in an OSI-based real-life office scenario. The demonstration also proved that existing word processors and office systems that are familiar to the users can be adapted to ODA-based document interchange.

PODA-2 toolkit. In order to develop ODA applications, a main memory representation for ODA documents and an ODA programming interface are required by the nature of the ODA interchange format (ODIF), which is a data stream and therefore has to be processed sequentially. Document processing applications, including converters, need random access to the components of an ODA document. This requirement has to be implemented through a data stream representing the ODA document in the main memory. Because of the complexity of the ODA standard, the implementation of such basic facilities causes much ef-

fort and high costs. As these tools are required for all ODA applications, only one set of tools was developed and shared by nearly all of the PODA and PODA-2 participants.

The PODA-2 toolkit consists of the following components among others:

- An ODA API (application programming interface) that allows the constituents and attributes of ODA documents to be accessed and manipulated in a random-access mode. This component is composed of appropriate data structures with access and manipulation functions and a storage management system to cope with documents that do not fit into the available main memory area. The first part is called *Stored-ODA* (SODA), the second part is called *ODA-StorageManager* (ODASM). The SODA-ODASM component was developed within the project by ICL.
- Diagnostic tools for viewing and debugging ODA documents. In particular, these tools provide a tree representation of the logical and layout structures and a human-readable interface to inspect ODA constituents.
- Conversion tools to compress and decompress raster graphics.
- A conversion tool that supports the conversion of the stored ODA format to and from the ODA interchange form (ODIF).
- An ODA formatter that performs the ODA layout process. It converts processable ODA documents to formatted processable documents.

The toolkit is a set of generic and highly portable software that simplifies the development of a number of ODA applications. It was used in a series of public demonstrations such as the CeBIT '88, '89, and '90 fairs. The experiences gained from the PODA-2 project show that more than 50 percent of the code of document converters is the common toolkit.

The advantages of such a toolkit as experienced in PODA led to the foundation of the ODA Consortium, which will commercially provide a similar toolkit. Partners in this consortium are Bull SA, DEC, IBM, ICL PLC, Siemens-Nixdorf Informations-systeme AG (SNI), and Unisys. The participating companies are forming a European Economic Interest Grouping, based in Brussels. The development of this toolkit will be carried through by the members; however, the toolkit will be openly licensed. Additionally, the specification of toolkit components was submitted to ECMA[26] as input to their development of a standardized ODA API.

PODA-SAX project. PODA-SAX (Piloting the ODA Extensions and their Applications to Systems, ESPRIT project 5320) has the overall objective of pulling together the results of work done within the ODA and OSI applications arenas and demonstrating document interchange and document access based on international standards. Whereas in PODA-2 the focus of work was the interchange of documents in an open office environment mainly through electronic mail, one main goal of PODA-SAX is the storing and access of documents in open document stores and the access to image libraries. Also, enhancement to the communication standards now used is envisaged. For document mailing, X.400 and the Internet-defined Simple Mail Transfer Protocol (SMTP) will be used; for access to document stores, the recently defined Document Filing and Retrieval (DFR) ISO standard will be used; and for information retrieval, the Directory (X.500) will be used.

The major aim of PODA-SAX is on the integration aspect: it combines existing solutions and components. Its most important topics are:

- To demonstrate a way of interchanging documents between heterogeneous office systems in a multivendor environment
- To demonstrate the employment and the usability of ODA application programming interfaces and toolkits
- To integrate results of work on further ODA applications, such as document security or access to spreadsheets, and OSI communication protocols into a systems demonstrator
- To demonstrate applications that are accessing ODA document libraries

The partners within PODA-SAX are a subset of the PODA-2 members: Bull, IBM, ICL, Olivetti, SNI, and the University College London. For this reason all partners have experience in ODA document interchange and OSI-based communication. The PODA-SAX project started in July 1991 and has a duration of 24 months.

Because one of the major aims of PODA-SAX is to demonstrate ODA- and OSI-based systems, one of the first goals of this project was the development

of a demonstration system architecture, which builds a framework for the services to be provided to users of open office systems. This architecture and an overview of these services is outlined in the following subsection. One of the central services, access to open document stores, is discussed after that in more detail.

PODA-SAX demonstration system. All work done in the PODA-SAX project will be aligned to an overall system and demonstration architecture (see Figure 8) that defines the interworking of different services within the scope of the project. It also defines the protocols used to access different information stores. Stores of special interest are:

- The Project Management Database used for project control documents in the ODA format and accessible over both DFR and X.400
- The ODA Reference Documents Base useful for testing of ODA applications
- The Security Information Directory Information Base (DIB) where all related data for security applications are being held
- The User DIB for information on project participants and especially their networking addresses

Services in the context of the PODA-SAX project are based on the capabilities of ODA and OSI applications. They can be distinguished in five groups:

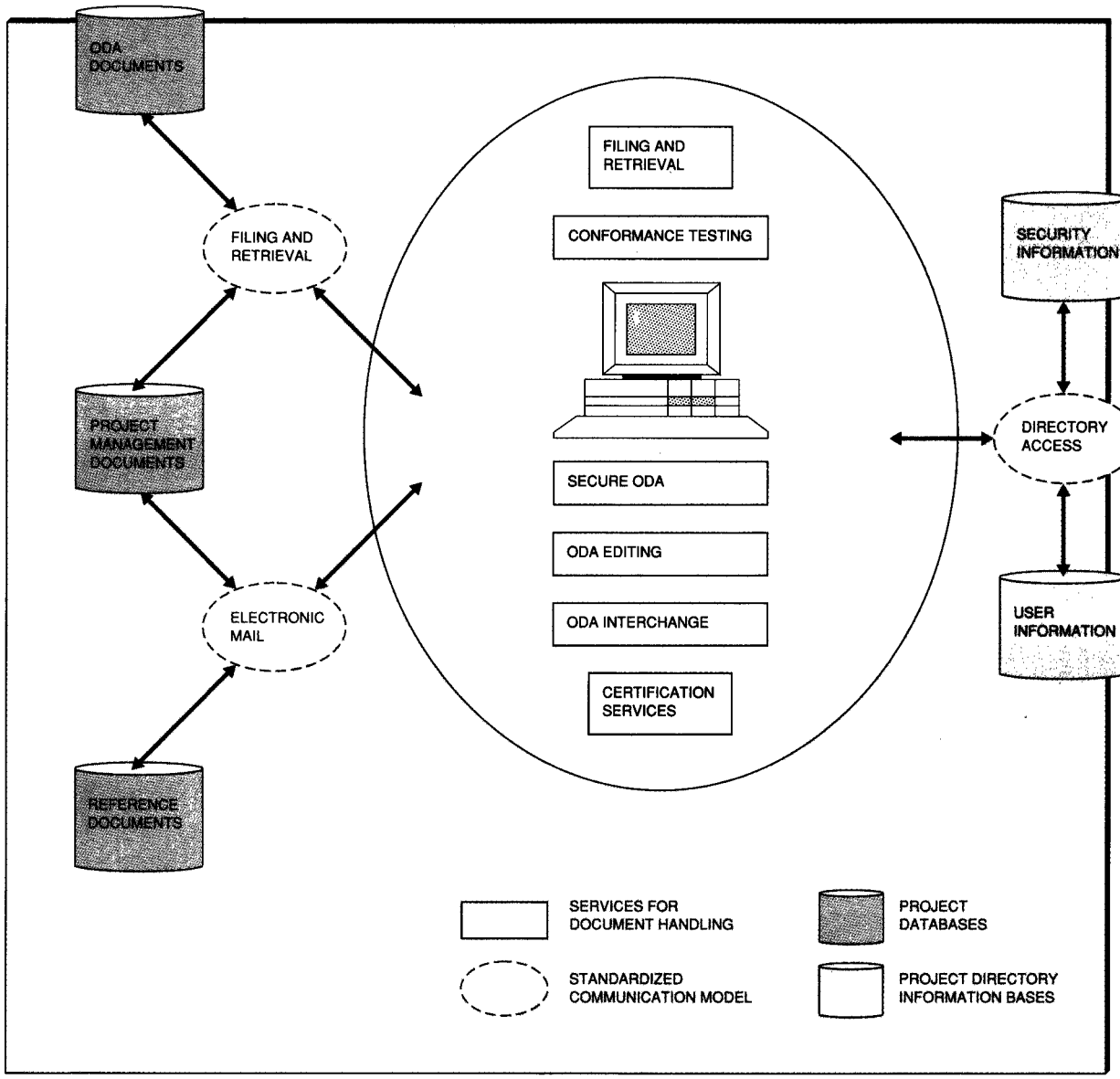
- *Handling of compound documents.* This group includes the handling and the conversion of ODA documents to and from native formats and the processing of image applications.
- *Storage and retrieval services.* Here, the communication aspects and search facilities for remote document stores are considered. In addition, automatic conformance testing for ODA documents is implemented.
- *Directory services.* This group contains services for security features, such as certification distribution, as well as for a project-related information base.
- *Document transmission services.* This group includes different transmission services for ODA documents. Besides X.400 and DFR services, SMTP services are envisaged for some pilot trial with the Internet Engineering Taskforce (IETF) User Community.
- *Application services.* Here, services that are beyond the scope of the currently existing ODA standard, such as the integration of spreadsheets into documents, will be realized.

The results of the PODA-SAX project will be shown in public demonstrations. Additionally, the above services will be used to support the management of the PODA-SAX project. Proving that ODA and OSI services bring real benefits to the work of the project members will be the most convincing argument for the success of PODA-SAX.

PODA-SAX document stores. One of the major aims of PODA-SAX is access to open document stores. In the earlier section entitled "The Need for an Open Document Architecture," it was stated that document stores are another important application of standardized document architectures. The basic design is similar to the one used in the document interchange scenario (compare with Figure 6); however, a different model of the communicating applications is being used. Instead of the X.400 model for an electronic mail system, a model of a client-server document filing and retrieval system is being employed. In addition, the communication protocol in such a model is different, because in this scenario the document-requesting application has to initiate the communication to the document provider (see Figure 9). An interactive application, the so-called client application, requests operations from the server application that is managing the document store. On the client and the server side the applications should provide the means for document searching and for document storage and retrieval. Features for management of the document store are important at the server side (e.g., versioning of documents and access control). Between the client and the server system operations have to be interchanged. Also, documents are transferred in this model (e.g., as the result of a retrieval operation). This will be done through the use of ODIF.

Storage, retrieval, and searching for documents are carried out in this model in an interactive mode. It can be implemented by DFR. In addition to the communication model, which is based on the client-server concept, DFR provides an information model, allowing the store to be organized in a tree-like structure. Operations on a DFR document store are, for example, *search*, *write*, and *read* documents. Security features such as authentication and access control are included, as

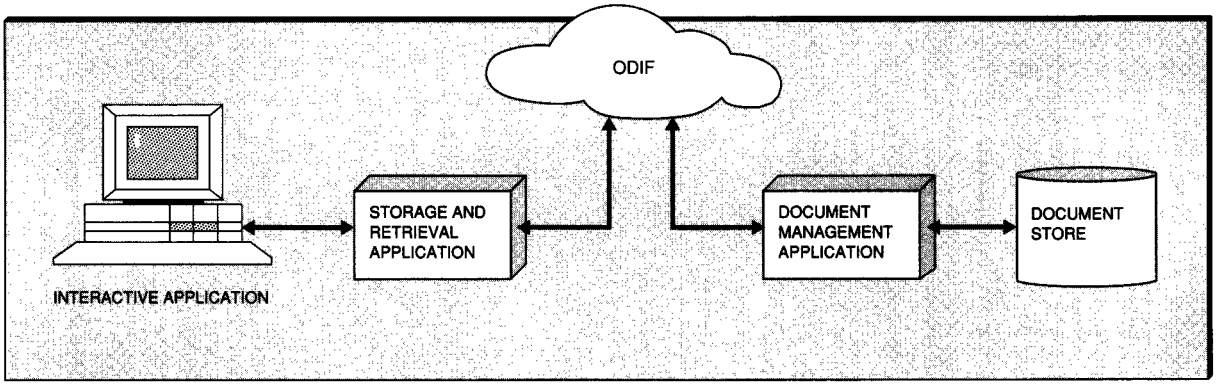
Figure 8 PODA-SAX system architecture



well as document versioning mechanisms. Within the PODA projects work started on prototyping and interworking between DFR implementations.

A first scenario within PODA-SAX for the employment of the DFR implementation will be access to the project management database, which contains all important project documents such as meeting minutes, project control documents as address lists, and technical reports. All of these documents will be transmitted between the server machine and the requester's machine in ODIF format according to the scenario portrayed in Figure 9. Another feature of the project management database is the searching capability provided by DFR, which uses the DFR document attributes to

Figure 9 Document storage scenario



search for documents. These attributes include among others the title of the document, the authors, and several dates, but also keywords and an abstract. Most of these attributes are also defined within ODA and can be part of transmitted ODA documents. Therefore, extraction and integration processes are to be defined in order to build a link between the ODA-defined attributes and those of the DFR.

Some implementation issues on the DFR system prototype as it is being developed within the PODA projects are portrayed in the next section.

Open document technology

In this section the main parts of the technology developed in the projects are outlined.

As discussed before, one of the main objectives of the PODA-2 project was to interconnect existing office systems. In the sequel, a description of the system developed by IBM in order to participate in the PODA-2 multivendor demonstration at CeBIT '90 is given. It is important to note that other participants have designed different systems with different word processors, communication stacks, and office products. Most of those systems, however, are similar with respect to the design decisions that had to be taken for the integration of ODA document interchange capability into existing office system products.

Basic design decisions. For the PODA-2 multivendor demonstrations, the OfficeVision/VM system running on the Virtual Machine/Extended Architecture (VM/XA*) operating system[27] was used. Among other functions, this office system includes components for mailing and storing documents. It has been chosen because it is widely used and, therefore, fits well with the objective to add the open document interchange as an additional function to well-accepted systems.

In order to choose an appropriate editor for this office system platform, which is suitable for open document interchange based on ODA and the Q112 functional profile, the following criteria had to be considered:[28]

- The editor should be able to handle dissimilar content types such as text, raster graphics, and geometric graphics.
- Support for enhanced layout and logical features such as multiple columns, numbered segments, footnotes, and running headers and footers is required.
- The editor should be able to handle multiple fonts and extended character presentation features (e.g., bold, italics, or underlining).

The members of the DisplayWrite editor family fulfill the listed requirements. Additionally, they are running on different platforms such as VM, Multiple Virtual Storage (MVS), Operating System/400* (OS/400*), and Operating System/2* (OS/2*) and use a uniform data stream format for the exchange of documents, namely RFT:DCA, which enables the conveyance of raster and geometric graphics conforming to the IOCA[29] and

GOCA[30] architectures, respectively. For the specific requirements of the demonstration at CeBIT '90 we chose the DisplayWrite 5/2 (DW 5/2) editor running under OS/2, because, in contrast to the DW/370 editor under VM, this editor supports the in-text presentation of graphics and the presentation of multiple fonts text. However, all editors of the DisplayWrite family support RFT:DCA. Therefore, the DW/370 editor is also usable within the demonstration system described below.

Editors of the DisplayWrite family are, of course, not native ODA editors. Thus, a conversion facility from the ODA interchange format to the internal RFT:DCA format and vice versa had to be realized. The converter prototype is described later in this section.

A basic design issue is the integration of such a converter into the OfficeVision/VM system. OfficeVision is able to exchange notes and documents within a homogeneous environment. With the help of PROFS* Extended Mail,[31] XPC,[32] and ONDS,[32] it is also possible to interchange notes in a heterogeneous environment, namely between OfficeVision and X.400 users. The additional feature of the prototype system described in this section that is not part of the products mentioned before is the extension of this functionality for the interchange of ODA documents. The objective is to make maximal use of the OfficeVision product features. The decision on how to integrate the additional components into the products depends on the interfaces provided by OfficeVision as well as on user requirements. In order to protect the user from the inconvenience of having to handle different types of data streams, it was decided to run the conversion process as a transparent step of the X.400 mailing gateway. This decision will be discussed in the following paragraphs.

In principle there are two different solutions for converter integration: first, the conversion process can be initiated by the users as an explicit command or by exploiting an editor gateway, or second, it can be realized as an action of the office system that is transparently performed for all incoming and outgoing documents. The first solution implies high flexibility and is well-suited to office systems that are supporting more than one internal document architecture. However, it means that the user has to be aware of the fact that a document can be represented in different data stream formats, in ODA, or in a proprietary format. The second solution is more suited to homogeneous office systems supporting one internal architecture. The fact that the conversion is performed transparently hides the existence of different data stream formats from the user.

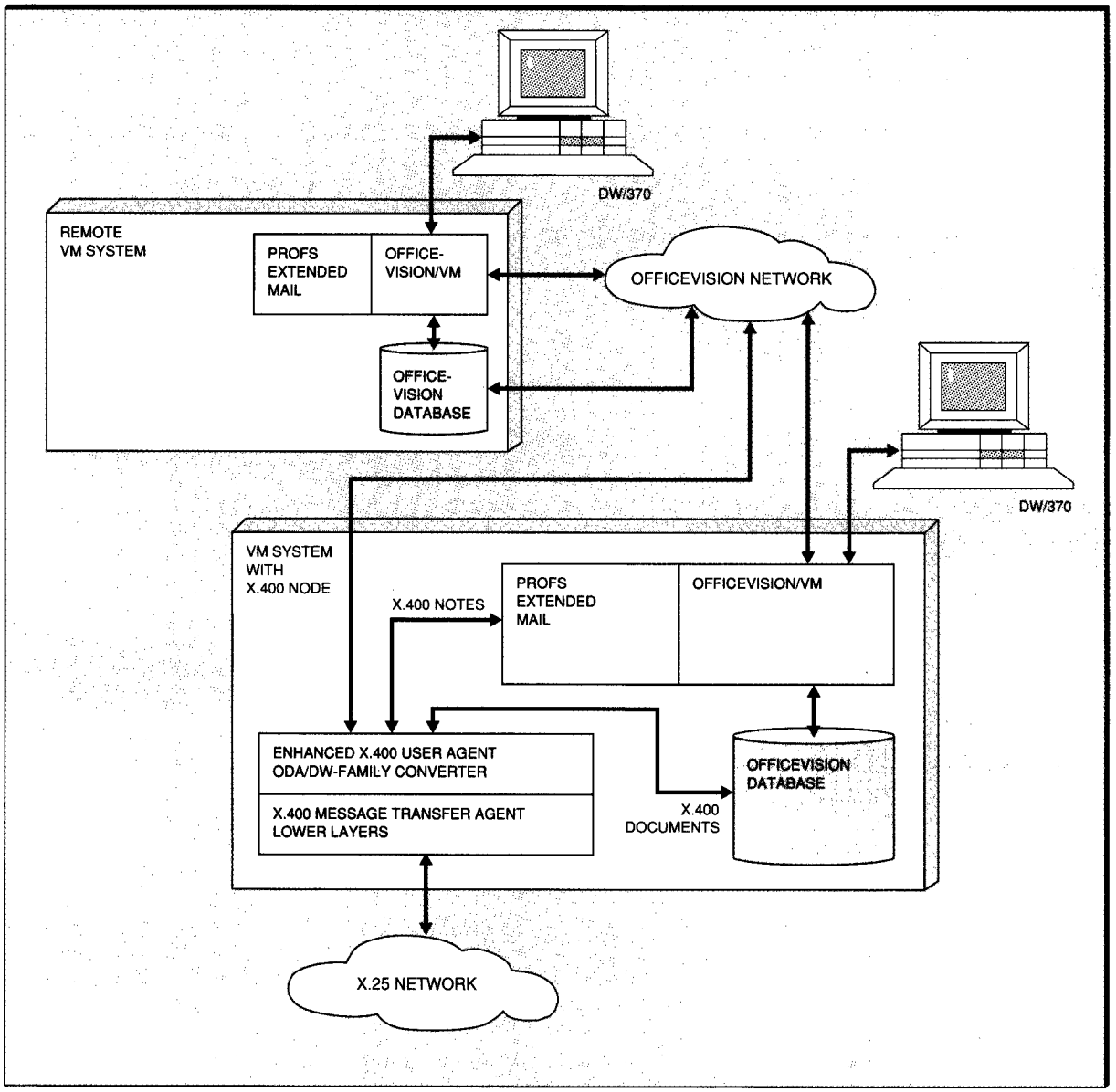
An office system like OfficeVision/VM is more suited for the second approach because it provides a variety of document processing facilities all using the same document architecture. For example, in the OfficeVision system several components such as editors, the mailing system, and document libraries deal with documents. All of these components depend on the internal document format, RFT:DCA. For this reason a change in the internal document representation or the inclusion of an additional format affects all of these applications. Hence, the internal document handling of such an office system should not be changed when the system is enhanced with respect to ODA support. For this reason it was decided to implement the second approach, where documents are converted from ODA to the internal format at the time they enter the system and are converted from the internal format to ODA when they leave the system. Within the PODA-2 project, documents were exchanged using X.400 services. Thus, the converter has been integrated into the X.400 mailing gateway of OfficeVision/VM.

The decision to integrate the converter into the mailing system allows a user who receives documents from the outside world to be unaware of the conversion process. Notification of arriving documents will be received after the document has been converted into the internal format. When sending documents, the user notices no difference when documents are sent to either a recipient within the OfficeVision network or a recipient within the open network. The conversion process is transparent.

A document exchanged between two OfficeVision systems, both utilizing RFT:DCA as their internal document format, need not be converted to the intermediate ODA format. The conversion process has only to be performed in a heterogeneous environment. Thus, the decision to convert an outgoing document depends on the receiving system and can be performed automatically.

The Heidelberg System. The Heidelberg System, shown in Figure 10, was IBM's contribution to the project demonstration at CeBIT '90. The design

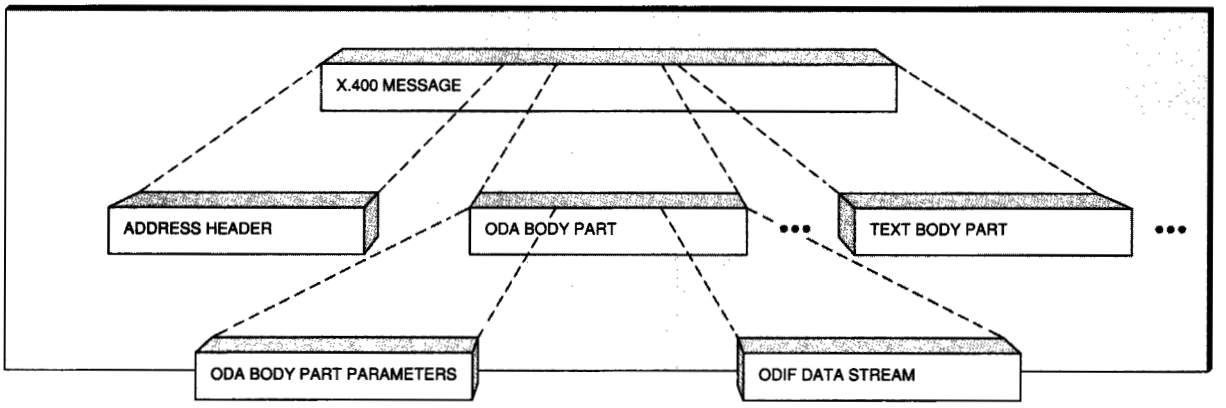
Figure 10 The Heidelberg System



of this system resulted from the above considerations.

The converter runs on the VM operating system as part of the X.400 message-handling system as described in the previous section. For document browsing and processing, either the DW/370 editor or the DW 5/2 editor[33] can be used. The X.400 standard of 1984 was agreed upon as the communication platform for the project demonstrations because, as of today, it is the basis for the majority of X.400 products. In order to convey ODA documents as X.400 body parts some modifications of the existing X.400 product were necessary, because X.400 in its 1984 version does not support an ODA body type.[34] For this reason, in Figure 10 the

Figure 11 Construction of a sample X.400 message



Enhanced X.400 User Agent is shown instead of the standard product XPC. According to the X.400 standard, a message consists of a header containing address information and other message parameters and a sequence of body parts. Each part can be of type text or it can be of type ODA document. Some other body types are also defined in the X.400 standard, as shown in Figure 11. The X.400 system has to parse incoming messages and initiate specific operations for each single body part contained in them. In the case of an ODA body part, the ODA body part parameters, containing the properties of the ODIF data stream in the message, can be used by the X.400 system to initiate a call to the appropriate conversion process with the ODIF data stream. The result of the conversion, which is an RFT:DCA document, is then put into the OfficeVision document database, and the receiver of the document will be notified of the arrival. These activities of the X.400 system are running as part of the *user agent*, which is an integral part of the X.400 system.

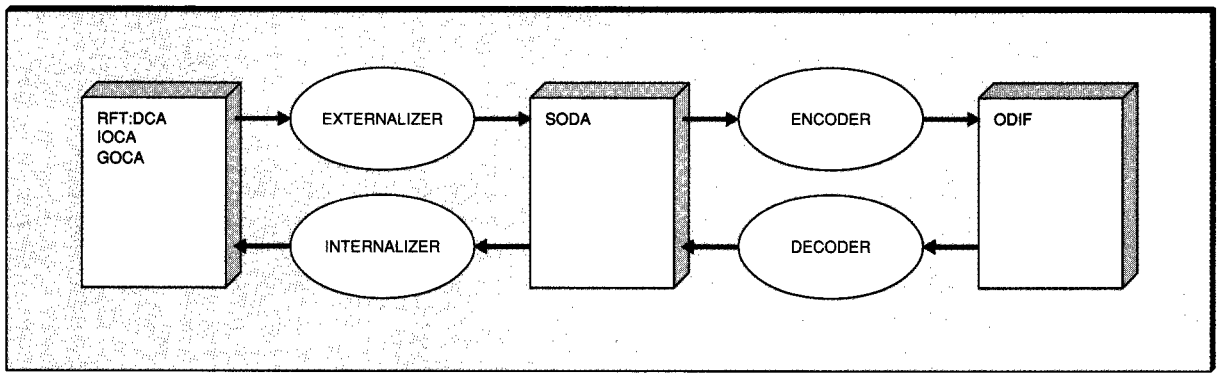
After receiving a document by means of the X.400 system as described above, the document can be accessed and manipulated within the office system using the mentioned editors and other functions provided by OfficeVision. To send a document to an external partner, an appropriate X.400 header is built,[35] and the document is appended and sent to the extended X.400 system. There the conversion process takes place and the resulting ODA document is sent over the X.25 network.

The extensions of the X.400 system are needed for only one node in the OfficeVision network. The X.400 system is able to forward mail and documents not addressed to the local node to any other node on the OfficeVision[36] network. Support of networks of OfficeVision systems by one enhanced X.400 system allows open document interchange to be provided for all users in a network with only one installation of the X.400 system and only one access to the X.25 network. The conversion process is performed on the system where the extended X.400 system is operating, and the submission of documents is carried out in the RFT:DCA format. Likewise, when a user on a remote OfficeVision node sends documents to external recipients, they are converted on the enhanced X.400 system.

Converter design. The converter plays a central role within the Heidelberg System. It transforms a document that is represented in the ODIF data stream format to and from the RFT:DCA data stream format. If the ODIF document contains graphic or image information, this information is transformed into GOCA or IOCA format and included in the RFT:DCA stream. When converting RFT:DCA documents to ODIF format, the text, the GOCA, and the IOCA portions of the data stream are converted to ODIF format.

Figure 12 illustrates the high-level design of the converter. Four modules comprise the converter. SODA is an intermediate representation for ODA documents in main memory, where ODA constituents and their attributes can be directly addressed. Therefore, the conversion process is carried out in two steps. The encoder module is producing an ODIF sequential data stream from

Figure 12 Converter design

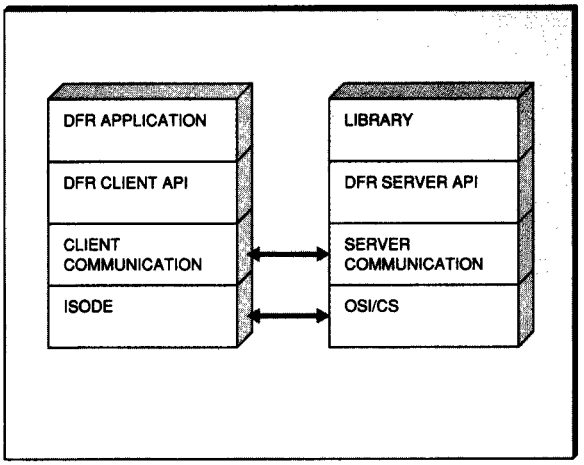


the SODA representation of the document. The decoder is parsing such an ODIF stream and produces the intermediate representation. Both modules aim to perform a mapping between semantically equal data representations. Therefore, no loss of information relevant to an application occurs during these conversions. For the implementation on VM the emphasis in these modules was on the efficiency of the algorithms for parsing and generating the ASN.1-defined data streams representing ODIF in an efficient way. The reason was that the modules were exposed as the most time-consuming task within the conversion process.

More design issues had to be solved in the externalizer and internalizer modules because the ODA and RFT:DCA architectures differ from one another in nature and in some of their supported features. The task of both modules is to convert between the intermediate format and the RFT:DCA format and to convert graphics and images. The mapping between document descriptions corresponding to the ODA standard and those corresponding to RFT:DCA can be done in a number of ways. However, the design decisions taken for these components have a major impact on the interoperability of the overall system with other ODA applications. The focus of the work in the project was on a comprehensive and extensible design for these modules. The interworking experiences gained from the multivendor demonstrations proved its suitability.

We now illustrate with an example the kind of design decisions to be made for the internalizer and externalizer. A more detailed discussion of the mapping, including a comparison between the ODA and the RFT:DCA concepts, and also a detailed description of the converter design and the gained interoperability experiences, are described in Filip et al.[37] Advanced word processors allow different page layouts for the left pages and the right pages to be specified, as, for example, in a manual. Typically the page layout differs in the position of the page number and perhaps also in the margins of the text. The concept in ODA that allows such different layouts to be specified is called recto/verso pages. RFT:DCA allows a page to be split into three main areas: an optional header area followed by a required body area followed by an optional footer area. The basic concept for recto and verso pages within RFT:DCA is the different content and layout of running headers and footers, usable, for example, for page numbers on the right side or on the left side, depending on whether the page is even or odd. However, it is not possible to independently specify different positions and margins for recto and verso body frames. In the ODA architecture a greater degree in flexibility is provided by which recto and verso pages can be independently specified. This implies that a mapping of RFT:DCA to ODA can be done without loss of information. The mapping of ODA to RFT:DCA cannot be done without loss of formatting information when the ODA document contains classes with recto/verso page specifications. In particular, these documents are the ones that have different body frame specifications for recto and verso pages. In the externalizer a so-called fallback for these cases has to

Figure 13 DFR prototype system architecture



be designed that maps the two different recto-verso body frame specifications in the ODA document to one body frame specification in the RFT: DCA document.

DFR implementation. To gain experience with the ODA technology, also in the scenario "open document archives," as explained above, prototyping and interworking work for DFR has been part of PODA-2 and PODA-SAX. Within the PODA-2 project a DFR prototype was developed and demonstrated to study the capabilities of DFR and to investigate the interworking between ODA and the access and communication protocol defined by DFR. This work is being continued within PODA-SAX. The prototype provides the required communication facilities and general services that can be used by various applications via a DFR application programming interface (see Figure 13) designed as part of PODA-2. The DFR system is implemented in a way that enables the porting of the DFR client code on partners' systems. To allow this porting, the client communication is based on the ISODE[38] toolkit, which ensures maximum portability. The server communication is using OSI/CS.[39] On top of the DFR server system a flexible library interface is implemented to allow the use of different local document stores by using DFR server API functions.

Two important areas of work are considered in the PODA projects: first, the development of links between the ODA- and DFR-defined attributes and, second, the development of a DFR client API. The first item is advantageous for searching ODA documents in the DFR-defined document store. Here mechanisms are being provided that allow access to the document content, especially to the ODA-defined document attributes stored in a DFR-defined document base. The second item of the DFR implementation work will be the development of the DFR client API. This API allows the design and implementation of DFR-based applications without caring about the details of the communication protocol.

Some new requirements for storing ODA documents in a DFR system have been identified in PODA-2. Hypertext systems, for example, make it necessary to access not only complete documents, but also parts of documents. That requirement has to be supported by the document server. PODA-SAX will continue the work on DFR enhancements and the interoperability between existing DFR implementations.

As a store and forward solution for access to a document store, an X.400-based "active mailbox" was developed and extensively used in PODA-2. The active mailbox allows all project partners to access a document store because all of them have working X.400 connections that could provide, with minor enhancements, storage and retrieval. Within PODA-2 this active mailbox was a valuable help in the preparation phases of public demonstrations. All test documents were stored on this system and could be retrieved by all partners. Within PODA-SAX such a store and forward access method to the project management database will be provided in addition to the DFR access. This provision allows two access methods to a document store, both complying with the DFR model. The store and forward solution is advantageous for these partners, who have no working DFR implementations running.

Conclusion

A commonly available information architecture like ODA is important for widespread professional use of computer networks for interpersonal cooperation. The applications that are used in such cooperations include electronic mail as well as document archiving and retrieval systems. In order to cover all of these applications, ODA is de-

signed as an architecture for the needs of a broad spectrum of applications. Because of this design, the pathway for ODA from being formulated as a standard to the marketplace is not as straightforward as it may be for standards with a more limited scope. Cooperation of the different system vendors is a prerequisite for achieving high interchange fidelity. The PODA projects have contributed to four activities that are essential for the emergence of a consistent set of ODA products:

1. The market-driven development of Document Application Profiles: The generality of the base standard had to be narrowed to the functionality of the most important applications. PODA has decided to push the development of the Q112 Document Application Profile in order to achieve interworking of today's office editors. Both architectural work and implementation experiences from the project have contributed to the development of this profile that is implementable and aligned with the needs of the market.
2. Identification of a consistent base technology: Consistency is the key to the success of ODA in two areas. First, the concept of using Document Application Profiles to narrow the standard for specific application areas requires that all DAPs use the base ODA concepts in a consistent way. Second, consistent implementations on different hardware and software platforms have to be ensured to achieve interworking. PODA has defined a toolkit that has been used by all partners on their platforms to achieve this consistency. Although limited in its functionality, the concept was very successful. It has led to the formation of the ODA Consortium that is commercially developing a comprehensive set of tools for building ODA applications.
3. Adapting existing office systems: Within the PODA projects ODA support was developed for well-known office systems. This work proved that the Q112 profile enables high-fidelity document interchange between modern word processors. The cooperation provides a useful framework for interworking tests.
4. Promotion of ODA: Establishing early contacts with potential users of ODA solutions is important for controlling the direction of the technical development. PODA has established this contact, for instance, by several public demonstrations and publications.

The Heidelberg System that is an enhancement to IBM's OfficeVision for ODA-based document interchange is a well-tested prototype and was designed to allow users to interchange documents in open networks in the same way to which they are accustomed in homogeneous environments. Leading European customers want to use this system to fulfill their requirements of open document processing. Currently, field trials with these customers are being established to respond to their requests.

Acknowledgments

The success of cooperation projects mainly depends on the readiness of all partners to pursue the common objectives. For all colleagues within the PODA projects we would like to thank C. Bathe (SNI), V. Gallo (Olivetti), P. van Houten (Océ), P. Kirstein (UCL), C. Kou (Bull), C. Nye (British Telecom), R. Pennell (ICL), and F. Reynaud (TITN) for their fruitful collaboration. Contributions to the technology and the interworking demonstrations described in this paper from H. Bunz, W. Filip, W. Knobloch, D. Kropp, M. Mattingley-Scott, J. Monnery, B. Paul, W. Racke, and H. Schmied are gratefully acknowledged. Discussions with J. Czyszczewski, F. Dawson, and D. Gottschall had a profound influence on the design of the Heidelberg System. In addition, we thank all of the anonymous reviewers for their constructive comments. All of this work would not have been possible without the efforts of C. Berta and G. Müller who were able to have IBM participate in the PODA projects.

Cited references and notes

1. *Information Processing—Text and Office Systems—Office Document Architecture (ODA) and Interchange Format*, ISO IS 8613 (1989).
2. *Document Content Architecture: Revisable-Form-Text Reference*, SC23-0758, IBM Corporation (1986); available through IBM branch offices.
3. M. R. DeSousa, "Electronic Information Interchange in an Office Environment," *IBM Systems Journal* **20**, No. 1, 4–22 (1981).
4. N. Naffah, "Multimedia Applications," *Computer Communications*, 243–249 (1990).

5. F. G. Halasz, "Reflections on NoteCards: Seven Issues for the Next Generation of Hypermedia Systems," *Communications of the ACM* **31**, 836–852 (1988).
6. *Information Processing Systems—Text and Office Systems—Standard Generalized Markup Language (SGML)*, ISO IS 8879 (1986).
7. C. Goldfarb, *The SGML Handbook*, Oxford University Press, Oxford (1990).
8. J. Rosenberg, M. Sherman, A. Marks, and J. Akkerhuis, *Multi-Media Document Translation—ODA and the EXPRES Project*, Springer, New York (1991).
9. R. Hunter, P. Kaijser, and F. Nielsen, "ODA: A Document Architecture for Open Systems," *Computer Communications* **12**, 69–79 (1989).
10. Frames are rectangular areas on a page that are specified by their position and dimension.
11. *Information Processing Systems—Open Systems Interconnection—Specification of Abstract Syntax Notation One (ASN.1)*, ISO IS 8824 (1990).
12. *Information Processing Systems—Open Systems Interconnection—Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)*, ISO IS 8825 (1990).
13. *Information Technology—International Standardized Profile FOD11—Office Document Format—Simple Document Structure—Character Content Architecture Only*, ISO DIS 10610-1 (1991).
14. *Information Technology—International Standardized Profile FOD26—Office Document Format—Enhanced Document Structure—Character, Raster Graphics, and Geometric Graphics Content Architectures*, ISO DIS 11181-1 (1991).
15. *Information Technology—International Standardized Profile FOD36—Office Document Format—Extended Document Structure—Character, Raster Graphics, and Geometric Graphics Content Architectures*, ISO DIS 11182-1 (1991).
16. *ODA Document Application Profile Q111—Processable and Formattable Documents—Basic Character Content*, EWOS ENV 41509 (1990).
17. *ODA Document Application Profile Q112—Processable and Formattable Documents—Extended Mixed Mode*, EWOS ENV 41510 (1990).
18. A. Spiceley, "ODA Profiles: Application and Development," *Computer Networks and ISDN Systems* **21**, 165–173 (1991).
19. *Information Technology—Document Filing and Retrieval* (DFR), ISO IS 10166 (1989).
20. U. Bormann and C. Bormann, "Standards for Open Document Processing: Current State and Future Developments," *Computer Networks and ISDN Systems* **21**, 149–163 (1991).
21. J. Nelson, C. Bathe, I. Campbell-Grant, M. Coon, K. Fischer, P. Kirstein, G. Kroenert, and M. Mabrouk, "The Role of the PODA Project in the Adoption and Development of ODA," *Computer Networks and ISDN Systems* **21**, 175–185 (1991).
22. "Message Handling Systems—System and Service Overview," *CCITT Recommendation X.400 Blue Book*, ITU, Geneva (1989).
23. EurOSInet is a European marketing association that promotes and demonstrates the practical use of Open Systems in multivendor environments. A liaison between the PODA-2 project and EurOSInet was established resulting in the foundation of the EurOSInet ODA subgroup.
24. EDIFACT (Electronic Data Interchange for Administration, Commerce, and Transport) is a way to interchange electronic data within the financial community using OSI services.
25. FTAM (File Transfer, Access, and Management) is an OSI application-layer standard, specifying a service and a protocol for accessing and manipulating files.
26. The creation of a task group under the aegis of the European Computer Manufacturer's Association (ECMA TC29 / TG-API) for the purpose of standardizing an ODA API was a PODA-2 initiative.
27. The developed technology is designed for migration to OfficeVision/MVS.
28. It is not necessary that only one editor has to accomplish all of these stated tasks, because it is perfectly acceptable to provide a suite of editors for that.
29. *Image Object Content Architecture Reference*, SC31-6805-0, IBM Corporation (1990); available through IBM branch offices.
30. *Graphics Object Content Architecture Reference*, SC31-6804-0, IBM Corporation (1990); available through IBM branch offices.
31. PROFS Extended Mail provides enhancements to the OfficeVision mail functions to facilitate exchanging mail between OfficeVision and other electronic mail systems.
32. XPC, the X.400 PROFS Connection, and ONDS, the Open Network Distribution Services, are IBM's implementation of X.400 for VM.
33. Because this editor is running on PS/2 machines, its usage requires that the RFT:DCA documents are transferred between the System/370 and PS/2 systems.
34. X.400 in the version of 1988 has the ability to convey ODA documents. Within the PODA-2 project extensions to the 1984 standard were agreed upon that correspond to the 1988 standard. These enhancements to the X.400 Version '84 had been brought forward to the standards bodies and will become part of the ISO profiles 10610, 11181, and 11182.
35. This is handled by an OfficeVision extended mail interface.
36. This is in fact handled by the communication network that is used within a distributed OfficeVision/VM system.
37. W. Filip, J. Kaemper, W. Knobloch, "Conversion Between ODA Level 2 and RFT:DCA—Experiences with a Prototype," *Computer Networks and ISDN Systems* **21**, 197–210 (1991).
38. ISODE, the ISO Development Environment is a publicly available set of code, comprising low-level functions as well as some OSI applications, such as X.400 and X.500 implementations.
39. OSI/CS, OSI Communication Subsystem, is a set of IBM OSI products running on several system platforms.

Accepted for publication June 25, 1992.

Heinz Fanderl *IBM European Networking Center, Tiergartenstrasse 8, D-6900 Heidelberg, Germany (electronic mail: fanderl@mazvm01.vnet.ibm.com).* Dr. Fanderl received his diploma and his Ph.D. in computer science from the Technical University in Munich, Germany, in 1984 and 1989. In 1989 he joined the IBM European Networking Center in Heidelberg. Since then he has been working within the PODA projects. Since early in 1991 he has been project leader of PODA-2, and since June 1991 he has been project leader of the PODA-SAX project.

Kristian Fischer *IBM European Networking Center, Tiergartenstrasse 8, D-6900 Heidelberg, Germany (electronic mail: kfischer@mazvm01.vnet.ibm.com).* Dr. Fischer received his doctorate in computer science from the University of Passau in 1986. He joined IBM Germany in 1986 at the European Networking Center (ENC). He worked in the area of prototyping and interworking of standardized message-handling systems, the ODA support for IBM's office systems platforms, and in the definition of the PODA projects. From 1989 to 1991 he was project leader of the PODA-2 project. He is currently manager of the Open Document Communication department within the ENC.

Jürgen Kämper *Wittgensteiner Kuranstalten, Im Herrengarten, D-5920 Bad Berleburg, Germany.* Dr. Kämper received his diploma from the University of Dortmund in 1986 and his Ph.D. from the University of Oldemburg in 1989, both in computer science. In 1989 he joined the IBM European Networking Center. He was responsible for designing and prototyping parts of the ODA-to-RFT:DCA converter. In 1992 he left IBM and is now working in a health center in Germany.

Reprint Order No. G321-5495.