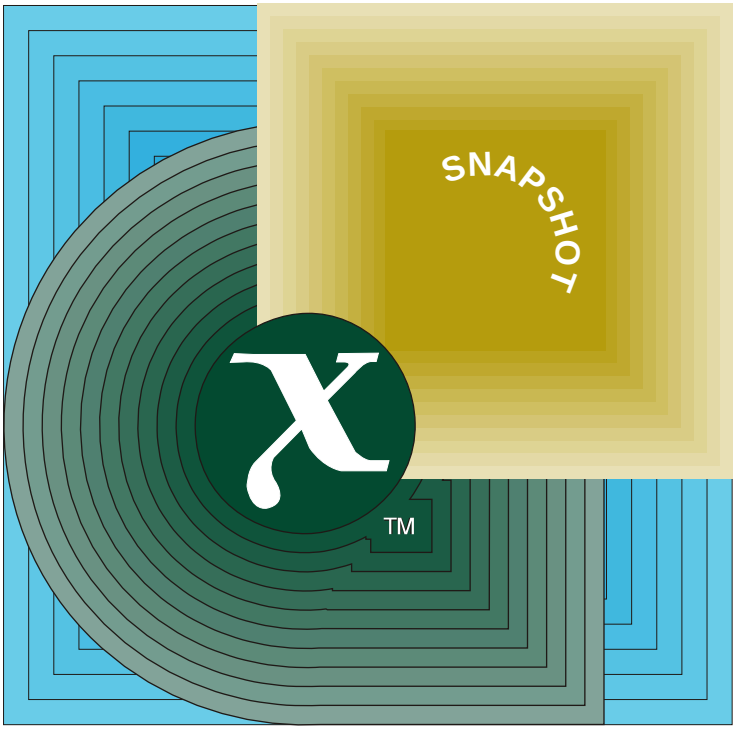
Snapshot

Document Interchange Reference Model



THE Open GROUP

[This page intentionally left blank]

X/Open Snapshot

Document Interchange Reference Model

X/Open Company Ltd.

X/Open Snapshot

Document Interchange Reference Model

ISBN: 1 872630 50 2
X/Open Document Number: S209

Set in Palatino by X/Open Company Ltd., U.K.
Published by X/Open Company Ltd., U.K.

Any comments relating to the material contained in this document may be submitted to X/Open at:

X/Open Company Limited
Apex Plaza
Forbury Road
Reading
Berkshire, RG1 1AX
United Kingdom

or by Electronic Mail to:

XoSpecs@xopen.co.uk

Contents

Preface

X/Open

X/Open is an independent, worldwide, open systems organisation supported by most of the world's largest information systems suppliers, user organisations and software companies. Its mission is to bring to users greater value from computing, through the practical implementation of open systems.

X/Open's strategy for achieving this goal is to combine existing and emerging standards into a comprehensive, integrated, high-value and usable system environment, called the Common Applications Environment (CAE). This environment covers the standards, above the hardware level, that are needed to support open systems. It provides for portability and interoperability of applications, and allows users to move between systems with a minimum of retraining.

The components of the Common Applications Environment are defined in X/Open CAE Specifications. These contain, among other things, an evolving portfolio of practical application programming interfaces (APIs), which significantly enhance portability of application programs at the source code level, and definitions of, and references to, protocols and protocol profiles, which significantly enhance the interoperability of applications.

The X/Open CAE Specifications are supported by an extensive set of conformance tests and a distinct X/Open trademark - the XPG brand - that is licensed by X/Open and may be carried only on products that comply with the X/Open CAE Specifications.

The XPG brand, when associated with a vendor's product, communicates clearly and unambiguously to a procurer that the software bearing the brand correctly implements the corresponding X/Open CAE Specifications. Users specifying XPG-conformance in their procurements are therefore certain that the branded products they buy conform to the CAE Specifications.

X/Open is primarily concerned with the selection and adoption of standards. The policy is to use formal approved *de jure* standards, where they exist, and to adopt widely supported *de facto* standards in other cases.

Where formal standards do not exist, it is X/Open policy to work closely with standards development organisations to assist in the creation of formal standards covering the needed functions, and to make its own work freely available to such organisations. Additionally, X/Open has a commitment to align its definitions with formal approved standards.

X/Open Specifications

There are two types of X/Open specification:

- *CAE Specifications*

  CAE (Common Applications Environment) Specifications are the long-life specifications that form the basis for conformant and branded X/Open systems. They are intended to be used widely within the industry for product development and procurement purposes.

Developers who base their products on a current CAE Specification can be sure that either the current specification or an upwards-compatible version of it will be referenced by a future XPG brand (if not referenced already), and that a variety of compatible, XPG-branded systems capable of hosting their products will be available, either immediately or in the near future.

CAE Specifications are not published to coincide with the launch of a particular XPG brand, but are published as soon as they are developed. By providing access to its specifications in this way, X/Open makes it possible for products that conform to the CAE (and hence are eligible for a future XPG brand) to be developed as soon as practicable, enhancing the value of the XPG brand as a procurement aid to users.

- *Preliminary Specifications*

  These are specifications, usually addressing an emerging area of technology, and consequently not yet supported by a base of conformant product implementations, that are released in a controlled manner for the purpose of validation through practical implementation or prototyping. A Preliminary Specification is not a ''draft'' specification. Indeed, it is as stable as X/Open can make it, and on publication has gone through the same rigorous X/Open development and review procedures as a CAE Specification.

  Preliminary Specifications are analogous with the ''trial-use'' standards issued by formal standards organisations, and product development teams are intended to develop products on the basis of them. However, because of the nature of the technology that a Preliminary Specification is addressing, it is untried in practice and may therefore change before being published as a CAE Specification. In such a case the CAE Specification will be made as upwards-compatible as possible with the corresponding Preliminary Specification, but complete upwards-compatibility in all cases is not guaranteed.

In addition, X/Open periodically publishes:

- *Snapshots*

  Snapshots are ''draft'' documents, which provide a mechanism for X/Open to disseminate information on its current direction and thinking to an interested audience, in advance of formal publication, with a view to soliciting feedback and comment.

  A Snapshot represents the interim results of an X/Open technical activity. Although at the time of publication X/Open intends to progress the activity towards publication of an X/Open Preliminary or CAE Specification, X/Open is a consensus organisation, and makes no commitment regarding publication.

  Similarly, a Snapshot does not represent any commitment by any X/Open member to make any specific products available.

X/Open Guides

X/Open Guides provide information that X/Open believes is useful in the evaluation, procurement, development or management of open systems, particularly those that are X/Open-compliant.

X/Open Guides are not normative, and should not be referenced for purposes of specifying or claiming X/Open-conformance.

This Document

A document is a set of information, organised as a unit and intended for human perception. Definitions have changed over time, to embrace the increasing complexity of printable information, of all types and in electronic as well as paper form. The challenge for X/Open is to specify interchange formats and APIs which accommodate the wide range of document models that exist.

This document looks at existing internationally recognised Open Document Architecture (ODA) document profiles, which specify what elements a document may comprise. It then introduces the use of object-oriented programming for writing applications which can accommodate the wide range of document types, and from this proceeds to examine implicit document structure when a document is viewed as a set of objects. It then assesses the API Service goals and requirements for document interchange, and considers some API implementation issues.

This specification relates to various international standards and recommendations which are listed in **Referenced Documents**. It should be used in conjunction with those International Standards or Recommendations.

This Snapshot is being published for information, and to stimulate comment and provide an opportunity for wider review throughout the industry.

Trademarks

UNIX® is a registered trademark of UNIX System Laboratories Inc. in the U.S.A. and other countries.

Palatino® is a registered trademark of Linotype AG and/or its subsidiaries.

X/Open™ and the ''X'' device are trademarks of X Company Ltd. in the U.K. and other countries.

CDA is a trademark of Digital Equipment Corporation.

Referenced Documents

The following documents are referenced or are related to the content of this document:

CCITT T.4
: CCITT Recommendation T.4 - Standardization of group 3 facsimile apparatus for document transmission (1988).

CCITT T.6
: CCITT Recommendation T.6 - Facsimile coding schemes and coding control functions for group 4 facsimile apparatus (1988).

ECMA TC29/92/8
: ECMA TC29/92/8 - Open Document Architecture (ODA) - Application Programming Interface - Application Profile Interface for Handling Compound Documents.

ECMA TC29/92/9
: ECMA TC29/92/9 - Open Document Architecture (ODA) - Application Programming Interface - Constituent Level Interface for Handling Compound Documents.

ISO 8613-1
: ISO 8613-1 : 1989, Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 1: Introduction and general principles.

ISO 8613-2
: ISO 8613-2 : 1989, Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 2: Document Structures.

ISO 8613-4
: ISO 8613-4 : 1989, Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 4: Document profile.

ISO 8613-5
: ISO 8613-5 : 1989, Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 5: Office document interchange format.

ISO 8613-6
: ISO 8613-6 : 1989, Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 6: Character content architectures.

ISO 8613-7
: ISO 8613-7 : 1989, Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 7: Raster graphics content architectures.

ISO 8613-8
: ISO 8613-8 : 1989, Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 8: Geometric graphics content architectures.

ISO 8613-1 : (to be published)
: ISO 8613-1 : (to be published), Information processing - Text and office systems; Office Document Architecture (ODA) and interchange format - Part 1: DAD - A document application profile proforma and notation.

ISO/IEC 646
: ISO/IEC 646 : 1991, Information technology - ISO 7-bit coded character set for information interchange.

ISO 8859-1

ISO 8859-1 : 1987, Information processing - 8-bit Single- byte coded graphic character sets - Part 1: Latin alphabet No. 1.

ISO 6937-2

ISO 6937-2 : 1983, Information processing - Coded character sets for text communication - Part 2: Latin alphabet and non-alphabetic characters.

ISO 2022

ISO 2022 : 1986, Information processing - ISO 7-bit and 8-bit coded character sets - Code extension techniques.

ISO 7350

ISO 7350 : 1984, Text communication - Registration of graphic character subrepertoires.

ISO / IEC 8824

ISO / IEC 8824 : 1990, Information technology - Open Systems Interconnection - Astract Syntax Notation One (ASN.1).

ISO / IEC 8825

ISO / IEC 8825 : 1990, Information technology - Open Systems Interconnection - Basic encoding rules for abstract syntax notation one (ASN.1).

ISO 8632

ISO 8632 : 1987, Information processing systems - Computer graphics - Metafile for the storage and transfer of picture description information.

Part 1: Functional specification

Part 3: Binary encoding.

ISO 8879

ISO 8879 : 1986, Information processing - Text and office systems - Standard Generalized Markup Language (SGML).

ISO / IEC ISP 10610-1

ISO / IEC 10610-1, Information technology - International Standardized Profile FOD11 - Office Document Format - Simple document structure - Character content architecture only - Part 1: Document application profile.

ISO / IEC ISP 11181-1

ISO / IEC 11181-1, Information technology - International Standardized Profile FOD26 - Office Document Format - Enhanced document structure - Character, raster graphics and geometrical graphics content architectures - Part 1: Document application profile.

ISO / IEC ISP 11182-1

ISO / IEC 11182-1, Information technology - International Standardized Profile FOD36 - Office Document Format - Extended document structure - Character, raster graphics and geometric graphics content architectures - - Part 1: Document application profile.

XDIF

X / Open Document Interchange Format, Preliminary Specification, Part Number P217, 1992.

ISO 8613-7, ISO 8613-8, CCITT Recommendations T.4, T.6 and ISO 8632 are not applicable to the FOD11 document model since they concern content types not present in FOD11.

Chapter 1

Definition of a Document

Note: This document relates to various international standards and recommendations. These are listed in **Referenced Documents**, and should be referenced wherever the need for further definitive information arises.

A document is a set of information, organised as a unit and intended for human perception. ISO 8613 defines documents more rigorously and X/Open endorses that definition.

The definition of document has changed over time based on available technology. Any concept of *document* reflects one's assumptions about what technology is available for processing that document. Thus, a document might be:

- printed text on a series of sheets of paper (reflecting the implicit assumption that a typewriter or similar device is used for input and output)
- an electronic text-based composition that reflects the abilities of current products, for example to highlight text, to add structure to the text, and to include graphics (reflecting current office system technologies)
- an electronic composition modelled after a book or magazine, in which text coexists with photographs and anything else that might be represented on a page (reflecting desktop publishing and advanced office systems technologies)
- an electronic composition based on future document models. It is widely accepted that such future models will not necessarily assume that the information in a document is static, or that a document could be represented on a series of pages, or even that the user's perception of the document is based on the sense of vision. Already, technology will let a computer system associate with a document, or store in a document, a wide range of diverse information, such as spreadsheets, sound recordings, or live video.

The only ultimate restriction on the scope of a document is that it is an amount of structured information intended for human perception that a user wishes to create, file, retrieve, manipulate or interchange as a unit. A document can contain any type of information that a user wants stored in the document.

The challenge for X/Open is to specify interchange formats and APIs that accommodate this wide range of document models, and accommodate future document models, in a unified way.

Chapter 2

X/Open Document Models

The application program must make assumptions about the range of elements that a document may comprise. In doing so it defines the set of objects that it is prepared to process meaningfully. Several industry and standards organisations have made progress in this area.

2.1 ODA Document Profiles

The Open Document Architecture (ODA) has the following as internationally standardised profiles:

FOD11 ISO ISP FOD11 is the *office document format profile for the interchange of basic-function, character-content documents, in processable and formatted forms.*

FOD11 documents contain only character text. They may range from memos and letters to simple structured documents.

FOD26 ISO ISP FOD26 is the *office document format profile for the interchange of enhanced-function, mixed-content documents, in processable and formatted forms.*

FOD26 documents can contain character text, raster graphics, or geometric graphics. They may range from simple documents to highly structured technical reports, articles, and typeset documents such as monochrome newsletters.

FOD36 ISO ISP FOD36 is the office document format profile for the interchange of extended-function, mixed-content documents, in processable and formatted forms.

FOD36 documents include the capabilities of the FOD26 document model, as well as monochrome documents prepared by personal publishing systems. The FOD36 profile supports extended document structure such as table, form, reference and extended layout capability. It also supports typographic features such as leading, kerning, ligatures and ruby.

These three document models are strictly subsetted. Any valid document under the FOD11 model is valid under the FOD26 and FOD36 models; any valid document under FOD26 is valid under FOD36.

The associated X/Open Document Interchange Formats Preliminary Specification (reference **XDIF**), endorses these FOD specifications for interchange of documents between personal word processing and publishing applications.

The Open Document Architecture Consortium (ODA Consortium) is developing a specification for Application Programming Interfaces (APIs) for document access and manipulation. In the process, the ODA Consortium has specified the abstract internal structure of documents at three levels of complexity. These levels correspond directly to the three ISO International Standardised Profiles (ISPs) for the use of ODA, described above. This specification work is currently available through two ECMA documents (references **ECMA TC29/92/8** and **ECMA TC29/92/9**).

2.2 SGML

The Standard Generalized Markup Language (SGML) is a widely-used language for implementing markup-based technical publishing systems. The CALS (Computer-aided Acquisition and Logistics Support) project has specified a number of document types which make use of the SGML language for encoding documents for interchange. X/Open endorses this emerging form of specification as a method for interchanging documents between large-scale technical publishing applications.

The use of SGML depends upon agreement on a document type definition. Such document type definitions are separate from the three ISPs listed above but are not specified in the same format. Once standards and conventions in this area are established, X/Open hopes to publish a description of such SGML standard document type definitions in a format comparable to the three ISPs listed above.

2.3 Endorsed Document Models

An application program using the X/Open Document API for document processing must assume one of the following document models:

- ISP FOD11
- ISP FOD26
- ISP FOD36
- SGML/CALS - document type definition(s) to be identified
- other models that X/Open may define in the future.

X/Open intends to extend the reference model to add intrinsic support for spreadsheets in documents, EDI (with messages adhering to the EDIFACT standard and UN-registered message types), and the emerging STEP standard for CAD/CAM.

X/Open intends to prepare materials that help application writers understand which X/Open document model or models are suitable for each application they intend to write.

2.4 Method of Adding Document Models

The architecture described in this chapter allows X/Open to define additional document models with minimal impact on existing X/Open application programmes.

All X/Open documents have the same implicit document structure (see Chapter 4 on page 11). This is true no matter what document model the application writer selects. A document is modelled as a structured set of objects.

Each X/Open document model listed above, and any document model that X/Open subsequently publishes, involves constraints. These are rules regarding the permitted use of document objects.

Thus, an additional X/Open document model would involve specification of new objects, object elements, and constraints. It would not result in any changes to currently-specified data structures, nor change the syntax or effect of any service in the X/Open Document API when applied to currently-specified objects.

2.5 Typical Constraints

A given document model may impose the following constraints on the use of document objects, among others:

- object O1 that X/Open defines may not exist in documents under this model
- object O1 may exist in documents under this model, but O1 may not contain certain elements that X/Open lists as valid elements of that object
- object O1 may contain objects of type O2, but the total number of O2 within any instance of O1 is limited to N
- object O1 may contain objects of type O2 recursively, but the nesting of O2 within any instance of O1 is limited to a depth of N levels.

Each object determines whether it meets any applicable constraint. The nature of this is described in Chapter 4 on page 11 and Chapter 5 on page 13.

Chapter 3

Object-Oriented Document Architecture

X/Open recommends the use of object-oriented programming to facilitate writing applications that can accommodate wide varieties of present and future document types. Object-oriented programming provides that:

- a given API service may perform different operations depending on the object to which the service is applied
- procedure-like knowledge about objects resides with the objects themselves, not in the library code that may implement the X/Open Document API.

This enables dynamic addition of new document models by defining new objects and new constraints.

3.1 Categories of Object Types

Applications from different disciplines may have to maintain specialised types of objects, specific to each discipline. For example, CAD/CAM documents may include circuit topologies; programming documents may include source code statements. However, many types of information are common to a large number of disciplines. For example, text documents, multimedia documents, image documents, and hypermedia linkages may all make use of paragraph-like text objects.

Effective communication between different applications requires a large common repertoire of objects with standard semantics, plus standard generic and discipline-specific information types. In addition, many applications need to be able to create their own private types of information object.

There are thus three categories of object types:

- a core set of generic object types that are common to a wide range of applications and are allowable in all documents that follow the X/Open reference model for documents
- discipline-specific object types, specified by X/Open but not supported in some documents - notably, discipline-specific object types for particular disciplines and for document models that X/Open has not yet published
- application-specific objects not specified by X/Open. The reference model must provide for these objects because new, application-specific object types will continue to emerge as technology advances.

3.2 Unrecognised or Unprocessable Objects

Making provision to accommodate an ever-expanding number of document objects means that:

- an application may encounter objects that it does not recognise or have a way to process meaningfully
- an application may produce, or use internally, objects that cannot be carried by the chosen interchange format
- an application may declare its support of a given X/Open document model for which some objects cannot determine whether they satisfy the constraints
- in passing information from one application to another application of a different type, information may be lost or may lose fidelity. For example, information that function as a command at the source application might, at the destination application, preserve most of its visual appearance but not function as a command.

X/Open hopes to minimise these problems in the following ways:

- by defining a large number of "core" object types, X/Open hopes to increase the likelihood that a large number of applications use the same object in the same way
- by endorsing object-oriented programming, X/Open envisages that an intelligent response to such problems will be stored within the object itself. For example, each object contains knowledge about useful "fallback" representations of its information when it is transported to more limited document models.

3.3 Example

Suppose an author uses a desktop publishing application to write a book. The application uses the X/Open Document API to create text and graphic objects and arrange them in a structure based on the author's commands. At this point, the application is solely responsible for the structure, contents, and appearance of the book. The application has not necessarily selected an X/Open document model.

Writing the book to storage or to a communication interface requires that the application select an interchange format. Selection of an X/Open document interchange format requires the book to conform to the corresponding X/Open document model.

When the application makes this selection, it puts appropriate constraints into effect. If the selection was made before the author began writing the book, the author's selection of document objects would have been constrained, and the resulting book is known to be interchangeable. However, if the selection is made only after the book is written, each object in the document must be checked against the constraint, in order to verify that the document is suitable for the selected interchange.

If some objects then fail the constraints, the information they contain must be transformed into a form that meets the constraints before the document can be interchanged.

3.4 Rationale for an Object-Based Architecture

The X/Open concept of document objects does not mean that X/Open simply defines data types and specifies them as a list of parameters in functions. The problem with this traditional, procedural approach is that the industry's concept of a document is only now maturing, and will continue to evolve as customers require new forms of information to be included in documents. For example, a document may include audio, video, or live spreadsheets, and the document user may want to "edit" any or all of these components.

Object-oriented programming enables this growth with a minimum of recoding of existing applications. An object includes methods, which are essentially procedures for operating on the object. An X/Open document application can accommodate object types that were unknown at the time the application was written. The object itself carries with it methods that relate it to the application.

3.5 Typical Object Methods

Typically, an X/Open document object will have methods that:

- describe its attributes and capabilities to the application
- produce a visual representation of the information in the object
- given some X/Open document model that the application supports, produces a rendition of the information in the object using only the standard object and element types X/Open specifies for that document model
- test the information in the object to see whether it meets a specified constraint
- let the user manipulate the information in the object directly, without involvement of the application program.

Chapter 4

Implicit Document Structure

The X/Open Document API presents documents, and information within documents, to the application as a structured set of objects, adhering to the following description. X/Open does not specify or restrict how a compliant implementation represents these data structures internally, although an internal storage architecture with similarities to this document structure may result in implementation efficiencies.

The application perceives the following pieces of information in a document:

- Objects

  Application objects may be either:

  — generic (core) objects

  — discipline-specific objects

  — instances of an application-defined type.

  All such objects are structured aggregates of elements, defined below. For example, the objects include:

  — groups or sequences of content elements of a particular content type - either text, pictures, or other media types

  — attributes describing characteristics of the object

  — pointers to other objects.

  Each object contains a structured list of elements. The X/Open Document API allows the application to add elements to, or remove elements from, objects in a way that is consistent with the definition of the object type.

- Elements

  An object is made up of elements. The intention is to define a large set of element types. Generic object types and discipline-specific object types may use only these standard elements. However, any application can define other elements for use by those objects it defines.

  A partial list of element types is:

  — content portion elements (O1 contains content "...")

    This element carries content of a single standardised type (such as text or graphics) to be associated with an object.

  — pointer (structure) elements

    Pointer elements refer to an object elsewhere in the document. Pointer elements in a document define the structure of the document.

    Container elements (O1 contains O2, O3, ...) enable a hierarchical structure of objects in a document. For example, sections may contain subsections; subsections may contain paragraphs; and so on.

    Other pointer elements have varying semantics. For example, O1 with hyper-link point to O2 helps the document user quickly move to the specified destination in the document, whereas O1 with cross-reference to O2 may instead retrieve text from the specified

destination (for example, a referenced section number or name) and make it appear at the location of O1.

— attribute elements

O1 has semantic S1, S2, ...
: A semantic attribute element describes what the object is (how it behaves).

O1 is formatted by ...
: A formatting attribute element identifies an X/Open reference formatting specification that describes how the object is to be formatted within the document.

O1 has appearance attributes ...
: An appearance attribute element specifies the object's layout and presentation.

The original version of O1 is ''...''
: An original version attribute element keeps track of alternate versions of an object. For example, when an object is received using an interchange format, and an application reads it in assuming an X/Open document model which cannot faithfully represent the total information in the object, this element refers to the original version of the object, permitting restoration of the original at some later time.

Application-defined elements
: Applications may define private object types and private elements. In particular, private objects may use non-standard content types, semantic specifications, formatting specifications, and appearance attributes. When an application defines a private type for an object, attribute, or element, it may also specify an alternate representation in terms of standard types.

- Views of Objects

  A single object may contain alternate representations using different sets of elements. Each set is called a view. One use of multiple views for an object is to describe how a single object conforms to different constraints (see below).

- Constraints

  Constraints specify the elements that can be used in certain object types, the permitted structure of certain objects, and linkage among objects. In addition, constraints specify how a document is formed from a set of objects. A set of constraints describe a single standard interchange format. For example, if a set of objects conforms to the set of constraints X/Open publishes for the X/Open FOD36 document model, then those objects constitute a valid document under that X/Open document model.

  By using views, a single set of objects may conform to many different, and perhaps inconsistent, constraints.

Chapter 5

Document API Service Overview

5.1 Goals and Requirements

The major objectives of the X/Open Document API are to:

- provide a large set of standard generic object types and a large set of standard discipline-specific object types
- let applications define private object types
- let X/Open define constraints that describe the interchange formats that it endorses
- let X/Open define other constraints and object types to extend the applicability of the X/Open Document API to business graphics, EDI, and spreadsheet standards, as these emerge
- let applications be constrained to choose to define only generic or generic and discipline-specific object types, and to conform to one or more sets of constraints.

The Document API should also facilitate mapping onto conventions that are widely accepted in the industry, such as the CDA™ (Compound Document Architecture) information structures. This requirement means that the API must:

- include a wide range of object types and attributes that allow documents to be represented without information loss in the application object space
- let such documents be encoded using as many different standards as possible
- let the application test an object for adherence to constraints that represent these standards.

The API minimises the visibility of particular interchange formats and information representations, replacing them with a generalised system of object types. The API therefore does not favour one format over another nor preclude adding new standard formats into the API. The restrictions imposed by a given format are codified as constraints on objects.

5.2 Document API Operations

Building on the object, element, and constraints described earlier, the X/Open Document API provides the following abilities:

- Reading and Writing of Standard Encoded Files

  Converting a document, received by an interchange format which X/Open endorses, into a structured set of objects is a single API call. Similarly, writing a set of objects (or some view of it) as a document under an X/Open interchange format is a single API call. If the document already complies with the constraints imposed by that interchange format, the application does not need to do any extra work.

- Creating, Deleting, and Cloning Objects

  The application can use the API to create and manipulate objects, including creation of an object that duplicates some other object.

- Element Manipulation

  The application can add elements to, or delete elements from, an object. For generic or discipline-specific object types, only certain manipulations are permitted.

- Defining New Objects, Elements and Constraints

  The application can extend the object space, and describe whether those extensions meet particular sets of constraints.

- Applying Constraints

  The application can apply constraints to single objects or groups of objects - for example, to restrict a document to a form that can be conveyed over an X/Open interchange format. The application has two options:

  — the application can invoke constraints to remain implicitly in force, checked at the end of any API call that creates or modifies objects. In this case, the set of objects under the application's control is always in conformance with the constraints, and can be sent using an X/Open interchange format at any time.

  — the application may test constraints only on command. This allows the flexibility of creating non-conforming objects, but may require more work later to make those objects conform to the constraints. Information may be lost or may lose fidelity when the constraints are finally applied.

  Constraint checking may be interactive, to let the user specify fallback representations and thus control information loss.

- Using Constraint Rules to Guide Transformations

  When an object space fails to meet a constraint, the object (or some view of it) must be modified. The API includes a validation operation to confirm adherence to a specified set of constraints. This service tells the application which objects do not meet the constraints, and why. This allows non-conforming objects to change to come into conformance.

- Object Queries

  The API allows an application to query objects in order to find information that may guide its operations on them. For example, to format a specific object, the application may make a query to determine the object's parent. From that information, it may make another query to find out formatting contexts such as indentations or fonts that the parent object defines.

Chapter 6

Example: ODA to CALS SGML

Consider an application written to translate an ODA Level 3 DAP encoded document into a particular CALS SGML DTD. The X/Open API for document processing implicitly performs most of the required work. The program consists of the following steps:

1. Read in the ODA document and create an object space that matches the document. This is a single operation in the API.
2. Create a new view, perhaps called CALS, that contains the same objects and elements as the existing (ODA) objects.
3. Apply the CALS constraints to the new view. The CALS constraints are a predefined part of the API, so this step is also just a single operation in the API.

   If the objects meet the constraints, then the translation is done, and the program proceeds to Step 5.
4. For each object that does not pass the CALS constraints, take some specific action with that object to bring it into conformance. Here the application delivers its added value.

   The application defines various procedures for dealing with different constrain failures. For features supported by ODA but not CALS, the application may delete from the CALS view some objects or elements. For information types supported by ODA but not CALS, the application may translate some fallback representation defined by the objects themselves. For example, an ODA-processable object might in the CALS view, become a formatted object.
5. Finally, write the conforming CALS view of the object space as a CALS document in a single operation.

Chapter 7

Implementation Possibilities

Vendors of an X/Open-compliant Document API may implement additional APIs for document processing. A call to a service in the X/Open Document API may map to one or more calls in these other APIs in an implementation-dependent manner.

Two other levels of API are typical:

- an object-oriented API that allows manipulation of objects of a specific document interchange model. Examples include APIs for access/manipulation of documents adhering to the ODA FOD profiles or to SGML CALS profiles.
- a procedural API that allows manipulation of actual datastream elements of a specific interchange format. Examples are ODIF elements, MicroSoft Rich Text Format (RTF), and other proprietary formats. An API for a proprietary interchange format might be provided by the owner of the format.

X/Open may specify one or more APIs. Use of the lower-level APIs by an application program might result in more efficiency in a narrowly-defined task, such as generating or receiving a document in one specific interchange format. However, a general-purpose application should confine itself to the object-oriented API presented in this document in order to be adaptable to newly-invented document types, document objects, and interchange formats.

The X/Open Document API is flexible enough to manipulate documents received from the interchange formats that X/Open endorses. The exact method of implementing different views of the same application object space involves an object-oriented approach using multiple inheritances. This makes it possible to create appropriate document objects from all these interchange formats and vice versa.

Appendix A

Glossary

API
Application Programming Interface

attribute
A defined property of an entity or object, usually defining a characteristic feature. An attribute is named, and is often inplemented as fields with values.

CALS
Computer-aided Acquisition and Logistics Support. A US Department of Defense information technology scheme to ensure open systems facility to interchange information between computer systems, for the purpose of integrating all aspects of procurement, manufacturing and maintenance information.

CCITT
Comite Consultatif Internationale de Telegraphique et Telephonique. A United Nations agency which makes recommendations regarding telecommunications. Its technical recommendations often become internationally recognised standards (V-series, X-series).

EDI
Electronic Data Interchange. The electronic exchange of structured information between computer systems.

EDIFACT
EDI for Administration, Commerce and Transport

ISO
International Organization for Standardization. Body made up of committees of representatives from the standards bodies of member countries, with particular interest in electronic interchange of information.

ISP
ISO Internationally Standardized Profile

Object-oriented
Software approach in which everything (processes, files, etc.) is represented as objects. Objects are data structures in memory that may be manipulated by the system; communication between objects is achieved by passing messages. In this regard, an object is a package of information and a description of its manipulation, and a message is a specification of one of an object's manipulations.

ODA
Office Document Architecture. More generally called Open Document Architecture, it is an architecture of documents which is defined in a set of ISO standards (8613), aimed at enabling interchange of information between computer systems.

SGML
Standard Generalized Markup Language. A family of ISO standards for labeling electronic versions of text, enabling both sender and receiver of the text to identify its structure.

Index