DEC ODA Compound Document Architecture (CDA) Gateway

User's Guide

Order Number: AA–PB6KA–TE

April 1991

This guide describes how to use the DEC ODA Compound Document Architecture (CDA) Gateway.

Revision Update Information:	This is a new guide.
Operating System and Version:	VMS Version 5.4
Software Version:	DEC ODA Compound Document Architecture (CDA) Gateway Version 1.0

Digital Equipment Corporation
Maynard, Massachusetts

April 1991

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copiec only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

Restricted Rights: Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computeı Software clause at DFARS 252.227-7013.

The postpaid Reader's Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

This guide is available on CDROM.

This document was prepared using VAX DOCUMENT, Version 1.2.

Contents

Figures

'ables

Preface

This guide describes how to use the DEC ODA Compound Document Architecture (CDA) Gateway.

> **Note**
>
> The DEC ODA Compound Document Architecture (CDA) Gateway is referred to as the ODA CDA Gateway in this guide.

’urpose of This Guide

The purpose of this guide is to describe how to use the ODA CDA Gateway to interchange documents between the Office Document Architecture (ODA) and the Compound Document Architecture (CDA).

Vho Should Use This Guide

This guide is intended for users who want to:

- Convert an ODA document to CDA
- Convert a CDA document to ODA
- Get an overview of ODA and CDA

This guide assumes that readers are familiar with basic operating system concepts and with the operation of a command line interface. Some knowledge of compound document processing is helpful but not required.

Structure of This Guide

This manual has four chapters, an appendix, a glossary, and an index.

- Chapter 1 gives an overview of ODA and CDA, and the ODA CDA Gateway
- Chapter 2 describes how to use the ODA CDA Gateway to interchange documents between ODA and CDA.
- Chapter 3 describes the main principles of CDA.
- Chapter 4 describes the main principles of ODA.
- Appendix A lists and explains the messages issued by the ODA CDA Gateway software.
- The Glossary defines some important terms used in this guide.

Associated Documents

For information about installing the ODA CDA Gateway, see the *DEC ODA Compound Document Architecture (CDA) Gateway Installation Guide*.

Related Documents

For more information about ODA, see the ODA standard, *ISO 8613, Information Processing—Text and Office Systems—Office Document Architectur (ODA) and Interchange Format.*

For more information about CDA, see the following:

- *Using the CDA Converters*
- *Technical Guide to the CDA Converters*
- *CDA Reference Manual*

Conventions

The following conventions are used in this guide:

Convention	Description
[]	In command and option descriptions, brackets indicate optional elements.
-	Hyphens in command descriptions indicate that the command is continued on the next line.
Note	A note contains information that is of special importance to the reader.
`Color`	Text in color indicates user input.
UPPERCASE	File names and command names in text are shown in uppercase.
boldface type	New terms, when used for the first time, are shown in bold. New terms are defined in the Glossary.
italic type	Italic type indicates attribute names and the complete titles of other documents. In command descriptions, information that can vary is shown in italics, for example, *output-file*.
sans-serif type	Sans-serif type indicates attribute values.

1 ODA CDA Gateway Overview

This chapter briefly describes the ODA CDA Gateway and its components. It describes the following:

- Office Document Architecture (ODA)
- Compound Document Architecture (CDA)
- CDA Converter Architecture
- ODA CDA Gateway

The ODA CDA Gateway allows the interchange of documents between ODA and CDA, which means you can process ODA documents, using CDA applications.

1.1 Office Document Architecture

ODA is an international standard[1] that defines a structure for **compound documents** and a way of interchanging them between proprietary systems. Compound documents contain text, graphics, raster images, style and structure information, and other data, for example, sound.

ODA was originally intended for the creation and interchange of simple office documents such as memos, letters, and reports. However, its scope has been expanded and ODA now provides comprehensive facilities for describing the structure and content of complex compound documents.

ODA stores documents in the Office Document Interchange Format (ODIF), which is also defined in the ISO 8613 standard. For more information about ODA, see Chapter 4.

[1] The standard that defines ODA and the ODIF interchange format is *ISO 8613, Information Processing—Text and Office Systems—Office Document Architecture (ODA) and Interchange Format.*

1.2 Compound Document Architecture

CDA is Digital's compound document architecture. CDA allows compound documents to be easily interchanged between conforming applications on any system in a network.

CDA stores documents in the Digital Document Interchange Format (DDIF) and the Digital Tabular Interchange Format (DTIF). For more information about CDA, see Chapter 3.

1.3 CDA Converter Architecture

The ODA CDA Gateway is based on the CDA Converter Architecture, which defines a two-stage method of converting compound documents between any two formats. The first stage is to convert the input document to the CDA format. The second stage is to convert the CDA document to the output format.

The CDA Converter Architecture has the following components:

- Front end

  The front end reads an input file of a particular format and converts the file to the CDA format.

- Back end

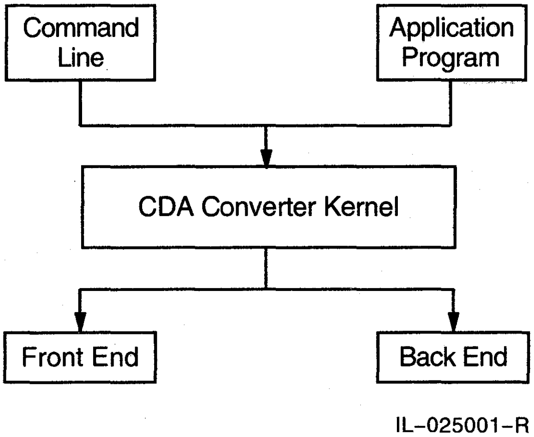
  The back end reads a CDA document and converts the document to an output file of a different format.

- CDA Converter Kernel

  The CDA Converter Kernel performs generic conversion functions (functions that must be performed for every document conversion such as reading and writing files). The CDA Converter Kernel also invokes the front and back ends for the input and output formats.

Figure 1–1 shows the components of the CDA Converter Architecture.

Figure 1–1 CDA Converter Architecture Components



For more information on the CDA Converter Architecture and on CDA converters, see the *CDA Reference Manual,* the *Technical Guide to the CDA Converters,* and *Using the CDA Converters.*

1.4 ODA CDA Gateway

The ODA CDA Gateway converts processable ODA documents to the CDA format, and converts CDA documents to ODA documents that belong to the processable document architecture class. For more information about processable documents, see Section 4.1.

The architecture of the ODA CDA Gateway is based on the CDA Converter Architecture, and uses the CDA Converter Kernel.

The ODA CDA Gateway has the following components:

- ODA front end

  The ODA front end converts an ODA document to the CDA format.

- ODA back end

  The ODA back end converts a CDA document to the ODA format.

- CDA Converter Kernel

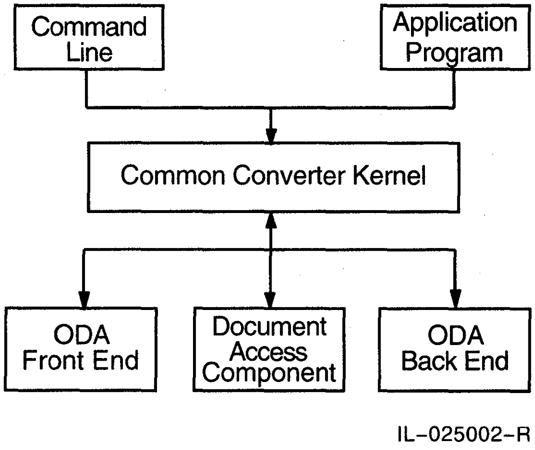
  This is the same converter kernel that is part of the CDA converter architecture.

- Document Access Component (DAC)

  This component reads, writes, and manipulates ODA documents. It is used by both the front end and the back end.

Figure 1–2 shows the components of the ODA CDA Gateway architecture.

Figure 1–2 ODA CDA Gateway Architecture Components



2

Using the ODA CDA Gateway

This chapter describes how to invoke the ODA CDA Gateway to interchange documents between ODA and CDA. It contains information about the following:

- How to convert CDA documents to ODA
- How to convert ODA documents to CDA
- Processing options that you can use to control the conversion
- Options files, in which you specify the processing options to be used during conversions
- Data mapping between ODA and CDA, and possible discrepancies that can arise during conversions

2.1 Converting ODA Documents to CDA

Use the CONVERT/DOCUMENT command to convert an ODA document to CDA, as follows:

```
$  CONVERT/DOCUMENT[/OPTIONS=options-file] -
_$  input-file/FORMAT=ODA output-file[/FORMAT=DDIF]
```

The *input-file* and *output-file* parameters specify the names of the ODA input file and the CDA output file respectively.

The OPTIONS qualifier specifies an options file that contains the processing options to be applied during the conversion. For more information on processing options and options files see Section 2.3 and Section 2.4.

2.2 Converting CDA Documents to ODA

Use the CONVERT/DOCUMENT command to convert a CDA document to ODA, as follows:

```
$  CONVERT/DOCUMENT[/OPTIONS=options-file] -
_$  input-file[/FORMAT=DDIF] output-file/FORMAT=ODA
```

The *input-file* and *output-file* parameters specify the names of the CDA input file and the ODA output file respectively.

The OPTIONS qualifier specifies an options file that contains the processing options to be applied during the conversion. For more information on processing options and options files see Section 2.3 and Section 2.4.

2.3 Processing Options

When interchanging documents between ODA and CDA, you can use processing options to control how the ODA CDA Gateway processes an input file and creates an output file. For example, you can choose the page size for an output file. You specify processing options, using an **options file.** The options file contains a list of the processing options to be applied during a conversion. Section 2.4 describes how to create an options file.

2.3.1 Front End Processing Options

You can use the following processing options with the ODA front end:

IMAGE[=filename.IMG]
Specifies the name of the CDA file used to store raster image data in the ODA document.

ODA documents store raster image data as part of the document. However, the ODA front end stores each raster image in a separate file. The IMAGE option allows you to specify the names of the image files that the ODA front end creates when it converts the ODA document to CDA. The ODA front end creates a separate CDA file for each image in the ODA document and inserts references at the appropriate points in the main CDA document.

Storing raster images in separate files allows you to process the images, using image-handling software such as DECpaint.

The ODA front end generates a unique file name for each raster image file name by appending a number to the specified file name. For example, if you specify IMAGE=PHOTO.IMG, the front end writes the first ODA raster image object to a file called PHOTO_0.IMG, the second to a file called PHOTO_1.IMG, and so on.

If you do not specify a file name, the ODA front end uses the file name of the ODA document to create file names for the raster image objects. For example, if the file name of the ODA document is REPORT.ODA, the ODA CDA Gateway writes the first raster image object to a file called ODA$REPORT_IMAGE_0.IMG, the second to a file called ODA$REPORT_IMAGE_1.IMG, and so on.

[NO]LOG[=filename.LOG]

Specifies whether the ODA front end creates a log file that contains messages issued during the interchange. If you specify LOG without a file specification, the ODA front end directs messages to the default standard error location. The default standard error location is usually the terminal screen, although it can be redefined to be a file.

The log file contains a listing of all the messages issued by the ODA front end. If you specify a file name and do not specify the file type, then the file type is .LOG by default.

The ODA front end creates the log file in your current default directory unless you specify otherwise. If you specify NOLOG, you do not see any messages unless errors occur during the conversion. See Appendix A for explanations of the messages issued by the ODA front end.

If the ODA front end encounters errors before it processes the options file, it writes the messages to the standard error location.

The default is LOG.

PAGE=page_size

Specifies the page size of the CDA document created by the ODA front end. The page size can be A3, A4, or NAL (North American Letter).

The PAGE option is useful when you are converting ODA documents that conform to the Q121 DAP, because these documents do not contain page size information. If the ODA document conforms to a DAP that specifies a page size, the page size in the document profile supersedes the value specified by the PAGE option.

If the ODA document does not specify a page size, and you omit the PAGE option, the default page size is A4.

2.3.2 Back End Processing Options

You can use the following processing options with the ODA back end:

CHARACTERSET=character_set_name

Specifies the character set for the ODA document created by the ODA back end. You can specify either the ISO Latin1 character set, or the ISO 6937/2 character set.

The character_set_name parameter must have one of the following values:

6937	Use the ISO 6937/2 character set.
LATIN1	Use the ISO Latin1 character set.

The default is 6937.

[NO]COMPRESSION[=compression_type]

Specifies the image compression scheme that the ODA back end uses to encode the raster images that it creates from the raster images in the CDA document.

The compression_type parameter controls the image compression scheme that the ODA back end uses when it converts raster images in the CDA document to raster images in the ODA document, as follows:

BITMAP
The ODA back end does not use image compression, it converts the CDA raster image to an ODA raster image, or bitmap.

G31D
The ODA back end converts CDA raster images to ODA images that use the **CCITT Group 3** one-dimensional encoding.

G32D
The ODA back end converts CDA raster images to ODA images that use the CCITT Group 3 two-dimensional encoding.

G42D
The ODA back end compresses CDA raster images to ODA images that use the **CCITT Group 4** two-dimensional encoding.

If you do not specify the COMPRESSION option, the ODA back end converts CDA raster images to ODA bitmaps. If you specify the COMPRESSION option without specifying a value for the compression_type parameter, the default value of the parameter is G42D.

DAP=dap_name

Specifies the document application profile (DAP) to which the ODA document created by the ODA back end will conform. See Section 4.5 for more information about DAPs.

The dap_name parameter must have one of the following values:

Q111
Create an ODA document that conforms to the Q11 DAP.

Q112
Create an ODA document that conforms to the Q112 DAP.

Q121
Create an ODA document that conforms to the Q121 DAP.

The default is Q112.

[NO]LOG[=filename.LOG]
Specifies whether the ODA back end creates a log file that contains messages issued during the interchange. If you specify LOG without a file specification, the ODA back end directs messages to the default standard error location. The default standard error location is usually the terminal screen, although it can be redefined to be a file.

The log file contains a listing of all the messages issued by the ODA back end. If you specify a file name and do not specify the file type, then the file type is .LOG by default.

The ODA back end creates the log file in your current default directory unless you specify otherwise. If you specify NOLOG, you do not see any messages unless errors occur during the conversion. See Appendix A for explanations of messages issued by the ODA back end.

If the ODA back end encounters errors before it processes the options file, it writes them to the standard error location.

The default is LOG.

!.4 Options Files

An options file contains a list of the processing options to be applied when converting an ODA document to CDA, or vice versa. Each line of the options file begins with one of the following keywords, followed by a processing option:

- ODA

  The options specified on the line apply both to ODA documents being converted to CDA and to CDA documents being converted to ODA.

- ODA_INPUT

  The options specified on the line apply only to ODA documents being converted to CDA.

- ODA_OUTPUT

  The options specified on the line apply to CDA documents being converted to ODA.

Options files have a default file type of .CDA$OPTIONS.

Each line in the options file consists of a keyword and a processing option. The keyword and the processing option are separated by a slash (/), or by one or more tabs or spaces.

Processing options can contain uppercase and lowercase alphabetic characters (the case is not significant), numerals (0–9), dollar signs ($), and underscores (_). If an option requires a value (for example, a file specification), the value follows the processing option and is separated from the option by spaces, tabs, or an equal sign (=).

If you want to specify several options, you must place each option on a separate line in the options file. Each line in the options file can be preceded by spaces and tabs and can be terminated by any character except those used in the processing options.

The ODA CDA Gateway checks the validity of the processing options you specify, and if they are valid, it applies them during the conversion. If you specify an invalid option, the ODA CDA Gateway returns an error message. Messages are written to the default standard error location unless you specify a log file. Message log files have a default file type of .LOG.

You can specify the same log file name for the input log file and the output log file. However, the ODA CDA Gateway writes messages to different versions of the file (which have the same file name), rather than to the same file.

If you do not create an options file or do not specify input or output log file names in the options file, messages concerning your input and output files are written to the standard error location.

The following is an example of an options file:

```
ODA              LOG=REPORT.LOG
ODA_OUTPUT       COMPRESSION=G32D
```

This options file contains the following information:

- A log file called REPORT.LOG is to be produced when converting ODA documents to CDA and when converting CDA documents to ODA.
- Raster images in the ODA document will be compressed according to the CCITT Group 3 compression scheme.

2.5 Data Mapping

The ODA CDA Gateway interchanges ODA and CDA documents by mapping data between the two architectures. To create an ODA document containing the content of the CDA document, the ODA CDA Gateway maps the content of the CDA document segments to ODA document objects. Similarly, to create a CDA document containing the content of the ODA document, the ODA CDA Gateway maps the content of the ODA document objects to CDA document segments.

For more information about CDA, see Chapter 3. For more information about ODA, see Chapter 4.

2.5.1 Mapping from ODA to CDA

When you invoke the ODA CDA Gateway to convert an ODA document to CDA, the ODA front end maps ODA document components as follows:

- Document profile

  The ODA front end reads the information in the ODA document profile and stores it in the CDA document header.

- Document structure

  The ODA front end uses the structure of the ODA document to create the structure of the CDA document.

- Document content

  The ODA front end maps ODA document content to CDA as follows:

  - Text content

    If the ODA document contains text content, the ODA front end stores the content as CDA text content.

  - Raster image content

    If the ODA document contains raster image content, the ODA front end converts the content to CDA, stores the content in a separate file, and inserts a reference to the file in the CDA document.

  - Graphics content

    If the ODA document contains geometric graphics content, the ODA front end converts the content to CDA and stores the content as CDA graphics content.

2.5.2 Possible Discrepancies During ODA Front End Conversion

The ODA front end converts, without any data loss, all text, graphics, and raster image content in an ODA document from ODA to CDA. However, the ODA CDA Gateway does not support the following features of the ODA architecture:

- Text synchronization

  The pages in an ODA document can contain two or more text blocks in parallel. The page layout can specify that the text blocks are synchronized so that they move as one block. The ODA CDA Gateway does not support text synchronization, and it does not convert the text synchronization information to CDA. However, the ODA front end converts the text content of the blocks to CDA without any data loss.

- Clipping of images

  The raster images in ODA documents can have a clipping rectangle that encloses the part of the image to be displayed. The ODA CDA Gateway does not support clipping rectangles, and it does not convert the clipping rectangle information to CDA. However, the ODA CDA Gateway does convert the raster image to CDA, and the resulting CDA document displays the whole image.

- Transparency of frames

  The frames in ODA documents can be transparent or opaque. The ODA CDA Gateway does not support opaque frames. It converts all opaque frames in the ODA document to transparent frames in the CDA document when it converts the ODA document to CDA.

- Frame borders

  The frames in ODA documents can have different border patterns on different sides. The ODA CDA Gateway does not support this feature, and it does not convert variable border patterns to CDA. However, the ODA CDA Gateway converts the content of the ODA frames to CDA without any data loss.

- Variable frames

  The most basic frames in the layout of an ODA document can be either fixed or variable. A variable frame has variable size and position, both of which depend on the content of the frame (and preceding frames). The ODA CDA Gateway does not support the conversion of variable frames. Instead, it produces a fixed-size frame whose dimensions are equal to the maximum allowed dimensions of the variable frame and whose position is controlled by the position of preceding frames.

- Character spacing

  ODA documents can specify variable character spacing (pitch). The ODA CDA Gateway does not support this feature, and it does not convert pitch information to CDA. However, the ODA CDA Gateway converts all of the text content to CDA without any data loss.

- Complex ODA page layouts

  If the body frame area of an ODA page layout contains more than one type of frame object, the ODA front end converts every frame in the page to a CDA document frame. The CDA document frame corresponds to the first type of frame that occurs in the ODA document. For example, if the ODA page layout contains a frame that has the attribute SingleColumnFrame followed by several frames with the attribute SnakingColumnsFrame, the ODA front end converts all of the frames to the CDA frame type that corresponds to the single column frame.

- Alphabetic and Roman page numbers

  If a page number in an ODA document contains uppercase or lowercase alphabetic characters or Roman numerals, the ODA front end will not convert the page number to CDA.

- Character sets

  ODA documents can contain text in any internationally-registered character set. However, the ODA CDA Gateway supports only the following character sets:

  - ISO 6937/2
  - ISO Latin1
  - ISO Latin2
  - ISO Latin Arabic
  - ISO Latin Greek
  - ISO Latin Hebrew

Note

If you convert an ODA document to CDA to use it with the DECwrite document processing application, the only text that will be fully intelligible in the DECwrite document is text in either the ISO Latin1 or the ISO 6937/2 character set. Text in the other character sets supported by the ODA CDA Gateway will be converted to CDA, but will not be displayed correctly in the DECwrite document. However, when you convert the DECwrite document to ODA with the ODA back end, the text will be intact.

If the ODA document contains text in other character sets, the ODA CDA Gateway does not convert the character set information to CDA. However, it inserts the characters as ISO Latin1 characters in the CDA document. These characters will not be fully intelligible in the CDA document.

- Uncompressed raster images

  Compressed raster images in ODA documents can use the Consultative Committee for International Telephony and Telegraphy (CCITT) Group 3 or Group 4 compression schemes. When the ODA front end converts an image from ODA to CDA, it identifies the compression scheme and decompresses the image accordingly. However, some images that use these compression schemes can also contain chunks of uncompressed data. The ODA front end cannot decompress these images so these images do not appear in the CDA document.

2.5.3 Mapping from CDA to ODA

When you invoke the ODA CDA Gateway to convert a CDA document to ODA, the ODA back end maps CDA document components as follows:

- Document profile

  The ODA back end creates an ODA document profile in which it stores information about the ODA document that it creates. See Section 4.4 for more information about document profiles.

- Document structure

  The ODA back end uses the structure of the CDA document to create the structure of the ODA document. The structure of the ODA document satisfies the requirements of the DAP to which the ODA document is to conform.

- Document content

  The ODA back end stores the content of the CDA document as the content of the ODA document. The ODA back end maps CDA document content to ODA as follows:

  - Text content

    If the CDA document contains text content, the ODA back end stores the content as ODA text content.

  - Raster image content

    If the CDA document contains raster image content, the ODA back end converts the content to ODA and stores the content as ODA raster image content.

- Graphics content

  If the CDA document contains geometric graphics content, the ODA back end converts the content to ODA and stores the content as ODA graphics content.

2.5.4 Possible Discrepancies During ODA Back End Conversion

If all of the features of a CDA document can be represented within the constraints specified by the DAPs that are supported by the ODA CDA Gateway, the ODA back end converts the CDA document to ODA without any data loss. If the CDA document contains some information that is more complex than that allowed by these DAPs, this information will not be converted. Instead, the ODA back end issues a message informing you of the expected data loss. See Section 4.5 for more information about the DAPs supported by the ODA CDA Gateway.

ODA does not support the following features of CDA:

- LiveLink connections

  CDA documents can contain LiveLink connections, but ODA documents cannot contain LiveLink connections. The ODA back end ignores LiveLink connections when it converts CDA documents to ODA. As a result, the information connected to a CDA document by LiveLink connections will not be converted and will be missing from the ODA document.

  See Chapter 3 for information about LiveLink connections.

- Support for the DTIF format

  If the CDA document contains some information that is stored in the DTIF format, the ODA back end ignores this information when it converts the CDA document to ODA.

- Annotated images

  If the CDA document contains annotated raster images, the annotation information is lost because the DAPs supported by the ODA CDA Gateway do not support the inclusion of annotated raster images.

- Table of contents

  When the ODA back end converts CDA documents to ODA, the table of contents will be lost because the DAPs supported by the ODA CDA Gateway do not support the inclusion of a table of contents.

- Index

  When the ODA back end converts CDA documents to ODA, the index will be lost because the DAPs supported by the ODA CDA Gateway do not support the inclusion of an index.

- Raster image compression

  Both CDA and ODA support CCITT Group 3 and Group 4 raster image compression schemes. However, CDA also supports private raster image compression schemes. When the ODA back end converts CDA documents to ODA, it does not convert compressed raster images that use a private compression scheme.

- Frames in DECwrite documents

  When you are converting DECwrite documents to ODA, geometric graphics or raster images can occur only in in-line frames. Graphics or images that occur in fixed frames, or directly on the page without a frame, are part of the specific page layout of the DECwrite document page, and are not converted to ODA.

  The ODA back end converts CDA document pages that contain combinations of the following frame types to ODA without any data loss:

  - A single column body frame that fills the entire body area of the page
  - A set of snaking columns that fills the entire body area of the page (all columns must be the same height)
  - A set of columns stacked vertically in the body area (all columns must be the same width)

  If the CDA document contains any other combination of frames and frame types, the ODA back end creates a single column body frame in the ODA document. This frame contains the content of all the frames on the page.

- Empty header and footer frames

  If a page in a CDA document has a header or footer frame that does not contain document content, the ODA back end does not create the corresponding header or footer frame in the ODA document.

- Character sets

  CDA documents contain text in the following internationally-recognized character sets:

  - ISO Latin1
  - ISO Latin2

  - ISO Latin Arabic
  - ISO Latin Greek
  - ISO Latin Hebrew

  The ODA back end converts text in all of these character sets to ODA without any data loss. However, CDA documents can also contain text in Digital-specific character sets. The ODA back end cannot convert this text to ODA, and the text is lost when you convert your document from CDA to ODA.

- Typeface information

  The following typefaces are supported by the ODA CDA Gateway: Courier, Helvetica, and Times. If the CDA document that you convert to ODA is to conform to the Q112 DAP, and the text content is only in these typefaces, the ODA back end converts all the typeface information, including rendition changes (for example, from roman to italic), to ODA without any data loss.

  If the CDA document contains text in other typefaces, the ODA back end does not convert the typeface information to ODA. Instead, the ODA back end issues an error message and inserts the text content in the ODA document, but in the Courier typeface. However, the rendition changes in the CDA document are retained in the ODA document.

  If the CDA document that you convert to ODA is to conform to the Q111 or Q121 DAPs, any typeface information in the CDA document is not converted to ODA. However, the rendition changes in the CDA document are retained in the ODA document if the DAP allows such changes.

3 Compound Document Architecture

This chapter gives an overview of the Compound Document Architecture (CDA). It describes the following:

- CDA documents
- Document content and content types
- Document segments
- Document hierarchy and informational structures
- Logical and layout structures in CDA documents

3.1 CDA Documents

CDA is Digital's compound document architecture. A document created by an application that conforms to CDA can be used by all applications that conform to CDA. For example, you can use the DECchart charting application to create a chart, using data from a DECdecision spreadsheet. You can then include the chart in a DECwrite document, without modification.

Like ODA documents, CDA documents can contain text and graphics content in a single document. Unlike ODA documents, however, CDA documents can also contain LiveLink connections, that is, connections to data stored elsewhere on the same system, or even on a different system. For example, a DECwrite document can contain a LiveLink connection to a DECchart chart. When you update the chart, the DECwrite document is automatically updated.

CDA, through the CDA Converters, allows documents to include text, graphics, or tabular data, generated using other vendors' applications. For example, if you want to include a Lotus 1-2-3 spreadsheet in a CDA document, you can convert the spreadsheet to the CDA format. The CDA document contains the data from the Lotus 1-2-3 spreadsheet, and other information such as formulas, functions, and cell information. For more information on the CDA Converters, see *Using the CDA Converters* and the *Technical Guide to the CDA Converters*.

CDA stores documents in the DDIF (Digital Document Interchange Format) format and the DTIF (Digital Tabular Interchange Format) format.

3.2 Document Content

CDA documents contain several types of content. Document content can be either hard or soft. **Hard content,** for example, paragraphs of text, is entered by the creator of a document. **Soft content,** for example, a chart linked to a document, is generated by software and is subject to recalculation when the document is revised. Table 3–1 lists and describes the CDA content types that can be used in a document.

Table 3–1 CDA Content Types

Content Type	Description
Text content	Text that consists of characters from standard character sets. There are two types of text content: • Latin1 (text that is in the ISO Latin1 character set) • General (text that can be in any character set)
Graphics content	Graphics objects such as lines and curves. CDA uses the following content types to represent graphics: • Arc content—to specify arcs • Bézier content—to specify Bézier curves • Polyline content—to specify polylines • Fill Area Set content—to specify a closed area to be filled
Raster image content	Raster images such as those created by image scanners.
Computed content	Document content computed by a formatter or other document processor. Page numbers and chapter numbers are examples of computed content.

(continued on next page)

Table 3–1 (Cont.) CDA Content Types

Content Type	Description
Restricted content	Document content that is not revisable. There are two types of restricted content: • Page description language (PDL) content PDL content is for display only and is not suitable for revision. • Private content Specifies content that cannot be processed by CDA applications, for example, because the content was created by an application that does not conform to CDA. CDA applications also specify application-specific information as private content. When other applications process the document, they do not try to process the private content.
Content references	Instead of including document content directly, a CDA document can contain a content reference to point to the content. The content reference can point to any CDA content, such as a character, a word, a graphic, or even an entire document.
External content	Instead of being stored in the document, the content is stored in a separate file. The document contains a content reference that points to the file. The application includes the content of the external document when it processes the document containing the content reference.

(continued on next page)

Table 3–1 (Cont.) CDA Content Types

Content Type	Description
Directives	Formatting instructions (for example, page breaks) that control the appearance of the CDA document. There are tw types of directive: • **Soft directive** A soft directive is a directive that an application insert in the document. For example, when a formatter paginates a document, it inserts page breaks at the appropriate places. These page breaks are soft directives. • **Hard directive** A hard directive is a directive that the user inserts in a document. For example, if you insert page breaks at specific places in a document, those page breaks are hard directives.

3.3 Segments

The content of a CDA document is arranged in segments. A **segment** is a portion of document content to which a specified set of characteristics applies. These characteristics are called the **segment attributes,** and they define the presentation and processing characteristics of the segment. Segments are similar to objects in an ODA document. Segment attributes are similar to OD object attributes.

There are two types of segment:

- **Named segment**

  A named segment forms part of the logical structure of the CDA document Named segments are used to represent document structures that occur often in a document. For example, the generic logical structure of a document can define a named segment for paragraphs, because every paragraph in a document has the same basic attributes (for example, typeface, font size, and indentation). The named segment attributes are defined on the **type definitions,** which are part of the **root segment** (see Section 3.5 for more information about the root segment.)

Every segment has its own attributes that apply only to a particular occurrence of the named segment. The extra attributes are called **local** attributes. The local attributes supersede the attributes of the generic segment. For example, if a particular paragraph is in italics, it is contained in a generic paragraph named segment, but this particular segment has a local attribute specifying that the text of the paragraph is in italics.

- **Unnamed segment**

  An unnamed segment is a segment that is not based on a named segment, but has attributes that apply to the content of the segment. For example, if a sentence is in boldface type in the paragraph that is already in italics, the sentence is contained in an unnamed segment that specifies bold as an attribute.

Segments can contain other segments, called **nested segments.** A nested segment is a segment that is below its parent in the document hierarchy. For example, a chapter segment might contain a segment for each section in the chapter. The section segments are nested below their parent, the chapter segment. The nested segments inherit attributes from their parent. A nested segment can accept these inherited attributes, specify additional attributes, or overwrite the inherited attributes. Nested named segments in CDA documents are similar to subordinate objects in ODA documents.

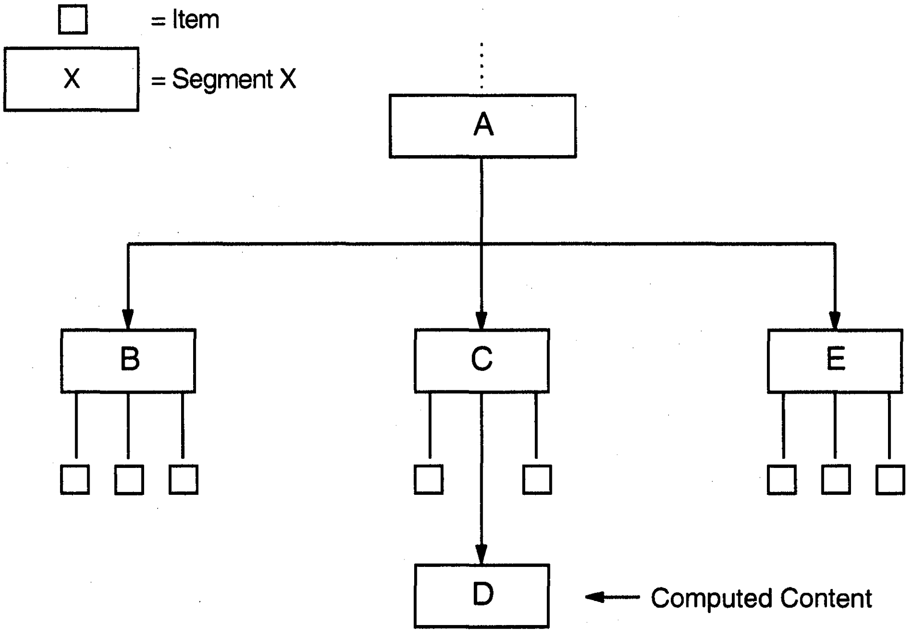
In the case of the italicized paragraph containing one sentence in bold, the segment containing the sentence in bold is nested below the segment that contains the italicized paragraph.

3.4 Document Hierarchy

CDA represents a document as an ordered hierarchy of segments. Each segment in a document contains content and nested segments. The hierarchy of segments is an inverted tree structure. CDA stores segments from the top down and from left to right. Content elements (the text, graphics, and raster images of the document) are displayed in this order.

Figure 3–1 illustrates the principle of document hierarchy.

Figure 3–1 Document Hierarchy



IL-025010-R

The segments of the document shown in Figure 3–1 are stored in the order A, B, C, D, E. In the figure, segment A has B, C, and E as its content. Segments B, C, and E have three content elements each. One of the content elements of segment C (segment D) is a nested segment containing computed content.

The hierarchy shown in Figure 3–1 is part of a CDA document structure. The rules governing the structure of a CDA document depend on the creator of the document. However, the document must begin according to a predefined structure, because each document must contain the information that allows the document to be processed by other applications. Section 3.5 describes this information.

3.5 Document Information Structures

Every CDA document must contain the following information, and it must precede any segments that contain document content:

- **Document root**

  The document root identifies the document. In the hierarchical representation of a CDA document, the document root is at the top of the inverted tree structure.

- **Document descriptor**

  The document descriptor specifies information such as the name of the product that created the document.

- **Document header**

  The document header specifies information such as the document title, author, version number, and other information that applies to the document as a whole.

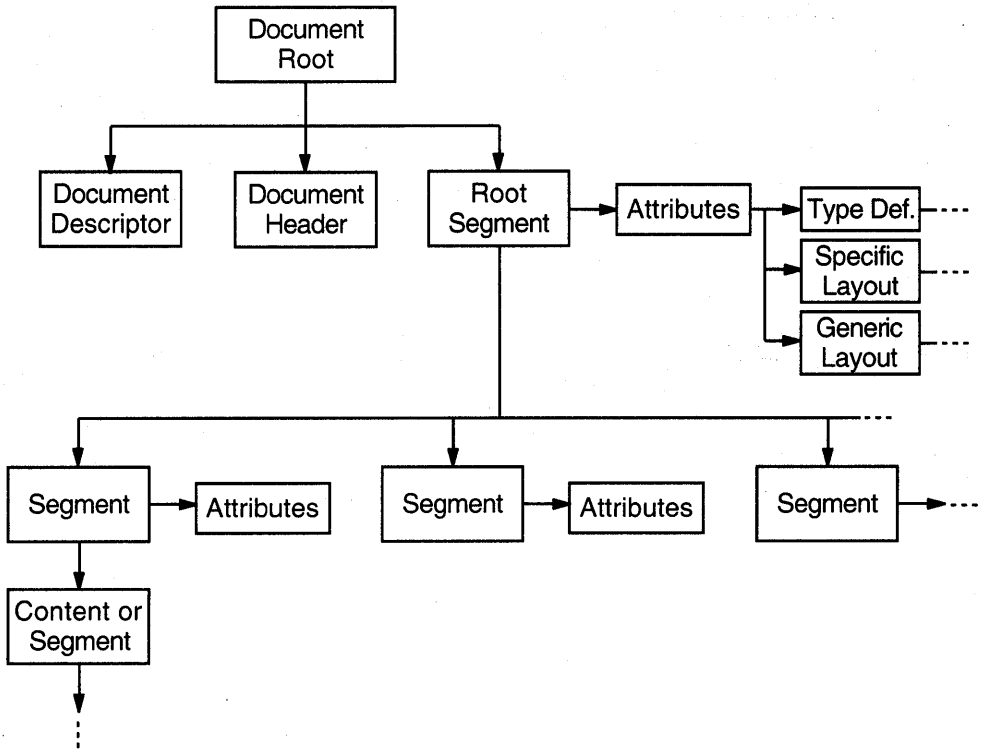
  The document descriptor and the document header combined are similar to the document profile in an ODA document.

- **Root segment**

  The document contains a **root segment** that contains all of the other document segments. That is, all segments that contain document content are nested under the root segment. Attributes that apply to the entire document are specified as attributes of the root segment. The root segment has a special set of attributes called **type definitions.** The type definitions define the named segments that are used in the document. There is one type definition for each named segment. The type definition also defines the attributes of the named segments.

Figure 3–2 shows the top-level structure of a CDA document.

Figure 3–2 CDA Document Structure



IL-025011-R

3.6 Document Structure

CDA documents have separate structures that describe their logical content and their layout. There are also structures that apply to groups of documents, or to a particular document.

3.6.1 Logical Structure

The **logical structure** of a document specifies the structure of the logical content. For example, the logical structure of a document can specify that chapters can contain one or more sections, and sections can contain one or more paragraphs, but appendixes cannot contain chapters. All documents that adhere to a particular set of these rules have the same **generic logical structure.**

A particular document conforming to this generic logical structure might have, for example, six chapters and two appendixes. The six chapters and two appendixes make up the logical structure of this particular document. This logical structure is called the **specific logical structure** of the document.

3.6.2 Layout Structure

The **layout structure** of a document specifies the physical structure of the document, for example, how pages are divided into frames, and frames in turn are divided into text or graphics blocks. For example, the layout structure of a document can specify the page size, frame width, space between frames, and so on. All documents that adhere to a particular set of these rules have the same **generic layout structure.**

A particular document conforming to this generic layout structure might have, for example, 100 pages. The 100 pages, the frames they contain, and so on make up the layout structure of this particular document. This layout structure is called the **specific layout structure.**

CDA documents describe the generic and specific layout structures of a document in the root segment. CDA applications ignore layout structures specified on the other segments.

3.7 Document Layout

Document layout describes the manner in which document content elements (graphics, text, and raster images) are arranged on a page or a series of pages. A CDA document can contain several different page layout styles. For example, the number of columns on a page and the placement of page numbers can vary from page to page.

The layout of a document is specified using the following **layout parameters:**

- **Page layout**

  The page layout describes a page. It includes such information as the page size, left and right margins, and so on.

- **Page description**

  The page description describes a page either as a single page layout or as set of page layouts. If the page description contains a set of page layouts, it also specifies the conditions under which the different page layouts are used. For example, a page description might consist of two page layouts: one for left-hand pages, and one for right-hand pages.

- **Galleys**

  Galleys in a document control the flow of text along a series of parallel paths. For example, in a magazine, articles are often spread over several pages. CDA stores the content of an article in a single segment. When th formatter creates the final form of the document, it inserts the content of the segment in a galley. The attributes of a galley specify its dimensions, the content of the galley, and so on.

CDA stores document content in logical order, that is, the order in which a reader normally reads it. The content of a magazine, for example, is stored oı article at a time, whereas in the final form, several articles are interspersed on a page. The formatter performs the change from this logical order to final form.

4 Office Document Architecture

This chapter provides an overview of the Office Document Architecture (ODA). It describes the following:

- Types of ODA documents
- ODA document structure
- How formatters format ODA documents
- ODA document hierarchy and informational structures
- ODA document application profiles (DAPs)

The purpose of ODA is to allow the interchange of compound documents either for presenting as specified by the creator, or for processing such as editing and reformatting.

ODA stores documents in the Office Document Interchange Format (ODIF), which is defined in ISO 8613[1], the same international standard that defines ODA.

4.1 Document Architecture Classes

The ODA standard defines three document architecture classes, which determine how documents can be processed. The document architecture classes are as follows:

- **Formatted**

  Formatted (final form) documents can only be presented (printed, displayed online, and so on). You cannot edit or reformat them.

[1] The standard that defines ODA and ODIF is *ISO 8613, Information Processing—Text and Office Systems—Office Document Architecture (ODA) and Interchange Format.*

In a formatted document, all the formatting decisions (such as hyphenation and page breaks) have been resolved. All the text elements, such as page and section numbers, have been inserted. Also, any externally-referenced document content has been included. A formatted document does not make any distinction between content that a user enters and content that was generated by a formatter.

- **Processable**

  Processable (revisable) documents can be processed (revised or edited). To present processable documents, you must first format them.

  Processable documents contain document content, and they also specify the document structure (for example, the relationships in a chapter, such as *a paragraph is part of a section*, and *a section is part of a chapter*).

  Processable documents do not contain page numbers or section numbers. Instead, they specify parameters that control the creation of these elements in the formatted document. Processable documents do not specify the exact layout of the document content. Instead, they contain a template for the page layout and the parameters that control the way the content is arranged in the formatted document.

- **Formatted processable**

  Formatted processable documents can either be presented or processed. You can reduce them to formatted form for presenting, or you can reduce them to processable form for further processing.

4.2 Document Structure

Like CDA documents, ODA documents have a generic logical and a generic layout structure, and a specific logical and a specific layout structure. The generic structures specify logical and layout structures to which a group of documents conforms. Each document in a group also has a logical structure and a layout structure that apply only to that document. These structures are the specific logical and layout structures of the document.

See Section 3.6.1 and Section 3.6.2 for more information on logical and layout structures.

4.2.1 Object Classes

Every item in the generic logical or layout structure of an ODA document is called an **object class.** The object class specifies **attributes** that contain information about the object class. In the specific logical or layout structure, each item is called an **object.** Objects are specific occurrences of object classes. When an application processes a document, every object inherits the attributes of the object class to which it belongs. ODA document objects are similar to CDA document segments. Object attributes are similar to CDA segment attributes.

For example, the generic layout structure can contain an object class that describes paragraphs. One of the paragraph attributes might specify the text font for paragraphs. Paragraph objects in the specific layout structure use the text font specified for the object class because they inherit that attribute from the object class that defines the paragraph.

Objects or object classes that contain other objects or object classes are called **composite.** Objects or object classes contained in composite objects or object classes are called **subordinate.** Subordinate objects or object classes in ODA documents are similar to nested segments in CDA documents. Objects or object classes that do not have any subordinates are called **basic.**

Every nonbasic object or object class has an attribute called *generator for subordinates.* The *generator for subordinates* attribute is part of the generic structure of the document. It describes the structure of the object or object class. The *generator for subordinates* attribute has one of the following values:

- optional (OPT)

  The object or object class can contain one occurrence of the subordinate object or the subordinate object class.

- repetitive (REP)

  The object or object class must contain one or more occurrences of the subordinate object or the subordinate object class.

- required (REQ)

  The object or object class must contain one occurrence of the subordinate object or the subordinate object class.

- optional and repetitive (OPT REP)

  The object or object class can contain one or more occurrences of the subordinate object or the subordinate object class.

The *generator for subordinates* attribute also has one of the following values:

- aggregate (AGG)

  The subordinate objects or object classes can occur in any order.

- choice (CHO)

  Only one from a set of subordinate objects or object classes can occur.

- sequence (SEQ)

  The subordinate objects or object classes must occur in a specified order.

Figure 4–1 shows the logical structure of a chapter from a manual. The chapter consists of a title and a body. The body contains sections, which in turn contain figures, paragraphs, and lists.

Figure 4–1 Logical Structure

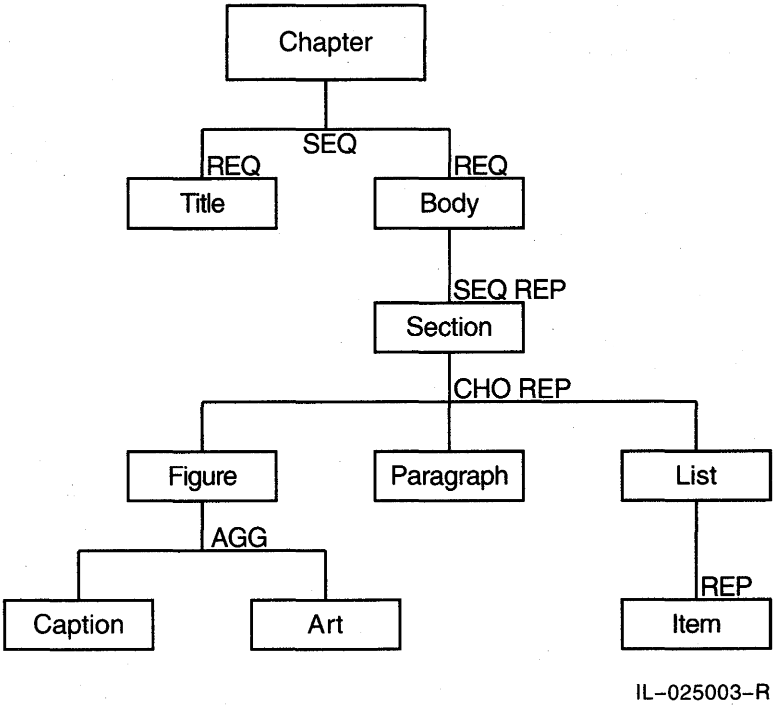
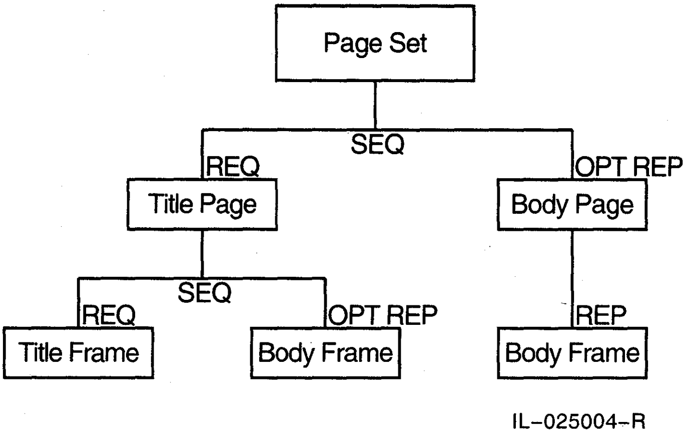


Figure 4–2 shows the layout structure of the same chapter. The chapter is a page set, which contains pages. The page set has one title page followed by a number of body pages. Each page in the page set contains blocks, which contain document content.

Figure 4–2 Layout Structure

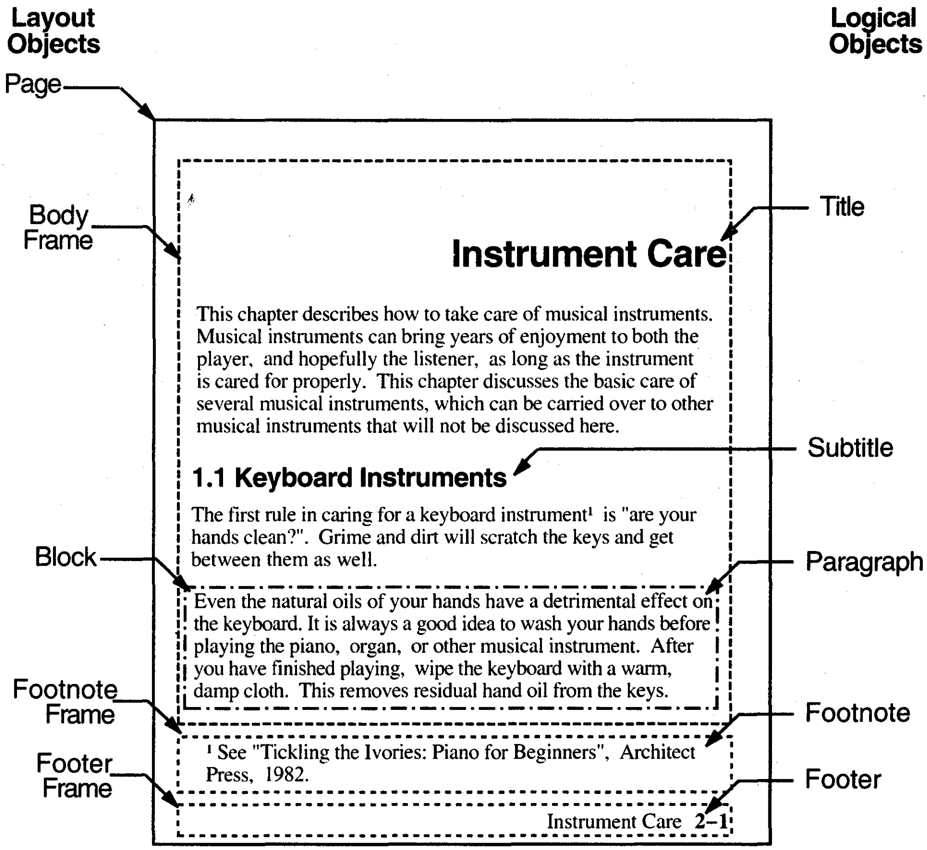


4.3 Document Formatting

Formatting merges the logical structure with the layout structure; that is, when a formatter formats a document, it fits the content of the logical objects into the layout objects. The logical structure and the layout structure depend on each other because the text in the logical objects is the same as the text in the layout objects.

Figure 4–3 shows a page that illustrates how the layout structure is related to the logical structure.

Figure 4–3 Formatted Page



Instrument Care

This chapter describes how to take care of musical instruments. Musical instruments can bring years of enjoyment to both the player, and hopefully the listener, as long as the instrument is cared for properly. This chapter discusses the basic care of several musical instruments, which can be carried over to other musical instruments that will not be discussed here.

1.1 Keyboard Instruments

The first rule in caring for a keyboard instrument[1] is "are your hands clean?". Grime and dirt will scratch the keys and get between them as well.

Even the natural oils of your hands have a detrimental effect on the keyboard. It is always a good idea to wash your hands before playing the piano, organ, or other musical instrument. After you have finished playing, wipe the keyboard with a warm, damp cloth. This removes residual hand oil from the keys.

[1] See "Tickling the Ivories: Piano for Beginners", Architect Press, 1982.

Instrument Care 2–1

4.3.1 Positioning Blocks in Frames

The size of a block is determined by the content of the block. For example, a paragraph block can have a predetermined width, but its length depends on the number of lines in the block. The following attributes control page layout:

- *Layout path*

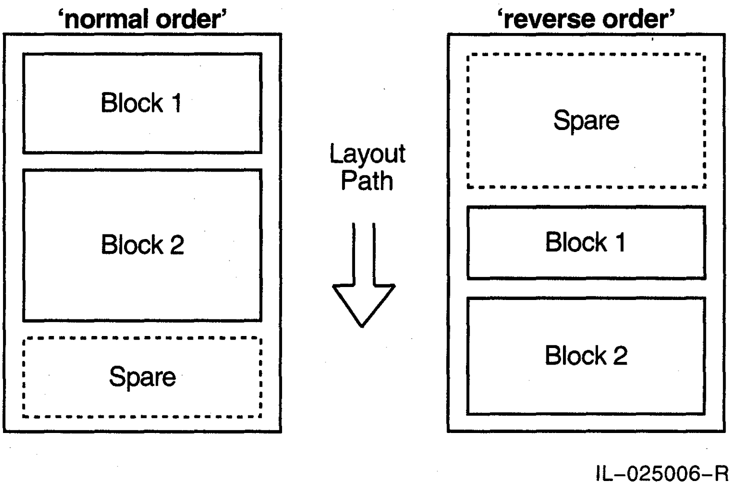
  Specifies the direction in which the formatter places the blocks in a frame. If the layout path is downwards, the formatter places the first block at the top of the frame, the next block under the first, and so on. If the layout path is upwards, the formatter places the first block at the bottom of the frame, the next frame on top of the first, and so on.

- *Fill order*

  Specifies the order in which the formatter places the blocks in a frame. If the *fill order* attribute has the value normal order, the formatter places the blocks in the frame in the direction of the layout path. If the *fill order* attribute has the value reverse order, the formatter places the blocks in the frame in the opposite direction to the layout path.

  For example, if the layout path is downward, and the *fill order* attribute has the value normal order, the first block is at the top, with subsequent blocks underneath. If the *fill order* attribute has the value reverse order, the first block is at the bottom and subsequent blocks push the first block upwards to make room for themselves underneath. Figure 4–4 shows these two cases.

Figure 4–4 Normal and Reverse Fill Order



- *Offset*

  Specifies the distance between a block and the frame that contains it.

- *Separation*

  Specifies the distance between two adjacent blocks.

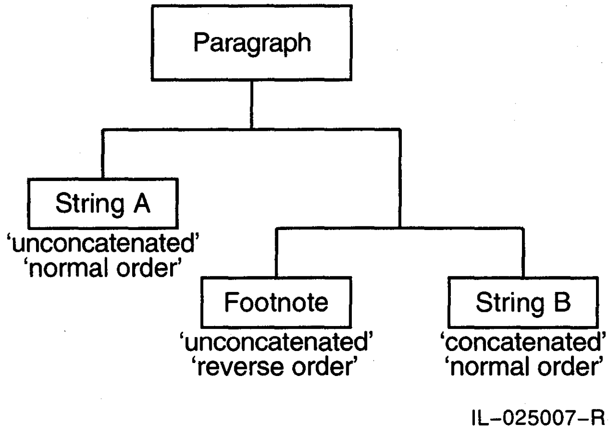
- *Concatenation*

  Specifies that an object shares a block with the previous objects that have the same structure, layout category, and fill order.

  For example, a paragraph contains footnote text interspersed with the text strings that make up the paragraph. The footnote text logically belongs with the paragraph, but in terms of layout, it belongs at the bottom of the page. All the text strings are laid out in a single block.

  The logical structure of the paragraph in Figure 4–5 has the *concatenation* and the *fill order* attributes shown.

Figure 4–5 Logical Structure for Paragraph

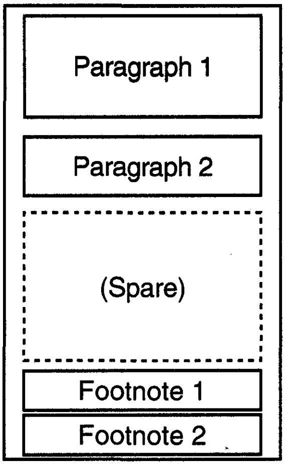


The *concatenation* attribute for string A, and for the footnote, has the value unconcatenated. The *concatenation* attribute for string B has the value concatenated. This means that string B is concatenated with string A, rather than with the intervening footnote.

The *fill order* attribute of both text strings has the value normal order. The *fill order* attribute of the footnote has the value reverse order.

If the page contains two paragraphs, one of which contains two footnotes, the final layout will be as in Figure 4–6.

Figure 4–6 Layout of Paragraphs and Footnotes



IL–025008–R

4.4 Document Profile

Every ODA document has a **document profile,** which contains attributes that provide information about the document as a whole. The attributes can be divided into three sets:

- Presence of document constituents

  The attributes in this set indicate what constituents are present in the document body (not all parts of the document have to be present). These attributes have the value present. For example, if the document contains a specific logical structure, the profile contains an attribute called *specific logical structure,* which has the value present. The value not present does not exist — if the document does not have a specific logical structure, the attribute is not specified.

- Document characteristics

  The attributes in this set specify a range of information about the document as a whole. For example, these attributes specify the version of the ODA architecture to which the document conforms, the fonts used in the document, and so on.

  One of the attributes in this set is the *document application profile defaults* attribute. This attribute specifies default values for a number of other attributes. These defaults are used if values for the attributes are not specified elsewhere in the document.

- Document management attributes

  The attributes in this set specify items such as the document title and creation date.

The document profile for an ODA document is similar to the document descriptor and document header combined for a CDA document.

4.5 Document Application Profile

A **document application profile** (DAP) specifies a combination of ODA features that is supported by a particular application. ODA documents that conform to a DAP use only the ODA features specified in the DAP. This ensures that an application that supports the ODA features specified in the DAP will be able to process any document that conforms to the DAP.

The following sections describe the DAPs that are supported by the ODA CDA Gateway.

4.5.1 Q111

The Q111 DAP allows documents containing character content to be exchanged between basic word-processing systems.

The logical structure of the document is a sequence of paragraphs. This sequence can be divided into groups of one or more paragraphs. Different layout and presentation features can be specified for each group. Pages can be numbered automatically.

The layout structure of the document is a sequence of pages, each containing a single column of text. A part of the page can be reserved for header or footer content.

Layout and presentation parameters such as margins, emphasis, and so on can be applied to each paragraph in the document.

4.5.2 Q112

The logical structure of a Q112 document is a tree structure that can be used to represent chapters, sections, or nested subsections in a document. Pages can be numbered automatically. Footnotes can also be numbered automatically and referred to in the document.

The layout structure of a Q112 document is a sequence of pages, with one or more columns of text allowed on each page. The number of columns can vary on an individual page and from one page to the next. Footnotes are automatically laid out at the bottom of each page. Geometric graphics and raster images can be included in the document and can be numbered automatically. Layout and presentation parameters, such as indentation and emphasis, can be applied to the text of the document.

4.5.3 Q121

The Q121 DAP allows documents containing only characters to be exchanged in processable form. This means that documents can be exchanged and presented despite differences in the processing and presentation capabilities of the sender and receiver. The receiver has to format the document. The logical structure of the document is a sequence of paragraphs. The document does not have a layout structure.

A
ODA CDA Gateway Messages

This appendix lists the messages issued by the ODA CDA Gateway. There are two sections. Section A.1 lists the messages that are issued by the ODA front end. Section A.2 lists the messages that are issued by the ODA back end.

In each section, the messages are listed in alphabetical order, by message name.

A.1 ODA Front End Messages

ALLOCFAIL, Memory allocation failure

Level: Fatal.

Explanation: The standard memory allocation procedure failed to allocate or deallocate dynamic memory.

User Action: None.

BADCSISEQ, Invalid control sequence in character content

Level: Fatal.

Explanation: The ODA front end encountered an invalid CSI (control sequence introducer) sequence in the ODA document.

User Action: Check the ODA document to ensure that all CSI sequences are valid.

BADESCSEQ, Invalid escape sequence in character content

Level: Fatal.

Explanation: The ODA front end encountered an invalid escape sequence in the character content of the ODA document.

User Action: Check the ODA document to ensure that all escape sequences are valid.

BADDOCARCHCL, Invalid document architecture class found

Level: Fatal.

Explanation: The document that the ODA CDA Gateway was attempting to convert to CDA did not specify a valid document architecture class. Valid document architecture classes are formatted, processable, and formatted processable.

User Action: None.

BADFONT, Bad font in profile—defaulting to 12 point Courier

Level: Warning.

Explanation: The ODA CDA Gateway found a font definition in the ODA document profile that the ODA CDA Gateway does not support. Text that uses this font in the ODA document uses the 12 point Courier font in the CDA document.

User Action: None.

BADIFCLASS, Invalid interchange format class found

Level: Fatal.

Explanation: The interchange format class specified by the ODA document is invalid. This means that the ODA document does not conform properly to ODA and the ODA front end cannot convert the document to CDA. See the international standard that defines the ODA architecture, ISO 8613, for information about interchange formats.

User Action: None.

CDAFAIL, Unexpected error from a CDA toolkit routine

Level: Fatal.

Explanation: An unexpected status occurred in a CDA Toolkit routine used by the ODA front end.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

ERRREADDOC, Error occurred while reading ODA input file

Level: Fatal.

Explanation: The ODA CDA Gateway could not read the ODA document.

User Action: Check whether the file exists. If it exists, check the protection on the file, and check that the document is a valid ODA document. If you still receive this message, and you are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

FAILDECOMP, Failed to decompress image—discarding and continuing

Level: Warning.

Explanation: The ODA document contains a raster image that the ODA front end could not decompress. The ODA front end discards the image and continues with the conversion.

User Action: Check the raster image in the ODA document to ensure that it uses a compression scheme that is supported by the ODA CDA Gateway. If you still receive this message, and you are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

FORMDOC, Document architecture class is 'formatted'—conversion unsupported

Level: Fatal.

Explanation: The ODA document specified on the command line is a formatted document. The ODA front end does not support the conversion of formatted documents.

User Action: Ensure that the ODA document to be converted is a processable document.

FORMPROCDOC, Document architecture class is 'formatted processable' – conversion unsupported

Level: Fatal.

Explanation: The ODA document specified on the command line is a formatted processable document. The ODA front end does not support the conversion of formatted processable documents.

User Action: Ensure that the ODA document to be converted is a processable document.

IFBDOC, ODA document interchange format is interchange format b – conversion unsupported

Level: Fatal.

Explanation: The ODA document specified on the command line specifies the interchange format "b". The ODA front end does not support the conversion of interchange format "b" documents. See the international standard that defines ODA, ISO 8613, for information about interchange format "b" documents.

User Action: Ensure that the ODA document to be converted is an interchange format "a" document.

IGNORCCHAR, Control character with no DDIF equivalent found—discarding

Level: Warning.

Explanation: The ODA document contains a control character that does not have an equivalent character in CDA. The ODA front end discards the character and continues with the conversion.

User Action: None.

IGNORESCS, Ignoring SCS sequence

Level: Warning.

Explanation: The ODA document specifies character spacing attributes. The ODA CDA Gateway does not support these attributes so the ODA front end ignores the attributes when it converts the ODA document to CDA.

User Action: None.

IGNORESHS, Ignoring SHS sequence

Level: Warning.

Explanation: The ODA document specifies character spacing attributes. The ODA CDA Gateway does not support these attributes so the ODA front end ignores the attributes when it converts the ODA document to CDA.

User Action: None.

INTERR, Internal error—submit an SPR

Level: Fatal.

Explanation: An internal error occurred in the ODA CDA Gateway.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

INVCSIPAR, Invalid CSI parameter found—ignoring

Level: Warning.

Explanation: The ODA front end encountered a CSI (control sequence introducer) sequence that has an invalid parameter. The ODA front end ignores the CSI sequence and continues with the conversion.

User Action: Check the ODA document to ensure that all CSI sequences and parameters are valid.

INVCHAR, Invalid character occurred in data stream

Level: Error.

Explanation: The ODA front end encountered an invalid character in the text content of the ODA document.

User Action: Check the ODA document to ensure that the text content of the document contains only valid characters.

INVQ111, Document does not conform to Q111 DAP

Level: Fatal.

Explanation: The ODA document profile identifies the document as conforming to the Q111 DAP, but the document does not conform to this DAP.

User Action: Ensure that the ODA document conforms to the Q111 DAP.

INVQ112, Document does not conform to Q112 DAP

Level: Fatal.

Explanation: The ODA document profile identifies the document as a document that conforms to the Q112 DAP, but it does not conform to this DAP.

User Action: Ensure that the ODA document conforms to the Q112 DAP.

INVQ121, Document does not conform to Q121 DAP

Level: Fatal.

Explanation: The ODA document profile identifies the document as a document that conforms to the Q121 DAP, but it does not conform to this DAP.

User Action: Ensure that the ODA document conforms to the Q121 DAP.

NOCHARSPAC, Unable to convert character spacing—ignoring

Level: Warning.

Explanation: The ODA document specifies character spacing attributes. The ODA CDA Gateway does not support these attributes so the ODA front end ignores the attributes when it converts the ODA document to CDA.

User Action: None.

NOREQATTR, Required attribute missing in ODA input document

Level: Warning.

Explanation: A document attribute that is required to conform to the specified DAP is missing from the ODA document.

User Action: Check the ODA document to ensure that it conforms to the DAP specified in the document profile.

ODIFFAIL, Unexpected error from an ODA DAC routine

Level: Fatal.

Explanation: An unexpected status occurred in an ODA Document Access Component (DAC) routine used by the ODA front end.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

RASNOTRFP, Formatted raster object found—discarding

Level: Warning.

Explanation: The ODA front end encountered a formatted raster image in the ODA document. The ODA CDA Gateway does not support this image type so the ODA front end discards the formatted raster image when it converts the ODA document to CDA.

User Action: Ensure that the ODA document contains only formatted processable raster images.

STNOTFND, String terminator not found in SOS—ST string

Level: Fatal.

Explanation: The ODA document contained a string of formatting instructions that was not terminated with a string terminator (ST) escape sequence. This means that there is an error in the ODA document.

User Action: None.

- ISO Latin Arabic
- ISO Latin Greek
- ISO Latin Hebrew

The ODA back end converts text in all of these character sets to ODA without any data loss. However, CDA documents can also contain text in Digital-specific character sets. The ODA back end cannot convert this text to ODA, and the text is lost when you convert your document from CDA to ODA.

- Typeface information

  The following typefaces are supported by the ODA CDA Gateway: Courier, Helvetica, and Times. If the CDA document that you convert to ODA is to conform to the Q112 DAP, and the text content is only in these typefaces, the ODA back end converts all the typeface information, including rendition changes (for example, from roman to italic), to ODA without any data loss.

  If the CDA document contains text in other typefaces, the ODA back end does not convert the typeface information to ODA. Instead, the ODA back end issues an error message and inserts the text content in the ODA document, but in the Courier typeface. However, the rendition changes in the CDA document are retained in the ODA document.

  If the CDA document that you convert to ODA is to conform to the Q111 or Q121 DAPs, any typeface information in the CDA document is not converted to ODA. However, the rendition changes in the CDA document are retained in the ODA document if the DAP allows such changes.

INVCSIPAR, Invalid CSI parameter found—ignoring

Level: Warning.

Explanation: The ODA front end encountered a CSI (control sequence introducer) sequence that has an invalid parameter. The ODA front end ignores the CSI sequence and continues with the conversion.

User Action: Check the ODA document to ensure that all CSI sequences and parameters are valid.

INVCHAR, Invalid character occurred in data stream

Level: Error.

Explanation: The ODA front end encountered an invalid character in the text content of the ODA document.

User Action: Check the ODA document to ensure that the text content of the document contains only valid characters.

INVQ111, Document does not conform to Q111 DAP

Level: Fatal.

Explanation: The ODA document profile identifies the document as conforming to the Q111 DAP, but the document does not conform to this DAP.

User Action: Ensure that the ODA document conforms to the Q111 DAP.

INVQ112, Document does not conform to Q112 DAP

Level: Fatal.

Explanation: The ODA document profile identifies the document as a document that conforms to the Q112 DAP, but it does not conform to this DAP.

User Action: Ensure that the ODA document conforms to the Q112 DAP.

INVQ121, Document does not conform to Q121 DAP

Level: Fatal.

Explanation: The ODA document profile identifies the document as a document that conforms to the Q121 DAP, but it does not conform to this DAP.

User Action: Ensure that the ODA document conforms to the Q121 DAP.

NOCHARSPAC, Unable to convert character spacing—ignoring

Level: Warning.

Explanation: The ODA document specifies character spacing attributes. The ODA CDA Gateway does not support these attributes so the ODA front end ignores the attributes when it converts the ODA document to CDA.

User Action: None.

NOREQATTR, Required attribute missing in ODA input document

Level: Warning.

Explanation: A document attribute that is required to conform to the specified DAP is missing from the ODA document.

User Action: Check the ODA document to ensure that it conforms to the DAP specified in the document profile.

ODIFFAIL, Unexpected error from an ODA DAC routine

Level: Fatal.

Explanation: An unexpected status occurred in an ODA Document Access Component (DAC) routine used by the ODA front end.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

RASNOTRFP, Formatted raster object found—discarding

Level: Warning.

Explanation: The ODA front end encountered a formatted raster image in the ODA document. The ODA CDA Gateway does not support this image type so the ODA front end discards the formatted raster image when it converts the ODA document to CDA.

User Action: Ensure that the ODA document contains only formatted processable raster images.

STNOTFND, String terminator not found in SOS—ST string

Level: Fatal.

Explanation: The ODA document contained a string of formatting instructions that was not terminated with a string terminator (ST) escape sequence. This means that there is an error in the ODA document.

User Action: None.

UNCT4T6RASTER, Uncompressed mode T.4 or T.6 image cannot be converted— discarding

Level: Warning.

Explanation: The ODA document contains a Group 3 (CCITT Recommendation T.4) or Group 4 (CCITT Recommendation T.6) compressed raster image that contains uncompressed data. The ODA CDA Gateway does not support the conversion of such images, so the ODA front end discards the image when it converts the ODA document to CDA.

User Action: Ensure that Group 3 or Group 4 compressed raster images in the ODA document do not contain uncompressed chunks of data.

UNEXISLERR, Unexpected error from an ISL routine

Level: Fatal.

Explanation: An error occurred in an Image Services Library (ISL) routine that the ODA front end was using to convert an ODA raster image to a raster image in the CDA document.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

UNRECCOMP, Image uses unrecognized compression scheme—discarding image

Level: Warning.

Explanation: The ODA document contained a raster image that specified an unrecognized image compression scheme. The ODA front end discards the image and does not convert it to CDA.

User Action: Correct the image compression scheme specification in the ODA document so that it specifies a valid image compression scheme.

UNRECESCSEQ, Unrecognized escape sequence found

Level: Warning.

Explanation: The ODA front end encountered an unrecognized escape sequence in the ODA document. The ODA front end ignores the escape sequence and continues converting the ODA document.

User Action: None.

UNSCHARSET, Character set cannot be converted—defaulting

Level: Warning.

Explanation: The ODA document being converted to CDA contains text in a character set that is not supported by the ODA CDA Gateway. The ODA front end stores each character from the unsupported character set as the ISO Latin1 character that has the same character code.

User Action: None.

UNSUPDAP, Unsupported DAP

Level: Fatal.

Explanation: The ODA document conforms to a DAP that is not supported by the ODA CDA Gateway. See Section 4.5 for information about the DAPs supported by the ODA CDA Gateway.

User Action: None.

A.2 ODA Back End Messages

ALLOCFAIL, Memory allocation failure

Level: Fatal.

Explanation: The standard memory allocation procedure failed to allocate or deallocate dynamic memory.

User Action: None.

BADMAJVER, DDIF file major version incompatible with this converter

Level: Fatal.

Explanation: The application that created the CDA document used a different major version of the DDIF format than the ODA CDA Gateway. The ODA back end cannot create the ODA document.

User Action: Ensure that applications that create CDA documents use the same major version of the DDIF format as the ODA CDA Gateway.

BADMINVER, DDIF file minor version incompatible with this converter

Level: Warning.

Explanation: The application that created the CDA document used a different minor version of the DDIF format than the ODA CDA Gateway. The ODA back end creates the ODA document, but it is possible that some of the information in the ODA document will be lost during the conversion.

User Action: Ensure that applications that create CDA documents use the same minor version of the DDIF format as the ODA CDA Gateway.

CANTCONVCHAR, Discarding character that has no ISO 6937-2 equivalent

Level: Warning.

Explanation: The CDA document contains an ISO Latin1 character for which there is no equivalent character in the ISO 6937/2 character set. The ODA back end discards the character and continues with the conversion.

User Action: Remove the ISO Latin1 character from the CDA document, or use the CHARACTERSET processing option to create an ODA document in which the character content is in the ISO Latin1 character set.

CDAFAIL, Unexpected error from a CDA Toolkit routine

Level: Fatal.

Explanation: An error occurred in a CDA Toolkit routine used by the ODA back end.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

ERRCONODA, DDIF document not convertible to DAP-conformant ODA

Level: Fatal.

Explanation: The ODA back end cannot convert a DDIF format document to ODA if the document contains only one flow, and there is no content in the flow. The DAPs supported by the ODA CDA Gateway do not support this feature of CDA.

User Action: None.

ERRWRITEDOC, Error occurred while writing to ODA output file

Level: Fatal.

Explanation: When the ODA back end converts a CDA document to ODA, it stores the ODA document in memory, and writes the ODA document to a file when the conversion has finished. As the ODA back end writes the ODA document to the file, it verifies that the document is a valid ODA document and issues this message if it finds errors in the document.

User Action: Check whether the file exists. If it exists, check the protection on the file, and check that the document is a valid ODA document. If you still receive this message, and you are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

INTERR, Internal error—submit an SPR

Level: Fatal.

Explanation: An internal error occurred in the ODA CDA Gateway.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

INVSEGTAG, Invalid segment tag for this level of the document

Level: Error.

Explanation: The ODA back end encountered a segment tag that was not in its correct place in the CDA document structure. This means that the structure of the CDA document is invalid.

User Action: Correct the structure of the CDA document.

KILLEDCRF, DDIF content reference could not be converted—discarding

Level: Informational.

Explanation: The CDA document contains a content reference that the ODA back end cannot convert to the ODA format, so the ODA back end discards the content reference and continues with the conversion.

User Action: None.

KILLEDINDEX, DDIF index could not be converted—discarding

Level: Informational.

Explanation: The CDA document contains an index. The DAP to which the ODA document conforms does not support indexes, so the ODA back end discards the index and continues with the conversion.

User Action: None.

KILLEDTOC, DDIF table of contents could not be converted—discarding

Level: Informational.

Explanation: The CDA document contains a table of contents. The DAP to which the ODA document conforms does not support tables of contents, so the ODA back end discards the table of contents and continues with the conversion.

User Action: None.

NOGEOMETRIC, Geometric graphic content (CGM) has been discarded

Level: Warning.

Explanation: The ODA back end tried to convert a CDA document containing geometric graphics to an ODA document that conforms to a DAP that does not support the inclusion of geometric graphics. The ODA back end discards the geometric graphic content and continues with the conversion.

User Action: Specify a DAP that supports the inclusion of geometric graphics.

NORASTER, Image content (raster graphic) has been discarded

Level: Warning.

Explanation: The ODA back end tried to convert a CDA document containing raster image graphics to an ODA document that conforms to a DAP that does not support the inclusion of raster images. The ODA back end discards the raster image content and continues with the conversion.

User Action: Specify a DAP that supports the inclusion of raster images.

ODIFFAIL, Unexpected error from an ODA DAC routine

Level: Fatal.

Explanation: An error occurred in an ODA Document Access Component (DAC) routine used by the ODA back end.

User Action: If you receive this message and are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

REVERTCONTENT, Unexpected content found—content being converted without structure

Level: Informational.

Explanation: The ODA back end encountered unrecognized structure information in a CDA document segment. The ODA back end ignores this information and converts the content of the segment to ODA without converting the structure.

User Action: None.

UNEXCONT, Unexpected Content Category found—discarding

Level: Warning.

Explanation: The ODA back end encountered a CDA document segment that contains unrecognized content, that is, content other than text, geometric graphics, or raster images. The ODA back end discards the segment and continues with the conversion.

User Action: None.

UNEXISLERR, Unexpected error from an ISL routine

Level: Fatal.

Explanation: An error occurred in an Image Services Library (ISL) routine that the ODA back end was using to convert a CDA raster image to a raster image in the ODA document.

User Action: Check that the CDA raster image file is valid. If the image file is valid, and you are under a service contract with Digital, call your customer service center. Otherwise, submit a Software Performance Report (SPR).

UNRECCOMP, Image uses unrecognized compression scheme—discarding image

Level: Warning.

Explanation: A CDA raster image was compressed using an image compression scheme that is not supported in ODA. The ODA back end discards the image and continues with the conversion.

ODA supports the CCITT Group 3 and Group 4 image compression schemes.

User Action: Ensure that compressed raster images in the CDA document use one of the compression schemes supported by ODA.

UNSCHARSET, Character set cannot be converted—defaulting

Level: Warning.

Explanation: The CDA document being converted to ODA contains text in a character set that is not supported in ODA. The ODA back end stores each character from the unsupported character set as the ISO Latin1 character that has the same character code.

User Action: Remove the text in the unsupported character set from the CDA document before converting the document to ODA.

Glossary

The Glossary defines some important terms used in this guide.

attribute

Specifies information about a CDA segment or an ODA object or object class. For example, a CDA text segment may have an attribute that specifies the typeface of the segment.

basic object

An ODA object that does not contain any subordinate objects.

basic object class

An ODA object class that does not contain any subordinate object classes.

CCITT

Consultative Committee for International Telephony and Telegraphy.

CCITT Group 3 compression scheme

An international standard compression scheme for raster images. The Group 3 compression scheme is defined by CCITT Recommendation T.4.

CCITT Group 4 compression scheme

An international standard compression scheme for raster images. The Group 4 compression scheme is defined by CCITT Recommendation T.6.

CDA

Compound Document Architecture.

compound document

A document that contains text, geometric graphics, raster images, data from other applications, and other forms of data such as sound.

composite object

An ODA object that contains subordinate objects.

composite object class

An ODA object class that contains subordinate object classes.

DAP

Document application profile. Specifies the set of ODA features supported by a particular application. An application that conforms to a DAP can process any ODA document created by any other application that conforms to the DAP.

DDIF

Digital Document Interchange Format.

directive

A formatting instruction that affects the final appearance of a CDA document.

document application profile

See DAP.

document architecture class

Specifies a category of ODA documents. The category to which a document belongs determines the way in which it can be processed. There are three document architecture classes: formatted, processable, and formatted processable.

document descriptor

Specifies information that describes the characteristics of a CDA document such as the name of the product that created the document and the CDA version used by the document.

document header

Specifies information about a CDA document such as the document title, author, file name, and creation date.

document processing model

A collective name for the stages in the processing of an ODA document. The stages are the editing process, the layout process, and the imaging process.

document profile

Specifies information that describes the characteristics of an ODA document such as the title and the name of the author, and the DAP to which the document conforms.

document root

Identifies a CDA document. The document root is at the top of the inverted tree structure of the CDA document.

DTIF

Digital Tabular Interchange Format.

editing process

The creation and revision of a document. There are two stages: content editing (creating or modifying the document content), and logical structure editing (creating or modifying specific logical objects).

final form document

A document that can be printed or displayed but not edited.

formatted document

The ODA term for a final form document.

formatted processable document

An ODA document that is made up of the formatted form and the processable form of a document.

galley

A series of connected blocks of text in a CDA document. The blocks do not have to be adjacent to one another to be in the same galley. For example, in a magazine, articles are often spread over several pages. The text of the article forms a single galley, but the text blocks that make up the galley appear on different pages.

generic layout structure

The information that defines the physical structure of a class of CDA or ODA documents, for example, pages can be divided into frames and frames divided into blocks of text or graphics. All documents that adhere to a particular set of these rules have the same generic layout structure.

generic logical structure

The information that defines the logical structure of a class of CDA or ODA documents, for example, chapters can be divided into sections, sections can be divided into subsections, and so on. All documents that adhere to a particulaı set of these rules have the same generic logical structure.

hard content

The content that the user enters directly in the document.

hard directive

A directive that the user enters directly in the document.

ISO

International Standards Organization.

layout parameter

A parameter that determines the appearance of pages in a CDA document.

layout style

The collective term for the attributes that control the layout in an ODA document.

local attribute

An attribute that applies only to a single occurrence of a segment.

named segment

A segment that forms part of the generic logical structure of the CDA document. Named segments and their attributes are defined on type definitions, which are attached to the root segment.

nested segment

A segment that is contained in another segment.

object

An item in the specific structure of an ODA document. The object belongs to an object class, and it inherits the attributes of the object class when the application processes the document.

object class

An item in the generic structure of an ODA document. The object class definе a set of attributes that applies to every occurrence of the object class.

ODA

Office Document Architecture.

ODIF

Office Document Interchange Format.

options file

A file containing processing options. To specify processing options, you create an options file and specify that file when you invoke the ODA CDA Gateway.

page description

Describes a page either as a single page layout or as a set of page layouts. If the page description contains a set of page layouts, the page description also specifies the conditions under which each page layout is used.

page layout

Describes a page in a CDA document. The page layout includes information on the page size, left and right margins, and so on.

page set

The set of page types used in a document. Each page in the page set specifies the circumstances under which that page is selected, and the page layout used to display that page.

presentation style

The collective term for all ODA attributes that control how the ODA document content is presented. The presentation style determines the appearance of individual blocks.

processable document

The ODA term for a revisable document.

processing option

When you invoke the ODA CDA Gateway, processing options allow you to control aspects of the interchange. For example, when converting a CDA document to ODA, you can specify the DAP to which the ODA document must conform.

revisable document

A document that can be revised (edited) but cannot be printed or displayed without being formatted.

root segment

The root segment of a CDA document. All segments that contain document content are nested under the root segment.

segment

An item of CDA document content to which a specified set of attributes applies. A segment is part of the specific logical structure of a CDA document.

soft content

The content that is created by an application (for example, page numbers).

soft directive

A directive that is created by an application (for example, page breaks inserted as a result of repagination).

specific layout structure

A layout structure that is specific to a particular document.

specific logical structure

A logical structure that is specific to a particular document.

subordinate object

An ODA object that is nested below some other object in the logical or layout structure.

subordinate object class

An ODA object class that is nested below some other object class in the logical or layout structure.

type definition

A set of definitions that is attached to the root segment of the CDA document. The type definition defines the named segments that are used in the document. It also defines the attributes of the named segments.

Index

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-DEC-DEMO (800-332-3366) using a 1200- or 2400-baud modem. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	————	Local Digital subsidiary or approved distributor
Internal[1]	————	USASSB Order Processing - WMO/E15 *or* U.S. Area Software Supply Business Digital Equipment Corporation Westminster, Massachusetts 01473

[1] For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

Reader's Comments

DEC ODA Compound Document Architecture (CDA) Gateway User's Guide
AA–PB6KA–TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	□	□	□	□
Completeness (enough information)	□	□	□	□
Clarity (easy to understand)	□	□	□	□
Organization (structure of subject matter)	□	□	□	□
Figures (useful)	□	□	□	□
Examples (useful)	□	□	□	□
Index (ability to find topic)	□	□	□	□
Page layout (easy to find information)	□	□	□	□

I would like to see more/less ______________________________

____

What I like best about this manual is ______________________________

____

What I like least about this manual is ______________________________

____

I found the following errors in this manual:
Page Description
_____ ______________________________
_____ ______________________________
_____ ______________________________

Additional comments or suggestions to improve this manual:

____

____

____

I am using **Version** _____ of the software this manual describes.
Name/Title ______________________ Dept. __________
Company ______________________ Date __________
Mailing Address ______________________________
______________________ Phone __________

— Do Not Tear - Fold Here and Tape

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications — Spit Brook
ZKO1-3/J35
110 SPIT BROOK ROAD
NASHUA, NH 03062-9987

— Do Not Tear - Fold Here

Cut Along Dotted Line