[MS-DCLB]:

Desktop Clipboard Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs**. To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
8/14/2009	0.1	Major	First Release.
9/25/2009	0.1.1	Editorial	Changed language and formatting in the technical content.
11/6/2009	0.1.2	Editorial	Changed language and formatting in the technical content.
12/18/2009	0.1.3	Editorial	Changed language and formatting in the technical content.
1/29/2010	0.2	Minor	Clarified the meaning of the technical content.
3/12/2010	0.2.1	Editorial	Changed language and formatting in the technical content.
4/23/2010	0.2.2	Editorial	Changed language and formatting in the technical content.
6/4/2010	0.3	Minor	Clarified the meaning of the technical content.
7/16/2010	0.3	None	No changes to the meaning, language, or formatting of the technical content.
8/27/2010	0.3	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2010	0.3	None	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	0.3	None	No changes to the meaning, language, or formatting of the technical content.
1/7/2011	0.3	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2011	0.3	None	No changes to the meaning, language, or formatting of the technical content.
3/25/2011	0.3	None	No changes to the meaning, language, or formatting of the technical content.
5/6/2011	0.3	None	No changes to the meaning, language, or formatting of the technical content.
6/17/2011	0.4	Minor	Clarified the meaning of the technical content.
9/23/2011	0.4	None	No changes to the meaning, language, or formatting of the technical content.
12/16/2011	0.4	None	No changes to the meaning, language, or formatting of the technical content.
3/30/2012	0.4	None	No changes to the meaning, language, or formatting of the technical content.
7/12/2012	0.4	None	No changes to the meaning, language, or formatting of the technical content.
10/25/2012	0.4	None	No changes to the meaning, language, or formatting of the technical content.
1/31/2013	0.4	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
8/8/2013	0.4	None	No changes to the meaning, language, or formatting of the technical content.
11/14/2013	0.4	None	No changes to the meaning, language, or formatting of the technical content.
2/13/2014	0.4	None	No changes to the meaning, language, or formatting of the technical content.
5/15/2014	0.4	None	No changes to the meaning, language, or formatting of the technical content.
6/30/2015	0.4	None	No changes to the meaning, language, or formatting of the technical content.
10/16/2015	1.0	Major	Significantly changed the technical content.
7/14/2016	1.0	None	No changes to the meaning, language, or formatting of the technical content.
6/1/2017	1.0	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction 6
1.1 Glossary 6
1.2 References 8
1.2.1 Normative References 8
1.2.2 Informative References 8
1.3 Overview 9
1.4 Relationship to Other Protocols 9
1.5 Prerequisites/Preconditions 9
1.6 Applicability Statement 9
1.7 Versioning and Capability Negotiation 9
1.8 Vendor-Extensible Fields 9
1.9 Standards Assignments 9

2 Messages 10
2.1 Transport 10
2.2 Message Syntax 10
2.2.1 Common Field Values 10
2.2.1.1 ClipboardFormatName 10
2.2.1.2 DDETopicType 11
2.2.1.3 ExecuteCommandType 11
2.2.1.4 Notifications 11
2.2.1.5 PaletteEntryFlags 12
2.2.1.6 SharingStatusType 12
2.2.2 Control Information 12
2.2.2.1 EXECCOMMAND 12
2.2.2.2 SHARE_LIST_ENTRYA 13
2.2.2.3 SHARE_LISTA 13
2.2.2.4 SHARE_LIST_ENTRYW 13
2.2.2.5 SHARE_LISTW 14
2.2.2.6 CLIPFORMAT_LIST_ENTRYA 14
2.2.2.7 CLIPFORMAT_LISTA 14
2.2.2.8 CLIPFORMAT_LIST_ENTRYW 14
2.2.2.9 CLIPFORMAT_LISTW 15
2.2.3 Clipbook Data 15
2.2.3.1 CLIPDATA_METAFILEPICT 15
2.2.3.2 CLIPDATA_ENHMETAFILE 15
2.2.3.3 CLIPDATA_BITMAP 16
2.2.3.4 CLIPDATA_PALETTE_ENTRY 16
2.2.3.5 CLIPDATA_PALETTE 16
2.2.3.6 CLIPDATA_OTHERFORMATS 17

3 Protocol Details 18
3.1 Server Details 19
3.1.1 Abstract Data Model 19
3.1.2 Timers 19
3.1.3 Initialization 19
3.1.4 Higher-Layer Triggered Events 19
3.1.5 Message Processing Events and Sequencing Rules 19
3.1.5.1 Command Message Processing 20
3.1.5.1.1 CMD_INITSHARE 20
3.1.5.1.2 CMD_DELETE 20
3.1.5.1.3 CMD_SHARE 20
3.1.5.1.4 CMD_UNSHARE 20
3.1.5.1.5 CMD_PASTE 20
3.1.5.2 Responses to Data Requests 20

3.1.6 Timer Events 21
3.1.7 Other Local Events 21
3.2 Client Details 21
3.2.1 Abstract Data Model 21
3.2.2 Timers 21
3.2.3 Initialization 21
3.2.4 Higher-Layer Triggered Events 22
3.2.5 Message Processing Events and Sequencing Rules 22
3.2.5.1 Command Messages 22
3.2.5.2 Data Requests 22
3.2.6 Timer Events 22
3.2.7 Other Local Events 22
4 Protocol Examples 23
5 Security 25
5.1 Security Considerations for Implementers 25
5.2 Index of Security Parameters 25
6 Appendix A: Product Behavior 26
7 Change Tracking 27
8 Index 28

1 Introduction

This is a specification of the Desktop Clipboard Protocol, which uses the **Network Dynamic Data Exchange (NetDDE)** Protocol to implement a distributed store for graphical user interface (GUI) objects for desktop cut-and-paste operations. It specifies the mechanism by which the Windows ClipBook Viewer application (the Windows **clipboard**) communicates information between remote users.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

American National Standards Institute (ANSI) character set: A character set defined by a code page approved by the American National Standards Institute (ANSI). The term "ANSI" as used to signify Windows code pages is a historical reference and a misnomer that persists in the Windows community. The source of this misnomer stems from the fact that the Windows code page 1252 was originally based on an ANSI draft, which became International Organization for Standardization (ISO) Standard 8859-1 [ISO/IEC-8859-1]. In Windows, the ANSI character set can be any of the following code pages: 1252, 1250, 1251, 1253, 1254, 1255, 1256, 1257, 1258, 874, 932, 936, 949, or 950. For example, "ANSI application" is usually a reference to a non-**Unicode** or code-page-based application. Therefore, "ANSI character set" is often misused to refer to one of the character sets defined by a Windows code page that can be used as an active system code page; for example, character sets defined by code page 1252 or character sets defined by code page 950. Windows is now based on **Unicode**, so the use of ANSI character sets is strongly discouraged unless they are used to interoperate with legacy applications or legacy data.

ASCII: The American Standard Code for Information Interchange (ASCII) is an 8-bit character-encoding scheme based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text. ASCII refers to a single 8-bit ASCII character or an array of 8-bit ASCII characters with the high bit of each character set to zero.

big-endian: Multiple-byte values that are byte-ordered with the most significant byte stored in the memory location with the lowest address.

bitmap: A collection of structures that contain a representation of a graphical image, a **logical palette**, dimensions and other information.

client: Synonym for client computer.

clipboard: A program provided by the operating system that enables local data transfer between applications by using the cut, copy, and paste operations.

clipboard format: An unsigned integer that uniquely identifies the format of a data packet that is stored in a binary large object (BLOB) and can be shared between processes through the operating system clipboard or other means.

clipbook: **Clipboard** data that is stored separately from the system **clipboard**.

color plane: One of the dimensions of a color space.

device-independent bitmap (DIB): A file format that was designed to help ensure that bitmap graphics that were created by using one application can be loaded and displayed in another application exactly as they appeared in the originating application.

Dynamic Data Exchange (DDE): An inter-process communication method that is featured in Windows. DDE allows two or more applications that are running simultaneously to exchange data and commands.

enhanced metafile format (EMF): A file format that supports the device-independent definitions of images.

intensity: The magnitude of a component color in the color space.

little-endian: Multiple-byte values that are byte-ordered with the least significant byte stored in the memory location with the lowest address.

logical palette: A **palette** that defines colors as device-independent values. Unlike the **system palette**, which has predefined, device-specific color definitions, a logical palette contains color values that can be defined entirely by an application. A logical palette entry is mapped to the **system palette** entry in order for the custom colors to appear when the application is run.

mapping mode: The way in which logical (device-independent) coordinates are mapped to device space (device-specific) coordinates. It also specifies the orientation of the axes and size of the units used for drawing operations.

metafile: A sequence of record structures that store an image in an application-independent format. Metafile records contain drawing commands, object definitions, and configuration settings. When a metafile is processed, the stored image can be rendered on a display, output to a printer or plotter, stored in memory, or saved to a file or stream.

METAFILEPICT: A structure that defines the **metafile** picture format. **METAFILEPICT** is used for exchanging **metafile** data through the **clipboard**. See [MSDN-METAFILEPICT] and [MSDN-CLIPFORM] for further information.

NetBIOS: A particular network transport that is part of the LAN Manager protocol suite. **NetBIOS** uses a broadcast communication style that was applicable to early segmented local area networks. A protocol family including name resolution, datagram, and connection services. For more information, see [RFC1001] and [RFC1002].

network dynamic data Exchange (NetDDE): A technology that allows applications using **dynamic data exchange (DDE)** to transparently share data over a network.

palette: An array of values, each element of which contains the definition of a color. The color elements in a palette are often indexed so that clients can refer to the colors, each of which can occupy 24 bits or more, by a number that requires less storage space.

server: A computer on which the remote procedure call (RPC) server is executing.

system palette: The **palette** that is actually in use to reproduce colors on a device such as a computer screen. A system palette has predefined, device-specific colors that are used by default, so that individual applications do not have to set them up.

Tag Image File Format (TIFF): A format for bitmapped image data that comes from scanners, frame grabbers, and photo-retouching applications. It supports the exchange of image data between applications, taking advantage of the varying capabilities of imaging devices. TIFF supports a number of compression schemes that allow the choice of the best space or time tradeoff for applications. For more information see [RFC3302] and [TIFF].

Unicode: A character encoding standard developed by the Unicode Consortium that represents almost all of the written languages of the world. The **Unicode** standard [UNICODE5.0.0/2007] provides three forms (UTF-8, UTF-16, and UTF-32) and seven schemes (UTF-8, UTF-16, UTF-16 BE, UTF-16 LE, UTF-32, UTF-32 LE, and UTF-32 BE).

Windows metafile format (WMF): A file format used by Windows that supports the definition of images, including a format for clip art in word-processing documents.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[ISO/IEC-8859-1] International Organization for Standardization, "Information Technology -- 8-Bit Single-Byte Coded Graphic Character Sets -- Part 1: Latin Alphabet No. 1", ISO/IEC 8859-1, 1998, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=28245

Note There is a charge to download the specification.

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[MS-EMF] Microsoft Corporation, "Enhanced Metafile Format".

[MS-WMF] Microsoft Corporation, "Windows Metafile Format".

[MSDN-DdeClientTransaction] Microsoft Corporation, "DdeClientTransaction function", http://msdn.microsoft.com/en-us/library/windows/desktop/ms648743(v=vs.85).aspx

[MSDN-DdeConnect] Microsoft Corporation, "DdeConnect function", http://msdn.microsoft.com/en-us/library/windows/desktop/ms648745(v=vs.85).aspx

[MSDN-DdeCreateDataHandle] Microsoft Corporation, "DdeCreateDataHandle function", http://msdn.microsoft.com/en-us/library/windows/desktop/ms648747(v=vs.85).aspx

[MSDN-NDdeShareAdd] Microsoft Corporation, "NDdeShareAdd function", http://msdn.microsoft.com/en-us/library/windows/desktop/aa365766(v=vs.85).aspx

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, https://www.rfc-editor.org/rfc/rfc2119.html

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.txt

1.2.2 Informative References

[MSDN-CLIPBOARD] Microsoft Corporation, "Clipboard", http://msdn.microsoft.com/en-us/library/ms648709(VS.85).aspx

[MSDN-GlobalLock] Microsoft Corporation, "GlobalLock function", http://msdn.microsoft.com/en-us/library/windows/desktop/aa366584(v=vs.85).aspx

[MSDN-METAFILEPICT] Microsoft Corporation, "METAFILEPICT structure", http://msdn.microsoft.com/en-us/library/ms649017(VS.85).aspx

[MSDN-META] Microsoft Corporation, "Metafiles", http://msdn.microsoft.com/en-us/library/dd145051(VS.85).aspx

[MSDN-NETDDE] Microsoft Corporation, "Network Dynamic Data Exchange", http://msdn.microsoft.com/en-us/library/aa365778(VS.85).aspx

1.3 Overview

The Desktop Clipboard Protocol is used by the Windows ClipBook Viewer application [MSDN-CLIPBOARD]. This protocol allows for transfer of **clipboard** data such as text and **bitmap** images between two participating machines.

In this protocol, the **server** role is defined as the actor that shares its clipboard data and provides this data when requested by the **client**. The client role is the actor that requests clipboard data from the server and is able to display this data to the user.

A sequence diagram showing this relationship is presented in section 3.

1.4 Relationship to Other Protocols

The Desktop Clipboard Protocol is implemented on top of the **NetDDE** protocol, as shown in the following diagram.

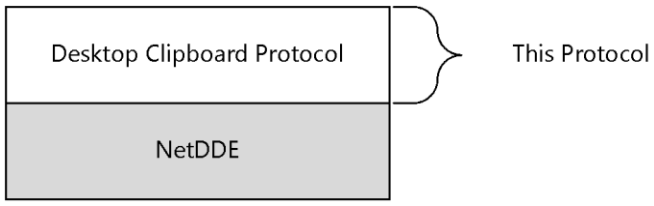


Figure 1: Relationship to other protocols

1.5 Prerequisites/Preconditions

The Desktop Clipboard Protocol requires the **NetDDE** API [MSDN-NETDDE].

1.6 Applicability Statement

The Desktop Clipboard Protocol is used for transferring **clipboard** information between remote machines.

1.7 Versioning and Capability Negotiation

The Desktop Clipboard Protocol does not have multiple versions.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

This protocol references commonly used data types as defined in [MS-DTYP].

2.1 Transport

The **NetDDE** API is used to initiate and maintain the network connections used by this protocol.

Desktop Clipboard messages are encapsulated in data blocks delivered using NetDDE as specified in section 3.1. The NetDDE transport ensures that the total message size is known.

2.2 Message Syntax

Note All unsigned 16-bit and unsigned 32-bit values are specified in **little-endian** format. Depending on the hardware architectures of the **client** and the **server**, multiple-byte little-endian versus **big-endian** reordering can determine how values are marshaled by the sender and interpreted by the receiver.

2.2.1 Common Field Values

2.2.1.1 ClipboardFormatName

The ClipboardFormatName constants are null-terminated **ANSI** strings ([ISO/IEC-8859-1]) that specify the **clipboard format**.

Constant/value	Description
CF_BITMAP "&Bitmap"	A handle to a **bitmap**.
CF_DIB "&DIB Bitmap"	A memory object containing a **Device-independent bitmap (DIB)** structure ([MS-WMF] section 2.2.2.9), which consists of an information header followed by the bitmap bits.
CF_DIF "&DIF"	Software Arts' Data Interchange Format.
CF_DSPTEXT "Disp&lay Text"	Text display format associated with a private format.
CF_DSPBITMAP "Displa&y Bitmap"	Bitmap display format associated with a private format.
CF_DSPENHMETAFILE "Display En&hanced Metafile"	**EMF** display format associated with a private format.
CF_DSPMETAFILEPICT "Display Pict&ure"	**Metafile** picture display format associated with a private format.
CF_ENHMETAFILE "&Enhanced Metafile"	A handle to an EMF metafile [MS-EMF].
CF_METAFILEPICT "&Picture"	A handle to a metafile picture format.
CF_OEMTEXT "&OEM Text"	Text format containing characters in the OEM character set. Each line ends with a CR/LF combination. A null character signals the end of the data.
CF_PALETTE "Pal&ette"	A handle to a color **palette**.
CF_PENDATA "Pe&n Data"	Data for the pen extensions to the Microsoft Windows for Pen Computing.
CF_RIFF "&RIFF"	Represents audio data that is more complex than can be represented in a CF_WAVE standard wave format.

Constant/value	Description
CF_SYLK "&Sylk"	Microsoft Symbolic Link (SYLK) format.
CF_TEXT "&Text"	Text format. Each line ends with a carriage return/linefeed (CR/LF) combination. A null character signals the end of the data. Use this format for ANSI text.
CF_TIFF "T&IFF"	**Tag Image File Format (TIFF)**.
CF_UNICODETEXT "&Unicode Text"	**Unicode** text format. Each line ends with a CR/LF combination. A null character signals the end of the data.
CF_WAVE "&Wave Audio"	Represents audio data in one of the standard wave formats, such as 11 kHz or 22 kHz Pulse Code Modulation (PCM).

2.2.1.2 DDETopicType

The DDETopicType constants are character strings that specify the string constants used in **dynamic data exchange (DDE)** function calls.

Constant/value	Description
SZDDESYS_ITEM_TOPICS "Topics"	A list of the topics that are supported by the **server** at the current time.
SZDDESYS_TOPIC "System"	The system topic.

2.2.1.3 ExecuteCommandType

The ExecuteCommandType enumeration specifies the type of command message sent between actors in a **clipbook**-sharing session.

Constant/value	Description
CMD_DELETE "[delete]"	The sender is directing the recipient to remove a clipbook entry from its set of **clipboard** shares.
CMD_INITSHARE "[initshare]"	The recipient has been requested to initialize its list of clipbook entries intended for sharing.
CMD_PASTE "[paste]"	The sender is directing the recipient to create a new clipbook entry.
CMD_SHARE "[markshared]"	The sender is directing the recipient to share a clipbook entry.
CMD_UNSHARE "[markunshared]"	The sender is directing the recipient to stop sharing a clipbook entry.

2.2.1.4 Notifications

The Notification constants are used by clients to request **server** action using the server's **DdeCallback** function.

Constant/value	Description
XTYP_ADVREQ 0x2022	Informs the server that a data request is outstanding on the specified topic name and item name pair, and that data corresponding to the pair has changed.
XTYP_REQUEST 0x20B0	Requests data from the server.
XTYP_EXECUTE 0x4050	Sends a command string to the server.

2.2.1.5 PaletteEntryFlags

The PaletteEntryFlags constants are used to describe **palette** data sent between users in a **clipbook**-sharing session.

Constant/value	Description
PC_DEFAULT 0x0000	Specifies default behavior for the **logical palette** entry.
PC_RESERVED 0x0001	Specifies that the logical palette entry be used for palette animation. This flag prevents other windows from matching colors to the palette entry since the color frequently changes. If an unused **system palette** entry is available, the color is placed in that entry. Otherwise, the color is not available for animation.
PC_EXPLICIT 0x0002	Specifies that the low-order word of the logical palette entry designates a hardware palette index. This flag allows the application to show the contents of the display device palette.
PC_NOCOLLAPSE 0x0004	Specifies that the color be placed in an unused entry in the system palette instead of being matched to an existing color in the system palette. If there are no unused entries in the system palette, the color is matched normally. Once this color is in the system palette, colors in other logical palettes can be matched to this color.

2.2.1.6 SharingStatusType

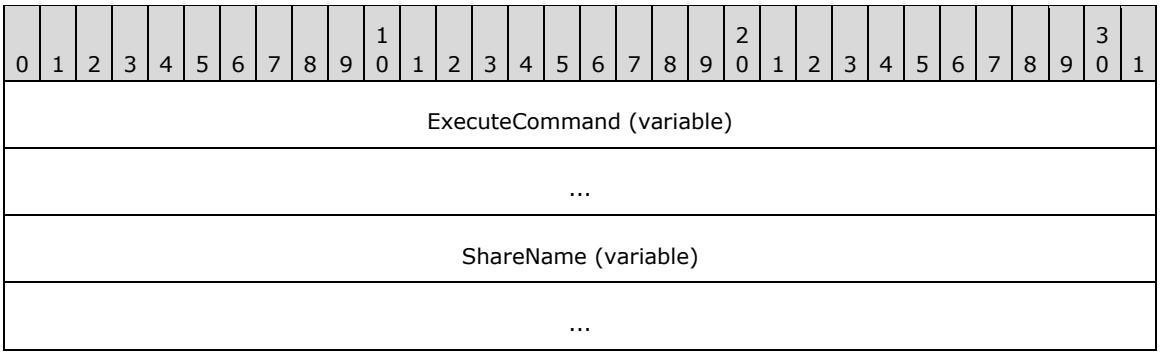
The SharingStatusType constants are **ANSI** characters that specify the sharing status of a **clipbook**.

Constant/value	Description
STATUS_SHARED "$"	The clipbook is shared.
STATUS_UNSHARED "*"	The clipbook is not shared.
STATUS_UPDATED "?"	The clipbook sharing status has been updated.

2.2.2 Control Information

2.2.2.1 EXECCOMMAND

The EXECCOMMAND structure specifies state changes for the **clipbook**-sharing session.

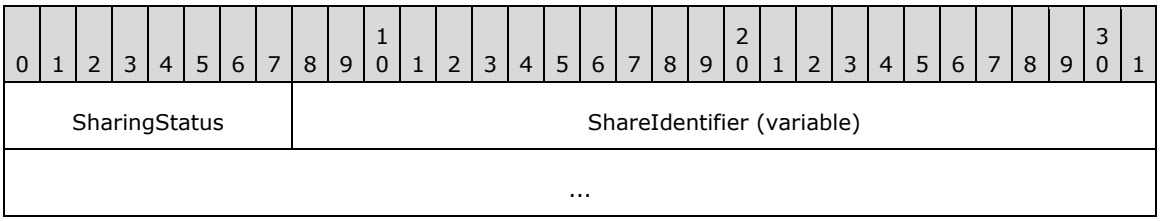


ExecuteCommand (variable): An array of **ANSI** characters that specifies the **ExecuteCommandType**. Note that this array is not zero-terminated. The ExecuteCommand value MUST be one of the **ExecuteCommandType** constants (section 2.2.1.3).

ShareName (variable): An optional, null-terminated ANSI string that specifies the share on which the command operates. This field MUST NOT be present if **ExecuteCommand** is CMD_INITSHARE, and it MUST be present if **ExecuteCommand** is not CMD_INITSHARE.

2.2.2.2 SHARE_LIST_ENTRYA

The SHARE_LIST_ENTRYA structure contains information about a shared **clipbook**.



SharingStatus (1 byte): An 8-bit unsigned integer that specifies the SharingStatusType (section 2.2.1.6).

ShareIdentifier (variable): An array of **ANSI** characters that specifies the name of the clipbook share. Note: this array is NOT zero-terminated.

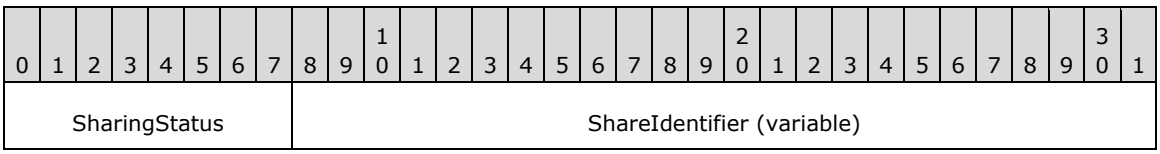
2.2.2.3 SHARE_LISTA

The SHARE_LISTA structure contains information about a set of **clipbooks**. A SHARE_LISTA structure conforms to the following ABNF [RFC5234]:

```
SHARE_LISTA = SHARE_LIST_ENTRYA *(%x09 SHARE_LIST_ENTRYA) %x00
```

2.2.2.4 SHARE_LIST_ENTRYW

The SHARE_LIST_ENTRYW structure contains information about a shared **clipbook**.



...

SharingStatus (1 byte): A 16-bit unsigned integer that specifies the SharingStatusType (section 2.2.1.6).

ShareIdentifier (variable): An array of **Unicode** characters that specifies the name of the clipbook share. Note: this array is NOT zero-terminated.

2.2.2.5 SHARE_LISTW

The SHARE_LISTW structure contains information about a set of **clipbooks**. A SHARE_LISTW structure conforms to the following ABNF [RFC5234]:

```
SHARE_LISTW = SHARE_LIST_ENTRYW *(%x00.09 SHARE_LIST_ENTRYW) %x00.00
```

2.2.2.6 CLIPFORMAT_LIST_ENTRYA

The CLIPFORMAT_LIST_ENTRYA structure describes a **clipboard format**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
ClipFormatName (variable)																															
...																															

ClipFormatName (variable): An array of **ANSI** characters that specifies the name of the clipboard format. For predefined clipboard formats, this value is a ClipboardFormatName constant (section 2.2.1.1). Note: this array is NOT zero-terminated.<1>

2.2.2.7 CLIPFORMAT_LISTA

The CLIPFORMAT_LISTA structure describes a set of **clipboard formats**. A CLIPFORMAT_LISTA structure conforms to the following ABNF [RFC5234]:

```
CLIPFORMAT_LISTA = CLIPFORMAT_LIST_ENTRYA *(%x09 CLIPFORMAT_LIST_ENTRYA) %x00
```

2.2.2.8 CLIPFORMAT_LIST_ENTRYW

The CLIPFORMAT_LIST_ENTRYW structure describes a **clipboard format**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
ClipFormatName (variable)																															
...																															

ClipFormatName (variable): An array of **Unicode** characters that specifies the name of the clipboard format. For predefined clipboard formats, this value is a ClipboardFormatName (section 2.2.1.1) constant. Note this array is NOT zero-terminated.<2>

2.2.2.9 CLIPFORMAT_LISTW

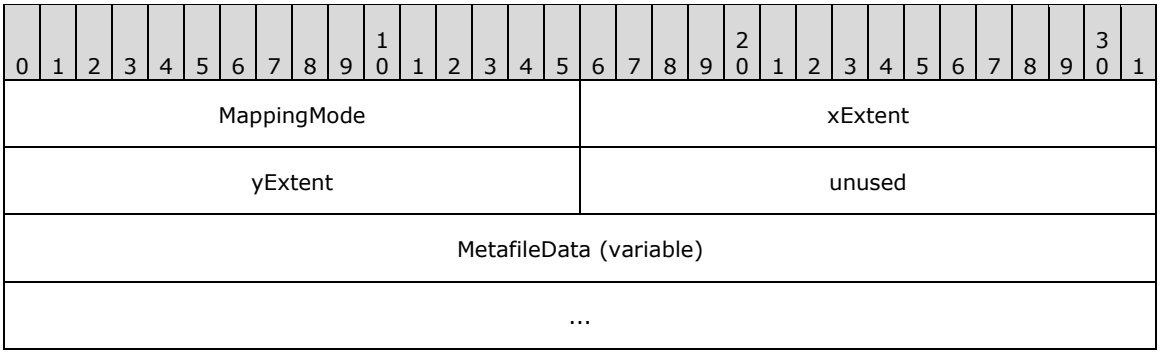
The CLIPFORMAT_LISTW structure describes a set of **clipboard formats**. A CLIPFORMAT_LISTW structure conforms to the following ABNF [RFC5234]:

```
CLIPFORMAT_LISTW = CLIPFORMAT_LIST_ENTRYW *(%x00.09 CLIPFORMAT_LIST_ENTRYW) %x00.00
```

2.2.3 Clipbook Data

2.2.3.1 CLIPDATA_METAFILEPICT

The CLIPDATA_METAFILEPICT structure contains **metafile** data.<3>



MappingMode (2 bytes): A 16-bit unsigned integer that specifies the **mapping mode** in which this picture is to be drawn.

xExtent (2 bytes): A 16-bit unsigned integer that specifies the width of the rectangle within which the picture is to be drawn. The coordinates are in units that correspond to the mapping mode.

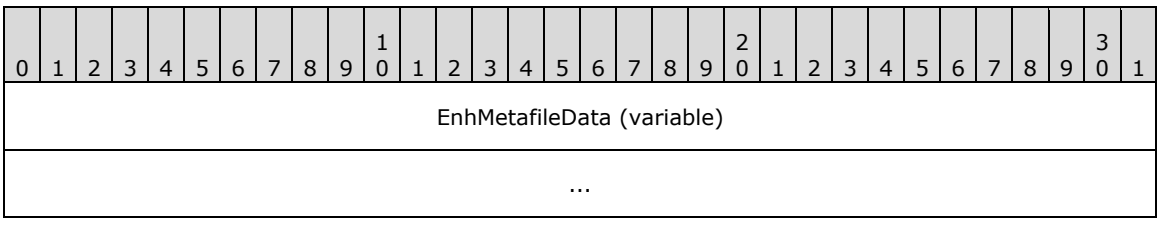
yExtent (2 bytes): A 16-bit unsigned integer that specifies the height of the rectangle within which the picture is to be drawn. The coordinates are in units that correspond to the mapping mode.

unused (2 bytes): Unused. SHOULD be zero.

MetafileData (variable): Data corresponding to a memory-based **Windows metafile format (WMF)** metafile [MS-WMF].<4>

2.2.3.2 CLIPDATA_ENHMETAFILE

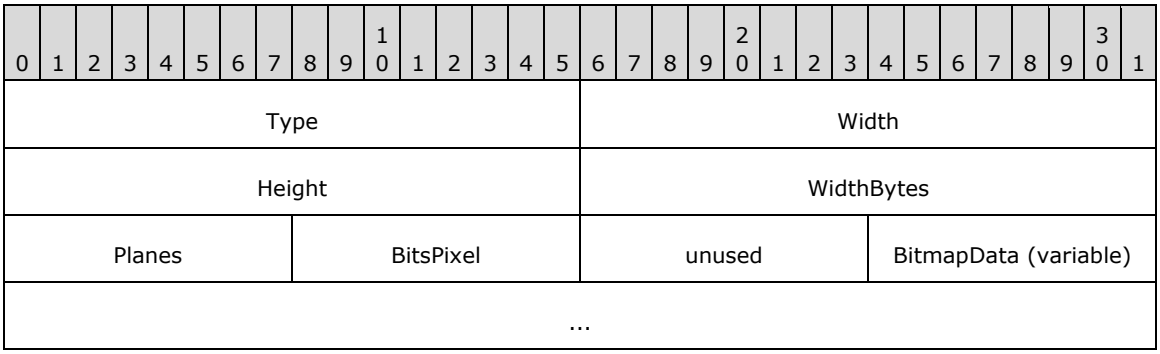
The CLIPDATA_ENHMETAFILE structure contains **EMF** data.



EnhMetafileData (variable): Data corresponding to a memory-based EMF **metafile** [MS-EMF].<5>

2.2.3.3 CLIPDATA_BITMAP

The CLIPDATA_BITMAP structure contains **bitmap** data.



Type (2 bytes): A 16-bit unsigned integer that specifies the bitmap type. MUST be set to 0x0000.

Width (2 bytes): A 16-bit unsigned integer that specifies the width, in pixels, of the bitmap.

Height (2 bytes): A 16-bit unsigned integer that specifies the height, in pixels, of the bitmap.

WidthBytes (2 bytes): A 16-bit unsigned integer that specifies the number of bytes in each scan line. This value MUST be divisible by 2, because the system assumes that the bit values in a bitmap form an array that is word aligned.

Planes (1 byte): An 8-bit unsigned integer that specifies the count of **color planes**.

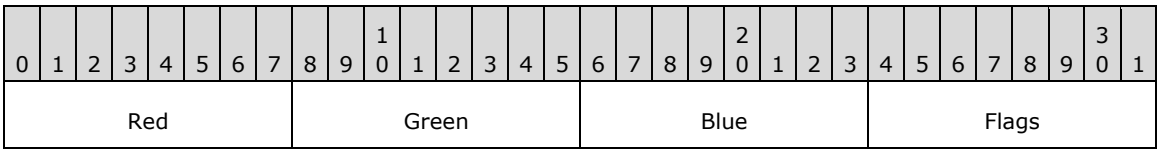
BitsPixel (1 byte): An 8-bit unsigned integer that specifies the number of bits required to indicate the color of a pixel.

unused (1 byte): Unused. SHOULD be zero.

BitmapData (variable): An array of byte values forming the bitmap data as specified by the previous fields.

2.2.3.4 CLIPDATA_PALETTE_ENTRY

The CLIPDATA_PALETTE_ENTRY structure contains **palette** color information.



Red (1 byte): An 8-bit unsigned integer that specifies the red **intensity** value for the palette entry.

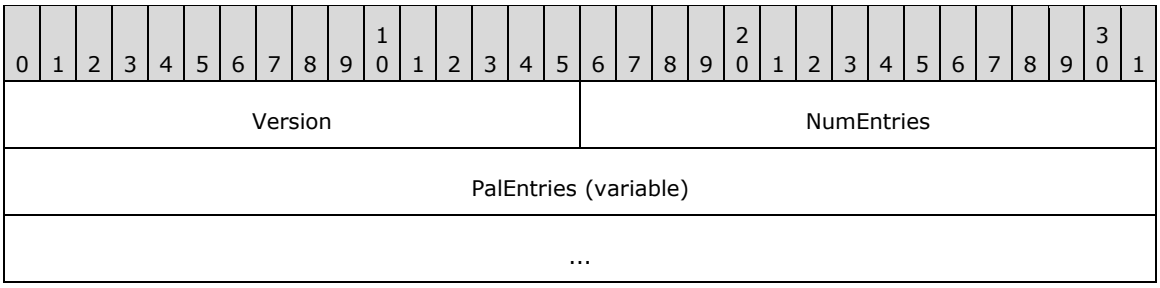
Green (1 byte): An 8-bit unsigned integer that specifies the green intensity value for the palette entry.

Blue (1 byte): An 8-bit unsigned integer that specifies the blue intensity value for the palette entry.

Flags (1 byte): An 8-bit unsigned integer that specifies the PaletteEntryFlags (section 2.2.1.5) usage of the palette entry.

2.2.3.5 CLIPDATA_PALETTE

The CLIPDATA_PALETTE structure contains **palette** data.



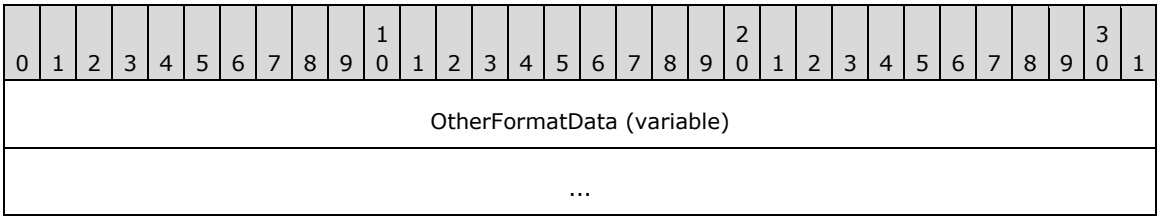
Version (2 bytes): A 16-bit signed integer that specifies the version number of the system. MUST be set to 0x0300.

NumEntries (2 bytes): A 16-bit unsigned integer that specifies the number of entries in **PalEntries**.

PalEntries (variable): A series of **NumEntries** CLIPDATA_PALETTE_ENTRY (section 2.2.3.4) structures that specifies the palette information.

2.2.3.6 CLIPDATA_OTHERFORMATS

The CLIPDATA_OTHERFORMATS structure contains data corresponding to arbitrary **clipboard formats**.



OtherFormatData (variable): Data corresponding to arbitrary clipboard formats.<6>

3 Protocol Details

In the Desktop Clipboard Protocol, a **server** allows a **client** to access a set of shared **clipbooks**. Implementations can simultaneously participate in both client and server roles to allow for two-way sharing of clipbooks. Implementations can represent the local **clipboard** as a clipbook shared by a server, so that client software can interact agnostically with the local clipboard and remote clipbooks.

In the following diagram, a client and server first negotiate a **DDE** conversation as described in sections 3.1.3 and 3.2.3.

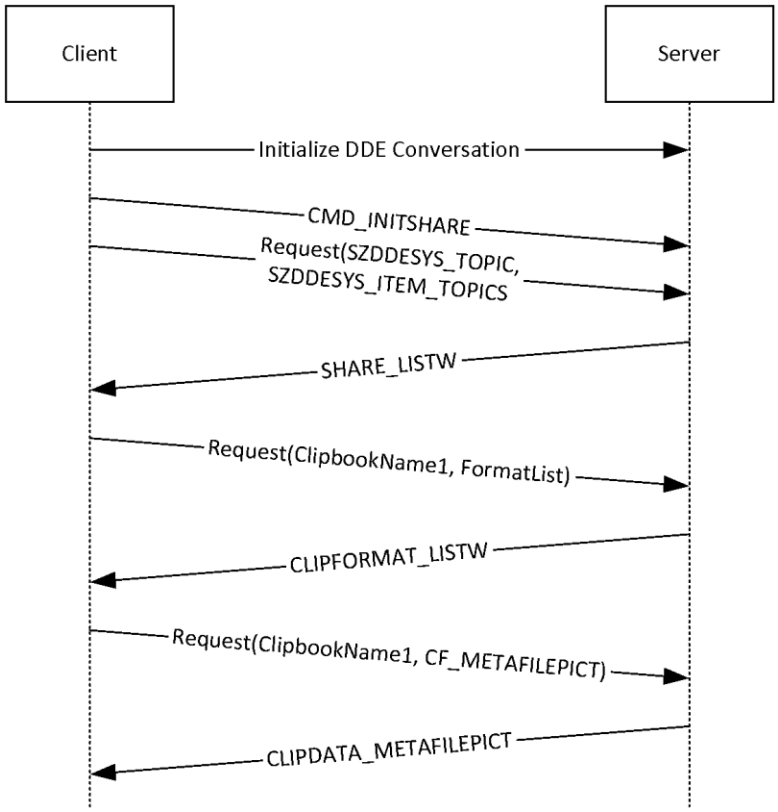


Figure 2: Clipboard sharing session

The following messages are then transmitted:

1. The client sends a CMD_INITSHARE (section 3.1.5.1.1) message, directing the server to initialize its **SharedClipbookData** information.
2. The client sends a DDE request with SZDDESYS_TOPIC as the topic and SZDDESYS_ITEM_TOPICS as the item (section 2.2.1.2).
3. The server sends a SHARE_LISTW (section 2.2.2.5) message containing the set of shared clipbooks.
4. Knowing the available clipbooks, the client sends a DDE request with "ClipbookName1" as the topic and "FormatList" as the item.
5. The server sends a CLIPFORMAT_LISTW (section 2.2.2.9) message containing the set of supported formats for the "ClipbookName1" clipbook.

6. Knowing the supported **clipboard formats**, the client sends a DDE request with "ClipbookName1" as the topic and **CF_METAFILEPICT** as the item.

7. The server sends a CLIPDATA_METAFILEPICT (section 2.2.3.1) message containing the **metafile** information for the "ClipbookName1" clipbook. The client renders this information to the user.

In this exchange, the set of supported formats returned in step 5 includes CF_METAFILEPICT; but in step 6, the client can request any other clipboard format, using one of the ClipboardFormatName (section 2.2.1.1) values that is supported in the set returned in step 5. The client can continue by requesting other formats.

3.1 Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that a **server** implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

Abstractly, **clipbook** information is kept as follows:

ClipbookData.ClipbookName: A string uniquely identifying the clipbook.

ClipbookData.SharingStatus: The SharingStatusType (section 2.2.1.6) value of the clipbook.

ClipbookData.Formats: An array of **clipboard formats** present in the clipbook.

ClipbookData.Formats[N].FormatName: The ClipboardFormatName (section 2.2.1.1) value identifying the clipboard format within the clipbook.

ClipbookData.Formats[N].Data: The data contained for the clipboard format within the clipbook.

Servers of the Desktop Clipboard Protocol SHOULD maintain the following state.

SharedClipbookData: An array of **ClipbookData** entries representing the clipbooks shared by the server.

3.1.2 Timers

None.

3.1.3 Initialization

A **NetDDE** server is initialized as follows. See [MSDN-NETDDE] for additional information.

The NetDDE server machine creates a static network **DDE** share using NDdeShareAdd ([MSDN-NDdeShareAdd]). The DDE share MUST be named "CLPBK$" and MUST have the static topic list of "ClipSrv|System".

The client initiates the NetDDE conversation as specified in section 3.2.3.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

Malformed, unrecognized, and out-of-sequence packets MUST be ignored by the server.

3.1.5.1 Command Message Processing

The server MUST handle command messages received as XTYP_EXECUTE notifications (section 2.2.1.4) to its **DdeCallback** function. An EXECCOMMAND (section 2.2.2.1) data block is extracted using the **DdeGetData** function and is processed as follows.

3.1.5.1.1 CMD_INITSHARE

The server MUST perform any initialization required to populate **SharedClipbookData**. An implementation can load shared **clipbook** data from a user-configured persisted storage location.

3.1.5.1.2 CMD_DELETE

The server MUST remove the **ClipbookData** with the **ClipbookName** specified by **EXECCOMMAND.ShareName** from **SharedClipbookData**.

3.1.5.1.3 CMD_SHARE

The server MUST find the **ClipbookData** with the **ClipbookName** specified by **EXECCOMMAND.ShareName** and set **SharingStatus** to STATUS_SHARED.

3.1.5.1.4 CMD_UNSHARE

The server MUST find the **ClipbookData** with the **ClipbookName** specified by **EXECCOMMAND.ShareName** and set **SharingStatus** to STATUS_UNSHARED.

3.1.5.1.5 CMD_PASTE

The server MUST add the **ClipbookData** with the **ClipbookName** specified by **EXECCOMMAND.ShareName** to **SharedClipbookData**.

3.1.5.2 Responses to Data Requests

The server MUST handle data request messages received as XTYP_ADVREQ or **XTYP_REQUEST** notifications (section 2.2.1.4) to its **DdeCallback** function.

If the **DdeCallback** **DDE** topic is SZDDESYS_TOPIC and the **DdeCallback** item is SZDDESYS_ITEM_TOPICS (section 2.2.1.2):

- If the requested **clipboard format** is CF_TEXT, the server MUST respond by creating a SHARE_LISTA (section 2.2.2.3) structure corresponding with **SharedClipbookData** and returning it via DdeCreateDataHandle ([MSDN-DdeCreateDataHandle]).
- If the requested clipboard format is CF_UNICODETEXT, the server MUST respond by creating a SHARE_LISTW (section 2.2.2.5) structure corresponding with **SharedClipbookData** and returning it via DdeCreateDataHandle.

If the **DdeCallback** item is "FormatList":

- If the requested clipboard format is CF_TEXT, the server MUST respond by creating a CLIPFORMAT_LISTA (section 2.2.2.7) structure corresponding to the **ClipbookData** with the **ClipbookName** specified by the **DdeCallback** topic and returning it via DdeCreateDataHandle.

- If the requested clipboard format is CF_UNICODETEXT, the server MUST respond by creating a CLIPFORMAT_LISTW (section 2.2.2.9) structure corresponding to the **ClipbookData** with the **ClipbookName** specified by the **DdeCallback** topic and returning it via DdeCreateDataHandle.

In all other cases, the server MUST find the **clipboard** data with the **ClipbookName** specified by the **DdeCallback** topic and the **FormatName** specified by the **DdeCallback** item. This data is then serialized and returned to the client via DdeCreateDataHandle as follows:

- If the requested clipboard format is CF_ENHMETAFILE, the server MUST return a CLIPDATA_ENHMETAFILE (section 2.2.3.2) structure.
- If the requested clipboard format is CF_METAFILEPICT, the server MUST return a CLIPDATA_METAFILEPICT (section 2.2.3.1) structure.
- If the requested clipboard format is CF_PALETTE, the server MUST return a CLIPDATA_PALETTE (section 2.2.3.5) structure.
- If the requested clipboard format is CF_BITMAP or CF_DIB, the server MUST return a CLIPDATA_BITMAP (section 2.2.3.3) structure.
- Otherwise, the server MUST return a CLIPDATA_OTHERFORMATS (section 2.2.3.6) structure.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Client Details

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that a **client** implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

The Desktop Clipboard Protocol provides the means for clients to request the **SharedClipbookData** information from the server, including enumerating the shared **clipbooks** and retrieving the clipbook data.

A client application can render this information in a form visible to the user, allow the user to copy the clipbook contents into their local system **clipboard**, or allow the user to save the clipbook contents.

3.2.2 Timers

None.

3.2.3 Initialization

A **NetDDE** conversation is initialized by a **client** as follows. See [MSDN-NETDDE] for additional information.

1. The NetDDE client machine initiates the **DDE** conversation using DdeConnect ([MSDN-DdeConnect]). The DdeConnect service name MUST be of the form "\\computername\NDDE$", where computername is the **NetBIOS** name of the server machine. The DdeConnect topic name MUST be "CLPBK$".

2. The client then sends an XTYP_EXECUTE notification (section 2.2.1.4) to the server with a CMD_INITSHARE (section 3.1.5.1.1) message as the contents of DdeClientTransaction ([MSDN-DdeClientTransaction]).

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

Malformed, unrecognized, and out-of-sequence packets MUST be ignored by the client.

3.2.5.1 Command Messages

A client can send command messages as XTYP_EXECUTE notifications (section 2.2.1.4) to the **server's DdeCallback** function. An EXECCOMMAND (section 2.2.2.1) data block is set up according to the specific command.

3.2.5.2 Data Requests

A client can send data request messages as XTYP_ADVREQ or XTYP_REQUEST notifications (section 2.2.1.4) to the **server's DdeCallback** function.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

In this example, the **server** machine is sharing a **clipboard** entry called "ShareName" which contains "Sample Text" as its textual data.

1. The **client** makes a SZDDESYS_ITEM_TOPICS request (section 2.2.1.2) of the server.

   The following is the hexadecimal representation of the SHARE_LISTA (section 2.2.2.3) data the server returns:

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0000	3F	09	24	53	68	61	72	65	4E	61	6D	65	00	00	00	00

   - SharingStatus: (1 byte, offset 0x0000), 0x3F is STATUS_UPDATED.
   - ShareIdentifier: (0 bytes, offset 0x0001), empty value.
   - Tab delimiter: (1 byte, offset 0x0001), 0x09 as required.
   - SharingStatus: (1 byte, offset 0x0002), 0x24 is STATUS_SHARED.
   - ShareIdentifier: (9 bytes, offset 0x0003), "ShareName" in **ASCII**.
   - Null terminator: (1 byte, offset 0x000C), 0x00 as required.

2. The client recognizes the "ShareName" share is shared and requests its supported **clipboard formats** with the "FormatList" **DDE** message.

   The following is the hexadecimal representation of the CLIPFORMAT_LISTA (section 2.2.2.7) data the server returns:

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0000	26	55	6E	69	63	6F	64	65	20	54	65	78	74	09	09	26
0010	54	65	78	74	09	26	4F	45	4D	20	54	65	78	74	09	43
0000	6C	69	70	62	6F	6F	6B	20	50	72	65	76	69	65	77	00

   - ClipFormatName: (13 bytes, offset 0x0000), "&Unicode Text" in ASCII.
   - Tab delimiter: (1 byte, offset 0x000D), 0x09 as required.
   - ClipFormatName: (0 bytes, offset 0x000E), empty value.
   - Tab delimiter: (1 byte, offset 0x000E), 0x09 as required.
   - ClipFormatName: (5 bytes, offset 0x000F), "&Text" in ASCII.
   - Tab delimiter: (1 byte, offset 0x0014), 0x09 as required.
   - ClipFormatName: (9 bytes, offset 0x0015), "&OEM Text" in ASCII.
   - Tab delimiter: (1 byte, offset 0x001E), 0x09 as required.
   - ClipFormatName: (16 bytes, offset 0x001F), "Clipbook Preview" in ASCII.

- Null terminator: (1 byte, offset 0x002F), 0x00 as required.

3. The client determines that **Unicode** text is its preferred format for consumption of the data, so it makes a request for CF_UNICODETEXT.

   The following is the hexadecimal representation of the CLIPDATA_OTHERFORMATS (section 2.2.3.6) data the server returns:

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0000	53	00	61	00	6D	00	70	00	6C	00	65	00	20	00	54	00
0010	65	00	78	00	74	00	00	00								

   - OtherFormatData: (20 bytes, offset 0x0000), "Sample Text" in Unicode characters.
   - Null terminator: (2 bytes, offset 0x0016), 0x0000 as required.

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Windows 2000 operating system Service Pack 4 (SP4)
- Windows XP operating system
- Windows Server 2003 operating system

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

<1> Section 2.2.2.6: In Windows environments, for non-predefined **clipboard formats** the **clipboard** name corresponds with return values of the **GetClipboardFormatName** function [MSDN-CLIPBOARD].

<2> Section 2.2.2.8: In Windows environments, for non-predefined clipboard formats, the clipboard name corresponds with return values of the **GetClipboardFormatName** function [MSDN-CLIPBOARD].

<3> Section 2.2.3.1: In Windows environments, these structure fields map to the **METAFILEPICT** structure [MSDN-METAFILEPICT].

<4> Section 2.2.3.1: In Windows environments, this is the data used by the **SetMetaFileBitsEx** function and returned by the **GetMetaFileBitsEx** function [MSDN-META].

<5> Section 2.2.3.2: In Windows environments, this is the data used by the **SetEnhMetaFileBits** function and returned by the **GetEnhMetaFileBits** function [MSDN-META].

<6> Section 2.2.3.6: In Windows environments, the arbitrary clipboard format data is obtained by performing a GlobalLock ([MSDN-GlobalLock]) operation on the handle returned by the **GetClipboardData** function [MSDN-CLIPBOARD].

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

A

Abstract data model
  client 21
  server 19
Applicability 9

C

Capability negotiation 9
CF_BITMAP 10
CF_DIB 10
CF_DIF 10
CF_DSPBITMAP 10
CF_DSPENHMETAFILE 10
CF_DSPMETAFILEPICT 10
CF_DSPTEXT 10
CF_ENHMETAFILE 10
CF_METAFILEPICT 10
CF_OEMTEXT 10
CF_PALETTE 10
CF_PENDATA 10
CF_RIFF 10
CF_SYLK 10
CF_TEXT 10
CF_TIFF 10
CF_UNICODETEXT 10
CF_WAVE 10
Change tracking 27
Client
  abstract data model 21
  higher-layer triggered events 22
  initialization 21
  local events 22
  message processing 22
    command message 22
    data requests 22
    overview 22
  other local events 22
  sequencing rules 22
    command message 22
    data requests 22
    overview 22
  timer events 22
  timers 21
CLIPDATA_BITMAP packet 16
CLIPDATA_ENHMETAFILE packet 15
CLIPDATA_METAFILEPICT packet 15
CLIPDATA_OTHERFORMATS packet 17
CLIPDATA_PALETTE packet 16
CLIPDATA_PALETTE_ENTRY packet 16
CLIPFORMAT_LIST_ENTRYA packet 14
CLIPFORMAT_LIST_ENTRYW packet 14
CMD_DELETE 11
CMD_INITSHARE 11
CMD_PASTE 11
CMD_SHARE 11
CMD_UNSHARE 11

D

Data model - abstract
  client 21
  server 19
Details 18

E

EXECCOMMAND packet 12

F

Fields – vendor extensible 9
Fields - vendor-extensible 9

G

Glossary 6

H

Higher-layer triggered events
  client 22
  server 19

I

Implementer - security considerations 25
Implementer – security considerations 25
Index of security parameters 25
Informative references 8
Initialization
  client 21
  server 19
Introduction 6

L

Local events
  client 22
  server 21

M

Message processing
  client 22
    command message 22
    data requests 22
    overview 22
  server 19
    command message
      CMD_DELETE 20
      CMD_INITSHARE 20
      CMD_PASTE 20
      CMD_SHARE 20
      CMD_UNSHARE 20
      overview 20
    overview 19
    response to data requests 20
Messages
  syntax 10
  transport 10

N

Normative references 8

O

Other local events
  client 22
  server 21
Overview 9
Overview (synopsis) 9

P

Parameters - security index 25
Parameters – security index 25
PC_DEFAULT 12
PC_EXPLICIT 12
PC_NOCOLLAPSE 12
PC_RESERVED 12
Preconditions 9
Prerequisites 9
Product behavior 26
Protocol Details
  overview 18

R

References 8
  informative 8
  normative 8
Relationship to other protocols 9

S

Security
  implementer considerations 25
  parameter index 25
Sequencing rules
  client 22
    command message 22
    data requests 22
    overview 22
  server 19
    command message
      CMD_DELETE 20
      CMD_INITSHARE 20
      CMD_PASTE 20
      CMD_SHARE 20
      CMD_UNSHARE 20
      overview 20
    overview 19
    response to data requests 20
Server
  abstract data model 19
  higher-layer triggered events 19
  initialization 19
  local events 21
  message processing 19
    command message
      CMD_DELETE 20
      CMD_INITSHARE 20
      CMD_PASTE 20
      CMD_SHARE 20
      CMD_UNSHARE 20
      overview 20
    overview 19
    response to data requests 20
  other local events 21
  sequencing rules 19
    command message
      CMD_DELETE 20
      CMD_INITSHARE 20
      CMD_PASTE 20
      CMD_SHARE 20
      CMD_UNSHARE 20
      overview 20
    overview 19
    response to data requests 20
  timer events 21
  timers 19
SHARE_LIST_ENTRYA packet 13
SHARE_LIST_ENTRYW packet 13
Standards assignments 9
STATUS_SHARED 12
STATUS_UNSHARED 12
STATUS_UPDATED 12
Syntax 10
SZDDESYS_ITEM_TOPICS 11
SZDDESYS_TOPIC 11

T

Timer events
  client 22
  server 21
Timers
  client 21
  server 19
Tracking changes 27
Transport 10
Triggered events - higher-layer
  client 22
  server 19

V

Vendor-extensible fields 9
Versioning 9

X

XTYP_ADVREQ 11
XTYP_EXECUTE 11
XTYP_REQUEST 11