The Component Object Model Specification

Version 0.9
October 24, 1995

This document contains the specification to the Component Object Model (COM), an architecture and supporting infrastructure for building, using, and evolving component software in a robust manner. This specification contains the standard APIs supported by the COM Library, the standard suites of interfaces supported or used by software written in a COM environment, along with the network protocols used by COM in support of distributed computing. This specification is still in draft form, and thus subject to change.

Note: This document is an early release of the final specification. It is meant to specify and accompany software that is still in development. Some of the information in this documentation may be inaccurate or may not be an accurate representation of the functionality of the final specification or software. Microsoft assumes no responsibility for any damages that might occur either directly or indirectly from these inaccuracies. Microsoft may have trademarks, copyrights, patents or pending patent applications, or other intellectual property rights covering subject matter in this document. The furnishing of this document does not give you a license to these trademarks, copyrights, patents, or other intellectual property rights.

Copyright © 1992-95 Microsoft Corporation. All Rights Reserved

The Component Object Model Specification
Draft Version 0.9, October 24, 1995
Microsoft Corporation and Digital Equipment Corporation

Copyright © 1992-95 Microsoft Corporation.

Microsoft does not make any representation or warranty regarding the Specification or any product or item developed based on the Specification. Microsoft disclaims all express and implied warranties, including but not limited to the implied warranties of merchantability, fitness for a particular purpose and freedom from infringement. Without limiting the generality of the foregoing, Microsoft does not make any warranty of any kind that any item developed based on the Specification, or any portion of it, will not infringe any copyright, patent, trade secret or other intellectual property right of any person or entity in any country. It is your responsibility to seek licenses for such intellectual property rights where appropriate. Microsoft shall not be liable for any damages arising out of or in connection with the use of the Specification, including liability for lost profit, business interruption, or any other damages whatsoever. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages; the above limitation may not apply to you.

0Table of Contents

How to Read This Document

Part I: Component Object Model Introduction

1. Introduction

1.1 Challenges Facing The Software Industry
1.2 The Solution: Component Software
1.3 The Component Software Solution: OLE's COM
1.4 Objects and Interfaces
1.5 Clients, Servers, and Object Implementors
1.6 The COM Library
1.7 COM as a Foundation

Part II: Component Object Model Programming Interface

2. Component Object Model Technical Overview

2.1 Objects and Interfaces
2.2 COM Application Responsibilities
2.3 The COM Client/Server Model
2.4 Object Reusability
2.5 Connectable Objects and Events
2.6 Persistent Storage
2.7 Persistent, Intelligent Names: Monikers
2.8 Uniform Data Transfer

3. Objects And Interfaces

3.1 Interfaces
3.2 Globally Unique Identifiers
3.3 The IUnknown Interface
3.4 Error Codes and Error Handling
3.5 Enumerators and Enumerator Interfaces
3.6 Designing and Implementing Objects

4. COM Applications

4.1 Verifying the COM Library Version
4.2 Library Initialization / Uninitialization
4.3 Memory Management
4.4 Memory Allocation Example

5. COM Clients

5.1 Identifying the Object Class
5.2 Creating the Object
5.3 Obtaining the Class Factory Object for a CLSID
5.4 Initializing the Object
5.5 Managing the Object
5.6 Releasing the Object
5.7 Server Management

6. COM Servers

6.1 Identifying and Registering an Object Class
6.2 Implementing the Class Factory
6.3 Exposing the Class Factory
6.4 Providing for Server Unloading
6.5 Object Handlers
6.6 Object Reusability
6.7 Emulating Other Servers

7. Interface Remoting

7.1 How Interface Remoting Works
7.2 Architecture of Custom Object Marshaling
7.3 Architecture of Standard Interface / Object Marshaling
7.4 Architecture of Handler Marshaling
7.5 Standards for Marshaled Data Packets

Copyright © 1992-95 Microsoft Corporation
All Rights Reserved

7.6 Creating an Initial Connection Between Processes
7.7 Marshaling Interface and Function Descriptions
7.8 Marshaling - Related API Functions
7.9 IMarshal interface
7.10 IStdMarshalInfo interface
7.11 Support for Remote Debugging

8. Security

8.1 Activation Security
8.2 Call Security

Part III: Component Object Model Protocols and Services

9. Connectable Objects

9.1 The *IConnectionPoint* Interface
9.2 The *IConnectionPointContainer* Interface
9.3 The *IEnumConnectionPoints* Interface
9.4 The *IEnumConnections* Interface

10. Persistent Storage

11. Persistent Intelligent Names: Monikers

11.1 Overview
11.2 IMoniker interface and Core Monikers
11.2

12. Uniform Data Transfer

Part IV: Type Information

13. Interface Definition Language

13.1 Object RPC IDL Extensions
13.2 Mapping from ORPC IDL to DCE RPC IDL

14. Type Libraries

Part V: The COM Library

15. Component Object Model Network Protocol

15.1 Overview
15.2 Service Control Manager
15.3 Data types and structures
15.4 IRemUnknown interface
15.5 The Object Exporter
15.6 Wrapping DCE RPC calls to interoperate with ORPC
15.7 Implementing ORPC in RPC

Appendix B: Bibliography

Appendix C: Specification Revision History

Appendix D: Index

1How to Read This Document

This specification is written to help a variety of readers understand the design and implementation of the Component Object Model (referred to herein simply as COM) as much as they would like. The presentation of COM gradually progresses from high-level overviews to COM benefits and eventually into the underlying mechanisms and programming interfaces to COM. This section is intended to help the reader determine what parts of this document to read.

This specification is divided into four parts, each of which contains one or more chapters. Part I is an overview and introduction. Chapter 1, the only chapter in Part I, explains at a high level the motivations of COM and the problems it addresses. It describes what COM is and its features, and describes the major benefits and advantages of COM. All readers should be interested in this chapter.

Part II contains the programming interface to COM, the suite of interfaces and APIs by which Component Object Model software is implemented and used. Chapters 2 through 8 are in Part II.

Chapter 2 goes into more detail about COM features and mechanisms without getting into the details of function call specifications and code. The chapter is intended for technical readers who want to know more than simply what COM is and what problems it solves, and therefore delves deeper into how applications use COM and the benefits of such use.

Chapters 3-6 contain programming-level information for readers who are interested in actually making use of COM in an application. These chapters explain the fundamentals of objects in COM and the creation of object clients as well as object servers. Chapter 3 details the basic object structures and mechanisms and provides the functional specifications of the core of COM. Chapter 4 covers the COM programming interfaces that all applications making use of COM must follow. Chapter 5 then deals specifically with COM clients; Chapter 6 specifically with COM servers.

Chapter 7 contains more detailed information about how COM clients and servers communicate with objects. This information is generally needed only by sophisticated programmers. Nevertheless, program-

This page left intentionally blank.

mers may find this chapter enlightening and can gain a clear understanding of all the underlying mechanisms that make COM truly powerful.

Chapter 8 contains information on how communications between COM clients and severs can be made secure.

Part III (Chapters 9-12) provides the functional specifications for the extended features of COM, including storage, naming, and exchange of data. These added features are built on top of the core COM functionality described in the previous chapters.

Part IV specifies standards relating to tools used to assist the authorship of COM software. It includes Chapter 13, which specifies the COM extensions to the standard Interface Definition Language (IDL) of the Open Software Foundation (OSF) Distributed Computing Environment (DCE). This will be of interest primarily to tools vendors who support tools that work with this language. Chapter 14 covers Type Libraries which are the binary equivalent to IDL.

Finally, Part V specifies information needed by programmers who will be implementing COM on other platforms—that is, the programmer who will be implementing COM on a systems level rather than an application level. Within Part V, Chapter 15 specifies the protocol used by COM when performing distributed computing between machines over a network. This chapter heavily references the OSF DCE RPC specification, noted in the Bibliography as [CAE RPC].

Copyright © 1992-95 Microsoft Corporation
All Rights Reserved

2Part III: Component Object Model Protocols and Services

3Appendix B: Bibliography

[CAE RPC]	*CAE Specification, X/Open DCE: Remote Procedure Call,* X/Open Company Limited, Reading, Berkshire, UK (xopen.co.uk), 1994. X/Open Document Number C309. ISBN 1-85912-041-5.

This page intentionally left blank.

Copyright © 1992-95 Microsoft Corporation
All Rights Reserved

4Appendix C: Specification Revision History

06 March 1995	First major draft.
24 October 1995	Second major draft.

This page intentionally left blank.

5Appendix D: Index

aggregation, 72, 130
APIs
    CoCopyProxy, 166
    CoGetCallContext, 168
    CoImpersonateClient, 169
    CoInitializeSecurity, 162
    CoQueryAuthenticationServices, 163
    CoQueryClientAuthenticationInfo, 168
    CoQueryProxyAuthenticationInfo, 166
    CoRegisterAuthenticationService, 163
    CoRevertToSelf, 169
    CoSetProxyAuthenticationInfo, 166
artificial reference counts, 74

Causality ID, 202
CLSCTX, 97
CoBuildVersion, 87
CoCopyProxy, 166
CoCreateInstance, 99
CoCreateInstanceEx, 99
CoDisconnectObject, 116, 144
CoFreeUnusedLibraries, 104
CoGetCallContext, 168
CoGetClassObject, 96
CoGetCurrentProcess, 91
CoGetMalloc, 41, 90
CoGetMarshalSizeMax, 146

DebugORPCClientFillBuffer, 155
DebugORPCClientGetBufferSize, 155
Delta Pinging, 201
direct mode, 54
DllCanUloadNow, 115
DllGetClassObject, 110

endpoints, 213
Enumerators, 79
Error Info, 210
exceptions, 74
Extended Error Info, 210

facility codes, 75
FACILITY_CONTROL, 76
FACILITY_DISPATCH, 75
FACILITY_ITF, 75, 76
FACILITY_NULL, 75
FACILITY_RPC, 75
FACILITY_STORAGE, 75
FACILITY_WIN32, 75
FACILITY_WINDOWS, 76
FAILED, 78
Fault PDU, 198

This page intentionally left blank.

CoGetStandardMarshal, 145
CoGetTreatAsClass, 122
CoImpersonateClient, 169
CoInitialize, 88
CoInitializeSecurity, 162
CoIsHandlerConnected, 104
CoMarshalInterface, 142, 160
COMSERVERINFO, 97
Constants
    RPC_C_AUTHN, 162
    RPC_C_IMP, 162
context handles, 200
CoQueryAuthenticationServices, 163
CoQueryClientAuthenticationInfo, 168
CoQueryProxyAuthenticationInfo, 166
CoRegisterAuthenticationService, 163
CoRegisterClassObject, 112
CoReleaseMarshalData, 141, 145
CoRevertToSelf, 169
CoRevokeClassObject, 113
CoSetProxyAuthenticationInfo, 166
CoTaskMemAlloc, 41, 91
CoTaskMemFree, 91
CoTaskMemRealloc, 91
CoTreatAsClass, 122
CoUninitialize, 88
CoUnmarshalInterface, 144, 160
CTextRender, 84
    functions
        Load, 92
CTextRenderFactory
    functions
        CreateInstance, 108

Debug, 210
Debug Information Body Extension, 210

GUID, 69

handler, 44, 97, 104, 106, 116, 131
handler marshalling, 149
HRESULT, 75
HRESULT_CODE, 78
HRESULT_FACILITY, 78
HRESULT_SEVERITY, 78

IClassFactory
    functions
        CreateInstance, 95
        LockServer, 96
IClassFactory Interface, 95
IClientSecurity, 164
    functions
        CopyProxy, 165
        QueryBlanket, 164
        SetBlanket, 165
IConnectionPoint, 172
        GetConnectionInterface, 173
        GetConnectionPointContainer, 173
        Advise, 173
        Unadvise, 174
        EnumConnections, 175
IConnectionPointContainer, 175
        EnumConnectionPoints, 175
        FindConnectionPoint, 176
ICreateErrorInfo, 210
identity, 70
IEnum
    functions
        Clone, 81

Copyright © 1992-95 Microsoft Corporation
All Rights Reserved

        - ***Next***, 80
        - ***Reset***, 80
        - Skip, 80
- **IEnumConnectionPoints**, 176
        - Next, 177
        - Skip, 178
        - Reset, 178
        - Clone, 179
- **IEnumConnections**, 179
        - Next, 180
        - Skip, 180
        - Reset, 181
        - Clone, 181
- IErrorInfo, 210
- IMalloc
    - functions
        - Alloc, 89
        - **DidAlloc**, 90
        - **Free**, 89
        - **GetSize**, 89
        - **HeapMinimize**, 90
        - **Realloc**, 89
- ***IMalloc Interface***, 88
- IMarshal, 208
    - functions
        - Disconnect, 149
        - GetMarshalSizeMax, 148
        - GetUnmarshalClass, 147
        - MarshalInterface, 147
        - ReleaseMarshalData, 149
        - UnmarshalInterface, 148
- in parameter, 41
- in-out parameter, 41
- interface
    - definition, 65
- *Interfaces*
    - *IClientSecurity*, 164
    - *IServerSecurity*, 166
- IObjectExporter, 199, 201
    - **functions**
        - **ComplexPing**, 220
        - **ResolveOxid**, 215
        - **SimplePing**, 219
- IPID, 198
- IPSFactoryBuffer, 130, 132
    - functions
        - CreateProxy, 132
        - CreateStub, 133
- IRemUnknown, 211
    - functions
        - **RemAddRef**, 212
        - RemQueryInterface, 202, 212
        - **RemRelease**, 213
- IRpcChannelBuffer, 130, 133
    - functions
        - FreeBuffer, 137
        - GetBuffer, 135
        - GetDestCtx, 137
        - IsConnected, 138
        - SendReceive, 136
        - SendRecieve, 141
- IRpcProxyBuffer, 119, 130, 138
    - functions
        - Connect, 138
        - Disconnect, 138
- IRpcStubBuffer, 130
    - functions
        - Connect, 139
        - CountRefs, 141
        - DebugServerQueryInterface, 141
        - DebugServerRelease, 142
        - Disconnect, 139
        - Invoke, 135, 136, 139
        - IsIIDSupported, 141
- *IServerSecurity*, 166
    - functions
        - ImpersonateClient, 167
        - QueryBlanket, 167
        - RevertToSelf, 168
- IStdMarshalInfo, 131, 207
    - **functions**
        - **GetClassForHandler**, 150
- ISupportErrorInfo, 210
- **IUnknown**
    - **functions**
        - **AddRef**, 71
        - **QueryInterface**, 70
        - **Release**, 71
- IUnknown Interface, 70

- ***MAKE_HRESULT***, 79
- ***Marshalled Interface References***, 199
- middleman, 208, 214, 216
- MSHCTX, 143
- MSHCTX_DIFFERENTMACHINE, 143
- MSHCTX_NOSHAREDMEM, 143
- MSHCTXDATA, 143
- MSHLFLAGS, 142
- MSHLFLAGS_NORMAL, 145
- MULTI_QI, 99

- Object Exporter, 199, 213
- **Object Exporter Well-known Endpoints**, 214
- ***Object Exporters***, 199
- Object Handlers, 116
- OBJREF, 160, 199, 207
- **OBJREF_CUSTOM**, 208
- **OBJREF_HANDLER**, 207
- **OBJREF_LONGHDLR**, 208
- **OBJREF_LONGSTD**, 207
- **OBJREF_NULL**, 207, 212
- **OBJREF_STANDARD**, 207
- OID, 199
- ORPCINFOFLAGS, 209
- ORPCTHAT, 198, 210
- ORPCTHIS, 198, 209
- out parameter, 41
- OXID, 199
- OXID object, 199, 211

- ping period, 201
- Ping periods, 220
- ***Pinging***, 200

- ***QueryInterface***, 202

- ***Reference Counting***, 200
- REGCLS, 112
- **Remote Debugging**, 150
- remote reference counting, 200
- Request PDU, 198, 209
- Response PDU, 198
- **RPC_C_AUTHN**, 162
- RPC_C_IMP, 162
- RPCOLEDATAREP, 134
- RPCOLEMESSAGE, 134
- RPCOLEMESSAGE and related structures, 133

- S_FALSE, 76
- S_OK, 76
- SORFLAGS, 208
- STDOBJREF, 199, 208
- StringFromCLSID, 131
- StringFromIID, 131
- ***SUCCEEDED***, 78

transacted mode, 54

Unicode, 64

well-known endpoints, 213, 214

Copyright © 1992-95 Microsoft Corporation
All Rights Reserved