DCOM: A Business Overview

Microsoft Corporation

April 1997

Summary: This paper provides a business overview of DCOM, a technology that enables software components to communicate directly with each other across networks, including the Internet and intranets.

Introduction

With the advent of the Java programming language and the growth of the Internet, information technology (IT) managers are again excited at the prospect of using component software technology—the idea of breaking large, complex software applications into a series of prebuilt and easily developed, understood, and changed software modules called components—to deliver software solutions much more quickly and at a lower cost.

A component architecture for building software applications will enable the software provider to achieve economies of scale for software deployment by:

- **Speeding development**—enabling programmers to build solutions faster by assembling software from prebuilt parts.
- **Lowering integration costs**—providing a common set of interfaces for software programs from different vendors means less custom work is required to integrate components into complete solutions.
- **Improving deployment flexibility**—making it easier to customize a software solution for different areas of a company by simply changing some of the components in the application.
- **Lowering maintenance costs**—isolating software function into discrete components provides a low-cost, efficient mechanism to upgrade a component without having to retrofit the entire application.

A distributed component architecture applies these benefits across a broader scale of multiuser applications. The Distributed Component Object Model (DCOM) has three unique strengths that make it a key technology for achieving these benefits:

- **DCOM is based on the most widely-used component technology today.**
  DCOM is simply "COM with a longer wire"—a low-level extension of the Component Object Model, the core object technology within Microsoft® ActiveX®. Major development tools vendors—including Microsoft, Borland, Powersoft/Sybase, Symantec, ORACLE, IBM, and Micro Focus—already sell software development tools that produce ActiveX components. These tools and the applications they produce automatically support DCOM, providing the broadest possible industry support. Additionally, over 1,000 existing commercial software components that will work with DCOM are already available for use by developers.

- **DCOM is the best networking technology to extend component applications across the Internet.**
  Because it is an ActiveX technology, DCOM works natively with Internet technologies like TCP/IP, Java, and HTTP, enabling business applications to work across the Web. DCOM enables distributed Java today without requiring any communications-specific code or add-ons.

- **DCOM is an open technology that runs on multiple platforms.**
  Microsoft is openly licensing DCOM technology to other software companies to run on all of the major operating systems, including multiple implementations of UNIX-based systems. Software AG has DCOM running on the Solaris-based operating system today. Additionally, Microsoft is handing over DCOM technology with other core ActiveX technologies to The Open Group. The Internet Draft technical publication that contains a publicly available description of the DCOM network protocol can be found at http://www.dc.luth.se/doc/id/draft-brown-dcom-v1-spec-00.txt.

The combination of these three factors—the largest installed base, native support for Internet protocols, and open support for multiple platforms—means that businesses can gain the benefits of a modern component application architecture without having to replace investments in existing systems, staff, or infrastructure.

Components and Desktop Development
Component-based development is established today as a mainstream business technology on the desktop.

DCOM has its roots in Microsoft's object technology, which has evolved over the last decade from DDE (Dynamic Data Exchange, a form of messaging between Windows® programs), OLE (Object Linking and Embedding, putting visual links between programs within an application), COM (the Component Object Model, used as the basis for all object binding), and ActiveX (COM enabled for the Internet). The evolution of this technology has a consistent theme: each iteration reduces the complexity of building large applications while enabling the delivery of successively richer functionality to the user. This can lower application development costs because developers can use prebuilt components and programming interfaces, greatly reducing the integration and test overhead required when integrating the work of multiple developers.

Applications built from components are easier to debug and add to than large, monolithic applications. As an example, consider the "Year 2000 Problem" which has many large organizations scrambling to fix their production systems to avoid failure when the date changes to the new millennium. In reality, this is an application design problem, not a date problem. If applications were written with a common date component, the fix would be easy to isolate and inexpensive to repair.

Most Windows developers understand these benefits and use the ActiveX component architecture. There are over three million professional programmers trained on ActiveX and its technologies—OLE, COM, and DCOM—and hundreds of independent software companies shipping prebuilt software components. These components can be used by developers working with Microsoft Visual Basic®, PowerBuilder, Micro Focus Visual Object COBOL, and other popular tools.

The key business benefits of ActiveX on the desktop automatically extend to DCOM:

- **Language-neutral**—developers can build ActiveX components from any language—ActiveX components built in COBOL can work with components built in Java.
- **Broad tools support**—most major software development tools vendors support ActiveX.
- **Broad industry support**—thousands of prebuilt components are available on the market.
- **Accessible technology**—a large pool of professional developers work with ActiveX now.

**Components and the Network**

The logical boundary for component applications is no longer on a single computer.

Companies want to apply the benefits of component software, rapid reuse, broad industry support, and availability of thousands of components across shared applications that operate on multiple computers. These types of applications are referred to as three-tier or n-tier applications, where tiers of application logic, presentation services, business services, and information retrieval and management services are broken into different components that can communicate directly with each other across a network. To the user, these applications appear as a seamless extension of their existing desktop environment. For the IT manager, they are an opportunity to apply the economics and flexibility of desktop development across a broader set of application problems.

For example, a business may deploy a new sales management system based on a multitier application design that uses components. The application includes different order entry components, each one designed for a separate sales channel. These components all use a common tax calculation component that runs on a server. As tax laws change, the company has to change only the tax component located on the server, without having to retrofit the order entry components for each of the different sales channels.

DCOM is an ideal technology for multitier applications because it enables ActiveX components to work across networks, enabling developers to easily build systems that span computer boundaries. Developers add components together without having to worry about network programming, system compatibility, or integration of components built from different languages. This can lower the cost and complexity of building distributed applications from components. DCOM leverages the investments companies have already made in ActiveX by providing the following benefits:

- **Growth path for desktop developers**—DCOM enables the millions of existing ActiveX programmers to build server components that natively interoperate with the desktop counterparts, thereby broadening their target audience.
Common component model for clients and servers—Companies can deploy the same set of developers for clients and servers, thus lowering training, tools, and support costs.

Multiplatform support—DCOM is designed to run on Windows 95, Windows NT®, Macintosh, UNIX, and legacy operating systems, providing companies with the basis for a common application infrastructure across their entire IT environment which can lower integration costs and reduce integration complexity.

Based on public, open standards—DCOM is layered on the Open Group Distributed Computing Environment (DCE) RPC mechanism, a network communications protocol developed and endorsed by IBM, Sun Microsystems, Hewlett-Packard, and Digital Equipment Corporation, a source code implementation of which is available for free from the Open Group. The additional object-related extensions to the DCE RPC required by DCOM are publicly available at http://www.dc.luth.se/doc/id/draft-brown-dcom-v1-spec-00.txt.

Components and the Internet

Businesses can use component-based applications to project a business presence onto the Web.

The simplicity, ubiquity, and industry momentum of standard Internet protocols such as the HyperText Transfer Protocol (HTTP) make the Internet an ideal technology for linking application components that span computer boundaries. HTTP is easy to program, is inherently cross-platform, and supports an accessible, universal naming service. And much of the excitement around the Java language derives from its potential as a tool to build distributed component applications for the Internet.

DCOM enables these component applications to operate across the Internet. DCOM is ideally positioned to become a mainstream Internet technology for business applications:

- **Transport-neutral**—DCOM enables components to communicate with each other over any network transport, both connection-oriented and connectionless, including TCP/IP, UDP/IP, IPX/SPX, AppleTalk, and HTTP.
- **Provides distributed Java today**—Since DCOM is language neutral, Java applets can communicate directly with each other over the Internet (and with any ActiveX component, regardless of authoring language).
- **Evolutionary technology**—In addition to Java support, DCOM enables components written in other languages, including C, COBOL, Basic, and Pascal, to communicate over the Internet, providing a growth path for existing applications to support Web technology.
- **Common components for the browser and Web server**—Since ActiveX components can be embedded into browser-based applications, DCOM enables a rich application infrastructure for distributed Internet applications using the latest browser technology.
- **Security**—DCOM integrates Internet certificate-based security with rich Windows NT-based security, combining the best of both worlds.
- **Standards-based**—Microsoft is working with Internet standards bodies, including the IETF and the W3C, to offer DCOM to the Internet community as an open technology. DCOM is based on the Open Group DCE RPC, an open and widely-deployed communications technology. The DCOM wire protocol extensions have been submitted as an Internet draft and are available at http://www.dc.luth.se/doc/id/draft-brown-dcom-v1-spec-00.txt.

Components and the Enterprise

As distributed applications built from simple components and Internet protocols emerge, a new set of enterprise platform services for component applications will be required.

A key goal of any component software architecture is to separate business logic—how a tax component calculates tax rates—from execution logic—whether the tax component runs in a browser or on a multiprocessor server. DCOM extends this separation even further because the same components can communicate with each other across processes in a single computer or between computers over the Internet.

However, components by themselves do not solve all of the issues of enterprise application complexity. For example, suppose a business wants to rapidly build and deploy a customer order entry application that involves five different areas of functionality: tax calculation, customer credit verification, inventory management, warranty update, and order entry. The application will be built from five separate components and will operate on a Web server. How does the developer handle exceptions? System failures? Network outages? Peaks in performance load? Must these be hand-coded into the application?
It defeats the two main goals of component-based development—fast time to market and lower development costs—if companies are forced to hand-code the mission-critical services that are required for online production systems.

To address enterprise requirements for a distributed component architecture without sacrificing rapid development and cost effectiveness, Microsoft is integrating DCOM into the Active Server. The Active Server is a series of technology services that speed deployment of component-based applications for the Internet and corporate intranets. These services include:

- **Transactions**—traditional rollback and recovery for component-based applications in the event of system failure.
- **Queuing**—integration of component communication with reliable store-and-forward queues, which enables component applications to operate on networks that are occasionally unavailable.
- **Server scripting**—easy integration of component applications on the server with HTML-based Internet applications.
- **Legacy access**—integration of component applications with legacy production systems, including mainframe systems running CICS and IMS.

The Active Server technologies use publicly obtainable Internet protocols and are currently available.

**DCOM Availability**


- DCOM first shipped with Windows NT Server 4.0 and Windows NT Workstation 4.0.
- DCOM for Windows 95 is available for download from [www.microsoft.com/com/dcom/dcom1_2/dcom1_2.asp](http://www.microsoft.com/com/dcom/dcom1_2/dcom1_2.asp).

© 1997 Microsoft Corporation. All rights reserved.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

ActiveX, BackOffice logo, Microsoft, Visual Basic, Windows, and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Other product or company names mentioned herein may be the trademarks of their respective owners.