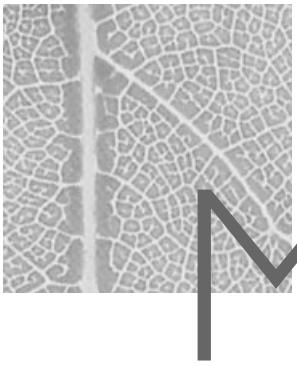
MCSE: Exchange 5.5 Study Guide 2nd Ed.

Contents at a Glance

1

Introduction to Microsoft Exchange



Microsoft Exchange is a client/server enterprise messaging product. "OK," you say, "but what is a client/server enterprise messaging product?" This chapter answers that question and shows how Microsoft Exchange meets the criteria. This chapter also discusses the major industry standards on which Microsoft Exchange is based. All of this information serves as an introduction to the Exchange product and to the topics in the remainder of the book.

In this chapter, we will address the following subjects:

- Messaging systems
- The client/server model
- Enterprise-wide computing
- Industry standards used by Exchange

The first three items relate to our previous question, "What is a client/server enterprise messaging product?" Messaging systems relate to *what* Exchange does, client/server models relate to *how* it does it, and enterprise-wide computing relates to the *context* in which it does it.

Messaging Systems

An enterprise needs information in order to get work done. Information is its oxygen. Frequently information *is* the work *and* the product (for example, a consulting company). In this context electronic messaging has become a mission-critical function in most organizations. While electronic mail (e-mail) is still the core ingredient, other applications are now included in

this category. The category of messaging can be divided into the following subcategories:

- E-mail
- Workflow
- Electronic forms
- Groupware
- Other messaging applications

Each of these categories, and how Exchange addresses them, are briefly discussed in the following text.

Due to the multiple functionality of some of the client programs, a single program could fit into more than one of the previous categories. For example, Microsoft Outlook includes e-mail functions and groupware functions like group scheduling.

E-Mail

An e-mail program allows a user to create, send, read, store, and manipulate electronic messages and attachments. E-mail is an example of *push-style* communication, meaning that the sender initiates the communication. Because of the importance of e-mail in the overall communication of organizations, these programs have evolved from merely creating and sending text messages, into multifeatured programs.

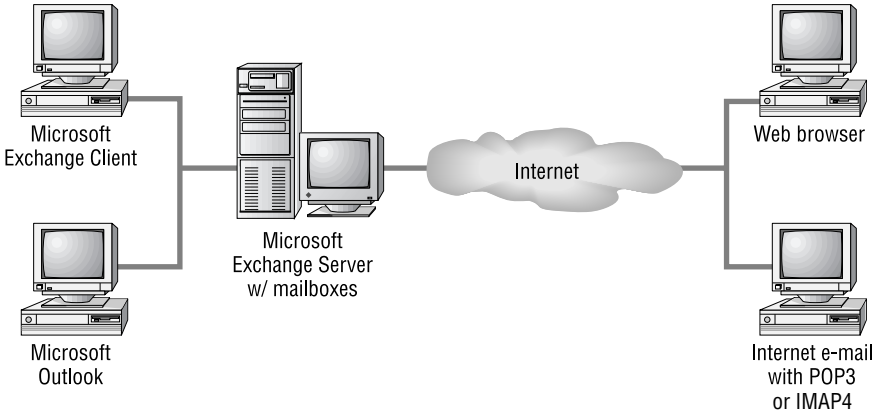
Microsoft Exchange ships with the Outlook client applications for Microsoft Windows 3.*x*, Windows 95 and Windows NT, and Macintosh. Also included is an Exchange client application for MS DOS.

Microsoft also has server components that enable Internet clients to be Exchange e-mail clients. Those Internet clients include:

- Web browsers
- Internet e-mail programs with the Post Office Protocol version 3 (POP3)
- Internet e-mail programs with Internet Message Access Protocol version 4 (IMAP4) support

Figure 1.1 illustrates these e-mail client applications.

FIGURE 1.1
E-mail clients to Exchange Server



Microsoft Exchange Client

The Exchange Client is a feature-rich e-mail program. Some of its features include the following:

- **Universal inbox (mailbox)** This central storage area can hold not only e-mail messages, but other data such as word processing documents, spreadsheet files, faxes, electronic forms, even voice mail files.

Two terms refer to a user's mailbox, *mailbox* and *inbox*. The most common usage in this book will be mailbox. This is our primary term for two reasons. One, Microsoft divides a mailbox into folders, one of which is labeled the inbox. Using the term inbox for only the folder helps prevent confusion. The other reason is that the Exchange object name for a mailbox is *mailbox*.

- **Hierarchical data storage** The Exchange Client organizes the client's mailbox into four default folders: Inbox, Outbox, Deleted Items, and Sent Items. Users can also create their own folders, thereby creating a personalized organization of their data.
- **Customized views** Users have the ability to determine what and how data is presented to them on their screens. Messages can be ordered by sender, date, priority, subject, and other properties.
- **Search tool** Users can search and retrieve messages in their mailboxes using a variety of search criteria, such as sender, date, and subject.

- **AutoAssistants** Exchange includes software routines that can automatically carry out actions based upon rules the user defines. This functionality is referred to as AutoAssistants. Some of the actions AutoAssistants can perform are alerting, deleting, moving, copying, forwarding, and replying.
- **Rich-text message content** Historically, most e-mail content was simple text. The Exchange Client message editor enables the creation of rich-text message content which can include multiple fonts, sizes, colors, alignments, and other formatting controls.
- **Microsoft Word as message editor** Even though the Exchange Client includes a rich message editor, it can be configured to use Microsoft Word as its message editor.
- **Compound messages and drag-and-drop editing** The Exchange Client program is OLE 2.0 (Object Linking and Embedding) compliant, and therefore allows the creation of compound documents. For example, a user could drag-and-drop a group of cells from a spreadsheet into an e-mail message.
- **Secure messages** Digital signatures and message encryption are advanced security features built into the Exchange Client.
- **Remote mail** Because more and more workers spend some of their work time outside of the office, special features relating to remote mail have been incorporated. One example is the ability to remotely access a mailbox and selectively download new mail to the remote computer, and to send out all outgoing mail that was created on the remote computer.
- **Delegate access** Some users need to allow other users to access their mailbox. For example, a manager might want a secretary to read meeting request messages in order to handle the manager's schedule. In many mail systems, this would be accomplished by having the secretary log on as the manager. This creates an obvious security hole. Microsoft solves this problem by allowing the manager to grant the secretary permission to access the manager's mailbox. This permission can be restricted to certain folders. The secretary can also be granted permission to send messages "on behalf of" the manager, or even send messages as the manager, called "send as."
- **Send/Receive electronic forms** Users can send and receive electronic forms through the Exchange Client application.

These are just some of the many features of the Exchange Client program. Chapters 6 and 7 discuss these features in greater detail.

Microsoft includes versions of this e-mail program for the following operating system platforms:

- MS-DOS
- Microsoft Windows 3.*x*
- Microsoft Windows for Workgroups
- Microsoft Windows 95
- Microsoft Windows NT
- Apple Macintosh

Microsoft Outlook

Outlook is a new e-mail client that ships with Exchange Server and Microsoft Office 97 as a stand-alone product. It has a 32-bit version designed for Windows 95 and Windows NT, as well as versions that run on Windows 3.x and Macintosh. Outlook is referred to as a desktop information manager because it is more than just an e-mail client. It also performs such tasks as calendaring, scheduling, and task and contact management. Outlook is intended to be a central program for client management of data.

Microsoft Outlook includes the feature set listed previously for the Exchange Client, but also adds some additional advanced features. Some of them that specifically relate to e-mail are listed here:

- **MessageFlags** These tags can be associated with e-mail messages to help the user prioritize follow-up action. MessageFlags include reply, read, and "for your information." Users can sort their messages by these MessageFlags, which helps with task management.
- **Voting** Outlook has the ability to add voting buttons in the header of a mail message and to collect the responses. This allows surveys to be conducted through e-mail.
- **MessageRecall** Permits a sent message to be recalled, provided the recipient has not opened the message. Users can also replace a sent message with a new message.
- **AutoPreview** Outlook can automatically display the first three lines of a mail message so users can quickly decide which messages to read or delete.

- **AutoCreate** Outlook can automatically convert one Outlook item into another. For example, a mail message may contain an action item which the user can simply drag-and-drop into the Task folder. Outlook would automatically convert the mail message into a task. This saves the user from retyping.
- **Recover deleted items** Enables users of Outlook version 8.03 to recover deleted items in a mailbox or public folder.
- **Expanded storage capacity** Users can store up to 64,000 items in their personal folder or an offline folder.

These are just some of the e-mail features of Microsoft Outlook. This client program will be discussed further in Chapters 6 and 7.

Web Browsers

Exchange Server 5.5 ships with a component called Outlook Web Access that can run on Microsoft Internet Information Server (IIS) and enable Web browsers to access Exchange resources such as mailboxes. Any standard Web browser can be used, such as Microsoft Internet Explorer or Netscape Navigator. This Exchange functionality permits users of other operating system platforms, such as UNIX or IBM's OS/2, to also be Exchange clients. Chapter 11 covers the Exchange components required for web browser clients.

Internet E-Mail Programs with POP3

Exchange has built-in support for the Post Office Protocol version 3 (POP3). POP enables mail clients to retrieve mail messages stored on a remote mail server. Exchange's support for this protocol allows Internet e-mail programs that support POP3 to access their Exchange mailbox and download their messages. Chapter 11 will cover POP in the Exchange environment.

Internet E-Mail Programs with IMAP4

Exchange also has built-in support for the Internet Message Access Protocol version 4 (IMAP4). IMAP is similar to POP in that it is a mail retrieval protocol. But IMAP has more features than POP, such as the ability to select the messages to download rather than having to download all new messages. Chapter 11 will provide further details on IMAP.

Electronic Forms

Electronic forms are electronic messages with built-in fields. They can be used instead of paper-based forms to automate and streamline organizational processes. Items such as expense reports, order entry, and purchase requests can all be implemented with electronic forms.

The two Exchange e-mail clients—Exchange Client and Outlook—can send and receive electronic forms. Exchange also ships with two programs that can be used to create customized electronic forms:

- Microsoft Exchange Forms Designer
- Microsoft Outlook Forms Designer

Microsoft Exchange Forms Designer

The Exchange Forms Designer is a visual, nonprogrammatic design and creation tool for electronic forms. Users simply draw a form's appearance and choose its behavior. The Forms Designer program is actually a special version of Microsoft's Visual Basic development tool. Because of this, forms created in the Forms Designer environment can be further modified in Visual Basic.

Electronic forms can be used with two types of applications:

- Stand-alone form applications
- Folder-based applications

Stand-alone form applications are electronic forms that are going to be sent from one person to another person or persons. An example is an expense report form that would be filled out and sent to the user's manager.

Folder-based applications utilize *public folders* and can have electronic forms associated with them. Both public folders and the way in which electronic forms relate to them will be discussed in the Groupware section that follows. Chapter 8 will cover the topic of forms in detail.

Microsoft Outlook Forms Designer

The Outlook Forms Designer is a program that can create 32-bit forms. It contains the basic functionality of the Exchange Forms Designer, and adds some powerful new functions. One of those new functions is the ability to use Microsoft Office 97 applications to create templates for Outlook forms. Chapter 8 will cover the topic of the Microsoft Outlook Forms Designer.

Workflow

Microsoft Exchange Server 5.5 makes it possible to create simple workflow applications using the Microsoft Exchange Scripting Agent. This capability enables developers to create event- or time-based workflow applications using VBScript or JavaScript that run on the server. For example, a user could submit a purchase request via an electronic form that is sent to a designated public folder. Once the form reaches the public folder, a script could activate, which would examine the value in the cost field of the form. If the amount is less than $100, the request is sent to the purchasing department. If the amount is greater than $100, the request is sent to a manager for approval. Chapter 9 will examine creating workflow applications with public folders and the Microsoft Exchange Scripting Agent.

Groupware

A simple definition of *groupware* is any application (the *ware* in groupware) that allows *groups* to store and share information. That is a very broad definition, and one that includes applications like e-mail and electronic forms. And indeed, as you will see, they are important ingredients in groupware. But the emphasis in groupware is collaboration, not merely sending something—but enabling many people to collaborate.

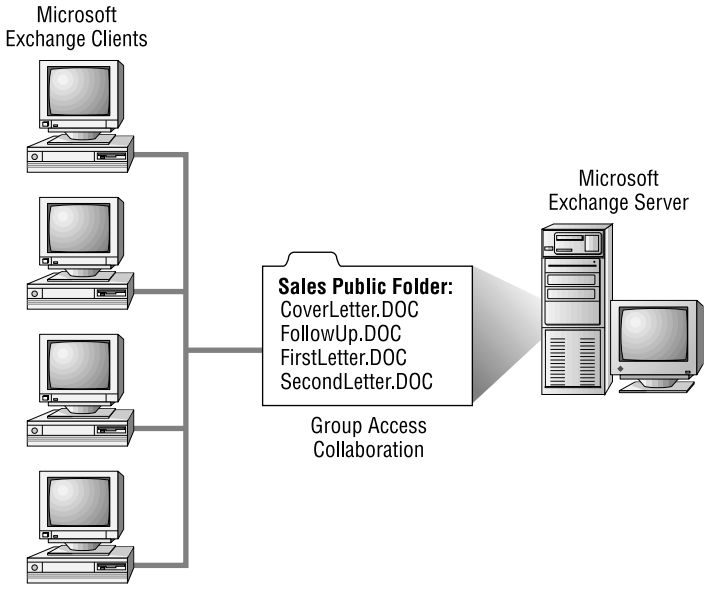
Microsoft Schedule+, which comes with the Exchange Server, is a straightforward groupware application. One of its functions is to enable many people to share calendar and scheduling information. A user, with the necessary permissions, can view the schedule of another user. This allows people to collaborate on their schedules. Microsoft Outlook also incorporates many groupware functions, such as the ability to share a calendar, schedule, task list, and contact list.

Another example of groupware is *folder-based* applications. These applications utilize public folders. A public folder is a special storage area for group access. Various types of information can be contained in a public folder, such as documents, spreadsheets, graphics, e-mail messages, forms, and many other types of information. Along with storing information, a public folder can be assigned security, so that only selected users or groups can access the public folder. Other features like views and rules can be assigned to a public folder. A simple folder-based application would be where a sales department places all their sales letters in a specified sales public folder. Only the employees in the sales department would be given permission to access this public folder.

Folder-based applications can also utilize electronic forms. A specific electronic form or forms can be associated with a public folder. Users can then fill

out and "post" the form to the public folder. Other users can access the public folder and view the posted information. An example of this type of application is a discussion-and-response application. A product manager could create a public folder for discussion about a product under development. That manager could also create customized electronic forms that people could use to enter their comments and then send, through e-mail, to the public folder. The product manager and product developers could then access the public folder to read the comments. It is even possible to set up customized views of the content in the public folder in order to view only data on a specific topic. A marketing person might want to see only comments relating to the possible market for the product. Folder-based applications are examples of *pull-style* communication, because users go to the information and decide what is relevant to them. See Chapter 9 for more on the topic of public folders. Figure 1.2 illustrates folder-based applications.

FIGURE 1.2
Folder-based applications



Another groupware element of Exchange is the Microsoft Exchange Chat Service. The Chat Service enables users to conduct online meetings, discussions, or collaborate on documents or projects online and in real time. The Chat Service is based on the Internet protocol Internet Relay Chat (IRC). Chapter 11 will discuss this service.

Other Messaging Applications

Along with e-mail, electronic forms, and groupware, there are many other types of messaging applications. Exchange provides an open platform that encourages the integration of other types of applications, some of which include:

- **Workflow** While Exchange includes some basic workflow capabilities, some third-party workflow solutions that work with Exchange are also available, such as Keyflow from Keyfile Corporation.
- **Fax** Fax software can be integrated with the Exchange clients so that e-mail and faxes can be sent and received from the same location, as well as share the same address book.
- **Paging** Some paging products can enable an Exchange client to send an electronic page through an Exchange server, to a wireless paging service, and to another person's pager.
- **Video conferencing** Products like Microsoft's NetMeeting can be integrated with the Exchange clients to provide video conferencing, as well as other features like *whiteboard conferencing* (i.e., collaborative drawing) and text-based chatting.
- **Voice mail** There are various voice mail products that integrate with Exchange and store their messages in an Exchange mailbox.

Client/Server Model

Microsoft Exchange uses a client/server computing model to implement its messaging system. To better understand the client/server model, two other models will be briefly discussed to provide a contrast to the client/server model. The three models discussed in this section are:

- Host-based computing
- LAN-based shared-file computing
- Client/server computing

Host-Based Computing

Host-based computing consists of a powerful host computer, such as a mainframe computer or minicomputer, and numerous input-output devices attached to the host, such as terminals, printers, and personal computers running terminal emulation software. The advantages of this architecture are the powerful, centralized computing processing, administration, and backup. These features permit a large number of users on these systems. The disadvantages are that these features incur high costs, that personal computing power and applications are not leveraged, and that most of these systems have a proprietary architecture. Examples of messaging systems that use this type of model are IBM PROFS (Professional Office System) and OfficeVision.

Figure 1.3 illustrates the host-based computing model.

LAN-Based Shared-File Computing

This network computing model works in a local area network (LAN) context. At least one powerful personal computer is used as a server computer to store files. Users, working on their own networked personal computers, access and share the files on the server computer. Microsoft Mail is a messaging system that uses this type of architecture.

Using this model, a shared-file mail system has *active clients* and *passive servers*. Each mail user is assigned a mailbox. A mailbox is actually a directory on the server where mail messages will be placed. The server software is passive in that its main task is to store mail messages.

The client software is said to be active because it performs almost all mail activities. Along with the normal mail activities of creating and reading mail, the client software is also responsible for sending mail to mailboxes and checking for new mail (this is referred to as *polling)*.

This could be compared to a postal system where people must take their outgoing mail to the post office and place it in the respective recipients' mail slots. They must also visit the post office to check their mail slots for any new mail. The primary duty of the post office is to store the mail. This is analogous to the shared-file mail system in that the clients are active and the server is passive.

The advantages of shared-file mail systems include the following:

- **Minimal server requirements** Because the server has a passive role, it does not need to run on a high-end hardware platform.
- **Minimal server configuration in a single-server environment** Because the server is mainly a storage location, it does not need a lot of configuration.

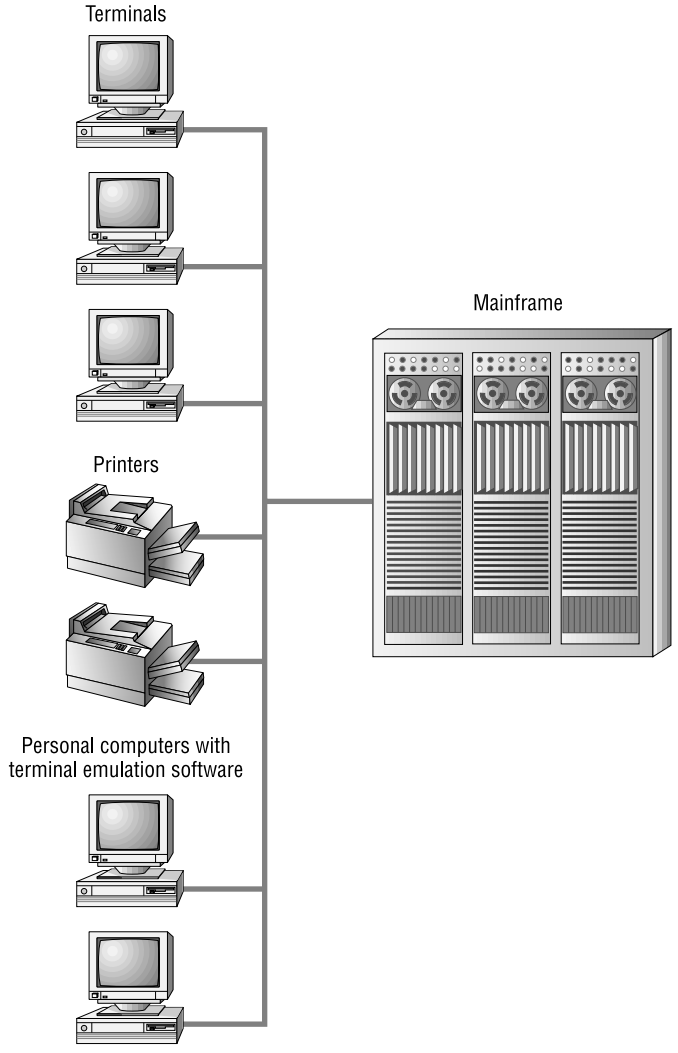


FIGURE 1.3
Host-based computing model

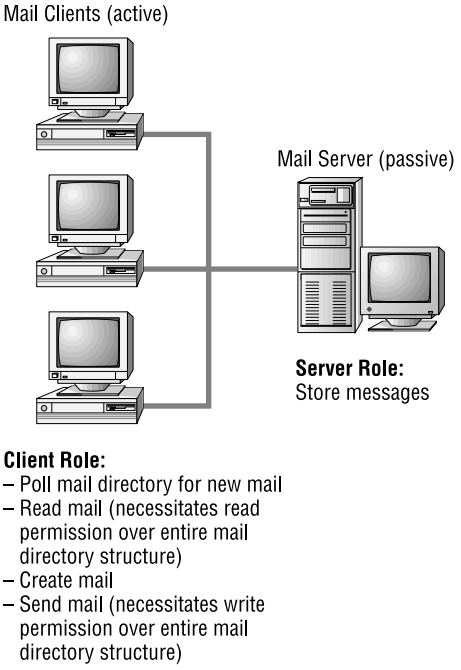
The disadvantages of shared-file mail systems include the following:

- **Limited security** Because the client software is responsible for sending mail to a recipient's mailbox, each client must have write permissions on each mail directory. Each client must also have read permissions on the entire mail directory structure in order to read forwarded or copied messages. From a security standpoint, this is considered an excessive level of permissions.

- **Increased network traffic** The periodic client polling of mailboxes for new mail increases network traffic.
- **Increased client load** The active clients do almost all of the processing work.
- **Limited scalability** These systems cannot accommodate large numbers of users due to the shared-file model. Users must access common files that can be opened by only one process at a time.

Figure 1.4 illustrates a shared-file mail system.

FIGURE 1.4
Shared-file mail system



Client/Server Computing

Client/server computing is where a computer task is divided between the client processes and server processes. Each side, while performing specific parts of the task, works to accomplish the task. The two processes are usually running on separate computers and are communicating over a network. The communication is in the form of requests and replies passed back and forth through messages.

The client side includes a user's personal computer or workstation and client software. The client software provides the interface to the user for manipulating data and making requests to and receiving replies from the server. The processing power to carry out those tasks is provided by the client's computer.

The server side includes the server computer and server software. The server software receives and processes client requests, provides storage capabilities, implements security, provides for administrative functions, and performs many more duties. The server's processor, or processors, power these functions.

When this model is applied to a mail system, both the client side and the server side are active participants. Mail activities are divided between the two sides in a way that takes advantage of both parties. The client software enables users to initiate mail activities like creating, sending, reading, storing, and forwarding mail and attachments.

The server software also has an active role. Some of its tasks are implementing security, placing messages in mailboxes (as opposed to the client software doing it), notifying clients of new mail (which eliminates clients polling their mailboxes), and performing specified actions on mail, such as applying rules, rerouting messages, along with many other tasks. Many of the mail activities that are initiated by the client software are actually implemented on the server. For example, when a client initiates the reading of a message, the client software sends a read request to the server where the message physically resides. The server software receives this request, processes it (for example, checks security to see if this user is permitted to read this message), and then sends the message to the client. The user can then use the client software and processor to manipulate the message (edit the message, for example). This illustrates how both sides are active.

In this model, the software running on the client machine is frequently referred to as the *front-end* program, while the software running on the server is referred to as the *back-end* program.

The advantages of the client/server model include the following:

- **Distributed computer processing** The computer processing power of both the client and server machines are utilized. The client processor handles the end-user mail activities, such as creating, reading, and manipulating mail, while the server processor (or processors) handles the security, routing, and special handling of mail. This spreads the processing load over a multitude of client processors, while still utilizing the powerful processing of the server machine.

The Big Ox Fallacy

Grace Hopper, one of the developers of the COBOL programming language, once told a story about a farmer who came upon a huge tree that had fallen across his road. The farmer brought out his ox and some rope and tried in vain to move the tree. The farmer's next act was to take the ox and breed it with another big ox hoping to produce an even bigger ox. He figured he would do this until he had bred an ox big enough to move that huge tree. Grace Hopper's commentary was that obviously the farmer's logic was flawed. What the farmer should have done was simply go get all his neighbors' oxen and hook them up to the tree and pull the tree off the road. This story illustrates the need to utilize the distributed processing power of all the machines on a network.

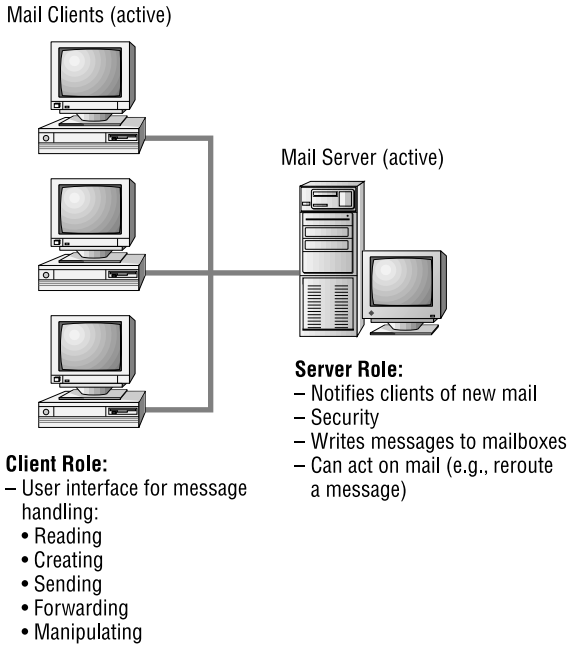
- **Tight security** The server software is responsible for the security of the mail system. The server software is the entity that actually places messages in mailboxes. The clients therefore do not need permissions to all mailboxes. This creates a much more secure mail system.
- **Reduced network traffic** Because the server software informs clients of new mail, the client software does not have to poll the server, thus reducing network traffic.
- **Scalable** The term *scalable* relates to the ability to grow easily. A client/server mail system can scale to any size organization.

The primary disadvantage of the client/server model is the following:

- **Increased server hardware requirements** Because the server has an active role in the messaging environment, there are greater requirements for the server hardware platform. This should not be seen as much of a disadvantage in light of the advantages of scalability, central administration, backup, and other advantages.

Figure 1.5 illustrates the client/server mail system.

FIGURE 1.5
Client/server mail system



Exchange is a client/server messaging system. The Exchange Server software runs as an application on a Windows NT Server. It provides server-side messaging functions for the client applications. Exchange also ships with the client applications noted earlier in this chapter. These programs, along with third-party applications like Web browsers, provide the client-side functions such as making requests to the server and creating and manipulating data.

So far we have learned *what* features make up Exchange 5, and *how* the system is implemented, namely the client/server model. Now we need to turn our attention to the context or scale in which Exchange can be implemented.

Enterprise-Wide Computing

Microsoft Exchange was designed to be an *enterprise* messaging system, meaning one that spans an entire organization. While an enterprise can be of small or medium size with straightforward requirements, it can also be very large and have very complex requirements. For Exchange to be an

enterprise messaging system, a large number of technologies had to be included or leveraged from other products (such as Microsoft Windows NT Server). This section briefly discusses the technologies that make Exchange a true enterprise messaging system. Those technologies fall into six categories:

- Enterprise-quality application platform
- Scalability
- Interoperability
- Performance
- Administration
- Reliability

Enterprise-Quality Application Platform

Before a determination can be made as to whether or not a product can scale to the size an organization needs, it must be determined that the product can do the things it needs to do. Exchange provides the necessary application platform to meet the requirements of almost any organization. The following are some of the elements of the Exchange application platform:

- **Supports large number of messaging services** E-mail, electronic forms, groupware, and add-on products for fax, paging, video conferencing, voice mail, and many services are supported.
- **Supports large number of client platforms** There is client software that runs on MS-DOS, Windows 3.*x*, Windows 95, Windows NT Workstation, Apple Macintosh, UNIX, and IBM OS/2.
- **Integrates with other client applications** The Exchange client programs that ship with Exchange have very tight integration with the most popular application suite on the market, Microsoft Office.
- **Provides open architecture/extensibility** Exchange is based on an open architecture, meaning that the specifications of many of its protocols are available in the public domain. Examples of published protocols include the Messaging Application Programming Interface (MAPI), Internet protocols, and various CCITT protocols. Developers can use this openness to create additional applications and programs that work with or extend Exchange. That is what is meant by *extensible*. One example of the way Microsoft encourages this is by including a single-user version of the

Microsoft Visual InterDev product with Exchange Server. Developers can use Visual InterDev to create Web-based applications that enable Web clients to access Exchange resources. Chapter 11 will briefly cover this topic.

- **Based on industry standards** The Exchange protocols, along with being open and extensible, are based on industry standards (protocols can be open and extensible but not industry standards). The MAPI protocol is considered an industry standard. Some of the industry standard Internet and CCITT protocols used in Exchange are:

  Internet mail Simple Mail Transfer Protocol (SMTP), Post Office Protocol 3 (POP3), and Internet Message Access Protocol version 4 (IMAP4). See Chapter 11.

  Internet Chat Internet Relay Chat (IRC). See Chapter 11.

  Internet directory access Lightweight Directory Access Protocol (LDAP). See Chapter 11.

  Internet news services Network News Transfer Protocol (NNTP). See Chapter 11.

  Internet management Simple Network Management Protocol (SNMP). See Chapter 12.

  Internet security Secure MIME (S/MIME), Secure Sockets Layer version 3 (SSL), and Simple Authentication and Security Layer (SASL).

  Internet Web protocols HyperText Transfer Protocol (HTTP) and HyperText Markup Language (HTML). See Chapter 11.

  CCITT message transfer (International Telegraph and Telephone Consultative Committee): X.400. See the section "Industry Standards" later in this chapter.

  CCITT directory X.500. See the section "Industry Standards" later in this chapter.

- **Security features** Using the Internet security protocols listed above, along with other protocols, Exchange can provide advanced security features. For example, messages can be sent with a digital signature to confirm the identity of the sender, and message content can be encrypted to prevent unauthorized viewing. Chapters 10 and 11 discuss the protocols and administration of advanced security in Exchange. Further security

features, and ones that are leveraged from Microsoft Windows NT Server, include:

Mandatory logon A user must have a domain account and password to log on to a Windows NT Server domain.

Discretionary access control An Exchange administrator can use NT security to control access to Exchange resources. For example, one administrator could have security to manage one particular Exchange server, but not other servers.

Auditing Windows NT can be configured to monitor and record certain events. This can help diagnose security events. The audit information is written to the Windows NT Event Log.

Scalability

Once a product has been determined to accomplish the types of things you need to get done, then you must find out if it can do it on the scale you need. Exchange is extremely scalable due to the following features:

- **Software scalable** Exchange can be implemented with a single Exchange server, or dozens of servers, depending on the message requirements. Even with multiple Exchange servers, a single enterprise messaging system exists. This is due to the Exchange features that enable communication between servers. Some of those features include message routing, directory replication, and data replication. This functionality permits Exchange to scale from a single server to multiple server implementations. Microsoft itself uses Exchange for its worldwide messaging system.
- **Hardware scalable** Scalability is also evidenced by the maximum hardware specifications that Exchange can utilize.
  - **CPUs** Scalable from 1–8 processors.
  - **RAM** Maximum is 4 gigabytes.
- **Disk Storage** Storage is limited only by hardware capacity.The Standard edition of Microsoft Exchange Server 5.5 has a 16GB storage limit on each of the Exchange databases. The Enterprise edition has a 16TB (i.e., terabyte; trillion bytes) limit on each of those databases.

Interoperability

For a product to fit into an enterprise, it might need to work with an existing messaging system. This is called *interoperability* or coexistence. An organization might need to move all its existing messaging data to a new messaging product. This is called a *migration*. Exchange addresses both of these issues.

To interoperate with various non-Exchange systems, referred to as *foreign systems*, Microsoft had to write special software programs called *connectors*. Connectors are similar to translators who understand both Exchange and the foreign system and translate between them. Third-party companies also have written similar programs. Microsoft refers to these programs as *gateways*. Some of the messaging systems that Exchange can interoperate with include:

- Internet mail
- X.400 mail systems
- Microsoft Mail
- Lotus cc:Mail
- Lotus Notes
- IBM PROFS and OfficeVision
- Digital Equipment Corporation (DEC) All-IN-1
- Verimation MEMO

The Standard edition of Exchange Server 5.5 ships with connectors for Internet mail, Microsoft Mail, Lotus cc:Mail, and Lotus Notes. The Enterprise edition adds to that list connectors for X.400 systems, and IBM OfficeVision/PROFS and IBM SNADS–based systems. Connectivity to other systems, such as DEC ALL-IN-1 or Verimation MEMO, is provided through third-party gateway products.

For Exchange to interoperate with some of the previous systems, third-party software is required. Chapters 14 and 15 discuss interoperability in more detail.

Some of the messaging systems that Exchange can perform a migration with include:

- Microsoft Mail
- Lotus cc:Mail
- Novell GroupWise
- Netscape Collabra
- IBM PROFS and OfficeVision
- DEC All-IN-1
- Verimation MEMO

For Exchange to perform a migration with some of the previous systems, third-party software is required. Migration is discussed in Chapter 16.

Performance

A messaging system requires adequate performance to be used on an enterprise scale. Exchange meets that requirement by being a 32-bit, multithreaded program running on a high-performance operating system, Microsoft Windows NT Server. Exchange also ships with a program to help optimize performance, called the Exchange Optimizer. Another Exchange program that relates to performance is Load Simulator. This program can simulate load on the system and predict the system's performance.

Administration

An important element of any enterprise application is the ability to effectively and efficiently administer it. Exchange meets this need by including powerful administration programs, one of which is the Exchange Administrator program. This program provides a single point of administration for an entire Exchange organization. Exchange servers anywhere in the enterprise can be managed from this program, and such activities as creating mailboxes, configuring a server, and managing connections to foreign systems, can all be managed from this program. The Exchange Administrator program also enables

the creation of mailboxes in a batch process, and provides for the tracking of messages. The Exchange Administrator program includes these administrative utilities:

- **Link Monitor** Examines the link between two servers.
- **Server Monitor** Examines the status of a particular server.

Exchange Server 5.5 has also added support for the Simple Network Management Protocol (SNMP). This enables third-party SNMP monitor programs to collect various management information about an Exchange server, such as performance information collected by NT Performance Monitor. The topic of Exchange Server administration is covered in Chapter 12.

Along with its own administrative utilities, Exchange can leverage the administrative capabilities of the Windows NT Server operating system. Exchange integrates with Windows NT Server utilities like Performance Monitor and Event Viewer.

Reliability

Because of the importance of a messaging system to an enterprise, it must be reliable. Exchange provides reliability through the following:

- **Transaction log files** Data that is to be written to an Exchange database is first written to these log files (which can be done very fast). The data is later written to the appropriate database, which takes longer because of the structured nature of a database. If for whatever reason, a server has an unintended shutdown, data that has not been written to the database is not lost; it can be automatically reconstructed from the transaction log files. Chapter 12 will discuss this topic further.
- **Windows NT Backup program** When Exchange is installed, it adds extensions to the Windows NT Backup program, allowing that program to backup Exchange information.
- **Replicas** Exchange can be configured to have multiple copies, called *replicas*, of a single public folder on different servers. This prevents a single point of failure in terms of data access.
- **Intelligent message routing** This feature allows multiple routes to a destination, thereby preventing a single point of failure for message delivery.

- **Windows NT Server fault tolerance** Exchange takes advantage of the many fault tolerant features of the Windows NT Server operating system, such as disk mirroring and disk striping with parity. The Enterprise Edition of Exchange Server 5.5 supports Microsoft's Cluster Server product. Cluster Server provides fault tolerance in the event of an NT server malfunction. If a server fails, another server can take its place, thereby providing uninterrupted service to users.

Industry Standards

Microsoft Exchange is based on industry standard technologies. Someone once joked that the best thing about standards was that there were so many of them from which to choose. But being a standards-based product ensures an open architecture and therefore extensibility (i.e., the ability to easily add on to the product). An adequate understanding of the standards used in Exchange will help in utilizing it. This section presents a brief coverage of the following standards:

- Messaging Application Programming Interface (MAPI)
- The Remote Procedure Call (RPC) protocol
- CCITT X.400
- CCITT X.500

The Internet standards are also very important in Exchange, and they will be discussed in Chapters 11 and 14.

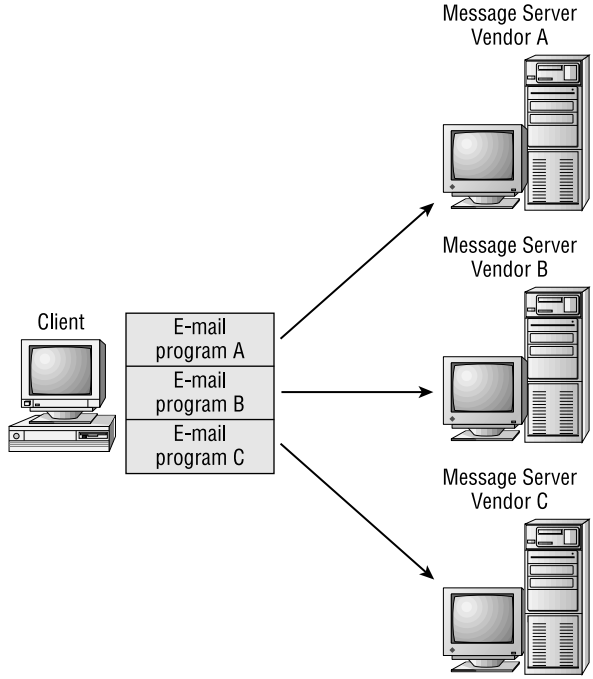
Messaging Application Programming Interface (MAPI)

To understand MAPI, you must first understand what an application programming interface is. At the code level, a program's functions are invoked through specific instructions. The collection of those instructions are referred to as an application programming interface (API). That phrase is appropriate because the API allows a programmer to interface with the functions of a program. For

example, if a program has the ability to read a message, there is a specific API instruction, also called a *function call*, that can invoke that ability. If two programs need to interact, they must do so with an API they both understand. For example, if program A sends the instruction `Read_Message 4` to program B, but program B only understands the instruction `Message_4_Read` then the instruction will not be understood. Humans can use slightly different grammar and still understand one another, but computers are not that forgiving.

In the past, many client/server messaging products had their own APIs for the client/server interaction. If someone wrote a client program, it would only work with the messaging system whose API it used. If a user needed to connect to multiple messaging systems, multiple client programs were needed (see Figure 1.6).

FIGURE 1.6
Multiple messaging APIs require multiple programs



Microsoft decided to remedy that situation by creating a standard messaging architecture, referred to as the Messaging API (MAPI). MAPI accomplishes two broad goals. One, it provides a standard API for client/server

messaging interaction. This role makes MAPI a type of *middleware*, meaning that it stands in the middle between clients and servers. Some authors refer to middleware as the slash (/) between the words client and server. MAPI makes it possible for a single client application to access different messaging servers (see Figure 1.7 for an illustration).

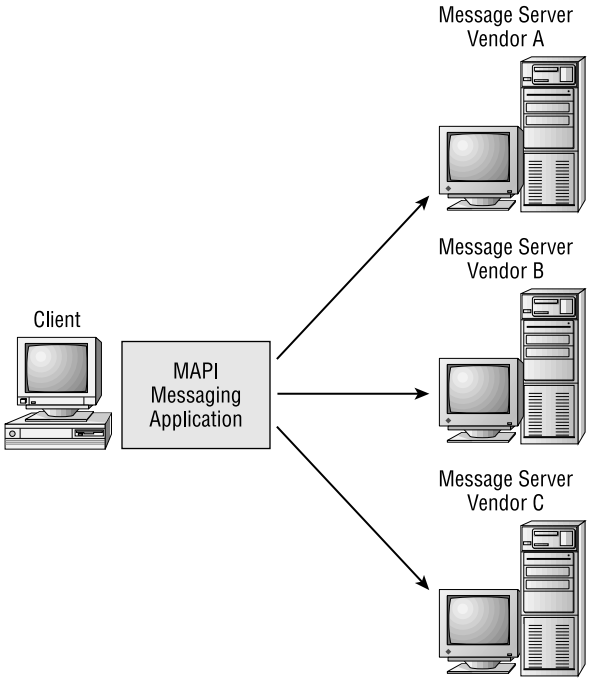


FIGURE 1.7
Accessing different messaging servers through MAPI

The second broad goal of MAPI is to provide a standard set of services to client messaging applications. These services include address books, message storage, and transport mechanisms. Even when using different types of MAPI applications, like e-mail, fax, and voice mail, a user can access a single address book (universal address book) and store different data types in the same folder (universal inbox). The transport mechanisms relate to a single client application connecting to different messaging systems. A single MAPI e-mail application can access a Exchange server, a Microsoft Mail post office, an Internet mail server, and others. Chapter 6 examines the MAPI services in more detail.

Although MAPI includes individual API instructions, it most often communicates those instructions in an object-oriented manner. An object is a container: in this context, it functions as a container of API instructions. The Microsoft specification for object-oriented programming is called the Component Object Model (COM). MAPI, OLE, ActiveX, and other technologies are part of the COM standard.

The original version of MAPI (called Simple MAPI) was developed by Microsoft. But in the subsequent version (MAPI 1.0), Microsoft worked with over 100 different vendors to develop an industry standard. Microsoft has also turned over the vast majority of the MAPI specification to standards organizations, while still taking a leadership role by including the core MAPI component with the Microsoft Windows 95 and Windows NT operating systems.

While MAPI deals with instructions, the next section discusses the protocols that enable those instructions to be passed between clients and servers.

Procedure Calls

We now know the instruction standard used by the Exchange client/server messaging applications, namely MAPI. But client/server applications are divided across physical machines. When a client issues a read instruction for a message, that message could be on the server. The server could understand that instruction and could send the message, but the instruction has to get to the server and the message has to get back to the client. MAPI does not handle those procedures. From MAPI's perspective, the physical distinction of the client and the server does not exist, it is transparent. Microsoft uses the Remote Procedure Call protocol (RPC) to pass instructions and data between machines. Before discussing the RPC protocol, we will first define what a procedure call is, and discuss the two types of procedure calls, local and remote.

Procedure calls handle the transfer of instructions and data between a program and processor, or processors. When a program issues an instruction, that instruction is passed to the processor for execution, and the results of the execution are passed back to the program. There are two main types of procedure calls:

- Local procedure calls
- Remote procedure calls

Local Procedure Calls

When a program issues an instruction that is executed on the same computer as the program executing the instruction, the procedure is referred to as a *local procedure call*. When Exchange Server components perform activities on that server, they issue instructions that are executed by that server's CPU or CPUs. That is an example of a local procedure call. Exchange uses a Microsoft protocol called the Local Procedure Call (LPC) to implement this mechanism. Examples of how Exchange uses LPCs will be forthcoming in Chapter 2.

Remote Procedure Calls

A *remote procedure call* is similar to a local procedure call in that it relates to the transfer of instructions and data between a program and processor. But unlike a local procedure call, a remote procedure call enables an instruction issued on one computer to be sent over the network to another computer for execution, with the results being sent back to the first computer. The computer making the instruction and the computer performing the execution are remote from each other. The transfer of instructions and data between the computers is totally transparent to the original program and to the user. To the program issuing the instruction, all its instructions appear to be locally executed. Remote procedure calls are a key ingredient in distributed processing and client/server computing.

The RPC mechanism permits the optimizing of different computers for specific tasks. For example, some programs require lots of processor power, memory, storage, or all three. It would be impractical to give every computer running these applications those level of resources. But one specialized computer could be given, for example, 4 processors, 512MB of RAM, and 16GB of storage. Clients could use those resources through the RPC mechanism.

Because the request/reply aspect of RPC is intended to be transparent to the client program and user, the speed of network communication is a factor. The computers involved in a RPC session need to have a high-speed permanent link between them, such as a local area network (LAN) or a high-speed wide area network (WAN).

Exchange uses remote procedure calls in many of its communications. The protocol that Exchange uses to implement remote procedure calls is also called Remote Procedure Call (RPC). This protocol is discussed in the following section.

Remote Procedure Call (RPC) Protocol

As previously stated, the protocol that Exchange uses to implement remote procedure calls is also called Remote Procedure Call (RPC). It is based on a protocol created by the standards group Open Software Foundation (OSF) and is part of the OSF's Distributed Computing Environment (DCE) protocol suite. Microsoft includes the RPC protocol with their Windows NT operating system, and this is what Exchange uses for much of its communication.

When a user chooses to read a message, the Exchange client program issues a MAPI instruction (MAPIReadMail). The RPC protocol on the client transfers this instruction to the Exchange server where the message physically resides. This is called a *request*. The RPC protocol on the server receives this request, has it executed, and sends the message back to the client's screen. The is called a *reply*. RPC clients make requests, and RPC servers make replies. RPC is sometimes referred to as a request/reply protocol. RPCs are also used in some Exchange server-to-server communications. Figure 1.8 illustrates the RPC mechanism.

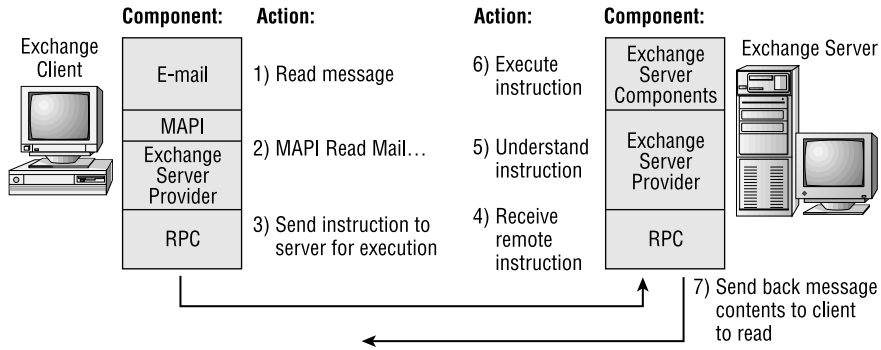


FIGURE 1.8
The Remote Procedure Call protocol

CCITT X.400

For most of the history of electronic messaging in the private sector, there were no widely accepted messaging standards. Different messaging products used vastly different messaging protocols. This prevented, or made difficult and costly, interoperability between different systems. To address this situation, different standards organizations began to develop what they

hoped would become internationally recognized messaging standards. One of those standards organizations was the Comité Consultatif International Telegraphique et Telephonique (CCITT). This is translated in English as the International Telegraph and Telephone Consultative Committee. One of the standards they developed was the X.400 Message Handling System (MHS) standard. Exchange uses some of the technologies of the X.400 standard.

The CCITT is now a subdelegation of the International Telegraph Union (ITU), which is an agency of the United Nations. The State Department is the voting member from the United States.

The different versions of the X.400 standard are referred to by the year they were officially published and by a specified color. Versions to date are:

- 1984 "Red Book"
- 1988 "Blue Book"
- 1992 "White Book"

The Message Handling System (MHS) discussed in this section is not the same standard as the Novell-related Message Handling System (MHS).

X.400 is a set of standards relating to the exchange of electronic messages (messages can be e-mail, fax, voice mail, telex, etc.). The goal of X.400 is to enable the creation of a global electronic messaging network. Just as you can make a telephone call from almost anywhere in the world to almost anywhere in the world, X.400 hopes to make that a reality for electronic messaging. X.400 only defines application-level protocols and relies on other standards for the physical transportation of data (e.g., X.25, and others).

X.400 Addressing: Originator/Recipient Address

Try to imagine the American telephone system if different parts of the country used different numbering schemes: different number lengths, different placement of the area code, etc. Obviously that would lead to a lot of complexity

and problems: hence a standard numbering scheme exists. Electronic messaging also needs a standard addressing scheme to avoid the same sort of chaos.

One might think that you could simply list people's names in alphabetical order. But there are many problems with that scheme. The addressing scheme needs to potentially scale to the entire world's population. An alphabetical list would be quite long. There is also the problem of what constitutes a last name, different countries have different methods (e.g., Anwar el-Sadat, Willem de Kooning). A truly global address scheme needs to be totally unambiguous.

The address scheme that X.400 uses is called the Originator/Recipient Address (O/R Address). It is similar to a postal address in that it uses a hierarchical format. While a postal address hierarchy is country, zip code, state, city, street, and recipient's name, the O/R address hierarchy consists of countries, communication providers (like AT&T), companies or organizations, and other categories. Figure 1.9 and Table 1.1 present some of these categories, called *fields*.

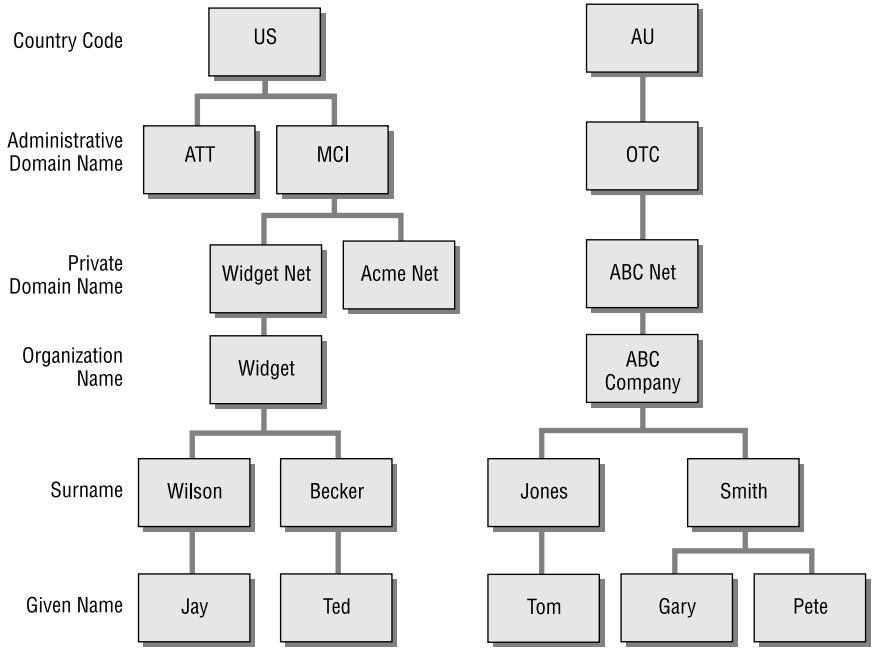


FIGURE 1.9
X.400 Originator/ Recipient address example

TABLE 1.1 Some X.400 Originator/Recipient address fields

Field	Abbreviation/ Example	Description
Country code	c=us	Country
Administrative Management Domain (ADMD)	a=mci	The third-party networking system used (e.g., ATT, MCI, Sprint, etc.)
Private Management Domain (PRMD)	p=WidgetNet	Subscriber to the ADMD (company name)
Organization	o=Widget	Name of company or organization
Surname	s=Wilson	Last name
Given name	g=Jay	First name

The O/R Address specifies an unambiguous path to where the recipient is located in the X.400 network (it does not specify a path the message might take, only the path to where the recipient is located).

In actual practice, this addressing scheme is not as standardized as Table 1.1 seems to indicate, nor is it used in the standardized way. Although the address fields have always been specified, the order in which to write them was not specified until 1993. Consequently, you will see them written in different ways. Some X.400 implementations have modified the standard.

X.400 Message Format: Interpersonal Messaging (IPM)

X.400 also specifies the protocols for formatting messages. The most common one is called Interpersonal Messaging (IPM), and is used for e-mail messages. There are other protocols for other types of messaging, such as electronic data interchange (EDI).

X.400 Message Routing: Message Transfer Agent (MTA)

Another very important X.400 protocol is Message Transfer Agent (MTA). MTA is the protocol that runs in the message routing machines (i.e., routers). MTA is like a local post office, in that it receives and routes

messages to their ultimate destinations. And just like a postal system (a snail mail system), electronic messages can go through several MTAs before they arrive at their ultimate destinations. This type of delivery method is called *store and forward*. An MTA machine receives a message, stores it so it can calculate its next route, and then forwards it either to another MTA machine or its ultimate destination. This method eliminates the need for the sender's application and the recipient's application to perform any simultaneous actions in order to exchange data. A sender's message is simply packaged with all the necessary addressing information, and is sent to the next store-and-forward MTA machine (i.e., router). That MTA can route it to the next MTA, and so on, until it reaches it final destination.

Other X.400 Information

While the X.400 standard does not define the protocols for the physical transporting of messages, it does specify what other standards it can use. They include the following OSI (Open Systems Interconnection) protocols:

- TP0/X.25
- TP4 (CLNP)
- TP0/RPC 1006 to TCPIP

TP stands for Transport Protocol.

Third-party X.400 networks that can be subscribed to include: AT&T Mail, AT&T EasyLink, MCI Mail, Sprintmail, Atlas 400 (France), Envoy 100 (Canada), Telebox 400 (Germany), and Telecom Australia. Microsoft Exchange is an X.400 messaging product.

CCITT X.500

The CCITT X.500 standard defines the protocols for a global directory service. A directory service is a database of information about resources. Resources can be user accounts, user groups, mailboxes, printers, fax machines, and many other items. These resources are officially referred to as objects. The information about an object, such as a mailbox, can include the owner of the mailbox, the owner's title, phone number, fax number, and many other types of information. The information about an object is referred to as its *properties* or

attributes. A directory enables objects and their properties to be made available to users and administrators.

The directory's importance cannot be overstated. To use a telephone analogy, imagine the current global telephone system without telephone directories. The technology to make a call would be in place, but you would have a hard time locating a person's number to call. The creation of a global, electronic yellow pages could go a long way toward solving the "I know it's out there, I just can't find it" problem.

To create a directory service, X.500 addresses two main areas:

- **Directory structure** How resources should be organized.
- **Directory access** How one is able to read, query, and modify a directory.

X.500 Directory Structure

The X.500 directory structure is hierarchical, which facilitates a logical organization of information, which leads to finding information easier. Figure 1.10 illustrates the X.500 directory structure, and Table 1.2 explains it.

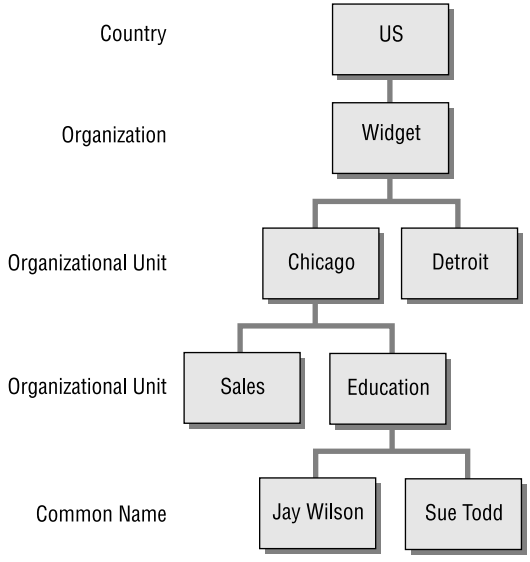


FIGURE 1.10
X.500 directory structure example

TABLE 1.2 Descriptions of X.500 objects

X.500 Object	Abbreviation/Example	Description
Country	c=us	Country of the organization
Organization	o=Widget	Name of the organization
Organization Unit	ou=Chicago ou=Detroit	Subcategory of the organization
Organization Unit	ou=Sales ou=Education	Subcategories under the ou=Chicago
Common Name	cn=JayWilson	Name of a specific resource (username, fax name, printer name, etc.)

The X.500 terminology for the structure of a directory is the Directory Information Tree (DIT). The term for the information in the directory is Directory Information Base (DIB).

To communicate the location of an object in the directory hierarchy, list the path to that object, starting at the top and moving down. This is called a Distinguished Name (DN). The DN of the example in Figure 1.10 is:

c=us; o=widget; ou=Chicago; ou=Education; cn=jaywilson

The differences between an X.500 address, the distinguished name (DN), and an X.400 address are due to their different purposes. A DN is the location of an object in the directory, whereas the X.400 address is the location of an object in a message system. Getting back to the telephone analogy, a DN is the location of a person in the phone book, and an X.400 address is where they are in the physical telephone system. This is illustrated by the fact that an X.400 address can include information about third-party messaging networks used to physically deliver a message, some examples being AT&T, MCI, and Sprint.

The 1988 release of X.400 incorporated the use of a DN address instead of, or along with, an O/R address. Some implementations of X.400 also incorporated some of the X.500 fields, like ou= and cn=.

A directory also puts a more natural interface to network resources. Many communication objects have long numeric identifiers that are hard to remember. A directory allows objects to be presented to users by a natural descriptive term. The directory then maps the descriptive term to the numeric.

X.500 has a 1988 version and a 1993 version.

Directory Access

Having a directory is only half the equation. Users and administrators must also be able to access it to read, query, and write to it. A user might query the directory for a printer in the sales department, fourth floor, and the directory could respond with the needed information about the printer. Other issues that must be addressed are security (e.g., who can access an object and modify its properties) and directory replication (a true global directory would need to be on more than one machine). These issues are addressed by directory access protocols.

The standard access protocol in the X.500 recommendations is the Directory Access Protocol (DAP). DAP is considered more of a model than a real-world protocol. This is because DAP is very computer-resource intense (i.e., heavy) on client machines, and the few implementations of it were proprietary. But a newer access protocol that is getting a lot of attention today is the Lightweight Directory Access Protocol (LDAP). LDAP is an Internet protocol derived from the X.500 DAP. One of the reasons LDAP is called lightweight is because it requires fewer computer resources on the client. While LDAP is an Internet protocol, it is designed to enable access to an X.500-type directory. Almost every major software vendor has pledged support for LDAP.

Summary

The Exchange product is a powerful client/server enterprise messaging product.

The types of applications in an Exchange environment are:

- Electronic mail (e-mail)
- Electronic forms
- Workflow
- Groupware
- Other applications, such as fax, paging, video conferencing, and voice mail

The client applications that are shipped with Exchange are:

- Exchange Client
- Exchange Forms Designer
- Schedule+ 7.5
- Outlook
- Outlook Forms Designer

The network computing model that Exchange uses to implement its messaging system is the client/server model. This model utilizes the computing power of both client computers and server computers.

Exchange was designed for enterprise-wide implementations and consequently meets the following requirements:

- Enterprise-quality application services
- Scalability
- Interoperability
- Performance
- Administration
- Reliability

The following industry standards are used by Exchange:

- Messaging Application Programming Interface (MAPI)
- Internet protocols (e.g., POP3, IMAP4, LDAP, SNMP, and others)
- Remote Procedure Call (RPC)
- X.400
- X.500

Review Questions

1. The type of utility that would allow you to transfer data from a foreign message system to Exchange is:

   A. Administrative tool

   B. Security utility

   C. X.500

   D. Migration tool

2. This type of mail system has a passive server:

   A. Shared-file mail system

   B. Client/server mail system

   C. Host-based mail system

   D. Exchange server

3. Which of the following is NOT a feature of a client/server messaging system?

   A. Distributed processing

   B. Tight security

   C. Passive client application

   D. Reduced network traffic

4. Microsoft Outlook is a desktop information manager application that runs on the following operating system:

   A. Windows 3.*x*

   B. Windows for Workgroups

   C. Windows NT

   D. 32-bit UNIX

5. The following types of components can be used to connect Exchange to a foreign messaging system:

   A. Gateway

   B. Connector

   C. Groupware

   D. Key Management

6. The universal inbox is part of this software:

   A. Exchange Client

   B. SMTP clients

   C. POP3 clients

   D. LDAP clients

7. The following is an e-mail protocol:

   A. LDAP

   B. PNP

   C. PPP

   D. POP3

8. Which of the following could NOT be used as an Exchange e-mail client?

   A. Word processing program

   B. Web browser

   C. Internet e-mail program

   D. Microsoft Outlook

9. Which Exchange Client feature enables e-mail content to include multiple format types, such as fonts, sizes, and colors?

   A. WordPerfect

   B. This software cannot do this

   C. Richman message content

   D. Rich-text message content

10. This Microsoft technology enables the Exchange Client to create compound documents using drag-and-drop:

    A. OLE

    B. OOP

    C. DLL

    D. SNADS

11. Recalling a sent, but unopened, message is a feature of what program or programs?

    A. Outlook

    B. Exchange Client

    C. Public folders

    D. Forms

12. Which of the following relate to the performance of Exchange?

A. Load Simulator

B. 32-bit program

C. Multithreaded

D. Auditing

13. Since Exchange is scalable, it cannot be used in small to medium environments.

A. True

B. False

14. Exchange could use the Internet as part of its WAN design.

A. True

B. False

15. The following feature enables a Exchange client to create rules for processing incoming mail:

A. Search tool

B. Customized views

C. Universal inbox

D. AutoAssistants

16. What is the name of the mechanism when two Exchange components on the same machine pass instructions and data?

A. Remote procedure call

B. Remote instruction call

C. Local instruction call

D. Local procedure call

17. What is the name of the mechanism when a client-read instruction is sent to an Exchange server for execution?

    A. Local procedure call

    B. Local instruction call

    C. Remote instruction call

    D. Remote procedure call

18. Exchange Server 5.5 supports what newer Internet protocol that enables the retrieval of e-mail messages from a remote server?

    A. SNMP

    B. IMAP

    C. POP

    D. SSL

19. The addition of the Microsoft Exchange Scripting Agent enables the adding of event or time-based features to a public folder thereby creating what type of application?

    A. Chat

    B. Remote mail

    C. Security

    D. Workflow

20. Which of the following is an Internet management protocol that has been added to Exchange Server 5.5?

    A. SNMP

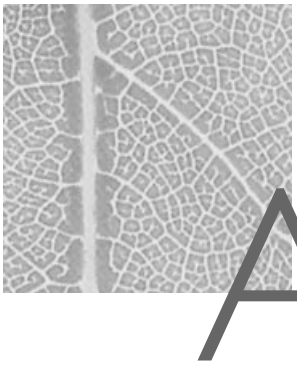
    B. SMTP

    C. MMX

    D. MID

2

Microsoft Exchange Architecture



An *architecture* is a description of the structure of something. A structure can be something physical, like a building or bridge, or something abstract, like the principles of a philosophy. When applied to a software product, an architecture is a description of the software components of the product, what they are, what they do, and how they relate to each other. Part of what software components do is create and manage *objects* (i.e., resources) like servers, mailboxes, public folders, address books, etc. How those objects are structured or organized is also part of a software architecture.

There are many practical benefits to understanding the architecture of Microsoft Exchange. It will aid a person in designing, installing, administering, and troubleshooting an Exchange system. For example, understanding component functionality will assist in deciding what optional components, if any, to choose during an installation. Troubleshooting can frequently benefit from a good understanding of an architecture; just understanding some error messages requires this knowledge. This chapter also serves as a good conceptual background for the topics in the remainder of the book.

In this chapter, we will address the following issues:

- The organization of Microsoft Exchange objects
- Core components of Microsoft Exchange
- Optional components of Microsoft Exchange

Microsoft Exchange Object Organization

Many of the Microsoft Exchange components create and manage objects. Objects, as stated earlier, are resources such as servers, mailboxes, public folders, and address books. The organization of these objects is part of

the architecture of Microsoft Exchange. Finding an object in the Exchange directory requires understanding object organization and object management. As you will learn, if a number of objects need the same configuration, and those objects are all grouped under a single object, the configuration can be given to the single object and then inherited by the objects underneath it.

Exchange is organized hierarchically. A *hierarchy* is the grouping and arrangement of items by rank, order, class, etc. A family tree is a good example of a hierarchy. At the top of the tree are the family patriarch and matriarch. Below them are their children, and their children below them, and so on. The family tree shows the relationships between people. One of these relationships is the parent-child relationship. Inherent to that relationship is the passing of traits from parent to children. A person can be both a parent and a child. A person will always be a child in reference to their parents, and that person can have children, making them also a parent. All of these concepts will relate to Microsoft Exchange.

Three objects constitute the main structures in the Exchange hierarchy:

- Organization
- Sites
- Servers

These objects contain other objects, but these three form the major structure of the hierarchy. Later in this chapter, under "Core Components" and "Directory Service," additional information will be given on them, as well as the other objects contained in them. Figure 2.1 illustrates the three main structures.

Organization

The largest unit, and the one at the top of the Exchange hierarchy, is the organization. All other Exchange structures are contained under this unit. The organization is the parent to the sites. This relationship permits an administrator to configure something at the organization level and have it apply or be available at all the sites (its children).

Because an organization encompasses an entire Exchange system, each company or business should create only one organization.

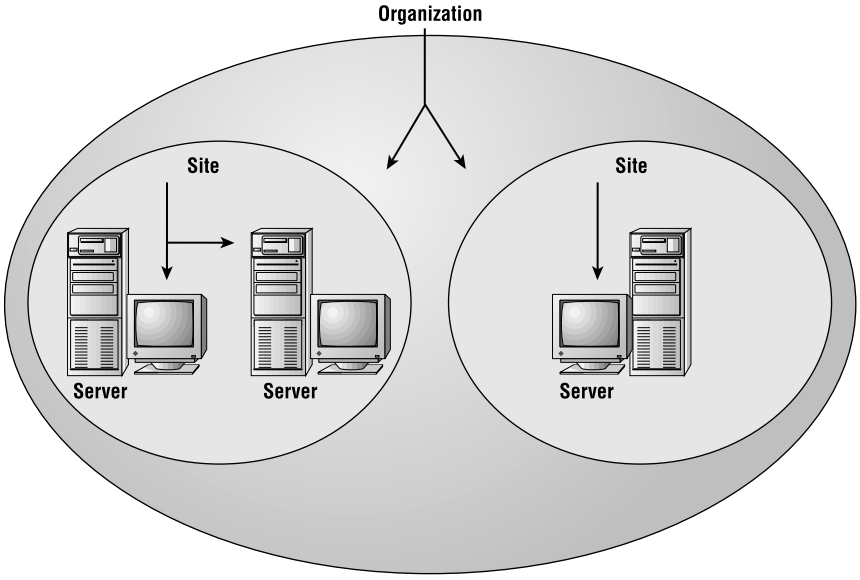


FIGURE 2.1
Three main structures of the Microsoft Exchange hierarchy

Sites

Sites are logical groupings of one or more Exchange servers. Even though resources reside on different servers in the site, the site groups all those resources without reference to their locations. This grouping makes using resources in the site very easy. For example, let us say that a certain mailbox physically resides on site server A. That site server is called the mailbox's *home server*. Senders do not need to know the physical location of the mailbox in order to send messages to it. They simply see the mailbox in the site listing, and send it a message. The same principle applies to public folders in a site. The particular server a public folder is stored on is of no concern to the users wanting to access it. They simply see the public folder listed in their site and access it. This is called *location transparency*. From the user's perspective, a site creates a transparent messaging environment.

A site also creates a smaller, more manageable Exchange network within the larger network of the organization. Administrators can assign a setting to the site (the parent), and it will be inherited by its site servers (the children).

There are two primary prerequisites for a site to be created:

- **Permanent, high-speed connections between the site servers** All communication that takes place within a site is done with RPCs. As you learned earlier, the RPC protocol transfers instructions issued at one machine to another machine for execution, and waits for the data to be

sent back (a request/reply mechanism). For this to work efficiently, there must be a relatively high-speed connection between all the machines in the site. The connection also needs to be permanent. It is strongly recommended that all site servers be within a local area network (LAN) or high-speed wide area network (WAN) with permanent links.

- **All the site servers must be in the same Windows NT security context (*context* relates to location)** When an Exchange component on one site server initiates communication with a component on another site server, the initiating component logs on to the other site server. The Windows NT account that is used for logging on is called the Site Services Account (the actual name of the account could be anything, Jane, Steve, etc., but it is referred to as the Site Services Account). This means all the site servers must be in the same Windows NT domain or domains with trust relationships.

Servers

Exchange servers comprise the final main structure in the Exchange hierarchy. These computers run the Windows NT Server operating system and the Exchange Server software. The Exchange servers are the physical location for mailboxes, folders, and other data and information for the site. Individual servers, while inheriting certain configuration parameters from the site (the parent), can also be individually configured. For example, even though recipients can be managed at the site level, they can also be managed at the server they were created on, their home server. All Exchange objects, as well as all related processes, are created and managed by the software components that make up the Exchange product. These components are divided into two main groups, core components and optional components, which are covered in the following discussion.

Core Components

The Exchange components are executable programs that perform the Exchange functions. Some are in the form of EXE files, others are in the form of Dynamic Link Libraries (DLLs). They are referred to as core components because they are necessary for Exchange to be operational. They are also

referred to as services, because they run as services on the Microsoft Windows NT Server operating system. The core components include:

- Directory Service (DS)
- Information Store (IS)
- Message Transfer Agent (MTA)
- System Attendant (SA)

Table 2.1 uses a post office analogy to illustrate the functions of the core components.

TABLE 2.1 Post office analogy for core components

Exchange Core Component	Post Office Service
Directory Service	Creation of a comprehensive address book
Information Store	Storage and delivery of mail
Message Transfer Agent	Routing decisions for mail to be sent between post offices
System Attendant	Post office management

Directory Service (DS)

The Directory Service (DS) creates and manages the storage of all information about Exchange objects, such as the organization, site, servers, mailboxes, distribution lists, and public folders. The characteristics of these objects are called *properties* or *attributes*. The DS organizes all this information into a hierarchical database called the Directory, which is contained in the file DIR.EDB. The Directory hierarchy is patterned after the X.500 standard.

Structure and Objects of the Directory

This section provides additional details about the main objects in the Directory—organization, site, and server— and information on the other objects in the Directory. Figure 2.2 and Table 2.2 illustrate an X.500 directory structure along with the corresponding Exchange structures.

As can be seen in Figure 2.2 and Table 2.2, the Exchange Directory uses the same basic hierarchical structure as X.500. A minor difference is the naming

FIGURE 2.2
The Exchange Directory and X.500

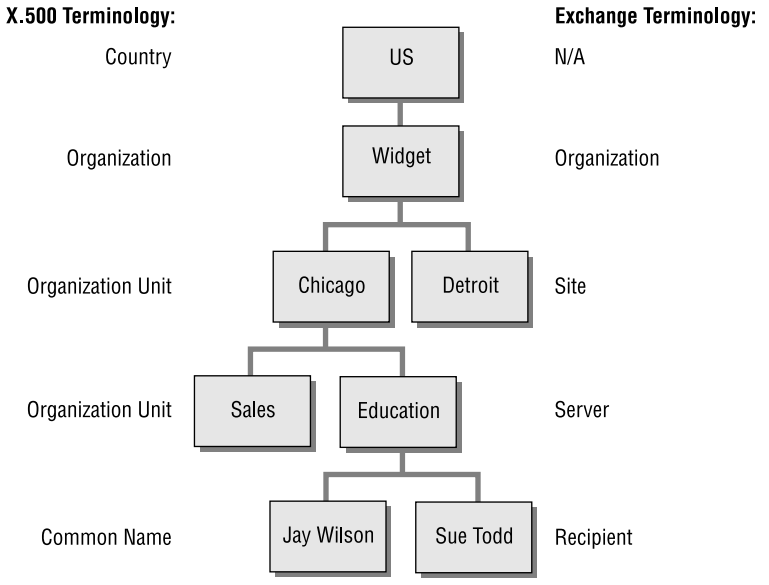


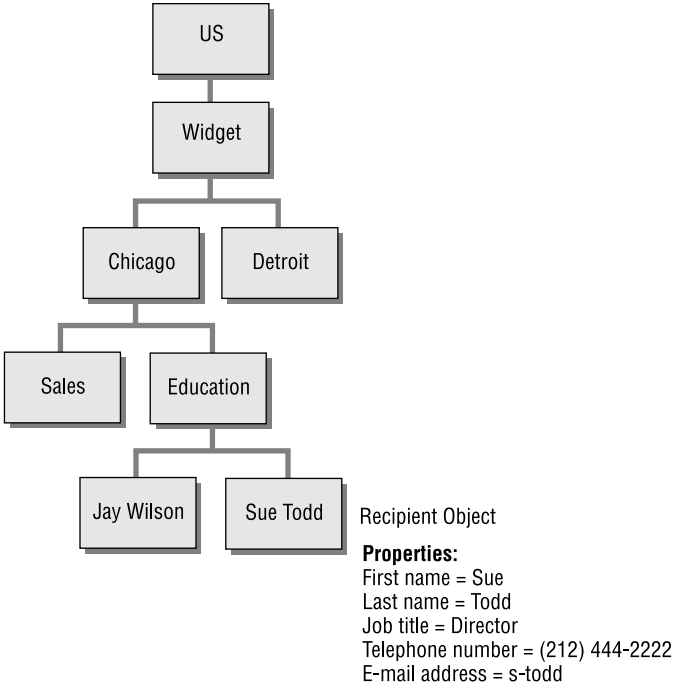
TABLE 2.2
The Exchange Directory and X.500

X.500 Object	Example	Description	Mapping to Exchange
Country	c=US	Country of the organization	N/A
Organization	o=Widget	Name of the organization	Organization name
Organization Unit	ou=Chicago ou=Detroit	Subcategory of the organization	Site name
Organization Unit	ou=Sales ou=Education	Subcategories under the ou=Chicago	Server names
Common Name	cn=Jay Wilson cn=Sue Todd	Name of a specific resource (e.g., username, fax name, printer name, etc.)	Recipient (e.g., mailbox, public folder)

conventions. For example, the Exchange Directory refers to site names and X.500 refers to organization units, but they are easily mapped to each other.

The Exchange Directory, like X.500, contains objects. Objects are the representation, or abstraction, of a resource. That resource can be a person (e.g., Jay Wilson), or even an entire Exchange organization (e.g., Widget). Objects have properties, which are the characteristics of the object. The Jay Wilson object can have *properties* such as first name, last name, job title, telephone number, e-mail address, and many others. Properties are also called *attributes*. Figure 2.3 illustrates the properties of an object.

FIGURE 2.3
Properties of an object



While the Exchange Directory objects map to the main structures in the X.500 hierarchy, Exchange also adds some additional objects. The three main structures (organization, site, server) are discussed next, along with the other objects within them.

Organization Object As stated earlier, the organization encompasses all the Exchange objects of a company or other enterprise. The organization structure is an object and, as an object, it has properties. The primary property is *permissions,* which determine what functions the specified people can perform at the organization level. For example, one Exchange administrator can be given permission to create child objects under the organization, such as additional sites.

Because other objects are contained under the organization object, it is considered a container object. Along with site objects, two other objects are under the organization object:

- **Global Address List (GAL)** The GAL object contains a comprehensive list of all the published mail addresses in the organization. Client software can access this list in order to send information to anybody in the organization. The global address list is considered a *container object* because it contains other objects, in this case, message recipients. These objects include mailboxes, distribution lists (multiple addresses grouped as one), and public folders. Each of these objects are considered *leaf objects*, because they do not have any other objects contained in them.

- **Folders** The folders object contains public and system folders. An example of a public folder could be a folder named Technical Help containing a listing of all the people who provide technical help on the network. Examples of system folders are the organization forms and offline address book. An example of a folder property is replicas, which are the Exchange servers that contain a copy of that folder.

See Figure 2.4 for a depiction of the organization container.

Site Object A site is a logical grouping of Exchange servers. The purpose of the grouping is to create a transparent messaging environment (users do not need to know on which server a resource is located). The properties of a site are the same type as with the organization object, the main one being permissions. Permissions can be assigned to a particular administrator allowing that administrator to create or modify child objects (like mailboxes) in the site. The difference between permissions granted at the organization object versus the site object is the *context* (i.e., the location in the hierarchy). Permissions granted at a site only apply to objects in that site, not to other sites and not to the organization.

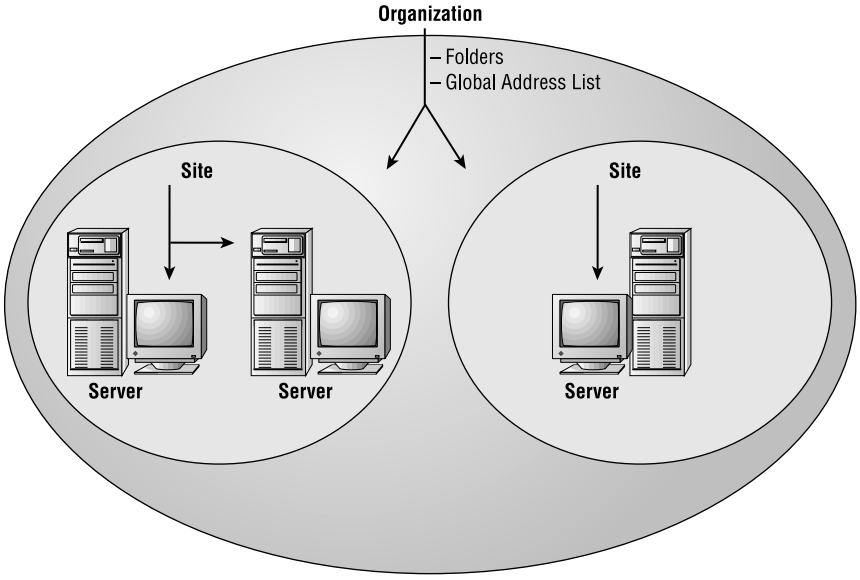


FIGURE 2.4
The organization container object

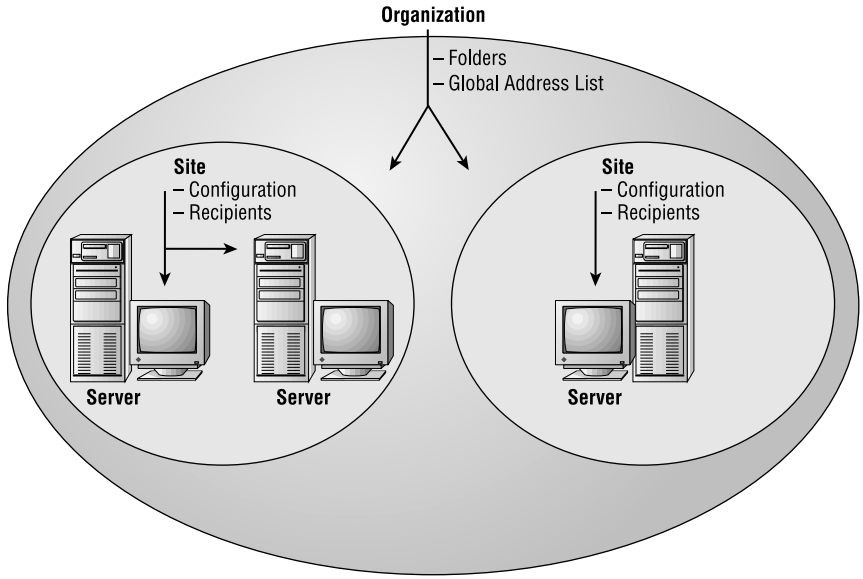
Sites are also container objects with the following objects under them:

- **Configuration** This container object holds seven additional container objects that pertain to numerous aspects of a site configuration. There are also four non-container objects under Configuration. These eleven objects are listed here (each item will be discussed in Chapter 12):
  - **Add-Ins** (container object)
  - **Addressing** (container object)
  - **Connections** (container object)
  - **Directory Replication** (container object)
  - **Monitors** (container object)
  - **Protocols** (container object)
  - **Servers** (container object)
  - **DS Site Configuration**
  - **Information Store Site Configuration**
  - **MTA Site Configuration**
  - **Site Addressing**

- **Recipients** These are objects to which Exchange users can send messages. Recipient objects include:
  - **Mailboxes** A location from which messages can be received, sent, stored, etc. Also stores other types of data.
  - **Distribution lists** A grouping of individual recipients.
  - **Public folders** A storage container for group access.
  - **Custom recipients** E-mail addresses that represent a foreign mail system address.

See Figure 2.5 for an illustration of the site container.

FIGURE 2.5
The site container object



Servers Object Configuration The Servers' container object is under the Configuration container. This object contains a listing of all the servers in the selected site. Each particular server object can be managed individually and contains the following two container objects:

- **Protocols** This object contains the protocols, such as LDAP, NNTP, IMAP4, and POP3, that can be configured on the selected server. (These protocols will be discussed in Chapter 11.)

- **Server Recipients** This object contains all the recipient objects (mailboxes, distribution lists, public folders, and custom recipients) stored on the selected server.

Figure 2.6 illustrates the server container and its objects.

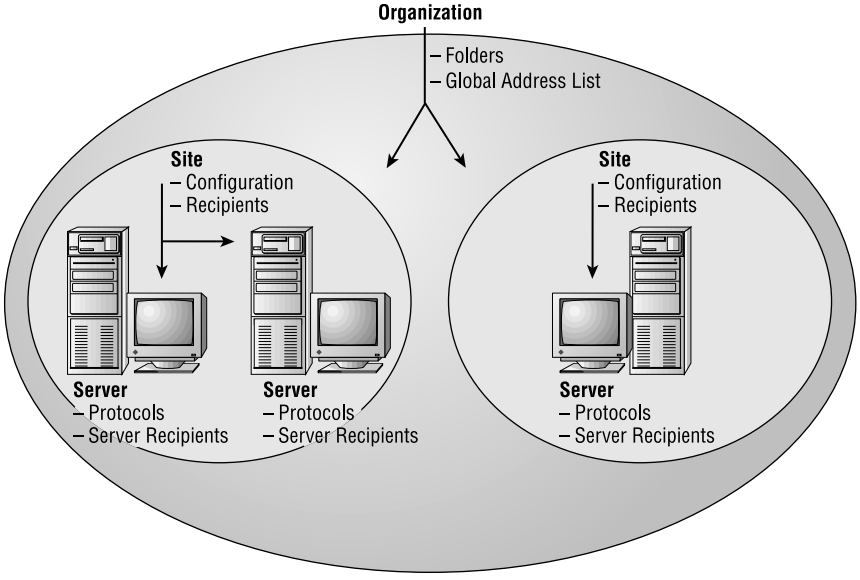


FIGURE 2.6
Exchange server container and objects

Directory Replication

All Exchange servers in a site contain a complete copy of the directory information of that site. This is accomplished by automatic directory replication between servers of a site. When an Exchange object, for example a mailbox, is created on a particular Exchange server, that mailbox information is automatically copied to all the other servers in that site. This is an important factor in creating a transparent messaging environment. Users see all site recipients in their site address book. And while users do not need to know the physical location of a recipient, each Exchange server can route a message to the correct server because it has a copy of the site directory which contains this information.

There can also be directory replication between sites. This is not an automatic process, and must be configured by an administrator. This ability allows

administrators to decide what resources to share with other sites. But directory replication between sites can be used to create an enterprise messaging environment. More on the topic of directory replication appears in Chapter 13.

Directory Access to the DS Directory

The DS component supports directory access through the MAPI interface and through the LDAP interface. This enables both Microsoft Exchange Client software and Internet LDAP-enabled applications to access the Exchange directory. Administrators access the directory through the Microsoft Exchange Administrator program, which is a MAPI program.

Benefits of the DS

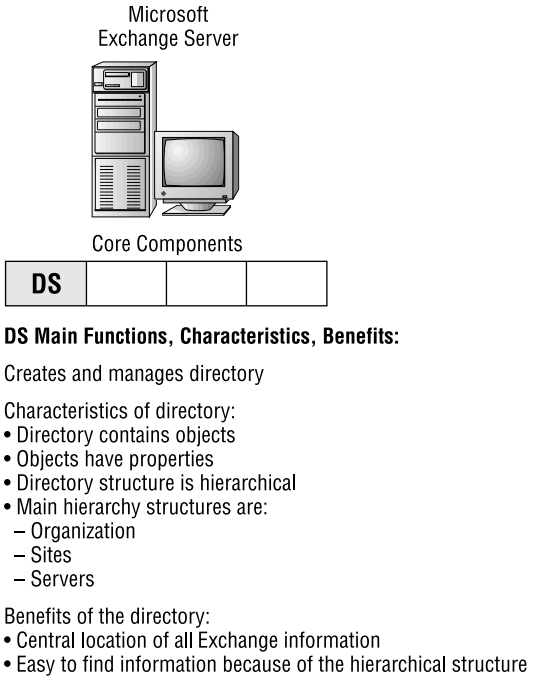
There are many benefits to having an object-oriented hierarchical directory like the one the DS creates and manages. Because all Exchange configuration information is stored in the directory, the other Exchange components know where to go to reference information. Exchange administrators benefit from a graphical view of the Exchange environment, which helps in creating and managing Exchange resources. Administrators manage the Exchange environment by configuring the properties of objects in the directory. Users also benefit from a graphical view of resources, such as public folders and mailboxes, which makes it much easier to find resources. See Figure 2.7 for a review of the main points of the Directory Service.

Information Store (IS)

The Information Store (IS) provides the storage for Exchange message data. The IS is the back-end component for mailboxes, public folders, and their data. Many features of the Exchange client applications, such as folders, permissions, rules, and views, are made possible by the IS. It is also responsible for delivering messages to users on the same server as the sender, as well as notifying users of new messages in their mailboxes.

One of the powerful storage techniques used by the IS is single-instance storage. This means that a message sent to multiple users on the same server is stored only once. Each user would see the message in their mailbox, even though the message is not physically stored in the mailbox. Each of those mailboxes would have a pointer to the physical location of the message. This technique helps minimize storage levels.

FIGURE 2.7
Directory Service (DS)



In addition to supporting access by MAPI clients, the IS also supports access by Internet clients using the Post Office Protocol 3 (POP3) and the Internet Message Access Protocol 4 (IMAP4). Those two Internet protocols are built into the IS.

Even though POP3 is built into the Information Store, it is still sometimes referred to as the Microsoft Exchange POP3 Service.

The IS storage consists of two databases: the Public Information Store, which contains all the Exchange public folders; and the Private Information Store, which contains all the Exchange mailboxes. By default, the IS creates both of these databases on an Exchange server. However, an administrator can choose to have only one of those databases on a server; this will be discussed later in this chapter under the heading "Dedicated Servers."

Public Information Store

The Public Information Store contains all the public folders of a particular server. Public folders are containers for shared information. In a sense, they are like a public mailbox. Different types of data can be stored in a public folder, such as e-mail messages, word processing documents, graphic files, and many other types. Public folders are used to create groupware applications. The Public Information Store is contained in the file PUB.EDB and stored on an Exchange server.

Private Information Store

The Private Information Store contains all user mailboxes. Security makes mailboxes accessible only to their owners and to others who have been given access permission. This database is contained in the file PRIV.EDB and stored on an Exchange server.

By default, a user's messages are stored in their mailbox in this database. Although mailboxes are always located in the Private Information Store on an Exchange server, messages do not have to be stored there. A user can specify a personal location for message storage. This type of folder is called a personal folder and has a PST file extension. A personal folder can be located on a user's local computer or on a server. Even when a user is using a personal folder for message storage, their mailbox is still on their Exchange server and in the Private Information Store. When they are sent a message, the message still first goes to their mailbox in the Private Information Store, and then is rerouted to a personal folder (the message would then physically reside only in the personal folder). The advantage of storing messages in a personal folder, especially if it is on a local computer, is that the user can access it without logging on to the server. The advantages of keeping message storage in the Private Information Store, which is the default, are centralized backup and fault tolerance.

Dedicated Servers

By default, an Exchange server contains both the Public and Private databases. But Exchange servers can also be configured to store only one of the two. If only the Private Information Store were configured, that server would act as a dedicated mailbox server. A dedicated folder server would only have the Public Information Store configured. Dedicated servers are created to improve the performance of information access.

Information Store Management Properties

Before we can begin a discussion of IS management, we need to introduce some IS properties that relate to management issues:

- **Replication** Public folders can be configured to be replicated (copied) to other Exchange servers. This can be done for different reasons, including locating folders for easier network access and fault tolerance.
- **Storage limits** The IS can be configured to set a storage limit for each mailbox.
- **Aging** Folders can be configured to mark and delete information that has been in them for a specified length of time.
- **Views of resource usage** An administrator can view the resources used by a user in the IS. An administrator can view resources by either mailbox or public folder. Resources viewed include disk space, number of stored items, log on time, and last log off time.

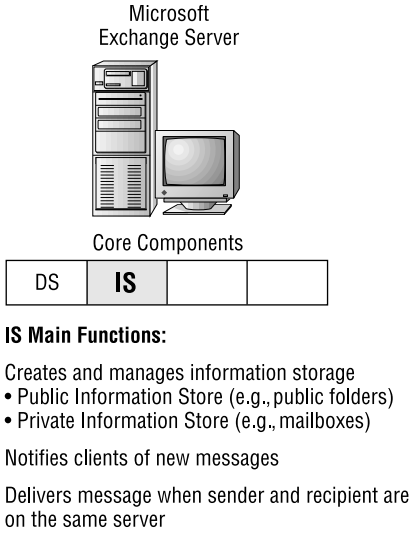
Chapter 9 discusses the management of public folders. Chapter 12 discusses the overall management of an Exchange server, which includes managing the IS.

As an analogy, the IS as a service can be compared to a postal worker who stores letters and packages. If a letter's destination is that post office, this person will place the letter in the correct mailbox. The IS as a database is like the mailboxes and other storage at the post office. Figure 2.8 depicts the Information Store and some of its features.

Message Transfer Agent (MTA)

The Message Transfer Agent (MTA) manages the *routing* of messages between servers in a site, between servers in different sites, and between Exchange and some non-Exchange systems, such as Microsoft Mail. The MTA uses other components, called *connectors,* to manage the transfer of data, while the MTA handles the routing functions. The MTA could be compared to a post office that receives mail and routes it to another post office. The MTA component is based on the CCITT 1984 and 1988 X.400 standard.

FIGURE 2.8
Information Store (IS)



The main routing functions of the MTA component include:

- Addressing messages: O/R Addresses
- Translating message formats
- Making routing decisions

Addressing Messages: O/R Addresses

Because the MTA component is based on X.400, it also uses an Originator/Recipient Address (O/R Address). Chapter 1 discussed the X.400 addressing scheme. Figure 2.9 and Table 2.3 provide a review of the O/R Address and how Exchange information is applied to this addressing scheme.

The example in Figure 2.9 and Table 2.3 could be written the following way:

c=us; a=mci; p=widget; o=Chicago; s=wilson; g=jay

If you compare the way Exchange uses the X.400 address format (Figure 2.9 and Table 2.2) and the X.500 name format (Figure 2.2 and Table 2.2), you will notice a difference in how Exchange uses the organization field. In the X.400

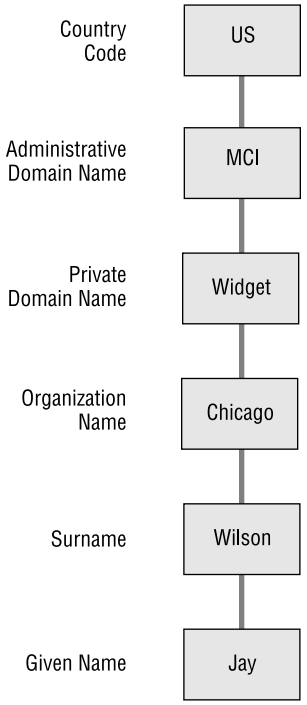


FIGURE 2.9
A Microsoft Exchange network mapped to an X.400 address format

organization field, Exchange places the site name rather than the Exchange organization name. This is probably due to the fact that the 1984 X.400 standard, which predated the X.500 standard, did not include the organization unit field. But when X.500 was published in 1988, the X.400 standard was revised to include X.500 names, such as organization unit and common name. Microsoft Exchange adheres to both the 1984 and 1988 X.400 standard. Exchange by default places the Exchange site name in the organization field. This value can be left in place, or the administrator can easily change it to the Exchange organization name. The administrator can also include an organization unit (ou=) field, placing the site name as its value. Using this format, the X.400 address would look like the following:

c=us; a=mci; p=widget; o=widget; ou=chicago; s=wilson; g=jay

The directory location to modify these addresses is Site/Configuration/Site Addressing.

TABLE 2.3 An X.400 addressing scheme and Microsoft Exchange

Field	Abbreviation	Description	Mapping to Exchange
Country code	c=us	Country	Country
Administrative Management Domain (ADMD)	a=mci	The third-party networking system used (e.g., ATT, MCI, Sprint, etc.)	Abbreviation of third-party network, or if not using one, leave field with a space
Private Management Domain (PRMD)	p=Widget	Subscriber to the ADMD (i.e., company name)	Exchange organization name
Organization	o=Chicago	Name of company or organization	Exchange site name
Surname	s=Wilson	Last name	Last name attribute field for a mailbox
Given name	g=Jay	First name	First name attribute field for a mailbox

Translating Message Content Format

If a message is being routed from Exchange to an X.400 message system, the MTA translates the message from the Exchange format, called the Microsoft Database Exchange Format (MDBEF) to the X.400 message format, called the Interpersonal Message (IPM) format.

This translation ability allows Microsoft Exchange clients to exchange mail with X.400 mail users, or use an X.400 network as a backbone to another Exchange environment and a user on that system. Figure 2.10 illustrates this latter scenario.

Making Routing Decisions

When the MTA receives a message to be forwarded, it first stores that message to determine its next route. This process involves comparing the message

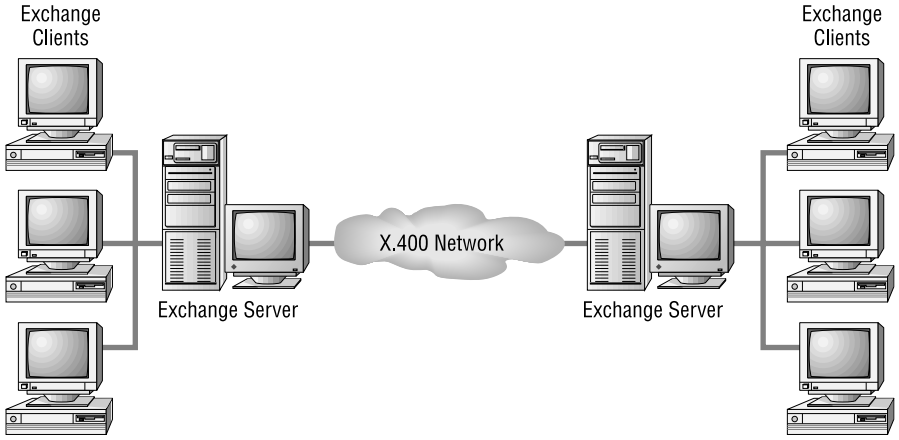


FIGURE 2.10
An X.400 network as a backbone for Exchange networks

recipient's address (either DN or O/R) to the routing table. The routing table, called the Gateway Address Routing Table (GWART), is the list of reachable destinations. If the MTA finds a route to the message recipient's address, it sends the message to the next location in that route. Because there could be multiple routes to a destination, an administrator can assign *costs* to different routes. Costs allow priorities to be given to different routes. The MTA will choose the lowest cost route when selecting a route. If the MTA cannot forward a message, a Non-Delivery Report (NDR) is sent to the message originator. Chapter 13 discusses message routing in greater detail. Figure 2.11 depicts the role of the MTA.

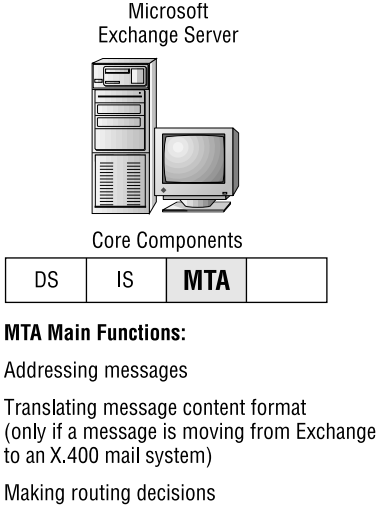


FIGURE 2.11
Message Transfer Agent (MTA)

The Microsoft Exchange MTA is based on the 1984 and 1988 versions of X.400 standards.

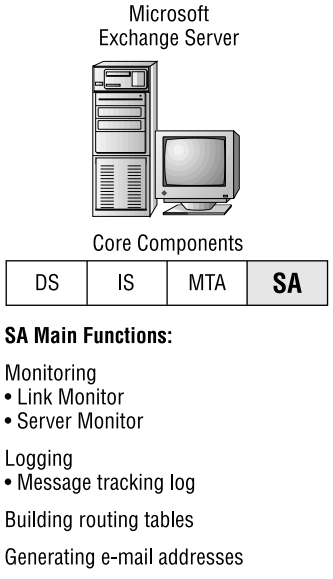
System Attendant (SA)

This service monitors and logs information about the other Exchange services. It also builds and maintains the routing tables for the site. The primary functions of the System Attendant are as follows:

- **Monitoring** Two Exchange features, Link Monitor and Server Monitor, use the SA component to monitor. The Link Monitor sends test messages between servers to see if there is a connection between them. The Server Monitor checks the status of Exchange or other NT services on a server by sending test messages to that server. Both of these features use the SA component to send and receive the test messages.
- **Verifying information** The SA verifies the accuracy of a server's directory replication information. If it is incorrect, the SA will repair the inconsistencies.
- **Logging** The SA writes message tracking information to the message tracking log. Information tracked includes the route a message took, and if the message arrived at its destination.
- **Building routing tables** The SA builds the routing tables of a site. The MTA references these tables when making routing decisions. By default, the routing table is automatically rebuilt once per day. This process can be configured and can be manually initiated through the properties of the MTA (even though it is carried out by the SA).
- **Generating e-mail addresses** When new recipients are created by the DS, the SA ensures that their e-mail addresses are generated.
- **Assisting with advanced security** The SA accepts client requests for advanced security initialization and passes those requests on to the Exchange server advanced security components. Advanced security relates to digital signatures and data encryption. Advanced security will be introduced later in this chapter under the "Optional Exchange Components" section. It is also covered in more detail in Chapter 10.

Figure 2.12 illustrates the main functions of the SA component.

FIGURE 2.12
System Attendant (SA)



Optional Exchange Components

Optional components, as the name implies, are not necessary for the basic operation of Exchange. But these components could be needed for additional functionality in your Exchange environment. Optional components include:

- Connectors or gateways
- Outlook Web Access
- Chat Service
- Scripting Agent
- Key Management

Individual gateways will not be covered because they are documented in their own third-party literature.

Connectors or Gateways

Most of the optional components are connectors or gateways. These are components that enable Exchange to interoperate with non-Exchange systems, sometimes referred to as foreign systems. Connectors are developed by Microsoft, whereas gateways are developed by third-party companies (i.e., not Microsoft). Examples of foreign message systems that have connectors include the following:

- Internet Mail Service (IMS)
- Microsoft Mail Connector
- Schedule+ Free/Busy Connector
- Lotus cc:Mail Connector
- Lotus Notes Connector
- IBM OfficeVision/VM

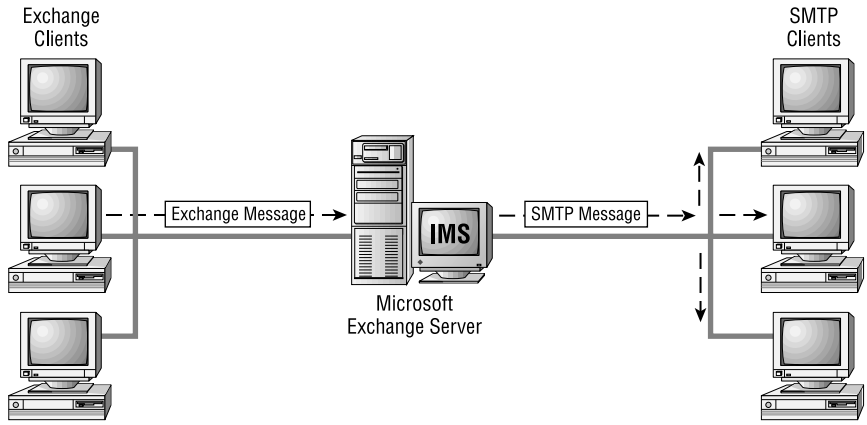
Internet Mail Service (IMS)

The Internet Mail Service (IMS) connector enables an Exchange system to interoperate with an SMTP mail system. SMTP, Simple Mail Transfer Protocol, is the standard mail exchange protocol for the TCP/IP protocol suite. It is used in many intranet mail systems (inside a company or organization), as well as on the Internet. The IMS permits Exchange users and SMTP mail users to send and receive mail to each other. Organizations that already have an SMTP mail system can add Microsoft Exchange, gaining its rich feature set, and yet still have enterprise messaging because of the interoperability through the IMS. The IMS connector also permits an organization with geographically disperse locations to use the Internet as a messaging backbone. Messages can be sent from one Exchange user, through the Internet, to the Exchange recipient. Figure 2.13 illustrates the IMS component. Chapter 14 provides further information on the IMS.

Internet News Service (INS)

The Internet News Service (INS) connector permits an Exchange system to interoperate with USENET newsgroups. USENET is a network of host machines containing posted discussions on various topics. A discussion area is referred to as a newsgroup. Users can access a newsgroup and read the text of a discussion or post their own comments. The USENET uses the Internet, as well as other networks. USENET uses Network News Transport Protocol (NNTP) to replicate newsgroup information between hosts.

FIGURE 2.13
Internet Mail Service (IMS)



The Microsoft Internet News Service includes the NNTP protocol and can download newsgroup information from the USENET into an Exchange public folder, and vice versa. The Internet New Service is discussed in more detail in Chapter 11.

Microsoft Mail Connector

An Exchange system can interoperate with a Microsoft Mail system through the Microsoft Mail Connector. Microsoft Mail is a mail product for PC or AppleTalk networks. The Microsoft Mail Connector can translate Exchange messages to and from the Microsoft Mail format. This permits Exchange users and Mail users to exchange mail. You will find more on the Microsoft Mail Connector in Chapter 15.

If an Exchange system and a Microsoft Mail system want to interoperate, they must share their directory information (e.g., listings of mailboxes, distribution lists, and folders). This involves Exchange exporting its directory information to Microsoft Mail, and importing directory information from Microsoft Mail. The Exchange component that performs this is the Directory Synchronization Agent (DXA). Chapter 15 will provide more information on the DXA.

Schedule+ Free/Busy Connector

Microsoft Schedule+ performs personal and group calendaring, scheduling, and task management. The group scheduling feature allows a user who wants to schedule a meeting to view the schedules of other users. The user scheduling the meeting can see if a particular time period is unscheduled for the other users, called *free*, or if it is already scheduled, called *busy*.

Microsoft Exchange ships with Schedule+ 7.5, while Microsoft Mail ships with the earlier version 1.0. These two versions are incompatible in terms of viewing each others free/busy status. To solve this incompatibility, Microsoft Exchange ships with the Schedule+ Free/Busy Connector which allows the two versions to share their free/busy information. This connector requires the Microsoft Mail Connector. Further information on the Schedule+ Free/Busy Connector is found in Chapter 15.

Connector Lotus cc:Mail

This connector permits Exchange to interoperate with a Lotus cc:Mail system. Users of each system can exchange mail with the other system. As with the other connectors, this connector permits organizations to implement Exchange with another messaging system and still have interoperability. Chapter 15 provides further discussion on the Connector for Lotus cc:Mail.

Lotus Notes Connector

Both the Standard and Enterprise editions of Exchange Server 5.5 include the connector for Lotus Notes. This connector provides message exchange and directory synchronization between Exchange and Lotus Notes or Domino. The Lotus Notes system must be version 3 or 4, or Domino version 4.x. Through this connector, Exchange users can send rich-text (colors, fonts, italic, etc.) to Notes users, receive status messages about message delivery, or read receipts, and non-delivery reports (NDRs). Exchange users can even receive Notes document links. This connector, however, does not provide a connection between Exchange public folders and Lotus Notes databases.

IBM OfficeVision/VM Connector

The Microsoft Exchange Connector for IBM OfficeVision/VM provides connectivity between Exchange and two IBM host-based messaging applications, OfficeVision/VM (Virtual Machine) and PROFS (Professional Office System). Because of the large number of users of IBM host-based systems, this connector can help provide true enterprise messaging in an environment with both Microsoft and IBM. This connector features support for the IBM Network Job Entry (NJE) protocol, the exchange of PROFS and CMS notes and documents, mapping between the IBM host and Exchange status messages, and the ability of the host-based users to send mail to the Internet through Exchange.

This connector requires the Microsoft SNA Server for basic connectivity to the IBM host system. This connector is only shipped with the Enterprise Edition of Exchange Server 5.5.

Outlook Web Access

Outlook Web Access, formerly called *Active Server Components* in version 5.0 of Exchange, are ActiveX programs that ship with Microsoft Exchange Server, but that run on a Microsoft Internet Information Server 3.0 (IIS). Outlook Web Access enables users of any standard Web browser to connect through an IIS server and access Exchange resources, such as mailboxes, public folders, and schedules. Because of the popularity of Web browsers, Exchange Outlook Web Access greatly expands the number of clients for Exchange Server. Chapter 11 provides more extensive coverage on Outlook Web Access.

Chat Service

The Microsoft Exchange Chat Service permits Exchange users and users of the Internet Relay Chat (IRC) protocol to interact through an online, real-time, text-based, conversation. The Chat Service is an example of a groupware feature of Exchange.

Scripting Agent

The Microsoft Exchange Scripting Agent adds some workflow capabilities to Exchange. Developers can write JScript and Visual Basic scripts that add customized functions to an Exchange public folder. The Scripting Agent component executes these scripts. For example, a script could be written to read a value in a purchase request form sent to a public folder, and then based upon that value route the form to the relevant party.

Key Management

The key management components provide additional security features such as encryption and digital signatures. Encryption prevents unauthorized people from reading messages as they are being sent over the network. Digital signatures ensure that the person whose name appears in the From field is the actual sender of the message. Chapter 10 covers the topic of key management.

Summary

The three main objects (structures) of the Exchange hierarchy are the organization, sites, and servers. Each company or enterprise defines one Exchange organization object. The organization object encompasses all the other objects of an Exchange implementation. Two of the objects contained within the organization object are the Global Address List (GAL) and Folders. The GAL is a comprehensive listing of all e-mail addresses of the organization. The folders object contains public folders and systems folders.

Sites are logical groupings of Exchange servers. They create a transparent messaging environment. Site users can use Exchange resources in the site without regard to the physical location of those resources. Two of the objects contained within the site object are the Configuration object and the Recipients object. The Recipients object holds all the recipients created in that site. Examples of recipients are mailboxes, distribution lists, public folders, and custom recipients.

Servers are the computers running the Exchange Server components. They are the physical location of the Exchange mailboxes, public folders, and other information.

Exchange Server is made up of various software components. The components that are required for Exchange to be operational are called core components. The following are the four core components:

- Directory service
- Information store
- Message transfer agent
- System attendant

Other components, called optional components, provide additional functionality that could be needed depending on the particular implementation.

Review Questions

1. The following is NOT a core component of Exchange:
   - A. Internet Mail Service
   - B. System Attendant
   - C. Message Transfer Agent
   - D. Information Store

2. The Information Store is made up of these two databases:
   - A. SQL and ODBC
   - B. Message tracking log
   - C. Private and public information store
   - D. MTA and SA databases

3. This component generates new e-mail addresses:
   - A. Directory Service
   - B. System Attendant
   - C. Global Address List
   - D. Client components

4. This component is involved in advanced security:
   - A. Directory Service
   - B. Information Store
   - C. Key Management
   - D. Message Transfer Agent

5. This component builds routing tables:

   A. System Attendant

   B. Key Management

   C. Information Store

   D. Internet Mail Service

6. This component notifies clients of new mail:

   A. Directory Service

   B. Microsoft Mail Connector

   C. Information Store

   D. System Attendant

7. The Internet Mail Service relates to which protocol?

   A. X.400

   B. RPC

   C. OSF

   D. SMTP

8. Which of the following is NOT an example of a foreign system?

   A. An Exchange server in a different site

   B. IBM SNADS

   C. SMTP system

   D. Microsoft Mail system

9. This component calculates the route a message could take to its destination:

   A. System Attendant

   B. Message Transfer Agent

   C. Information Store

   D. Internet Mail Service

10. This standard is used as the pattern of the Exchange Directory:

   A. X.400

   B. SMTP

   C. LDAP

   D. X.500

11. The following object creates a transparent messaging environment for Exchange users:

   A. Site

   B. System Attendant

   C. Protocols

   D. INS

12. The Global Address List is contained in this object:

   A. Site

   B. Server

   C. Enterprise

   D. Organization

13. Users must know the name of the server on which a public folder is homed in order to access the public folder.

    A. True

    B. False

14. This component enables World Wide Web users to access Exchange resources:

    A. Outlook Web Access

    B. SNMP

    C. X.400

    D. X.500

15. The Message Transfer Agent is based on this standard:

    A. SNMP

    B. SMTP

    C. X.500

    D. X.400