INTRODUCTION TO MIDDLEWARE

Web Services, Object Components, and Cloud Computing

Letha Hughes Etzkorn
Contents

Online Resources ............................................................................................................................ xix
Preface ............................................................................................................................................. xxi
Author ............................................................................................................................................ xxv

SECTION I  The Different Paradigms

Chapter 1  Introduction .................................................................................................................. 3
  1.1 What Is Middleware? ............................................................................................................. 4
  1.2 Technology Review: Sockets ............................................................................................ 5
    1.2.1 Socket Data Structures .............................................................................................. 5
    1.2.2 Socket Library Calls .................................................................................................. 7
    1.2.3 Network Byte Order and How It Is Used with Sockets ............................................. 8
    1.2.4 General Socket Operation ....................................................................................... 9
    1.2.5 Simple Socket Example ......................................................................................... 10
    1.2.6 Sending Data Other than Char Data—Problems with Endianness .......................... 17
  1.3 Brief Introduction to Other Middlewares ........................................................................ 18
    1.3.1 What Are Remote Procedure Calls?—Also Introduction to Synchronous and Asynchronous Operation .......................................................... 18
    1.3.2 What Are Distributed Object-Oriented Components? .......... ................................ 20
    1.3.3 What Is Message-Oriented Middleware? ............................................................... 20
    1.3.4 What Are Service-Oriented Architectures? ............................................................. 21
    1.3.5 What Are Web Services? ....................................................................................... 22
    1.3.6 What Is Cloud Computing? ................................................................................... 23
  1.4 Environmental Monitoring Project ................................................................................... 23
  1.5 Sailboat Marina Management Project ............................................................................. 25
  Exercises ...................................................................................................................................... 26
  Conceptual Questions ................................................................................................................. 27
  Bibliography ............................................................................................................................... 27

Chapter 2  Software Architectural Styles/Patterns for Middleware ............................................. 31
  2.1 Just What Is a “Software Architecture,” Anyway? .......................................................... 31
  2.2 Architectural Styles/Patterns ............................................................................................. 33
  2.3 Architectural Styles/Patterns for Middleware ................................................................ 35
    2.3.1 Gomaa’s Architectural Patterns .............................................................................. 35
    2.3.2 Fielding’s Architectural Styles .............................................................................. 37
    2.3.3 Fielding’s Architectural Properties ....................................................................... 39
  2.4 Architectural Styles/Patterns for Distributed Object-Oriented Components .......... 43
  2.5 Architectural Styles/Patterns for Service-Oriented Architectures .................................. 44
  2.6 Architectural Styles/Patterns for Web Services ............................................................... 46
  2.7 Architectural Styles/Patterns for Cloud Computing ......................................................... 46
  Exercises ....................................................................................................................................... 48
  Conceptual Questions ................................................................................................................. 48
  Bibliography ............................................................................................................................... 48
## SECTION II  Enabling Technologies for Middleware

### Chapter 3  Introduction to Internet Technologies

| 3.1 | Just What Is the Internet, Anyway? .......................................................... | 55 |
| 3.2 | Brief Introduction to TCP/IP and UDP ...................................................... | 55 |
| 3.3 | IP Addresses (IPv4 and IPv6) and Subnetting ........................................... | 57 |
| 3.3.1 | IPv4 Addresses ................................................................................. | 57 |
| 3.3.1.1 | Private IP Addresses and Network Address Translation ......................... | 58 |
| 3.3.2 | IPv6 Addresses ................................................................................. | 58 |
| 3.3.3 | Subnetting ....................................................................................... | 59 |
| 3.4 | Port Numbers .......................................................................................... | 60 |
| 3.5 | Other Important Network Information ..................................................... | 61 |
| 3.5.1 | Internet Control Message Protocol ....................................................... | 61 |
| 3.5.2 | LAN Protocols: Ethernet and Wi-Fi ....................................................... | 61 |
| 3.5.3 | Media Access Control Addresses .......................................................... | 62 |
| 3.5.4 | Hubs, Bridges, Switches, and Routers .................................................... | 62 |
| 3.5.5 | Autoconfiguration for IPv4: Dynamic Host Configuration Protocol ............. | 63 |
| 3.5.6 | Autoconfiguration for IPv6 ..................................................................... | 64 |
| 3.5.6.1 | DHCP for IPv6 .................................................................................. | 65 |
| 3.5.7 | Virtual Local Area Network .................................................................. | 65 |
| 3.6 | Universally Unique Identifiers .................................................................. | 66 |

### Exercises .................................................................................................................. | 68 |
### Conceptual Questions ............................................................................................... | 68 |
### Bibliography ............................................................................................................. | 69 |

### Chapter 4  Introduction to World Wide Web Technologies

| 4.1 | Just What Is the Web, Anyway? ................................................................. | 71 |
| 4.2 | Hypertext Transfer Protocol ........................................................................ | 71 |
| 4.3 | HTML, XML, and HTML Forms .................................................................. | 72 |
| 4.4 | XML Schema Basics ................................................................................... | 79 |
| 4.5 | JavaScript Object Notation (JSON) ........................................................... | 81 |
| 4.6 | Internet Media Types (MIME Types) .......................................................... | 82 |
| 4.7 | Base 64 Encoding .................................................................................... | 82 |
| 4.8 | URL Encoding (Percent Encoding) and URL Base 64 Encoding ................. | 85 |
| 4.9 | Domain Names and Domain Name Servers ............................................... | 85 |
| 4.10 | Document Object Model and Browser Object Model .................................. | 87 |
| 4.11 | Popular Web Servers ............................................................................... | 88 |
| 4.11.1 | Web/Application Servers: LAMP versus Windows/ASP versus Java .......... | 89 |
| 4.11.1.1 | MySQL—The M in LAMP .................................................................... | 89 |
| 4.11.1.2 | Using Java Database Connectivity (JDBC) with MySQL ......................... | 91 |
| 4.11.1.3 | Using MySQLi in PHP with MySQL ...................................................... | 93 |
| 4.11.2 | GlassFish Application Server .................................................................. | 95 |
| 4.11.2.1 | How to Start GlassFish and Run an Application on GlassFish ................. | 95 |
| 4.12 | cURL ......................................................................................................... | 99 |

### Exercises .................................................................................................................. | 101 |
Chapter 5

Security Basics ............................................................................................................. 105

5.1  Just Why Should Anyone Care About Security, Anyway? .......................... 105
5.2  Symmetric Key Cryptography and Asymmetric Key/Public
    Key Cryptography ........................................................................................ 105
5.3  Hash (Message Digest) Functions ................................................................. 105
5.4  Digital Signatures and Message Authentication Codes ............................... 106
5.5  Public Key Infrastructure and Certificate Authorities.................................. 107
5.6  Transport Layer Security and Secure Sockets Layer .................................... 108
5.6.1  But How Does TLS Work? .................................................................... 108
5.7  Cryptographic Message Syntax..................................................................... 110

Exercises .................................................................................................................. 112

Chapter 6

Microsoft Technologies Basics .................................................................................. 115

6.1  Microsoft “World” versus the Rest of the World ......................................... 115
6.2  Dynamic Link Library Files and Windows Side by Side ............................. 115
6.3  Common Language Runtime (CLR)............................................................. 116
6.4  Global Assemblies Cache ............................................................................. 117
6.5  Named Pipes in Windows ............................................................................ 118

Exercises .................................................................................................................. 119

Chapter 7

Cloud Technologies Basics ...................................................................................... 121

7.1  What You Need to Know for the Cloud....................................................... 121
7.2  Just What Are Disk Images and Virtual Machine Images, Anyway?......... 121
    7.2.1  Various Kinds of Disk Images and Virtual Machine Images.......... 121
7.3  Just What Are Hypervisors and Virtual Machines, Anyway? ................. 123
    7.3.1  Some Examples of Type 2 Hypervisors ........................................ 124
    7.3.1.1  Some VirtualBox Installation Hints .................................. 125
    7.3.2  Some Examples of Type 1 Hypervisors ........................................ 125
    7.3.2.1  libvirt ............................................................................. 126
7.4  Software-Defined Networking and Network Virtualization ....................... 126
    7.4.1  Open vSwitch/OpenFlow and Linux Bridge .................................. 128
    7.4.1.1  OpenFlow ....................................................................... 129
    7.4.1.2  How Open vSwitch Works .............................................. 130
7.5  Virtualization Security ................................................................................. 130
    7.5.1  Hypervisor Security ..................................................................... 131
7.6  Cloud Security .............................................................................................. 134
    7.6.1  Physical Data Center Security .................................................... 136

Exercises .................................................................................................................. 137

Conceptual Questions .............................................................................................. 101
Bibliography............................................................................................................. 102

Chapter 6

Microsoft Technologies Basics .................................................................................. 115

6.1  Microsoft “World” versus the Rest of the World ......................................... 115
6.2  Dynamic Link Library Files and Windows Side by Side ............................. 115
6.3  Common Language Runtime (CLR)............................................................. 116
6.4  Global Assemblies Cache ............................................................................. 117
6.5  Named Pipes in Windows ............................................................................ 118

Exercises .................................................................................................................. 119

Conceptual Questions .............................................................................................. 112
Bibliography............................................................................................................. 112

Chapter 7

Cloud Technologies Basics ...................................................................................... 121

7.1  What You Need to Know for the Cloud....................................................... 121
7.2  Just What Are Disk Images and Virtual Machine Images, Anyway?......... 121
    7.2.1  Various Kinds of Disk Images and Virtual Machine Images.......... 121
7.3  Just What Are Hypervisors and Virtual Machines, Anyway? ................. 123
    7.3.1  Some Examples of Type 2 Hypervisors ........................................ 124
    7.3.1.1  Some VirtualBox Installation Hints .................................. 125
    7.3.2  Some Examples of Type 1 Hypervisors ........................................ 125
    7.3.2.1  libvirt ............................................................................. 126
7.4  Software-Defined Networking and Network Virtualization ....................... 126
    7.4.1  Open vSwitch/OpenFlow and Linux Bridge .................................. 128
    7.4.1.1  OpenFlow ....................................................................... 129
    7.4.1.2  How Open vSwitch Works .............................................. 130
7.5  Virtualization Security ................................................................................. 130
    7.5.1  Hypervisor Security ..................................................................... 131
7.6  Cloud Security .............................................................................................. 134
    7.6.1  Physical Data Center Security .................................................... 136

Exercises .................................................................................................................. 137

Conceptual Questions .............................................................................................. 137
Bibliography............................................................................................................. 137
SECTION IV Middleware Using Web Services

Chapter 9 Web Services Architectures .......................................................... 287
9.1 Service-Oriented Architectures ................................................................. 287
9.1.1 In-Depth Look at Service-Oriented Architecture ............................... 288
9.1.2 The Relationship between SOA and Cloud Computing ....................... 291
9.1.3 Implementations of Service-Oriented Architectures............................ 292
9.2 RESTful Architectural Style and Non-RESTful versus RESTful Web Services ................................................................. 293
9.2.1 RESTful Architectural Style .................................................................. 293
9.2.2 Non-RESTful versus RESTful Web Services ........................................ 295
Exercises ........................................................................................................... 297
Conceptual Questions ...................................................................................... 297
Bibliography ...................................................................................................... 298

Chapter 10 Non-RESTful Web Services ......................................................... 301
10.1 Just What Do We Mean by “Non-RESTful Web Services,” Anyway? ...... 301
10.2 SOAP Messaging Protocol ................................................................. 302
10.2.1 Overall SOAP Message Format ....................................................... 303
10.2.2 Encoding of SOAP Messages ........................................................... 303
12.2 Technology Review: ASP.NET Model–View–Controller and ASP.NET CORE

12.2.1 The Model–View–Controller Pattern

12.2.2 What Are ASP.NET, ASP.NET MVC, and ASP.NET CORE, Anyway?

12.2.2.1 Razor

12.2.2.2 Roadmap for Examining ASP.NET MVC and ASP.NET CORE

12.2.3 The Marina Project

12.2.4 ASP.NET MVC4 in Visual Studio 2012

12.2.4.1 Create MVC Project

12.2.4.2 Create Model

12.2.4.3 Create Controller

12.2.4.4 Do a Build Followed by Start without Debugging

12.2.4.5 Examine Default Views of Marina Database

12.2.4.6 Enter Data in Database

12.2.4.7 Show Data Directly from Database

12.2.4.8 Use Other Default Views of Marina Database

12.2.4.9 Change Layout in Shared View

12.2.4.10 Create a New Controller Method (Action Method) and Its New View

12.2.5 Add Asynchronous Controller to Marina Project in ASP.NET MVC4

12.2.5.1 Scheduling Maintenance Using PHP Server

12.2.5.2 Scheduling Maintenance Using Internal ASP.NET MVC4 Server

12.2.6 How Asynchronous Calls Really Work in ASP.NET MVC

12.2.6.1 Simple Asynchronous Example

12.2.6.2 (Async or Sync?) Example Using Two PHP Servers

12.2.7 Redirect

12.2.8 How to Access ASP.NET MVC Capabilities with cURL Commands

12.2.9 Access Controller Using HTTP Commands Other than GET

12.2.10 Routing in ASP.NET MVC

12.2.11 HTML Helpers

12.2.11.1 EditorFor versus TextBoxFor

12.2.11.2 ValidationMessageFor

12.2.11.3 RadioButton

12.2.11.4 TextArea and TextAreaFor Example

12.2.11.5 ListBoxFor

12.2.11.6 DropDownList

12.2.12 ASP.NET Core 1.0 in Visual Studio 2015

12.2.12.1 Create MVC Project

12.2.12.2 Create Model

12.2.12.3 Create Controller

12.2.12.4 Enter Data

12.2.12.5 Use Other Default Views of Marina Database

12.2.12.6 Add RequestMaintenance Action Method and Associated View

12.2.12.7 Add Asynchronous Controller to Marina Project in ASP.NET CORE

12.2.13 Is ASP.NET MVC Really RESTful?
 SECTION V  Middleware for the Cloud

Chapter 13  Introduction to the Cloud and Introduction to the OpenStack Cloud ............... 523
  13.1  Just What Is a Cloud, Anyway? ........................................................................ 523
  13.2  Why the Cloud? Why Not the Cloud? And Who Are the Clouds? .......... 524
    13.2.1  Who Are the Clouds? ........................................................................ 526
    13.2.2  Open Source Software to Support Platform as a Service (PaaS) ... 526
    13.2.3  Open Source Cloud Platforms That Support Infrastructure as a Service .................................................. 527
  13.3  OpenStack Cloud .......................................................................................... 528
    13.3.1  OpenStack Installation Tips............................................................. 531
    13.3.2  How to Use OpenStack .................................................................. 532
    13.3.3  How to Log into OpenStack Instance with SSH ............................ 539
    13.3.4  OpenStack Keystone ........................................................................ 541
      13.3.4.1  Security Groups ................................................................ 542
      13.3.4.2  Tokens .............................................................................. 542
      13.3.4.3  Open Standard for Authorization (OAuth) in Keystone .... 549
      13.3.4.4  Keystone Trusts ................................................................ 549
    13.3.5  OpenStack Nova.............................................................................. 551
    13.3.6  OpenStack Neutron.......................................................................... 554
    13.3.7  OpenStack Block Data Storage and Object Data Storage............... 556
  13.4  Technology Review: OpenStack RESTful Interface .................................... 557
    13.4.1  The Various OpenStack Interfaces and Port Numbers .................... 558
    13.4.2  Accessing the OpenStack RESTful Interface Using cURL ............. 558
      13.4.2.1  Authentication Using Keystone........................................ 558
      13.4.2.2  Starting an Instance (Server) Using Nova ....................... 563
    13.4.3  Accessing the OpenStack RESTful Interface Using Python ......... 568
      13.4.3.1  Authentication Using Keystone........................................ 568
      13.4.3.2  Starting an Instance (Server) Using Nova ....................... 569
    13.4.4  Accessing the OpenStack RESTful Interface Using Command-Line Clients .................................................. 571
      13.4.4.1  Using the Common Client—The OpenStack Command Line Client (OSC) .................................................. 571
      13.4.4.2  Using Separate Keystone and Nova Clients ....................... 572
  Exercises ............................................................................................................ 574
  Conceptual Questions .......................................................................................... 574
  Bibliography........................................................................................................ 575

Chapter 14  Introduction to Amazon Web Services and Introduction to the CloudStack Cloud .......................................................... 579
  14.1  Amazon Web Services Cloud ..................................................................... 580
    14.1.1  How to Use Amazon Web Services ............................................ 581
    14.1.2  How to Log into AWS EC2 Instance with SSH ............................ 583
SECTION VI  Message-Oriented Middleware

Chapter 15  Introduction to Message-Oriented Middleware.............................................617
15.1 Just What Is Message-Oriented Middleware, Anyway?........................................617
15.2 Point-to-Point versus Publish/Subscribe ...............................................................617
15.3 When Would You Want to Use Message-Oriented Middleware?.............................618
15.4 What Middleware Technologies Can Be Described as Message-Oriented Middleware?..........................................................619
15.5 CORBA Event Service and CORBA Notification Service........................................620
15.6 Java Message Service and Message Beans...............................................................621
15.6.1 Message Beans Accessed by Java Message Service ........................................622
15.7 Object Management Group (OMG) Data Distribution Service (DDS).....................623
15.8 Advanced Message Queueing Protocol (AMQP)......................................................625
Exercises .......................................................................................................................626
Conceptual Questions .................................................................................................626
Bibliography ................................................................................................................626

SECTION VII  Comparison of Middlewares

Chapter 16  Introduction to Comparing Middlewares .......................................................631
16.1 How Can Different Middlewares Be Compared, Anyway?.......................................631
16.2 How Do Middleware Technologies Map to Our Selected Architectural Styles/Patterns? ........................................................639
16.2.1 Architectural Styles/Patterns Related to the Cloud.............................................642
6 Microsoft Technologies Basics

6.1 MICROSOFT “WORLD” VERSUS THE REST OF THE WORLD

Over the years, Microsoft has had a number of business reasons to have their technologies be just a little different from other companies. For one thing, for many years they owned the desktop operating system market. So there was a big incentive to make technologies that mapped into the Windows operating system, and not nearly as many incentives to make sure that all those technologies also mapped into Linux, or Solaris, or OS/2 (which hasn’t been supported for several years now, although I believe it still has some legacy places where it’s being used). Remember that Microsoft has been a very important driver of the PC industry for three decades; other operating systems have come and gone in that same time period.

The world of Middleware has seen several different Microsoft offerings. Back in the 1990s, Distributed Component Object Model (DCOM) was a major competitor to CORBA (Common Object Request Broker Architecture). Starting in 2002, .NET Remoting became a widely used Microsoft middleware. Then in 2006, Microsoft released Windows Communication Foundation, which has been a competitor to JAX-WS, JAX-RS, and AJAX, while .NET Remoting has been a Microsoft competitor to CORBA and to Enterprise Java Beans.

We’re going to look at .NET Remoting in Chapter 8, then at Windows Communication Foundation in Chapters 10 and 12. Here in this chapter, however, we’re going to do an overview of some Windows technologies that will provide you with some background later on for understanding these Microsoft middleware technologies. And what’s more important, it will give you some essential background you may need later on when you actually use these Microsoft middleware technologies.

6.2 DYNAMIC LINK LIBRARY FILES AND WINDOWS SIDE BY SIDE

Dynamic Link Library (DLL) files are what Microsoft used to implement a shared library. A shared library contains library routines that are stored separately from the calling program executable (.exe in Microsoft Windows) files. The library routines called by the calling program .exe file are not loaded into memory until runtime. The library routines are stored separately from the .exe file. That way the same library files can be shared by different .exe files.

The alternative to a shared library is a static library. With a static library, the library routines are compiled, and the calling program is compiled; then the calling program is linked together with the library routines by the linker, and included together in a single .exe file. A separate .exe file that also needed the same library routines as the first .exe file would have to link them together into its own .exe file. This means that there can be multiple copies of the same library routines on the same hard drive (or flash memory), one copy of the library routines for the first .exe file, and another copy of the library routines for the second .exe file. Having multiple copies of the same thing at the same time is a waste of memory, both hard disk (flash) and RAM (after the .exe file is loaded into RAM in order to run). So the .dll files (shared libraries) were intended to save the ‘multiple copies of library routines’ problem.

However, the use of .dll files by Microsoft led to a strange situation known as “DLL Hell.” One example of this is a situation called “DLL Stomping”: When installing Application B sometime after Application A was installed, the .dll file associated with Application B (which has the same name as the .dll file associated with Application A) overwrote the Application A .dll file. Unfortunately, Application B’s .dll file is an older version of the .dll file that is used by Application A. So Application A no longer works.
The main fix for this is the Windows Side by Side directory (WinSxS). Go to \Windows\WinSxS to see this directory in either Windows 7 or Windows 8. Here is an example from the computer where I’m typing this—as I write this I’m using Windows 8.1:

\Windows\WinSxS\x86_wpf-windowsbase_31bf3856ad364e35_6.3.9600.16384_none_ec0cd16b635fb4bd contains the file:

WindowsBase.dll Date modified: 7/6/2015 at 8:07am

It also contains the directory:

\Windows\WinSxS\x86_wpf-windowsbase_31bf3856ad364e35_6.3.9600.17810_none_ec54726d632a972e which contains the file:

WindowsBase.dll Date modified: 4/30/2015 at 3:35pm

So we have two different .dll files, with the same name, each with a different date and inside a directory with a really long horrible name. These names are “strong names” and are generated by using the private key that comes with the public key distributed with the assembly.

The Windows Side by Side allows an application to keep its older version of the .dll file, and continue to use it, when an updated .dll file of the same name has been installed.

By the way, here is a helpful trick you can use to find out which version of Windows you’ve got: Google (or Bing) the following string, “Which Windows operating system am I running” and the first result will (probably) be the following web page:

http://windows.microsoft.com/en-us/windows/which-operating-system which will tell you which version of Windows you have.

Of course you could always just go directly to this web page, but who remembers that URL. ☺ And of course there are other ways to do this, too.

6.3  COMMON LANGUAGE RUNTIME (CLR)

The Microsoft Common Language Runtime (CLR) is the virtual machine used by all .NET routines. CLR implements a program code execution environment which is defined by the Common Language Infrastructure (CLI). The lowest level human readable language in the CLI is called the Common Intermediate Language, Microsoft’s implementation (the Microsoft bytecode) is called the Microsoft Intermediate Language (MSIL).

The CLR does many of the same things that the Java Virtual Machine (JVM) does; however, the CLR is Microsoft and .NET based, whereas Java implements its own separate technologies. The JVM predated the CLR: the CLR dates from .NET version 1, in 2002, whereas the JVM dates from the late 1990s. However, the concept of virtual machines in general can be dated from the late 1960s—see Everything VM (2010).

The CLR does garbage collection. That is, any memory that is no longer used is returned automatically to the list of free memory that can be reused (heap) by the CLR. This kind of data is called “managed” data because its lifetime is managed by some software entity (in this case the CLR). The primary reason for managed data is to prevent memory leaks. We’ll discuss memory leaks in depth later on in Section 8.2, when we talk about how CORBA handles memory leaks. However, briefly, what happens in a memory leak is, somehow or other when you’re done with using dynamically allocated memory, you forgot to return that memory to the heap. Even if this is an infrequent bug, eventually all the free memory on your system will be used up, and your system will crash! But the memory leak
bug that caused the issue may have actually happened a long time before—hours or perhaps days. All you will be able to see when you look at the immediate cause of the bug is that your code tried to allocate new memory, but there was no free memory available, so the system crashed.

The CLR also supports a Common Type System that defines how types are declared, used, and managed in the CLR. The Common Type System is used to integrate different programming languages by specifying rules for languages to follow, which enables different languages to interact.

The Common Type Systems supports two different kinds of data types:

1. Value types—directly contain their own data. Allocated on the stack or inline in a structure. Can be user-defined or built in.
2. Reference types—store a reference to the memory address of a value. Allocated on the heap. They can be self-describing (arrays or class types), pointer types, or interface types.

The CLR also performs thread management.

6.4 GLOBAL ASSEMBLIES CACHE

Windows computers running the Common Language Runtime have a collection of assemblies located in the Global Assembly Cache, otherwise known as the “GAC.” The GAC stores assemblies that are meant to be shared by multiple applications on the same computer.

The GAC is located at either the directory:

\Windows\Microsoft.NET\assembly

or

\Windows\assembly

depending on which version of .NET you are using. According to MSDN, \Windows\Microsoft.NET\assembly is for .NET Framework 4 and later.

As I write this, I’m using Windows 10 (yes, I upgraded from when I wrote the earlier parts of this chapter). When I look at the first directory, I see the information in Figure 6.1.

When I look at the second directory, I see the information in Figure 6.2.

So clearly there’s some backward compatibility going on. 😊

Assemblies in the GAC_MSIL directory can run in either 32-bit or 64-bit mode, they’re stored in MSIL format and converted to the correct word size on the fly (JIT compiled/interpreted). The 32-bit directory runs 32-bit code, and the 64-bit directory runs 64-bit code.

If you want to install an assembly yourself into the GAC to make the assembly global to your computer, one way to do it is to use the Global Assembly Cache Tool, gacutil.exe. This is part of the Microsoft Windows SDK. You have to have administrator privileges to use gacutil.exe.

FIGURE 6.1  GAC at \Windows\Microsoft.NET\assembly.
I don’t want to go through the details of how to install an assembly in the GAC (there are some references in the bibliography that will help if you find you need to do this). Briefly:

- `gacutil /i` installs an assembly in the GAC
- `gacutil /u` uninstalls an assembly in the GAC

Note that you need to know more to do this; this isn’t sufficient. ☺ Read through more details from the references before you try.

However, just to show a brief example of `gacutil.exe`, we will look at Figure 6.3, where I used `gacutil.exe` to simply look at the contents of my GAC (I ran this one on a Windows 7 machine—yes, we’re seeing lots of different versions of Windows in this chapter).

### 6.5 NAMED PIPES IN WINDOWS

Dating back to the early days of UNIX back in the 1970s, a “pipeline” of processes is when the standard output stream of one process maps to the standard input stream of the next process. Each individual instance of mapping one process’ output stream to the next process’ input stream is called a “pipe.” For example:

```
process1 | process2 | process3
```

means that the output of process1 is the input to process2, etc. Here the individual pipes exist only inside the Linux kernel and are not accessible outside it.

On Unix/Linux systems, a “named pipe” is an extension of the pipeline concept where the name of the named pipe is a file name, and processes can read and write to the named pipe as if it were a file; however, the named pipe data is passed internally by the Linux kernel without storing it in the file system.

A named pipe in Windows is different from a named pipe in Unix/Linux systems in that it works more like a socket, providing a conduit for communication between a client and a server. In Windows (and in .NET Remoting), a named pipe is used for communication between a pipe server and pipe client(s) and can provide communication between processes on the same computer or processes on a network.
EXERCISES

1. Think about garbage collection in terms of its use in middleware. What are its advantages? What are its disadvantages? Are there any middleware-related situations where someone would be better off not using garbage collection?

2. In middleware, especially in special-purpose small embedded systems (hint, hint), are there any situations in which shared runtime libraries such as .dlls would be useful? When would the positives outweigh the negatives?

3. Why did Microsoft bother implementing named pipes in Windows?
CONCEPTUAL QUESTIONS

1. Distributed Computing Object Model (DCOM) is still supported by several Windows systems. Do a web search to find this out: is DCOM still used for new development today? Also, did CORBA “win” versus DCOM in the middleware wars of the late 1990s and early 2000s?

2. Compare the Java Virtual Machine to the Common Language Runtime. How are they alike? How are they different? In terms of features, usability, and performance, is one better than the other?

3. Do you think shared libraries are a good idea? Why or why not? If you’re interested in performance, especially in saving space in compiled applications, do you have any other options, other than shared libraries or statically-linked libraries?

4. Compare Microsoft middleware technologies in general to their major non-Microsoft competitors. What are the advantages? What are the disadvantages? Do a web search and a formal literature review.

5. Why do you think Value types are allocated on the stack, whereas Reference types are allocated on the heap?

BIBLIOGRAPHY


