UNDOCUMENTED WINDOWS 2000 SECRETS

UNDOCUMENTED WINDOWS 2000 SECRETS:

A PROGRAMMERS COOKBOOK

SVEN B. SCHREIBER

ADDISON–WESLEY

Boston San Francisco New York Toronto Montreal
London Munich Paris Madrid Capetown
Sydney Tokyo Singapore Mexico City

Pearson Education Corporate Sales Division
One Lake Street
Upper Saddle River, NJ 07458
(800) 382-3419
corpsales@pearsontechgroup.com

Visit AW on the Web: www.awl.com/cseng/

Library of Congress Cataloging-in-Publication Data

Schreiber, Sven B., 1958–
Undocumented Windows 2000 secrets: a programmer's cookbook / Sven B.Schreiber
p. cm.
Includes bibliographical references and index.
ISBN 0-201-72187-2
1. Microsoft Windows (Computer file) 2. Operating systems (Computers)
I. Title.
QA76.73.O63S389 2001
005.4'4769—dc21

00–054836

ISBN 0-201-72187-2
Text printed on recycled paper
1 2 3 4 5 6 7 8 9 10—CRS—0403020100
First printing, May 2001

To all the people in the world who never stopped asking "why" . . .

Contents

Preface

After finishing the manuscript of my first book, *Developing LDAP and ADSI Clients for Microsoft Exchange* (Schreiber 2000) in October 1999, I was honestly convinced for some time that I would never write a book again. Well, this phase didn't last long, as the pages you are reading right now prove. Actually, I was already starting to think about writing another book as early as November 1999 while I was playing around with the latest release candidate of Microsoft Windows 2000. Examining the kernel and its interfaces and data structures, I was very pleased to find that this operating system—despite its ugly name that reminded me too much of Windows 95 and 98—was still a good old Windows NT.

Poking around in the binary code of operating systems has always been one of my favorite pastimes. Just a couple of weeks before I had the idea to write this book, my article "Inside Windows NT System Data" (Schreiber 1999), showing how to retrieve internal system data by means of the undocumented kernel API function *NtQuerySystemInformation()*, had been published in *Dr. Dobb's Journal*. The preparatory research to this article left me with a huge amount of unpublished material that longed for being printed somewhere, and so I yelled: "Hey, how about a book about Windows 2000 Internals?" Because of the obvious similarities between Windows NT 4.0 and Windows 2000, plus my pile of interesting undocumented information too valuable to be buried, this seemed to be a great idea, and I am proud that this idea took the physical form of the book you are holding in your hands. While transforming the stuff I had collected into something that was readable by other people, I discovered lots of other interesting things, so this book also features a great deal of brand-new material that I hadn't planned to include beforehand.

Addison-Wesley has a long and glorious tradition of publishing books of this kind. The list of milestones is long, containing the two-volume printed version of Brown and Jim Kyle's classic "Interrupt List," *PC Interrupts* and *Network Interrupts* (Brown and Kyle 1991, 1993, 1994, available online at

http://www.cs.cmu.edu/afs/cs.cmu.edu/user/ralf/pub/WWW/files.html); two editions of *Undocumented DOS* (Schulman et al., 1990, 1993); the Windows 3.1 update thereof, titled *Undocumented Windows* (Schulman et al., 1992); Matt Pietrek's *Windows Internals* (Pietrek 1993); as well as `DOS Internals` by Geoff Chappell (Chappell 1994) and *Dissecting DOS* by Michael Podanoffsky (Podanoffsky 1994). Other authors have contributed invaluable material in other areas such as PC hardware programming (Sargent III and Shoemaker 1994, van Gilluwe 1994). Andrew Schulman and Matt Pietrek both have written two more books of this "undocumented" kind about Windows 95, this time for IDG Books Worldwide (Schulman 1994, Pietrek 1995). However, what has been painfully missing in the past few years was a similar book for Windows NT system programmers. Fortunately, some information has been made available in article form. For example, Matt Pietrek has filled some parts of this huge gap in his "Under the Hood" column (e.g., Pietrek 1996a-d) in *Microsoft Systems Journal* (*MSJ,* meanwhile transformed into *MSDN Magazine*), and `The NT Insider` journal, published bimonthly by Open Systems Resources (OSR), is an indispensable source of know-how (check out *http://www.osr.com/publications_ntinsider.htm* for a free subscription). We should also not forget about the incredible SysInternals Web site operated by Mark Russinovich and Bryce Cogswell at *http://www.sysinternals.com*, who have brought us numerous powerful utilities—some even including full source code—that really should have been part of the Microsoft Windows NT Resource Kit.

For a long time, the Microsoft Windows NT Device Driver Kit (DDK) has been the only comprehensive source of information about Windows NT system internals. However, the DDK documentation is tough to read—some people even say it's anything but didactic. Moreover, the DDK is documenting just a small part of the available system interfaces and data structures. The largest part belongs to the category of the so-called internal features. This has made it next to impossible to write nontrivial kernel-mode software for Windows NT without searching for additional information somewhere else. In the past, however, reliable documentation and sample code was hard. This situation improved somewhat in 1997 when *The Windows NT Device Driver Book* by Art Baker and Rajeev Nagar's *Windows NT File System Internals* appeared. The year 1998 was marked by the heavily revised update of Helen Custer's two-volume set *Inside Windows NT* and *Inside the Windows NT File System* (Custer 1993, 1994) by David Solomon (Solomon 1998), who added a lot of new, previously unpublished material. In early 1999, *Windows NT Device Driver Development*, written by NT kernel-mode experts Peter G. Viscarola and W. Anthony Mason, was published (Viscarola and Mason 1999). Both are consulting partners of Open Systems Resources (OSR), and the OSR people are well known for their excellent kernel-mode driver and file system programming courses, so having their first-hand knowledge in printed form was a real benefit. In the same year, Edward N. Dekker and Joseph M. Newcomer's *Developing Windows NT Device Drivers* (Dekker and Newcomer 1999) appeared. Both books were probably the first

comprehensive and *practically useful* NT kernel-mode programming tutorials. If you want to read thoughtful and accurate reviews about several Windows NT driver programming books, including some of those mentioned above, just peek into the November/December 1999 issue of *The NT Insider* (Open Systems Resources 1999c).

In late 1999, more authors delved into the depths of the Windows NT kernel. A trio of system programmers and consultants from Puna, India, Prasad Dabak, Sandeep Phadke, and Milind Borate, brought us the first *Undocumented Windows NT* book (Dabak, Phadke, and Borate 1999). Because it was published before the release of the final version of Windows 2000, it covers Windows NT 3.51 and 4.0 and the latest 2000 beta then available. After this overdue release, another must-have book for anyone in Windows 2000 system programming appeared in January 2000: Gary Nebbett's *Windows NT/2000 Native API Reference* (Nebbett 2000). His work provides a thorough, comprehensive, and extraordinarily detailed documentation of `all` API functions and the structures they involve, providing the first opportunity to double-check my findings about `NtQuerySystemInformation()` in *Dr. Dobb's Journal* (Schreiber 1999). I was pleased to see that not only did our technical details match perfectly but so did most of the symbolic names we had chosen independently. This book should have been published by Microsoft years earlier!

You may think that everything has already been said about the internals of Windows 2000. Not so! The internal functions and data structures inside the kernel involve such a vast area of knowledge that they can hardly be covered by just two books. The book you're reading now is just one of the building blocks required for a better understanding of the architecture of Windows 2000. Hopefully, the coming years will bring many more publications of this kind. One recent publication is the third edition of *Inside Windows NT*. Because it covers many of the new features introduced with Windows 2000, it is consequently now called *Inside Windows 2000* (Solomon and Russinovich 2000). This third edition is a genuine member of the former *Inside Windows NT* series. In this edition, Mark Russinovich is co-author with David Solomon, which is a reliable indicator that this book can be bought blindly. There is no significant overlap between the contents of *Inside Windows 2000* and *Undocumented Windows 2000 Secrets: A Programmer's Cookbook,* so it is probably a good idea to have both on the shelf.

THE TOPICS DISCUSSED IN THIS BOOK

If I had to compare this book with its predecessors, I'd say it's written in the tradition of the old *Undocumented DOS* volumes (Schulman et al., 1990, 1993). I treasured these books back in the days of DOS programming, because they involved an ideal trade-off between comprehensiveness and an in-depth treatment of an essential subset of topics. In my opinion, it is nearly impossible to write a comprehensive documentation of the internals of a complex operating system in a single volume without losing detail. If you

don't opt for a multitome encyclopedia, you can either write a reference handbook, such as Nebbett's (Nebbett 2000), or focus on specific topics, as Andrew Schulman and friends did. Nebbett didn't leave much for other reference authors, but the in-depth documentation of special Windows NT/2000 topics is still a wide-open field.

Like *Undocumented DOS*, this book of introduces Windows 2000 programming topics that I have found both interesting and useful. In the "undocumented" business, there is always a danger of doing "art for art's sake." Unveiling undocumented features of an operating system is usually very exciting and fulfilling for the person doing the work, but might be quite irrelevant for others. Not everything that is undocumented is automatically useful as well. Some operating system internals are just internals in their strictest sense, that is, implementation details. However, many internals of Microsoft systems are much more than that. Microsoft has a notorious reputation for intentionally preventing the developer community from knowing too much about their target operating system. My favorite example is the ingenious and extremely useful MS-DOS Network Redirector Interface described in Chapter 8 of the second edition of *Undocumented DOS* (Schulman et al., 1993). Much time would have been saved and much trouble avoided if Microsoft had documented this wonderful interface when they introduced it. Unfortunately, Microsoft has pursued this information policy with the follow-up systems Windows 3.x, Windows 9x, Windows NT, and Windows 2000. However, books such as this are being written to provide further information.

After introducing you to the basic architecture of Windows 2000 and helping you to set up your workstation for Windows 2000 kernel spelunking, this book leads you through some very exciting corners inside the world of kernel programming. Typically, each chapter first discusses the essential theory you need to know about the topic, and then immediately presents sample code to illustrate the respective features. The language chosen for the samples is plain C. The probability is high that the readers of this book are comfortable with C, and this language is well supported by Microsoft's Windows 2000 development tools.

This book deliberately does `not` attempt to give a broad overview of the architecture of the Windows 2000 kernel, although it discusses parts of it in some chapters. If you are looking for such information, see *Inside Windows 2000* (Solomon and Russinovich 2000) instead, which takes a very general and theoretical approach to the Windows 2000 internals. Neither *Inside Windows 2000* nor the *Undocumented Windows 2000 Secrets: A Programmers Cookbook Windows NT/2000 Native API Reference* (Nebbett 2000) provide practical code examples and full-featured sample applications that interact live with the system contains reprinted code samples in abundance, accompanying the Windows 2000 concepts and features under discussion. The companion CD contains all of this code in ready-to-run applications that you can extend, tear to shreds for use in other applications, or simply use as is. Basically, you will be led through the following topics:

- Using the Windows 2000 debugging interfaces
- Loading, parsing, and using the Windows 2000 symbol files
- Dissecting the Microsoft PDB file format
- Interfacing to the kernel's Native API
- Writing simple kernel-mode drivers
- Exploring Windows 2000 system memory
- Hooking and monitoring calls to the Native API
- Calling kernel functions from user-mode applications
- Calling nonexported kernel functions
- Exploring the world of Windows 2000 kernel objects

You also will learn many more details about the system both in the text and samples. My foremost intention while writing the manuscript was to share with you anything I knew about the topics covered.

THE ORGANIZATION OF THIS BOOK

After considering which sequence of chapters would be optimal for all potential readers of this book, I have arranged the seven chapters in the order that I thought would be best for a novice Windows 2000 system programmer. Therefore, any new concepts or techniques introduced in a chapter are explained at their first appearance. Consequently, newbies should read the chapters sequentially as they appear in the book. This approach may bore expert readers who look for more "off-road" information. However, it is easier for an expert to skip over familiar details than for a novice to keep up if the material is presented in a nondidactic sequence.

Here is what awaits you in each chapter:

- **Chapter 1** begins with a guided tour through setup and use of the Windows 2000 Kernel Debugger, because this is one of the most helpful tools for system exploration. Other highlights are the official Windows 2000 debugging interfaces in the form of the `psapi.dll, imagehlp.dll,` and `dbghelp.dll` components. The chapter closes with detailed descriptions of the layouts of Microsoft CodeView and Program Database (PDB) files, complemented by a sample symbol file parser DLL and an accompanying client application.

- **Chapter 2** introduces the Windows 2000 Native API, discussing the main system service dispatcher, the various API function groups exported by ntdll.dll and ntoskrnl.exe, and the data types most frequently used by these components.
- **Chapter 3** is a short and easy introduction to basic kernel-mode driver development. It is by no means intended as a tutorial for heavy-duty hardware driver developers. It simply points out essential information required to understand the sample code in subsequent chapters, including loading and unloading driver modules at runtime via the Service Control Manger interface. Probably the most interesting highlight is the description of the customizable driver wizard included with full source code on the companion CD.
- **Chapter 4** is certainly the most challenging chapter for readers suffering from hardware phobia, because it starts with a detailed description of the Intel Pentium CPU features used by the Windows 2000 memory manager. Anyone who survives this section is rewarded by extensive sample code of a memory spy device that supports the visualization of prohibited memory regions and internal memory manager data structures. Also included is a Windows 2000 memory map that outlines how the system makes use of the vast 4-GB address space offered by the Pentium CPU family.
- **Chapter 5** explains in detail how you can hook Native API functions, mainly focusing on call parameter monitoring and file/registry tracking. This chapter makes heavy use of inline assembler code and CPU stack twirling.
- **Chapter 6** proposes a general-purpose solution for something that is commonly considered impossible in the Windows 2000 programming paradigm: calling kernel-mode code from user-mode applications. The sample code in this chapter builds a bridge from the Win32 subsystem to the main kernel interfaces inside ntoskrnl.exe, hal.dll, and other core components. The chapter also describes how to call nearly any kernel function as long as its entry point is provided in the Windows 2000 symbol files.
- **Chapter 7** delves deeply into the mysterious Windows 2000 object manager. The internal structure of kernel objects is one of the best-kept secrets, because Microsoft doesn't give you more information about an object than an opaque void* pointer. This chapter unveils what this pointer really points to, and how object structures and handles are maintained and managed by the system. As a special feature, the layout of process and thread

objects is discussed in detail. The last section of the chapter is a sample application that displays the hierarchical arrangement of kernel objects by tracing the relations of various undocumented object structures.

- **Appendix A** is related to Chapter 1 and contains all commands and command options of the Windows 2000 Kernel Debugger.
- **Appendix B** is related to Chapter 2 and summarizes several API functions exported by the Windows 2000 kernel modules.
- **Appendix C** provides an extensive collection of Windows 2000 constants and data types in alphabetical order. This reference list documents several undocumented kernel structures introduced and used throughout the book.

As you see, this book discusses much information that merits the attribute "undocumented," and some of the material has never been published before.

THE AUDIENCE OF THIS BOOK

Undocumented Windows 2000 Secrets: A Programmer's Cookbook is intended for system programmers who want to maximize the features of their target operating system. First disclaimer: If the target platform of your software is Windows 95, 98, or Me (Millennium Edition), don't read any further. Because of the architectural differences of the Windows 9x/Me and NT/2000 platforms, you won't benefit from reading this book. Second disclaimer: This book does not contain information on the Alpha processor or multiprocessor systems—I will target the 32-bit Intel i386 single-processor platform exclusively. Third disclaimer: Be aware that this text is not written for the faint-hearted. You will be faced with programming techniques the average Win32 programmer has never seen. The Windows 2000 kernel is an entirely different world, bearing little resemblance to the Win32 subsystem built upon it. Some of the interfacing techniques introduced toward the end of the book might be new to even experienced kernel-mode programmers. Let me put it this way: This is the book your high-school teachers and Microsoft representatives have always warned you about!

If you are still reading on, you are obviously an open-minded, courageous person who wants to know everything about the things lurking beneath the surface of the Windows 2000 operating system. That's great! Even if you won't use the know-how you gain from this book on a daily basis, you will certainly benefit from it. Knowing what is going on under the surface of an application interface is always advantageous. It facilitates debugging and optimization, and helps avoid unwanted side effects caused by misconceptions about the hidden mechanics of the system.

The only expertise I'm expecting from my readers is "talking C" fluently and basic knowledge of Win32 programming. If you have already written kernel-mode drivers, you're in an even better position, but that's not a requirement. You will find an introduction to kernel-mode driver programming in this book, telling you everything you need to know within its scope. However, please note that this is not a comprehensive kernel-mode tutorial. If you are specifically interested in kernel-mode driver development, please get one of the good books that deal with this topic exclusively (e.g., Viscarola and Mason 1999, or Dekker and Newcomer 1999).

Some chapters of this book include heavy use of inline assembly language (ASM). I don't expect you to have thorough ASM programming experience, but a basic knowledge of ASM will be certainly helpful. If you have never written a line of ASM code, you may find these chapters difficult or you may choose to skip them entirely. However, I encourage you to read at least parts of them, only skipping the subsections that explain the details of the ASM code. Because the ASM code snippets used in the samples are always well encapsulated in C function wrappers, you can usually ignore their internals and still benefit from the remaining material that surrounds them.

THE CONVENTIONS USED IN THIS BOOK

The target operating system of this book is Windows 2000. However, you will find that most of the information also applies to Windows NT 4.0, and most of the sample applications run on this platform as well. Note that I am not covering Windows NT 3.x. Although it is probably outdated now, NT 3.51 has been my favorite NT so far because it was relatively small and fast and did not burn up too many CPU MIPS in its user interface. With respect to Windows NT 4.0, I have done everything to keep the software compatible with this version, because it is still in use. Many companies have designed their corporate networks for the classic Windows NT domain concept, and they will need time to adopt the new Active Directory paradigm introduced with Windows 2000. In some cases, it is not possible to provide common code for both operating system versions because of differences in the layout of some internal data structures. Therefore, some portions of the sample code contain version checking and separate execution paths for different versions.

Concerning terminology, when I use the term Windows 2000, it usually includes Windows NT 4.0 as well. Remember that this funny name is just a marketing gimmick to propel the sales of a system that should have been named Windows NT 5.0. In the same way, the term Windows NT without a version specification refers to the NT platform in general, including Windows 2000 and Windows NT 4.0. Note again that Windows NT 3.x is not on the list. In discussing version-specific

features, I will use the terms Windows 2000 and Windows NT 4.0 in a contrasting fashion, pointing out the respective differences.

THE SAMPLE CODE ON THE CD

The sample code included on the companion CD and partially reprinted in this book has been written for Microsoft Visual C/C++ 6.0 with Service Pack 3. If you want to rebuild or change the samples, you should also install the latest releases of the Win32 Platform Software Development Kit (SDK) and the Windows 2000 Device Driver Kit (DDK). These development kits contain the latest header file and import library updates. Be sure to set the compiler's and linker's search paths appropriately to guarantee that the SDK and DDK files are found before the header and library files installed with Visual C/C++. Both kits are distributed on CD or DVD as part of the Microsoft Developer Network (MSDN) Professional and Universal Subscriptions. If you aren't a subscriber yet, go for it! You will receive all updates of Microsoft's operating systems and development kits, plus more than 1 gigabyte of first-hand technical documentation. The subscription is somewhat expensive, but I think it is worthwhile. More information is available from Microsoft's MSDN Web site at *http://msdn.microsoft.com/subscriptions/prodinfo/overview.asp*.

The CD included with this book contains both the C source code of all samples and compiled and linked binary builds thereof for immediate use. All directories are set up as Microsoft Visual Studio 6.0 expects them: There is a base directory for each module containing the source files (C code and header files, resource scripts, definition files, project and workspace information, etc.) and a subdirectory called `release` holding the binaries (executables, object code, import libraries, etc.). *Figure P-1* outlines the overall directory structure of the CD. All source and project files are found in the `\src` tree. It contains a subtree for each sample project, plus a `common` directory for header and library files shared by them. The `\bin` directory contains all `.exe`, `.dll`, and `.sys` files of the samples, allowing you to start all applications directly from the CD. The `\tools` tree is a collection of third-party tools that I thought would be helpful for readers of this kind of book.

To rebuild a sample, simply copy the module's base directory including the `release` subdirectory to the folder where you are keeping your own projects. The base directory contains `.dsw` and `.dsp` workspace and project definition files providing build information for Visual Studio. Rebuilding is easy: Open the `.dsw` file in Visual Studio, choose the active configuration (e.g., Release) and select **Build \ Build** or **Build \ Rebuild All** from the main menu. Please note that some header files contain additional linker directives in the form of `#pragma` statements. This neat trick allows you to rebuild all samples with default Visual Studio settings—no need to enter anything into the project setting dialogs.

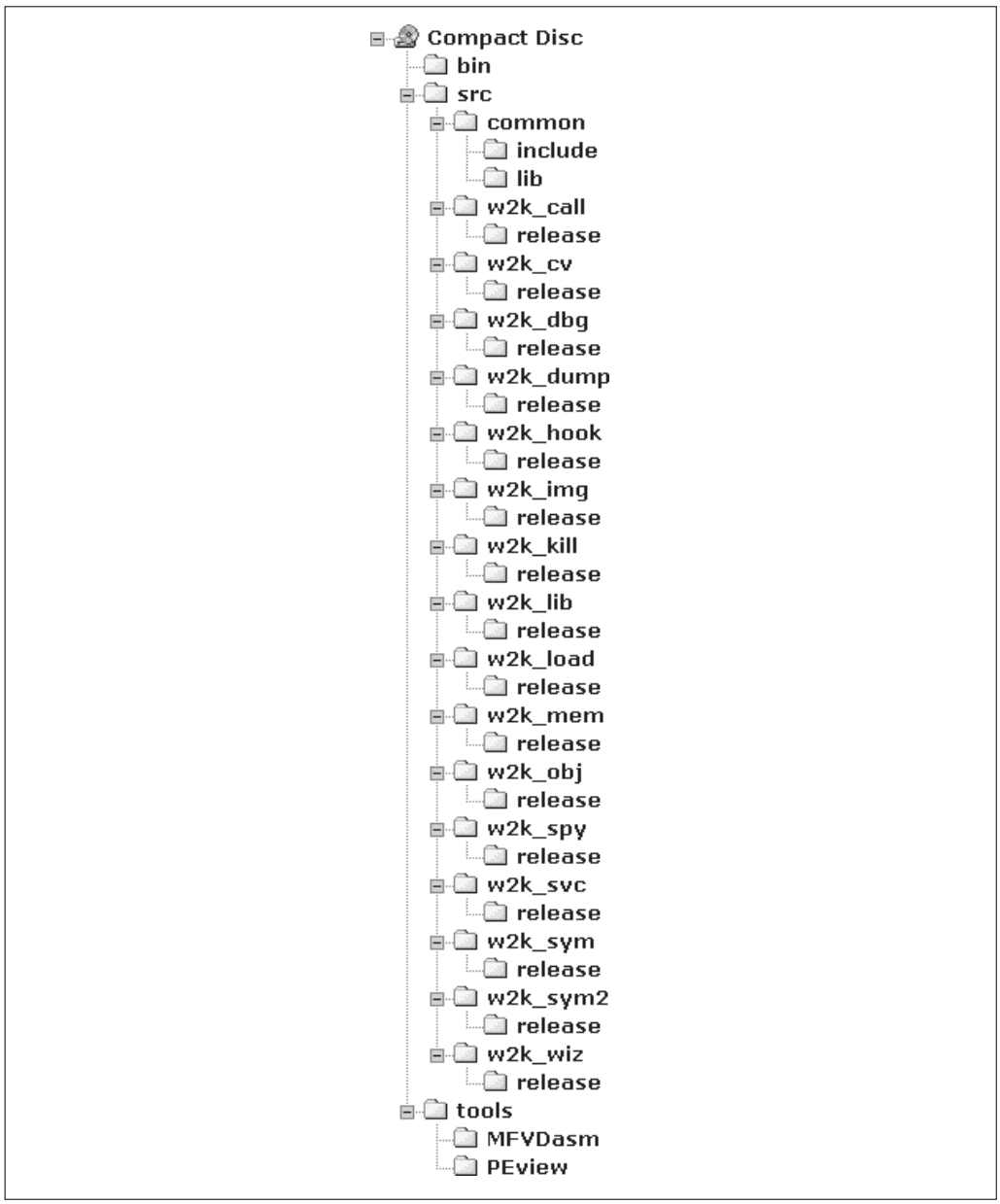

FIGURE P-1. *Directory Structure of the Sample CD*

Because most of the code is guaranteed to be `not` Windows 9x compatible, I did not provide support for ANSI characters. The native character set of Windows 2000 is Unicode, featuring 16-bit characters. Therefore, strings are defined as `WORD` arrays or pointers, except for the rare occasions that the system actually expects ANSI strings. This makes it a lot easier, doesn't it? Good Win32 programmers always try to support both ANSI and Unicode side by side to give legacy operating systems such as Windows 95 and 98 a fair chance to execute their code. This problem does not arise here—the code presented in this book usually goes well below the Win32 layer, so we can make full use of all native features of Windows 2000. The only notable exception is the `w2k_img.dll` library project and its companion application `w2k_sym2.exe`, both discussed toward the end of Chapter 1. The DLL supports both ANSI and Unicode, and the application is compiled for ANSI. This is the only sample application on the CD that runs even on Windows 95.

There are manifold dependencies between the sample projects on the CD. For example, the kernel-mode driver `w2k_spy.sys` and the utility library `w2k_lib.dll` are referenced by several applications, introduced in different chapters. *Figure P-2* outlines these dependencies. The diagram should always be read from left to right. For example, the `w2k_obj.exe` application imports API functions from `w2k_call.dll` and `w2k_lib.dll`, and `w2k_call.dll` in turn relies on `w2k_lib.dll` and `w2k_img.dll`, additionally performing Device I/O Control calls into `w2k_spy.sys`. This means that you should always place the dependent files into the same directory with `w2k_obj.exe` to be able to run this application reliably. Note that I have added `imagehlp.dll` and `psapi.dll` to the diagram, although they are Microsoft components. This is just to emphasize that the `w2k_dbg.dll` library relies on these additional DLLs, whereas the other sample programs do not.

One of the sample modules should be mentioned explicitly here: It is the "Windows 2000 Utility Library" found in the `w2k_lib` directory branch. As well as hosting large portions of the sample code reprinted in the following chapters, it also contains lots of Win32 boilerplate code not specifically related to the focus of this book that you might want to reuse in other projects. It features memory, registry, object pool, and linked-list management, CRC32 computation, pseudo–random number generation, operating system and file version checking, and much more. The huge `w2k_lib.c` file is a repository of general-purpose code I have written for myself in the past few years, and it is intended to make the life of Win32 programmers somewhat easier. Enjoy!

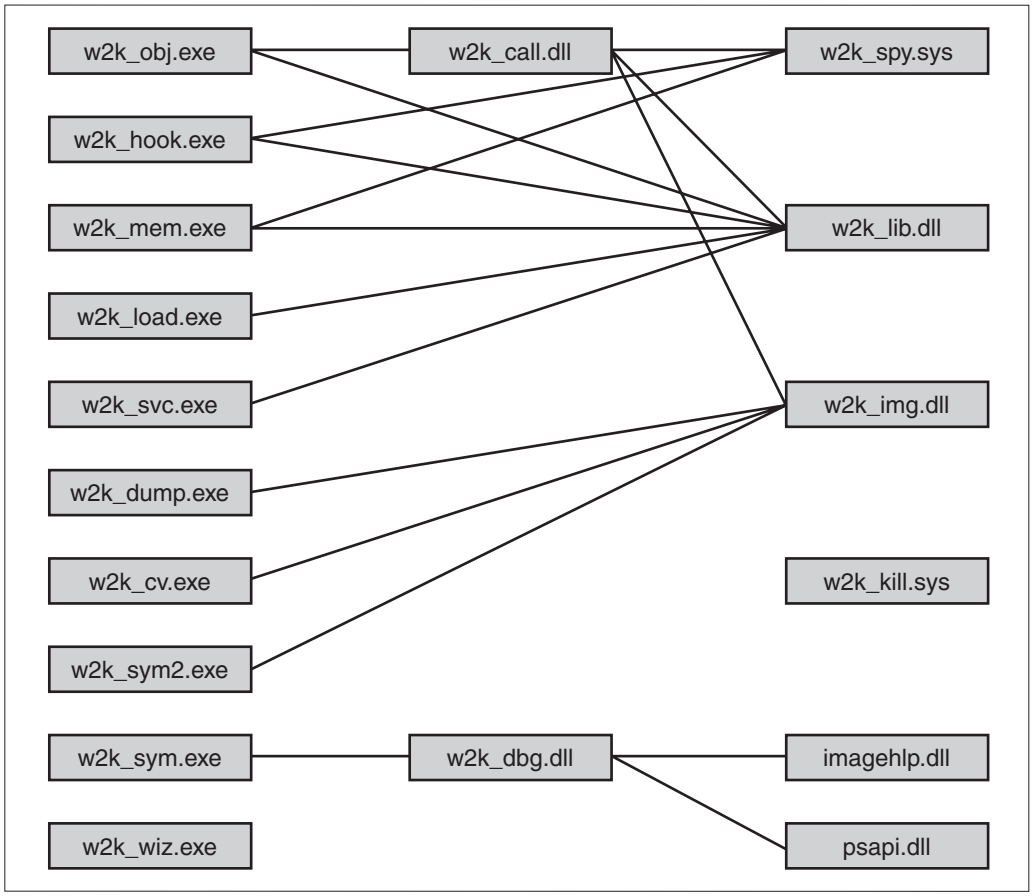

FIGURE P-2. *Dependencies of the Sample Programs*

THE THIRD-PARTY TOOLS ON THE CD

Along with the sample code I wrote exclusively for this book, the companion CD contains valuable tools contributed by others. I am indebted to Jean-Louis Seigné and Wayne J. Radburn for allowing me to distribute their finest tools on this CD. In the `\tools` directory, you will find the following goodies:

- *The Multi-Format Visual Disassembler* (MFVDasm) is written by Jean-Louis Seigné, who has been in the Windows software development business since 1990. Actually, MFVDasm is much more than just a disassembler—it is a Portable Executable (PE) file cruncher, disassembler, hex

dump utility, and ASM code browser in one. The `\tools\MFVDasm` directory on the CD contains a fully functional timed demo version, protected with the Softlocx software produced by BitArts. You can get an unlimited version by paying US$100.00 to Jean-Louis Seigné via credit card. The latest version of MFVDasm is available at *http://redirect.to/MFVDasm*; inquiries about the software should be emailed to *MFVDasm@redirect.to*.

- The *PE and COFF File Viewer* (PEview) is contributed by Wayne J. Radburn and is given royalty-free as a special bonus for the readers of this book. Wayne is a die-hard Win32 assembly language programmer like me, except that he is still doing the job while I have moved to C. Application programming in ASM is tough, so you can be sure that Wayne is a true expert. PEview is certainly the most versatile PE file browser I'm aware of, and it is an essential tool for operating system spelunkers. It provides a quick and easy way to view the structure and content of 32-bit Portable Executable (PE) and Component Object File Format (COFF) files, and it supports the viewing of `.exe, .dll, .obj, .lib, .dbg,` and other file types. Meet Wayne on the Web at *http://www.magma.ca/~wjr/*, or send email to *wjr@magma.ca*.

Because this is third-party software, included here by special license agreement with the respective authors, neither Addison-Wesley nor I take any responsibility for its usage. Please read the licensing information displayed by these programs for more details.

THE "WITHOUT WHOM" SECTION

Before writing my first book, I wasn't aware that so many people participate in a book's production process. Writing the manuscript is just one of many steps to be taken until the first printed copy appears on the shelves of the bookstores. This section is dedicated to the numerous people who have contributed substantially to the making of this book.

First, I want to thank Gary Clarke, formerly of Addison-Wesley, for putting this book project onto the track. Unfortunately, Gary left Addison-Wesley just before the first manuscript line was written, but fate brought me together again with Karen Gettman and Mary Hart, who coordinated the manuscript creation phase of my first book. This was good luck; I can't think of a better team for the birth of a new book. However, my joy halted abruptly just as I was writing the last paragraphs of Chapter 7, when I got notice that Mary left Addison-Wesley. Fortunately, Emily Frey jumped in and saved the day. I'm also indebted to Mamata Reddy, who coordinated the book's production, Curt Johnson, Jennifer Lawinski, and Chanda D. Leary-Coutu

of the marketing team, Katie Noyes for doing the cover artwork, and Beth Hayes for faithfully copyediting my raw manuscript. Several other people at Addison-Wesley whose names are unknown to me were involved in the production and marketing of this book, and I thank them all from my heart. Additionally, special thanks go to my queen of hearts Gerda B. Gradl, my parents Alla & Olaf Schreiber, and my colleague Rita Spranger for continuous encouragement and support throughout the entire project—writing a book is sometimes a lonesome job, and it is good to have someone with whom to talk about it sometimes.

And finally, a big, big "thank you" to Roine Stolt, leader of the Swedish Progressive-Rock band `The Flower Kings` (*http://www.users.wineasy.se/flowerkings/*), for his unbelievable, indescribable music, which has been an endless source of inspiration to me during manuscript writing. He always keeps on reminding me that "Stardust we are."

We believe in the light,
We believe in love,
Every precious little thing.
We believe you can still surrender,
You can serve the Flower King.

NOTE: *Excerpt from "The Flower King" by Roine Stolt, 1994. Used by kind permission.*