DECLARATION OF JUNE ANN MUNFORD

1. My name is June Ann Munford. I am over the age of 18, have personal knowledge of the facts set forth herein, and am competent to testify to the same.

2. I earned a Master of Library and Information Science (MLIS) from the University of Wisconsin-Milwaukee in 2009. I have over ten years of experience in the library/information science field. Beginning in 2004, I have served in various positions in the public library sector including Assistant Librarian, Youth Services Librarian and Library Director. I have attached my Curriculum Vitae as Appendix CV.

3. During my career in the library profession, I have been responsible for materials acquisition for multiple libraries. In that position, I have cataloged, purchased and processed incoming library works. That includes purchasing materials directly from vendors, recording publishing data from the material in question, creating detailed material records for library catalogs and physically preparing that material for circulation. In addition to my experience in acquisitions, I was also responsible for analyzing large collections of library materials, tailoring library records for optimal catalog

search performance and creating lending agreements between libraries during my time as a Library Director.

4. I am fully familiar with the catalog record creation process in the library sector. In preparing a material for public availability, a library catalog record describing that material would be created. These records are typically written in Machine Readable Catalog (herein referred to as "MARC") code and contain information such as a physical description of the material, metadata from the material's publisher, and date of library acquisition. In particular, the 008 field of the MARC record is reserved for denoting the date of creation of the library record itself. As this typically occurs during the process of preparing materials for public access, it is my experience that an item's MARC record indicates the date of an item's public availability.

5. Typically, in creating a MARC record, a librarian would gather various bits of metadata such as book title, publisher and subject headings among others and assign each value to a relevant numerical field. For example, a book's physical description is tracked in field 300 while title/attribution is tracked in field 245. The 008 field of the MARC record is reserved for denoting the creation of the library record itself. As this is the only date reflecting the inclusion of said materials within the library's collection, it is my experience

that an item's 008 field accurately indicates the date of an item's public availability.

6. I have reviewed Exhibit LG-1010, Introducing Microsoft Windows 95 by Brent Ethington.

7. Attached hereto as Appendix ETHINGTON01 is a true and correct copy of the MARC record for Introducing Microsoft Windows 95 as held by the University of California - Los Angeles Library. I secured this record myself from the library's public catalog. The MARC record contained within Appendix ETHINGTON01 accurately describes the title, author, publisher, and ISBN number of Introducing Microsoft Windows 95.

8. Attached hereto as Appendix ETHINGTON02 is a true and correct copy of Introducing Microsoft Windows 95 as held by the University of California - Los Angeles Library. These scans were secured from the library's collection. In comparing Exhibit LG-1010 to Appendix ETHINGTON02, it is my determination that Exhibit LG-1010 is a true and correct copy of Introducing Microsoft Windows 95 by Brent Ethington.

9. The 008 field of the MARC record in Appendix ETHINGTON01 indicates the date of record creation. The 008 field of Appendix ETHINGTON01

indicates the University of California - Los Angeles Library first acquired Introducing Microsoft Windows 95 as of November 2, 1994. Considering this information, it is my determination that Introducing Microsoft Windows 95 was made available to the public shortly after its initial acquisition as of November 2, 1994.

10. I have reviewed Exhibit LG-1011, "Multitrees: Enriching and Reusing Hierarchical Structure" by George W. Furnas & Jeff Zacks as published in Proceedings of the CHI '94 Conference on Human Factors in Computing Systems.

11. Attached hereto as Appendix FURNAS01 is a true and correct copy of the MARC record for Proceedings of the CHI '94 Conference on Human Factors in Computing Systems as held by the Carnegie Mellon University Library. I secured this record myself from the library's public catalog. The MARC record contained within Appendix FURNAS01 accurately describes the title, publisher and ISBN number of Proceedings of the CHI '94 Conference on Human Factors in Computing Systems.

12. Attached hereto as Appendix FURNAS02 is a true and correct copy of "Multitrees: Enriching and Reusing Hierarchical Structure" as published in

Proceedings of the CHI '94 Conference on Human Factors in Computing Systems as held by the Carnegie Mellon University Library. These scans were secured from the library's collection. In comparing Exhibit LG-1011 to Appendix FURNAS02, it is my determination that Exhibit LG-1011 is a true and correct copy of "Multitrees: Enriching and Reusing Hierarchical Structure" by George W. Furnas & Jeff Zacks as published in Proceedings of the CHI '94 Conference on Human Factors in Computing Systems.

13. The 008 field of the MARC record in Appendix FURNAS01 indicates the date of record creation. The 008 field of Appendix FURNAS01 indicates the Carnegie Mellon University Library first acquired Proceedings of the CHI '94 Conference on Human Factors in Computing Systems as of May 19, 1994. Considering this information, it is my determination that Proceedings of the CHI '94 Conference on Human Factors in Computing Systems and therefore "Multitrees: Enriching and Reusing Hierarchical Structure" was made available to the public shortly after May 19, 1994.

14. I have been retained on behalf of the Petitioner to provide assistance in the above-illustrated matter in establishing the authenticity and public availability of the documents discussed in this declaration. I am being compensated for my services in this matter at the rate of $200.00 per hour

plus reasonable expenses. My statements are objective, and my compensation does not depend on the outcome of this matter.

15. I declare under penalty of perjury that the foregoing is true and correct. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Dated: 12/19/2024



June Ann Munford

Appendix CV

June A. Munford
Curriculum Vitae

Education

University of Wisconsin-Milwaukee - MS, Library & Information Science, 2009
Milwaukee, WI

- Coursework included cataloging, metadata, data analysis, library systems, management strategies and collection development.
- Specialized in library advocacy, cataloging and public administration.

Grand Valley State University - BA, English Language & Literature, 2008
Allendale, MI

- Coursework included linguistics, documentation and literary analysis.
- Minor in political science with a focus in local-level economics and government.

Professional Experience

Researcher / Expert Witness, October 2017 – present
Freelance ● Pittsburgh, Pennsylvania & Grand Rapids, Michigan

- Material authentication and public accessibility determination. Declarations of authenticity and/or public accessibility provided upon research completion. Experienced with appeals and deposition process.

- Research provided on topics of public library operations, material publication history, digital database services and legacy web resources.

- Past clients include Alston & Bird, Arnold & Porter, Baker Botts, Fish & Richardson, Erise IP, Irell & Manella, O'Melveny & Myers, Perkins-Coie, Pillsbury Winthrop Shaw Pittman and Slayden Grubert Beard.

Library Director, February 2013 - March 2015
Dowagiac District Library ● Dowagiac, Michigan

- Executive administrator of the Dowagiac District Library. Located in

Southwest Michigan, this library has a service area of 13,000, an annual operating budget of over $400,000 and total assets of approximately $1,300,000.

● Developed careful budgeting guidelines to produce a 15% surplus during the 2013-2014 & 2014-2015 fiscal years while being audited.

● Using this budget surplus, oversaw significant library investments including the purchase of property for a future building site, demolition of existing buildings and building renovation projects on the current facility.

● Led the organization and digitization of the library's archival records.

● Served as the public representative for the library, developing business relationships with local school, museum and tribal government entities.

● Developed an objective-based analysis system for measuring library services - including a full collection analysis of the library's 50,000+ circulating items and their records.

November 2010 - January 2013
Librarian & Branch Manager, Anchorage Public Library ● Anchorage, Alaska

● Headed the 2013 Anchorage Reads community reading campaign including event planning, staging public performances and creating marketing materials for mass distribution.

● Co-led the social media department of the library's marketing team, drafting social media guidelines, creating original content and instituting long-term planning via content calendars.

● Developed business relationships with The Boys & Girls Club, Anchorage School District and the US Army to establish summer reading programs for children.

June 2004 - September 2005, September 2006 - October 2013
Library Assistant, Hart Area Public Library
Hart, MI

● Responsible for verifying imported MARC records and original MARC

cataloging for the local-level collection as well as the Michigan Electronic Library.

- Handled OCLC Worldcat interlibrary loan requests & fulfillment via ongoing communication with lending libraries.

Professional Involvement

Alaska Library Association - Anchorage Chapter
- Treasurer, 2012

Library Of Michigan
- Level VII Certification, 2008
- Level II Certification, 2013

Michigan Library Association Annual Conference 2014
- New Directors Conference Panel Member

Southwest Michigan Library Cooperative
- Represented the Dowagiac District Library, 2013-2015

Professional Development

Library Of Michigan Beginning Workshop, May 2008
Petoskey, MI
- Received training in cataloging, local history, collection management, children's literacy and reference service.

Public Library Association Intensive Library Management Training, October 2011
Nashville, TN
- Attended a five-day workshop focused on strategic planning, staff management, statistical analysis, collections and cataloging theory.

Alaska Library Association Annual Conference 2012 - Fairbanks, February 2012
Fairbanks, AK
- Attended seminars on EBSCO advanced search methods, budgeting, cataloging, database usage and marketing.

Depositions

2019 ● Fish & Richardson

Apple v. Qualcomm (IPR2018-001281, 39521-00421IP, IPR2018-01282 and 39521-00421IP2)

2019 ● Erise IP

Implicit, LLC v. Netscout Systems, Inc (Civil Action No. 2:18-cv-53-JRG)

2019 ● Perkins-Coie

Adobe Inc. v. RAH Color Technologies LLC (Cases IPR2019-00627, IPR2019-00628, IPR2019-00629 and IPR2019-00646)

2020 ● O'Melveny & Myers

Maxell, Ltd. v. Apple Inc. (Case No. 5:19-cv-00036-RWS)

2021 ● Pillsbury Winthrop Shaw Pittman LLP

Intel v. SRC (IPR2020-1449)

2022 ● Perkins-Coie

Realtek v. Future Link (IPR2021-01182)

2023 ● Fish & Richardson

Neuroderm Ltd. v. Abbvie, Inc (Case No. PGR2022-00040)

2023 ● Fish & Richardson

Nearmap US Inc. v. Pictometry International Corp. (IPR2022-00735)

2023 ● Fish & Richardson

Samsung Electronics v. MemoryWeb LLC (Case No. 39843-0136PS1)

2023 ● Pillsbury Winthrop Shaw Whitman LLP

Gravel Rating Systems v. Costco Wholesale Corp. (Civil Action No. 4:21-cv-149-ALM)

2024 ● Willkie-Farr

Netflix, Inc. v. VideoLabs. Inc. (IPR2023-00628)

Limited Case and Clientele History

Alston & Bird

- Ericsson

  v. Collision Communications (IPR2022-01233)

- Nokia

  v. Neptune Subsea, Xtera (Case No. 1:17-cv-01876)

- Universal Electronics Inc

  v. Roku Inc (IPR2022-00818)

Arnold & Porter

- Ivantis

  v. Glaukos (Case No. 8:18-cv-00620)

- Samsung

  v. Jawbone (Case No. 2:21-cv-00186)

Benesch Friedlander Coplan & Aronoff

- Voyis

  v. Cathx (Case No. 5:21-cv-00077-RWS)

Deschert LLP

- Smaxtec Animal Care

  v. ST Reproductive Technologies, LLC (IPR2024-00885)

Erise I.P.

- Apple

  v. Ericsson Inc. (IPR2022-00715)

  v. Future Link Systems (IPRs 6317804, 6622108, 6807505, and 7917680)

  v. INVT (Case No. 20-1881)

  v. Navblazer LLC (IPR2020-01253)

  v. Qualcomm (IPR2018-001281, 39521-00421IP, IPR2018-01282, 39521-00421IP2)

  v. Quest Nettech Corp (Case No. 2:19-cv-00118-JRG)

  v. Telefonaktiebolaget LM Ericsson (IPR2022-00275)

  v. Theta IP, LLC (IPR2024-00818)

- Fanduel

  v. CGT (Case No. 19-1393)

- Garmin

  v. Phillips North America LLC (Case No. 2:19-cv-6301-AB-KS)

- Netscout
  - v. Longhorn HD LLC (Case No. 2:20-cv-00349)
  - v. Implicit, LLC (Civil Action No. 2:18-cv-53-JRG)

- Sony Interactive Entertainment LLC
  - v. Bot M8 LLC (IPR2020-01288)
  - v. Infernal Technology LLC (Case No. 2:19-CV-00248-JRG)

- Unified Patents
  - v. GE Video Compression (Civil Action No. 2:19-cv-248)

Fish & Richardson

- Apple
  - v. AliveCor (3:21-cv-03958)
  - v. LBS Innovations (Case No. 2:19-cv-00119-JRG-RSP)
  - v. Koss Corporation (IPR2021-00305)
  - v. Masimo (IPR 50095-0012IP1, 50095-0012IP2, 50095-0013IP1, 50095-0013IP2, 50095-0006IP1, 50095-0135IP1)
  - v. Neonode (Case No. 21-cv-08872-EMC)
  - v. Qualcomm (IPR2018-001281, 39521-00421IP, IPR2018-01282, 39521-00421IP2)

- Dell
  - v. Neo Wireless (IPR2022-00616)

- Dish Network
  - v. Entropic Communications, LLC (Case No. 2:2023-CV-01043)

v. Realtime Adaptive Streaming (Case No. 1:17-CV-02097-RBJ)

v. TQ Delta LLC (Case No. 18-1798)

- Evapco Dry Cooling

  v. SPG Dry Cooling (IPR2021-00688)

- Genetec

  v. Sensormatic Electronics (Case No. 1:20-CV-00760)

- Huawei

  v. Bell Northern Research LLC (IPR2019-01174)

- Kianxis

  v. Blue Yonder (Case No. 3:20-cv-03636)

- LG Electronics

  v. Bell Northern Research LLC (Case No. 3:18-cv-2864-CAB-BLM)

- Metaswitch

  v. Sonus Networks (IPR2018-01719)

- Microsoft

  v. Throughputer Inc (IPR2022-00757)

- Mom Enterprises

  v. Ddrops Company (Case No. 1:22-cv-00332)

- MLC Intellectual Property

v. MicronTech (Case No. 3:14-cv-03657-SI)

- Nearmap Inc
  - v. EagleView Technologies (IPR2022-01009)

- Neuroderm Ltd.
  - v. Abbvie, Inc (Case No. PGR2022-00040)

- Realtek Semiconductor
  - v. Future Link (IPR2021-01182)

- Quectel
  - v. Koninklijke Philips (Case No. 1:20-cv-01710)

- Samsung
  - v. Aire Technology (IPR2022-00877)
  - v. Bell Northern Research (Case No. 2:19-cv-00286-JRG)
  - v. Communication Technologies Inc (IPR2022-01221)
  - v. Jawbone Innovations (IPR2022-00865)
  - v. MemoryWeb LLC (IPR2022-00885)
  - v. Dodots Licensing Solutions (IPR2023-00701)
  - v. Telefonaktiebolaget LM Ericsson (IPR2021-00615)

- Texas Instruments
  - v. Vantage Micro LLC (IPR2020-01261)

- Xilinx
  - v. Sentient Sensors LCC (Case No. 1:22-cv-00173)

Groombridge Wu

- Nearmap

  v. Eagle View Technologies Inc. and Pictometry International Corp. (Case No. 2:21-cv-00283-TS-DAO)

Irell & Manella

- Curium

O'Melveny & Myers

- Apple

  v. Maxell (Case No. 5:19-cv-00036-RWS)

- Disney Media & Entertainment Distribution LLC

  v. Digital Media Technology Holdings, LLC (Case No. 2:22-cv-01642)

- Micron Technology Inc.

  v. Besang Inc (IPR2023-00900)

- Samsung

  v. Daedalus Prime LLC (1335 Investigation)

Perkins-Coie

- Amazon

  v. Nokia (IPR2024-00847 and IPR2024-00848)

- Heru Industries

  v. The UAB Research Foundation (IPR2022-01148)

- Intel Corporation

  v. BeSang Inc. (IPR2023-00991)

  v. BitMICRO (Case No. 5:23-cv-00625)

  v. Telefonaktiebolaget L M Ericsson (IPR2024-00610)

- Realtek Semiconductor

  v. Future Link (IPR2021-01182)

- Twitter Inc

  v. VOIP-Pal.com (Case No. 3:20-cv-02397-JD)

- TCL Industries

  v. Koninklijke Philips NV (IPR2021-00495, IPR2021-00496 and IPR2021-00497)

Pillsbury Winthrop Shaw Pittman

- Intel

  v. FG SRC LLC (Case No. 6:20-cv-00315)

- Gravel Rating Systems

  v. Costco (Case No. 4:21-cv-149)
  v. Lowe's Home Centers (Case No. 4:21-cv-150)
  v. T-Mobile USA (Case No. 4:21-cv-152)
  v. Kohl's Inc. (Case No. 4:21-cv-258)
  v. Under Armor (Case No. 4:21-cv-356)

Quinn Emanuel

- ServiceNow, Inc.

  v. InQuisient, Inc. (Case No. 22-900-CJB)

- Wolfspeed

  v. The Trustees of Purdue University (Case No. 1:2021-cv-00840)

Sheppard, Mullin, Richter & Hampton LLP

- Cadence Design Systems Inc.

  v. Semiconductor Design Technologies (Case No. 3:23-cv-01001)

Stern, Kessler, Goldstein & Fox

- Preliminary Research

Willkie, Farr & Gallagher LLP

- Neurent

  v. The Foundry, LLC (IPR2024-00669)

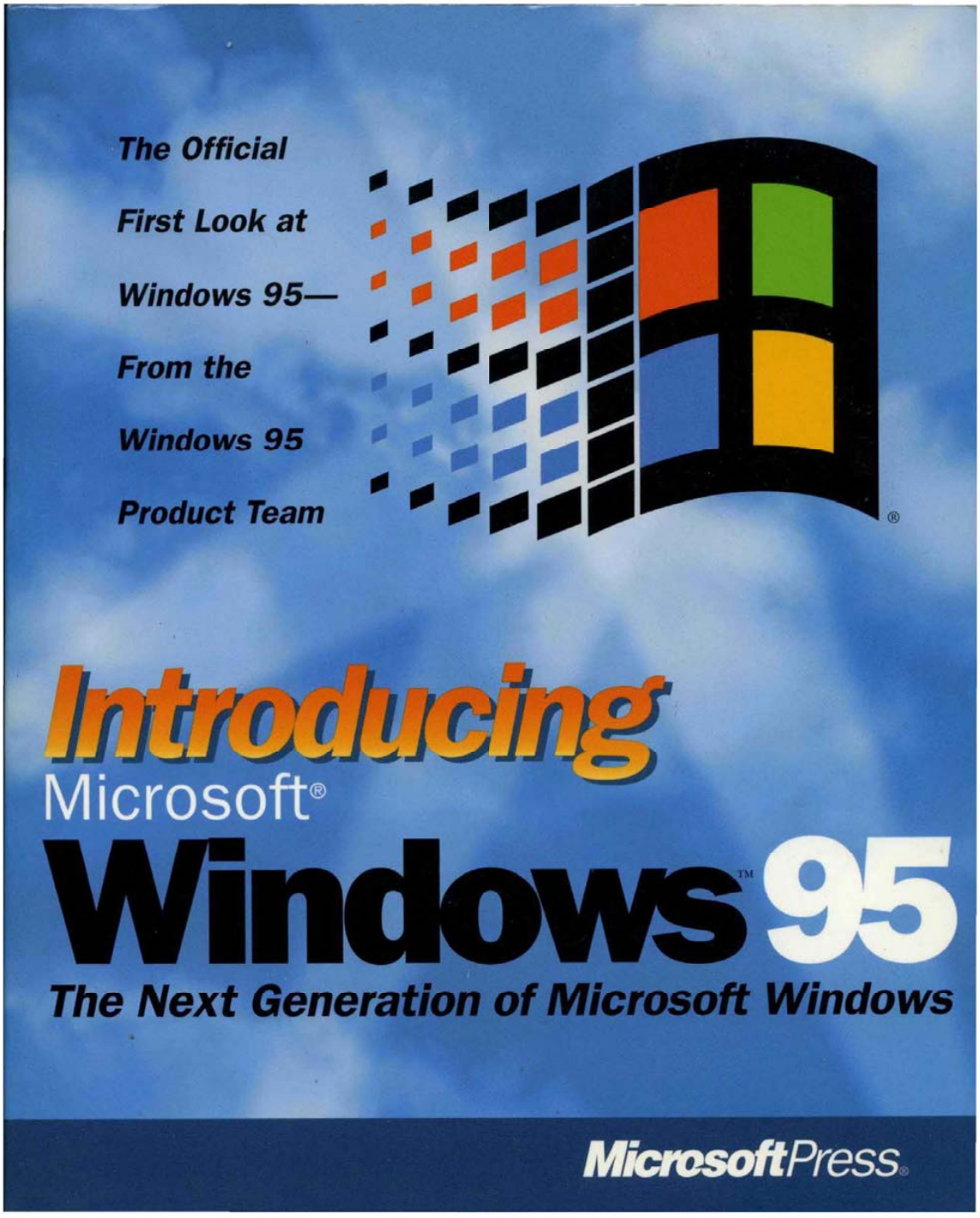
- Netflix, Inc.

  v. VideoLabs, Inc. (IPR2023-00628)

Appendix ETHINGTON01

```
leader  02653cam a2200481 a 4500
001     9912804090706531
005     20241114213017.0
008     941102s1995    waua          001 0 eng
010     ##$a   94041890
020     ##$a1556158602 $q(paperback)
020     ##$a9781556158605 $q(paperback)
035     ##$a(CLU)1167175-ucladb-Voyager
035     ##$a(OCoLC)31606209
035     ##$a(OCoLC)ocm31606209
040     ##$aDLC $beng $cDLC $dXDM $dOCLCQ $dNLGGC $dYDXCP $dOCLCO $dOCLCA $dOCLCF $dOCLCQ $dOCLCA $dOCLCO $dOCLCQ $dOCLCL $dOCLCQ $dOCLCL $dOCLCO
049     ##$aMAIN
050     00$aQA76.76.O63 $bE87 1995
082     00$a005.4/469 $220
084     ##$a54.54 $2bcl
100     1#$aEthington, Brent.
245     10$aIntroducing Microsoft Windows 95 / $cBrent Ethington (and the Microsoft Windows 95 Team).
260     ##$aRedmond, WA : $bMicrosoft Press, $c©1995.
300     ##$axxiv, 344 pages : $billustrations ; $c23 cm
336     ##$atext $btxt $2rdacontent
337     ##$aunmediated $bn $2rdamedia
338     ##$avolume $bnc $2rdacarrier
500     ##$aIncludes index.
505     0#$aThe Microsoft windows operating system family -- An overview of windows 95 -- The windows 95 user interface -- Base system architecture -- Robustness -- Support for running MS-DOS-based applications -- Plug and play -- Device support
599     ##$aNEW $bUclaCollMgr $bMonosBKS19951998 $bNew collection customer.207.15 $9LOCAL
650     #0$aOperating systems (Computers) $0http://id.loc.gov/authorities/subjects/sh85094982
650     #6$aSystèmes d'exploitation (Ordinateurs)
650     #7$aoperating systems. $2aat
650     #7$aOperating systems (Computers) $2fast
650     17$aMicrosoft Windows 95. $2gtt
630     00$aMicrosoft Windows (Computer file)
630     07$aMicrosoft Windows (Computer file) $2fast
710     2#$aMicrosoft Corporation. $1https://id.oclc.org/worldcat/entity/E39QH7JmqqxpQpQVq7hHYMXxPF $0http://id.loc.gov/authorities/names/n86810571
758     ##$ihas work: $aIntroducing Microsoft Windows 95 (Text) $1https://id.oclc.org/worldcat/entity/E39PD33rXYhmV36CMyckyYdgfC $4https://id.oclc.org/worldcat/ontology/hasWork
908     ##$aWorldCat Daily Updates 2022-07-02 $bWorldCat record variable field(s) change: 040, 650
908     ##$aWorldCat Daily Updates 2023-10-24 $bWorldCat record variable field(s) change: 020, 040
908     ##$aWorldCat Daily Updates 2024-11-14 $bWorldCat record variable field(s) change: 040, 630, 650
951     ##$aUclaCollMgr MonosBKS19951998 20210426 $9LOCAL
951     ##$aa=ma 960705/1 $a13 Dec 95 fk $9LOCAL
951     ##$aMARS $9LOCAL
992     ##$aMCS636286 $bIP1381195 $9LOCAL
992     ##$c07-ADQ-6863 $9LOCAL
996     ##$a1167175-ucladb $9LOCAL
```

Appendix ETHINGTON02


The Official
First Look at
Windows 95—
From the
Windows 95
Product Team
Introducing
Microsoft®
Windows™ 95
The Next Generation of Microsoft Windows
Microsoft Press®

Introducing Microsoft Windows 95

Brent Ethington

(and the Microsoft Windows 95 Team)

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Copyright © 1995 by Microsoft Corporation

All rights reserved. No part of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

Library of Congress Cataloging-in-Publication Data
Introducing Microsoft Windows 95 / Microsoft Corporation.
p. cm.
Includes index.
ISBN 1-55615-860-2
1. Operating systems (computers) 2. Microsoft Windows 95
I. Microsoft Corporation.
QA76.76.O63I64 1994
005.4 ' 469--dc20 94-41890
CIP

Printed and bound in the United States of America.

5 6 7 8 9 QMQM 0 9 8 7 6 5

Distributed to the book trade in Canada by Macmillan of Canada, a division of Canada Publishing Corporation.

A CIP catalogue record for this book is available from the British Library.

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office. Or contact Microsoft Press International directly at fax (206) 936-7329.

Adaptec is a trademark of Adaptec Inc. Adobe is a registered trademark of Adobe Systems, Inc. AT&T is a registered trademark American Telephone and Telegraph Company. Apple, Macintosh, and TrueType are registered trademarks of Apple Computer, Inc. Artisoft and LANtastic are registered trademarks of Artisoft, Inc. Banyan and VINES are registered trademarks of Banyan Systems, Inc. Quattro Pro is a trademark of Borland International, Inc. COMPAQ and Qvision are registered trademarks of Compaq Computer Corporation. CompuServe is a registered trademark of CompuServe, Inc. Clipper is a trademark of Computer Associates International, Inc. CorelDRAW is a registered trademark of Corel Systems Corporation. Alpha AXP, DEC, and PATHWORKS are trademarks of Digital Equipment Corporation. TrueSpeech is a trademark of DSP Group, Inc. Kodak is a registered trademark of Eastman Kodak Company. Future Domain is a registered trademark of Future Domain Corporation. LaserJet is a registered trademark of Hewlett-Packard Company. Doom is a trademark of ID Software. Intel is a registered trademark and i386, i486, Indeo, and Pentium are trademarks of Intel Corporation. IBM is a registered trademark and PowerPC is a trademark of International Business Machines Corporation. Toshiba is a registered trademark of Kabushiki Kaisha Toshiba. 1-2-3, Freelance, Lotus, and Lotus Notes are registered trademarks of Lotus Development Corporation. Microsoft, MS-DOS, Multiplan, PowerPoint, and Win32 are registered trademarks and DriveSpace, Microsoft At Work, Windows, and Windows NT are trademarks of Microsoft Corporation. MIPS is a registered trademark of MIPS Computer Systems, Inc. NetWare and Novell are registered trademarks of Novell, Inc. UNIX is a registered trademark of Novell, Inc. in the U.S.A. and other countries, licensed exclusively through X/Open Company, Ltd. America Online is a registered trademark of Quantum Computer Services, Inc. Shiva is a registered trademark of Shiva Microsystems Corporation. Silicon Graphics is a registered trademark of Silicon Graphics, Inc. Sun is a registered trademark of Sun Microsystems, Incorporated. Western Digital is a trademark of Western Digital Corporation. WordPerfect is a registered trademark of WordPerfect Corporation.

Contents

Acknowledgments ..... **xvii**

Foreword ..... **xix**

Introduction ..... **xxi**
Where We've Been xxi
Where We Are Today xxi
Where We're Headed xxii
How We Get There xxiii

Chapter 1 The Microsoft Windows Operating System Family **1**
Windows 95 and Windows NT Workstation 1
Which Operating System? 3
Windows 95 Usage Scenarios 3
Windows NT Workstation Usage Scenarios 4
Evaluating Windows 95 5
Ease of Use 5
Performance 6
Compatibility of Device and Application Support 7
Support for Networking and Connectivity 7
Support for Manageability and Administration 8
Support for Communications and Messaging 9
Support for Mobile Services and Remote Access 9

Chapter 2 An Overview of Windows 95 ..... **11**
Key New Features 12
Ease of Use 12
Speed and Power 13
Compatibility 15

Chapter 3 The Windows 95 User Interface **17**
Designing the Windows 95 User Interface 17
Design Methodology 18
Putting New Designs to the Test 19

Easy to Learn 20
The Desktop: Neat, Clean, and Logical....20
The Taskbar: Home Base 22
My Computer: An Easier Model for File Management and Browsing....25
Long Filenames: Greater Flexibility When Naming Files 27
The Network Neighborhood: Accessing Networking Features 28
The Recycle Bin: Easy Deleting and Undeleting of Files....30
Focus on Documents: Working the Way Users Work 30
Backtracking: Undoing File Operations....31
Wizards: Guides to Powerful Capabilities 32
A New Help Engine: Accessible and Useful Online Information 33
Transition Aids: Easy Migration to the Windows 95 User Interface....35
Powerful Features 35
The Windows Explorer: For File Management and Information Browsing 36
Shortcuts: For Accessing Objects....37
Properties: For Customizing All Objects 39
Right-Clicking: For Performing Actions on Objects 40
The Control Panel: The Consolidated Control Center 42
Finding Files or Folders: Easy and Efficient....44
The Printers Folder: Consolidated Printer Control 45
Font Settings: More Powerful Font Management and Preview 46
Quick Viewers: Examining Files Without Opening Them 47
Compatibility 48
For Users of Windows 3.1....48
For Users of MS-DOS 48

Chapter 4 Base System Architecture....51

Summary of Improvements over Windows 3.1 51
A Fully Integrated Operating System 52
A Preemptive Multitasking Operating System....52
No CONFIG.SYS or AUTOEXEC.BAT? 53
No MS-DOS? 54
32-Bit Versus 16-Bit Components....54
Virtual Device Drivers....55
The System Architecture Layout in Windows 95....57

Support for Win16–Based Applications 58
Compatibility 58
Size 58
Performance 59
Protection 59
Robustness Improvements 60
Support for MS-DOS–Based Applications 60
Protection 61
Robustness Improvements 61
Support for Win32–Based Applications 61
Preemptive Multitasking 62
Separate Message Queues 62
Flat Address Space 62
Compatibility with Windows NT 63
Long Filename Support 63
Memory Protection 63
Robustness Improvements 63
32-Bit File System Architecture 64
Architecture Overview 64
The Installable File System Manager 65
File System Drivers 66
The Block I/O Subsystem 68
Long Filename Support 70
Hidden File Extensions 71
Additional File Date/Time Attributes 71
Coordinated Universal Time Format 72
Exclusive Access for Disk Management Tools 72
DriveSpace Disk Compression 73
System Capacity Improvements 74
System Resource Limitation Improvements 74
Better Memory Management 76
Linear Memory Addressing for Win32–Based Applications 76
Compatibility with the Windows NT Memory Model 76
Virtual Memory Support (Swapfile) Improvements 77

The Registry: A Centralized Configuration Store 78
Solutions to .INI Problems 78
No .INI Files? 80
Role in Plug and Play 80
Remote Access to Registry Information 80
Better Font Support 80
The 32-Bit TrueType Rasterizer 81

Chapter 5 Robustness 83

System-Wide Robustness Improvements 83
Better Local Reboot 84
Virtual Device Driver Thread Clean-Up 85
Per-Thread State Tracking 85
Virtual Device Driver Parameter Validation 86
Robustness for MS-DOS–Based Applications 86
Virtual Machines Protection Improvements 86
Better Clean-Up When a Virtual Machine Ends 87
Robustness for Win16–Based Applications 87
Per-Thread State Tracking 87
Parameter Validation for Win16 APIs 88
Robustness for Win32–Based Applications 88
A Private Address Space for Each Win32–Based Application 89
Parameter Validation for Win32 APIs 89
Per-Thread Resource Tracking 89
Separate Message Queues for Win32–Based Applications 89
Local Reboot Effectiveness 91
Structured Exception Handling 91

Chapter 6 Support for Running MS-DOS–Based Applications 93

Summary of Improvements over Windows 3.1 93
Zero Conventional Memory Footprint Components 94
Compatibility Improvements 96
Single MS-DOS Application Mode 97
Support for Graphic-Intensive MS-DOS–Based Applications 98
Memory Protection 98
Better Defaults for Running MS-DOS–Based Applications 98

Consolidated Customization of MS-DOS–Based Application Properties 99
Toolbars in MS-DOS Windows 100
Scaleable MS-DOS Windows 101
Ending MS-DOS–Based Applications Gracefully 102
Local Virtual Machine Environment Settings 103
Support for Accessing Network Resources with UNC Pathnames 104
New MS-DOS Prompt Commands 104
Support for Long Filenames 105

Chapter 7 Plug and Play 107
The Problem with Today's PCs 108
Higher Flexibility Requirements for Mobile Computers 109
The Plug and Play Solution 109
Plug and Play Support in Windows 95 111
The Benefits of Plug and Play with Windows 95 112
Hardware Design Guide for Microsoft Windows 95 113
The Device Manager 114

Chapter 8 Device Support 117
Device Driver Philosophy 117
Disk Device Support 118
Support for IDE Drives and Controllers 119
Support for SCSI Devices and Controllers 119
Support for ESDI Controllers 120
The High-Speed Floppy Disk Driver 120
Display Adapter and Monitor Support 120
Summary of Improvements over Windows 3.1 121
Driver Stability and Reliability Improvements 121
Video Display Performance Improvements 122
Support for More Video Display Adapters Than Windows 3.1 122
Robustness Improvements 123
New Control Panel Enhancements and Customization Properties 123
Image Color Matching Support 124
Energy Star Monitor Support 124

Mouse and Pointing Device Support 126
Summary of Improvements over Windows 3.1 126
Windows Mouse Driver Improvements 126
Mouse Control Panel Enhancements 127

Chapter 9 Networking 129

Summary of Improvements over Windows 3.1 and Windows for Workgroups 3.11 130
Easier Networking with Windows 95 130
Great Novell NetWare Integration 131
The "Well-Connected Client" Operating System 131
Internet Information with a Mouse Click 132
"Point and Click" Networking 133
Easier Mobile Network Support 133
Windows 95 Client: Designed for Manageability 134
Network Architecture in Windows 95 137
Network Provider Interface: Concurrent Support for Multiple Network Servers 138
Installable File System: Support for Multiple Network Redirectors 140
NDIS 3.1: Multiple Protocol Support 141
Novell NetWare Integration 142
The 32-Bit Microsoft Client for NetWare Networks 142
File and Printer Sharing for NetWare Networks 144
Other NetWare Interoperability 147
Microsoft Network Integration 147
The 32-Bit Client for Microsoft Networks 148
The 32-Bit Microsoft Network Peer Services 148
Network Compatibility 149
Protocol Support 150
The IPX/SPX-Compatible Protocol 150
The TCP/IP Protocol 150
Network Interprocess Communications Interfaces 153
Long Filename Support 153
Network Printing 153
Network Security 154
Password Control: Unified Logon 154
User-Level Security 155
Dial-Up Server Remote Access Gateway 157

Chapter 10 Systems Management 159
The Registry 160
User Management 164
User Profiles 164
System Policies 165
Registry Tools 169
The Role of the Server in Systems Management 170
System Management 170
The Windows 95 Tools 171
Performance Monitoring 172
Network Management 173
Server-Based Backup 173
Network Management Tools 175
The Windows 95 Tools 176

Chapter 11 Printing 177
Summary of Improvements over Windows 3.1 177
The 32-Bit Print Subsystem 178
The 32-Bit Preemptive Spooler 178
Support for MS-DOS–Based Applications 181
Support for Deferred Printing 182
Image Color Matching Support 182
Installing and Configuring a Printer 183
Easy Printer Setup 183
Configuring a Printer 184
Managing Print Jobs 185
Network Printing Improvements 186
Plug and Play Support 186

Chapter 12 Communications 189
Summary of Improvements over Windows 3.1 189
The Communications Architecture 190
Communications Goals of Windows 95 190
Kernel Improvements 191
Driver Architecture 192
The Telephony API (TAPI) 195
Sharing Communications Devices 197

Centralized Modem Setup and Configuration 197
Modem Configuration in Windows 3.1 198
Modem Configuration in Windows 95 198
Device/Hardware Support 201
Support for 16550A UART FIFO 201
Support for More Ports 202
Support for Future Parallel Port Modems 202
Plug and Play Support 202
Modems 202
HyperTerminal 203
The Phone Dialer 204

Chapter 13 Mobile Computing Services 205

The Windows 95 Vision of Mobile Computing 205
Staying Connected 207
Dial-Up Networking 207
Dynamic Networking 211
Password Management 211
Moving to and from the Desk 212
Hot-Docking Support 212
PCMCIA Support 213
Power Management 214
Flexible Video Resolution Support 214
Pointing Devices 215
Dealing with the Mobile Environment 215
The Briefcase 215
Microsoft At Work Fax 217
"Local" Connections 217
Document Viewers 218
Deferred Printing 218
Remote Mail 218

Chapter 14 Microsoft Exchange: E-Mail, Faxes, and More 221

The Windows Messaging Subsystem 222
An Open Architecture for Open Connectivity 224

Microsoft Exchange 225
Summary of Improvements over Current Microsoft Mail 225
Working with Microsoft Exchange 226
Rich-Text Mail Messages 228
The Personal Address Book 229
The Information Stores 230
Remote Mail 232
The Microsoft Mail Post Office 233
The Microsoft Internet Mail Drivers 234
Microsoft At Work Fax 235
Working with Microsoft At Work Fax 237
Rich Messaging Capabilities 237
Workgroup Fax Features for MIS Organizations 238
Easy Access to Fax Information Services 239
The Fax Viewer and Fax Cover Page Designer 240
Secure Faxing with Encryption and Digital Signatures 240
Compatibility with Fax Modems and Fax Machines 241
Coexistence with Windows–Based Telecommunications Applications 242
Integration of Fax and Applications 242

Chapter 15 The Microsoft Network 243

Easy to Get Started 243
Easy to Use and Understand 244
Highlights 244
Shortcuts 244
E-Mail 244
Multitasking 245
Worldwide Access 245
Getting Started with MSN 245

Chapter 16 Multimedia Services 247

A Little History 248
A New High-Performance Multimedia Platform 249
Making Multimedia Easier 250
Plug and Play Support 250
AutoPlay 251
Built-In Support for Digital Video 252

Built-In Support for Sound and MIDI 252
The CD Player 253
Making Multimedia More Engaging 253
Built-In CD+ Support 253
Bigger, Faster, Better-Looking 32-Bit Digital Video Playback 254
Multitasking and Threading 255
Built-In Support for Fast CD-ROMs 255
Hardware Support for TV-Like Video 256
Making Windows More Fun 256
Fast DIB Drawing 256
A Powerful Development Environment 257
Sound Compression for CD-Quality Sound 258
Polymessage MIDI Support for Better Sound 258
Multitasking 258
Professional Quality 259
Capture and Compression of Bigger Digital Video 259
General MIDI for Specific Sounds 259
Built-In Support for Multimedia Devices 259
Multimedia PCs for 1995 260
New Opportunities for Great-Sounding Audio 260
Taking Advantage of New Video Card Features 261
The Multimedia Architecture 262
The Multimedia Graphics Architecture 262
Multimedia Data Routing 263

Chapter 17 Installation and Setup 265

Summary of Improvements over Windows 3.1 265
A Modular Setup Architecture 266
A GUI-Based Setup Program 266
The Leveraging of Detection Code 267
Customization Improvements 267
Hardware Detection Improvements 267
The Setup Process 268
The Hardware Detection Phase 268
The Configuration Questions Phase 269
The Copying Files Phase 269
The Final System Configuration Phase 269
Better Control over Installed Components 270
Setup's Smart Recovery Mechanism 270

Built-In Verification of System Files 270
Network Setup Improvements 271
Network Installation Location Remembered 271
Batch Installation Support 271
Windows or Windows for Workgroups Configuration Preserved 272

Chapter 18 International Language Support 273

Summary of Improvements over Windows 3.1 273
The Localization of Windows 95 275
International Language Issues 275
Date and Time Formats 276
Sorting and Searching 277
National Character Sets, Keyboards, and Fonts 278
The Solution: Multilingual Content Support 278
Easy Switching Among Languages and Keyboards 279
Multilingual Extensions to the ChooseFont Dialog Box 280
Multilingual Support for Exchanging Information via the Clipboard 281
The Win32 National Language Support APIs 282

Chapter 19 Accessibility 283

Summary of Improvements over Windows 3.1 284
General Features of Accessibility Enhancement 285
Online Help 285
Controlling the Accessibility Features 285
Emergency Hotkeys 286
The Accessibility TimeOut 287
The Accessibility Status Indicator 287
Features for Users with Low Vision 287
Scaleable User Interface Elements 287
A Customizable Mouse Pointer 287
High-Contrast Color Schemes 288
High-Contrast Mode 288
Features for Easier Keyboard and Mouse Input 288
StickyKeys 288
SlowKeys 289
RepeatKeys 289
BounceKeys 289
MouseKeys 289
ToggleKeys 290

Features for Users Who Are Hearing-Impaired....292
ShowSounds....292
SoundSentry....292
Support for Alternative Input Devices....292
SerialKeys....292
Support for Multiple Pointing Devices....292
Features for Software Developers....293
Accessibility Guidelines for Software Developers....293
Methods for Simulating Input....293
Chaining Display Drivers....293
New Common Controls....293

Chapter 20 Applications and Utilities....295
The Quick Viewers....295
WordPad....298
Paint....299
Backup....300
HyperTerminal....301
The New MS-DOS–Based Editor....302
The WinPad Organizer....303
Disk Utilities....304
The Disk Defragmenter....304
ScanDisk....304

Chapter 21 What Makes a Great Windows 95 Application?....307
The Win32 Application Programming Interface....307
OLE Functionality....308
The Features of OLE....309
The *Windows 95 User Interface Design Guidelines*....312
Support for Handling Plug and Play Events....315
Support for Quickly Identifying Files....316
Consistent Setup Guidelines....317

Chapter 22 The Windows 95 Logo Program....319
The Availability of Windows 95–Based Products....320
Licensing Criteria for the Windows 95 Logo....320
For More Information....320

Chapter 23 Questions and Answers About Windows 95 321
General Questions 321
Upgrading to Windows 95 323
Ship Dates and Packaging Plans 326
User Interface 327
Architecture 328
Plug and Play 329
Application Support 330
Networking 330
Systems Management 331
Messaging and Mail 332
The Microsoft Network 332
Mobile Computing 333
For More Information 334

Microsoft WinNews—Get the Latest Information on Microsoft Windows

Windows 95 is continuing to evolve as we get closer to the release of the product. To help keep you informed of the latest information on Windows, Microsoft has created the WinNews information forum, an easily accessible electronic distribution point for new white papers, press releases, and other pertinent documentation. If you have a modem or access to the Internet, you can always get up-to-the-minute information on Windows 95 directly from Microsoft on WinNews. Use the following electronic addresses to access further information:

CompuServe	GO WINNEWS
Internet	ftp.microsoft.com/peropsys/win_news
Worldwide Web	http://www.microsoft.com
GEnie	WINNEWS Download area in Windows RTC
Prodigy	Jumpword WINNEWS
AOL	Keyword WINNEWS

To subscribe to Microsoft's WINNEWS Electronic Newsletter, send e-mail to enews@microsoft.nwnet.com with the words SUBSCRIBE WINNEWS in your message.

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

INFORMATION PROVIDED IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND FREEDOM FROM INFRINGEMENT. The user assumes the entire risk as to the accuracy and the use of this document. This document may be copied and distributed subject to the following conditions: 1) All text must be copied without modification and all pages must be included; 2) All copies must contain Microsoft's copyright notice and any other notices provided therein; and 3) This document may not be distributed for profit.

Copyright © 1994 Microsoft Corporation. All Rights Reserved.

Acknowledgments

Special thanks to the contributors to this book, including (but not limited to):

Russ Arun
Richard Barton
Joe Belfiore
Andreas Berglund
Eric Bidstrup
Jeff Camp
Ina Chang
Brad Chase
Joyce Cox
Deborah Epstein-Celis
Dhiren Fonseca
Pat Fox
Richard Freedman
Andy Hill
Jean Kaiser
Bill Koszewski
Keith Laepple
Greg Lowney
Peggy McCauley
Yusuf Mehdi
George Moore
Doralee Moynihan
John Parkey
Dave Pollon
Rob Price
Victor Raisys
Jim Reitz
Brad Silverberg
Bob Taniguchi
Jeff Thiel
Mike Tuchen
Bill Veghte
Rogers Weed
Christian Wildfeuer

Dedicated to the Windows 95 Project Team—
for the development of one really cool operating system!

Foreword

Microsoft Windows 95 is the next major release of Windows from the Personal Systems Division at Microsoft. It's an integrated and complete Windows operating system that starts with the basic functionality found today in MS-DOS, Windows, and Windows for Workgroups. It improves ease of use, enhances workgroup functionality, makes it easier to find and share information, enables new application capabilities, and offers advanced operating system performance. The phrase "integrated and complete" is key—it means that Windows 95 is the first high-volume desktop release of Windows that does not sit on top of (or require any version of) MS-DOS. Windows 95 will be a compelling upgrade for existing Windows users and a compelling new computing environment for people who have not yet moved to Windows.

The development of Windows 95 began in early 1992 with a core group of developers. Over the course of several months, this group defined the mission for what was to become Windows 95. They closely examined the feature set of the existing versions of Windows and considered how computers would be used and what technologies would be common when Windows 95 was to be delivered, and beyond. They looked at the capabilities of existing PC hardware and the promise of new capabilities in a platform for the future. Their vision of the future included the following elements: PCs must be easier to use in order to expand the world of computing; PC hardware must be easier to install and configure to reduce the need for support and shorten the time required to set up and add peripherals; capabilities such as multimedia must be enhanced and made easier to use because more PCs will be used in the home; PCs must be easier to manage in networked environments; communication facilities such as access to electronic information from online services and the Internet must be more accessible; support for mobile computing must deliver easy access to remote information and leverage the dynamic nature of docking/undocking operations; and an operating system infrastructure must be developed that can be easily extended to accommodate future application requirements and yet be backwards-compatible with existing standards and existing software and hardware.

The mission of Windows 95, derived from these observations, defined a mainstream Windows operating system that would make PCs easier to use, would be faster and more powerful, would integrate network connectivity and manageability, and would preserve the users' investment in software and hardware. Windows 95 delivers on this mission.

To make PCs easier to use, we provided a new shell, integrated Plug and Play into the operating system, included wizards to guide users through system operations, and delivered a consistent user interface across all operating system components and namespaces. The increased speed and power come from a modern 32-bit operating system with preemptive multitasking, threads, and a platform for a new generation of 32-bit applications. Windows 95 is a complete operating system that is not limited by MS-DOS and has all the benefits of a completely self-contained graphical operating system. In the area of connectivity—either in a LAN, a WAN, or a mobile dial-up environment—we made connecting to remote resources as easy as connecting to local resources, and we built in access to electronic information for electronic mail and online services. And finally, we made Windows 95 compatible with the software and hardware that users own today.

Windows 95 offers something for everybody. The administrator in a corporation with more than 10,000 PCs will gain from improved manageability; the person in a branch office with 10 PCs will benefit from built-in networking and dial-up remote access. The home PC user will gain from improved and built-in multimedia functionality, and the small business owner will profit from built-in peer networking. Windows 95 offers features and capabilities to make you more productive and keep you in touch with the evolving world of online information and e-mail. Windows 95 will change the industry and the way people use PCs in a way that no other operating system release ever has. We are excited about the possibilities that Windows 95 offers, and we hope you are too.

This book, *Introducing Microsoft Windows 95*, provides insight into Windows 95 from a unique perspective—straight from the project team that developed it. Enjoy!

Brad Silverberg
Senior Vice President, Personal Systems Division
Microsoft Corporation
Redmond, Washington
November 1994

Introduction

As the successor to Microsoft's MS-DOS, Windows 3.1, and Windows for Workgroups 3.11, Windows 95 is the next major release of the standard operating system for desktop and portable PCs. This release offers something for everyone, including a more intuitive way to work, new capabilities such as "surfing the information highway," and better support for managing an installation site with hundreds of PCs.

Where We've Been

Over the past decade, the PC industry has delivered innovative, cost-effective products that have made the personal computer a widely used tool both in the office and in the home. The development of these products was made possible by a number of key advances during the 1980s and early 1990s:

- The adoption of MS-DOS as an operating system standard for PCs, which provided a platform for application development
- Rapid decreases in price and increases in performance made possible by innovative PC components, especially the Intel i386 and i486 microprocessors
- The wide adoption of the Microsoft Windows operating system, which provided an appealing graphical user interface and multitasking capabilities and made possible a new generation of graphical applications, thereby making PCs much easier to use

Where We Are Today

Although the PC has made dramatic gains over the past decade, a number of limitations prevent current users from taking full advantage of their PCs and discourage others from beginning to use them:

- **PCs are still not easy enough to use.** Learning how to use the PC is difficult because the PC does not lend itself to learning through experimentation. The wrong set of keystrokes can, for example, inexplicably cause programs to

disappear. As a result, learning basic processes such as launching or switching between applications is frustrating. Configuring the PC is also difficult; for example, tasks such as connecting to a printer or adding a CD-ROM drive are too complicated.

- **Users want software that takes full advantage of the power of their hardware.** The performance delivered by hardware—the CPU, video system, CD-ROM drive, video subsystem, and so on—is constantly accelerating. And new hardware, such as wireless, PCMCIA, and MPEG, continuously arrives on the market. Users want their software to fully exploit all this power.
- **Connecting to a network is hard.** The two primary PC operating systems, MS-DOS and Windows, were not designed with easy connection to a network in mind. Consequently, the basic task of connecting a PC to a server and to other PCs is still a challenge, in spite of the prevalence of networks in organizations of all sizes.
- **PCs are expensive to support.** The cost of hardware is small relative to the cost of installing, configuring, and maintaining a PC and of training and supporting users. These costs must be reduced to make PCs a more cost-effective tool for business.

Where We're Headed

Windows 3.1 moved the PC platform forward by making PCs easier to use. Yet the problems of today's users indicate that the ease, power, and overall usefulness of the PC must be further increased. Windows 95 raises the PC platform to new levels in the following ways:

- **Windows 95 makes PCs even easier to use.** Windows 3.1 put a friendly interface on top of MS-DOS to make common PC tasks easier. With Windows 95, the goal is to make those tasks more intuitive, or where possible, automatic. For example, adding and configuring new hardware devices has been simplified. Windows 95 automatically loads the appropriate drivers, sets IRQs, and notifies applications of the new capabilities of the hardware device without any action by the user. A redesigned user interface, highlighted by the Windows Taskbar, makes computing more automatic for novices. (Usability tests show a ten-fold improvement over Windows 3.1 in the time it takes to complete certain common tasks, such as starting an application.) The interface also makes the power of the PC more discoverable for intermediate and advanced users.

- **Windows 95 is a faster and more powerful operating system.** Ease on the surface requires power and speed at the core, and the modern, 32-bit architecture of Windows 95 meets these requirements. Freed from the limitations of MS-DOS, Windows 95 preemptively multitasks for better PC responsiveness; for example, users will no longer have to wait while the system copies files. It also delivers increased robustness and protection for applications and provides the foundation for a new generation of easier, more powerful, multithreaded 32-bit applications. And most important, Windows 95 delivers this power and robustness on today's average PC platform while scaling itself to take advantage of additional memory and CPU cycles.

- **Windows 95 integrates network connectivity and manageability.** Windows 3.1 gave users the power to make better use of their PCs, but it did not make the same strides for MIS organizations. Windows 95 addresses this deficiency by providing a system architecture that makes basic network connectivity easy. By integrating high-performance, 32-bit client support into the operating system—including the 32-bit Microsoft Client for Novell NetWare Networks—Windows 95 goes beyond simple connectivity to enable the central management and control of the PC. User profiles, policies, and the ability to leverage server-based security make it much easier for MIS organizations to administer and support large numbers of PCs.

Windows 95 is more than the next generation of Windows; it is a catalyst that will move the PC industry to a higher level. The release of Windows 95 is expected to spawn not only a new generation of PCs and peripherals that support Plug and Play, but also a new generation of powerful 32-bit Windows–based applications.

How We Get There

The development of Windows 95 has been guided by the primary principle that all technological improvements must translate into practical benefits for users, and that these benefits must be easily and inexpensively available to everyone. This principle has produced the following results:

- **Compatibility with existing MS-DOS–based and Windows–based applications.** Because Windows 95 is designed to add significant value to any PC without requiring additional software or hardware, compatibility with existing applications is essential. Windows 95 not only supports the enormous range of existing MS-DOS–based and Windows–based applications, it goes a step further by fixing key compatibility deficiencies of Windows 3.1; for example, it provides much better support for demanding MS-DOS–based applications (such as certain types of games, which can now be run from Windows).

- **Compatibility with existing hardware.** The wide range of hardware available for Windows 3.1—from scanners to plotters to fax modems to video capture boards—is enormously valuable to users. While providing support for new, easier-to-use peripherals through Plug and Play, Windows 95 preserves the huge investment of users and manufacturers by maintaining compatibility with existing peripherals and their associated device drivers.
- **Performance equal to or better than that of Windows 3.1.** In keeping with the goal of adding value to existing PCs without requiring additional hardware and software, Windows 95 matches or exceeds the performance of Windows 3.1 on today's average PC (386DX with 4 MB of RAM). As more memory is added to the PC, performance scales faster than Windows 3.1. As a result, access to all the new features in Windows 95 should not incur any performance penalty.
- **Safe, hassle-free upgrade and migration.** Windows 95 makes the upgrade process as easy as possible by providing a Setup program that installs cleanly over Windows 3.*x*. Windows 95 includes the Windows 3.1 Program Manager and File Manager so that users can migrate to the new user interface at their own pace.

The mission of Windows 95 is to go beyond making PCs easier to use and make them truly usable. A truly usable PC is more intuitive and automatic, integrates the latest technologies, offers superior responsiveness and stability, and is compatible with today's existing software and hardware. Windows 95 also aims to make the upgrade and transition easy, without pain and without loss of performance or capability. This book details the specific benefits of the many Windows 95 innovations, underscoring improvements over Windows 3.1. You'll see as you read through the chapters that the size and scope of Windows 95 is awesome, and that its features are worthy of its mission.

CHAPTER 1

The Microsoft Windows Operating System Family

The Microsoft Windows operating system family provides users and developers with a rich set of services that take full advantage of the broad range of available hardware platforms, from small form-factor portable systems to multiprocessor servers. Windows 95 is for users of Intel–based desktop and laptop PCs. Windows NT Workstation and Windows NT Server are advanced operating systems for workstations and network servers, respectively, and support both Intel and reduced instruction set computing (RISC) microprocessors, as well as symmetric multiprocessing (SMP).

The scaleable architecture of the Windows operating system family supports the same user interface, applications, and development tools across an ever-expanding range of hardware. At the same time, the Windows operating systems meet the spectrum of customer requirements, from productivity applications to powerful, secure, mission-critical applications.

Windows 95 and Windows NT Workstation

Because it is not currently possible to have one operating system that fully exploits the broad range of available hardware, the Microsoft Windows operating system family, shown in Figure 1, has two distinct design points: one centered on mainstream systems, and the other centered on leading-edge systems.

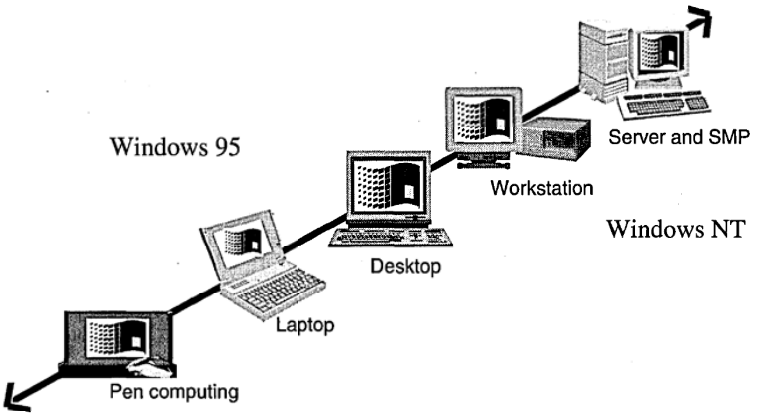


Figure 1. A consistent platform for development, deployment, and training

For mainstream systems (currently represented by products such as subnotebook and entry-level desktop machines), Windows 95 delivers responsive performance for a broad range of applications while conserving the amount of system resources used. Windows 95 is designed for use on Intel–based PCs and supports the Intel 80386DX, 80486, and Pentium processors used in mainstream desktop and portable PCs.

For leading-edge systems, Windows NT fully exploits the capabilities of the hardware and provides the most advanced services for the most demanding applications. Windows NT is designed for use on Intel–based and workstation PCs—for example, a MIPS R4400 or Digital Alpha AXP–based system, a dual-processor workstation, or a multiprocessor RISC server.

Because of the requirements placed on new enterprise solutions, all major operating system developers—including Microsoft, IBM, Sun (and most UNIX vendors), and Novell—have recognized the necessity of moving to a micro-kernel architecture for their leading-edge operating systems. Microsoft made this commitment over five years ago and began shipping Windows NT to developers in July 1992 with general availability in July 1993. This architecture allows vendors to enhance systems to respond to the rapidly changing requirements of evolving business solutions while maintaining the flexibility needed to exploit new hardware and peripherals.

Both Windows 95 and Windows NT Workstation provide a common base of functionality that is required by all customers, including ease of use, power, connectivity and manageability. Microsoft is committed to and will deliver parity in basic functionality (such as the user interface) to each platform as quickly as possible.

The differences between the two platforms are a result of their different design goals. Windows 95 is focused on making computing easier for anyone using a wide range of personal and business applications on desktop and portable computers. To protect their

current investment, these users require the highest level of compatibility with today's applications and device drivers. Windows NT Workstation is focused on providing the most powerful desktop operating system for solving complex business needs. For developers; technical/engineering/financial users; and business operations application users, Windows NT Workstation delivers the highest level of performance to support the most demanding business applications. It also provides the highest levels of reliability, protection, and security for critical applications while exploiting the latest hardware innovations, such as RISC processors and multiprocessor configurations. This focus on solving business needs is also reflected in the emphasis on maintenance and regular system updates.

Over time, as mainstream systems become more powerful, technologies implemented first in the leading-edge Windows operating system will migrate to the mainstream operating system. Sometimes technical innovations will appear first in the mainstream operating system because of timing of releases or because some features improve ease of use for general users. However, the guiding principle for product planning is that the leading-edge operating system will provide a superset of the functionality of the mainstream operating system.

For application developers, Microsoft provides only one Windows programming platform, defined by Win32—the 32-bit Windows application programming interface—and OLE. By following a few simple guidelines, developers can write a single application that runs across the Microsoft Windows operating system family. Optionally, developers can target a specific operating system whose functionality is important to a particular application, but targeting is not a requirement.

Which Operating System?

The decision about which of the Microsoft Windows operating systems to deploy should be based on the tasks to be accomplished. The two operating systems provide a complementary set of capabilities that can accommodate a broad range of usage scenarios.

Windows 95 Usage Scenarios

The following examples illustrate scenarios where Windows 95 is the best choice.

- Most office environments require people to perform a variety of general tasks, such as word processing, database queries, or spreadsheet analyses, using productivity applications, such as the Microsoft Office suite. These people may also be using applications that are specific to their particular business. Their companies have an installed base of personal computers, peripheral devices and applications, and Windows 95 allows them to maximize their investment in that computing infrastructure.

- Many companies have employees who spend a high percentage of their working hours away from the office—for example, at customer sites, in hotels, or out in the field—and rely on personal computers to help them perform their jobs. Windows 95 meets the requirements of these mobile computer users for the same application and device compatibility as their office-based colleagues, but also places lower demands on their hardware, including amount of memory, battery power, and use of disk space.
- Most users of computers in the home find them both challenging and unfriendly. However, they want to be able to take advantage of new capabilities, such as multimedia, and to easily access online information services. Windows 95 is easy for all family members to use, yet has the power to do what they want to do. Built into the operating system are rich multimedia capabilities; the highest levels of compatibility for running MS-DOS–based applications, such as games; and connectivity to information services, such as the Internet or other online services. In addition, technology such as Plug and Play allow some users to effortlessly add new components, such as printers, modems, and other peripherals, to their systems.

Windows NT Workstation Usage Scenarios

The following examples illustrate scenarios where Windows NT Workstation is the best choice.

- Engineers, scientific researchers, statisticians, and other technical users often need to use processing-intensive applications for data analysis and large design activities. Windows NT Workstation, with its support for SMP and its portability to high-performance platforms, such as those based on Pentium, Alpha, or MIPS CPUs, can provide the performance of a leading-edge workstation or minicomputer at a fraction of the cost. Moreover, with Windows NT Workstation, users can also run personal productivity applications on their systems.
- For industries that need to protect sensitive data or application files, such as banking and defense, Windows NT Workstation provides a secure desktop. The NTFS file system, combined with appropriate security procedures, helps prevent unauthorized access to systems and data, and the security model in Windows NT Workstation is designed to be compliant with C2-level certification. With these features, a single Windows NT system can be shared by multiple users and still maintain security for all files on the system.
- Many companies require high levels of availability and performance and cannot afford downtime, regardless of which application is running. Often these types of systems are being "right-sized" from mini and mainframe systems. For example, many manufacturing companies use 16-bit applications to manage their production lines. With Windows NT Workstation, these Win16–based applications can be run in separate address spaces (often referred to as separate virtual machines). Then even if one application fails, the other applications

continue to run. Windows NT Workstation also provides complete protection for 32-bit applications and automatic recovery (reboot, if necessary) if the system goes down.

Evaluating Windows 95

As you compare Windows 95 with other operating system products on the market, including Windows 3.1, you should examine the following areas to help identify the operating system that best meets your needs and the needs of your system's users:

- Ease of use
- Performance
- Compatibility of device and application support
- Support for networking and connectivity
- Support for manageability and administration
- Support for communications and messaging
- Support for mobile services and remote access

In the following sections, we briefly discuss these evaluation criteria. The remaining chapters in this book show how Windows 95 provides the best desktop operating system for mainstream platforms in each of these areas.

Ease of Use

It is important to look at the ease-of-use aspects of an operating system from the perspectives of both a novice and an experienced user. Novice users include both people who have never used a PC and people who have used one infrequently, often because they find PCs intimidating. Novices might have trouble moving around the user interface and might need more information or coaching—for example, from an online Help system. Experienced users generally interact with more areas of the operating system than novice users, and they demand flexibility, speed, and power.

As you evaluate the ease of use of an operating system, it's helpful to answer these questions:

- Is the operating system easy to learn and use and efficient for the widest range of users?
- Can users discover new features and new, more efficient ways of performing tasks as they become more experienced?
- Does the operating system make it easy for novice users to complete common tasks, such as starting new applications, switching between two or more active applications, or manipulating files?
- Is the operating system flexible enough that experienced users can customize it to reflect the way they interact with the computer?

Performance

The term *system performance* refers to how the operating system performs overall while performing a set of broad tasks—for example, running a group of applications and programs that are normally run simultaneously. The term *performance* also refers to the ability of individual system components or subsystems to perform a more narrow set of tasks—for example, file input/output (I/O) operations.

Several available suites of benchmarks test the ability of operating systems to complete a set of tasks that are designed to mimic real-world use of a particular PC/operating system combination. These benchmark suites produce numbers that represent the responsiveness of the operating system for a given set of commercially available applications. You can run the same set of applications in your environment and use the benchmark information to determine the relative performance of various operating systems.

However, benchmark suites don't tell the whole story. In addition to running application benchmark suites, you should isolate and separately test various components and subsystems of the operating system to obtain low-level results that indicate how well the operating system can support the services used by applications. Areas commonly isolated and benchmarked on standalone PCs include the performance of the local file system for disk and file I/O, the performance of the graphics subsystem and video display drivers for graphics and text I/O, and the performance of the printing subsystem for printing I/O. In addition, you should test desktop operating systems in networked environments for their ability to support network I/O throughput for the supported network clients, as well as server functionality responsiveness (if supported by the operating system).

All operating systems perform at their best on a PC that has the maximum amount of RAM. However, most users' PCs have less than the maximum amount. You should run performance tests against different hardware configurations, including memory ranges from 4 MB to 16 MB and PCs containing Intel 80386DX, 80486, and Intel Pentium–based CPUs. Because different hardware resources deliver different performance testing results, it's important to test not only on more than one PC configuration, but also on hardware that is currently mainstream in the industry.

As you evaluate the performance of an operating system, it's helpful to answer these questions:

- Does the operating system perform well on a wide variety of hardware and software?
- How well does the operating system complete benchmark tests on a suite of applications on a given hardware platform?
- How well does the operating system complete benchmark tests on individual components and device drivers provided as part of the system?

- Does the operating system perform well as far as network connectivity for supported network clients or provided network server functionality is concerned?

Compatibility of Device and Application Support

When it's time to replace an old operating system, a key question to consider is "Can my company still use its existing hardware and software with the new operating system?" Your company has probably invested a large amount of money in applications, printers, modems, and other PC-related peripheral devices. It's important to find out whether the replacement operating system can run with the existing hardware and software.

It's also important to know how broad a range of devices is supported by the operating system you choose. No doubt, as your company grows, your hardware needs will grow too. Your choice of an operating system should not unreasonably restrict the peripheral devices your company can buy later. The operating system you choose should include ample device drivers, not only to support the devices you currently own, but also those you will buy in the future.

When examining device support of an operating system, consider the number of devices supported, the industry standards that the operating system supports, and compatibility with existing device drivers shipped with earlier operating systems or with the devices themselves.

As you evaluate the device and application support of an operating system, it's helpful to answer these questions:

- Does the operating system provide broad support for your company's existing hardware and the associated MS-DOS–based and Windows–based device drivers?
- Are devices easily recognized, installed, and configured by the operating system?
- Does the operating system allow you to run your existing MS-DOS–based or Windows–based applications as well as MS-DOS 6.*x* or Windows 3.1?
- Does the operating system allow the easy exchange of information among applications, or does it support advanced interapplication communication mechanisms?
- Does the operating system provide services for new types of applications, such as multimedia, remote access, and communications-related applications?

Support for Networking and Connectivity

In a corporate environment, an operating system must be able to provide network support for a broad base of clients. You should compare each operating system's ability to support connectivity in a heterogeneous environment, as well as how successfully network functionality and other areas of the system, such as the user

interface, are integrated in each operating system. Bear in mind that, in general, companies are not looking for the incorporation of proprietary network functionality in an operating system. They want the operating system to support industry-wide standards so that they don't have to rely on a single vendor to support a multivendor environment.

As you evaluate the networking support of an operating system, it's helpful to answer these questions:

- Is the operating system an open, layered networking architecture that lets you mix and match best-of-breed components at every layer?
- Does the operating system have built-in, native support for popular networks?
- Does the operating system natively support a wide range of network transports, such as TCP/IP and IPX/SPX; industry-wide communication protocols, such as RPC, NetBIOS, DCE, and named pipes; and existing network device standards, such as NDIS and ODI?
- Does the operating system provide a simple, consistent user interface for accessing the network and using network resources?
- Does the operating system support an open architecture that allows third-party and network operating system vendors to easily integrate or add network connectivity enhancements or application support?

Support for Manageability and Administration

PCs are now one of the largest expenses of an MIS organization. Medium and large businesses invest tens of thousands of dollars each year, not only on the hardware and software for new and existing computer systems, but also for setup and administration of these systems. Currently, the available tools for managing and administering PCs in a networked environment have little consistency and almost no integration.

Standards organizations are now working to simplify system administration by developing standard methods for managing PCs. These standards will mean better and more integrated management tools for the network administrator. For an administrator to reap any benefits, however, the operating system must support management mechanisms that adhere to existing standards or its infrastructure must be designed for adaptability to a new standard.

As you evaluate the support for manageability and administration of an operating system, it's helpful to answer the following questions:

- Does the operating system provide the tools and platform infrastructure for supporting management mechanisms that adhere to existing industry standards, such as SNMP, and is it flexible enough to support future standards, such as DMI?

- Does the operating system provide tools and mechanisms for MIS organizations and administrators to customize and control the functionality and capabilities on the desktop?
- Does the operating system provide support for managing desktop PCs remotely over a network?

Support for Communications and Messaging

With the explosive growth of services such as CompuServe, America Online, and the Internet, the increase in demand for an operating system that provides access to online and mail services has been dramatic. The support and services provided by an operating system can open the door to the Information Age, allowing users to discover new communications and messaging possibilities.

As you evaluate the communications and messaging support of an operating system, it's helpful to answer these questions:

- Does the operating system support high-speed communications and background multitasking capabilities?
- Does the operating system provide support for communication hardware; for new communication functionality, such as sharing communication ports; for unified device configuration; and for emerging communications technology?
- Does the operating system provide support for industry-standard messaging services?
- Does the operating system provide broad communication and messaging capabilities, such as faxing, dial-up access to resources, and access to online information services, and consolidated information access?

Support for Mobile Services and Remote Access

To realize seamless mobility, users must be able to easily communicate and remain productive, whether they are in the office, at a customer site, or at home. Users must be able to communicate with coworkers and clients regardless of their location. In addition, transitions from home computer to portable computer to office computer must not cause interruptions in workflow. Including support for mobility services as part of the operating system ensures tight integration and connectivity between portable computers and desktop PCs, allowing minimal work interruptions as users switch from one location and/or computer to another.

As you evaluate the support for mobile services of an operating system, it's helpful to answer these questions:

- Does the operating system support remote access to the key services or information you need on your corporate network?

- Does the operating system have robust support for the dynamic nature of mobile hardware, such as PCMCIA, power management, and docking stations?

CHAPTER 2

An Overview of Windows 95

Windows 95 is an extremely feature-rich operating system. Virtually every aspect of Windows 95 reflects improvements over Windows 3.1 and Windows for Workgroups. This book discusses the areas of technology that make up Windows 95, focusing on the following features, functionality, and components:

- The Windows 95 user interface
- Base system architecture
- Robustness
- Support for running MS-DOS–based applications
- Plug and Play
- Device support
- Networking
- Systems management
- Printing
- Communications
- Mobile computing services
- Microsoft Exchange: e-mail, fax, and more
- The Microsoft Network online information service
- Multimedia services
- Installation and setup of Windows 95
- International language support
- Accessibility
- Applications and utilities

Where appropriate, each discussion includes the following:

- **A summary of improvements in Windows 95 over Windows 3.1.** This section provides a quick overview of ways in which Windows 95 addresses Windows 3.1 problems or improves upon Windows 3.1 functionality. These discussions also apply to Windows for Workgroups, even though Windows for Workgroups may not be explicitly identified.

- **A Try It! section.** You are encouraged to see for yourself that Windows 95 is a flexible, powerful, and robust operating system by following brief sets of instructions for performing specific tasks.

Key New Features

As you read this book, it's important to keep in mind the needs of the marketplace and how Windows 95 meets those needs. This section briefly outlines some of the key new features in Windows 95 and the problems they solve. Because the scope of the new features is broad and their appeal is wide, they are organized here in terms of improvements over Windows 3.1 in the following areas:

- Ease of use
- Speed and power
- Compatibility

The features are further organized by benefiting group:

- End-users
- MIS organizations

Ease of Use

For End-Users

For end-users and MIS organizations alike, improvements in ease of use in Windows 95 fix the problems identified in Windows 3.1. For example, less-experienced users found overlapping windows and tasks such as minimizing and maximizing windows too complex, while more experienced users craved greater efficiency. But the improvements go beyond simply solving these problems, by also encompassing hardware, connectivity, and applications. Windows 95 offers these solutions:

- **A new user interface (UI).** A shower of improvements greatly enhances ease of learning, usability, and efficiency for all users, no matter what their level of expertise. Novice users can get started more quickly, and experienced users can fully unlock the power of their PCs.
- **Plug and Play.** The goal of Plug and Play is simple: When a user installs a new hardware device, it works.
- **Long filenames.** Signaling the end of cryptic 8.3 filenames, long filenames are just one example of the many usability improvements in Windows 95.

For MIS Organizations

MIS organizations find their jobs increasing in difficulty as the number of PCs in a given corporation grows more rapidly than their support staff. In addition, the introduction of Windows 3.1 led to a series of security, reliability, and management issues because of its lack of integrated connectivity and lack of management infrastructure. The following features of Windows 95 attack these problems:

- **Built-in networking.** Whether a company is running NetWare or Microsoft networks; IPX/SPX, TCP/IP or NetBEUI; NDIS or ODI, Windows 95 has native, integrated support for its network. And additional LANs are easily supported.
- **The Registry.** By holding all pertinent information about the system—installed hardware, installed software, and user preferences and rights—and by exposing its contents remotely through a wide variety of industry standard interfaces—SNMP, DMI, Win32 plus RPC—the Registry provides the foundation for a highly manageable PC.
- **User Profiles and System Policies.** Windows 95 can provide different operating system and UI functionality depending on who has logged onto a PC. For example, a network administrator can prevent a particular user from deleting program items no matter which PC that user logs onto, or the administrator can enforce a policy that requires everyone in a particular group to have passwords of at least six characters.
- **Pass-through user-level security.** A PC running Windows 95 can be configured to require that remote users and their passwords be passed through to and validated by a NetWare server or Windows NT Advanced Server before those users are allowed to carry out certain tasks, such as changing the PC's Registry or accessing its files, either over the network or remotely. This user-level security allows the existing network-based namespace to be leveraged.
- **Network backup agents.** Windows 95 includes agents that support industry-leading server backup products, making it easier to back up information stored on PCs throughout the network.

Speed and Power

For End-Users

A major area of concern for end-users is improving the efficiency and power with which they use Windows. Users want to get their work done faster. They want to be able to run more than one application or computer process at a time instead of waiting for their PCs to finish one task before starting another. They want to be more effective without sacrificing system stability or performance. And perhaps most important of all, they want to escape the feeling that they take advantage of only a small fraction of their PCs' capabilities.

Windows 95 is designed to anticipate and exploit key emerging trends and technologies. For example, the need for seamless mobile computing is becoming more important as more hardware power is packed into smaller and lighter designs and more users work at home or on the road. The explosion of the home market has increased the demand for more powerful multimedia applications, which necessitates better multimedia support from the operating system.

The following features of Windows 95 bring more power and speed to users:

- **True preemptive multitasking.** Windows 95 can preemptively multitask 32-bit applications smoothly and efficiently.
- **Scaleable performance.** As the amount of RAM increases, the performance of Windows 95 increases more rapidly than that of Windows 3.1 because of the high-performance, 32-bit architecture of Windows 95.
- **Support for 32-bit applications**. Support for the Win32 API in Windows 95 means that users can look forward to a new generation of easier, more powerful multithreaded 32-bit applications.
- **Increased robustness**. New features mean greater robustness and protection for existing MS-DOS and Windows–based applications, and the highest level of protection for new 32-bit Windows–based applications.
- **Mobile computing anywhere**. Windows 95 provides both a remote networking client that allows dial-up access to any network (including the Internet) running IPX/SPX, TCP/IP or NetBEUI over PPP, and a dial-in server that lets any PC running Windows 95 act as a secure, single-line dial-in gateway to a network.
- **Faster printing**. The new 32-bit printing subsystem in Windows 95 means users spend less time waiting for print jobs to spool and finish.
- **High-performance multimedia components**. Both the video playback engine (Video for Windows) and CD-ROM file system (CDFS) are new 32-bit components that deliver smoother video and sound reproduction.
- **More memory for MS-DOS–based applications**. The use of protected-mode device drivers and file systems in Windows 95 means users will routinely have 600 KB or more of free conventional memory available in each MS-DOS session, even if they are connected to a network, using a CD-ROM drive, a mouse, and so on.

For MIS Organizations

Because MS-DOS and Windows were not designed for networked environments, their use in those environments can result in a string of problems well known to MIS professionals. These problems include not having enough real-mode memory to run mission-critical MS-DOS–based applications, and instability under Windows 3.1—for

example, turning off a Novell NetWare server completely hangs all connected machines running Windows.

The following features of Windows 95 are just some of the solutions that will make the job of MIS organizations easier:

- **32-bit NetWare connectivity**. Windows 95 includes a 32-bit network client for NetWare that is fast and reliable and has a zero footprint in conventional memory. Windows 95 also includes a similar client for Microsoft Networks.
- **Multiple network support**. A PC running Windows 95 can have multiple network clients (to connect concurrently to a Novell NetWare server, a Windows NT Server, and so on) and multiple transports (including IPX/SPX, TCP/IP, and NetBEUI) loaded and running simultaneously, providing connectivity in a heterogeneous environment.
- **Universal information client.** Microsoft Exchange, the universal information client provided with Windows 95, is integrated seamlessly into the user interface and provides a wide range of plug-in services, such as e-mail, online services, group applications, and so on. Thus the user has a single interface with the outside world of information, and the MIS administrator has a single client that supports multiple e-mail and other systems. Windows 95 includes support for services such as Microsoft Mail for e-mail, Microsoft At Work fax software for faxing, and Internet mail.

Compatibility

For End-Users

If an operating system upgrade requires new software, more memory, or new hardware, then the upgrade's cost is far higher than just its purchase price. Currently, users often have to wait a substantial amount of time—usually until their next PC purchase—before they can benefit from new technology. One of the biggest goals of Windows 95 was to make it possible for everyone to stay up to date with the latest version of Windows.

The following features were included in Windows 95 with this goal in mind:

- **Compatibility with existing MS-DOS–based and Windows–based applications.** Windows 95 works with and even improves today's software.
- **The same or better performance.** Windows 95 runs PCs with at least a 386DX processor and 4 MB of RAM at least as fast as Windows 3.1 does—faster in many cases. Windows 95 requires no additional RAM to maintain performance.
- **Backwards compatibility with existing hardware devices.** Windows 95 supports existing hardware and device drivers while enabling next generation, easier-to-use hardware through Plug and Play.

For MIS Organizations

In terms of cost, MIS organizations also have problems upgrading to a new operating system. However, maintaining compatibility with existing hardware and software is even more important, not only because of the larger scale of the upgrade, but also because different operating systems might be used by different groups within the organization. Having to support more than one operating system multiplies the problems MIS organizations face. MIS professionals also worry about the need to migrate users to a new operating system quickly, easily, and in an orderly way and about retraining users.

Windows 95 addresses these needs through the following features:

- **Network compatibility.** In addition to its 32-bit Microsoft Client for Novell NetWare networks, Windows 95 is compatible with existing real-mode network clients and existing logon procedures, such as the scripts used by NetWare.
- **A self-paced guided tour and Windows 3.1 help.** Windows 95 offers a guided tour to help novice users and users new to Windows 95 find their way around. In addition, an online help system that eases the transition of Windows 3.1 users to Windows 95 is only a mouse-click away.
- **Inclusion of the Windows 3.1 Program Manager and File Manager.** An explicit Windows 95 goal is to allow MIS organizations and users to switch to the new user interface at their own pace. In keeping with this goal, Windows 3.1 users can opt to continue working with Program Manager and File Manager until they feel comfortable with the new interface.

CHAPTER 3

The Windows 95 User Interface

When you first boot Microsoft Windows 95, you know immediately that the old world of Windows running on top of MS-DOS is no more. Gone are the character-mode boot messages that held meaning only for a very small minority of computer users. Instead, you are graphically carried to the desktop of the new Windows 95 user interface (UI).

More than any other part of the operating system, the UI defines the user's overall experience. The easier, more powerful, and more compelling the UI, the better the user feels about computing and the more productive the user is likely to be. A great UI helps the computer industry grow because it makes computing easier and more natural for *all* people, from the novice user to the power user.

This chapter discusses the design process that produced the UI in Windows 95 and then introduces the components of the UI, organized into the following categories:

- **Easy.** Outlines UI features that make Windows 95 easy to learn and use, especially for those new to Windows.
- **Powerful.** Outlines UI features that make Windows 95 more powerful, efficient, and customizable for the experienced Windows user.
- **Compatible.** Outlines UI features that make Windows 95 easy to learn and to use for those familiar with Windows 3.1.

Designing the Windows 95 User Interface

The overarching goal of the UI in Windows 95 is to make PCs even easier to use for *all* people. Fulfilling this goal is a challenge because different people work in very different ways. Novices want learning how to perform a task to be easy, even at the expense of efficiency. However, experienced users want to do more with their PCs, and they want efficiency and flexibility. In addition, users upgrading from Windows 3.1 want to make the transition without throwing out everything they have already learned.

Windows 95 meets these disparate needs by being scaleable—that is, by being able to fit the proficiency and preferences of the individual user. For novices, the most common and essential features of Windows 95, such as launching an application, task switching, and finding a file, are easily "discoverable" via the Taskbar, with its Start button and push-button task switching. For experienced users, Windows 95 promotes efficiency, customizability, and control via such power-user capabilities as the Windows Explorer, rich secondary mouse-button clicking, properties sheets, and shortcuts.

Design Methodology

The UI in Windows 95 was not constructed from a blueprint drawn from a master specification. It started with clear objectives, guiding design principles, and a skilled team. The design process started with the basic question, "How can the UI in Windows 3.1 be improved?" That question launched a continuous cycle of discarding old ideas, conceiving new ideas, and learning—a constantly iterating design-usability test-redesign loop like the one shown in Figure 2.

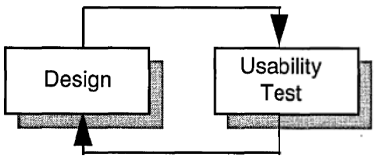


Figure 2. The design loop of Windows 95

Improving the Windows 3.1 User Interface

There was no shortage of information about how the Windows 3.1 UI might be improved. The table on the following page summarizes key findings.

The following mechanisms were used to compile this feedback data:

- **Usability tests.** The Microsoft Usability Lab, described on the following page, is primarily used for testing usability of new designs. However, to better understand how people use Windows 3.1 and to establish a baseline, several phases of testing were dedicated to Windows 3.1.
- **Focus groups.** Several focus groups were conducted with different levels of users to identify the problems people have with Windows 3.1.
- **Educator feedback program.** A team of UI designers and testers visited 12 independent software education companies. More than any other group of users, software educators understand the everyday usage challenges faced by novice and intermediate users. The educators were asked questions such as, "What are the

five hardest tasks for students to learn in Windows?" and "What five changes would you make to Windows to make it easier to learn?" The educators also tested prototypes of the UI in Windows 95.

- **Suggestion database.** Thousands of UI suggestions from Windows 3.1 users and corporate customers were compiled and analyzed, along with beta-tester UI feedback.

How Can the Windows 3.1 UI Be Improved?

Make It ...	Problem Areas
Easier to learn for novices	• Window management (overlapping and minimized windows) is confusing. • Hierarchical views (like those in File Manager) are confusing. • Double-clicking to launch applications is not discoverable. • Task switching is not discoverable. As a result, many users never run multiple applications.
More efficient and customizable for experienced users	• There is too much "middle management," with confusing and overlapping functionality between Program Manager, File Manager, Print Manager, Windows Setup, and Control Panel. • 8.3 filenames are restrictive. • The UI is not customizable. • Network and connectivity integration is poor.

Putting New Designs to the Test

Conducting extensive live tests in a variety of settings with a variety of subjects has been key to engineering a state-of-the-art UI. A large portion of the total development budget of Windows 95 has been expended on this critical activity, and Windows 95 is probably the most usability-tested product ever. The following methods have been used to test the UI in Windows 95:

- **Formative testing in the Usability Lab.** Conducted primarily in the groundbreaking Microsoft Usability Lab, formative testing collects data as test subjects perform specific tasks, such as launching a program, finding a file, and

installing a printer. The Usability Lab has nine testing suites, each with a one-way mirror, cameras, and other equipment for observing and recording users as they work. Central to the Lab's operations is online data-collection software that helps specialists collect cognitive and quantitative process data as subjects work through the sets of tasks.

Usability tests are observed firsthand by the design team and are essential in future designs. At the time this book was being written, more than 1000 hours of usability testing in 48 phases with more than 400 participants had been conducted. The experience of test subjects has ranged from novice users to intermediate/advanced users, so the test results focus on new computer users as well as users familiar with Windows.

- **Summative testing.** Conducted at customer sites and in the Usability Lab, summative testing involves testing the UI as a whole with real users over longer periods of time.
- **UI-expert and industry-expert review.** In the fall of 1993, a panel of UI experts and industry experts was assembled to review and critique the UI in Windows 95. In addition, four independent consultants each spent large blocks of time with Windows 95 and gave extensive feedback.

Easy to Learn

This section describes the features of the UI that are designed to make learning Windows 95 easy for novices.

The Desktop: Neat, Clean, and Logical

After users start their computers, they are presented with the Windows 95 desktop shown in Figure 3. It's neat and clean and displays only a few graphical objects.

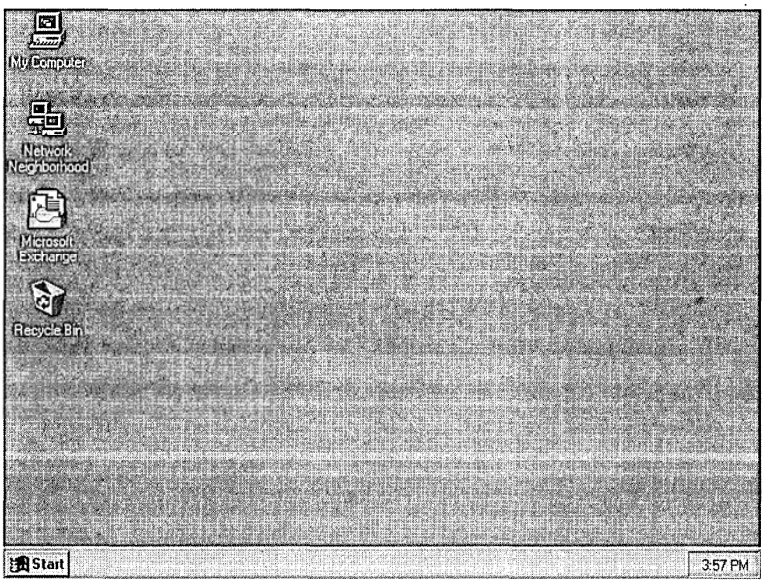


Figure 3. The desktop

The simplicity of the desktop appeals to all users' sense of organization, but it also serves to focus the novice user on the following essential elements:

- **The Taskbar.** Users can quickly start a program or open a document by clicking the Start button. And they can easily switch between tasks.
- **My Computer.** Browsing a PC is now logical and easy.
- **The Network Neighborhood.** In the world of mapped drives and complex interfaces, users can't browse their network. With the Network Neighborhood, they can easily browse the network, regardless of which network provider (such as Windows NT Server, a NetWare server, or Windows 95 itself) is being used.
- **The Recycle Bin.** The Recycle Bin allows users to recover deleted files and easily return them to their original location on the local system.

The Taskbar: Home Base

More than any other feature, the Taskbar, which is shown in Figure 4, exemplifies the order-of-magnitude improvement in ease of use and ease of learning of the UI in Windows 95. It is the UI's anchor. Its mission is to make 95 percent of what a typical user wants to do with the operating system easy to accomplish at all times. The Taskbar started out specifically as a program launcher and task switcher for novices. However, because of its simplicity and power, the Taskbar is also popular with experienced users, who can take advantage of its many other capabilities.



Figure 4. The Taskbar

The two key features of the Taskbar are the Start button and push-button task switching, which are examined in the next two sections.

The Start Button: Up and Running in Seconds

Usability tests on Windows 3.1 have shown that launching Write takes a new Windows user an average of nine minutes. With Windows 95, launching WordPad takes a new user an average of three minutes. If only the users that launch WordPad via the Start button (rather than by another means) are counted, the average launch time drops below one minute!

The main reason for this dramatic 3x–9x speed improvement is the Start button, which is shown in Figure 5. Without knowing about double clicking or complex hierarchies, a novice user of Windows 95 can quickly launch a program and get to work.

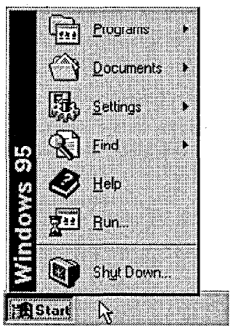


Figure 5. The Start button and its menu

However, the Start button is much more than a super-efficient program launcher. Its capabilities include the following:

- **Programs.** The Start button's Programs menu allows users to quickly launch programs. This menu is the equivalent of Program Manager in Windows 3.1, and in fact, when a PC running Windows 3.1 is upgraded to Windows 95, the contents of the program groups in Program Manager are transferred to the Programs menu.
- **Documents.** The Start button's Documents menu contains a list of the last 15 documents opened. This menu provides very quick access to the information most recently worked with and helps prevent time-consuming and frustrating browsing. It also helps users think of their work in terms of documents (a concept known as *document-centricity*), rather than applications.
- **Settings.** The Start button's Settings menu allows users to quickly change or view the PC's settings and options, including the Control Panel (for computer settings), the Start menu, and the Printers folder. It also allows users to customize the Taskbar to suit personal working preferences—for example, to specify which programs should be included on the Start button's Programs menu.
- **Find.** The Find item in Windows 95 goes far beyond the Search feature of File Manager in Windows 3.1. Searches do need not conform to the *.* searching syntax, and criteria such as last modification date, size of file, and actual text within a document can be used to find information.
- **Help.** Help has been overhauled in Windows 95 and is easily accessible via the Start button.
- **Run.** The Start button's Run item provides enhanced command-line type functionality.
- **Shutdown.** The Shutdown item provides easily accessible and safe shutdown, restart, and logoff.

Task Buttons: Task Switching Made Simple

Novices need to have powerful features presented to them in a simple and compelling way; otherwise they won't use these features. Research conducted with active Windows users indicates that fewer than 50 percent frequently use more than one application at a time and only 20 percent frequently use ALT+TAB task switching. These powerful features of Windows 3.1 are simply not discoverable.

The objective of the Taskbar is to make switching among multiple applications as simple as changing channels on a television set. Every open window has a button on the Taskbar, allowing the user to see which documents and applications are currently open. Switching applications is a simple matter of selecting the desired "channel" on the Taskbar. No more minimized program icons; no more disappearing windows. The user can see all the active tasks simply by looking at the Taskbar, the *TV Guide* of Windows 95. When a task is minimized into the Taskbar or maximized from the Taskbar, animation helps new users understand "where" the task goes.

Task buttons resize themselves automatically depending on the number of active tasks. If the buttons get too small to be useful, the user can customize the Taskbar. In fact, a host of other Taskbar configuration options allow the user to customize it in other ways, including the following:

- **Reposition.** The Taskbar can be dragged to any perimeter position on the screen.
- **Resize.** The width of the Taskbar can be widened by dragging its inside edge.
- **Auto Hide.** The Taskbar can be hidden and made to appear on the screen only when the mouse hits the screen edge, by selecting Settings and then Taskbar from the Start menu.

In addition to making task switching dramatically easier and more accessible via the Taskbar, the UI in Windows 95 includes an updated version of the familiar ALT+TAB "cool switch." It now displays an iconic road map of all active tasks to prevent users from getting lost in an infinite ALT+TAB loop, as was common under Windows 3.1.

Try It!

Customize the Start Button

1. Click *Start*, and then *Settings*, and then *Taskbar*.
2. On the *Change Start Menu* property sheet, select the programs you want to appear either at the *Start* button's first level or on the *Programs* menu.
3. Close the property sheet and check your new configuration by clicking the *Start* button.

Hint: You can also add a program to the *Start* button by dragging a shortcut defined for the program to the button.

Test a Novice

1. Find a stopwatch and a friend or family member who is a computer novice.
2. Sit the novice down at a PC that is running Windows 95 with no programs loaded and a clean desktop.
3. Ask the novice to start an application that you know is listed on the *Programs* menu. Note the time taken to successfully start the application.
4. Try the same task on a PC running Windows 3.1.
5. Compare the times to complete the task. The time using Windows 95 should be the same or faster than the time using Windows 3.1.

Display the Start Menu

- Press CTRL+ESC, and the *Start* menu pops up.

My Computer: An Easier Model for File Management and Browsing

File management and browsing in Windows 3.1 are not intuitive. Fewer than 55 percent of Windows users regularly use File Manager, and File Manager is especially confusing and intimidating for novice users.

Designing a discoverable and comfortable model for browsing and file management for the novice user has been a priority for the UI design team because of the observed difficulties with Windows 3.1. Several significantly different designs have been tested and thrown out. In the course of this testing, the design team made the following discoveries about basic file management and browsing:

- Exposed hierarchies are intimidating and unintuitive.
- Dual-pane views—hierarchy on the left and contents on the right—are also intimidating and unintuitive. Novices have difficulty understanding the connection between the logical tree hierarchy pane and the contents pane.
- An object-oriented UI works well for basic tasks but not for complex ones. The general belief is that the more object oriented a UI is, the easier it is to use. However, this is not the case. Although the direct manipulation of screen objects to achieve logical results is important for basic tasks (such as dragging a file from a folder to the desktop), direct manipulation to carry out more advanced tasks (such as dragging a file to a printer icon) is not intuitive. On the other hand, selecting an object with the mouse and then browsing menus or buttons for actions to perform on that object is intuitive.
- Large icon views are much more comfortable than list views.
- Whether novice users can find what they are looking for and whether they feel comfortable and "grounded" along the way are the defining characteristics of a good browsing experience. Efficiency and speed are less important.

The My Computer default browsing model is the result of this testing. A folder or drive can be opened by double-clicking it, or by selecting it and choosing Open from the File menu. The default browsing model brings up a new window in large icon view, as shown in Figure 6 on the following page.

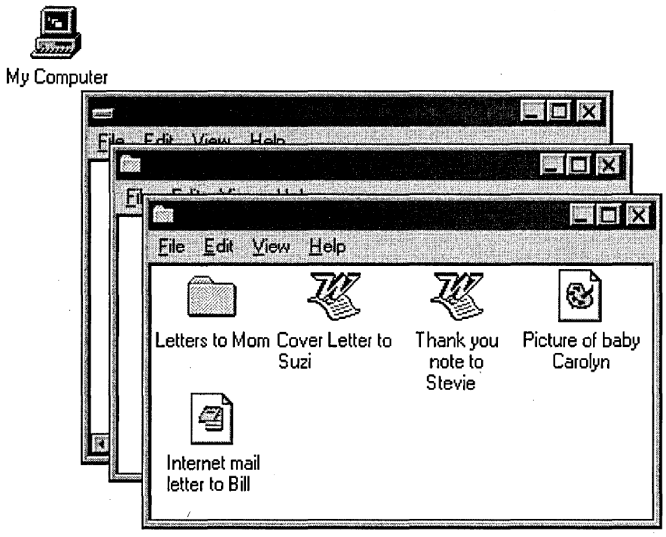


Figure 6. Browsing My Computer

To many advanced users this behavior seems cumbersome. "Why not open in list view?" they ask. "Why create a new window that just clutters up my screen?" "Why not open in a dual-pane view? It's much more efficient for me." "Why not turn the Toolbar on by default?" All of these possible default models and more were tested thoroughly and discarded because they caused confusion and stress for novices. Novices respond best when they are presented only with essential information and when they can easily "get back" to where they just were, so the default model was designed to meet these needs.

(Experienced users can select from multiple configuration options by choosing Options from the View menu. Also for experienced users, Windows 95 has a very powerful dual-pane browsing application called the Windows Explorer. In addition, File Manager from Windows 3.1 is still available and can be run for backward compatibility.)

The new capabilities of the default browsing model should not be overlooked in this discussion of simplicity. Folders can be created within folders. Files and folders respond logically to being dragged and dropped. Files and folders can be cut, copied, and pasted just like text and objects within applications can. Views can be customized, and each window "remembers" how it was last configured and opens automatically in that view. The best way to discover the capabilities of the default browsing model is to explore it, or better yet, watch a novice user explore it.

Try It!

Browse Folders with a Single Window

1. Double-click *My Computer.*
2. From the *View* menu, choose *Options*. On the *Folder* property sheet, select the *Browse folders with a single window that follows you as you open each folder* option.
3. Turn on the Toolbar by choosing *Toolbar* from the *View* menu.
4. Now double-click the icon for your hard drive. No new window opens.

Long Filenames: Greater Flexibility When Naming Files

By far the most-requested file system feature since Microsoft first released MS-DOS is support for long filenames, but until Windows 95 long filenames have not been possible. Windows 95 allows filenames of up to 255 characters. An example is shown in Figure 7. Eliminating the need to conform to the 8.3 naming convention results in obvious and large gains in usability. However, to ensure backward compatibility with existing MS-DOS and Win16–based applications, extensions have not been eliminated entirely; they are simply hidden from view by default.



Figure 7. A sample long filename

Files can be renamed in place in Windows 95 by selecting the file, clicking the filename, and typing the new name. The hidden file extension is not affected when a file is renamed. Files can also be renamed from within the new common dialog boxes, including the Open and Save dialog boxes.

Try It!

Display the File Extensions

1. From any folder, choose *Options* from the *View* menu.
2. Select the *View* tab.
3. Deselect the checkbox for the *Don't display MS-DOS file extensions for files that are properly registered* option.

The Network Neighborhood: Accessing Networking Features

This section discusses how the network client in Windows 95 makes browsing networks not only possible but easy, regardless of the network provider (Windows NT Server, Novell NetWare, Windows 95, and so on). For more details about the networking capabilities of Windows 95, see Chapter 9, "Networking."

Network browsing is accomplished by means of the Network Neighborhood, which sits on the desktop and logically represents the resources not available via My Computer. Its icon is shown in Figure 8.



Figure 8. The Network Neighborhood icon

Browsing the network via the Network Neighborhood is as easy as browsing a local hard disk.

- **Top-level configuration.** The Network Neighborhood can be configured by the network administrator to display only those PCs, servers, and printers that are in the user's immediate workgroup. Top-level configuration insulates the user from the vastness of large corporate networks. The user can still browse the larger network by opening Entire Network from within the Network Neighborhood. (Until Windows 95, browsing the larger network was not possible.) When a user browses a server, network connections are made without drive "mapping" (the assigning of new drive letters to a specific network resource).
- **Systemwide support for UNC pathnames.** This technology makes obsolete the process of mapping drives and allows natural network browsing via the Network Neighborhood. UNC pathname support allows a whole host of usability improvements of which network browsing is just one.
- **The Network Control Panel tool.** This tool consolidates all networking configuration in one location, and thereby eliminates the difficulty of configuring networking under Windows 3.1 and Windows for Workgroups 3.*x*.
- **Easy drive mapping.** A Map Network Drive button on the Windows Explorer and browsing window toolbars make drive mapping available in Windows 95. (Power users can also right-click My Computer.) Mapped drives appear as connections in My Computer.
- **Networking and mobility.** The UI in Windows 95 was designed from the ground up with networking and remote access in mind. For example, when a file is copied over a slow-link (a modem connection), the Copy dialog box includes an *Estimated time to completion* status message.

- **Networking integration with new common dialog boxes.** The new common dialog boxes, which are standardized in applications that make use of them, provide a consistent way to open and save files on network resources as well as on local drives. In addition, the Network Neighborhood can be browsed directly from the common dialog boxes, and the majority of basic file management tasks can be performed from them.

Try It!

Create a Shortcut to a Network Folder on the Desktop

1. Browse the *Network Neighborhood* until you find an often-used network folder.
2. Point to the folder, hold down the right mouse button, and drag the folder to the desktop.
3. Choose *Create Shortcut Here.*
4. Close the network window.
5. Double-click the shortcut. The network folder opens in a new window. The shortcut will be available every time you boot Windows 95.

Use the UNC Path to "Run" a Favorite Network Folder

1. From the *Start* menu, choose *Run.*
2. Type the full UNC path to your favorite network folder, such as *\\MKTG\PROGRAMS\SARAHB*, and press ENTER. The folder opens in a new window, with no drive mapping.

Create a New Folder from Within Common Dialog Boxes

1. Click the *Start* button, and then choose *Programs*, *Accessories*, and *WordPad.* (WordPad is the word processing equivalent in Windows 95 of Write in Windows 3.1. It uses the common dialog boxes.)
2. From the *File* menu, choose *Open*, and click the *Look in* drop-down box, which provides access to the entire PC hierarchy, including the Network Neighborhood.
3. From the *File* menu, choose *Save*, and click the *Create New Folder* icon. Unlike in Windows 3.1, where you have to start File Manager or exit to the MS-DOS command prompt to a create a new folder, you can create a folder when you save a document.

The Recycle Bin: Easy Deleting and Undeleting of Files

The Recycle Bin is an easily recognizable metaphor for being able to "throw away" files and then recover them by simply removing them from the bin. Files deleted in Windows 95, or deleted from the common dialog boxes in applications that support them, are moved to the Recycle Bin. Users can remove an item from the Recycle Bin and drag or cut/copy/paste it to another location, or they can restore it to its original location by choosing Undo Delete from the Edit menu.

The Recycle Bin graphically indicates whether it is empty or contains items. Information about "deleted" items is available in the Recycle Bin's details view, as shown in Figure 9.

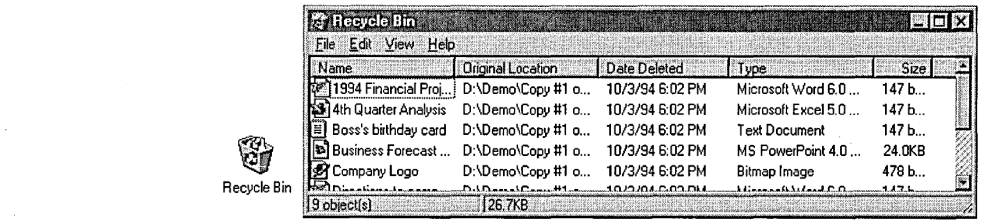


Figure 9. The Recycle Bin with deleted items and the Recycle Bin details view showing additional information

Focus on Documents: Working the Way Users Work

OLE introduced the concept of "document-centricity" by incorporating in-place editing of objects. In a document-centric environment, the application window changes and the document stays the same, so that software works the way people work, rather than vice-versa.

The UI in Windows 95 picks up on the concept of document-centricity in the following subtle but powerful ways:

- **Windows as views of objects.** When the user opens an object from anywhere in the UI, a new window opens. Logically, the title of the new window is the same as that of the object's icon. For example, when the icon of a Microsoft Word document called My Document is double-clicked from the anywhere in the UI, a new window opens entitled My Document—Microsoft Word.

- **Document creation from within folders and in the Windows Explorer.** From within any folder in Windows 95 or from the desktop, users can create new files in place by choosing New from the File menu and then selecting a file type. An icon like the one shown in Figure 10 is then created to represent the new file. This flexibility makes it very convenient to manage files based on projects, rather than at the whim of an application.



Figure 10. The icon for a new Word document

Try It!

Create a New WordPad Document from Within a Folder

1. Select a project folder in which you want to create a new WordPad document.
2. From the *File* menu, choose *New* and then select *WordPad Document.*
3. Type a name and press the ENTER key.
4. Double click the new document to open it in WordPad.

Hint: This functionality can also be accessed by right-clicking from within any folder or on the desktop.

Backtracking: Undoing File Operations

When working with files on your system, how many times have you said to yourself, "I didn't mean to do that!" after accidentally deleting, renaming, moving, or copying a file that you didn't intend to? Windows 95 has a simple answer for putting things back the way they were. Windows 95 provides a multilevel undo feature that allows users to undo one or more of their preceding actions. Users can undo file deletions, renames, moves, or copies by simply choosing Undo from the Edit menu of any UI window, as shown in Figure 11 on the following page.

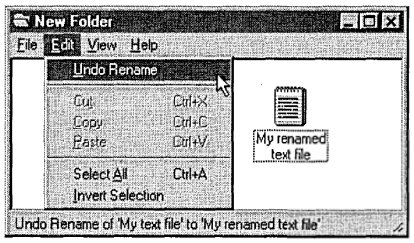


Figure 11. The Undo command on the Edit menu, which can be used to undo file operations

Try It!

Undo a File Operation

1. Open a folder and select a file.
2. Rename the selected file.
3. From the *Edit* menu, choose *Undo Rename* to undo the rename operation.

Undo Multiple File Operations

1. Open a folder and select a file.
2. Rename the selected file.
3. Drag the file from the folder to the desktop.
4. Delete the file.
5. Go back to the folder you first opened.
6. From the *Edit* menu, choose *Undo Delete* to undo the delete operation.
7. Choose *Undo Move* from the *Edit* menu to undo the move operation.
8. Choose *Undo Rename* from the *Edit* menu to undo the rename operation.

Wizards: Guides to Powerful Capabilities

Originally developed in Microsoft's Applications Group and used in applications such as Microsoft Word and Microsoft Excel, Wizards are a proven tool that make it easy for all classes of user to take advantage of powerful but complex functionality. The Wizards guide a user through a series of questions, which are posed to the user in a friendly and straightforward way, and walk the user through a process like the one shown in Figure 12.

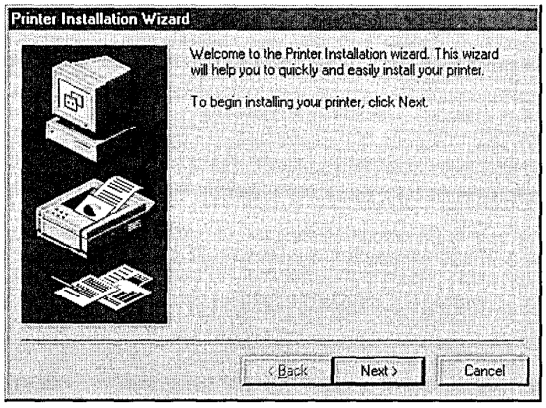


Figure 12. The Printer Installation Wizard, which walks the user through the printer installation process

Windows 95 uses Wizards throughout the operating system to assist all types of users. For example, Wizards are used to perform the following operations:

- Displaying Setup options to the user during the installation process
- Adding a new device to the system, such as a printer or modem
- Setting up remote access in the Network Neighborhood
- Creating a shortcut for an application
- Installing a new application
- Creating a Briefcase for synchronizing files between two PCs
- Creating a workgroup post office for use with the Microsoft Exchange e-mail client

A New Help Engine: Accessible and Useful Online Information

Online Help has been completely retooled in Windows 95. It underwent extensive usability testing and the result is a significantly easier-to-use and easier-to-learn Help system. Additionally, it is now dramatically easier for independent software vendors (ISVs) and corporate customers to customize and develop Windows help files. Following are brief descriptions of the major features of the new Help system in Windows 95:

- **Simplified interface.** Help in Windows 3.1 was difficult to learn and to use. It had three main functions: Contents, Search, and Glossary. The Contents view was not well organized and presented, and there was some ambiguity about which functions to use when. Windows 95 behaves much more intuitively and more like a real reference book. It only has two Tabs: Contents and Index.

- **The Contents tab.** Organized like a book's table of contents, the Contents tab displays top level "chapters" (iconically represented as books) from which users can "drill down" to subtopics (iconically represented as pages). Many chapters also have Tips and Tricks subsections that have proved popular in lab testing.
- **Short Help topics.** Topics all fit in one small window, so users don't have to scroll through large, complicated help information.
- **Shortcut buttons.** New shortcut buttons make using Help even easier in Windows 95. Some Help topics contain shortcut buttons, like the one shown in step 1 in Figure 13, that take users to the referenced area in Windows 95. For example, a user who is searching for help on how to change the date on a PC can "jump" right to the Clock Control Panel tool from within Help.

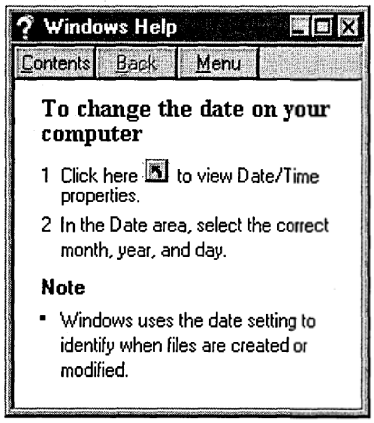


Figure 13. A Help shortcut button (in step 1)

- **What's This? button.** All Control Panel tools in Windows 95 have a ? button at the right end of their title bars. When the user clicks this button, the pointer changes to a question mark. Clicking any object in the dialog box with this pointer brings up a short description of the object. Users can also access the question-mark pointer by right-clicking within a Control Panel tool.

Try It!

Use Help's Shortcut Button to Change the Desktop Color

1. From the *Start* menu, choose *Help*.
2. Select the *Index* tab.
3. Type *display*. Double-click the *background pictures or patterns, changing* topic.

4. Click the *Properties for Display* shortcut button to move directly to this property sheet.

Transition Aids: Easy Migration to the Windows 95 User Interface

Windows 95 provides several aids for helping both users new to Windows and users of Windows 3.1 become productive quickly in the Windows 95 UI. Usability tests indicate that, with little or no additional training, users can complete common tasks under Windows 95 as quickly as they did under Windows 3.1, or even quicker. Windows 95 offers the following self-paced aids:

- **Quick tutorial of Windows 95.** This tutorial walks users through the basics of Windows 95 and covers topics such as starting programs, task switching between open windows, finding information on the local computer, and using the online Help system. The tutorial is designed for both novice users and experienced Windows users and shows how to complete common tasks.
- **Transition aids for Windows 3.1 users.** The Help system provides additional Help topics designed to make the transition to the UI in Windows 95 easier for Windows 3.1 users. These Help topics answer common questions to help users familiar with the components of Windows 3.1 quickly and easily find the respective tools or commands in Windows 95.
- **Readily available information in the online Help system.** The Help system in Windows 95 is designed to provide the information needed to complete the desired task. Topics include How To, Tips and Tricks, and Troubleshooting categories. Help is always available and can be easily accessed by choosing Help from the Start menu.

Powerful Features

Experienced users glean many of the same benefits from the taskbar and the Start button—quickly launching a new program, quickly switching to another task, and so on—as novices. However, experienced users need more, including the following:

- They need a powerful way to browse and manage file hierarchies, whether they are local or not.
- They need to be able to customize the UI to suit their needs and tastes.
- They need to be able to take shortcuts to get tasks done more quickly and efficiently.
- They need to be able to *do* more.

The new UI in Windows 95 definitely enables the experienced user to do more, as the following sections show.

The Windows Explorer: For File Management and Information Browsing

The Windows Explorer, shown in Figure 14, has been described as "File Manager on steroids." It is powerful, flexible, efficient, and extensible. It also solves many of File Manager's fundamental problems, such as having different windows for different drives. For many power users of Windows 95, the Windows Explorer will be the primary interface for navigating through information.

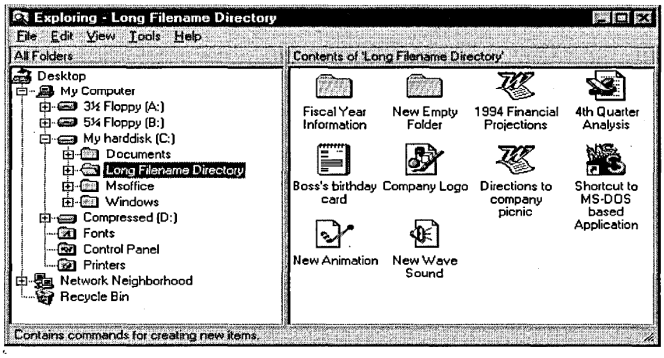


Figure 14. The Windows Explorer

The best way to understand the Windows Explorer is to experience it firsthand. However, here is an overview of its major features:

- **Single view of a world of information.** The Windows Explorer is the eyes of any PC running Windows 95. With it, users can view the entire namespace (all resources, local or connected) from the equivalent of 10,000 feet, or they can zoom down to the equivalent of 10 inches. My Computer and the Network Neighborhood can be browsed and managed, quickly and easily.
- **Flexible and customizable.** Via the Windows Explorer toolbar and the View menu, users can view folder contents in several ways, including large icon, small icon, list, and details views. Folder contents can easily be sorted by name, size, type, and modification date by selecting the column title. Users can also map network drives from the Windows Explorer toolbar.
- **Rich information about objects in details view.** Details view provides a wealth of context-sensitive information about folder contents. For example:
  - Files retain their identifying icons.

- Drive sizes and free space (even mapped network drives) are reported in My Computer.
- Descriptions of Control Panel tools are provided.
- Jobs in the print queue are listed in the Printers folder.
- Comments on other computers in the Network Neighborhood can be viewed.

All the powerful right-click and properties features described in the next two sections are supported in the Windows Explorer.

Try It!

Copy a File to a Different Drive Without Opening a New Window

1. Right-click *My Computer* and choose *Explore*. Maximize the window.
2. Select a file that you want to copy to a network or floppy drive.
3. Move to the left pane in the Windows Explorer and use the + icons to the left of the folder and drive icons to find the network folder to which you want to copy the file. Do not click the destination folder.
4. Go back to the right pane where the file is currently stored and drag or cut/copy/paste the file to the destination folder.

Operations like this one could not be performed in Windows 3.1 without opening two or more File Manager windows.

Right-Click to Create a New Folder

1. In the *Windows Explorer*, right-click an unused space inside a folder in which you want to create a new folder.
2. Choose *New Folder*.

Shortcuts: For Accessing Objects

Shortcuts are an extremely powerful tool for increasing efficiency. They are especially useful in a networked environment. Users can create a shortcut to any object, such as a file, program, network folder, Control Panel tool, or disk drive, and place it anywhere in the UI or in an application. Opening the shortcut opens the object that the shortcut is "pointing" to. For example, if a user creates a shortcut to My Network Folder on a network server and drops the shortcut on the local desktop, opening the shortcut actually opens My Network Folder. Shortcuts are represented by icons that have a small "jump" arrow in the lower-left corner, as shown in Figure 15 on the following page.

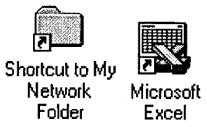


Figure 15. Shortcut icons for a folder and a program

Shortcuts are created by selecting an object and choosing Create Shortcut from the File menu or by right-clicking the object and choosing Create Shortcut. After creation, shortcuts can be renamed. If the shortcut is for an object that was created after installation of Windows 95 and the object is renamed, Windows 95 changes the shortcut definition to reflect the new name. For example, if a user creates a shortcut on the local desktop to \\Server\Share\Public Folder and the folder is subsequently renamed, the shortcut will still work. A shortcut can be deleted without affecting the object to which it points.

Uses for shortcuts are virtually limitless, but the following are some common powerful uses for shortcuts:

- **Shortcuts in the Programs folder.** Shortcuts are an extension of the icons that in Program Manager groups in Windows 3.1. The icons simply pointed to an executable file somewhere in the file system. In Windows 95, the icons that appear on the Start button's Programs menu, which can be customized by choosing Settings and then Start Menu from the Start menu, also appear as shortcuts in the Programs folder. When a shortcut is added to or deleted from the Programs folder, it is also added to or deleted from the Programs menu. As a result, users can keep shortcuts to all favorite programs in one central place, regardless of where the programs are actually stored.
- **Shortcuts on the desktop.** Power users can create shortcuts to commonly accessed files, programs, drives, folders, and utilities right on their desktops. This capability is especially powerful with network resources because no complicated browsing or drive mapping is required to access network folders.
- **Embedded shortcuts in applications.** A shortcut to a large file stored on the network can be dragged to an e-mail message. When the message recipient double-clicks the shortcut, the network file opens. This process is much more efficient than embedding the actual file in an e-mail message because the message is smaller, and embedding shortcuts cuts down on file version proliferation.

Try It!

Discover Where the Start Button's Programs Menu Is Stored

- From the *Start* menu, choose *Settings* and then *Start Menu*. The Start Menu folder, which is a sub-folder of the Windows folder, opens. The Programs folder contains the items that appear on the Start button's Programs menu.

The shortcuts and folders are those that appear on the Programs menu. Adding or deleting shortcuts and folders changes the items that appear on the menu.

Properties: For Customizing All Objects

Property sheets are a pervasive feature in Windows 95. All objects in the UI carry context-sensitive properties that can be accessed and customized by choosing Properties from the File menu or by right-clicking the object and choosing Properties. Good, consistent, easily accessible property sheets have been a favorite of power-user testers. Try the following examples to see how property sheets work.

Try It!

Rename a Hard Drive

1. In the *Windows Explorer* or *My Computer*, right-click to select your hard disk and choose *Properties* to open a property sheet like the one shown in Figure 16.
2. In the *Label* box, type a new name and choose *OK*.
3. From the *View* menu, choose *Refresh*.

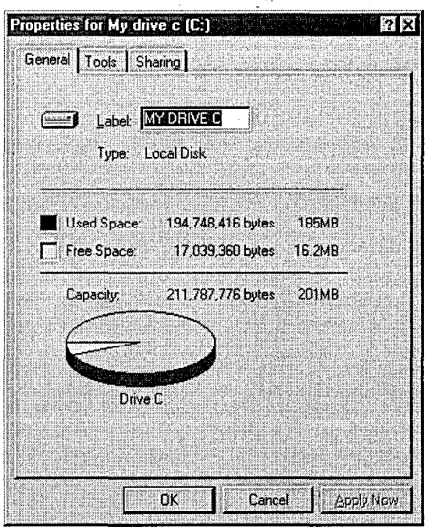


Figure 16. The properties for a disk drive

Share a Folder

1. In the *Windows Explorer*, right-click a folder you want to make available to others on your network and choose *Properties*.
2. Select the *Sharing* tab.
3. Select *Shared As*, and then complete the other fields in this dialog box.

Customize a Shortcut Icon

1. Click the *Start* button and choose *Settings* and *Start Menu*.
2. Open the *Programs* folder.
3. Right-click any shortcut and choose *Properties*.
4. Select the *Shortcut* tab.
5. Click the *Change Icon* button.
6. Select a new icon for the shortcut and choose *OK*.
7. From the *View* menu, choose *Refresh*.

Right-Clicking: For Performing Actions on Objects

Right-clicking, like properties, is a pervasive, context-sensitive feature of Windows 95. (In this book, "right-clicking" refers to clicking the secondary mouse button. Most right-handed people set their mouse options to use the left button as primary and the right button as secondary.) Usability tests have shown that right-clicking as a shortcut way of performing common actions on an object is a very popular power-user feature. However, in general, right-clicking is not a feature that novices discover or remember, so the vast majority of functions that can be performed by right-clicking can also be performed by choosing the corresponding menu commands.

The power of right-clicking can be explored by carrying out the following examples.

Try It!

Right-Click the Desktop to Customize It

1. Right-click a blank space on the desktop.
2. Choose *Properties*.

Minimize or Tile All Open Windows

1. Right-click a blank space on the *Taskbar*.
2. Choose *Minimize All* or *Tile Horizontally*.

3. To undo this operation, right-click a blank space on the *Taskbar* and choose either *Undo Minimize All* or *Undo Tile*.

Create a Shortcut

1. Right-click an object for which you want to create a shortcut.
2. Choose *Create Shortcut*.

Drag a File and Create a Shortcut

1. Right-click and drag a file from the *Windows Explorer* onto the desktop. A menu like the one shown in Figure 17 appears.
2. Choose *Create Shortcut(s) Here*.

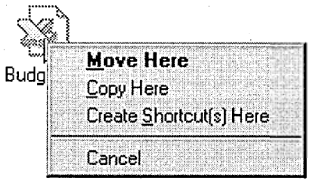


Figure 17. The menu that appears when a file is dragged using the right mouse button

Right-Click a Screensaver to Test It

1. Choose *Find Files or Folders* from the *Start* menu.
2. Type *bezier* and choose *Find Now*.
3. Right-click *Bezier*.
4. Choose *Test*.

Close a Task from the Taskbar

1. Right-click the *Task* button for a window or program you want to close.
2. Choose *Close*.

Access the Property Sheet for an Open Window

1. Right-click the mini-icon in the upper-left corner of any window.
2. Choose *Properties*.

The Control Panel: The Consolidated Control Center

The objective of the Control Panel is to consolidate all command, control, and configuration functions in one location. With Windows 3.1, these functions were difficult to find, use, and remember—for example, video resolution was changed in Windows Setup, but a printer was installed by selecting the Control Panel's Printers icon. As shown in Figure 18, in Windows 95 distinct graphics make all important functions instantly recognizable and previews are offered where appropriate.

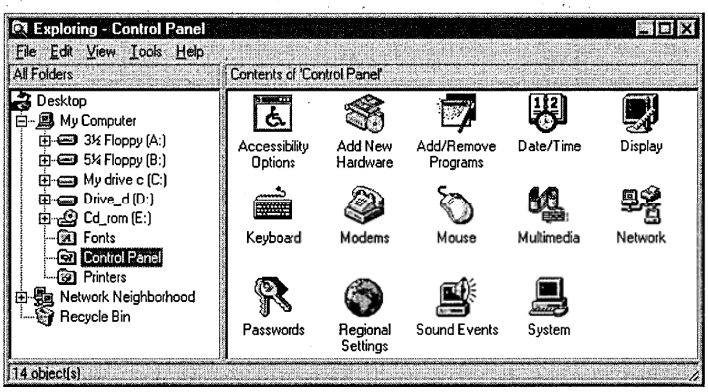


Figure 18. The large icon view of the Control Panel in the Windows Explorer

The individual functions available through the Control Panel tool are discussed in the relevant sections of this book—for example, the Network tool is discussed in Chapter 9, "Networking." However, one Control Panel tool, Display, controls the configuration of the UI in Windows 95 and allows users to customize the UI itself. As shown in Figure 19, its property sheet has the following four tabs:

- **Background.** Allows pattern and wallpaper configuration and preview.
- **Screen Saver.** Allows screen saver configuration and preview.
- **Appearance.** Allows configuration and preview of all of the user interface metrics (fonts, sizes, colors, and so on).
- **Settings.** Allows configuration of monitor resolution and color palette size.

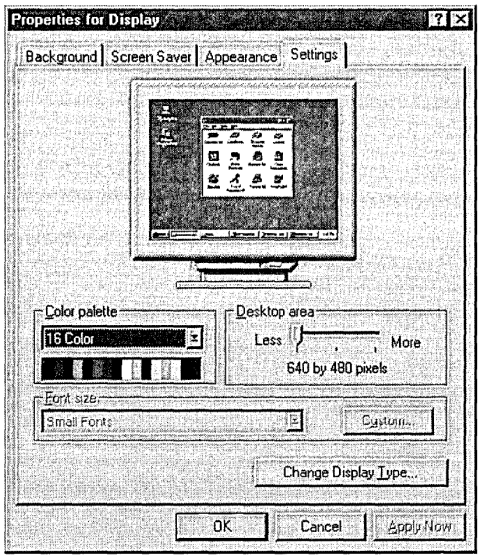


Figure 19. The display properties

Try It!

Switch the Display Resolution

Dynamic resolution switching allows the resolution of the display to be changed without having to restart Windows 95 or reboot the PC. This feature depends on several factors, including the type of video card and the selected color palette.

1. Choose *Settings* and then *Control Panel* from the *Start* menu.
2. Open the *Display* icon.
3. Select the *Settings* tab.
4. Set *Color Palette* to *256 colors* and click *Apply Now* to restart your PC.
5. After your PC restarts, repeat steps 1 and 2 to reopen the Display icon.
6. Choose another video resolution that is supported by your card by sliding the *Desktop Area* slider bar. For example, change the desktop area size from 640 x 480 to 1024 x 768.
7. Click *Apply Now*.
8. Now try playing an AVI clip.

Finding Files or Folders: Easy and Efficient

A powerful new Find utility is built into Windows 95. As shown in Figure 20 and Figure 21, it goes far beyond the minimal functionality of File Manager's Search utility in Windows 3.1.

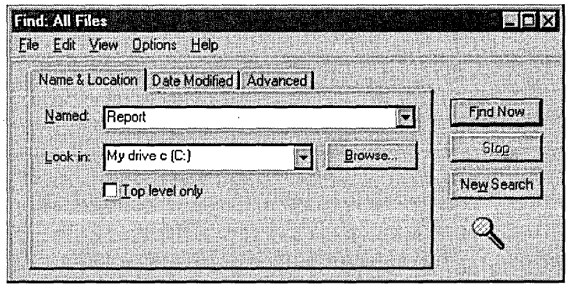


Figure 20. Finding files or folders in Windows 95

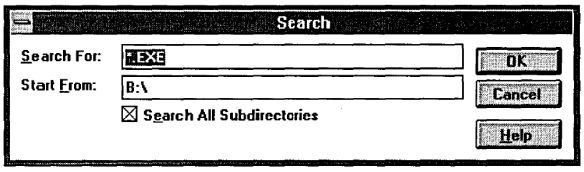


Figure 21. Searching in Windows 3.1

The Find utility includes the following features:

- **Partial name searches.** Type *rep* in the Find Files Named window, and all files and folders with *rep* somewhere in their names are found.
- **Searches on last modification date.** Because files can be searched on their last modification dates, users can perform searches such as *Find all Word documents modified in the last 3 days.*
- **Full-text searches.** Documents containing specified text can be searched for.
- **Saving of search results.** Complex or useful searches can be saved.
- **File management from the search results pane.** Operations such as renaming files or viewing file properties can be carried out within the results pane in the same way as in the Windows Explorer.

Try It!

Save the Results on a Complex Search

1. Click the *Start* button, and choose *Find* and then *Files or Folders*.
2. Type a partial string that you know will be present in many files, such as *rep* or *doc*.
3. Select the *Date Modified* tab.
4. Select *Modified during the previous seven days*.
5. Select the *Advanced* tab.
6. If necessary, select a file type.
7. Click *Find Now*.
8. When the find operation is complete, choose *Save Search* from the *File* menu. (Notice that because the Find feature is 32-bit preemptively multitasked, you have control and can go perform other tasks while Find is running.) A *Find Results* icon is automatically created on the desktop.
9. Double-click the *Find Results* icon.

The Printers Folder: Consolidated Printer Control

The Printers folder, shown in Figure 22, offers one-stop shopping for printer management and configuration. It replaces the troublesome Print Manager and Printers dialog box in the Windows 3.1 Control Panel, which is shown in Figure 23 on the following page.

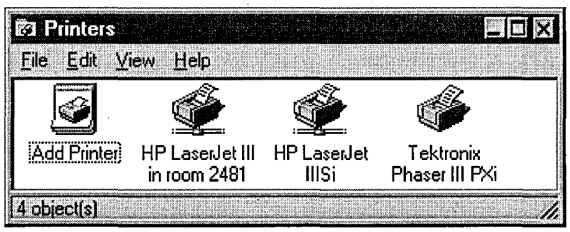


Figure 22. The Printers folder in Windows 95

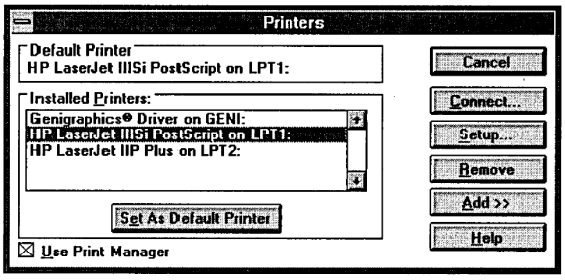


Figure 23. The Printers dialog box in Windows 3.1

The Printers folder is discussed in more detail in Chapter 11, "Printing."

Font Settings: More Powerful Font Management and Preview

The Fonts folder represents a single namespace in which all fonts used in the system can be installed or manipulated. (If any fonts are identified in the WIN.INI file, Windows 95 moves them to the Fonts folder on startup, so all fonts in the system reside in a single location.) Different views of the Fonts folder present additional information about the fonts installed in the system, as shown in Figure 24.

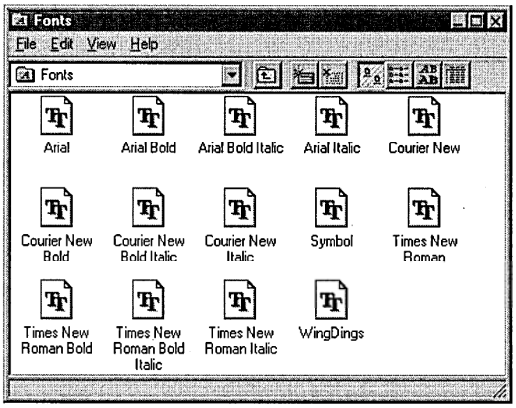


Figure 24. The large icon view of the Fonts folder

Operations can be performed on fonts in the same way they are performed on other file system objects. For example, a font can be removed from the Fonts folder by dragging it to another location, a font can be deleted from the system by deleting it from the

Fonts folder, and a font can be added to the system by dragging it from another location into the Fonts folder.

Try It!

Preview the Fonts

1. Open *My Computer* and double-click the *Fonts* folder, or open the *Fonts* icon in the *Control Panel*.
2. Right-click the font you want to preview.
3. Choose *Open*. Samples of the selected font are displayed and may be printed.

Quick Viewers: Examining Files Without Opening Them

The Quick Viewers allow users to preview a file from the UI without having to open the application that created the file. In fact, users don't even have to have the application that created the file on their system. As a result, documents can be sent over a network or through e-mail. Figure 25 shows a quick view of a Microsoft Excel worksheet.

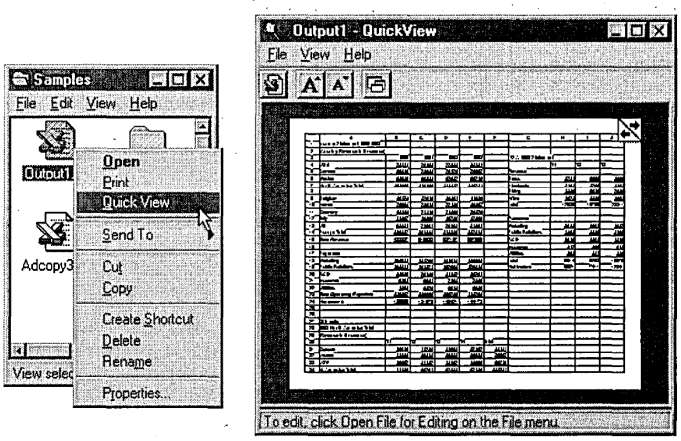


Figure 25. A quick view of a Microsoft Excel worksheet

For more information about the Quick Viewers in Windows 95, see Chapter 20, "Applications and Utilities."

Try It!

Quick-View a File

1. Right-click an icon for a file created by a registered application—for example, a bitmap, a text file, or a WordPad document.
2. Choose *Quick View*.

Compatibility

Compatibility is a requirement for Windows 95. It is a no-excuses, "no-brainer" upgrade from Windows 3.1. Overall, compatibility is most important for third-party software and hardware. However, it also applies to the UI. The UI in Windows 95 must be compatible with the way current Windows and MS-DOS users work, and it must scale itself to the level and preferences of individual users.

For Users of Windows 3.1

Of primary importance is that new UI features be easy for current Windows 3.1 users to learn at their own pace. In addition, UI visual elements and operations in Windows 95 must be consistent, to the extent possible, with the elements users are already familiar with in Windows 3.1. In addition to providing aids for users migrating from Windows 3.1, Windows 95 also includes tools familiar to Windows 3.1 users. For example, the system menu in the upper-left corner of most windows, and keyboard shortcuts such as ALT+F4, ALT+TAB, CTRL+X, CTRL+C, and CTRL+V, are present in Windows 95, easing the requirement for relearning or retraining.

With minimal changes in appearance, Program Manager and File Manager run under Windows 95 and are easily accessible via the Start button. As of this writing, the designs for access and default configuration of these Managers is not yet set. For example, when a user who has upgraded boots Windows 95 for the first time, the Program Manager window might be open. Or the Start button might have a Windows 3.1 Compatibility menu item that will launch Program Manager and File Manager. Regardless of the final design, many help and learning devices will be specifically designed for the upgrade, such as a Click Here to Begin arrow that zooms to the Taskbar when the user first boots Windows 95.

For Users of MS-DOS

Users of the command line in MS-DOS won't have to give it up when they move to the graphical UI of Windows 95. In fact, "command-line junkies" will find that the usability and power of the MS-DOS command prompt have been dramatically improved. New command-line functionality includes the following:

- Launching Windows–based applications

- Opening documents from a command-line
- Scaling the size of the MS-DOS command prompt
- Using cut/copy/paste operations to integrate information from MS-DOS and Windows–based applications
- UNC pathname support

See Chapter 6, "Support for Running MS-DOS–based Applications," for more information.

CHAPTER 4

Base System Architecture

Ease on the surface requires power and speed at the core, and the modern, 32-bit architecture of Windows 95 meets these requirements. Freed from the limitations of MS-DOS, Windows 95 preemptively multitasks for better PC responsiveness—for example, users no longer have to wait while the system copies files—and delivers increased robustness and protection for applications. Windows 95 also provides the foundation for a new generation of easier, more powerful multithreaded 32-bit applications. And most important, Windows 95 delivers this power and robustness on today's average PC platform while scaling itself to take advantage of additional memory and CPU cycles.

The mission of Windows 95 is to deliver a complete, integrated operating system that offers modern, 32-bit operating system technology and includes built-in connectivity support. In addition to the high-level mission of Windows 95, market requirements dictate delivery of a high performance, robust, and completely backward-compatible operating system that provides a platform for a new generation of applications.

This chapter discusses the base system architecture used by Windows 95. The base architecture covers low-level system services for managing memory, accessing disk devices, and providing robust support for running applications.

Summary of Improvements over Windows 3.1

Improvements made to the base architecture of Windows 95 result in many benefits to users. Following is a summary of some of the key improvements:

- **A fully integrated 32-bit protected-mode operating system.** The need for a separate copy of MS-DOS has been eliminated.
- **Preemptive multitasking and multithreading support.** System responsiveness and smooth background processing have been improved.

- **32-bit installable file systems.** Systems such as VFAT, CDFS, and network redirectors provide better performance, use of long filenames, and an open architecture that supports future growth.
- **32-bit device drivers.** Available throughout the system, these drivers deliver improved performance and intelligent memory use.
- **A complete 32-bit kernel.** Included are memory management, scheduler, and process management.
- **Improved system-wide robustness and cleanup.** This more stable and reliable operating environment also cleans up after an application ends or crashes.
- **More dynamic environment configuration.** The need for users to tweak their systems is reduced.
- **Improved system capacity.** Included are better system resource limits to address the problems Windows 3.1 users encountered when running multiple applications.

A Fully Integrated Operating System

The first thing that users of Windows 3.1 and MS-DOS will notice when they turn on their computers is the lack of an MS-DOS command prompt from which they would formerly have invoked Windows. Windows 95 is a tightly integrated operating system that features a preemptive multitasking kernel which boots directly into the graphical UI and also provides full compatibility with the MS-DOS operating system.

Many of the components of Windows 95 overcome limitations inherent in MS-DOS and Windows 3.1. However, these improvements do not come at the cost of compatibility with existing software, hardware, or computing environments.

A Preemptive Multitasking Operating System

The job of the operating system is to provide services to the applications running on the system and, in a multitasking environment, to provide support that allows more than one application to run concurrently. In Windows 3.1, multiple applications ran concurrently in a *cooperative* multitasking manner. The Windows 3.1 operating system required an application to check the message queue every once in a while to allow the operating system to give control to other running applications. Applications that did not check the message queue on a frequent basis effectively hogged all the CPU time and prevented switching to another running task.

Windows 95 uses a *preemptive* multitasking mechanism for running Win32–based applications, and the operating system takes control away from or gives control to another running task depending on the needs of the system. Unlike Win16–based applications, Win32–based applications do not need to *yield* to other running tasks in order to multitask in a friendly manner. (Win16–based applications are still cooperatively multitasked for compatibility reasons.) Windows 95 provides a

mechanism called *multithreading* that allows Win32–based applications to take advantage of the preemptive multitasking nature of the operating system and that facilitates concurrent application design. In operating-system terms, a running Win32–based application is called a *process*. Each process consists of at least a single *thread*. A thread is a unit of code that can get a time slice from the operating system to run concurrently with other units of code. It must be associated with a process, and it identifies the code path flow as the process is run by the operating system. A Win32-based application can *spawn* (or initiate) multiple threads for a given process. Multiple threads enhance the application for the user by improving throughput and responsiveness and aiding background processing.

Because of the preemptive multitasking nature of Windows 95, threads of execution allow background code to be processed in a smooth manner. For example, a word processing application (process) can implement multiple threads to enhance operation and simplify interaction with the user. The application might have one thread of code responding to the keys pressed on the keyboard by the user to enter characters in a document, while another thread is performing background operations such as spell-checking or pagination, and yet another thread is spooling a document to the printer in the background.

Some available Windows 3.1 applications provide functionality similar to that just described. However, because Windows 3.1 does not provide a mechanism for supporting multithreaded applications, the application developer has to implement a threading scheme. The use of threads in Windows 95 facilitates the adding of asynchronous processing of information to applications by their developers.

Applications that use multithreading techniques can also take advantage of improved processing performance available from a symmetric multiprocessing (SMP) system running Windows NT, which allows different portions of the application code to run on different processors simultaneously. (Windows NT uses a thread as the unit of code to schedule symmetrically among multiple processors.)

For information about how Windows 95 runs MS-DOS–based applications in a preemptive manner (as Windows 3.1 does today), Win16–based applications in a cooperative manner (as Windows 3.1 does today), and Win32–based applications in a preemptive manner (as Windows NT does today) see later sections in this chapter.

No CONFIG.SYS or AUTOEXEC.BAT?

Windows 95 doesn't need the separate CONFIG.SYS or AUTOEXEC.BAT file required by MS-DOS and Windows 3.1. Instead, Windows 95 is intelligent about the drivers and settings it requires and automatically loads the appropriate driver files or makes the appropriate configuration settings during its boot process. If a CONFIG.SYS or AUTOEXEC.BAT file is present, the settings in these files are used to set the global environment. For example, the default search path or the default appearance of the command prompt can be defined by using the appropriate entries in the AUTOEXEC.BAT file. While Windows 95 itself does not need a CONFIG.SYS

or AUTOEXEC.BAT file, compatibility is maintained with existing software or environments that may require one or both of these files.

No MS-DOS?

Unlike Windows 3.1, Windows 95 is not dependent on real-mode operating system components for its interaction with the file system. However, the Windows 95 boot sequence does begin by loading real-mode operating system components that are compatible with MS-DOS. During the boot sequence, support for loading any real-mode drivers and TSRs that are identified in a CONFIG.SYS or AUTOEXEC.BAT file is processed. Because these drivers explicitly look for or use MS-DOS application support, the real-mode operating system components of Windows 95 help maintain compatibility with software that users already have on their system. After the real-mode drivers are loaded, Windows 95 begins loading the protect-mode operating system components. In some cases where a protect-mode Windows–based driver is provided, Windows 95 actually removes real-mode drivers from memory. More information about this subject is given later.

32-Bit Versus 16-Bit Components

To provide a good balance between delivering compatibility with existing applications and drivers, decreasing the size of the operating system working set, and offering improved system performance over Windows 3.1, Windows 95 uses a combination of 32-bit and 16-bit code. In general, 32-bit code is provided in Windows 95 to maximize the performance of the system, while 16-bit code balances the requirements for reducing the size of the system and maintaining compatibility with existing applications and drivers. System reliability is also improved without a cost in terms of compatibility or increased size.

The design of Windows 95 deploys 32-bit code wherever it significantly improves performance without sacrificing application compatibility. Existing 16-bit code is retained where it is required to maintain compatibility, or where 32-bit code would increase memory requirements without significantly improving performance. All of the I/O subsystems and device drivers in Windows 95, such as networking and file systems, are fully 32-bit, as are all the memory management and scheduling components (the kernel and virtual memory manager). Figure 26 depicts the relative distribution of 32-bit code versus 16-bit code present in Windows 95 for system-level services.

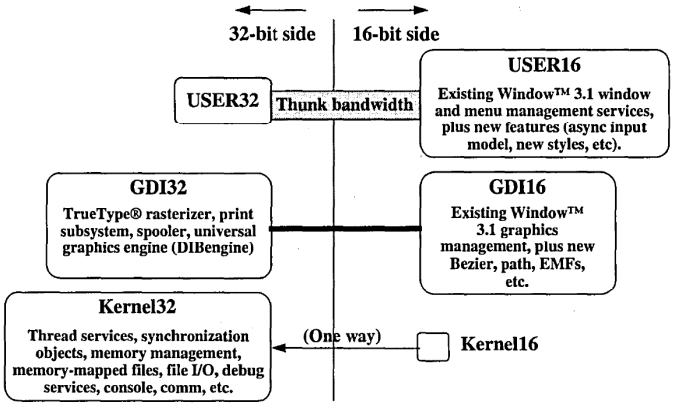


Figure 26. The relative code distribution in Windows 95

As shown in the figure, the lowest-level services provided by the operating system kernel are provided as 32-bit code. Most of the remaining 16-bit code consists of hand-tuned assembly language, delivering performance that rivals some 32-bit code used by other operating systems available on the market today. Many functions provided by the Graphics Device Interface (GDI) have been moved to 32-bit code, including the spooler and printing subsystem, the font rasterizer, and the drawing operations performed by the graphics DIB engine. Much of the window management code (User) remains 16-bit to retain application compatibility.

In addition, Windows 95 improves upon the MS-DOS and Windows 3.1 environments by implementing many device drivers as 32-bit protected-mode code. Virtual device drivers in Windows 95 assume the functionality provided by many real-mode MS-DOS–based device drivers, eliminating the need to load them in MS-DOS. This technique results in a minimal conventional-memory footprint, improved performance, and improved reliability and stability of the system over MS-DOS–based device drivers.

Virtual Device Drivers

A virtual device driver is a 32-bit, protected-mode driver that manages a system resource, such as a hardware device or installed software, so that more than one application can use the resource at the same time. To understand the improvements available in Windows 95 over the combination of MS-DOS and Windows 3.1, it helps to have a basic understanding of what virtual device drivers (VxDs) are and the role they play in the Windows 95 environment.

The term *VxD* refers to a general virtual device driver, with *x* representing the type of device driver. For example, VDD is a virtual device driver for a display device, a VTD is a virtual device driver for a timer device, a VPD is a virtual device driver for a

printer device, and so on. Windows uses virtual devices to support multitasking for MS-DOS–based applications, virtualizing the different hardware components on the system to make it appear to each MS-DOS virtual machine (VM) that it is executing on its own computer. Virtual devices work in conjunction with Windows to process interrupts and carry out I/O operations for a given application without disrupting how other applications run.

Virtual device drivers support all hardware devices for a typical computer, including the programmable interrupt controller (PIC), timer, direct-memory-access (DMA) device, disk controller, serial ports, parallel ports, keyboard device, math coprocessor, and display adapter. A virtual device driver can contain the device-specific code needed to carry out operations on the device. A virtual device driver is required for any hardware device that has settable operating modes or retains data over any period of time. In other words, if the state of the hardware device can be disrupted by switching between multiple applications, the device must have a corresponding virtual device. The virtual device keeps track of the state of the device for each application and ensures that the device is in the correct state whenever an application continues.

Although most virtual devices manage hardware, some manage only installed software, such as an MS-DOS device driver or a terminate-and-stay-resident (TSR) program. Such virtual devices often contain code that either emulates the software or ensures that the software uses only data applicable to the currently running application. ROM BIOS, MS-DOS, MS-DOS device drivers, and TSRs provide device-specific routines and operating system functions that applications use to indirectly access the hardware devices. Virtual device drivers are sometimes used to improve the performance of installed software—for example, the 80386 and compatible microprocessors can run the 32-bit protected-mode code of a virtual device more efficiently than the 16-bit real-mode code of an MS-DOS device driver or TSR. In addition, performance is enhanced by eliminating ring transitions that result in executing 32-bit applications that access 16-bit real-mode services, because with virtual device drivers, the system can stay in protected-mode.

Windows 95 benefits from providing more device driver support implemented as a series of VxDs in the Windows environment, instead of using the device drivers previously available as real-mode MS-DOS device drivers. Functionality that was previously supported as MS-DOS device drivers but is now supported as VxDs in Windows 95 includes the following components:

- MS-DOS FAT file system
- SmartDrive
- CD-ROM file system
- Network card drivers and network transport protocols
- Network client redirector and network peer server
- Mouse driver
- MS-DOS SHARE.EXE TSR
- Disk device drivers including support for SCSI devices
- DriveSpace (and DoubleSpace) disk compression

In summary, in Windows 95 VxDs provide the following advantages:

- Improved performance as a result of a 32-bit code path and the elimination or reduction of the need to switch between real and protected mode
- Reduced conventional memory footprint by providing device driver and TSR functionality as protected-mode components that reside in extended memory
- Improved system stability and reliability compared to MS-DOS device driver counterparts

Virtual device drivers in Windows 95 can be identified by .VXD extensions, and virtual device drivers from Windows 3.1 can be identified by .386 extensions.

The System Architecture Layout in Windows 95

Figure 27 illustrates the layout of the base system architecture for Windows 95. Components of the system are divided between Ring 0 and Ring 3 code, offering different levels of system protection. The Ring 3 code is protected from other running processes by protection services provided by the Intel processor architecture. The Ring 0 code consists of low-level operating system services such as the file system and the virtual machine manager.

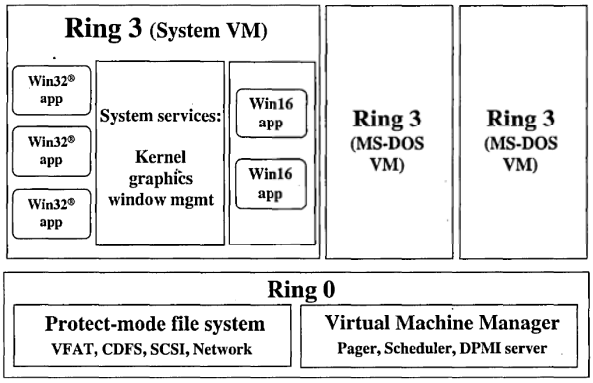


Figure 27. The integrated architecture of Windows 95, which supports running MS-DOS–based, Win16–based, and Win32–based applications

Figure 27 also depicts the way that MS-DOS–based, Win16–based, and Win32–based applications run in the system. The following section discusses the provisions that the system makes for running these applications.

Support for Win16–Based Applications

Win16–based (16-bit) applications run together within a unified address space and run in a cooperatively multitasking manner, as they do under Windows 3.1. Win16–based applications benefit from the preemptive multitasking of other system components, including the 32-bit print and communications subsystem and the improvements made in system robustness and protection from the system kernel in Windows 95.

Based on customer needs, resource needs, and market needs, three goals drove the architectural design of Win16–based application support: compatibility, size, and performance. Functionality adjustments, such as preemptively running Win16–based applications together in the Win16 subsystem or running Win16–based applications in separate VMs, were considered, but each of the options considered failed to meet the three design goals. The following discussion provides some insight into the architecture of Windows 95 as far as running Win16–based applications in a fast, stable, and reliable way is concerned.

Compatibility

First and foremost, Windows 95 needs to run existing Win16–based applications without modification. This factor is extremely important to existing users who want to take advantage of the new functionality offered in Windows 95, such as 32-bit networking, but don't want to have to wait until new Windows 95–enabled applications are available on the market.

Windows 95 builds upon the Windows 3.1 platform to provide support for running existing Win16–based applications and using existing Windows–based device drivers, while providing support for the next generation of 32-bit applications and components. Windows 95 extends the Windows 3.1 architecture in areas that have little or no impact on compatibility, as well as enhances the architecture to deliver a faster, more powerful 32-bit operating system.

Size

While many newer computer purchases are Intel 80486–based computers with 4 MB or 8 MB (or more) of memory, a high percentage of 80386DX–based computers with 4 MB of memory running Windows 3.1 are still in use. To support the needs of the market, Windows 95 must run on a base platform of an Intel 80386DX–based computer with 4 MB of RAM and provide access to its new features and functionality without requiring an upgrade of existing hardware or the addition of more RAM.

To meet its goals, Windows 95 is designed to occupy a working set of components no larger than Windows 3.1, thereby ensuring that any Win16–based application running at a perceived speed on a 4 MB or 8 MB (or greater) computer runs at the same (or higher) speed under Windows 95 without suffering any performance degradation. To meet the size goals of Windows 95, Win16–based applications run within a unified

address space, resulting in little overhead beyond that required by Windows 3.1 to support the running of Windows–based applications. Running in a unified address space allows Windows 95 not only to fit on a 4 MB computer, but also to perform well. The architecture of Windows 95 includes innovative design features, such as dynamically loadable VxDs, to decrease the working set of components and memory requirements used by the operating system.

Meeting the size design goal (as well as meeting the compatibility goal) precluded the strategies of running Win16–based applications in a separate VM (by running a separate copy of Windows 3.1 on top of the operating system, which would involve paying a several megabyte "memory tax" for each application) as OS/2 does, or of emulating Windows 3.1 on top of the Win32 subsystem (which would also involve paying a "memory tax" for running Win16–based applications) as Windows NT does.

Running Win16–based applications in separate VMs is very expensive memory-wise. This strategy would require separate GDI, USER, and KERNEL code in each VM that is created, increasing the working set by as much as 2 MB for each Win16–based application that is running (as is the case with OS/2 for Windows). On a computer with 16 MB or more, this increase may not appear significant. However, bearing in mind the existing installed base of computers, running Win16–based applications in their own separate VMs in 4 MB of memory is impossible, and running them in 8 MB with the level of performance observed and expected under Windows 3.1 is very difficult.

Performance

Users expect their existing Win16–based applications to run as fast as or faster than they do under Windows 3.1. Both Win16–based applications and MS-DOS–based applications benefit from the 32-bit architecture of Windows 95, including the increased use of 32-bit device driver components and 32-bit subsystems.

Win16–based applications run within a unified address space and interact with the system much as they do under Windows 3.1. Running Win16–based applications in separate VMs requires either mapping Win16 system components in each address space, as Windows NT does, or providing a separate copy of each system component in each address space, as OS/2 for Windows does. The additional memory overhead required for Win16 system components in each VM to run a Win16–based application has a negative impact on system performance.

Windows 95 balances the issue of system protection and robustness with the desire for better system performance and improves on the system robustness of Windows 3.1. The improvements in this area are briefly discussed in the next section and are described in greater detail in Chapter 5, "Robustness."

Protection

The support for running Win16–based applications provides protection of the system from other running MS-DOS–based applications or Win32–based applications. Unlike

Windows 3.1, an errant Win16–based application cannot easily bring down the system or other running processes on the system. While Win32–based applications benefit the most from system memory protection, the robustness improvements in Windows 95 result in a more stable and reliable operating environment than Windows 3.1.

Win16–based applications run within a unified address space and cooperatively multitask as they do under Windows 3.1. The improvements made to overall system-wide robustness greatly enhance the system's ability to recover from an errant application, and improved cleanup of the system lessens the likelihood of application errors. General protection faults (GPFs) under Windows 3.1 are most commonly caused by an application overwriting its own memory segments, rather than by an application overwriting memory belonging to another application. Windows 3.1 did not recover gracefully when a Windows–based application crashed or hung. When a GPF caused the system to halt an application, the system commonly left allocated resources in memory, causing the system to degenerate.

Because of improved protection in Windows 95, an errant Win16–based application cannot easily bring down either the system as a whole or other running MS-DOS or Win32–based applications. At most, it can impact other running Win16–based applications.

Other protection improvements include the use of separate message queues for each running Win32–based application. The use of a separate message queue for the Win16 address space and for each running Win32–based application provides better recovery of the system and doesn't halt the system if a Win16–based application hangs.

Robustness Improvements

System robustness when running Win16–based applications under Windows 95 is greatly improved over Windows 3.1. Windows 95 now tracks resources allocated by Win16–based applications and uses the information to clean up the system after an application exits or ends abnormally, thus freeing up unused resources for use by the rest of the system.

Robustness improvements are discussed in Chapter 5, "Robustness."

Support for MS-DOS–Based Applications

Windows 95 includes many improvements over Windows 3.1 for running MS-DOS–based applications. As with Windows 3.1, each MS-DOS–based application runs in its own VM. A VM takes advantage of the Intel 80386 (and higher) architecture, which allows multiple 8086-compatible sessions to run on the CPU and thereby allows existing MS-DOS applications to run preemptively with the rest of the system. As with Windows 3.1, the use of virtual device drivers provides common regulated access to hardware resources, causing each application running in a VM to think that it is

running on its own individual computer and allowing applications not designed to multitask to run concurrently with other applications.

Windows 95 provides a flexible environment for running MS-DOS–based applications. In Windows 3.1, users sometimes needed to exit Windows to run MS DOS–based applications that were either ill-behaved or required direct access to system resources. MS-DOS–based application compatibility is improved in Windows 95 to the point that almost all MS-DOS–based applications should run under Windows 95.

A detailed discussion of the improvements made to the support for running MS-DOS–based applications within the Windows environment is provided in Chapter 6, "Support for Running MS-DOS–based Applications."

Protection

In Windows, VMs are fully protected from one another, as well as from other applications running on the system. This protection prevents errant MS-DOS–based applications from overwriting memory occupied or used by system components or other applications. If an MS-DOS–based application attempts to access memory outside of its address space, the system notifies the user and terminates the MS-DOS–based application.

Robustness Improvements

System robustness is greatly improved when running MS-DOS–based applications in Windows 95. Robustness is discussed in Chapter 5, "Robustness."

Support for Win32–Based Applications

Win32–based applications can fully exploit and benefit significantly from the design of the Windows 95 architecture. In addition, each Win32–based application runs in its own fully protected, private address space. This strategy prevents Win32–based applications from crashing each other, from crashing running MS-DOS–based applications, from crashing running Win16–based applications, or from crashing the Windows 95 system as a whole.

Win32–based applications feature the following benefits over Win16–based applications in Windows 95 and over Windows 3.1:

- Preemptive multitasking
- Separate message queues
- Flat address space
- Compatibility with Windows NT
- Long filename support

- Memory protection
- Robustness improvements

Preemptive Multitasking

Unlike the cooperative multitasking used by Win16–based applications under Windows 3.1, 32-bit Win32–based applications are preemptively multitasked in Windows 95. The operating system kernel is responsible for scheduling the time allotted for running applications in the system, and support for preemptive multitasking results in smoother concurrent processing and prevents any one application from utilizing all system resources without permitting other tasks to run.

Win32–based applications can optionally implement threads to improve the granularity at which they multitask within the system. The use of threads by an application improves the interaction with the user and results in smoother multitasking operation.

Separate Message Queues

Under Windows 3.1, the system uses the point when an application checks the system message queue as the mechanism to pass control to another task, allowing that task to run in a cooperative manner. If an application doesn't check the message queue on a regular basis, or if the application hangs and thus prevents other applications from checking the message queue, the system keeps the other tasks in the system suspended until the errant application ends.

Each Win32–based application has its own message queue and is thus not affected by the behavior of other running tasks on their own message queues. If a Win16–based application hangs, or if another running Win32–based application crashes, a Win32–based application continues to run preemptively and can still receive incoming messages or event notifications.

Message queues are discussed in more detail in Chapter 5, "Robustness."

Flat Address Space

Win32–based applications benefit from improved performance and simpler construct because they can access memory in a linear fashion, rather than being limited to the segmented memory architecture used by MS-DOS and Windows 3.1. To provide a means of accessing high amounts of memory using a 16-bit addressing model, the Intel CPU architecture provides support for accessing 64K chunks of memory, called *segments*, at a time. Applications and the operating system suffer a performance penalty under this architecture because of the manipulations required by the processor for mapping memory references from the segment/offset combination to the physical memory structure.

The use of a flat address space by the 32-bit components in Windows 95 and by Win32–based applications allows application and device driver developers to write

software without the limitations or design issues inherent in the segmented memory architecture used by MS-DOS and Windows 3.1.

Compatibility with Windows NT

Win32–based applications that exploit Win32 APIs common to Windows 95 and Windows NT can run without modification on either platform on Intel–based computers. The commonality of the Win32 API provides a consistent programmatic interface and allows application developers to leverage a single development effort to deliver software that runs on multiple platforms. It also provides scaleability of applications and broadens the base of platforms available for running ISV or custom applications with minimal additional effort.

Application developers are encouraged to develop applications either under Windows 95 or under Windows NT and to test compatibility on both platforms.

Long Filename Support

Win32–based applications that call the file I/O functions supported by the Win32 API benefit from the ability to support and manipulate filenames of up to 255 characters with no additional development effort. To ease the burden of the application developer, the Win32 APIs and common dialog support handle the work of manipulating long filenames, and the file system provides compatibility with MS-DOS and other systems by automatically maintaining the traditional 8.3 filename.

Memory Protection

Each Win32–based application runs in its own private address and is protected by the system from other applications or processes that are running in the system. Unlike errant Win16–based applications under Windows 3.1, errant Win32–based applications under Windows 95 end only themselves, instead of bringing down the entire system if they attempt to access memory belonging to another application.

The use of separate message queues for Win32–based applications also ensures that the system continues to run if an application hangs or stops responding to messages or events.

Robustness Improvements

Win32–based applications benefit from the highest level of system robustness supported under Windows 95. Resources allocated for each Win32–based application are tracked on a per-thread basis and are automatically freed when the application ends. If an application hangs, users can perform a *local reboot* operation to end the hung application without affecting other running tasks, and the system then cleans up properly.

Detailed information about robustness enhancements is given in Chapter 5, "Robustness."

32-Bit File System Architecture

The file system in Windows 95 has been redesigned to support the characteristics and needs of the multitasking nature of its kernel. The changes present in Windows 95 provide many benefits to users and have the following results:

- **Improved ease of use.** Ease of use is improved by the support of long filenames because users no longer need to reference files by the MS-DOS 8.3 filename structure. Instead they can use up to 255 characters to identify their documents. Ease of use is also improved by hiding the filename extensions.
- **Improved performance.** As in Windows for Workgroups 3.11, file I/O performance is improved dramatically over Windows 3.1 by featuring 32-bit protected-mode code for reading information from and writing information to the file system, reading from and writing to the disk device, and intelligent 32-bit caching mechanisms (a full 32-bit code path is available from the file system to the disk device).
- **Improved system stability and reliability.** File system components implemented as 32-bit protected-mode device drivers offer improved system stability and reliability over their MS-DOS device driver counterparts because they can remain in protected mode for code execution and because they leverage existing driver technology first implemented in Windows NT and also available in Windows for Workgroups 3.11.

Architecture Overview

Windows 95 features a layered file system architecture that supports multiple file systems and provides a protected-mode path from the application to the media device, resulting in improved file and disk I/O performance over Windows 3.1. The following features are included in the new file system architecture:

- Win32 API support
- Long filename support
- 32-bit FAT file system
- 32-bit CD-ROM file system
- Dynamic system cache for file and network I/O
- Open architecture for future system support
- Disk device driver compatibility with Windows NT

Figure 28 depicts the file system architecture used by Windows 95.

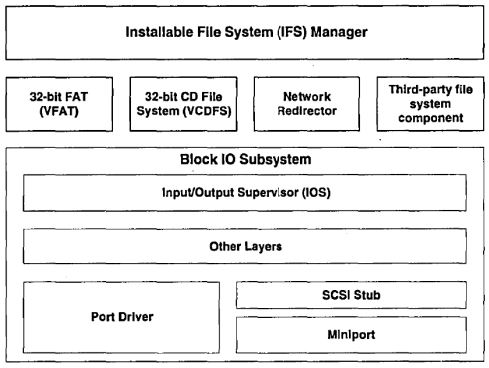


Figure 28. The file system architecture

The file system architecture in Windows 95 is made up of the following components:

- **Installable File System (IFS) Manager.** The IFS Manager is responsible for arbitrating access to different file system components.
- **File system drivers.** The file system drivers layer includes access to file allocation table (FAT)–based disk devices, CD-ROM file systems, and redirected network device support.
- **Block I/O subsystem.** The block I/O subsystem is responsible for interacting with the physical disk device.

Components of each of these layers are examined in the next three sections.

The Installable File System Manager

Under MS-DOS and Windows 3.1, the MS-DOS Int 21h interrupt is responsible for providing access to the file system to manipulate file information on a disk device. To support redirected disk devices, such as a network drive or a CD-ROM drive, other system components, such as the network redirector, would hook the Int 21h function so that it could examine a file system request to determine whether it should handle the request or the base file system should. Although this mechanism provided the ability to add additional device drivers, some add-on components were ill-behaved and interfered with other installed drivers.

Another problem with the MS-DOS–based file system was the difficulty in supporting the loading of multiple network redirectors to provide concurrent access to different network types. Windows for Workgroups provided support for running the Microsoft network redirector at the same time as an additional network redirector, such as Novell NetWare, Banyan VINES, or SUN PC-NFS. However, support for running more than two network redirectors at the same time was not provided.

The key to friendly access to disk and redirected devices in Windows 95 is the Installable File System (IFS) Manager. The IFS Manager is responsible for arbitrating access to file system devices, as well as other file system device components.

File System Drivers

Windows 95 includes support for the following file systems:

- 32-bit file allocation table (VFAT) driver
- 32-bit CD-ROM file system (CDFS) driver
- 32-bit network redirector for connectivity to Microsoft network servers, such as Windows NT Server, along with a 32-bit network redirector to connect to Novell NetWare servers

In addition, third parties will use the IFS Manager APIs to provide a clean way of concurrently supporting multiple device types and adding additional disk device support and network redirector support.

The 32-Bit Protected-Mode FAT File System

The 32-bit VFAT driver provides a 32-bit protected-mode code path for manipulating the file system stored on a disk. It is also re-entrant and multithreaded, providing smoother multitasking performance. The 32-bit file access driver is improved over that provided originally with Windows for Workgroups 3.11 and is compatible with more MS-DOS-device drivers and hard disk controllers.

Benefits of the 32-bit file access driver over MS-DOS–based driver solutions include the following:

- Dramatically improved performance and real-mode disk caching
- No conventional memory used (replacement for real-mode SmartDrive)
- Better multitasking when accessing information on disk with no blocking
- Dynamic cache support

Under MS-DOS and Windows 3.1, manipulation of the FAT and writing to or reading from the disk is handled by the Int 21h MS-DOS function and is 16-bit real-mode code. Being able to manipulate the disk file system from protected mode removes or reduces the need to transition to real mode in order to write information to the disk through MS-DOS, which results in a performance gain for file I/O access.

The 32-bit VFAT driver interacts with the block I/O subsystem to provide 32-bit disk access to more device types than are supported by Windows 3.1. Support is also provided for mapping to existing real-mode disk drivers that may be in use on a user's system. The combination of the 32-bit file access and 32-bit disk access drivers results in significantly improved disk and file I/O performance.

The 32-Bit Cache

The 32-bit VFAT works in conjunction with a 32-bit protected-mode cache (VCACHE) driver and replaces and improves on the 16-bit real-mode SmartDrive disk cache software provided with MS-DOS and Windows 3.1. The VCACHE driver features a more intelligent algorithm for caching information read from or written to a disk drive than SmartDrive, and results in improved performance when reading information from cache. The VCACHE driver is also responsible for managing the cache pool for the CD-ROM File System (CDFS) and the provided 32-bit network redirectors.

Another big improvement VCACHE provides over SmartDrive is that the memory pool used for the cache is dynamic and is based on the amount of available free system memory. Users no longer need to statically allocate a block of memory to set aside as a disk cache because the system automatically allocates or deallocates memory used for the cache based on system use. Because of intelligent cache use, the performance of the system also scales better than with Windows 3.1 or Windows for Workgroups 3.11.

The 32-Bit Protected-Mode CD-ROM File System

The 32-bit protected-mode CD-ROM file system (CDFS) implemented in Windows 95 provides improved CD-ROM access performance over the real-mode MSCDEX driver in Windows 3.1 and is a full 32-bit ISO 9660 CD file system. The CDFS driver replaces the 16-bit real-mode MSCDEX driver and features 32-bit protected-mode caching of CD-ROM data. The CDFS driver cache is dynamic and shares the cache memory pool with the 32-bit VFAT driver, requiring no configuration or static allocation on the part of the user.

Benefits of the new 32-bit CDFS driver include the following:

- No conventional memory used (replaces real-mode MSCDEX)
- Improved performance over MS-DOS–based MSCDEX and real-mode cache
- Better multitasking when accessing CD-ROM information, with no blocking
- Dynamic cache support to provide a better balance between providing memory to run applications versus memory to serve as a disk cache

If MSCDEX is specified in the AUTOEXEC.BAT, the 32-bit CDFS driver takes over the role played by the MSCDEX driver and communicates with the CD-ROM device. The use of MSCDEX is no longer necessary under Windows 95.

Users of CD-ROM multimedia applications benefit greatly from the new 32-bit CDFS. Their multimedia applications run smoother and information is read from the CD-ROM quicker, providing improved performance.

The Block I/O Subsystem

The block I/O subsystem in Windows 95 improves upon the 32-bit disk access fast-disk device architecture in Windows 3.1 and therefore improves performance for the entire file system and a broader array of device support.

As shown in Figure 29, the components of the block I/O subsystem include the high-level I/O Supervisor (IOS) layer, which provides an interface to the block I/O subsystem for the higher layer components; the port driver, which represents a monolithic disk device driver; the SCSI layer, which provides a standard interface and driver layer to provide device-independent control code for SCSI devices; and the SCSI mini-port driver, which contains the device-dependent control code responsible for interacting with individual SCSI controllers.

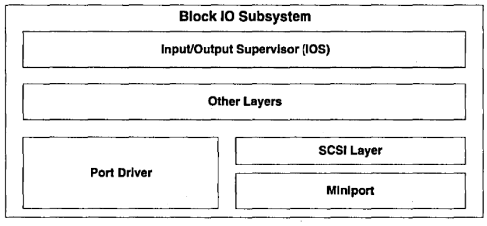


Figure 29. The architecture of the block I/O subsystem

The block I/O subsystem provides the following support in Windows 95:

- A fully Plug and Play–enabled architecture
- Support for mini-port drivers that are binary compatible with Windows NT
- Support for Windows 3.1 fast disk drivers for backward compatibility
- Protected-mode drivers that take over real-mode MS-DOS device drivers when safe to do so
- The ability to support existing MS-DOS real-mode disk device drivers for compatibility

The following sections examine the different areas that make up the block I/O subsystem. The explanations are provided to facilitate an understanding of the components, bearing in mind that the configuration of the disk device driver layers is isolated from the user.

The I/O Supervisor

The I/O Supervisor (IOS) provides services to file systems and drivers. The IOS is responsible for the queuing of file service requests and for routing the requests to the

appropriate file system driver. The IOS also provides asynchronous notification of file system events to installed drivers.

The Port Driver

The port driver is a monolithic 32-bit protected-mode driver that communicates with a specific disk device, such as a hard disk controller. This driver is specifically for use with Windows 95 and resembles the 32-bit disk access (fast disk) driver used in Windows 3.1, such as the WDCTRL driver used for Western Digital compatible hard disk controllers. In Windows 95, the driver that communicates with IDE/ESDI hard disk controllers and floppy disk controllers is implemented as a port driver. A port driver provides the same functionality as the combination of the SCSI manager and the mini-port driver.

The SCSI Layer

The SCSI layer applies a 32-bit protected-mode universal driver model architecture to communication with SCSI devices. The SCSI layer provides all the high-level functionality that is common to SCSI-like devices and then uses a mini-port driver to handle device-specific I/O calls. The SCSI Manager is part of this system and provides compatibility support for using Windows NT mini-port drivers.

The Mini-Port Driver

The mini-port driver model used in Windows 95 simplifies the task of writing device drivers for disk device hardware vendors. Because the SCSI Stub provides the high-level functionality for communicating with SCSI devices, disk device hardware vendors need to create only a mini-port driver that is tailored to their own disk device. The mini-port driver for Windows 95 is 32-bit protected-mode code and is binary compatible with Windows NT mini-port drivers, another factor that simplifies the task of writing device drivers. Binary compatibility with NT also results in a more stable and reliable device driver because hardware vendors need to maintain only one code base for device support. Users of Windows 95 also benefit because many mini-port drivers are already available for Windows NT.

Support for IDE, ESDI, and SCSI Controllers

Through the use of either a port driver or a mini-port driver, support for a broad array of disk devices will be available for Windows 95, including popular IDE, ESDI, and SCSI disk controllers. Users won't have to decide whether to use a port driver or a mini-port driver because the driver is provided by the hardware vendor and configuration of the driver is handled by the Windows 95 system.

The Real-Mode Mapper

To provide binary compatibility with real-mode MS-DOS–based disk device drivers for which a protected-mode counterpart does not exist in Windows 95, the block I/O subsystem provides a mapping layer to allow the protected-mode file system to communicate with a real-mode driver as if it were a protected-mode component. The layers above and including this real-mode mapper (RMM) are protected-mode code, and the real-mode mapper translates file I/O requests from protected mode to real mode so that the MS-DOS device driver can perform the desired read or write operation from or to the disk device. An example of when the real-mode mapper would come into play is when real-mode disk-compression software is running and a protected-mode disk-compression driver is not available.

Long Filename Support

The use of long filenames of up to 255 characters in Windows 95 overcomes the sometimes cryptic 8.3 MS-DOS filename convention and allows more user-friendly filenames. MS-DOS 8.3 filenames are maintained and tracked by the system to provide compatibility with existing Win16–based and MS-DOS–based applications that manipulate only 8.3 filenames, but as users migrate to Win32–based applications, the use of 8.3 filename conventions is hidden from the user.

Long filenames are supported by extending the MS-DOS FAT file system and using bits and fields that were previously reserved by the operating system to add special directory entries that maintain long filename information. Extending the MS-DOS FAT layout, rather than creating a new format, allows users to install and use Windows 95 on existing disk formats without having to change their disk structure or reformat their drives. This implementation provides ease of use and allows future growth while maintaining backward compatibility with existing applications.

Because Windows 95 simply extends the FAT structure, long filenames are supported on diskettes as well as hard disks. If a file on a diskette that has a long filename is viewed on a computer that is not running Windows 95, only the 8.3 filename representation is seen.

Figure 30 shows a disk directory with long filenames and their corresponding 8.3 filename mappings on a computer running Windows 95.

```
Volume in drive C is MY HARDDISK
 Volume Serial Number is 1B47-7161
 Directory of C:\LONGFILE

. <DIR> 05-11-94 10:34a .
.. <DIR> 05-11-94 10:34a ..
4THQUART XLS 147 05-11-94 12:25a 4th Quarter Analysis.xls
TEXTFILE TXT 147 05-11-94 12:25a TEXTFILE.TXT
```

(continued)

Figure 30. *(continued)*

```
THISISMY DOC 147 05-11-94 12:25a this is my long filename.doc
1994FINA DOC 147 05-11-94 10:35a 1994 Financial Projections.doc
 4 file(s) 588 bytes
 2 dir(s) 48,009,216 bytes free
```

Figure 30. A directory listed from the command prompt, showing both 8.3 and long filenames

Support for Existing Disk Management Utilities

For existing disk management utilities to recognize and preserve long filenames, utility vendors need to revise their software products. Microsoft is working closely with utilities vendors and is documenting long filename support and its implementation as an extension to the FAT format as part of the Windows 95 Software Development Kit (SDK).

Existing MS-DOS–based disk management utilities that manipulate the FAT format, including disk defragmenters, disk bit editors, and some tape backup software, may not recognize long filenames as used by Windows 95 and may destroy long filename entries in the FAT format. However, no data is lost if the long filename entry is destroyed because the corresponding system-defined 8.3 filename is preserved.

Hidden File Extensions

Like Windows 3.1, Windows 95 uses file extensions to associate a given file type with an application. However, to make it easier to manipulate files, file extensions are hidden from users in the Windows 95 shell and in the Windows Explorer, and instead, icons are used in the UI in Windows 95 to differentiate the documents associated with different applications. Information about file type associations is stored in the Registry, and the associations are used to map a given file with the icon that represents the document type. (For compatibility reasons, Windows 95 must track filename extensions for use with existing MS-DOS and Win16–based applications.)

In addition to hiding filename extensions in the Windows 95 shell and the Windows Explorer, application developers can hide filenames from users in their applications. Mechanisms for hiding filenames are documented in the Windows 95 SDK. A good Windows 95 application makes use of these mechanisms for handling files to be consistent with the rest of the Windows 95 environment.

Additional File Date/Time Attributes

To further enhance the file system, Windows 95 maintains additional date/time attributes for files that MS-DOS does not track. Windows 95 tracks the date/time when a new file was created, the date/time when a file was modified, and the date when a file was last opened. These file attributes are displayed in the file's property sheet, as shown in Figure 31 on the following page.

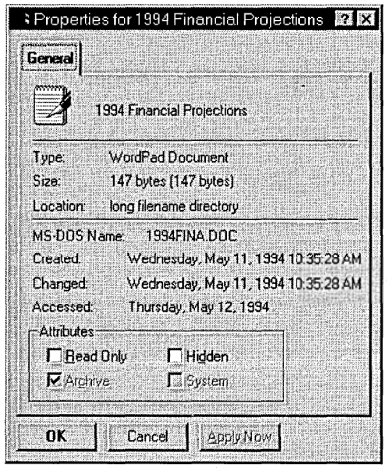


Figure 31. The properties for a file, showing the new file date/time attributes

Utilities vendors can take advantage of this additional date/time information to provide enhanced backup utilities—for example, to use a better mechanism when determining whether a given file has changed.

Coordinated Universal Time Format

MS-DOS has traditionally used the local time of the computer as the time stamp for the directory entry of a file, and continues to use local time for files stored on the local system. However, Windows 95 supports the use of the coordinated universal time (UTC) format for accessing or creating information on network file servers. This format provides the superior, more universal tracking of time information required by networks that operate across time zones.

Exclusive Access for Disk Management Tools

Disk management utilities, such as disk defragmenters, sector editors, and disk-compression utilities, don't get along well with Windows 3.1. File system programs, such as CHKDSK and DEFRAG, require exclusive access to the file system to minimize the disk access complexities that are present in a multitasking environment where disk I/O occurs. For example, without exclusive access to the disk, data corruption might occur if a user requests that a disk operation move information on the disk at the same time that another task is accessing that information or writing other information to disk. However, Windows 3.1 and MS-DOS do not provide a means of controlling access to the disk, so users have been forced to exit Windows and enter MS-DOS to run disk management utilities.

The file system in Windows 95 has been enhanced to support the use of Windows–based disk management utilities by permitting exclusive access to a disk device. Exclusive disk access is handled as part of the file system through a new API mechanism and can be used by utilities vendors to write Windows–based disk management utilities. Microsoft is encouraging third-party utilities vendors to use this API mechanism to move existing MS-DOS–based utilities to Windows, and is also using it to deliver disk management utilities as part of Windows 95.

For example, this mechanism is used by the Disk Defragmenter (Optimizer) utility delivered as part of Windows 95. Unlike the disk defragment utility used under the combination of MS-DOS and Windows 3.1, the Disk Defragmenter in Windows 95 can be run from the Windows 95 shell and can even be run in the background while users continue to work on their systems.

DriveSpace Disk Compression

Windows 95 provides built-in support for DriveSpace disk compression. Compatible with DoubleSpace and DriveSpace disk compression provided with MS-DOS, Windows 95 provides base compression in the form of a 32-bit virtual device driver that delivers improved performance over previously available real-mode compression drivers and frees conventional memory for use by MS-DOS–based applications. Users of MS-DOS–based DoubleSpace and DriveSpace don't need to change their existing compressed volume file (CVF) and thus don't need to take any special actions when they install Windows 95.

As shown in Figure 32, the DriveSpace disk compression tool provided with Windows 95 is GUI-based and provides the ability to compress a physical hard drive or removable floppy drive. The Compress a Drive dialog box, shown in Figure 33 on the following page, graphically depicts the amount of free space available before compression and the estimated space available after compression.

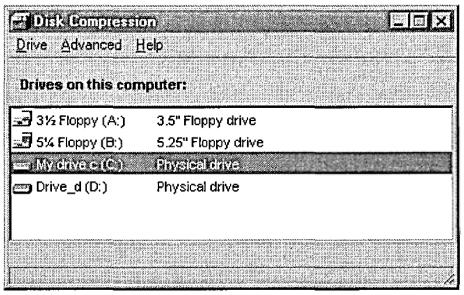


Figure 32. The DriveSpace disk compression tool

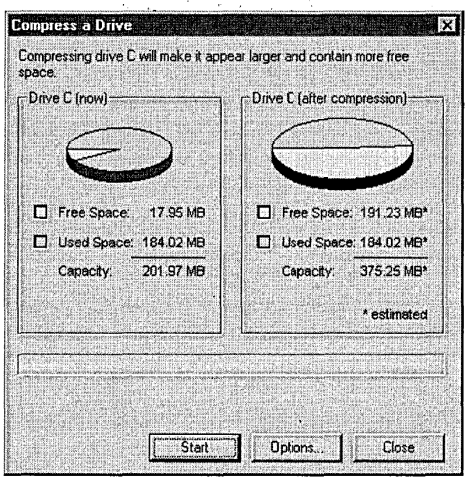


Figure 33. The Compress a Drive dialog box, which graphically displays free space

System Capacity Improvements

Windows 95 provides better system capacity for running MS-DOS and Win16–based applications than Windows 3.1. A number of internal enhancements to the base system prevent internal system resources from being exhausted as quickly as was possible when running multiple Windows–based applications under Windows 3.1.

Many of the artificial limitations present in Windows 3.1 were due to its architecture or internal data structures, which were in turn largely due to the fact that Windows 3.1 had to run on an Intel 80286–based computer. These limitations have for the most part been overcome in Windows 95, to the benefit of users as well as ISVs and other developers.

System Resource Limitation Improvements

Many users have encountered *Out of Memory* error messages when running multiple Windows–based applications under Windows 3.1, even though the system still reports several megabytes of available free memory. Typically these messages were displayed because the system could not allocate an internal memory resource in a Windows API function call due to lack of available space in a region of memory called a *heap*.

Windows 3.1 maintains heaps for system components called GDI and USER. Each heap is 64 KB in size and is used for storing GDI or memory *object* information allocated when an application calls a Windows API function. The amount of space

available in the combination of these two heaps is identified as the percentage of system resources that are free and is displayed in the About dialog box in Program Manager and other Windows applications, as shown in Figure 34.

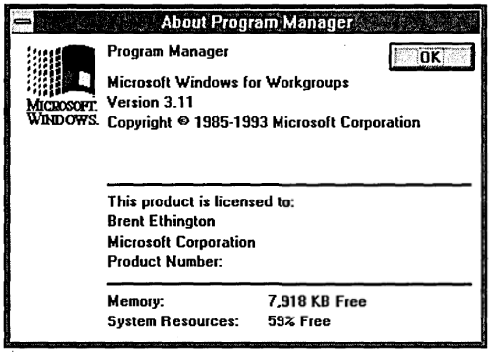


Figure 34. The About dialog box in Program Manager in Windows 3.1, showing free system resources

The percentage of free system resources displayed in the About dialog box is calculated using an internal algorithm to represent the aggregate percentage of free memory in the GDI and USER heaps. When the free system resources percentage gets too low, users commonly see an *Out of Memory* error message, even though the amount of free memory shown in the About dialog box is still quite high. This error can result from low memory in either the GDI or USER heap (or both).

To help reduce the system resource limitation, a number of the data structures stored in the 16-bit GDI and USER heaps in Windows 3.1 have been moved out of these heaps and stored in 32-bit heaps, providing more room for the remaining data elements to be created. As a result, system resources decrease less rapidly in Windows 95 than they did in Windows 3.1.

For compatibility, not all objects were removed from the 16-bit GDI or USER heap and placed in a 32-bit heap. For example, some Windows–based applications manipulate the contents of the GDI heap directly, bypassing the published API mechanisms for doing so, because their developers think direct manipulation increases performance. However, because these applications bypass the Windows API mechanisms, moving their data from the existing heap structures and placing them in 32-bit heaps would cause these applications to fail because of memory access violations.

Win16–based and Win32–based applications use the same GDI and USER heaps. The impact of removing selected items from the heaps was closely examined and objects were selected based on the biggest improvement that could be achieved while affecting the fewest number of applications. For example, the GDI heap can quickly become full because of the creation of memory-intensive region objects that are used by

applications for creating complex images and by the printing subsystem for generating complex output. Region objects were removed from the 64 KB 16-bit GDI heap and placed in a 32-bit heap, benefiting graphic-intensive applications and providing for the creation of more smaller objects by the system. Windows 95 improves the system capacity for the USER heap by moving menu and window handles to the 32-bit USER heap. Instead of the total limit of 200 for these data structures in Windows 3.1, Windows 95 allows 32,767 menu handles and an additional 32,767 window handles *per process* rather than system-wide.

In addition to moving information from the GDI and USER heaps, robustness improvements in Windows 95 that facilitate system cleanup of unfreed resources also relieve system resource limitations. When Windows 95 determines that the owner and other ended processes no longer need the resources in memory, Windows 95 cleans up and deallocates leftover data structures. The robustness improvements in Windows 95 are discussed in Chapter 5, "Robustness."

Better Memory Management

Windows 95 improves addressibility to provide better access to physical memory, as well as improves upon the swapfile implementation provided in Windows 3.1 to support virtual memory supplementation of physical memory.

Linear Memory Addressing for Win32–Based Applications

To support a 16-bit operating environment, the Intel processor architecture uses a mechanism, called *segments*, to reference memory by using a 16-bit segment address and a 16-bit offset address within the segment. A segment is 64 KB in size, and applications and the operating system pay a performance penalty when they access information across segments. For 32-bit operating system functionality and Win32–based applications, Windows 95 addresses this issue by using the 32-bit capabilities of the Intel 80386 (and above) processor architecture to support a flat, linear memory mode. A linear addressing model simplifies the development process for application developers, removes the performance penalties imposed by the segmented memory architecture, and provides access to a virtual address space that permits the addressing of up to 4 GB (4 gigabytes, or 4 billion bytes) of memory. Windows 95 uses the flat memory model internally for 32-bit components and virtual device drivers.

Compatibility with the Windows NT Memory Model

Windows 95 uses the same memory model architecture as Windows NT, providing high-end operating system functionality for the mainstream system. Windows 95 allows full use of the 4 GB of addressable memory space to support even the largest

desktop application. The operating system provides a 2 GB memory range for applications and reserves a 2 GB range for itself.

Virtual Memory Support (Swapfile) Improvements

Windows 95 addresses problems and limitations imposed in Windows 3.1 by its virtual memory swapfile implementation. With Windows 3.1, users were faced with a myriad of choices and configuration options for setting up a swapfile to support virtual memory. They had to decide whether to use a temporary swapfile or a permanent swapfile, how much memory to allocate to the swapfile, and whether to use 32-bit disk access to access the swapfile. A temporary swapfile did not need to be contiguous, and Windows would dynamically allocate hard disk space when it was started and free up the space when it was terminated. A permanent swapfile provided the best performance, but it had to be contiguous, had to be set up on a physical hard disk, and was statically specified by the user and not freed up when the user exited Windows.

The swapfile implementation in Windows 95 simplifies the configuration task for the user and, because of improved virtual memory algorithms and access methods, combines the best of a temporary swapfile and a permanent swapfile. The swapfile in Windows 95 is dynamic and can shrink or grow based on the operations performed on the system. The swapfile can occupy a fragmented region of the hard disk and it can be located on a compressed disk volume.

Windows 95 uses intelligent system defaults for the configuration of virtual memory, relieving the user of the task of changing virtual memory settings. Figure 35 shows the simplified virtual memory configuration settings.

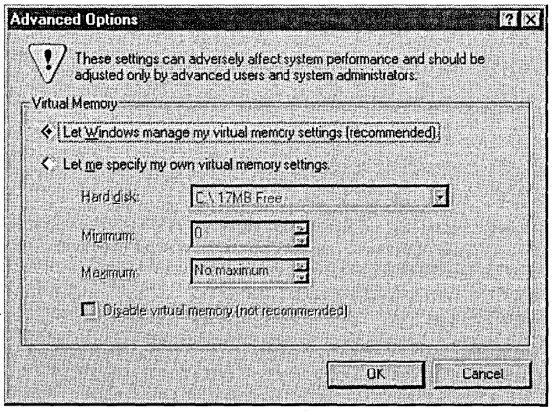


Figure 35. The simplified virtual memory settings

The Registry: A Centralized Configuration Store

Windows 95 uses a mechanism called the *Registry* to serve as the central configuration store for user, application, and computer-specific information. The Registry solves problems associated with the .INI files used in Windows 3.1 and is a hierarchical database that stores system-wide information in a single location, making it easy to manage and support.

Solutions to .INI Problems

Windows 3.1 uses initialization (.INI) files to store system-specific or application-specific information about the state or configuration of the system. For example, the WIN.INI file stores state information about the appearance or customization of the Windows environment; the SYSTEM.INI file stores system-specific information on the hardware and device-driver configuration of the system; and various .INI files, such as WINFILE.INI, MSMAIL.INI, CLOCK.INI, CONTROL.INI, and PROGMAN.INI, store application-specific information about the default state of an application.

Problems with .INI files under Windows 3.1 for configuration management include the following:

- Information is stored in several different locations, including CONFIG.SYS, AUTOEXEC.BAT, WIN.INI, SYSTEM.INI, PROTOCOL.INI, private .INI files, and private .GRP files.
- .INI files are text-based and limited in total size to 64KB, and APIs allow for get/write operations only.
- Information stored in .INI files is non-hierarchical and supports only two levels of information: key names broken up by section headings.
- Many .INI files contain a myriad of switches and entries that are complicated to configure or are used only by operating system components.
- .INI files provide no mechanism for storing user-specific information, thus making it difficult for multiple users to share a single computer.
- Configuration information in .INI files is local to each system, and because no API mechanisms are available for remotely managing configuration, managing multiple systems is difficult.

To solve these problems, the Registry was designed with the following goals in mind:

- Simplify the support burden.
- Centralize configuration information.

- Provide a means to store user, application, and computer-specific information.
- Provide local and remote access to configuration information.

The Registry is structured as a database of keys in which each key can contain a value or other keys (subkeys). As shown in Figure 36, the Registry uses a hierarchical structure to store text or binary value information and maintains all of the configuration parameters normally stored in the Windows system .INI files such as WIN.INI, SYSTEM.INI, and PROTOCOL.INI. Although similar in some ways to the Registration Database used in Windows 3.1, which served as a central repository for file associations and OLE registration information, the Registry in Windows 95 extends the Registration Database structure to support keys that can have more than one value and also support data of different types.

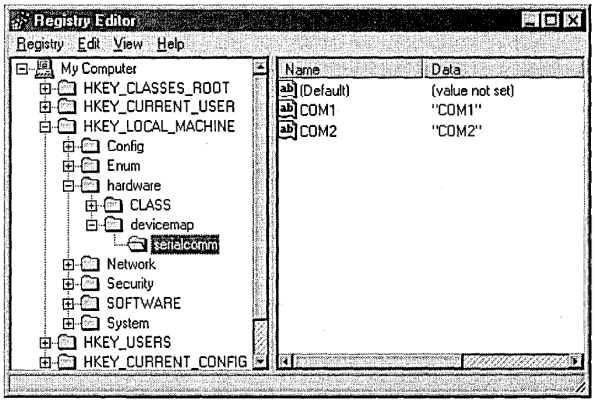


Figure 36. The hierarchy of the Registry, as displayed by the Registry Editor

The Registry is made up of several .DAT files that contain system-specific information (SYSTEM.DAT) or user-specific information (USER.DAT). System-specific information, such as the static reference to loading virtual device drivers, is moved as appropriate from the SYSTEM.INI file to the Registry.

System Switch Simplification

Another improvement in Windows 95 over the Windows 3.1 use of .INI files is related to system switch simplification. Windows 3.1 supports several hundred different configuration switches that can be specified in system .INI files, including WIN.INI and SYSTEM.INI. With intelligent enhancements made to the system and better dynamic configuration properties, Windows 95 has reduced the number of entries normally associated with .INI files. These reductions didn't result from simply moving .INI entries to the Registry but by examining and justifying the presence of each and every one.

No .INI Files?

Like CONFIG.SYS and AUTOEXEC.BAT, WIN.INI and SYSTEM.INI and application-specific .INI files still exist for compatibility reasons. The Win16 APIs for manipulating .INI files still manipulate .INI files, but developers of Win32–based applications are encouraged to use the Registry APIs to consolidate application-specific information.

Many existing Win16–based applications expect to find and manipulate the WIN.INI and SYSTEM.INI files to add entries or load unique device drivers, so Windows 95 examines .INI files during the boot process. For example, the [386Enh] section of SYSTEM.INI is checked for virtual device drivers.

Role in Plug and Play

One of the primary roles of the Registry in Windows 95 is to serve as a central repository for hardware-specific information for use by the Plug and Play system components. Windows 95 maintains information about hardware components and devices that have been identified through an enumeration process in the hierarchical structure of the Registry. When new devices are installed, the system checks the existing configuration in the Registry to determine which hardware resources—for example, IRQs, I/O addresses, DMA channels, and so on—are not being used, so that the new device can be properly configured without conflicting with a device already installed in the system.

Remote Access to Registry Information

Another advantage of the Registry for Win32–based applications is that many of the Win32 Registry APIs use the remote procedure call (RPC) mechanism in Windows 95 to provide remote access to Registry information across a network. As a result, desktop management applications can be written to aid in the management and support of Windows–based computers, and the contents of the Registry on a given PC can be queried over a network. Industry management mechanisms, such as SNMP or DMI, can easily be integrated into Windows 95, simplifying the management and support burden of an MIS organization. For more information about manageability and remote administration, see Chapter 9, "Networking."

Better Font Support

Font support in Windows 95 has been enhanced to provide better integration with the UI and has been optimized for the 32-bit environment. It also provides capabilities such as font smoothing for fonts that have not previously been offered as part of a mainstream operating system.

The 32-Bit TrueType Rasterizer

The rasterizer component for rendering and generating TrueType fonts is enhanced in Windows 95. The rasterizer is written as a 32-bit component, and delivers better fidelity from the mathematical representation to the generated bitmap, as well as better performance for rendering TrueType fonts.

In addition to performance enhancements, the new 32-bit rasterizer provides support for generating complicated glyphs—for example, Han—and results in a faster initial boot time in Windows 95 than in Windows 3.1 when many fonts are installed.

CHAPTER 5

Robustness

Windows 95 improves on the robustness of Windows 3.1 to provide great support for running MS-DOS–based, Win16–based, and Win32–based applications, and to provide a high level of system protection from errant applications.

Windows 3.1 provided a number of mechanisms to support a more robust and stable environment over Windows 3.0, including the following:

- **Better resource clean-up.** When an MS-DOS–based or Windows–based application crashed, users could continue running in a way that allowed them to save their work.
- **Local reboot.** Users could shut down an application that hung.
- **Parameter validation for API calls.** The system could catch many common application errors and fail the API call, rather than allow bad data to be passed to an API.

Just as the improvements in Windows 3.1 provided a more robust and stable environment than Windows 3.0, the improvements in Windows 95 provide an even better environment.

System-Wide Robustness Improvements

System-wide improvements resulting in a more robust operating system environment than Windows 3.1 include:

- Better local reboot
- Virtual device driver thread clean-up when a process ends
- Per-thread state tracking
- Virtual device driver parameter validation

Better Local Reboot

The capability whereby users can end an application or VM that hangs is called a *local reboot*. With Windows 3.1, users could perform a local reboot by pressing the three-key CTRL+ALT+DEL combination. Users could pretty easily end errant VMs with a local reboot request, but a local reboot request for a Windows–based application often didn't end the errant Windows–based process or brought the entire system down.

Windows 95 greatly improves the local reboot support by providing a means to end an MS-DOS–based application running in a VM, a Win16–based application, or a Win32–based application without bringing down the entire system. Moreover, the process of cleaning up the system after a local reboot is now more complete than for Windows 3.1. (This process is described later in this chapter.)

In Windows 3.1, when a user requests a local reboot, the system may identify the active application as the application that has the focus of the local reboot request, or it may report back that there is no application in a hung or inactive state. In Windows 95, the system displays the Close Program dialog box, which identifies the tasks that are running and the state that the system perceives each one to be in, as shown in Figure 37. This level of detail affords the user much more flexibility and control over the local reboot than with Windows 3.1.

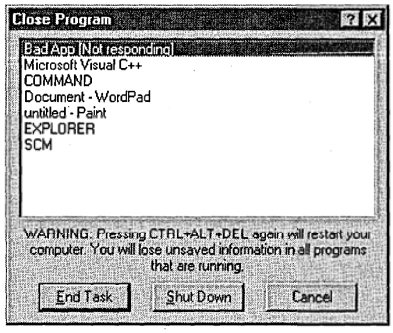


Figure 37. The Close Program dialog box

Applications are identified as "not responding" when they haven't checked the message queue for a period of time. Although some applications don't check the message queue while performing computationally intensive operations, well-behaved applications check the message queue frequently. In Windows 95, as in Windows 3.1, Win16–based applications must check the message queue to relinquish control to other running tasks.

Try It!

Perform a Local Reboot

1. With a couple of applications running, press CTRL+ALT+DEL. You are presented with a list of active applications. Applications that are hung are identified as *Not responding*.
2. Terminate one of the tasks by clicking *End Task*.

Virtual Device Driver Thread Clean-Up

Local reboot support is also aided by improved VxD thread clean-up when a given process ends. With Windows 3.1, the system often couldn't recover either if it was running real-mode code, such as BIOS routines, when an application ended abnormally, or if the user requested a local reboot to end a seemingly-hung application. For example, if an operation (such as a network operation in real-mode, a disk I/O, or an asynchronous application request) ended abnormally because of another application-based error, Windows 3.1 sometimes couldn't clean up properly to free allocated resources and sometimes couldn't even return control to the user.

Windows 95 improves system clean-up by providing each system VxD with the ability to track the resources it allocates on a per-thread basis. Because most computer system functionality and support is handled in Windows 95 by VxDs rather than by real-mode code or BIOS routines, Windows 95 can recover from errors or situations that, under Windows 3.1, would require that the computer be rebooted.

When Windows 95 ends a given thread (because the user exited the application, a local reboot was requested, or the application ended abnormally), each VxD receives notification that the thread is ended. This notification allows the VxD to safely cancel any operations it is waiting to finish and frees any resources that the VxD previously allocated for the thread or application. Because the system tracks each VM, Win16–based application, and Win32 thread as a separate per-thread instance, the system can clean up properly at each of these levels, without affecting the integrity of the system.

Per-Thread State Tracking

To aid in system clean-up, resource tracking in Windows 95 is much better than in Windows 3.1. In addition to tracking resources on a per-thread basis by system VxDs, resources such as memory blocks, memory handles, graphics objects, and other system items are allocated and also tracked by system components on a per-thread basis. Tracking these resources on a per-thread basis allows the system to clean up safely when a given thread ends, either normally—at the user's request—or abnormally. Resources are identified and tracked by both a thread ID and the major Windows version number that is stored in the .EXE header of the application.

Index

Numbers

16-bit code
  implemented to maintain compatibility, 54
32-bit code
  I/O subsystems and device drivers in Windows 95, 54
  provides maximum performance, 54
32-bit disk access
  block I/O subsystem, 68
32-bit file system architecture, 64–73

A

Accessibility, 283
  Accessibility status indicator, 287
  Americans with Disabilities Act, 283
  BounceKeys, 289
  controlling accessibility features, 285
  customizable mouse pointer, 287
  design goals for Windows 95, 284
  emergency hotkeys, 286
  guidelines for software developers, 293
  high-contrast mode, 288
  making keyboard and mouse input easier, 288
  MouseKeys, 289
  online help, 285
  RepeatKeys, 289
  SerialKeys, 292
  ShowSounds, 292
  SlowKeys, 289
  StickyKeys, 288
  summary of improvements over Windows 3.1, 284
Accessibility *(continued)*
  support for MS-DOS–based applications, 288, 291
  support for multiple pointing devices, 292
  ToggleKeys, 291
Admin Config tool, 169
Administrative support, 8
Application support, 7
Architecture
  communications architecture, 192
  how applications run in Windows 95, 57
  kernel improvements, 191
  mini-driver model, 117
  multimedia architecture, 262
  ring 0 and ring 3 code, 57
  size goals, 58
  summary of improvements over Windows 3.1, 51
AUTOEXEC.BAT, 53

B

Backup, 300–301
  server-based network backup, 13, 173
Block I/O subsystem, 68
Briefcase, 215
Browsing, 25

C

Cache, *See* VCACHE
CD-ROM File System (CDFS), 67
COMM.DRV, 193
Communications, 189–204
  16550A UART support, 201
  architecture discussion, 192
  centralized modem setup and configuration, 197
  configuring telephony services, 200

Communications *(continued)*
example of good Win32–based communications application, 203
goals in Windows 95, 190
improved device/hardware support, 201
in Windows 3.1, 193
in Windows 95, 193
more COM ports supported, 202
Plug and Play support, 191, 202
port drivers, 195
summary of improvements over Windows 3.1, 189
support, 9
TAPI, 194
universal modem driver, 194
use of TAPI in Windows 95 components, 196
Windows 95 kernel improvements, 191
Compatibility
user interface, 48
with existing hardware, 3, 15
with existing MS-DOS–based and Windows–based applications, 3, 15, 48–49
Compression, *See* DriveSpace
CONFIG.SYS, 53
Control Panel
access from the Start button, 23
improvements over Windows 3.1, 31, 42
properties for Display, 42
Coordinated Universal Time format, 72
Copying files, 28

D

Desktop Management Interface (DMI), 135
Desktop Systems Management, *See* Systems management
Device drivers, 28, 117–18
Device manager, 114–15
Device support, 7, 201–2
DHCP, *See* TCP/IP
Disk compression, *See* DriveSpace
Disk Defragmenter, 304
Disk device support
ESDI controllers, 120
high-speed floppy disk driver, 120
IDE, 119
logical block addressing, 118
removable media, 119
Disk utilities
supported under Windows 95, 73
Display adapter support
driver robustness improvements, 123
energy star monitor support, 124
image color matching, *See* Image Color Matching (ICM)
improved driver stability, 121
improved video display performance, 122
improvements over Windows 3.1, 121–22
new control panel enhancements, 123
VESA DPMS and DDC specification support, 121
Display resolution, 43
DMI (desktop management interface), 135
DNS, *See* TCP/IP
DoubleSpace, *See* DriveSpace
Drag and drop, 41
Drive mappings, 28
DriveSpace, 73
Dynamic networking, 211

E

Edit (MS-DOS–based editor), 303
Enhanced Metafile (EMF), *See* Printing
Exchange, *See* Microsoft Exchange

F

File extensions, 71
File synchronization, *See* Mobile computing
File system
32-bit protected mode FAT (VFAT), 66
additional date/time attributes, 71
architecture, 64

File system *(continued)*
benefits of changes from Windows 3.1, 64
exclusive volume access, 72
installable file system, 65
sample property sheet, 72
support for multiple network redirectors, 140
third-party functionality, 66
Filenames, *See* Long filenames
Files
quick-viewing contents of, 47
renaming files, 27
Find Utility, 23, 44
Folders within folders, 26
Font settings, 46–47
Fonts
32-bit TrueType rasterizer, 80
improved support in Windows 95, 80
integrated with the UI, 46
Fonts folder, 46

G

Graphics Device Interface (GDI)
32-bit code distribution, 55

H

Hardware Design Guide for Microsoft Windows 95, 113
Hearing-impaired users, 292
Help
new online Help, 33–35
shortcut buttons, 34
simplified interface, 33
table of contents, 34
What's This? button, 34
HyperTerminal, 203, 301–2

I

IDE disk drives, 119
IFS, *See* File system
Image Color Matching (ICM), 182
benefits to users, 182
InterColor 3.0, 183
.INI files
problems under Windows 3.1, 78
still used by Win16-based applications, 80
system switch simplification, 79
Installable File System, 140
Installable File System Manager, 65–66
Installation and setup of Windows 95, 265–72
International language support, 273–82
benefits for developers, 274
benefits for users, 274
date and time formats, 276
international language issues, 275
localization of Windows 95, 275
multilingual content support, 278
sorting and searching support, 277
summary of improvements over Windows 3.1, 273
support for different national character sets, 278
Win32 National Language Support APIs, 282
Internet access, 132, 234
interrupt latency
solution for communications problems, 192
I/O Supervisor (IOS), 68
IPX/SPX, 150

L

Linear memory addressing, 76
Local reboot
improved effectiveness for Win32–based applications, 91
in Windows 95, 84
terminating hung applications, 84
Long filenames
documentation for developers, 71
extensions to the MS-DOS FAT file system, 70
no reformatting necessary, 70
sample MS-DOS directory listing, 70

Long filenames *(continued)*
support for 8.3 MS-DOS filenames, 70
support for existing disk management utilities, 71
support on diskettes, 70
support on networks, 153
usability benefits, 27

M

Management information service (MIS) organizations, 13, 14–15, 16
Management support, 8
MAPI (Messaging Application Programming Interface) 1.0, 223
Master key, *See* Security
Memory
memory model, 76
zero conventional footprint, 94
Messaging Application Programming Interface (MAPI) 1.0, 223
Messaging support, 9
Microsoft At Work Fax, 217, 223, 235, 237–42
Microsoft CD-ROM Extensions (MSCDEX), 67
Microsoft Exchange, 221–42
combination of drivers supported, 224
common address book, 222
components of, 222
described, 225
drivers for use with Microsoft Mail, 232
improvements over Microsoft Mail, 225
information stores, 230, 242
Internet Mail drivers, 223, 234
Microsoft At Work Fax, 223, 235, 237–42
open architecture, 224
personal address book, 229
personal information store, 222, 230
remote Mail, 232
third-party MAPI driver vendors, 223
workgroup fax features for MIS, 238
workgroup post office, 233
Microsoft Network (MSN), 243–45
integration, 147
remote administration, 146, 149
share-level security, 149
user-level security, 149
Mini-port driver, 69
MIS (management information service) organizations, 13, 14–15, 16
Mobile computing, 205–19
deferred printing support, 218
docking support, 212
document viewers, 218
dynamic networking, 211
file synchronization, 215
flexible video resolution support, 214
framework, 206
local connections, 217
Messaging API (MAPI), 219
Microsoft At Work Fax, 217
password management, 211
PCMCIA support, 213
point devices support, 215
power management, 214
remote network access, 134, 207
support, 9
support for deferred printing, 182
TAPI, 210
UniModem, 210
Mobile services support, 9
Modem support
benefits of configuration in Windows 95, 197
centralized setup and configuration, 197
Plug and Play, 199, 202
Mouse
right-clicking, 40
Mouse device support, 126, 127
MSCDEX (Microsoft CD-ROM Extensions), 67

MS-DOS
  32-bit protect-mode device drivers, 94
  and Windows 95, 54
  Int 21h, 65
MS-DOS command prompt
  new and enhanced commands, 104
  support for long filenames, 105
  support for UNC pathnames, 104
MS-DOS–based applications
  better cleanup after termination, 87
  better defaults for running, 98
  graphic-intensive applications, 98
  hardware-intensive, 96
  improved compatibility, 96
  improved memory protection, 98
  improved protection for virtual machines, 86
  local VM environment settings, 103
  PIF editor in Windows 3.1, 99
  preemptively multitasked, 60
  property sheets for, 99
  run in separate VMs, 60
  single MS-DOS application mode, 97
  summary of improvements over Windows 3.1 for running, 93
  support for VGA graphics mode, 98
  system protection, 61
  terminating gracefully, 102
  text editor, 303
  toolbar in MS-DOS window, 100
  user-scaleable MS-DOS window, 101
Multimedia, 247–64
  32-bit digital video playback, 254
  architecture discussion, 262
  AutoPlay, 251
  benefits for consumers, 249
  benefits for developers, 250
  benefits for hardware makers, 250
  benefits from multitasking and threads, 255
  built-in support for digital video, 252
  CreateDIBSection, 257
  end of dancing postage stamp era, 249
Multimedia *(continued)*
  game support, 256
  growth of PC market in 1993, 247
  history, 248
  MIDI, 252
  MPEG support, 256
  opportunities for IHVs and OEMs, 260
  Plug and Play support, 250
  polymessage MIDI support, 258
  recommendations to OEMs, 260
  routing data through Windows, 263
  sound compression, 258
  support for DCI, 261
  support for fast CD-ROMs, 255
  Video for Windows, 248
  WinG, 257
Multiple users on the same PC, 161
Multitasking, 52
Multithreading
  Win32–based applications, 53
My computer browsing model, 21, 25–26

N

NDIS (Network Driver Interface Specification) 3.1, 141
NetBEUI, 152
NetWare connectivity
  solution for MIS problems, 15
NetWatcher, 136, 176
Network Driver Interface Specification (NDIS) 3.1, 141
Network management, 173
  NetWatcher, 176
  tools, 175
Network Neighborhood, 21, 28–9
Network provider interface, *See* Networking
Network redirectors
  support in file system for using multiple, 65
Network support, 7, 212–13
Network tool in Control Panel, 28

Networking
32-bit Microsoft client, 148
32-bit SMB interoperability, 148
architecture, 137
as easy as "point and click," 133
built-in, 13
compatible with existing networks, 149
consistent browsing across multiple providers, 133
dynamic networking for mobile computing, 211
integration with Novell NetWare, *See* Novell NetWare
interprocess communications interfaces, 153
managing network resources, 173
Microsoft Network integration, 147
Microsoft Network Peer Services, 148
mobile computing support, 133
modular, open design, 137
multiple protocol support, 137
NDIS 3.1, 141
Network Neighborhood, 133
network provider interface (NPI), 138
pass-through security, 145
Plug and Play enabled, 137
printing improvements, 186
protocol support, 150
server-based backup, 173
summary of improvements over Windows 3.1, 130
system login, 139
system management support, 134
well-connected client operating system, 131
Novell NetWare
32-bit protect-mode Microsoft Client for NetWare, 131
bindery support, 143
client support, 142
command-line utilities, 147
file and printer sharing for NetWare, 131, 144
Novell NetWare *(continued)*
improved performance with Windows 95 client, 143
integration in Windows 95, 131, 142–47
IPX/SPX interoperability, 150
login scripts, 144
Microsoft Print Server for NetWare, 146
NetWare API support, 147
network login, 139
remote network access, 157
RIPL support, 161
support for diskless workstations, 147
support for long filenames, 153
support for NetWare login scripts, 131
support in the Network Neighborhood, 133

O

OLE
drag and drop, 309
features of, 309
use in applications for Windows 95, 308
use in Paint, 299
use in WordPad, 298
visual editing, 309
Online Help, *See* Help

P

Paint, 299
Password management, 211
Performance
compared to Windows 3.1, 4
Performance monitoring, 172
Phone Dialer application, 204
Plug and Play
benefits with legacy hardware, 110
benefits with Windows 95, 112, 315–16
communications support, 191
described, 12, 107
dynamic operating environment, 111

Plug and Play *(continued)*
installation and configuration with no user intervention, 110
modem devices, 199
networking support, 134
new PC features, 110
PCMCIA support, 109
printer detection, 183, 186
problems with PCs today, 108
reduced PC support costs, 110
requirements for mobile computing, 109
solution to PC configuration problems, 109
support for communications devices, 202
support for multimedia devices, 250
support in Setup, 267
Port driver, 69
Printers folder, 45
Printing, 177–87
32-bit preemptive spooler, 178
32-bit print subsystem, 178
configuring a printer, 184
enhanced metafile (EMF) spooling, 179
local or remote print job management, 186
managing print jobs, 185
Microsoft Print Server for NetWare, 146
network printing support, 153
performance improvements in Windows 95, 180
Plug and Play printer detection, 183, 187
Printers folder, 183
printing from MS-DOS–based applications, 181
support for deferred printing, 182
support for "point and print" network printing, 184
Property sheets, 39

Q

Quick Viewers, 47, 295–97
Questions and answers about Windows 95, 321–33

R

Real-mode mapper
protect-mode to real-mode mapping, 70
provides compatibility with MS-DOS disk device drivers, 70
Recycle Bin, 21, 30
Registry
DAT files, 79
defined, 78
dynamic keys, 162
policies, 135
Registry Editor, 135, 169
registry tools, 169
remote access to configuration information, 80
role in Plug and Play support, 80
role in systems management, 160
solution to .INI file problems, 78
system management support, 135
Win32 API support, 162
Remote Access Gateway, 157
Remote access support, 9
Remote administration, 156, 186
Remote Mail, 218, 223, 235
Remote procedure call (RPC), 153
Right-clicking, 40
RMM, *See* Real-mode mapper
Robustness
better cleanup after virtual machine terminates, 87
debugging support in Windows 95 SDK, 88
for MS-DOS–based applications, 86
for Win16–based applications, 87
for Win32–based applications, 88

Robustness *(continued)*
improvements in Windows 3.1 over Windows 3.0, 83
local reboot, *See* Local reboot
parameter validation for Win16-based applications, 88
per-thread state tracking, 85
summary of improvements in Windows 95, 83
virtual device driver parameter validation, 86
virtual device driver thread cleanup, 85
RPC (remote procedure call), 153

S

Scaleable architecture, 1
ScanDisk, 305
SCSI (small computer system interface)
support described, 69
support from block I/O subsystem, 68
SCSI disk devices
32-bit ASPI drivers, 120
ASPI/CAM compatibility, 120
compatible with Windows NT mini-port drivers, 119
popular SCSI controllers supported, 119
Security
master key for unified logon, 154
network login, 139
network security, 154–57
pass-through security, 145, 156
support for user-level privileges, 145
user-level security, 13, 155
Settings
accessing from the Start button, 23
Setup, 136, 265
built-in verification of system files, 270
GUI-based, 266
improved hardware detection, 267
modular architecture, 266
smart recovery mechanism, 270
Setup *(continued)*
summary of improvements over Windows 3.1, 265
upgrading from Windows 3.1 or Windows for Workgroups 3.11, 272
Setup of Windows 95, 265–72
Shortcuts, 37–38
SNMP (Simple Network Management Protocol)
agent support, 134
MIB support, 135
support for network management, 175
Start button, 21, 22, 24
Structured exception handling, 91
Support
administrative, 8
application, 7
communications, 9
connectivity, 7
device, 7, 201–2
disk device, 118–20
display adapters, 120–23
flexible video resolution, 214
fonts, 80–81
hot-docking, 212
management, 8
mobile services, 9
monitors, 120, 124–25
mouse, 126–27
MS-DOS–based applications, 60–61
networking, 7, 212–13
PCMCIA, 213–14
Plug and Play, 111–12, 186–87, 202
remote access, 9
Win16–based applications, 58–60
Win32–based applications, 61–64
System management, 70–71
System performance, 6–7
System policies, 165
default settings, 166
for computers, 168
for users, 167
System Policy Editor, 136

System resources
  32-bit heaps in Windows 95, 75
  improvements in GDI, 76
  improvements in USER, 76
  improvements over Windows 3.1, 74–75
  percentage of, 75
Systems management
  DMI agent, 135
  management tools, 135
  managing desktops, 171
  managing user configurations, 164
  policies, 135
  registry-based, 135
  remote administration, 156
  role of server, 170
  SNMP agent, 134
  SNMP MIB support, 135
  system policies, 165
  user profiles, 164

T

TAPI (Telephony API), 191, 195, 210
  configuring telephony services, 200
  functionality supported, 196
  interfaces layer, 195
  role in Windows Open Services Architecture (WOSA), 195
  support for arbitrating access to modem or telephone device, 197
Task switching
  improvements to "cool switch," 24
Taskbar
  customizing the configuration, 24
  described, 21, 22
  objective for, 23
  Start button, 21, 22
  task buttons, 24
  task switching, 23
  undoing window operations, 41
TCP/IP (Transmission Control Protocol Internet Protocol), 150
  dynamic host configuration protocol (DHCP), 151
  OSF DCE Domain Naming Service (DNS), 152
  port-to-port protocol (PPP), 208
  PPP support, 157
  support for Windows Sockets, 151
  utility support in Windows 95, 150
  Windows Internet Naming Service (WINS), 152
  Windows NT Advanced Server, 151
Telephony API, *See* TAPI
Threads, 53
TrueType, *See* Fonts

U

UNC (Universal Naming Convention) pathnames
  obsolete "mapping" drives, 28
Undoing file operations, 31
UniModem, 191, 210–11
Upgrading, 4
Usability testing, 18, 19
User Interface
  design, 17–20, 312–15
  improving on Windows 3.1, 12, 18
  integration of printing functionality, 185
  migrating from Windows 3.1, 48
  objectives for, 17
  Printers folder, 183
  Start button, 22
  Taskbar, 22
  transition aids for users of Windows 3.1, 35
  using File Manager from Windows 3.1, 48
  using Program Manager from Windows 3.1, 48
User profiles, 13, 164
  supported settings, 165
UTC (Coordinated Universal Time), 72

V

VCACHE (32-bit file access cache), 67
VCOMM, 192
VFAT (32-bit protected mode FAT), 66
Virtual device drivers (VxDs)
  benefits under Windows 95, 57
  described, 55–56
  graceful cleanup from terminated application, 85
  parameter validation, 86
  replacement for real-mode MS-DOS device drivers, 56
  tracking allocated resources, 85
Virtual memory support, 77
VxD, *See* Virtual device drivers

W

Win16-based applications
  compatibility maintained in Windows 95, 58
  compatibility, size, and performance, 58
  impact of running in separate VMs, 59
  not running in separate VMs, 58
  parameter validation, 88
  performance, 59
  resource tracking, 87
  support for running, 58
  system protection, 59
Win32
  API, 307–8
  flat/linear address space, 62
Win32-based applications
  benefits over Win16-based applications, 61
  better local reboot, 91
  compatibility with Windows NT, 63
  improved robustness, 63, 88
  long filename support, 63
  memory protection, 63
  parameter validation, 89
  per-thread resource tracking, 89
  preemptive multitasking, 62
Win32-based applications *(continued)*
  private address space, 89
  separate message queues, 62, 89
  structured exception handling, 91
  support for, 61
  termination handling, 91
Windows 95 logo, 319–20
Windows Explorer, 36
Windows NT
  compatibility with NDIS mini-port drivers, 141
Windows NT Advanced Server
  domain login, 139
  network integration with, 147
  pass-through security in Windows 95, 145
  support for long filenames, 153
  user-level security privileges, 145
Windows NT Server, 184
Windows NT Workstation, 1–3, 4–5
Windows Sockets
  IPX/SPX support, 150
WINS, *See* TCP/IP
Wizards
  functionality in Windows 95, 32
  Modem Wizard, 199
  support for installing printers, 184
  wizards supported in Windows 95, 33
WordPad, 298–99

Z

Zero conventional memory footprint, 94–95

Introducing Microsoft® Windows 95

Get an Inside Look at Windows 95!

Topics include:

- The new Windows interface—and how Microsoft has made using PCs even easier
- The new 32-bit architecture enhancements and system improvements over Windows 3.1
- How Plug and Play simplifies the configuration of PCs and the addition of peripherals
- The new built-in networking functionality in Windows 95, including connectivity with Novell® NetWare®, Microsoft Windows NT™ Server, and the Internet
- How new system management functionality in Windows 95 makes managing PCs easier for system administrators
- How multimedia comes alive and is made easier with Windows 95
- How the new Microsoft Exchange mail client, Microsoft At Work™ fax software, and the Microsoft Network online service will expand the world of information for users of Windows 95
- New features you can expect from applications and hardware designed for use with Windows 95

Plus, you'll find a fascinating question-and-answer section that will help you separate rumor from fact. Accept no substitutes—this is everyone's guide to Windows 95.

In this exclusive Microsoft Press® special release, the Microsoft Windows 95 product team introduces you to Windows 95. Whether you're an individual computer user, a system administrator, a programmer, or an Internet surfer, you'll find valuable information on every aspect of Windows 95 (formerly known by the code name "Chicago"). INTRODUCING MICROSOFT WINDOWS 95 describes the features, benefits, and behavior of Windows 95. This comprehensive overview will leave you with a clear vision of how the software looks, works, and acts. It includes everything you've wanted to know about Windows 95 but didn't know who to ask!



The next-generation Windows interface makes using PCs even easier.



New features in Windows 95 such as mobile computing greatly increase access to information.

Operating Systems/Windows

ISBN 1-55615-860-2

90000

9 781556 158605

U.S.A.	**$12.95**
U.K.	£11.99
Canada	$17.95

[*Recommended*]

Microsoft Press®

Appendix FURNAS01

```
leader  01760nam a2200433Ia 4500
001     991000372529704436
005     20240613123624.0
008     940519s1994    nyua     b    101 0 eng d
020     ##$a0201765578
035     ##$z413456-o1cmu_inst
035     ##$a(OCoLC)30461984
040     ##$aGZM $cGZM $dPMS
049     ##$aPMSS $c[!38482006768556"]
099     ##$aQA75.5 $a.C53 1994
111     2#$aCHI '94 Conference $d(1994 : $cBoston, Massachusetts)
245     10$aHuman factors in computing systems : $b"Celebrating interdependence" : Boston, Massachusetts, USA, April 24-28, 1994 / $cedited by Beth Adelson, Susan Dumais, Judith Olson ;
260     ##$aNew York : $bAssociation for Computing Machinery, $cc1994.
300     ##$axviii, 483 p. : $bill. ; $c28 cm.
504     ##$aIncludes bibliographical references and index.
596     ##$a9
650     #0$aElectronic data processing $xCongresses.
650     #0$aElectronic digital computers $xPsychological aspects $xCongresses.
650     #0$aHuman engineering $xCongresses.
700     10$aAdelson, Beth.
700     10$aDumais, Susan.
700     10$aLandauer, Thomas.
700     10$aMackay, Wendy.
700     10$aOlson, Judith.
710     20$aSIGCHI (Group : U.S.)
740     01$aCHI '94 conference proceedings.
740     01$aCelebrating interdependence.
901     ##$aocm30461984
910     ##$a93941719G AL
916     ##$a0
917     ##$a19950326
918     ##$aBATCH
919     ##$a20040811
920     ##$aCATALOGER
999     ##$a004.019 C53H 1994 $wDEWEY $c1 $iA020411 $d1/21/2002 $e11/21/2001 $lBY-REQUEST $mOFFSITE $n18 $q22 $rY $sY $tBOOK $u3/26/1995 FORM=MARC
```

https://cmu.primo.exlibrisgroup.com/discovery/fulldisplay?context=L&vid=01CMU_INST:01CMU&search_scope=PITTSBURGH&tab=CMU&docid=alma99100037252970443 6 November 20, 2024

Carnegie Mellon University Libraries

SEARCH TIPS | BROWSE SEARCH | DATABASES A-Z | JOURNAL SEARCH | COLLECTION DISCOVERY | INTERLIBRARY LOAN | ... | SIGN IN | Menu

Search anything | Library Catalog | ADVANCED SEARCH

Chat with us

CONFERENCE PROCEEDING

Human factors in computing systems : "Celebrating interdependence" : Boston, Massachusetts, USA, April 24-28, 1994

CHI '94 Conference (1994 : Boston, Massachusetts); Adelson, Beth.; Dumais, Susan.; Landauer, Thomas.; Mackay, Wendy.; Olson, Judith.; SIGCHI (Group : U.S.)

c1994

Available at Offsite Repository BY-REQUEST (004.019 C53H 1994) >

TOP
FIND IN LIBRARY
LINKS
DETAILS
SEND TO
VIRTUAL BROWSE

Find in Library

Please sign in to check if there are any request options. Sign in

BACK TO LOCATIONS

LOCATION ITEMS

Offsite Repository
Available , BY-REQUEST ; 004.019 C53H 1994
(1 copy, 1 available, 0 requests)

Item in place
Sign in for loan information

Links

Report a Problem >
Signed in and still can't find what you're looking for? Let us know
Display source record >

Details

Title	Human factors in computing systems : "Celebrating interdependence" : Boston, Massachusetts, USA, April 24-28, 1994
Creator	CHI '94 Conference (1994 : Boston, Massachusetts) > Adelson, Beth. > Dumais, Susan. > Landauer, Thomas. > Mackay, Wendy. > Olson, Judith. > SIGCHI (Group : U.S.) >
Subject	Electronic data processing -- Congresses > Electronic digital computers -- Psychological aspects -- Congresses > Human engineering -- Congresses >
Other title	CHI '94 conference proceedings. Celebrating interdependence.
Publisher	New York : Association for Computing Machinery
Creation Date	c1994
Format	xviii, 483 p. : ill. ; 28 cm.
Bibliography Note	Includes bibliographical references and index.
Source	Library Catalog

Send to

EXPORT TO EXCEL | PRINT | EMAIL | PERMALINK | QR | CITATION | EXPORT RIS | EXPORT BIBTEX | EASYBIB

Virtual Browse




an factors
mputing
ems :
hing
ugh ...
1
Human factors in computing systems : reaching through ...
c1991
CHI '92, conference proceedings : striking a balance ...
c1992
Human factors in computing systems : "Celebrating interdepende ...
c1994
Human factors in computing systems : Conference companion : " ...
c1994
Human factors in computing systems : CHI '95 conference proceedings, ..
c1995

Appendix FURNAS02

Human Factors in Computing Systems

CHI '94 Conference Proceedings



acm PRESS

Sponsored by the Association for Computing Machinery/ Special Interest Group on Computer–Human Interaction

Human Factors in Computing Systems

CHI '94 Conference Proceedings

A0204111

004.019
C53h
1994

Addison
Wesley



In 1773, the Boston Tea Party signaled the beginning of a search for independence by the American Colonies. Today, cultures and civilizations around the world are beginning to recognize their *inter*dependence. Human-Computer Interaction (HCI) is about human and human-technology interdependence.

The annual CHI conference is the preeminent forum for exchanging information on all aspects of how people interact with computers and is sponsored by the Association for Computing Machinery's Special Interest Group on Computer-Human Interaction, ACM/SIGCHI. The CHI '94 conference theme, *Celebrating Interdependence*, invites you to explore the links and interdependencies among our disciplines and perspectives.

CONFERENCE TOPICS INCLUDE:

- User Interface Design
- Usability Testing
- State-of-the-Art GUI
- New Interface Tools and Environments
- Rapid Prototyping
- Visual Design and Layout
- Style Guidelines and Standards
- Interface Design
- Multimedia
- Designing for Users with Special Needs
- Participatory Design
- User Modeling
- Visual Design

Addison Wesley

90000>
9 780201 765571
ISBN 0-201-76557-8

University Libraries
Carnegie Mellon University
Pittsburgh PA 15213-3890



CHI '94

Conference Proceedings

Human Factors in Computing Systems

"Celebrating Interdependence"

Boston, Massachusetts
USA
April 24 – 28, 1994



Edited By:
Beth Adelson
Susan Dumais
Judith Olson

Technical Program Co-Chairs
Thomas Landauer
Wendy Mackay

Sponsored by the Association for Computing Machinery/Special Interest Group on Computer–Human Interaction (ACM/SIGCHI).

004.019
C53h
1994
c.2



The Association for Computing Machinery, Inc.
1515 Broadway
New York, NY 10036

Copyright © 1994 by the Association for Computing Machinery, Inc. Copying without fee is permitted provided that the copies are not made or distributed for direct commercial advantage and credit to the source is given. Abstracting with credit is permitted. For other copying of articles that carry a code at the bottom of the first page, copying is permitted provided that the per-copy fee indicated in the code is paid through the Copyright Clearance Center Inc., 222 Rosewood Drive, Danvers, MA. 01923. For permission to republish, write to Director of Publications, Association for Computing Machinery. To copy otherwise, or republish, requires a fee and/or specific permission.

...In proceedings of CHI, 1994 (Boston, MA, USA, April 24-28, 1994) ACM, New York, 1994, pp. 23-35

Ordering Information

Nonmembers
Nonmember orders placed within the U.S. should be directed to:

Addison-Wesley Publishing Company
Order Department
Jacob Way
Reading, MA 01867
Tel: +1 800 447 2226

Addison-Wesley will pay postage and handling on orders accompanied by check. Credit card orders may be placed by mail or by calling the Addison-Wesley Customer Service Department at the same number.

Please include the Addison-Wesley ISBN number with your order:

A-W ISBN 0-201-76557-8

Nonmember orders from outside the U.S. should be addressed as noted below:

Europe/Middle East

Addison-Wesley Publishing Group
Concertgebouwplein 25
1071 LM Amsterdam
The Netherlands
Tel: +31 20 6717296
Fax: +31 20 6645334

Germany/Austria/Switzerland

Addison-Wesley Verlag Deutschland GmbH
Wachsbleiche 7-12
5300 Bonn 1
Germany
Tel: +49 228 98 515 0
Fax: +49 228 98 515 99

United Kingdom/Africa

Addison-Wesley Publishers Ltd.
Finchampstead Road
Wokingham, Berkshire RG11 2NZ
United Kingdom
Tel: +44 734 794000
Fax: +44 734 794035

Asia

Addison-Wesley Singapore Pte. Ltd.
15 Beach Road
#05-02/09/10 Beach Centre
Singapore 0718
Tel: +65 339 7503
Fax: +65 339 9709

Japan

Addison-Wesley Publishers Japan Ltd.
Nichibo Building
1-2-2 Sarugakucho
Chiyoda-ku, Tokyo 101
Japan
Tel: +81 33 291 4581
Fax: +81 33 291 4592

Australia/New Zealand

Addison-Wesley Publishers Pty. Ltd.
6 Byfield Street
North Ryde, N.S.W. 2113
Australia
Tel: +61 2 878 5411
Fax: +61 2 878 5830

Latin America

Addison-Wesley Iberoamericana S.A.
Boulevard de las Cataratas #3
Colonia Jardines del Pedregal
Delegacion Alvaro Obregon
01900 Mexico D. F.
Tel: +52 5 660 2695
Fax: +52 5 660 4930

Canada

Addison-Wesley Publishing (Canada) Ltd.
26 Prince Andrew Place
Don Mills, Ontario M3C 2T8
Canada
Tel: +416 447 5101
Fax: +416 443 0948

ACM Members
A limited number of copies are available at the ACM member discount. Send order with payment in US dollars to:

ACM Order Department
P.O. Box 12114
Church Street Station
New York, N.Y. 10257

OR

ACM European Service Center
Avenue Marcel Thiry 204
1200 Brussels
Belgium

Credit card orders from U.S.A. and Canada:
+1 800 342 6626

New York Metropolitan Area, and outside of the the U.S.:
+1 212 626 0500

Please include your ACM member number and the ACM Order number with your order.

ACM Order Number: 608940
ACM ISBN: 0-89791-650-6

University Libraries
Carnegie Mellon University
Pittsburgh PA 15213-3890



CONTENTS

x. **From the ACM/SIGCHI Chair and Executive Vice-Chair**

xi. **Preface from the General Chairs and Editors**

xii. **Cooperating Societies and Special Thanks**

xiii. **CHI '94 Conference Committee**

xiv. **Papers Chairs, Associate Papers Chairs and Papers Reviewers**

xviii. **Corporate Sponsors**

1. **Papers**

Design in the Workplace

Session Chair: Maddy D. Brouwer-Janse, *Phillips Research Laboratories, IPO*
Discussant: Robert Kraut, *Carnegie Mellon University*

3 **Scandanavian Design: Users in Product Development**
Morten Kyng, *Aarhus University*

10 **Designing Workscape™: An Interdisciplinary Experience**
Joseph M. Ballay, *MAYA Design Group, Inc.*

Accessing and Exploring Information

Session Chair: Ruven Brooks, *Schlumberger*
Discussant: George Furnas, *Bellcore*

16 **An Improved Interface for Tutorial Dialogues: Browsing a Visual Dialogue History**
Benoît Lemaire, Johanna Moore, *University of Pittsburgh*

23 **Using Aggregation and Dynamic Queries for Exploring Large Data Sets**
Jade Goldstein, Steven F. Roth, *Carnegie Mellon University*

30 **An Image Retrieval System Considering Subjective Perception**
Haruhiko Nishiyama, Sumi Kin, Teruo Yokoyama, Yutaka Matsushita, *Keio University*

Supporting Distributed Work

Session Chair: Hiroshi Ishii, *University of Toronto and NTT*
Discussant: Gary Olson, *University of Michigan*

37 **Montage: Providing Teleproximity for Distributed Groups**
John C. Tang and Monica Rua, *SunSoft, Inc.*

44 **Courtyard: Integrating Shared Overview on a Large Screen and Per-user Detail on Individual Screens**
Masayuki Tani, Masato Horita, Kimiya Yamaashi, Koichiro Tanikoshi, Masayasu Futakawa, *Hitachi, Ltd.*

51 **Distributed Collaborative Writing: A Comparison of Spoken and Written Modalities for Reviewing and Revising Documents**
Christine M. Neuwirth, Ravinder Chandhok, *Carnegie Mellon University*, Davida Charney, *Penn State University*, Patricia Wojahn and Loel Kim, *Carnegie Mellon University*

Multimedia in Use

Session Chair: Marilyn Mantei, *University of Toronto*
Discussant: Dennis Egan, *Bellcore*

58 **Marquee: A Tool for Real-Time Video Logging**
Karon Weber and Alex Poon, *Xerox PARC*

65 **A Comparison of the Use of Text and Multimedia Interfaces to Provide Information to the Elderly**
Virginia Z. Ogozalek, *Worcester State College*

Expressive Interfaces

Session Chair: John Thomas, *NYNEX Science and Technology*
Discussant: Phil Barnard, *MRC/APU*

72 **Computers Are Social Actors**
Clifford Nass, Jonathan Steuer and Ellen R. Tauber, *Stanford University*

79 **Form-giving:Expressing the Nonobvious**
Gerda Smets, Kees Overbeeke, *Technische Universiteit Delft*, and William Gaver, *Technische Universiteit Delft, Rank Xerox EuroPARC*

85 **Using a Human Face in an Interface**
Janet H. Walker, *Digital Equipment Corporation*, Lee Sproull and R. Subramani, *Boston University*

Multimedia Interfaces

Session Chair: Lynn Streeter, *U S West*
Discussant: Don Norman, *Apple Computer*

92 **Designing Presentation in Multimedia Interfaces**
Alistair Sutcliffe and Peter Faraday, *City University, London*

99 **The "Starfire" Video Prototype Project: A Case History**
Bruce Tognazzini, *SunSoft*

Active Support for Interaction

Session Chair: Joëlle Coutaz, *LGI-IMAG*
Discussant: Allen Cypher, *Apple Computer*

106 **Creating Charts by Demonstration**
Brad A. Myers, Jade Goldstein and Matthew A. Goldberg, *Carnegie Mellon University*

112 **Interactive Graphic Design Using Automatic Presentation Knowledge**
Steven F. Roth, John Kolojejchick, Joe Mattis and Jade Goldstein, *Carnegie Mellon University*

118 **Repeat and Predict - Two Keys to Efficient Text Editing**
Toshiyuki Masui, *Sharp Corporation* and Ken Nakayama, *The University of Tokyo*

Studies of Communication and Cooperative Work

Session Chair: Janni Nielsen, *Copenhagen Business School*
Discussant: Wendy Mackay, *Rank Xerox EuroPARC*

124 **Communicating About Communicating: Cross-Disciplinary Design of a Media Space Interface**
Beverly Harrison, Marilyn Mantei, Garry Beirne and Tracy Narine, *University of Toronto*

131 **Informal Workplace Communication:What Is It Like and How Might We Support It?**
Steve Whittaker, *Lotus Development Corporation*, David Frohlich, *Hewlett-Packard Laboratories*, and Owen Daly-Jones, *Hewlett-Packard Laboratories, University of York*

138 **A Room of Our Own: Experiences from a Direct Office Share**
Annette Adler and Austin Henderson, *Xerox Corporation*

Tools for Design

Session Chair: Jürgen Ziegler, *Fraunhofer Institute, IAO*
Discussant: Kumiyo Nakakoji, *University of Colorado*

145 **Methods in Search of Methodology — Combining HCI and Object Orientation**
Susan McDaniel, Gary M. Olson and Judith S. Olson, *University of Michigan*

152 **Enhancing the Explanatory Power of Usability Heuristics**
Jakob Nielsen, *Bellcore*

159 **Development and Evaluation of a Model of Behavioral Representation Techniques**
J.D. Chase, Robert S. Schulman, H. Rex Hartson and Deborah Hix, *Virginia Tech*

Auditory Information Interfaces

Session Chair: Beth Wenzel, *NASA Ames*
Discussant: Bill Gaver, *Rank Xerox EuroPARC*

166 **Nonvisual Presentation of Graphical User Interfaces: Contrasting Two Approaches**
Elizabeth D. Mynatt, *Georgia Institute of Technology*, Gerhard Weber, *Universitaet Stuttgart*

173 **The Design and Evaluation of an Auditory-Enhanced ScrollBar**
Stephen A. Brewster, Peter C. Wright and Alistair D.N. Edwards, *University of York*

Accessing and Using Stored Documents

Session Chair: Yvonne Wærn, *Linköping University*
Discussant: Robin Jeffries, *SunPro*

180 **Protofoil: Storing and Finding the Information Worker's Paper Documents in an Electronic File Cabinet**
Ramana Rao, Stuart K. Card, Walter Johnson, Leigh Klotz and Randy Trigg, *Xerox PARC*

186 **The Marks are on the Knowledge Worker**
Alison Kidd, *Hewlett-Packard Laboratories*

Social Aspects of Design

Session Chair: Carrie Rudman, *U S WEST*
Discussant: Austin Henderson, *Xerox PARC*

192 **Raison d'Etre: Capturing Design History and Rationale in Multimedia Narratives**
John M. Carroll, *Virginia Tech*, Sherman R. Alpert, John Karat, *IBM T.J. Watson Research Center*, Mary S. Van Deusen, *InterMedia Enterprises* and Mary Beth Rosson, *Virginia Tech*



198 **Facilitating Effective HCI Design Meetings**
John L. Bennett, *Independent Consultant* and John Karat, *IBM T.J. Watson Research Center*

Designing Spoken Language Systems

Session Chair: Floris van Nes, *Phillips Research Laboratories, IPO*
Discussant: John Richards, *IBM T.J. Watson Research Center*

205 **Interface Techniques for Minimizing Disfluent Input to Spoken Language Systems**
Sharon Oviatt, *SRI International*

211 **An Object-Oriented Approach to Dialogue Management in Spoken Language Systems**
Randall Sparks, Lori Meiskey and Hans Brunner, *U S WEST Technologies*

Automatic Support in Design and Use

Session Chair: James Foley, *Georgia Institute of Technology*
Discussant: Clayton Lewis, *University of Colorado*

218 **Automatic Generation of Interactively Consistent Search Dialogs**
Dan R. Olsen Jr., *Brigham Young University*, and Walter Holladay, *WordPerfect Corporation*

225 **Automatic Generation of Help From Interface Design Models**
Roberto Moriyon, *Universidad Autonoma de Madrid*, Pedro Szekely and Robert Neches, *USC/ISI*

232 **Automating Interface Evaluation**
Michael D. Byrne, *Georgia Institute of Technology*, Scott D. Wood, *University of Michigan*, Piyawadee "Noi" Sukaviriya, *Georgia Institute of Technology*, James D. Foley, *Georgia Institute of Technology* and David Kieras, *University of Michigan*

Evaluation Methods

Session Chair: Guy Boy, *EURISCO*
Discussant: Clare-Marie Karat, *IBM Consulting Group*

238 **The Cost-of-Knowledge Characteristic Function: Display Evaluation for Direct-Walk Dynamic Information Visualizations**
Stuart K. Card, Peter Pirolli, Jock D. Mackinlay, *Xerox PARC*

245 **Comparative Usability Evaluation: Critical Incidents and Critical Threads**
Jürgen Koenemann-Belliveau, *Rutgers University*, John M. Carroll, Mary Beth Rosson, *Virginia Tech*, Mark K. Singley, *ETS*

252 **Usability Testing in the Field: Bringing the Laboratory to the User**
David E. Rowley, *Varian Associates*

Pen Input

Session Chair: Catherine Wolf, *IBM T.J. Watson Research Center*
Discussant: Chris Schmandt, *Massachusetts Institute of Technology*

258 **User Learning and Performance with Marking Menus**
Gordon Kurtenbach, *Xerox Palo Alto Research Center* and William Buxton, *University of Toronto*

265 **T-Cube: A Fast, Self-Disclosing Pen-Based Alphabet**
Dan Venolia, *Apple Computer, Inc.*, Forrest Neiberg

271 **Filochat: Handwritten Notes Provide Access to Recorded Conversations**
Steve Whittaker, *Lotus Development Corporation*, Patrick Hyland and Myrtle Wiley, *Hewlett-Packard Laboratories*

HCI Research?

Session Chair: Susan Dray, *Dray and Associates*
Discussant: Tom Landauer, *Bellcore*

278 **A Preliminary Analysis of the Products of HCI Research, using Pro Forma Abstracts**
William Newman, *Rank Xerox EuroParc*

Design Evaluation

Session Chair: Anker Helms-Jorgensen, *Copenhagen University*
Discussant: Ernest Edmonds, *Loughborough University of Technology*

285 **Supporting Knowledge-Base Evolution with Incremental Formalization**
Frank M. Shipman III and Raymond McCall, *University of Colorado*

292 **Seeding, Evolutionary Growth and Reseeding: Supporting the Incremental Development of Design Environments**
Gerhard Fischer, Ray McCall, Jonathan Ostwald, Brent Reeves and Frank Shipman, *University of Colorado*

299 **Talking Through Design: Requirements and Resistance in Cooperative Prototyping**
John Bowers and James Pycock, *University of Manchester*

Information Visualization

Session Chair: Gitta Salomon, *IDEO Product Design*
Discussant: Jim Hollan, *University of New Mexico*

306 **The Movable Filter as a User Interface Tool**
Maureen C. Stone, Ken Fishkin and Eric A. Bier, *Xerox PARC*

313 **Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays**
Christopher Ahlberg and Ben Shneiderman, *University of Maryland*

318 **The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus+Context Visualization for Tabular Information**
Ramana Rao and Stuart K. Card, *Xerox PARC*

Access to Organized Data Structures

Session Chair: Erik Nilsen, *Lewis and Clark College*
Discussant: Patricia Wright, *MRC/APU*

323 **Evaluating the Influence of Interface Styles and Multiple Access Paths in Hypertext**
Pawan R. Vora, Martin G. Helander and Valerie L. Shalin, *State University of New York at Buffalo*

330 **Multitrees: Enriching and Reusing Hierarchical Structure**
George W. Furnas and Jeff Zacks, *Bellcore*

GOMS Analysis

Session Chair: Jay Elkerton, *NYNEX*
Discussant: Bonnie John, *Carnegie Mellon University*

337 **A Keystroke Level Analysis of a Graphics Application: Manual Map Digitizing**
Peter Haunold and Werner Kuhn, *Technical University Vienna*

344 **A GOMS Analysis of the Advanced Automated Cockpit**
Sharon Irving, Peter Polson and J.E. Irving, *University of Colorado*

351 **A Validation of the GOMS Model Methodology in the Development of a Specialized, Commercial Software Application**
Richard Gong and David Kieras, *University of Michigan*

Visual Interaction Techniques

Session Chair: Saul Greenberg, *University of Calgary*
Discussant: Rob Jacob, *Naval Research Laboratory*

358 **A Taxonomy of See-Through Tools**
Eric A. Bier, Maureen C. Stone, Ken Fishkin, *Xerox PARC*, William Buxton, *University of Toronto* and Thomas Baudel, *Université de Paris-Sud*

365 **The Alphaslider: A Compact and Rapid Selector**
Christopher Ahlberg and Ben Shneiderman, *University of Maryland*

Designing Interaction Objects

Session Chair: Hans Brunner, *U S West*
Discussant: Dan Olsen, *Brigham Young University*

372 **Specification of Interface Interaction Objects**
David A. Carr, *University of Maryland and RMS, Inc.*

379 **Recursive Interfaces for Reactive Objects**
Michael Travers, *Massachusetts Institute of Technology*

HCI in the Real World

Session Chair: Carolanne Fisher, *MAYA Design Group*
Discussant: Dennis Wixon, *DEC*

386 **The Value of a Baseline in Determining Design Success**
Brenda Burkhart, Darold Hemphill and Scott Jones, *Bellcore*

392 **User Preferences for Task-Specific vs. Generic Application Software**
Bonnie A. Nardi, *Apple Computer* and Jeff A. Johnson, *FirstPerson*

399 **Surrogate Users: Mediating Between Social and Technical Interaction**
Deborah Lawrence, Michael E. Atwood and Shelly Dews, *NYNEX Science and Technology*

Evaluating Pointing Devices

Session Chair: Richard Pew, *Bolt Beranek and Newman*
Discussant: Jock Mackinlay, *Xerox PARC*

405 **Children's Use of Mouse-Based Interfaces to Control Virtual Travel**
Erik Strommen, *Children's Television Workshop*

411 **The Effect of Reducing Homing Time on the Speed of a Finger-Controlled Isometric Pointing Device**
Sarah A. Douglas and Anant Kartik Mithal, *University of Oregon*



417 **Two-Handed Input in a Compound Task**
Paul Kabbash, *University of Toronto*, William Buxton, *University of Toronto*, *Xerox PARC*, and Abigail Sellen, *Rank Xerox EuroPARC* and *MRC Applied Psychology Unit*

Analysis of Programming Environments

Session Chair: Javier Lerch, *Carnegie Mellon University*
Discussant: T.R.G. Green, *MRC/APU*

424 **In Search of Design Principles for Programming Environments**
Stephanie Houde, *Apple Computer, Inc.* and Royston Sellman, *Sheffield University*

431 **Programmable Design Environments: Integrating End-User Programming with Domain-Oriented Assistance**
Michael Eisenberg and Gerhard Fischer, *University of Colorado*

Cognitive Models

Session Chair: Michael Pavel, Oregon Graduate Institute
Discussant: John Long, *University College London*

438 **"Why Is a Raven Like a Writing Desk?" Lessons in Interface Consistency and Analogical Reasoning from Two Cognitive Architectures**
John Rieman, Clayton Lewis, *University of Colorado*, Richard M. Young, *MRC Applied Psychology Unit* and Peter G. Polson, *University of Colorado*

445 **A Model of the Acquistion of Menu Knowledge by Exploration**
Andrew Howes, *MRC Applied Psychology Unit*

Interacting in 3-D

Session Chair: Keith Butler, *Boeing*
Discussant: Steven Feiner, *Columbia University*

452 **Passive Real-World Interface Props for Neurosurgical Visualization**
Ken Hinckley, Randy Pausch, John C. Goble and Neal F. Kassell, *University of Virginia*

459 **The "Silk Cursor": Investigating Transparency for 3D Target Acquisition**
Shumin Zhai, *University of Toronto*, William Buxton, *University of Toronto*, *Xerox PARC*, and Paul Milgram, *University of Toronto*

465 **Direct and Intuitive Input Device for 3-D Shape Deformation**
Tamotsu Murakami and Naomasa Nakajima, *The University of Tokyo*

471 **Author Index**

472 **Keyword Index**

475 **Color Plates**

Multitrees: Enriching and Reusing Hierarchical Structure

George W. Furnas and Jeff Zacks

Computer Science Research
Bell Communications Research
445 South Street, Morristown NJ, 07960
Tel: +1 (201) 829-4289 Tel: +1 (201) 829-4320
E-mail: gwf@bellcore.com *E-mail: zacks@bellcore.com*

ABSTRACT

This paper introduces multitrees, a new type of structure for representing information. Multitrees are a class of directed acyclic graphs (DAGs) with the unusual property that they have large easily identifiable substructures that are trees. These subtrees have a natural semantic interpretation providing alternate hierarchical contexts for information, as well as providing a natural model for hierarchical reuse. The numerous trees found within multitrees also afford familiar, tree-based graphical interactions.

KEYWORDS: Information graphs, representation, hierarchies, reuse, directed graphs, hypertext structures, graphical browsers

INTRODUCTION - REPRESENTATION AND REUSE

Two motivations came together in this work. The first was a long standing interest in finding structures for representing information which might be richer than trees yet still viewable and navigable. Trees have a long history in representing information, from the Dewey Decimal system and tables of contents of books, to their use in online information systems and electronic document delivery systems such as Superbook^{tm}[3]. Trees have many strengths. They can be laid out nicely in the plane, they allow simple complete traversal algorithms, they offer a natural analog for the semantic notions of abstraction and aggregation. Trees also have weaknesses. They allow only one way to get from one node to another so there can be no shortcuts, no alternative organizations. In contrast, the fully general graphs often used in hypertext systems have complementary strengths and weaknesses. They allow many routes between things: cross references, multiple organizing contexts, etc., but the structures are not easily laid out, users are easily lost and abstraction is not well represented. Somewhere in between are directed acyclic graphs (DAGs), whose directed links are constrained to have no cycles. DAGs were proposed a decade ago[7] as an alternative information access mechanism. Like trees, the more general DAGs can represent semantic notions of abstraction: classes above subclasses, aggregations above subparts. The network of ISA relations of a knowledge base ("a dog ISA carnivore") and the rich multiple inheritance structures of object oriented systems form DAGs that capitalize on this semantics. For information access, these structures support a top down search strategy like trees, a natural orientation and abstraction mechanism. A problem with DAGs, however, is that like general graphs, they are perhaps underconstrained. In particular, they can be quite difficult to lay out and comprehend. Even small neighborhoods can easily be non-planar causing many edge crossings in viewers. (See [8] for a discussion of some of these topological issues and their implications for interfaces.)

Our first motivation, then, was the question: are there any other places of interest along this spectrum of structure from trees to graphs? In particular, there might be a class of structures in between trees and DAGs, that could at least offer new options in the design space for information representation.

Our second, very concrete motivation came from issues of reuse of hierarchical structure. Under a company-wide initiative huge collections of documentation were being assembled into a large online information resource. The whole assembly formed a great tree of information, with some hierarchy imposed to organize the collection of documents at the higher levels, and lower levels of structure coming from within the documents themselves. This structuring reflected a tentative acceptance of the advantages of the tree approximation of the universe, with its comparative comprehensibility, viewability, and navigability. Individual users, however, were having two sorts of difficulties. For one, the fixed structure of the tree organizing the documentation was indeed not suitable for everyone. As a result some alternative hierarchical organizations of the documentation were made available. Secondly users had trouble with the scale of this aggregate structure. They wanted smaller views, to allow quick access to those parts of the large tree that they needed for their tasks. Some devices were created to allow users to personalize their views of the database. They could collect documents on a virtual "bookshelf" and compile sets of pointers to document sections into lists of "bookmarks." Note however that both of these

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
CHI94-4/94 Boston, Massachusetts USA
© 1994 ACM 0-89791-650-6/94/0330...$3.50

techniques collect fragments of the existing structure only into unstructured lists. One might easily imagine users wanting to organize these sets of fragments into their own private trees, to suit their own purposes. Thus these two problems and their solutions have in common the notion of allowing fragments of some existing tree to be reused, reorganized into new trees. This simple idea turns out to have intriguing consequences.

This paper explores those consequences by introducing a new class of structures called *multitrees*. We will first illustrate the idea of a multitree with a concrete construction scenario. We will then explore some of the formal properties of such structures, particularly properties relevant to interfaces, and illustrate them with sample views from prototype browsers we have built. Then we will question and discuss the constraints imposed by multitrees, finishing with a general discussion.

AN INITIAL EXAMPLE: MULTITREES AND THE COLLEGE PROFESSOR

Reuse of hierarchical structure is not new. It occurs, for example, whenever a college professor creates a structured syllabus of readings for her course. Instead of writing a new document from scratch, she selects fragments from the existing body of literature. For the purposes here we let that body of literature be represented by a great tree, with organization at higher levels coming from, for example, the Dewey Decimal system or the Library of Congress classification system, and the lower levels from the internal hierarchical structure of chapters and sections within documents. From this large tree (Figure 1(i)), Professor A pieces together her own structure of course readings: a book from here, a chapter from there, a section from somewhere else, perhaps even a few short volumes from somewhere. Consider her new structure along with the original tree (Figure 1(ii)). What she has done is to select nodes from the original structure, together with their dependent subtrees, and spin a new hierarchical superstructure, a new tree, above those pieces. A second professor, B, could build another such structure, perhaps even using whole pieces from A's syllabus, or writing and including entirely new tree fragments. The only constraint is that each syllabus be a tree, i.e., it must start from disjoint tree fragments, and assemble them together into some new hierarchy.

The resulting overall structure we call a *multitree*. It has the unusual property that although it is not a tree, the descendents of any node form a tree. Thus familiar tree presentation and navigation techniques may be used to view large fragments of the structure. Perhaps one of the simplest examples of such a structure is when one has two trees sharing the same set of leaves, for example the Dewey Decimal and Library of Congress classification systems for books. General multitrees, however, may share complete subtrees, not just leaves. (By complete subtree we mean a node and all its descendents). That is, hierarchical structure can be shared at multiple granularities.

In hierarchical reuse, more can be shared than just the text itself. Joel Remde (personal communication) has been exploring the use of hierarchical indices for huge simple hierarchies of text. Each node in the simple tree structure has an associated index indicating for each word which immediate descendent node represents a text segment containing that word. One follows a chain of such pointers from index to subindex to find actual occurrences in the leaves. There are various advantages for updating, distributing the database, limiting the scope of searches, and displaying high level results against views of the hierarchy. In multitrees, such hierarchical indices can be shared as well. When Professor A includes a chapter in her syllabus, she gains access to the index for that chapter as well.

Multi-trees are DAGs, not trees, and as a result a node in the structure can have multiple parents. In fact it can therefore have multiple ancestral lineages: one for each tree that (re)used it. Thus looking upward from a node one can see the diverse hierarchical contexts in which the node has appeared. So, whereas looking downward one sees the various ***contents*** under a node (subsection 1, subsection 2, etc.), looking upward one sees its alternative ***contexts*** (as a chapter in Tree1, as Part 1b of Professor A's course, etc.)

There are several further uses for these multiple contexts. For example they can support a special kind of browsing. Suppose Professor A teaches a course on algorithms and includes a particular paper in her syllabus. Using the global

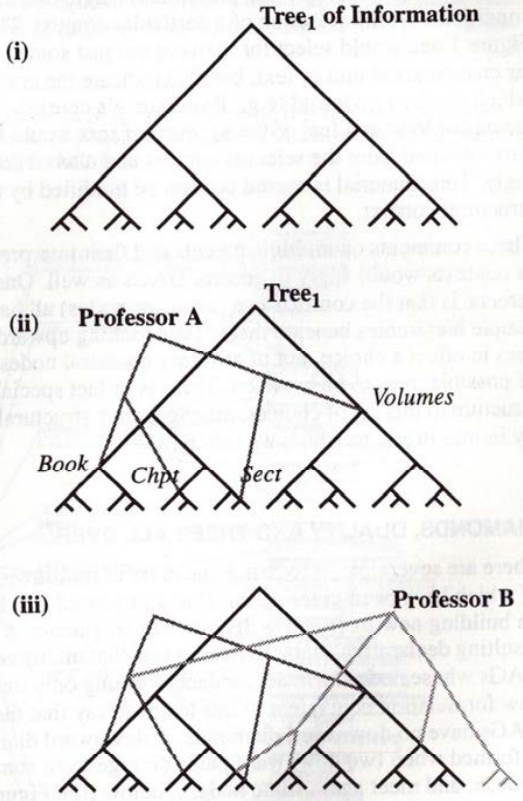


Figure 1 . Constructing a multitree - an example. Beginning with an initial tree of information, Professor A builds her course syllabus. In (a) she chooses a disjoint set of complete subtrees and adds new tree structure above them. Similarly, Professor B makes another, chooses fragments, including some of A's and some completely new ones, and builds a tree above to organize them.

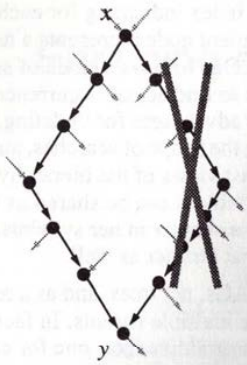


Figure 2 . Diamonds are not permitted in a multitree. A diamond occurs when two distinct directed paths occur between a pair of nodes. Thus in a multitree at most one directed path can exist between two nodes.

structure, she may look upward from that paper to find some other syllabus which also used that paper, move to the context of that syllabus and browse within it to get ideas of other papers to assign.

Contexts can also be used for conditional inheritance. One could, for example, make the presentation style of a node be conditional on the selection of a particular context. Thus in Figure 1 one would select for viewing not just some particular chapter sized unit of text, but also indicate the context in which it is being viewed (e.g., Professor A's course). The formatted look and feel of the segment of text would be in part inherited from the selected context and altered accordingly. Thus material is reused but can be modified by the structural context.

These comments on multiple parents and their interpretation as contexts would apply to general DAGs as well. One difference is that the contexts (i.e., ancestor nodes) all have simple hierarchies beneath them. Thus looking upwards one sees in effect a choice, not of arbitrary ancestral nodes, but of possible trees to be browsed. There is in fact special structure to this set of choices, arising from a structural duality in multitrees, to which we turn now.

DIAMONDS, DUALITY AND TREES ALL OVER

There are several ways to define the class of multitrees, two of which have been given so far. One was procedural, based on building new trees above disjoint tree fragments. A resulting declarative characterization was that multitrees are DAGs whose nodes have descendents forming only trees. A new formulation equivalent to this last is to say that these DAGs have no downward diamonds. A downward diamond is formed when two downward paths diverge from some node, x, and meet again some node, y, below (see Figure 2). Clearly if such a diamond exists, then the descendents of x do not form a tree, and if no such diamond exists they do. This formulation is interesting because it is what is called self-dual, that is it also holds for the links in the DAG considered in the other direction: there are no downward diamonds *iff* there are no upward diamonds. This in turn has the rather surprising consequence that not only does the set of *descendents* of any node form a tree, the set of its *ancestors* also form a tree (an inverted tree). Several such trees can be seen in Figure 1(iii), starting at a leaf and following paths upward that bifurcate to reach different roots. In summary,

Proposition 1. *The following properties are equivalent:*

(a) The DAG can be constructed by adding new tree structure above existing (or newly added) disjoint complete subtrees.

(b) The descendents of any node form a tree.

(c) The DAG is diamond free.

(d) The ancestors of any node form an inverted tree

Any DAG satisfying these conditions is called a multitree.

In this light it is interesting to note that genealogies, though often called "family trees", are in fact not trees at all. Except for intermarriage, which would introduce diamonds, they are multitrees. The property just derived makes sense of the fact that a given individual has both a tree of ancestors and a tree of descendants within the structure. In fact we have found it a useful exercise to represent our own family genealogies with these multitrees, and to test our browsers on them.

For information representation, Proposition 1 means that looking down from any node in the structure one sees a tree of contents, and looking up one sees a tree of contexts. As mentioned already, each of the ancestral contexts itself roots a downward tree. Thus, although the structure itself is not a tree, and not planar in general, it has remarkably large easily identifiable and semantically coherent subsets that are trees, and hence are planar, and amenable to easy layout.

In fact the ubiquity of trees in this structure goes even further. So far, the word "tree" has been used in the common computer science sense of rooted tree (with directed links always pointing away from the root). Following the common graph-theoretic sense of tree, we will say that an undirected graph is what we shall call here a "topological tree" or "t-tree" *iff* it is a connected graph without cycles. T-trees are important since they too are always planar, and allow nice layouts and familiar browsers. Note that though multitrees have no diamonds (a condition on their directed links), when considered undirected they do have cycles (e.g., in Figure 1(ii), the cycle from *Book* to *ProfessorA* to *Volumes* to the $Tree_1$ root back to *Book*), and so they are not topological trees. Yet multitrees do have large topological trees within them, as described in the following proposition. Let us say $x \geq y$ iff x is above y in the DAG.

Proposition 2. *Consider any two nodes $x \geq y$ in a multitree, and the necessarily unique path connecting them. The union of all the ancestors of this path and all the descendents of this path is a topological tree.*

This result is illustrated in Figure 3 and has several practical applications for making useful graphical interfaces for multitrees. We consider some of these in the next section, with illustrations from screendumps of our prototype browsers written in Tk/Tcl.

VIEWING AND BROWSING MULTITREES

First consider when $x=y$. Then we have the simple case of showing simultaneously the tree of descendent contents and the tree of ancestral contexts for an individual node. We call

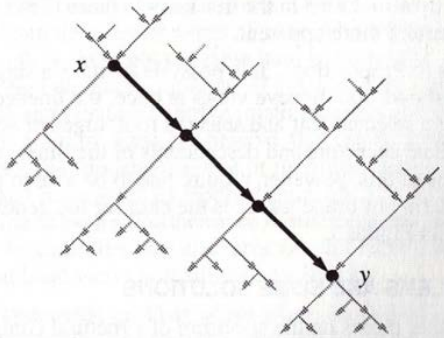


Figure 3 . Proposition 2 illustrated. Two nodes in a multitree, the unique path connecting them, and all the descendents and ancestors of this path together are guaranteed to form a topological tree.

this a *centrifugal* view of the structure, because it follows links in the direction away from some central point of focus.

Figure 4 shows our browser's centrifugal view of a dataset of local Bellcore information. This data set was drawn from a database maintained on our local file system at Bellcore. Users deposit useful information in a particular directory on an ad hoc basis. Anyone with an account is free to add or reorganize material. People have posted material related to corporate practices, life in Northern New Jersey, practical information such as directions, and diatribes on art. To create the multitree shown in these figures, the authors and one volunteer created new hierarchical organizations (trees) that used varying amounts of the pre-existing structure (the "original" tree from the file system). In Figure 4, the central highlighted cell is labeled "Directions." Looking down from the "Directions" node (to the right in this layout) we see the tree of files relating to how one travels to and from various Bellcore locations and related places. Looking up from "Directions" (to the left) we see that "Directions" is a descendent of the original tree and of the three post hoc classifications. Proposition 2 guarantees that both the upward and downward views are trees and that they can be shown simultaneously as single large t-tree. (Note that a centrifugal view may be truncated at any desired radius.)

An interesting consequence of Proposition 2 is that transitions between centrifugal views can be animated nicely. Consider browsing over multitrees using a centrifugal viewer and moving the focus around one link at time, say from node *a* to neighboring node *b*. In such a case, if *a* and *b* are adjacent, then one is above the other, and Proposition 2 applies, meaning that the union of the old and new views is in fact a t-tree, and can be displayed together in 2-D. This allows a nice animation option, showing the transition by first adding the new material, showing both together, then deleting the old.[1]

A third consequence of proposition 2 is for presenting various kinds of fisheye views of multitrees. Furnas[5][6] intro-

1. In practice there is some subtlety in that one might also want layouts to respect sibling order and direction of links (e.g., parents to the left of children).

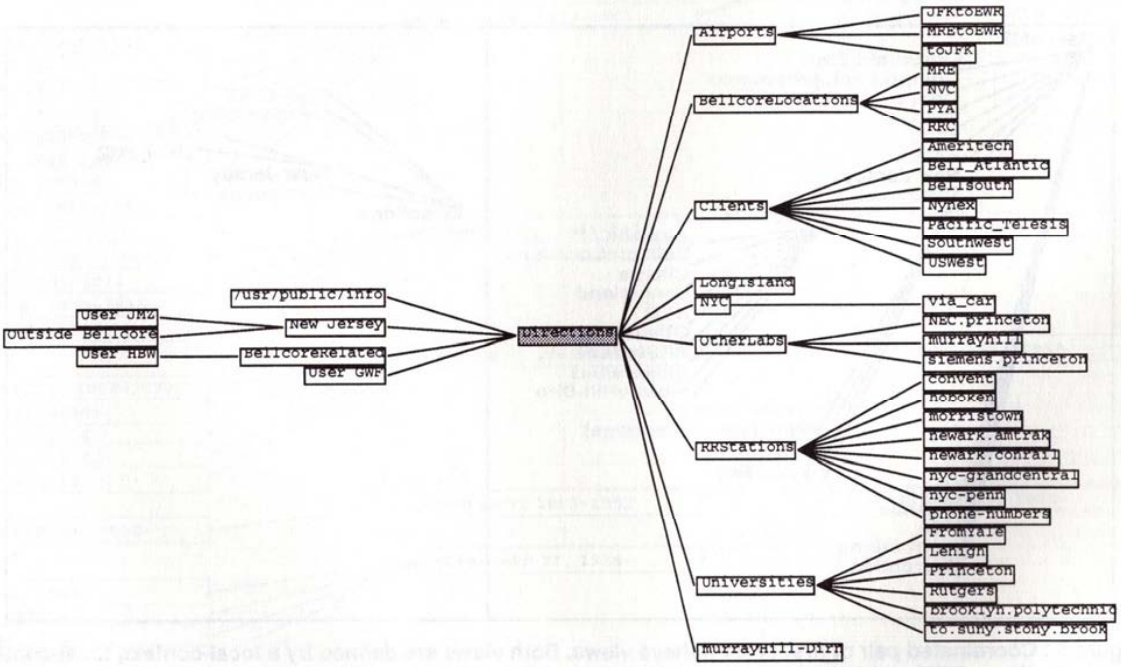


Figure 4 . Centrifugal view of the multitree built upon our /usr/public/info information repository. The view is centered on the node called "Directions" and shows the tree of ancestors (to the left) and the tree of descendents (to the right) of this node.

duced a notion of generalized fisheye views that tried to obtain a balance of local detail and global context. For simple trees it generated a scheme that assigned greatest interest to the ancestral lineage of a focal node, and next interest to nodes one link off that lineage, etc. A similar strategy is possible for multitrees and only requires that one specify not just a focal content node but a focal context node as well.Thus, for example, one could specify not just a leaf (as the focal content), but also a root (as focal context). The root selects a tree, and then an ordinary tree fisheye view can be generated. In the world of multitrees, however, there are a few new twists. For example one can shift around not only the leaf focus, but also the root focus, slowly changing contexts. A second interesting twist arises from the fact that just as there is a tree of contents from any chosen root, there is also an inverted tree of contexts looking upward from any leaf. Thus this inverted tree can also be viewed with a fisheye viewer -- so two fisheye views are possible for any chosen focal-context/focal-content pair.

Figure 5 shows a coordinated pair of first-order fisheye views for the same Bellcore information structure shown in Figure 4. The context focus is on one of the post-hoc organizations of the database and the content focus on the "Directions," both described above. The nodes in the lineage connecting them appear in black, the rest in gray. In the left pane we see the fisheye view of descendent trees; in the right view, the fisheye view of ancestor trees. There are pure indent-based layouts of both of these trees. Ordinary indenting used in familiar outlines is possible for the tree of descendents (a typical fisheye outline view). A complementary "outdenting" structure can be used for the tree of ancestors. This latter view takes getting used to so we have explicitly also drawn in the tree links to these views to make the structure more apparent.

Thanks to Proposition 2 it is possible to make a single view which shows both fisheye views at once: the lineage connecting a selected leaf and selected root, together with immediate ancestors and descendents of that lineage. Pretty versions of this, however, require one to be able to permute the ordering of branches, as is the case for the genealogy shown in Figure 6.

PROBLEMS AND SOME SOLUTIONS

Like other points on the spectrum of structural complexity from trees to graphs, multitrees have their own limitations. In this section we discuss some of these and related problems, as well as some strategies for working around them.

If diamonds are forever...

Multitrees are essentially defined by the absence of diamonds. This is a strong requirement and deserves more discussion. The existence of a single downward path from a node to a descendent is a familiar constraint on simple trees. It is a constraint we have often learned to live with, reflecting the ideal that the structure is at each level a partition. The tree successively refines the disjoint categorization of the world. If this ideal has merit then multitree structures offer a useful extension -- merely providing a family of alternative classifications in a single structure. (In fact, as it turned out from Proposition 1(c), tree-structured families of such classifications.)

Diamonds, however will not always go away. There are several circumstances under which we might want to consider

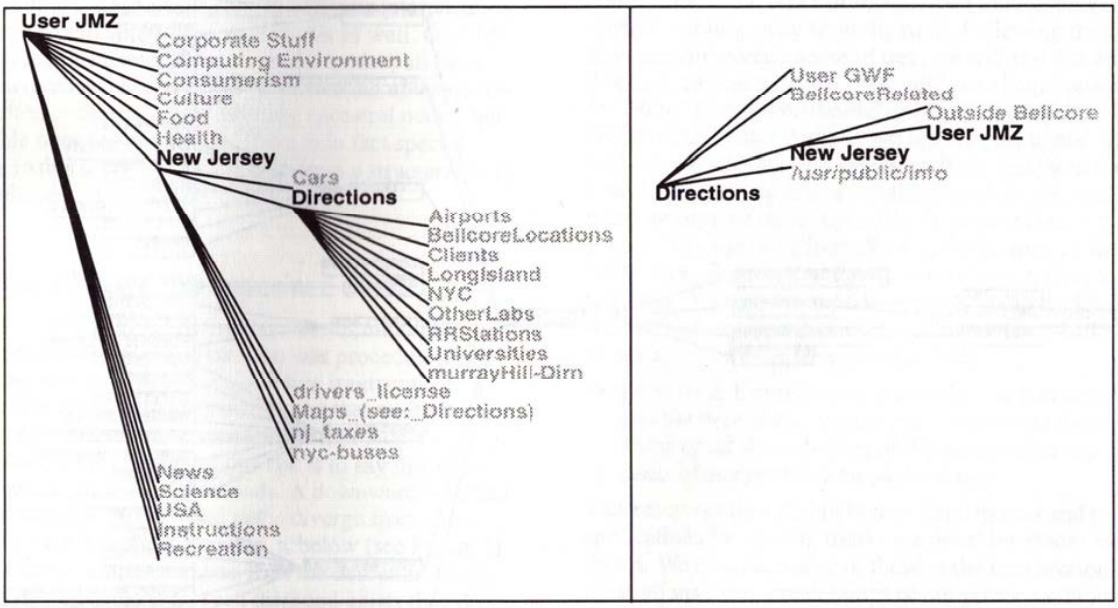


Figure 5 . Coordinated pair of first order fisheye views. Both views are defined by a focal-context, focal-content pair, here "User JMZ" and "Directions". The view on the left is a simple tree fisheye view of the downward tree of contents rooted at the context "User JMZ". It shows the ancestral lineage linking the focal-context and the focal-content, and the first-order descendents of the lineage. The view on the right is a similar tree fisheye view of the *inverted* tree of *contexts* rooted at the content "Directions". It too shows the ancestral lineage linking the focal-context and the focal-content, but this time with the first-order ancestors (contexts) of the lineage.

relaxing the diamond-free constraint. One case comes directly from the simpler world of trees, where from time to time, people want to put the same item in more than one place in the tree. Such violations of tree structure introduce diamonds and would similarly violate multitree structure. If diamonds are infrequent, we believe, this can be handled as with trees: The fundamental tree or multitree ideal is preserved, and some small exceptions are made, e.g., by virtual duplication of nodes. Furthermore if diamonds are long, as is often the case, then the structures are still locally multitrees, and local views will still behave ideally.

A more systematic problem arises when one has an internal node below which one would like to offer a choice of organization for descendents. For example one might have a subset of documents to be organized for access both alphabetically and by date. A whole suite of diamonds would begin at this node, and end at each of the cross-classified documents. For such cases, one might consider a new class of structures slightly more general than multitrees but from which sub-multitrees can be easily extracted. One would just need a way to break the diamonds in a systematic way to generate tree views as needed. For example, suppose any such node would be flagged (e.g., colored red) to indicate that its children offer two alternative breakdowns of its descendents, and that one must be chosen to preserve tree structure, and a default used in any needed view. A particularly clean version of this would require that its children really all have the same descendents, so that the semantics of the flagged node would be un-altered by the choice of subsequent classification.

A related problem concerns the organization of roots. In particular, it will be noted that there is no top-down structure over the set of roots (e.g., in Figure 1(iii)). It should be remembered, however, that the various contexts (including the various roots) can indeed be browsed, not top-down but from the bottom up, from the point of view of a particular focal content. If one wants to guarantee a view of all roots one could introduce an artificial leaf, descendent from all roots, whose upward view by design is therefore a tree containing all those roots.

In the end, of course, if hierarchical partition is rare, and partial overlap and random interconnection is the rule, then neither trees nor multitrees will be appropriate, and one needs to move back in the direction of general DAGs for representation, paying the corresponding price in complicating the interface.

Viewing the whole structure

For some applications a view of a non-tree portion of a multitree (e.g., the entire multitree) may be desirable. This might arise in particular when building on an existing multitree and wanting to take fragments from different subtrees, or perhaps in some unusual browsing situations. The fact that multitrees are not generally planar presents a challenge, and the problem becomes similar to one of general DAG visualization and browsing. A 3-dimensional layout may be

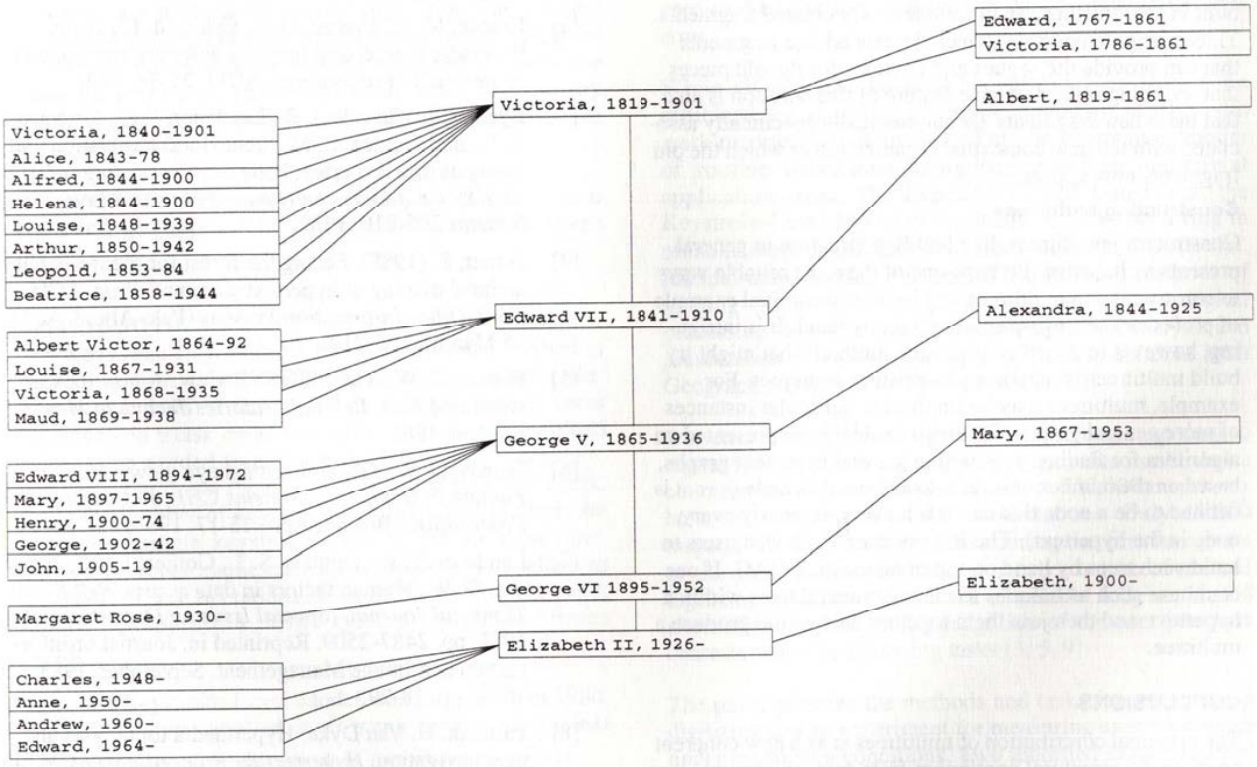


Figure 6 . Integrated fisheye view for a royal genealogy. This view shows both upward and downward fisheye views at once, i.e., the ancestral lineage (from Queen Victoria to Queen Elizabeth II), as well as both first order ancestors and first order descendants of that lineage.

best for this, with adjacent nodes placed near each other, and the direction of links preserved in the layout. Multitrees may allow special 3-D arrangements, since whole tree portions of interest may be assigned to their own planes in the structure.

Reuse out of context

There is a problem that always arises in reuse, no less for hierarchical structure and multitrees. That is, fragments composed in one specific context may not work well out of that context. This problem comes up in code reuse, where special discipline is needed to design and encapsulate modules so that they may be called from various contexts. In text systems fragments can contain anaphora, explicit reference to other sections, conceptual prerequisites, etc. all of which are resolved in a well written linear text. This has always caused problems for hypertext, where fragments may be encountered in various orders. The typical solution is to try to write fragments that stand alone, leading to an often jerky style.

For hierarchical reuse, the problem would be that a new tree assembled from fragments taken from elsewhere would not hang together well. The individual readings in a professor's course, just plucked out and glued together, might well lack establishing context and smooth transitions. There are two strategies however that should greatly ameliorate the problem. The first is to admit that some points in an existing structure might be more suitable candidates than others for reuse, and make the most use of those pieces that have less external dependency. The second is to note that new trees built in the structure can introduce newly created fragments. Thus it is quite possible to include new ad hoc fragments that can provide the segues and contexts for the old pieces that are being reused. A nice feature of this solution is also that these new fragments are automatically specifically associated with the new contextual organization in which the old fragments now appear.

Constructing Multitrees

Constructing multitrees, like building structure in general, presents a challenge. Perhaps one of the most reliable ways to construct good multitrees will be as in our initial example of professors setting up courses, i.e., by hand. It is intriguing, however to consider automatic methods that might try build multitrees from some pre-existing resources. For example, multitrees may be implicit in particular instances of more general graphs. Botafogo et al.[1] have presented an algorithm for finding trees within general hypertext graphs, based on the number of links into and out of a node (a root is defined to be a node that can reach every, or nearly every, node in the hypertext). The IGD system[4] allowed users to build such trees by hand on top of an existing DAG. If one could use such techniques to identify several trees within a hypertext, and then join them together, this would produce a multitree.

CONCLUSIONS

The principal contribution of multitrees is as a new coherent point on the structural continuum from trees to general graphs. Like all other points on that spectrum, they have their strengths and weaknesses, both in their graphical and semantic capabilities. As with DAGs in general, showing a whole multitree can be difficult, but large meaningful fragments can be shown as simple trees. Multitrees have natural notions of content and context, and seem appropriate for cases of reuse of hierarchical structure. Their appropriateness for more general representation (e.g., knowledge bases, or object oriented class systems) is an open question, but still they may provide a useful approximation just as trees sometimes do. For example Creech, et al.[2] have implemented a hypertext system for software reuse based on DAGs. It would be interesting to see if multitrees could serve as well. Even if multitrees are too constraining, it is possible that they can form a distinguished backbone, with other types of links serving to complete the structure. Thus the tree pieces can be used for some viewing and navigation purposes, and abandoned as needed.

Acknowledgments

We would like to thank Susan Dumais, Jakob Nielsen, and Scott Stornetta and Henri Weinberg for variously trying the browsers and reading drafts of this paper.

REFERENCES

[1] Botafogo, R.A., Rivlin, E., Schneiderman, B., Structural analysis of hypertext: identifying hierarchies and useful metrics, *ACM Transactions on Information Systems*, 10(2), 1992, 142-180.

[2] Creech, M. L., Freeze, D. F., Griss, M. L., Using hypertext in selecting reusable software components, *Hypertext'91Proceedings,* ACM, 25-38, 1991.

[3] Egan, D. E., Remde, J. R., Landauer, T. K., Lochbaum, C. L., and Gomez, L. M., Behavioral Evaluation and Analysis of a Hypertext Browser, *Proceedings of ACM CHI'89 Conference on Human Factors in Computing Systems* 205-210, 1989.

[4] Feiner, S. (1988). Seeing the forest for the trees: Hierarchical display of hypertext structure. *Proc. ACM Conf. Office Information Systems* (Palo Alto, CA, 23-25 March), 205-212.

[5] Furnas, G. W., The FISHEYE view: A new look at structured files. *Bell Laboratories Technical Memorandum*, 1982.

[6] Furnas, G. W., Generalized fisheye views. *Human Factors in Computing Systems CHI '86 Conference Proceedings*, Boston, April 13-17, 1986, 16-23.

[7] Landauer, T. K., Dumais, S. T., Gomez, L. M., & Furnas, G. W., Human factors in data access. *Bell System Technical Journal, (Special Issue on Data Bases)*, 61, 1982, pp. 2487-2509. Reprinted in: Journal of Information and Image Management. September, 1983, Vol 16(9), pp. 18-29.

[8] Parunak, H. Van Dyke, Hypermedia topologies and user navigation, *Hypertext'89 Proceedings*, ACM, 43-50, 1989.

Carnegie Mellon Offsite Facility
A020411
1 DAY LOAN

004.019 C53h 1994 c.2
CHI '94 Conference
Human factors in computing
systems :

Carnegie
Mellon
University Libraries
Pittsburgh PA 15213-3890
DEMCO

ROOM USE ONLY
UNTIL AUG 0 4 1994