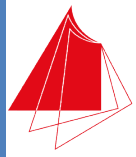
22. 05. 2009

Windows 7™

SEMINARARBEIT

SYSTEMARCHITEKTUR



Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Von Simon Loschko | betreut durch Prof. Dr. rer. nat. Lothar Gmeiner

Inhaltsverzeichnis

1. Vorwort

In dieser Seminararbeit beschäftige ich mich mit dem zukünftigen Betriebssystem „Windows 7“ und dessen grundlegenden Systemkomponenten und Systemmechanismen. Im Vordergrund steht dabei die elementare Architektur, die auf den Vorgängerversionen Windows Vista und Windows XP aufsetzt. Des Weiteren wird das Betriebssystem Windows 7 in die Geschichte der Microsoft Betriebssysteme eingeordnet.

Leider sind bisher nur wenige Informationen über das Innenleben von Windows 7 in der Öffentlichkeit bekannt. Daher wurden die aktuellen und teilweise noch geheimnisvollen Informationen aus mehreren Quellen entnommen. Sie beruhen auf dem Stand der Windows 7-Versionen: Beta 1 (Build 7000) und RC 1 (Build 7100). Man kann jedoch davon ausgehen, dass es keine großen Änderungen, bis zum Erscheinen der RTM[1]-Version gegen Ende des Jahres mehr geben wird. Verwendet wurden vorwiegend offizielle Quellen von Microsoft aber auch einige inoffizielle Quellen, deren Richtigkeit jedoch meinerseits stark angenommen wird. In dieser Seminararbeit werden die bedeutenden Komponenten nur sehr flüchtig vorgestellt. Das Thema ist so umfangreich, dass es problemlos möglich wäre, jede einzelne Komponente in einer eigenen Seminararbeit ausführlicher zu betrachten.

Insiderwissen zur Entwicklung (direkt von den Entwicklern von Windows 7), findet man unter den folgenden Links in Form von Blogs, Berichten, Interviews und Videos:
http://windowsteamblog.com/blogs/
http://blogs.technet.com/dmelanchthon/
http://blogs.technet.com/markrussinovich/
http://blogs.technet.com/springboard/
http://blogs.technet.com/sieben/
http://blogs.msdn.com/e7/
http://channel9.msdn.com/
http://channel9.msdn.com/tags/Windows+7/
http://www.microsoft.com/whdc/system/vista/kernel-en.mspx

Wer sich tief gehender mit der Architektur von Windows beschäftigen möchte, dem sei die Literatur zum Thema „Windows Internals“ von David A. Solomon und Mark E. Russinovich empfohlen. Das aktualisierte Buch ist demnächst in englischer Sprache in der 5. Auflage bei Microsoft Press Corp. erhältlich.

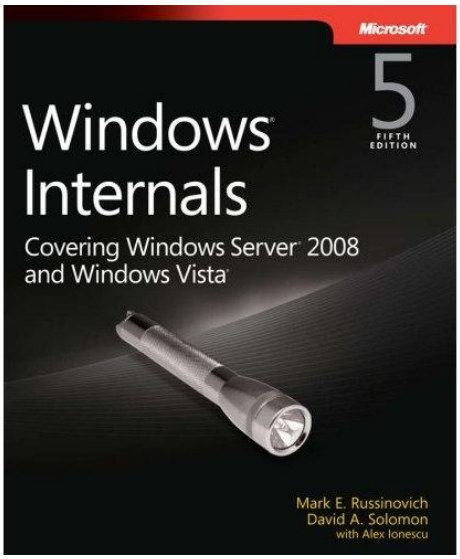


Abbildung 1: Das Standardwerk über den internen Aufbau von Windows

[1] Release to Manufacturing, Begriff aus der Softwareentwicklung

2. Einführung: Windows 7, ein neuer Vertreter der Windows NT Technologie

Spätestens im ersten Quartal 2010 wird die Finalversion des Windows Vista Nachfolgers mit dem Namen "**Windows 7"** veröffentlicht. Das Betriebssystem Windows 7 (NT 6.1) wird auf der seit 1993 entwickelten NT-Architektur, welche ab Windows NT 4.0 im Jahre 1996 so erfolgreich wurde und deren Fortsetzung in Windows 2000 (NT 5.0) und Windows Vista (NT 6.0) erfolgte, aufgebaut sein.

Über einen langen Zeitraum hatten Microsoft Betriebssysteme den Ruf instabil zu sein und technisch hinterher zu hinken. Dem hat Microsoft jedoch nach der Einführung der ersten Windows NT Version 3.1 im Jahr 1993 gegengewirkt und mit der NT-Architektur, als Gegenpol zu MS-DOS, eine fortschrittliche Grundlage geschaffen, deren Kernprinzipien nach 17 Jahren heute immer noch von großer Bedeutung sind und daher im aktuellen Windows 7 teils relativ ähnlich oder sogar in Teilen identisch wiederzufinden sind.

Die Entscheidung des Windows 7-Entwicklungsleiters Steven Sinofsky (Senior Vice Präsident Windows und Windows Live Engineering Group), das System auf der bestehenden Windows NT-Architektur und den zugehörigen Subsystemen aufzubauen, wurde nach der Bekanntgabe in der Öffentlichkeit gründlich diskutiert. Siehe Abbildung 2: NT-Architektur in Windows 7 (pro und contra). Laut diverser Microsoft Entwickler ist es nicht ohne Weiteres möglich, sich vom aktuellen Kernel einfach zu verabschieden und das Betriebssystem auf einer neuen Basis vollkommen neu zu entwickeln. Der Grund hierfür ist, dass man bei einem Produkt, das so umfangreich und erfolgreich wie Windows ist, nicht ohne weiteres substanzielle Änderungen vornehmen kann. Ein Beispiel hierfür ist Windows Vista. Es hat relativ wenige Veränderungen an den grundlegenden Elementen des Betriebssystems gegeben, doch selbst diese minimalen Änderungen hatten Auswirkungen auf die Anwendungskompatibilität, die für Unzufriedenheit unter den Kunden von Microsoft sorgte. Das bedeutet, dass selbst minimale Modifikationen extrem schwierig durchzuführen sind. Insgesamt ist Windows ein derart kompliziertes Produkt, dass Microsoft die grundlegenden Elemente nicht einfach im Rahmen eines Neuanfangs über den Haufen werfen kann.

Nun eine kurze Übersicht (pro und contra) von gesammelten Meinungen und Fakten zum Thema Windows NT-Architektur in Windows 7:

Vorteile:	Nachteile:
Gewährleistet eine optimale Abwärtskompatibilität -> bestehende Anwendungen können weitergenutzt werden.	Windows NT-Subsysteme und Kernelumgebung ist zu füllig geworden, da sie seit 1993 konstant gewachsen ist.
Bisherige WDM-Treiber (Windows-Treiber-Modell) der Hersteller können weiterwendet werden -> Windows 2000 und Windows Vista Hardware kann weiterverwendet werden.	Neue Kernelentwicklungen aus dem Hause Microsoft wie Singularity oder Midori sind kompakter, objektorientiert und total modular aufgebaut, was für Mobile und Embedded Geräte besser wäre.
Updaten schneller möglich mit geringerem Lernaufwand als bei ganz neuem System	Absoluter Neuanfang ohne die Altlasten vergangener Windows Versionen
Nach 17 Jahren wurde es erreicht, den Kernel und seine unmittelbare Umgebung sehr stabil und nahezu fehlerfrei zu bekommen.	Die derzeitige und auch die zukünftige Hardware kann von Grund auf besser berücksichtigt werden.

Abbildung 2: NT-Architektur in Windows 7 (pro und contra)

Bevor wir uns mit dem Aufbau der Windows NT-Architektur, vorwiegend der von Windows 7 beschäftigen werden, folgt nun ein Kapitel indem die Entwicklung und die Geschichte der Microsoft Betriebssysteme vor Windows NT (ab 1993) erläutert wird. Hierbei zu beachten ist, dass Microsoft

ursprünglich zwei Betriebssystemlinien angeboten hatte. Die Consumerlinie (für Privatanwender) bestand aus MS-DOS und der Windows GUI (anfangs noch optional). Die Businesslinie (für Geschäftskunden und erfahrene Benutzer) dagegen war stets MS-DOS frei und wurde ab dem Jahr 1993 auf NT-Technologie basierend aufgebaut. Eine Zusammenführung beider Linien sollte mit Windows 2000 erfolgen, was sich aber endgültig erst mit Windows XP erreichen ließ. Somit wurde von Microsoft ab 2001 nur noch eine Linie für Verbraucher und Firmenkunden zusammen entwickelt.

3. Neues in der Windows 7-Architektur

Mit Windows 7 wurden geringfügige Änderungen am Kern und den darüber liegenden Teilsystemen durchgeführt, die nun kurz erläutert werden. Eine genauere Darstellung dieser Änderungen ist in den einzelnen dazugehörigen Kapiteln dieses Dokuments zu finden.

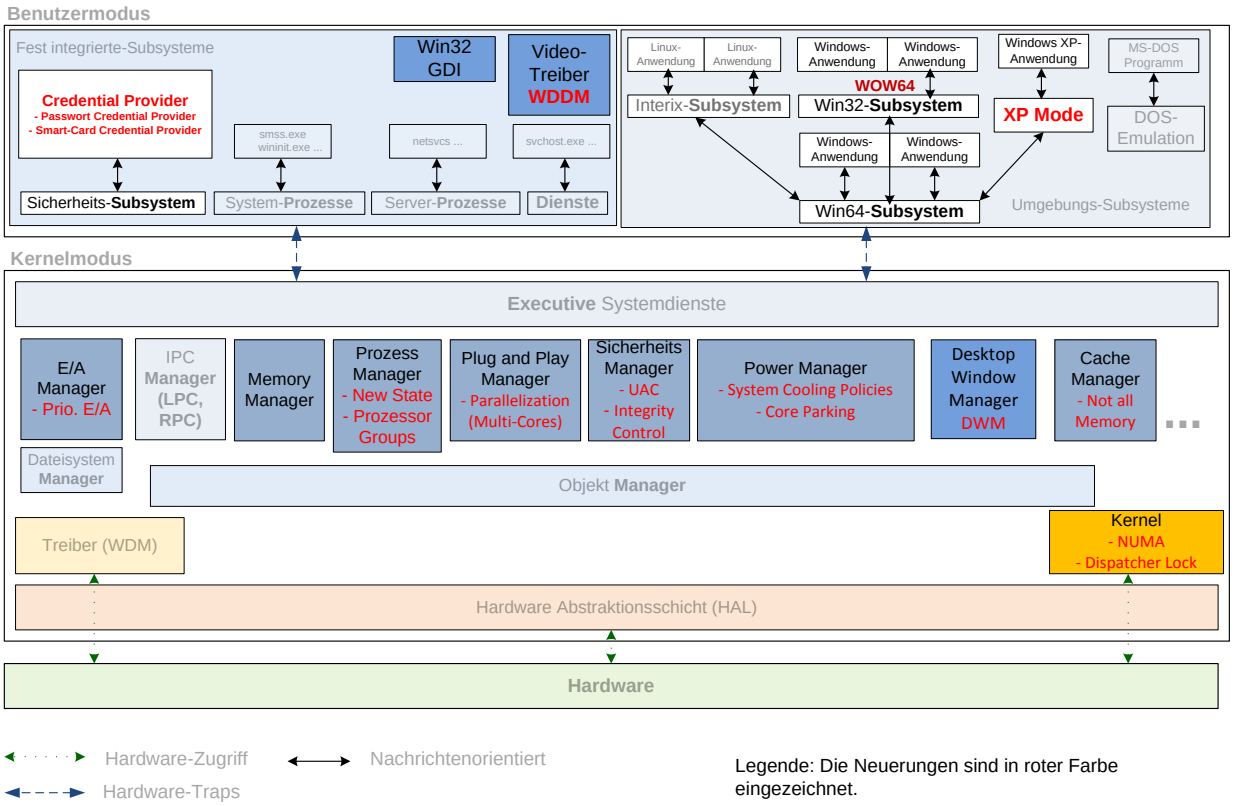


Abbildung 3: Neues in der Windows 7 Architektur

3.1. Booten der Windows Installation von USB-Geräten

Der Windowskern und die grundlegenden Teilsysteme können nun „Out-of-the-box“ von einem **USB-Stick** gestartet werden. Somit kann die Installationsumgebung, die ein minimales Windows darstellt, das ausschließlich zur Installation oder Reparaturzwecken verwendet wird, über einen USB-Stick gestartet werden.

3.2. User Account Control (Benutzerkontensteuerung) und Datei- und Registryvirtualisierung

Die UAC (User Account Control) wurde mit Windows Vista eingeführt und erhielt in Windows 7 geringe Verbesserungen. Sie setzt unmittelbar auf dem Sicherheitsmodell von Windows 7 auf, dass im Kapitel 5.21 erklärt wird. Die Funktionsweise der UAC ist in einem guten Artikel zusammengefasst, der unter http://technet.microsoft.com/en-us/magazine/cc138019.aspx abgerufen werden kann. Ziel der UAC ist es, Anwendungen im Benutzerkontext laufen zu lassen, auch wenn der Anwender mit einem administrativen Benutzerkontext angemeldet ist. Der administrative Kontext soll nur bei Bedarf aktiviert werden, um die Angriffsfläche für fehlerhafte Anwendungen und Viren zu minimieren. Außerdem sorgt eine Datei- und Registryvirtualisierung dafür, dass alte Anwendungen die ausschließlich mit administrativen Rechten arbeiten auch unter normalen Benutzerrechten arbeiten.

3.3. Anderes Verhalten beim Cache Manager

Wird erläutert in Kapitel 5.23.

3.4. Anderes Verhalten beim Dienstemanager

Einige Systemdienste werden nur bei Bedarf gestartet; damit soll der Rechnerstart merklich schneller funktionieren. Außerdem werden Dienste, die nicht benutzet werden, automatisch beendet und bei einer erneuten Benutzung erneut gestartet. Dieser Mechanismus spart Arbeitsspeicher und verringert gleichzeitig die Angriffsmöglichkeiten auf nicht verwendete Dienste.

3.5. Treiber Parallelisierung

Das Laden von Gerätetreibern wird parallelisiert, damit der Startprozess schneller durchgeführt werden kann. Dabei werden die Threads bei der Treiberinitalisierung bereits in der Startphase des Betriebssystems auf mehrere Prozessorkerne verteilt.

3.6. Credential Provider

Bis zur Version Windows XP wurde für die sichere Benutzerauthentifizierung und die interaktive Anmeldung die Softwarekomponente GINA[2] verwendet. GINA ist eine Dynamic Link Library (MSGINA.DLL), die zusammen mit dem Winlogon-Prozess beim Systemstart geladen wird. Sie verarbeitet unter anderem die Tastenkombination zur Anmeldung (Strg+Alt+Entf). Nach der ersten Anmeldung erstellt sie den ersten Benutzerprozess (Desktop und Taskleiste). Mit Windows Vista wurde GINA durch den neuen Credential Provider (LogonUI.exe) erneuert, der komplett modular aufgebaut ist. Es gibt zwei Module von Microsoft die im Betriebssystem enthalten sind: **Passwort Credential Provider** und **Smartcard Credential Provider**. Es werden verschiedene weitere Module unterstützt. So können Hersteller für spezielle Anmeldehardware wie z. B. Fingerprint, ein Modul selbst ausliefern und in den Credential Provider integrieren.

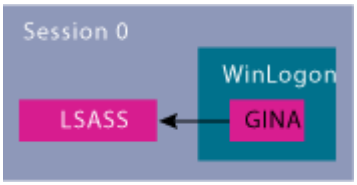


Abbildung 4: GINA vor Windows Vista

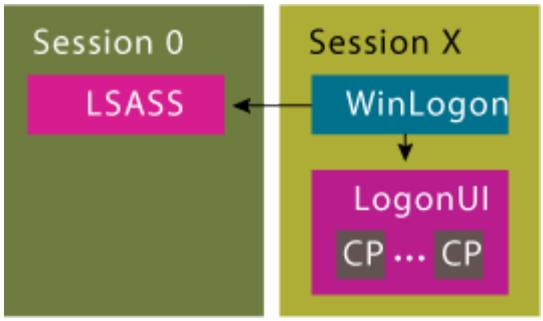


Abbildung 5: Credential Provider in Windows 7

[2] Graphical identification and authentication

Der Passwort Credential Provider übernimmt die Funktionen der ehemaligen GINA:

- Authentifizierung des Benutzernamens und Passworts mit einem Domain Controller oder mit dem lokalen Computer
- Anzeige von Statusinformationen, bevor der Anmeldedialog dargestellt wird.
- Automatische Anmeldung für Wartungszwecke. Die automatische Anmeldung kann so konfiguriert werden, dass sie nur eine bestimmte Zahl von Anmeldungen durchführt, bevor wieder eine Passwortabfrage erscheint.
- Bereitstellung des Dialogs „Windows Sicherheit“, mit diesem kann das Betriebssystem herunterfahren werden, oder ein Benutzer kann sich abmelden, sein Passwort ändern, den Taskmanager starten oder den Computer sperren.

3.7. Verbesserte Unterstützung der NUMA-Architektur[3] (Non-Uniform Memory Access)

Die 64-Bit-Versionen von Windows 7 und Windows Server 2008 R2 unterstützen nun dank Änderungen im Kernel, mehr als die ursprünglichen 32 logischen Prozessoren auf einem einzelnen Computer mit Non-Uniform Memory Access (NUMA) Hardware-Architektur. Windows 7 wird zukünftig bis zu 256 logische Prozessoren (also 256 Kerne) unterstützen. Das traditionelle Modell für die Mehrprozessor-Unterstützung ist Symmetric Multiprozessor (SMP). Es wird in Kapitel 5.10 näher beschrieben. In diesem Modell hat jeder Prozessor den gleichen Zugang zum Speicher und zur E/A[4]. Wenn mehrere Prozessoren hinzugefügt werden, ist die System-Performance deutlich durch den Prozessor-Bus beschränkt. Bei einem Computersystem wird jeder Prozessor-Kern in Windows als logischer Prozessor dargestellt. Der logische Prozessor ist sichtbar für die Anwendungen oder die Treiber. Um mehr als 32 logische Prozessoren unterstützen zu können, wurde das Konzept der Prozessorgruppen eingeführt. Siehe Kapitel 5.8, Abschnitt: **„Ein neuer Zustand und Prozessorgruppen“.** Der Windows 7 Kernel musste hierfür verändert werden. Siehe Abschnitt „**Die Überwindung des Dispatcherlocks (Lastenverteilers)“** in Kapitel 5.10.

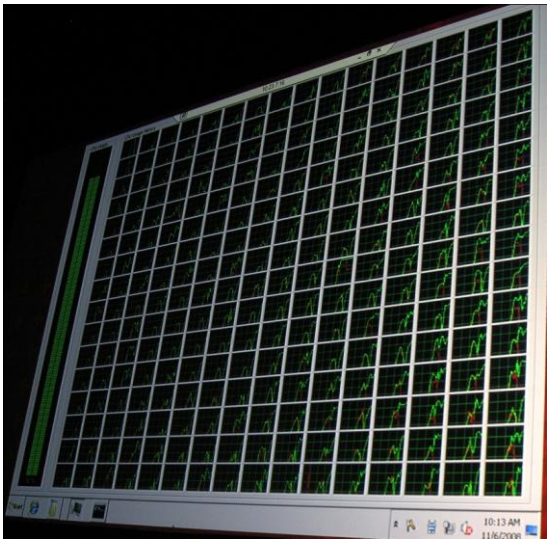


Abbildung 6: Windows 7 mit 256 Prozessoren

[3] Computer-Speicher-Architektur für Multiprozessorsysteme, bei denen jeder Prozessor eigenen, lokalen Speicher hat, aber anderen Prozessoren über einen gemeinsamen Adressraum direkten Zugriff darauf gewährt. (Distributed Shared Memory)

[4] Ein-/Ausgabe (abgekürzt E/A; engl. Input/Output; kurz I/O) ist ein Bereich der EDV. Als Bestandteil des Eingabe-Verarbeitung-Ausgabe-Prinzips und der Von-Neumann-Architektur ist er ein zentraler Bestandteil der Informatik.

3.8. Mandatory Integrity Control

Wird erläutert in Kapitel 5.21.

3.9. Geringerer Ressourcenverbrauch

Eine häufig zu lesende Befürchtung ist der steigende Ressourcenverbrauch von neueren Windows-Versionen. Microsoft hat bei der Entwicklung von Windows 7 sehr viel in die Reduzierung des Ressourcenverbrauchs investiert. An verschiedensten Stellen des Betriebssystems wurden Änderungen durchgeführt, um die Arbeit schneller zu machen und die Antwortgeschwindigkeit des Systems zu verbessern. Steve Sinofsky (Senior Vice Präsident Windows und Windows Live Engineering Group), zeigte auf der PDC[5] Windows 7 auf einem kleinen Netbook mit 1 GHz CPU und 1 GB RAM und sagte, dass die Hälfte des dort eingebauten Hauptspeichers nach dem Booten unter Windows 7 noch frei sei. Genaue Messungen haben ergeben, dass Windows 7 auf einem System mit 1 GB Arbeitsspeicher rund **344 MB für sich** beansprucht. Des Weiteren wurden die Festplatten Ein- und Ausgaben deutlich reduziert was auch aufgrund der verbesserten Windows Suche, ein Indexdienst der Dateien auf der Festplatte und im Netzwerk indiziert um diese schneller finden zu können, erreicht wurde. Dieser Indexdienst wird durch 3 Systemprozesse: SearchIndexer, SearchFilterHost und SearchProtocolHost ausgeführt. Diese gehen zukünftig noch verantwortungsvoller mit den Systemressourcen um. Sie dürfen keine Auswirkungen auf die Performance des Systems auf negative Art und Weise haben.

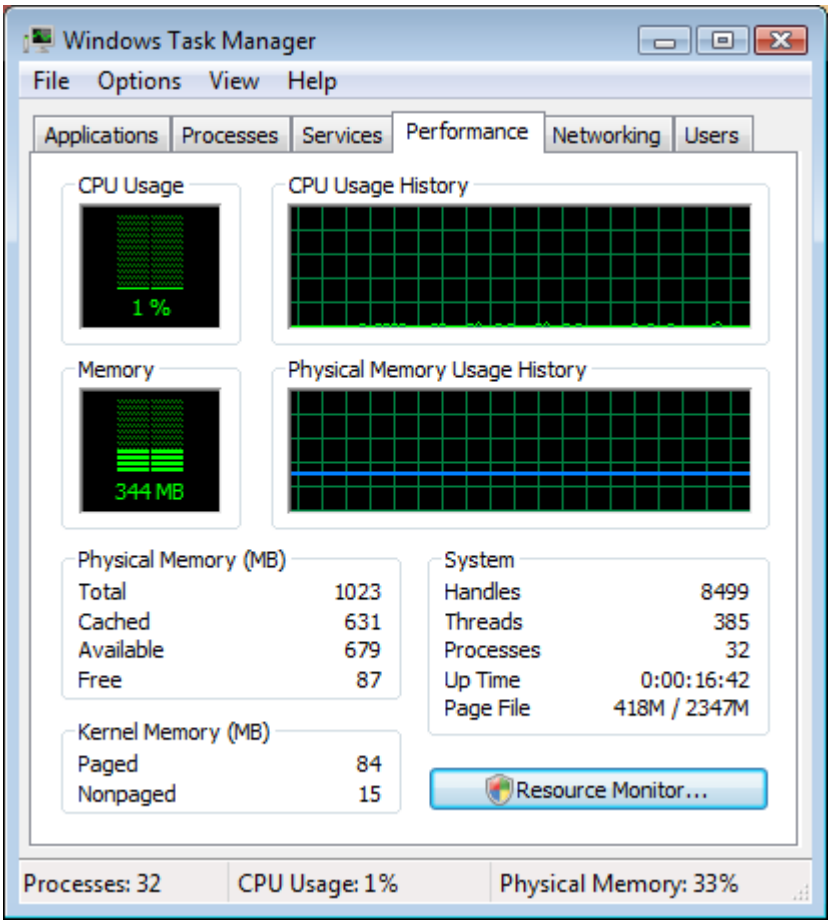


Abbildung 7: Speicherbedarf bei Windows 7 mit 1 GB Arbeitsspeicher

[5] Professional Developers Conference, Entwickler-Konferenz über neue Microsoft Produkte/Technologien

3.10. Priorisiertes E/A

Wird erläutert in Kapitel 5.15.

3.11. Win32 GDI mit verbesserten Antwortzeiten bei sehr vielen geöffneten Fenstern

Der Desktop Window Manager (DWM) ist für die Darstellung der Fenster auf dem Desktop zuständig. DWM wird durch das WDDM[6] angesteuert. Das neue WDDM-Treibermodell und eine neue Implementierung der Grafikkomponenten in Assembler-Code sind in der Lage den Speicherverbrauch zu reduzieren. Im Gegensatz zu Windows Vista ist der Speicherverbrauch für die Fensterdarstellung stets konstant also unabhängig von der Anzahl der geöffneten Fenster. Der Desktop Window Manager (DWM) wird im Kapitel 5.24 näher beschrieben.

3.12. Energiesparen

Durch ein stark verbessertes **Power Management** sind mit Windows 7 längere Akkulaufzeiten möglich als bei Windows Vista. Zwei neue Mechanismen wurden eingeführt: **System Cooling Policies** und **Core Parking**. Mit System Cooling Policies wird das Verhalten des Lüfters zusammen mit der Steuerung der Energiespar-Modi des Prozessors beeinflusst. Nach Wunsch des Anwenders dreht der Lüfter schneller (lauter) oder langsamer (leiser). Bei mobilen Computern, mit einem Akku, kann dann die System Cooling Policy getrennt für den Akku- und den Netzbetrieb eingestellt werden. Mit Core Parking wird Windows 7 in der Lage sein, die aktuelle CPU-Auslastung pro CPU zu überwachen und dynamisch einzelne Kerne abschalten zu können. Dadurch kann gerade bei Rechnern mit sehr vielen Kernen deutliche Energieeinsparung im Vergleich zum herkömmlichen Betrieb, bei dem jeder Kern möglichst gleichmäßig belastet werden würde, erreicht werden. Angenommen man arbeitet auf einer CPU mit vier Rechenkernen, die alle zu 20 Prozent ausgelastet sind. Dann ist es energieschonender, wenn alle Programme von einem Kern bedient werden, der dann zu 80 Prozent ausgelastet ist. Die restlichen drei Kerne legen sich schlafen.

[6] Windows Display Driver Model

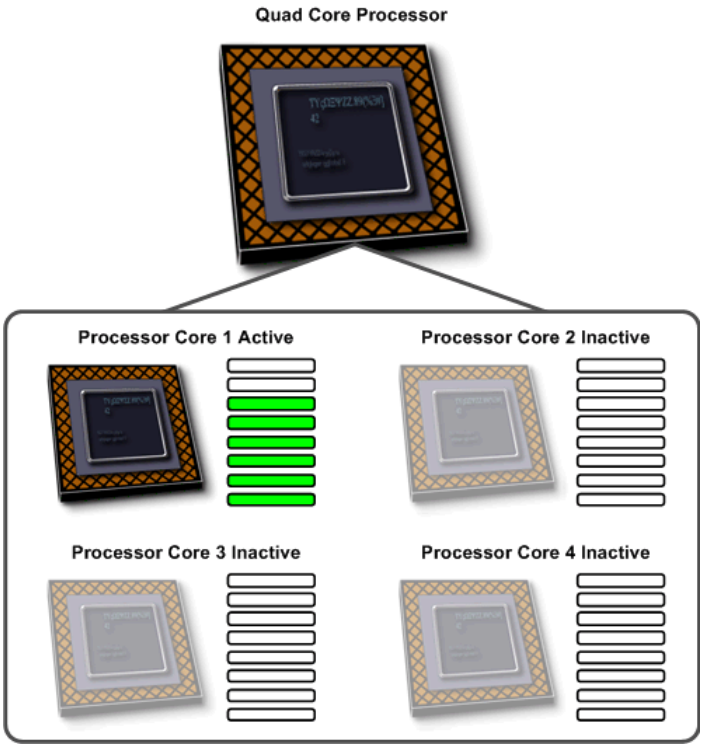


Abbildung 8: Core Parking

3.13. Windows XP Modus

Diese weitere Neuerung wurde mit dem Release Candidate 1 von Windows 7 präsentiert. Es ist möglich Programme, die nur auf Windows XP ausgeführt werden können und nicht auf Windows 7, in einer speziellen virtuellen Umgebung zu installieren und mit Windows 7 zu benutzen. Näheres hierzu finden Sie in Kapitel 5.6.

3.14. Die Windows 7 Produktpalette

Windows 7 Starter

- Weltweit nur als OEM-Version für neue PCs erhältlich
- Maximal drei gleichzeitig ausgeführte Programme
- Maximal 8 GB Arbeitsspeicher

Windows 7 Home Basic

- Nur in aufstrebenden Märkten/Schwellenländern erhältlich
- Maximal 8 GB Arbeitsspeicher

Windows 7 Home Premium

- Weltweit als OEM- oder reguläre Verkaufs-Version erhältlich
- "Premium"-Spiele
- Maximal 16 GB Arbeitsspeicher

Windows 7 Professional

- Weltweit als OEM- oder reguläre Verkaufs-Version erhältlich
- Maximal 192 GB Arbeitsspeicher

Windows 7 Enterprise

- Nur als Volumen-Lizenz erhältlich
- Maximal 192 GB Arbeitsspeicher

Windows 7 Ultimate

- Nur in limitierter Anzahl als OEM-Version erhältlich
- Maximal 192 GB Arbeitsspeicher

	Starter	Home Basic	Home Premium	Professional	Enterprise	Ultimate
Home Group	*	*	*	*	*	*
Keine Beschränkung der Anwendungszahl		*	*	*	*	*
Mobility Center		*	*	*	*	*
Erweiterte Netzwerkfunktionen		*	*	*	*	*
Aero Oberfläche			*	*	*	*
Media Center			*	*	*	*
Multi-Touch			*	*	*	*
Verschlüsseltes Dateisystem (EFS)				*	*	*
Domain Join				*	*	*
Netzwerk abhängiges Drucken (Location-aware Printing)				*	*	*
Entfernter Desktop				*	*	*
BitLocker					*	*
BitLocker to Go					*	*
AppLocker					*	*
Direct Access					*	*
BranchCache					*	*
Unterstützung mehrerer Sprachen gleichzeitig					*	*
Boot von virtueller Festplatte (VHD)					*	*

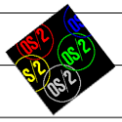
Abbildung 9: Funktionen der Windows 7 Versionen im Überblick

3.15. Die öffentlichen Windows 7 Vorabversionen

	Windows 7 Beta	Windows 7 Release Candidate
Build:	7000	7100
Image-Größe (32 Bit):	2,50 GB	2,48 GB
Image-Größe (64 Bit):	3,12 GB	3,18 GB
Release:	09. Januar 2009	05. Mai 2009
Gültig bis:	01. August 2009	01. Juni 2010
Geplantes Kontingent:	2.500.000 Downloads	Unbegrenzt

Abbildung 10: Windows 7 Vorabversionen

4. Die Zeit vor den NT-basierten Betriebssystemen

	DOS-Linie:	DOS-basiert:	spätere NT-Line:
			Microsoft XENIX 1.0
1981	PC-DOS 1.0		
1982	PC-DOS 1.1 / MS-DOS 1.25		Microsoft XENIX 2.0
1983	PC-DOS 2.0 / MS-DOS 2.11		
1984	MS-DOS 3.0 / 3.1		Microsoft XENIX 3.0
1985		Windows 1.0 / 1.01 / 1.02 / 1.03 / 1.04	Microsoft XENIX 4.0
1986			Microsoft XENIX 5.0
1987	MS-DOS 3.3	Windows 2.0 / 2.1 / 2.11	OS/2 1.0
1988	MS-DOS 4.0 / 4.01		OS/2 1.1
1989			 OS/2 1.2
1990		Windows 3.0 	
1991	MS-DOS 5.0		OS/2 1.3
1992		Windows 3.1	
1993	MS-DOS 6.0..6.22	Windows für Workgroups 3.11	Windows NT 3.1 
1994			
1995		Windows 95 (4.0)	Windows NT 3.5
1996			Windows NT 3.51
1997			Windows NT 4.0
1998		Windows 98 (4.10)	
1999			Windows 2000 (NT 5.0)
2000		Windows Me (4.90)	
2001			Windows XP (NT 5.1)
2002			
2003			
2004			
2005			
2006			
2007			Windows Vista (NT 6.0)
2008			
2009			
2010			Windows 7 (NT 6.1)

Zukunft:

Microsoft Singularity...

Microsoft Midori...

4.1. Die Anfänge: Interpreter, Compiler für CP/M

Die Firma "Micro-Soft“ (später Microsoft) beschäftigte sich nach ihrer Gründung im Jahr 1975, in ihren Anfangsjahren, mit der Programmierung eines Emulators für den Intel Prozessor 8080 und mit der Entwicklung eines BASIC-Interpreters "Microsoft BASIC". Dieser Interpreter wurde von diversen Firmen lizenziert und sorge für eine großflächige Verbreitung der Sprache "BASIC". Die Verbreitung von BASIC war so groß, dass praktisch jedem damals verkauften Computersystem ein BASIC zur Verfügung stand. Unter anderem entwickelte Mico-Soft auch Compiler für Fortran und COBOL. Viele der Interpreter wurden zunächst für das Betriebssystem CP/M entwickelt, welches von der Firma Digital Research Inc. damals entwickelt wurde.

Microsoft produzierte sogar eine Hardware, die „Microsoft Softcard“, eine Erweiterungskarte mit einem Z80-Prozessor für den damals sehr erfolgreichen Apple-II-Computer, welche es ermöglichte, die für CP/M geschriebene Software von Microsoft auf den Apple-Computern laufen zu lassen. Die Karte war ein großer Erfolg und übertraf die Programmiersprachen an Bedeutung. Es zeigte sich nun, dass fertige Anwendungsprogramme mehr gefragt waren als Programmiersprachen. Aufgrund dieser Erkenntnis begann Microsoft, den fast ausschließlichen Fokus auf Programmiersprachen endgültig aufzugeben.

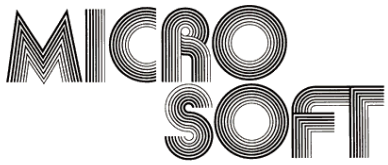


Abbildung 11: Altes Microsoft Logo

4.2. Die erste Betriebssystementwicklung: Microsoft XENIX

Im Jahr 1980 veröffentlichte Microsoft sein erstes Betriebssystem. Es handelte sich um ein System welches als "Microsoft XENIX" vermarktet wurde und auf einer UNIX-Architektur aufbaute. Viele der UNIX-Kernbestandteile hat Microsoft von AT&T lizenziert und weiterentwickelt bzw. neu implementiert. Der reine monolithische Kernel wurde auf die Intel-8086-Rechnerarchitektur portiert. Mehrere Versionen wurden von diesem System veröffentlicht und jeweils an damalige OEM-Hersteller verkauft. Diese nahmen teils eigene Anpassungen am System vor und veröffentlichten teils eigene Versionen des Betriebssystems XENIX. Daher gibt es insgesamt zahlreiche verschiedene Versionen von diesem Betriebssystem.
Microsoft hatte ursprünglich vor, die ersten Versionen von Microsoft XENIX als Standard-Betriebssystem für den damals bevorstehenden IBM PC zu etablieren. Daraus wurde jedoch nichts, da das Betriebssystem aufgrund der UNIX-Architektur zwingend eine Festplatte und 256 kByte Arbeitsspeicher benötigte und die ersten IBM PCs ab 1981 aber nur mit maximal 64 kByte Arbeitsspeicher ausgeliefert wurden.
Als Microsoft im Jahr 1987 mit der IBM die Entwicklung des modernen OS/2-Betriebssysteme startete, verlor Microsoft das Interesse an XENIX. Das System wurde von einem der OEM-Anbieter nämlich SCO weiterentwickelt und schließlich bis 1991 noch unter dem Namen "XENIX" verkauft.
Aus heutiger Sicht ist interessant, dass Microsoft auf seinem Campus noch Mitte der 90er Jahre XENIX basierte Rechner produktiv eingesetzt hat. Bilder zu MS XENIX befinden sich unter dem folgenden Link: http://und0cumented.ntinternals.net/tmp/ww/msxenix/pics/ .

Jahr	Version	Beschreibung	Sprache	Architektur	Plattform
1980	1.0	Sollte ursprünglich das PC-Betriebssystem werden.	Assembler und C	16 Bit Monolithisch	8086
1984	3.0 V/286			16 Bit Monolithisch	80286
1987	V/386			32 Bit Monolithisch	80386

Abbildung 12: MS XENIX Versionen

Abbildung 13: Startbildschirm von MS XENIX 3.0

4.3. Der geniale Deal mit der IBM: PC-DOS bzw. MS-DOS

Der Konzern IBM benötigte für seinen verspäteten Einstieg ins Homecomputer-Geschäft schleunigst ein geeignetes ressourcenarmes Betriebssystem. IBM schloss mit Microsoft einen Vertrag über 186.000 Dollar ab, in dem Microsoft sich verpflichtete, ein geeignetes Betriebssystem für den neuen IBM PC zu liefern. Dieser Vertrag legte sicher den Grundstein für den Erfolg von Microsoft bei der Betriebssystementwicklung.
Da das Microsoft eigene XENIX aufgrund der hohen Hardwareanforderungen nicht für die neuen IBM-PCs geeignet war, kauften sie zwei Tage nach dem IBM-Vertragsabschluss, das Betriebssystem QDOS (quick and dirty operating system) von Seattle Computer Products. Microsoft sah darin den Vorteil, das QDOS stark an CP/M angelehnt war und Microsoft mit CP/M reichlich Erfahrung gesammelt hatte. Für 50.000 Dollar wurden QDOS und dessen Programmierer Tim Paterson übernommen. Unter dem Codename „Project Chess“ modifizierten Gates, Allen und Paterson QDOS in ein monolithisches 16-Bit-Betriebssystem, das in der Lage war CP/M-Anwendungen auszuführen. **4.000 Codezeilen** umfasste das fertige MS-DOS 1.0 das 1981 an die IBM übergeben wurde und auf den ersten IBM PCs als PC-DOS veröffentlicht wurde. Das Betriebssystem hatte Erfolg, da existierende Anwendungen für CP/M auf PC-DOS liefen und somit ein Umstieg problemlos möglich war. Die Abwärtskompatibilität von DOS sorgte schließlich für eine rasche Verbreitung und viel Erfolg.
Die Version 2.0 wurde in weiten Teilen von Microsoft und IBM neu geschrieben. Es war nun möglich Unterverzeichnisse anzulegen und weitere Gerätetreiber manuell zu laden. Festplatten-Support wurde ebenfalls eingebaut. Im Jahre 1983 ist die Zusammenarbeit von Microsoft mit der IBM beendet und Microsoft führt die Entwicklung der MS-DOS Reihe eigenständig fort. Auch die IBM entwickelt PC-DOS eigenständig fort.

Mit der Version 3.0 werden Partitionen auf der Festplatte unterstützt, mit der Version 3.1 Netzwerksupport, Speichernutzung oberhalb 640 KB. Mit Version 5.0 wird der EMS-Speicher eingeführt und die DOS-Shell, eine windowsartige GUI um mit Dateien und Ordnern zu navigieren. Die letzten vollwertigen MS-DOS Versionen sind die Versionen 6.0, 6.10, 6.21 und schließlich 6.22. In diese Versionen wurde ein Virenschutz, eine Defragmentierung-Software, eine Unterstützung von CD-ROM-Laufwerken und die Möglichkeit der Komprimierung von Dateien integriert. Speziell für Notebooks gibt es PCMCIA-Unterstützung und des Weiteren noch Streamer-Back-up Software. Datenträgerfehler werden mit dem Diagnoseprogramm Scandisk behoben.

Jahr	Version	Beschreibung	Sprache	Architektur	Plattform	Größe
1981	1.0	IBM PC-DOS	Nur Assembler	16 Bit Monolithisch	x86	160 KB
1982	1.1	IBM PC-DOS				360 KB
1982	1.25	MS-DOS				
1983	2.0	IBM PC-DOS				<2 MB
1983	2.11	MS-DOS				

Abbildung 14: Versionen von DOS in Zusammenarbeit mit der IBM

Jahr	Version	Beschreibung	Sprache	Architektur	Plattform	Größe
1984	3.0	MS-DOS	Nur Assembler	16 Bit Monolithisch	x86	Ca. 3 MB
1985	3.1					
1987	3.3					
1988	4.0					Ca. 5 MB
1991	5.0					Ca. 6 MB
1993	6.0-6.22					Ca. 8 MB

Abbildung 15: MS-DOS Versionen ohne die Zusammenarbeit mit der IBM

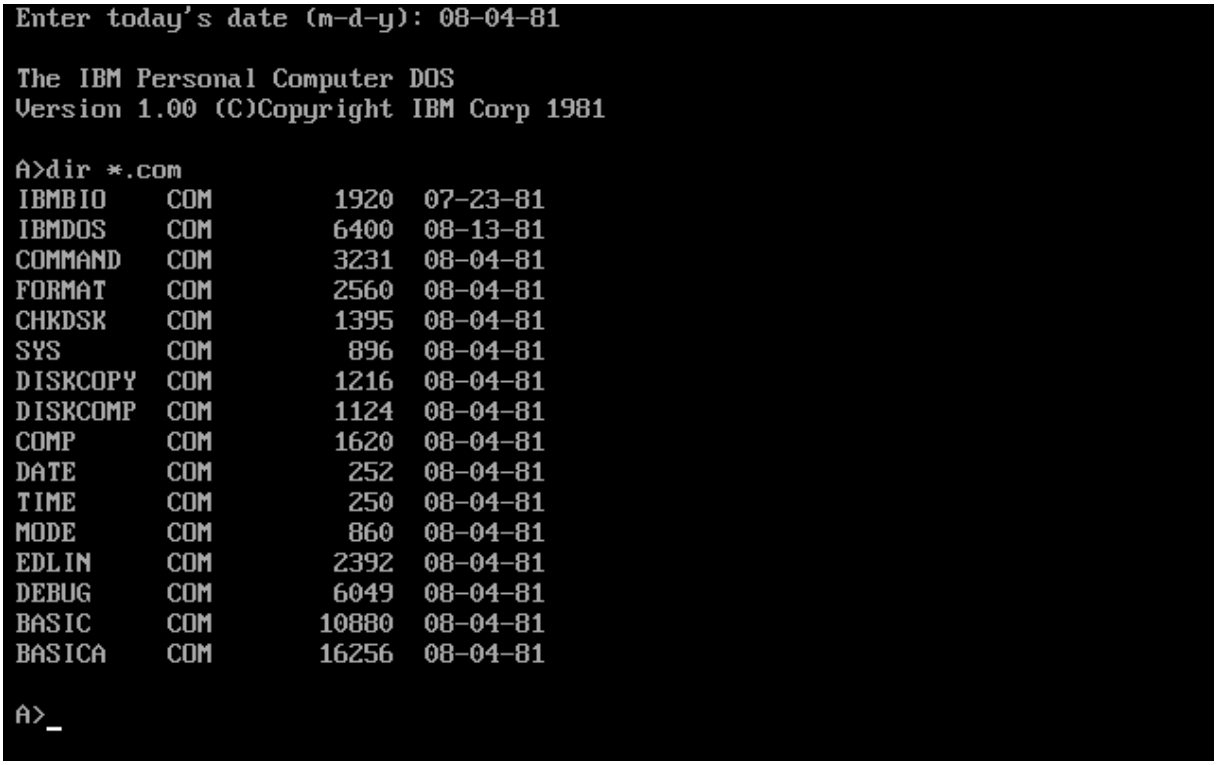


Abbildung 16: PC-DOS 1.00 auf dem ersten PC

4.4. Einführung der GUI: MS Interface Manager

Nach der Übergabe von MS-DOS an IBM im Jahr 1981 begannen bei Microsoft noch im selben Jahr die Arbeiten am Projekt "Interface Manager", der eine Zwischenschicht zwischen Betriebssystem und Anwendung darstellen sollte.
Die Zwischenschicht, ist eine grafische Oberfläche, die es dem Anwender erleichtert mit dem Computer umzugehen. Inspiriert von Xerox Star (1981) und Apple Lisa (1983) wurde der Interface Manager mit Fenstern, Pull-down-Menüs, Dialogboxen, Buttons und Mausbedienung realisiert.
Im Jahr 1983 veröffentlichte die Firma VisiCorp die erste grafische Oberfläche für MS-DOS bzw. PC-DOS. Des Weiteren wurde bekannt, dass IBM mit TopView ebenfalls daran war, eine grafische Oberfläche für MS-DOS zu entwickeln. Microsoft stand unter dem Zwang ebenfalls ein Produkt auf dem Markt zu etablieren, das eine grafische Oberfläche bereitstellt. Sie versprachen daraufhin der Öffentlichkeit eine Microsoft eigene GUI die unter dem Namen "Windows" ab Ende 1984 veröffentlicht werden soll. Leider konnte Microsoft diesen Termin nicht einhalten und musste die Produkteinführung ständig verschieben, was bei der Öffentlichkeit damals für Empörung sorgte.

4.5. Das erste Windows: Windows 1.0

Im Jahr 1985 war es endlich soweit, dass Microsoft seine GUI für MS-DOS "Windows 1.0" veröffentlichen konnte.
Nach **3 Jahren Entwicklungszeit** stellten die **24 Entwickler** die erste Version fertig, die in der Öffentlichkeit jedoch keine großen Erfolge feierte.
Die GUI setzte auf dem monolithischen MS-DOS-Unterbau auf und verwaltete nur Teile der Hardware in Form von Treibern selbstständig (Bsp. Grafikkarte). Der hardwarenahe Code (ca. 15 %) für die Intel-8086-Rechnerarchitektur wurde stets in **Assembler** geschrieben. Die GUI und deren Anwendungen (Paint, Write, Rechner, Kalender, Kartenmanager, Uhr) wurden in **C** geschrieben, was in späteren Versionen von Windows durch Pascal-Code ersetzt bzw. ergänzt wurde. Windows 1.0 speicherte alle Einstellungen und Anpassungen in INI-Dateien.
Das Problem war, dass Windows 1.0 auf damaliger Hardware viel zu träge war und zusätzlich viel Speicherplatz benötigte und Konkurrenzprodukte zudem teilweise günstiger waren. Geeignete Hardware war damals einfach noch nicht geboren.

Jahr	Version	Sprache	Architektur	Plattform
1985	1.0	85 % C-Code 15 % Assembler	16 Bit MS-DOS-Unterbau	x86
.	1.01			
.	1.02			
.	1.03			
1987	1.04			

Abbildung 17: Windows 1.0 Versionen

Im Jahr 1987 wurde Windows 2.0 veröffentlicht. Windows 2.03 war das letzte Windows, das sich auf Disketten installieren und ohne Festplatte verwenden ließ. Neuerungen waren bessere Treiberunterstützung und der nutzbare 64 KB Erweiterungsspeicher. Mit Windows 2.11 gab es Verbesserungen bei Speicherzugriffen durch XMS und EMS.

Jahr	Version	Sprache	Architektur	Plattform
1987	2.0x	85 % C-Code 15 % Assembler	16 Bit MS-DOS-Unterbau	x86
.	2.1			
.	2.11			

Abbildung 18: Windows 2.0 Versionen

4.6. Windows wird ein Erfolg: Windows 3.0

Mit Windows 3.0 (1990) wurde der Programm Manager und Datei Manager eingeführt. Die Dateiverwaltung konnte nun komplett über Windows erfolgen. DOS-Kenntnisse waren nicht mehr nötig um Bürotätigkeiten durchführen zu können. Die Zusammenarbeit mit Intel wurde stark ausgebaut, was zur Folge hat, dass verschiedene Intel-Prozessoren unterstützt werden. Windows 3.0 kommt mit den Intel Prozessoren 8088, 80286, 80386 zurecht und kann in einem Modus von insgesamt 3 betrieben werden: Dem Real Mode 8086 oder dem Standard Mode 80286 oder dem Extended Mode 80386. Windows 3.0 beherrscht nur kooperatives Multitasking. Die Größe einer Installation auf der Festplatte ist auf 30 MB angestiegen, was ein Rechner von 1990 mit Standardfestplatte von 100 MB aufnehmen kann. Des Weiteren wird VGA unterstützt was die erstmalige Darstellung von Multimediainhalten in vielen Farben ermöglichte und zusammen mit der Microsoft Software Multimedia Extensions 1.0 ein Multimedia fähiges System aufbaute.
Zusätzlich wurde die Registrierungsdatenbank (Registry)[7] eingeführt, die bis heute im aktuellen Windows 7 erhalten geblieben ist. Diese speichert Einstellungen und Anpassungen in einer zentralen Datenbank anstatt in mehreren INI-Dateien. Allerdings wurde die Registrierungsdatenbank in Windows 3.0 ausschließlich für die Erkennung von Dateierweiterungen genutzt. Einstellungen werden weiterhin in INI-Dateien abgespeichert. Erst ab Windows NT bzw. Windows 95 wird die Registrierungsdatenbank als die zentrale Einstellungsdatenbank genutzt.

Ab 1992 kam der Nachfolger **Windows 3.1** der zum Verkaufsschlager im Consumermarkt wurde, obwohl die neue Version nicht besonders abwärts kompatibel war. Mit Windows 3.1 hat Microsoft in größeren Teilen einen Neuanfang gewagt. Microsoft hat den bisherigen C-Code durch Pascal ersetzt, sodass Windows 3.1 aus Assembler und Pascal-Code besteht.
Geblieben ist die bekannte Windows 3.0-Optik, der MS-DOS-Unterbau die unterstützte Registerwortbreite von 16-Bit sowie die Unterstützung des kooperativen Multitasking. Neu dazugekommen ist SVGA-Support, TrueType, OLE, Drag-and-Drop und ein optional erhältliche Win32s-API[8] ein 32-Bit-Aufsatz. Diese Win32s-API (WOW32: Windows on Windows 32) kann nachinstalliert werden und ermöglicht es, Windows NT Programme auf Windows 3.1 auszuführen. Auch einige spätere Windows 95 Programme können auf dieser API laufen. Daher ist Windows 3.1 in Teilen bereits 32-Bit fähig. So kann z. B. der Internet Explorer 5.0 der während der Windows 98-Ära entwickelt wurde unter Windows 3.11 ausgeführt werden, wobei eine spezielle Installationsroutine für Windows 3.1 genutzt werden muss.

[7] Siehe Kapitel 6.3
[8] Application Programming Interface

Jahr	Version	Sprache	Architektur	Plattform	Größe
1990	3.0	85 % C-Code 15 % Assembler	16 Bit MS-DOS-Unterbau kooperatives Multitasking	x86	Ca. 30 MB
1992	3.1	85 % Pascal 15 % Assembler	16 Bit MS-DOS-Unterbau kooperatives Multitasking Win32s-API für 32-Bit	x86	> 30 MB
1993	3.11				> 35 MB

Abbildung 19: Windows 3.x Versionen

Ab 1993 wurde ein um Netzwerkfunktionen erweitertes Windows 3.1 veröffentlicht: **Windows for Workgroups 3.11**
Diese Version von Windows enthält einen vollwertigen, 32-Bit tauglichen Netzwerkstack der zusätzlich zum Microsoft eigenen NetBEUI auch das durch die UNIX-Welt vorangetriebene TCP/IP unterstützt. Netzwerk-, Modem-, ISDN- und DSL-Verbindungen sind mit Windows 3.11 möglich. Windows 3.11 ermöglicht den Zugriff auf LAN Manager Server und Windows NT Server Ressourcen und kann somit als Windows-Netzwerk-Client eingesetzt werden. Zusätzlich bietet es eine Peer-to-Peer-Funktionalität, die es ermöglicht Freigaben auf einem Client für einen anderen bereitzustellen. Freigaben können Drucker oder Laufwerke sein. Es war damit also möglich Drucker oder Laufwerke gemeinsam von mehreren Rechnern zu verwenden, ohne einen zentralen Server zu haben. Die Netzwerk Peer-to-Peer Funktionalität hatte einen großen Erfolg und ließ Windows for Workgroups zu einem weiteren Verkaufsschlager bei mittelständischen Unternehmen und auch im Consumermarkt werden.

Windows 3.1 und Windows for Workgroups 3.11 sind die letzten ressourcenschonenden Windows Versionen die mit wenig Hintergrundprozessen bzw. Diensten und mit geringem Arbeitsspeicher auskommen.

4.7. Eine neue Betriebssystemgeneration

Gegen Mitte der 80er Jahre wurde immer offensichtlicher, dass eine fundamentale Neuentwicklung eines GUI basierten Betriebssystems stattfinden musste. Gerade im Geschäftsumfeld wurde ein Betriebssystem benötigt, welches nicht auf Großrechnern verwendet werden sollte, sondern auf normalen PCs laufen sollte und eine höhere Stabilität als das bisherige Modell: MS-DOS + GUI aufweisen sollte. Es sollte für Serverdienste geeignet sein, die neueste Hardware besser ausnutzen und eine ernsthafte Alternative zu Novell Netware werden. Ein abstürzendes Programm sollte nicht den ganzen Rechner lahmlegen und zusätzlich sollte ein effizienteres Multitasking möglich sein, denn mit der Windows GUI oder anderen GUI Erweiterungen und dem MS-DOS-Unterbau war kein preemptives Multitasking möglich.
Microsoft und IBM wurde klar, dass die 2. PC-Generation aufgrund neuer Anforderungen ein neues Betriebssystem benötigte, welches nicht auf dem bisherigen monolithischen MS-DOS-Kern basierte, sondern eine komplett neue Architektur erforderte.

4.8. Der MS-DOS Nachfolger: OS/2

Ähnlich wie im Jahr 1981 kam es zu einer erneuten Kooperation zwischen Microsoft und IBM. Ein neues Betriebssystem mit den Namen OS/2 wurde entwickelt. Der Name OS/2 nimmt Bezug auf die damals neue Computerreihe Personal System/2 (PS/2). Chefentwickler auf Seiten von Microsoft wurde Gordon Letwin, der unter anderem das HPFS-Dateisystem entwickelte.

Beide Firmen entwickelten zusammen, die Vermarktung erfolgte jedoch getrennt. Microsoft verkaufte Microsoft OS/2 und IBM verkaufte IBM OS/2 in zwei verschiedenen Versionen (Standard und Extended). Insgesamt gab es also 3 Pakete zu kaufen. Die Installationsdisketten enthielten das Basisbetriebssystem, den Presentation Manager (GUI-System), Schriftarten und Treiber.
Die Netzwerkunterstützung musste man separat nachinstallieren. Es gab eine Client-Netzwerkunterstützung (LAN Manager Client) oder eine Server-Netzwerkunterstützung (LAN Manager) zum Nachinstallieren.

Jahr	Version	Architektur	Plattform	Größe
1987	1.0	16-Bit preemptives Multitasking	x86	Ca. 10 MB
1988	1.1			
1989	1.2			
1991	1.3			

Abbildung 20: OS/2 Versionen von Microsoft und IBM

Die OS/2 1.3 Architektur erfüllte folgende Anforderungen:

- DOS-Emulator: "Familienübergreifende" Anwendungen
- Preemptives Multitasking
- Komplett geschützter Speicher
- Virtueller Speicher (theoretisch 1 GB)
- Multithreading (Max. 512 Threads)
- Keine Speicherbegrenzungen wie in MS-DOS (640 K)
- Grafische Windows-ähnliche GUI (Presentation Manager)
- Neues Dateisystem HPFS, kein FAT als Standard
- Verwendung mit einer Intel 80286-CPU
- Windows 3.0 Optik
- Sehr gute Rückwärtskompatibilität bei zukünftigen Versionen

Die letzte Anforderung (Verwendung mit einer Intel 80286-CPU) brachte jedoch folgende fatale Einschränkungen mit sich:

- Maximale Nutzung von 16 MB Arbeitsspeicher
- Reine 16-Bit-Architektur

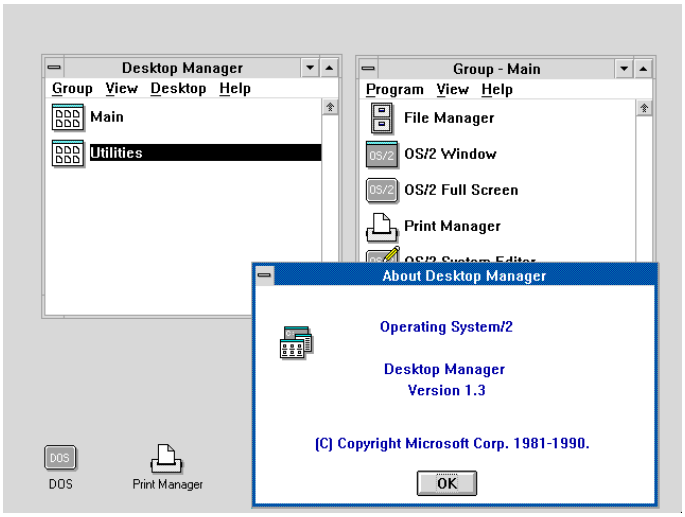


Abbildung 21: Die Optik von OS/2 1.3 ist angelehnt an die von Windows 3.0

Das OS/2-GUI-System wurde neu entwickelt und trägt den Namen "Presentation Manager". Die optische Darstellung wurde ähnlich gestaltet, wie die des "MS Interface Managers", dem Microsoft eigenen GUI-System, das in Windows verwendet wird. Daher entspricht die Optik von OS/2 der Optik von Windows 3.0 und dessen "Programm Manager" mit Programmgruppen und Systemsteuerung. Auch die Bedienung wurde an Windows angelehnt. Der einzige Unterschied bestand darin, dass der "Programm Manager" in OS/2 als "Desktop Manager" bezeichnet wurde. Ebenso wie den "Programm Manager" mit Programmgruppen und Systemsteuerung gibt es den aus Windows bekannten "Datei Manager" in OS/2. Im Basisumfang beiliegende Zubehör Programme gibt es dafür keine.

Die Anzahl an verfügbaren GUI Anwendungen für OS/2 hielt sich nach dessen Einführung stark in Grenzen was zur Folge hatte, dass nicht, wie vor allem von IBM erhofft, sehr viele professionelle Anwender (Poweruser) auf OS/2 umstiegen. Die Verbreitung von OS/2 hielt sich daher stark in Grenzen und das System hatte eigentlich nur im Serverumfeld einen Erfolg erzielen können, da es die damalige Hardware voll ausnutzen konnte. Für Entwickler war OS/2 ebenfalls gut geeignet, da "Familienübergreifende" Anwendungen erstellt werden konnten. Das heißt, man konnte unter OS/2 Anwendungen für MS-DOS programmieren und diese auch unter OS/2 ausführen (Kreuzkompilationen).

Mit der Version 1.2 wurden schließlich alle geplanten Eigenschaften in OS/2 implementiert. Auch das neue Dateisystem HPFS wurde eingeführt, das erstmals lange Dateinamen unterstützte (bis zu 255 Zeichen), erweiterte Attribute, doppelte Links zwischen den Datenstrukturen für Redundanz und Festplatten von theoretisch 64 GB. Eine weitere optionale Version HPFS386 hatte ACLs integriert und erlaubte somit das Setzen von detaillierten Berechtigungen.

1988 begannen bereits die Arbeiten für **OS/2 NT Version 2.0**, für das Microsoft einen neuen 32-Bit-Kernel und die Subsysteme entwickeln sollte. Das System sollte OS/2 kompatibel sein, eine hohe Zuverlässigkeit und Sicherheit aufweisen, POSIX-fähig und eine Multiprozessorfähigkeit aufweisen. Nach dem Erscheinen der OS/2 Version 1.3 im Jahr 1991 hatte Microsoft mit Windows 3.1 bereits im Jahr 1990 so ein großer Erfolg erlangt, dass sie statt **OS/2 NT Version 2.0** den Namen **Windows NT** ab der nächsten Version durchsetzen wollten. Nun an verfolgten sie eine direkte Systemkompatibilität mit Windows 3.x und die OS/2-Kompatibilität wurde in ein Teilsystem ausgelagert. Dadurch kam es zum Streit zwischen Microsoft und IBM und schließlich zur endgültigen Trennung im Jahr 1991. Zur damaligen Zeit war OS/2 das fortschrittlichste Betriebssystem auf dem Markt. IBM entwickelte das Betriebssystem alleine weiter und die nächste Version erschien als OS/2 Warp 3 im Jahr 1994. Weder Windows noch MacOS unterstützten preemptives Multitasking, Multithreading oder virtuellen Speicher. Nachteile waren die mangelnde Treiberunterstützung für Geräte aller Art und hohe Anforderungen an die Hardware. Das und die Tatsache, dass IBM ein schlechtes Marketing betrieb hatte zur Folge, dass sich OS/2 jedoch nie gegen MS-DOS + Windows durchsetzen konnte. Die meisten Anwender setzten weiterhin auf die DOS-basierenden Windows 3.x Nachfolger (**Windows 95, Windows 98**). Des Weiteren gilt das passive Verhalten Microsofts als weiterer Grund für das Scheitern von OS/2 im Massenmarkt. Microsoft selbst konnte erst mit Windows NT im Jahr 1994 ein Betriebssystem anbieten, das aus technischer Sicht das Niveau von OS/2 erreichen konnte. Weitere geschichtliche Informationen zu OS/2 findet man unter http://toastytech.com/guis/MOS2.html und unter http://www.operating-system.org/betriebssystem/_german/bs-os2.htm .

4.9. Weitere DOS-basierte Windows Versionen: Windows 9x und Me

Mit den Versionen Windows 95, Windows 98 und Windows ME brachte Microsoft weitere DOS-basierte Betriebssysteme auf den Markt. Diese wurden in Teilen mit 32-Bit-Komponenten ausgestattet und lösten die aus der Windows 3.x Ära stammende 16-Bit-Architektur in großen Teilen ab.

Aus technischer Sicht sind die Betriebssysteme eher unspektakulär, da sie im Vergleich zu MS-DOS mit Windows 3.x nicht wirklich viele Neuerungen im Bereich Kernel und Subsysteme boten.

Bilder und weitere interessante Fakten zu früheren Windows Betriebssystemen kann man auf der Seite http://www.winhistory.de finden.

5. Die Windows 7-Systemarchitektur

5.1. Microsoft geht seinen eigenen Weg: Windows NT

Nach der endgültigen Trennung von IBM begann Microsoft eigenständig ein neues, komplett eigenes Betriebssystem entwickeln. Jetzt konnte Microsoft das Betriebssystem nach eigenen Vorstellungen entwerfen. Die Entwicklungsabteilung konnte einiges an OS/2-Wissen und Ideen aus der bereits begonnenen **OS/2 NT Version 2.0** Entwicklung der vergangenen IBM-Zeit mitnehmen. Dennoch basiert das neue Windows NT-System (NT = New Technology) auf einer komplett neu entwickelten Architektur und unterscheidet sich von allen je erschienen OS/2-Betriebssystemen deutlich.

Microsoft konnte für die Entwicklung der Betriebssystemarchitektur David Neil Cutler, denjenigen Entwickler mit der bis dahin wohl größten Erfahrung beim Entwickeln von Betriebssystemen gewinnen. Cutler wurde mit seinem Entwicklerteam von der Firma Digital Equipment Corporation (DEC) bereits schon 1988, also noch während der OS/2-Zeit abgeworben. David Cutler hatte bereits als Chefentwickler mehrere Betriebssysteme entworfen, nämlich **RSX-11** eine Serie von Echtzeitbetriebssystemen, **VAXELN** und das Betriebssystem **VMS für VAX**, was damals ein fortschrittliches 32-Bit-Betriebssystem war, das Multiuser- und Multitasking-fähig war. Es war zudem eines der ersten Betriebssysteme mit virtueller Speicherverwaltung. Viele in diesen beiden Systemen umgesetzte Prinzipien und Konzepte wurden später von David Cutler in Windows NT fortgeführt. Ein weiterer Grund, wieso Bill Gates unbedingt David Cutler wollte, war die Tatsache, dass Cutler die UNIX-Architektur ablehnte und Assemblercode verabscheute. Das kam Microsoft wohl sehr entgegen, da Microsoft nicht auf UNIX aufbauen wollte, dieses Thema war nach dem Verkauf von MS XENIX endgültig vom Tisch. Ursprünglich sollte die Entwicklung gut zwei Jahre dauern, daraus wurden jedoch viereinhalb Jahre, bis die erste Version im Sommer 1993 freigegeben wurde.

5.2. Anforderungen und Designziele der Windows NT-Architektur

Die folgenden Anforderungen lagen der Spezifikation von Windows NT im Jahr 1989 zugrunde:

- **Portabilität:** An oberster Stelle der Anforderungen stand die Portabilität bzw. Prozessorunabhängigkeit. Das bedeutet, das neue Betriebssystem sollte auf verschiedenen Hardwareplattformen (CISC, RISC und Alpha) laufen und auch fähig sein in Zukunft verschiedene Registerwortbreiten und neue Prozessoren zu unterstützen. Aufgrund dieser wichtigen Anforderung war der Codename während der Entwicklung "**Portasys"**.

- **Zuverlässigkeit und Robustheit:** Die zweite wichtige Anforderung war eine sehr hohe Zuverlässigkeit zu erreichen, indem jeder Anwendungsprozess jeweils seinen eigenen privaten Adressraum erhalten soll. Das System soll sich selbst vor internen Fehlfunktionen als auch vor externen Eingriffen schützen können. Eine abstürzende Anwendung sollte nicht mehr das gesamte System zum Absturz bringen können. Anwendungen dürfen nicht in der Lage sein, das Betriebssystem oder andere Anwendungen zu beschädigen. Eine derartige Stabilität war unter Betriebssystemen wie VMS längst üblich. Programmen wurde es nun ausdrücklich unterbunden, direkt mit der Hardware zu kommunizieren. Alle Informationen laufen nun über die Betriebssystemschicht. Eine Ausnahme stellt die COM-basierte Programmierschnittstelle DirectX dar. Mit ihr ist es möglich, Teile der Grafikkarte direkt anzusprechen. Diese Schnittstelle wurde speziell für Spiele und Multimediaanwendungen implementiert, welche auf eine sehr hohe Leistung angewiesen sind.

- **Kernel:** Die neue Architektur sollte auf einem Kernel aufgebaut sein, der Eigenschaften eines Mikrokernels aufweist und im Gegensatz zu einem monolithischen Kernel nur grundlegende Funktionen (Speicher- und Prozessverwaltung, Prozesssynchronisation) bereitstellt. Alle weiteren Funktionen sind als eigene Serverprozesse ausgelagert. Diese wurden als Executive bezeichnet. Microsoft hat jedoch bei Windows NT 4.0, Windows 2000 und Windows XP aus Leistungsgründen das Grafiksystem in den Kernel versetzt, sodass diese Versionen laut Professor Andrew Tanenbaum über keinen richtigen Mikrokernel verfügen. Sie enthalten einen Hybridkernel. In Windows Vista und Windows 7 wurde der ursprüngliche Zustand wie bei Windows NT 3.1 und 3.5 wiederhergestellt. Dies konnte aufgrund moderner leistungsfähiger Hardware und einem neuen Grafikkartentreibermodell durchgeführt werden.

- **Erweiterbarkeit durch einen modularen Aufbau:** Das Subsystem soll aus einzelnen Modulen bestehen. Einzelne Systemfunktionen sollten aus einem Modul oder mehreren Modulen aufgebaut sein. Somit lassen sich neue Funktionen leichter integrieren und veraltete oder fehlerhafte Module können einfacher ausgetauscht werden. Die Module sollen in einen geschichteten Aufbau integriert werden.

- **Kein Assembler aber C und C++:** Im Gegensatz zu MS-DOS sollte kein Assembler bei der Programmierung verwendet werden. Das lässt sich aber wahrscheinlich bei keinem Betriebssystem ganz umgehen, daher musste der hardwarenahe Bestandteil, die Windows HAL in Assembler geschrieben werden. Die Subsysteme und die Executive wurden in C programmiert. Die überwiegenden Teile der grafischen Oberfläche sollten mit C++ programmiert werden, wurden aber aus Geschwindigkeitsgründen schließlich doch in C programmiert. In Windows 7 wurden Teile des WDDM-Grafikksystems ebenfalls in Assembler geschrieben. Der direkte Maschinencode ermöglicht eine bessere Geschwindigkeit auf dem Zielsystem.

- **Preemptives Multitasking und Wiedereintrittsfähigkeit:** Das Betriebssystem sollte mehrere Programme bzw. Prozesse scheinbar gleichzeitig bearbeiten können. Das unter den MS-DOS basierten Windows Versionen verwendete kooperative Multitasking hatte oft die Folge, dass sich das System festfährt, aufgrund eines hängenden Prozesses, der die CPU nicht mehr freigibt. Teilweise kamen erst nach Abarbeitung eines Prozesses andere Prozesse wieder zum Zug.
  Die neue NT-Architektur erhielt jedoch ein preemptives Multitasking. Dabei steuert der Betriebssystemkern die Abarbeitung der einzelnen Windows-Threads. Jeder Windows-Thread kann nach einer bestimmten Abarbeitungszeit zugunsten anderer Threads schlafen gelegt werden und später wieder geweckt werden. Zusätzlich verfügen die Threads über Prioritäten, die eine intelligentere CPU-Ausnutzung ermöglichen.

- **Symmetrische Multiprozessorunterstützung:** Die Consumer Versionen Windows 9x unterstützten seither keine Multiprozessorsysteme.

- **Reines 32-Bit-System:** Ab 1992 dominierten Intel 80486 Prozessoren den PC-Markt. Wie bereits sein Vorgänger (Intel 80386-DX Prozessor), unterstützte dieser vollwertige Registerwortbreiten von 32-Bit, sodass Windows NT als reines 32-Bit-System entwickelt wurde. Es enthält nur 32-Bit-Code.

- **16-Bit-Emulation für Legacy[9]-Anwendungen:** Notwendig sah Microsoft die Integration eines emulierten DOS-Subsystems, welches es ermöglichte einfachere, bestehende DOS-Anwendungen unter Windows NT auszuführen. Die direkten Hardwarezugriffe der DOS-Programme mussten virtuell über Betriebssystemfunktionen umgeleitet werden, da Windows NT keinen direkten Zugriff auf die Hardware erlaubte. Die Ausführung sollte im Fenstermodus möglich sein. In der Praxis hat sich gezeigt, dass eine Kompatibilität zu DOS-Anwendungen nur teilweise erreicht wurde.

- **Gewisse Anwendungskompatibilität durch verschiedene Subsysteme:** Windows NT sollte trotz der komplett neuen Technik, eine hohe Anwendungskompatibilität aufweisen. Es sollte zusätzlich zu neuen Windows NT Anwendungen auch Anwendungen ausführen können, welche für die Betriebssysteme Windows 3.x, UNIX (POSIX-kompatibel) und OS/2 geschrieben wurden. Die Subsysteme für UNIX und OS/2 wurden jedoch bei späteren Windows NT Versionen entfernt, da diese in der Praxis nicht wie gewünscht mit aktuellen UNIX bzw. OS/2 Anwendungen genutzt werden konnten. Schwierigkeiten bei der Anwendungskompatibilität zeigten sich jedoch bei diversen Windows 3.x Multimediaanwendungen, die einen direkten Zugriff auf Hardwarekomponenten erforderten, was unter Windows NT nicht mehr gestattet wurde. Infolgedessen war hier die einzige Möglichkeit auf Updates der Hersteller zu warten, bis sie ihre Programme an Windows NT angepasst hatten.

- **Sicherheitsmechanismen:** Mehre Benutzer können sich am System hintereinander anmelden und sie können verschiedenen Gruppen zugewiesen werden. Den Gruppen können Berechtigungen zugeteilt werden. Es gibt verschiedene Benutzerkontexte: Administratoren, Benutzer. Mit dem neuen Dateisystem konnten Sicherheitsrechte gesetzt werden. Die umfassenden ACLs (Access Control Lists) und ein gründlich durchdachtes Sicherheitssystem kamen jedoch erst mit Windows 2000.

- **Hohe Systemleistung:** Dies wurde zwar von Anfang an angestrebt, konnte aber bis zur Version 4.0 absolut nicht erreicht werden, was wohl der Hauptgrund für den ausgebliebenen Erfolg der ersten Versionen ist. Die auf MS-DOS basierten Windows Versionen benötigten weniger Hardwareressourcen und fühlten sich schneller an, was die Bedienung des Systems angeht. Erstmals mit Windows 4.0 und geeigneter Hardware konnte die damalige NT-Architektur endlich voll ausgenutzt werden.

- **Win32-API:** Dieses Interface ist aus funktionaler Sicht das Gegenstück, zu den Systemaufrufen unter UNIX-Betriebssystemen. Für den Anwendungsprogrammierer ist diese Schnittstelle von höchster Wichtigkeit. Das API[10] ist sehr umfangreich und sehr detailliert dokumentiert.

- **Unicode:** Die internen Zeichenfolgen werden als 16 Bit lange Unicode Zeichen gespeichert. Dies ermöglicht einen Einsatz von Windows im internationalen Markt.

[9] Veraltete Technik (Altlasten) in der Computertechnik
[10] Application Programming Interface

Während den Entwicklungsarbeiten an Windows NT 3.51 stellte sich ein verhängnisvoller Schwachpunkt der Windows NT-Architektur heraus: Windows NT war zwar ein Mehrbenutzersystem, das die Fähigkeit hat Arbeitsumgebungen für verschiedene Benutzer nacheinander bereitzustellen und voneinander abgrenzen zu können, eine gleichzeitige Ausführung von mehreren Benutzersitzungen war jedoch nicht möglich.

Microsoft löste dieses Problem mithilfe der Firma Citrix, die nach Einsicht in den Quellcode eine geeignete Technologie nämlich "Multiwin" an Microsoft lizenzierte. Microsoft integrierte diese mit der Bezeichnung Windows Terminal Server erstmalig in Windows NT 4.0 Terminal Server Edition.

5.3. Anforderungen und Designziele der Windows 7-Architektur

- **NT-Technologie**: Windows 7 basiert auf der gleichen Windows NT-Technologie wie die vorherigen Versionen. Es handelt sich um die bisher am eingehendsten getestete und die am stärksten optimierte Version der NT-Technologie, die bisher produziert worden ist.

- **Hohe Systemleistung:** Das im Jahr 2006 veröffentlichte Windows Vista stellt ähnlich wie die ersten NT-Versionen ebenfalls zu hohe Anforderungen an Hardware von 2006. Der sehr erfolgreiche Vorgänger Windows XP aus dem Jahr 2001 benötigt weniger Hardwareressourcen und fühlt sich schneller an. Der aktuelle Windows 7-Entwicklungsleiter Steven Sinofsky hat die Behebung der Leistungsprobleme, die sich in Vista bemerkbar machten, als das wichtigste Ziel der nächsten Windows Generation formuliert und dabei laut diversen Entwicklerinformationen massive Eingriffe in der Windows Speicherverwaltung vorgenommen, an die man sich in der Vergangenheit angeblich nicht ran getraut hatte. Die erste Betaversion von Windows 7 verspricht bereits wahre Wunder, da diese sogar Windows XP in Leistungstest deutlich schlägt und geringere Anforderungen an die Hardware stellt als Windows Vista.

- **Hohe Kompatibilität zu Windows Vista und Windows XP**

- **32-Bit und 64-Bit-Version**

5.4. Alle Windows NT-Versionen im Überblick

Windows NT 3.1

Jahr	Version	Bezeichnung	Editionen	Sprachen	Plattformen
Juli 1993	NT 3.1	Windows NT 3.1	- Workstation - Advanced Server	C Assembler (bisschen)	x86 (80386 und 80486) MIPS R4000

Die erste NT-Version, dessen Namen einen Bezug auf die optische Nähe zum damals vorherrschenden Windows 3.1 herstellt. Das System war noch nicht so ausgereift und stabil wie seine Nachfolger, jedoch umfangreicher als ursprünglich geplant. Die Voraussetzungen an die Hardware waren für damals zu hoch angesetzt (386DX, 12 MB RAM, 75 MB HDD). Die Software und Treiberunterstützung war schlecht. Enthält: **Windows 3.x Look-and-Feel, NTFS**, HPFS, FAT16, TCP/IP, unterstützt bis zu **4x CPU**'s, Domänennetzwerkmodell, usw.

Windows NT 3.51 (kurzeitig 3.5)

Jahr	Version	Bezeichnung	Editionen	Sprachen	Plattformen
September 1994 Mai 1995	NT 3.5 NT 3.51	Windows NT 3.5 Windows NT 3.51	- Workstation - Server	C Assembler (bisschen)	x86 MIPS-RISC Alpha-RISC IBM PowerPC (ab 3.5)

Extrem stabile und schnellere Version. Wurde oft als Entwickler- und CAD-Plattform genutzt. Diese enthält diverse Implementierungen, die mit Windows 3.1 nicht mehr fertiggestellt werden konnten. Die eigentliche Systemgröße wurde jedoch im Vergleich zu 3.1 reduziert, um eine höhere Systemgeschwindigkeit und Zuverlässigkeit zu erreichen. Unterstützt zusätzlich die NTFS-Datenkompression, OpenGL für CAD-Unterstützung, PCMCIA, Win32-API auf Windows 95 Stand, usw.

Windows NT 4.0

Jahr	Version	Bezeichnung	Editionen	Sprachen	Plattformen
Juli 1996	NT 4.0	Windows NT 4.0	- Workstation - Server - Terminal Server - Enterprise Edition - Small Business Server - Embedded	C Assembler (bisschen)	x86 IBM PowerPC MIPS-RISC Alpha-RISC

Erste NT-Version, mit welcher Microsoft große Erfolge erzielen konnte. Die Version 4.0 gilt als sehr robust und ausgereift. Enthält zusätzlich: **Windows 9x Look-and-Feel**, Kontextmenüs, DirectX, unterstützt bis zu **32x CPU**'s, usw.

Windows 2000

Jahr	Version	Bezeichnung	Editionen	Sprachen	Plattformen
Februar 2000	NT 5.0	Windows 2000	- Professional - Server - Advanced Server - Advanced Server Limited Edition - Datacenter Server - Datacenter Server Limited Edition - Windows Powered	C Assembler (bisschen) C++	x86 (IA-64)

Verspäteter Erfolg aber dafür ausgereifte Qualität. Neuer Rekord: Wurde von ca. 5000 Leuten auf die Beine gestellt. Enthält erstmalig: **Active Directory**, ACLs, Group Policy, Kerberos, DynDNS, NTFS hat erst jetzt den bereits 1993 geplanten Funktionsumfang: NTFS-Verschlüsselung und Kontingente, USB-Support, WDM-Treiber, Defrag, unterstützt bis zu **32x CPU**'s und 64 GB RAM, usw.

Windows XP und Windows 2003 Server

Jahr	Version	Bezeichnung	Editionen	Sprachen	Plattformen
August 2001 März 2003	NT 5.1 NT 5.2	Windows XP Windows 2003 Server	- Professional - Home Edition - Media Center - Tablet PC - Embedded - Starter Edition - Legacy PC - x64 Edition (ab April 05) - Standard Edition - Enterprise Edition - Datacenter Edition - Web Edition - Small Business	C Assembler (bisschen) C++	x86 AMD64/EM64T IA-64

Zunächst wurden zwei Versionen mit unterschiedlichem Funktionsumfang veröffentlicht. Windows XP Professional für Geschäftskunden und anspruchsvolle Benutzer und die Windows XP Home Edition für den Privatanwender. Die Home Edition unterschied sich durch ein künstlich abgespecktes Rechtemanagement und der fehlenden Active Directory Unterstützung von Professional. Windows XP gilt bei vielen Anwendern als extrem stabiles und schnelles Betriebssystem. Enthält zusätzlich: „Luna-Design“, Systemwiederherstellung, Remotedesktop, Firewall, NTFS 3.1, Clear Type, unterstützt bis zu **64x CPU**’s und **1 TB RAM**, usw.

Windows Vista und Windows Server 2008

Jahr	Version	Bezeichnung	Editionen	Sprachen	Plattformen
Januar 2007 Februar 2008	NT 6.0	Windows Vista Windows Server 2008	- Starter - Home Basic - Home Premium - Business - Enterprise - Ultimate - Express Edition - Standard Edition - Enterprise Edition - Datacenter Edition - Web-Server - Small Business	C Assembler (bisschen) C++ C# (.NET)	x86 AMD64/EM64T IA-64

Windows Vista und der Windows Server 2008 basieren auf dem identischen Kern. Enthält zusätzlich: „Aero-Design" eine vektorbasierte Oberfläche, WDDM-Grafik-Treibermodell, NET Framework 3.0, WPF = Windows Presentation Foundation, WF = Windows Workflow Foundation, WCF = Windows Communication Foundation, UAC = User Account Control, unterstützt bis zu **64x CPU**’s und **2 TB RAM**, usw.

Prognose: Windows 7 und Windows Server 2008 R2

Jahr	Version	Bezeichnung	Editionen	Sprachen	Plattformen
Ca. Januar 2010	NT 6.1	Windows 7 Windows Server 2008 R2	- Starter - Home Basic - Home Premium - Professional - Enterprise - Ultimate - *Express Edition??* - Standard Edition - Enterprise Edition - Datacenter Edition - Web-Server - *Small Business??*	C Assembler (bisschen) C++ C# (.NET)	x86 AMD64/EM64T IA-64 ARM???

Windows 7 und der Windows Server 2008 R2 basieren auf dem identischen Kern. Maximal möglicher Speicherausbau noch nicht endgültig veröffentlicht. Enthält zusätzlich: niedrigerer Ressourcenverbrauch, besserte Netbook Unterstützung, aufgrund von niedrigeren Systemanforderungen, Unabhängigkeit von der Windows HAL, verbessertes Energiemanagement, viele Prozessoren unterstützen, unterstützt bis zu **256x CPU**'s und sehr viel Arbeitsspeicher, usw.

5.5. Grundlegender Aufbau der Windows NT-Architektur

Es handelt sich um ein **Schichtenmodell**, wobei die unteren Schichten (HAL, Mikrokernel, Treiber) des Systems, die auf eine bestimmte Prozessorarchitektur zugeschnitten sind, als getrennte Module implementiert sind, sodass die oberen Schichten des Systems von den Unterschieden der Prozessorarchitektur abgeschirmt werden können.

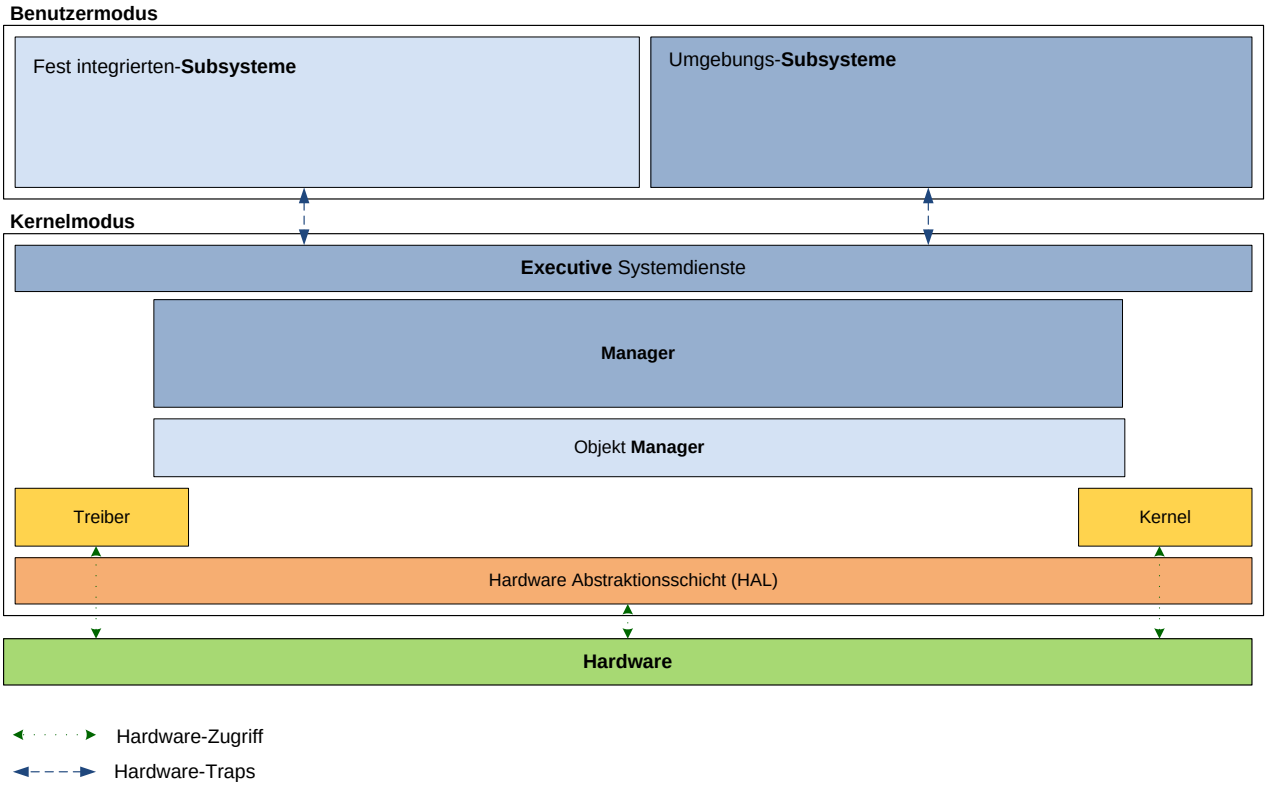


Abbildung 22: Windows NT-Betriebssystemmodell

Um die hohe Portabilität und eine hohe Integrität des Codes zu sichern, wurde auf ein Modell von Dr. Richard Rashid zurückgegriffen, der ab 1991 bei Microsoft Research, der Forschungsabteilung mitwirkte. Dieses Modell fand erstmals Einsatz im Betriebssystem Mach. Es besteht aus einem einfachen statischen Basisbetriebssystem und einzelnen Komponenten, die geschichtet darauf aufsetzen. Diese aufgesetzten Komponenten liefern die erweiterten Eigenschaften des Betriebssystems und können jederzeit verändert werden. Sie sind von den hardwarenahen Komponenten vollständig abgeschirmt. Dieses Modell erhielt Einzug in die Windows NT-Architektur in Form der **privilegierten Ausführungsschicht mit den Systemdiensten (Executive)** und den **nicht privilegierten aber geschützten Subsystemen**. Siehe hierzu Abbildung 22: Windows NT-Betriebssystemmodell. Die Ausführungsschicht wird hierbei im privilegierten Kernelmodus des Prozessors ausgeführt, in dem alle Maschinenbefehle genutzt werden können. Die Subsysteme werden im nicht privilegierten Benutzermodus ausgeführt, in dem nur ein eingeschränkter Satz von Maschinenbefehlen nutzbar ist. Module im **Kernelmodus** (Ring 0) haben direkten Zugriff auf die Hardware oder den Speicher. Das ermöglicht eine höhere Performance. Im Gegenzug steigt aber auch das Risiko eines fehlerhaften Speicherzugriffes. Module im **Benutzermodus** (Ring 3) sind komplett von der Hardware abgeschottet und können Systemfunktionen nur über die Ausführungsschicht mit den Systemdiensten (Executive) ausführen. Das ist eine Sammlung von Komponenten, die den Zugriff auf Hardware und Ressourcen verwalten.

Die Windows NT Architektur arbeitet, um eine hohe Portabilität sicherstellen zu können, ausschließlich mit zwei Prozessorzugriffsmodi (Ring 0 und 3), auch wenn viele Prozessoren mehrere Privilegierungsstufen anbieten.

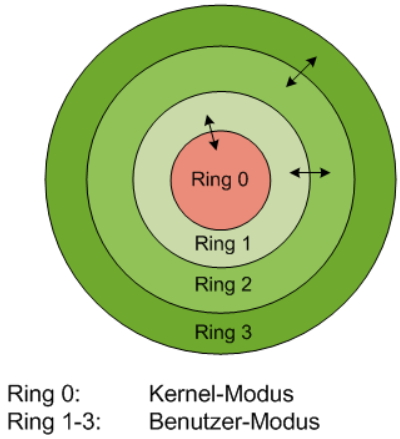


Abbildung 23: Privilegierungsstufen bei x86-kompatiblen Prozessoren

5.6. Die Umgebungssubsysteme

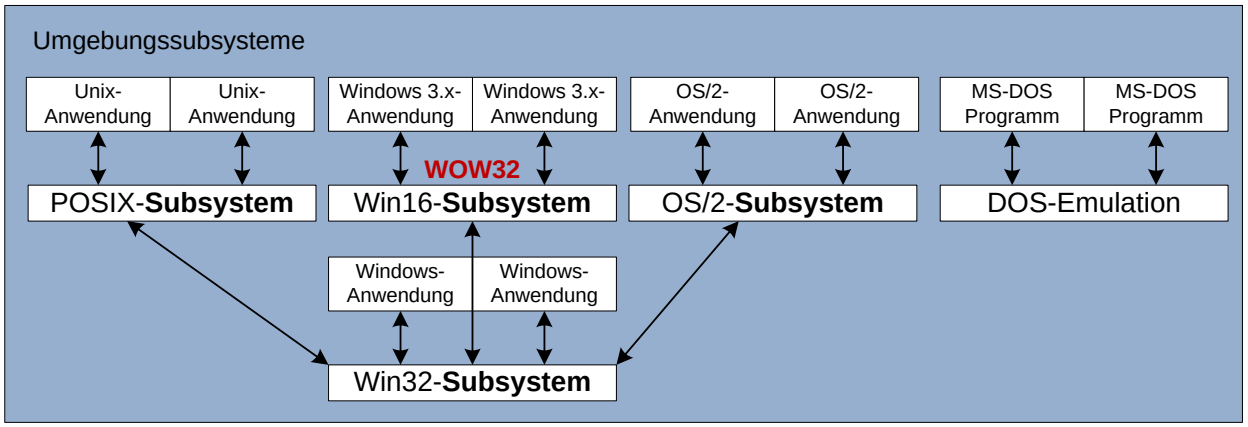


Abbildung 24: Die ursprünglichen Umgebungssubsysteme bis Windows 2000

Alle Subsysteme sind so konzipiert, dass keines das jeweils andere zum Absturz bringen kann. Dies wurde erreicht, indem alle Subsysteme in separaten geschützten Speicherbereichen ablaufen. Die Subsysteme bieten jeweils eigene Programmierschnittstellen, die APIs genannt werden. Diese können von Anwendungsprogrammen genutzt werden, um die Dienste des Betriebssystems zu beanspruchen.

Das zentrale Umgebungssubsystem ist das **Win32-Subsystem,** welches 32-Bit basiert arbeitet oder auf 64-Bit-Versionen das **Win64-Subsystem**, welches 64-Bit basiert arbeitet. Vom zentralen Umgebungssubsystem wird die **Win32-API** und **WIN32S-API** bereitgestellt. Alle modernen Windows Anwendungen, die auf dieser API arbeiten, werden von diesem Subsystem direkt zur Abarbeitung gebracht. Die Kommunikation zwischen Anwendungsprozess und Subsystem erfolgt **nachrichtenorientiert**[11]. Das Win32-Subsystem ist in den Dateien Csrss.exe und Win32k.sys enthalten.

[11] Realisiert durch Local Procedure Calls (LPC)

Die zugehörigen Bibliotheken[12] (Win32-Subsystem DLLs) sind in Abbildung 25: Zugehörigkeiten zum Win32-Subsystem dargestellt. Die Verwendung des Subsystems durch eine Anwendung wird in Abbildung 56: Eine Anwendung erstellt eine Datei demonstriert.

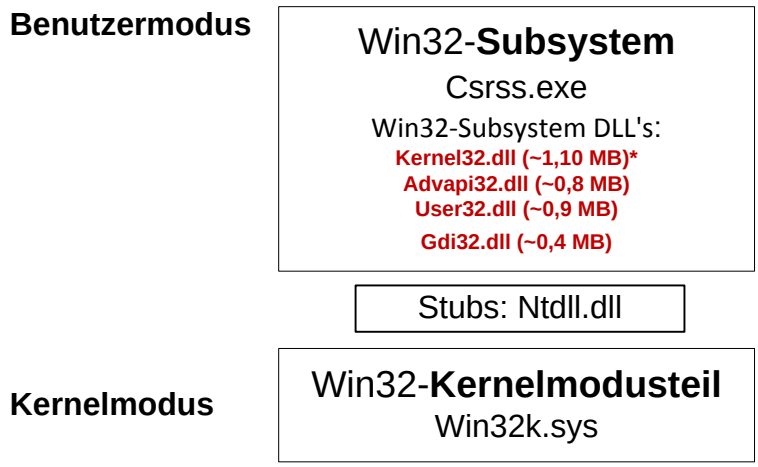


Abbildung 25: Zugehörigkeiten zum Win32-Subsystem bei Windows 7

Datei:	Beschreibung:
Csrss.exe	- Konsolenfenster (Text) - Veranlassen der Threadverwaltung - diverse Funktionen (z. B. GetTempFile...)
Win32k.sys	- Fensterverwaltung - Eingaben von Maus und Tastatur - GDI-Schnittstelle
Win32-Subsystem DLLs	Übersetzen von dokumentierten Win32-API-Funktionen in die entsprechenden Systemfunktionen.
Ntdll.dll	Stubs[13] für die Systemdienste der Ausführungsschicht. Siehe Abbildung 42: Dienstverteiler Stubs

NT-basiertes Windows, erkennt selbstständig für welches Betriebssystem eine Anwendung entwickelt wurde. Die Abarbeitung der Anwendung erfolgt dann über das entsprechende Umgebungssubsystem.

Das wichtigste Umgebungssystem ist das Win32/Win64-Subsystem. Es stellt die grafische Oberfläche bereit und überwacht sämtliche Benutzereingaben und Bildschirmausgaben.

Nicht Win32-API Anwendungen müssen während der Abarbeitung durch ein jeweils geeignetes, darüber liegendes Subsystem auf das zentrale Umgebungssubsystem transferiert werden. Die Kommunikation erfolgt ebenso nachrichtenorientiert. Zu den darüber liegenden Subsystemen gehört das **Win16-Subsystem (WOW32: Windows on Windows 32)** welches 16-Bit-Anwendungen zur Ausführung bringt. Es unterstützt Anwendungen, die ursprünglich für Windows 3.x entworfen wurden. Bei der 64-Bit-Ausgabe von Windows Vista und Windows 7 ist dieses Subsystem nicht mehr enthalten.

[12] Dynamic Link Library: Der Zweck von DLL-Dateien ist, den von einer Anwendung auf der Festplatte und im Hauptspeicher benötigten Speicherplatz zu reduzieren. Jeglicher Programmcode, der von mehr als einer Anwendung benötigt werden könnte, wird deshalb in einer einzelnen Datei auf der Festplatte gespeichert und nur einmal in den Hauptspeicher geladen, wenn mehrere Programme dieselbe Programmbibliothek benötigen.
[13] Ein lokaler Anknüpfungspunkt für Software.

Daher können dort keine 16-Bit-Windows-Anwendungen mehr ausgeführt werden. Alle NT-Versionen vor Windows XP enthielten noch zwei weitere Subsysteme: Für UNIX-Software wurde das **POSIX-Subsystem** aus vorwiegend politischen Gründen integriert. Es unterstützt reine 16-Bit-Konsolenanwendungen, die auf dem POSIX-Interface laufen (standardisiertes Application Programming Interface). Für die Nutzung von textbasierten 16-Bit-Anwendungen für OS/2 wurde ursprünglich noch ein **OS/2-Subsystem** eingebaut. Die zwei Subsysteme wurden jedoch bei der Entwicklung von Windows XP entfernt, da Microsoft keine Notwendigkeit mehr darin sah, diese beizubehalten und eine Fortführung wäre mit den aktualisierten Sicherheitssystemen in Windows XP nicht ohne größere Eingriffe möglich gewesen. Diese Subsysteme wurden bereits unter Windows 2000 kaum mehr benutzt, da der Großteil der Anwender inzwischen auf moderne für Windows geschriebene Anwendungssoftware umgestiegen ist. Seit dem Jahr 2003 bietet Microsoft die Virtualisierungssoftware „**Virtual PC**“ an, welche in der Lage ist, ein komplettes UNIX bzw. Linux Betriebssystem oder ein OS/2-Betriebssystem unter Windows XP bereitzustellen.

Die Windows NT basierten Betriebssysteme und die frühere Consumer Serie von Windows (Windows 9x, Windows ME) verfügen über eine gemeinsame Teilmenge von APIs (Win32 und COM) sowie in einigen Fällen über gemeinsamen Betriebssystemcode.

Mit der Version „Windows XP 64-Bit-Edition“ hat Microsoft der breiten Öffentlichkeit die erste 64-Bit-Version von Windows angeboten. Der Vorteil von 64-Bit liegt darin, dass mehr Speicher adressiert werden kann. Es können somit mehr als 3.5 GB Arbeitsspeicher genutzt werden. Es gab jedoch bereits schon eine „Windows 2000 64-Bit-Edition“ für die IA-64 Systemarchitektur, welche aber wahrscheinlich nur von Microsoft intern verwendet wurde. Windows Vista und das kommende Windows 7 sind ebenso in 64-Bit-Versionen erhältlich. Alle Windows 64-Bit-Versionen sind exklusiv für AMD- und Intel-Prozessoren mit AMD64/EM64T-Erweiterung gefertigt. Sie sind nahezu identisch mit der jeweiligen 32-Bit-Version, bis auf die Tatsache, dass kein Win16-Subsystem (WOW32: Windows on Windows 32) mehr enthalten ist, dafür aber weiterhin ein **Win32-Subsysten (WOW32: Windows on Windows 64)** um die sehr stark verbreitete 32-Bit-Software für die Win32-API problemlos zur Abarbeitung bringen zu können. Die in den 64-Bit-Versionen enthaltene Win32-API wurde auf 64-Bit portiert, behält jedoch den Namen Win32-API bei und verfügt über einen identischen Funktionsumfang. Dennoch wird sie des Öfteren als Win64-API bezeichnet.

Für Windows XP (nur 32-Bit), Windows Vista und Windows 7 ist optional ein rundum erneuertes UNIX-Subsystem, welches den Namen **Interix-Subsystem** hat und mit der Bezeichnung „Subsystem für UNIX-basierte Anwendungen“ vermarktet wird, installierbar. Es übersetzt UNIX-Systemaufrufe auf das Win32-API und ermöglicht das Ausführen von UNIX-Programmen. Mitgeliefert werden mehr als 350 UNIX-Hilfsprogramme, Compiler, diverse Bibliotheken und vieles mehr.

Des Weiteren existiert eine **16-Bit-MS-DOS-Emulation,** welche eine komplette MS-DOS-Umgebung simuliert. Die **cmd.exe** ist hierbei der Kommandozeileninterpreter analog wie in OS/2. Mit Windows 2000 beginnend, wurden die verfügbaren Konsolenprogramme stark ausgebaut, da diese für die Automatisierung von diversen Tätigkeiten gut zu gebrauchen waren. Mit Windows 7 zieht die Windows **PowerShell** als universelles Werkzeug und potenzieller Nachfolger der Kommandozeilen-Laufzeitumgebung in das Betriebssystem ein. Diese verbindet die aus UNIX-Shells bekannte Philosophie von Pipes und Filtern mit dem Paradigma der objektorientierten Programmierung.

Die **PowerShell Scripting Language** ermöglicht das Schreiben von Skripten und kann somit die Funktionen des MS-DOS Kommandozeileninterpreters im Zusammenspiel mit Batch[14]-Funktionalitäten problemlos ersetzen.

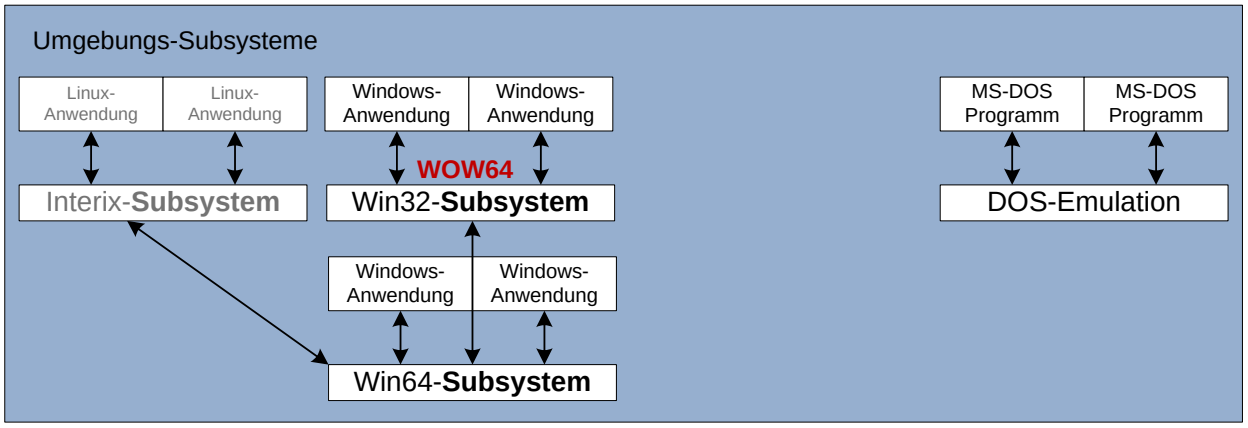


Abbildung 26: Die Umgebungssubsysteme der 64-Bit-Version von Windows 7

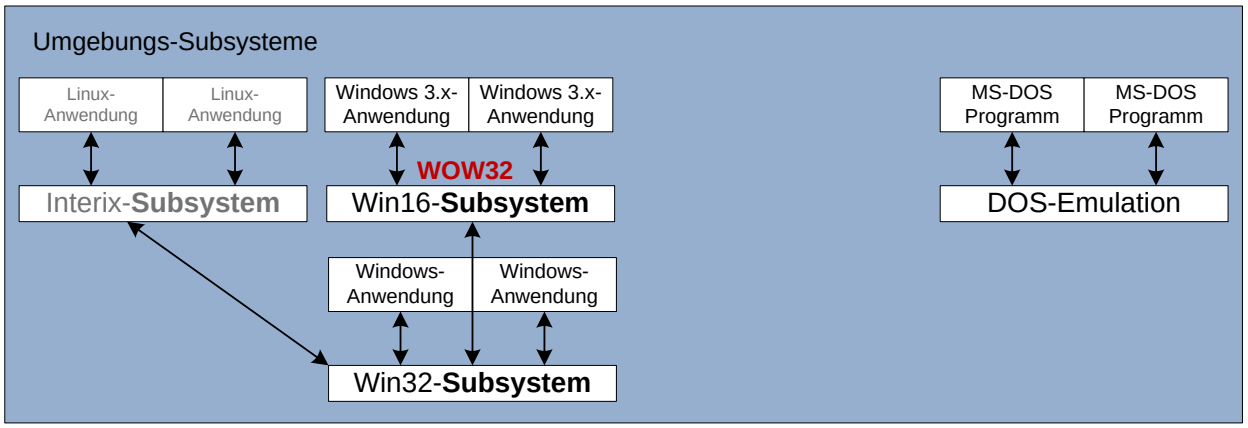


Abbildung 27: Die Umgebungssubsysteme der 32-Bit-Version von Windows 7

In Windows 7 wird es einen **Windows XP Modus** geben. Dieser baut auf dem Produkt Virtual PC auf (siehe oben) und muss optional aktiviert werden. Eine fertig konfigurierte virtuelle Maschine mit dem Betriebssystem Windows XP wird von Microsoft „Out-of-the-box“ für Kunden der Versionen Professional, Enterprise, und Ultimate kostenfrei zur Verfügung gestellt. Nach der Installation integriert sich diese in das Betriebssystem Windows 7 und ermöglicht es Anwendungsprogramme die auf dem virtuellen XP Computer installiert sind, in gewöhnlichen Windows 7 Fenstern laufen zu lassen. Diese Funktion wird auch als Seamless-windows bezeichnet. Im Gegensatz zu Virtual PC ist diese Art der Virtualisierung für den Anwender völlig Transparent, da die XP-Anwendungsprogramme sich vollständig in das Windows 7 Startmenü eintragen und der Windows XP Desktop mit Startmenü nicht in Erscheinung treten muss, um Windows XP Anwendungsprogramme zu starten.

[14] Stapelverarbeitungsprogramm

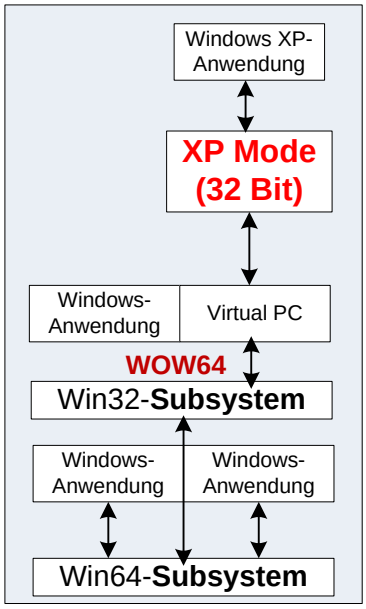


Abbildung 28: Der Windows XP Mode in Windows 7 64-Bit

5.7. Die wichtigsten integralen Subsysteme

Auch die fest integrierten Subsysteme sind so konzipiert, dass keines das jeweils andere zum Absturz bringen kann. Sie laufen wie die Umgebungssubsysteme, in separaten Speicherbereichen und im Benutzermodus des Prozessors ab und implementierten wichtige Betriebssystemfunktionalitäten.
Eine Anwendung die mit Hilfe ihres geeigneten Umgebungssubsystems ausgeführt wird, kann mit dem integralen Subsystem kommunizieren. **System- und Serverprozesse** und Dienste sind jeweils als integrales Subsystem implementiert. Ein weiterer Vertreter ist das **Sicherheitssubsystem**, dass für die Benutzerauthentifizierung und die Active Directory Anbindung zuständig ist. Es verwaltet die Zugriffsrechte auf Systemressourcen und stellt die Benutzerverwaltung bereit. Der Workstation-Dienst und der Serverdienst übernehmen diverse Verwaltungsaufgaben im Netzwerk.

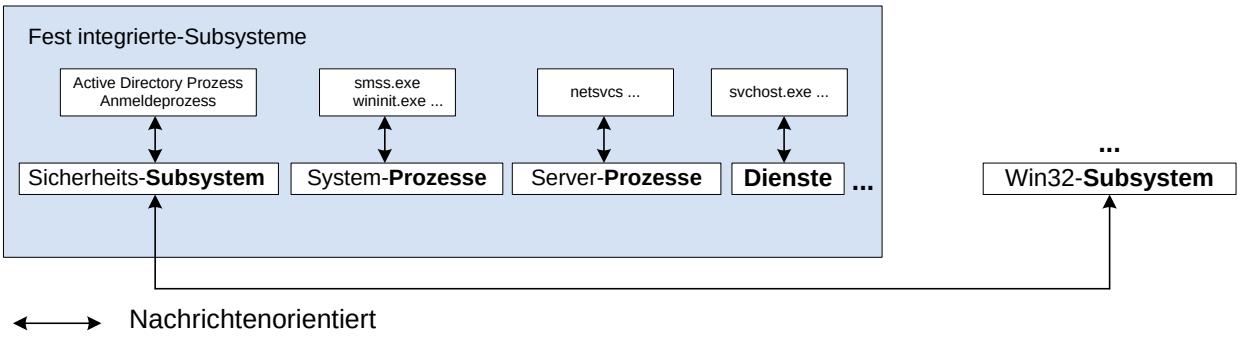


Abbildung 29: Windows 7 und seine fest-integrierten Subsysteme

Neue Subsysteme wurden bei jeder neuen Version, basierend auf der Windows NT-Architektur hinzugefügt. Bei Windows 7 wurde z. B. das **Biometrie-Subsystem** eingeführt, welches für die biometrische Nutzeridentifizierung zum Log-in und zur Bestätigung von Abfragen durch das Sicherheitssystem User Access Control (UAC)[15] dient.

[15] Benutzerkontensteuerung: Sie schützt Benutzer, welche mit administrativen Rechten arbeiten durch einen gezielten Hinweis, wenn diese von ihren administrativen Rechten Gebrauch machen.

In Windows NT 3.1, 3.5, Windows Vista und Windows 7 wird das **Graphics Device Interface** (GDI) in einem eigenen Subsystem und nicht im Kernel ausgeführt. Es ist verantwortlich für die Darstellung von grafischen Objekten und ihrer Übermittlung an Ausgabegeräte wie Bildschirme und Drucker. Mit Windows Vista wurden die zusätzlichen Techniken DWM[16] und WDDM[17] eingeführt, um die neue Aero-Oberfläche mit Hardwarebeschleunigung zu ermöglichen.

5.8. Windows-Prozesse und Threads

Prozesse unter Windows sind **spezielle Container** für Systemressourcen, die von Threads verwendet werden, die eine Instanz eines Programmes ausführen. Ein Prozess beinhaltet mindestens einen Thread und erhält einen eigenen privaten Speicherbereich aus dem virtuellen Adressraum der Windows-Speicherverwaltung[18]. Da jeder Prozess über einen eigenen Adressraum verfügt und nur ausschließlich mit diesem arbeiten kann, kann ein fehlerhafter Prozess nicht mehr das ganze System zum Abstürzen bringen. Unter Windows wird über die Win32-API Prozesse gestartet und verwaltet. Erledigt wird das dann durch den Prozess-Manager in der Schicht „Exekutive“. Der Zugriff auf die Kernelobjekte eines Prozesses ist die Aufgabe von sogenannten Handles.

Im Gegensatz zu UNIX gibt es unter Windows keinen strukturierten Prozessbaum. Dieser kann jedoch mit Hilfe von Auftragsobjekten, die mit Windows 2000 eingeführt wurden, nachgebaut werden.

Windows-Prozess (Container)
Besteht aus (Kontext): • einem privaten virtuellen Adressraum • mindestens einem Thread • ausführbares Programm, welches auf dem virtuellen Adressraum des Prozesses abgebildet wird • einer Liste offener Handles • einem Sicherheitskontext: Ein Zugriffstoken für Rechte • einer Prozess-ID / Client-ID

Abbildung 30: Aufbau eines Prozesses unter Windows

Threads teilen sich virtuelle Speicheradressen und Handles innerhalb ihres Prozess-Containers. Sie werden vom Kernel zur Ausführung gebracht, also dem Prozessor zugeteilt.

[16] Windows Display Driver Model (WDDM)
[17] Desktop Window Manager, ist ein Composite-fähiger Window Manager
[18] Siehe Kapitel 5.11

Windows-Thread *
Besteht aus (Kontext): • dem Inhalt eines Satzes von CPU-Registern, die den Prozessorstatus repräsentieren • zwei Stacks (einer Kernel-Modus, anderer Benutzer-Modus) • Einem TLS (privater Speicherbereich) für Laufzeitbibliotheken und Teilsysteme • Einer Thread-ID / Client ID • optional einem ergänzenden Sicherheitskontext

* musste bei jeder Rechnerarchitektur (x86, Alpha…) anders implementiert werden

Abbildung 31: Aufbau eines Threads unter Windows

Ein neuer Zustand und Prozessorgruppen in Windows 7

Um Threads besser auf viele CPU-Kerne zu verteilen, wurde mit Windows 7 ein neuer Zustand neben "Running" und "Waiting" eingeführt, den die Threads annehmen können: "Pre-**waiting**". Dieser soll den Umgang mit mehreren Kernen deutlich vereinfachen. Dabei steht der Thread nicht komplett, sondern es werden nur einzelne Objekte gesperrt, auf deren Input der Thread wartet. Unter Volllast kann Windows 7 also insgesamt schneller arbeiten, weil die Threads beschleunigt werden. Zudem wurde der "dispatcher-lock" entfernt, der die stabile Skalierung mit mehr als 32 Prozessoren verhindert hat. Siehe Abschnitt „**Die Überwindung des Dispatcher Lock (Lastenverteiler)“** in Kapitel 5.10.

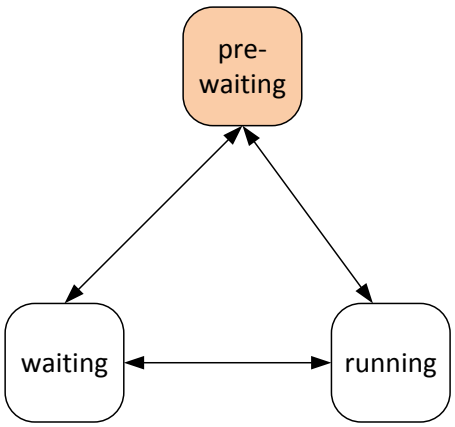


Abbildung 32: Ein neuer Thread Zustand in Windows 7

Threads sind mit einer binären Bitmaske codiert die 32 Zeichen lang ist und den Thread zu einem logischen Prozessor (Kern) zuordnet. Jeder Prozessor-Kern ist in Windows als logischer Prozessor vertreten.

Die Bitmaske kann folgendermaßen aussehen:

00000000 00000000 00000000 00000000 =
Thread kann auf alle Prozessoren zur Ausführung gebracht werden (Affinität grundsätzlich aus)

00000000 00000000 00000000 00000001 =
Thread läuft nur auf dem ersten Prozessor

00000000 00000000 00000000 00000010 =
Thread läuft nur auf dem zweiten Prozessor

00000000 00000000 00000000 00000100 =
Thread läuft nur auf dem dritten Prozessor

00000000 00000000 00000000 00000111 =
Thread kann auf den ersten drei Prozessoren verteilt werden

Wie man sehen kann, funktioniert dies mit bis zu 32 Kernen, denn jeder Kern benötigt für sich ein Bit. Eine Umstellung auf eine Bitmaske mit höherer Wortbreite wäre unmöglich gewesen. Daher wurde das System erweitert. Systeme, die über mehr als 32 logische Prozessoren verfügen, stellen Prozessorgruppen bereit. Maximal 64 logische Prozessoren bilden eine Gruppe, die als einzelne Einheit behandelt wird und vom Scheduler, als ein einzelner Prozessor betrachtet wird. Wenn das System gestartet wird, erstellt das Betriebssystem eine Prozessorgruppe und weist dieser logische Prozessoren zu. Ein System kann bis zu vier Gruppen haben. (0 bis 3). Systeme mit weniger als 32 logischen Prozessoren haben immer eine einzige Gruppe, die Gruppe 0. Ein System mit 256 logischen Prozessoren hätte alle vier Prozessor-Gruppen (4x64). Bei der Bildung der Prozessorgruppen wird die tatsächliche Zusammengehörigkeit von den Kernen zu den jeweiligen Prozessoren berücksichtigt. Die Kerne eines Prozessors bilden einen Knoten. Windows 7 wurde bereits auf einer Hardware mit 256 Kernen in der Öffentlichkeit vorgeführt.

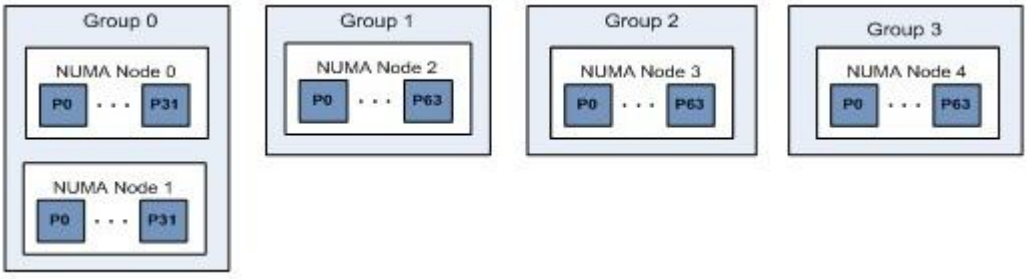


Abbildung 33: Die NUMA-Architektur wurde durch Prozessorgruppen realisiert

5.9. Grundtypen von Benutzermodus-Prozessen in Windows

Windows NT unterstützt vier Grundtypen von Benutzermodus-Prozessen, die man der allgemeinen Windows NT-Architektur (Abbildung 22: Windows NT-Betriebssystemmodell) oder hier am konkreten Beispiel von Windows 2000 zuordnen kann. Seit Windows Vista gibt es einen weiteren Typ, die **geschützten Prozesse**.

1. Prozesstyp:	**Fixe Prozesse**
Komponenten:	**Sicherheits-Subsystem, System-Prozesse**
Es handelt sich um fixe Prozesse, wie z. B. der Anmeldungsprozess oder der Sitzungsmanager die in Windows keine Dienste sind. (nicht vom Dienststeuerungs-Manager verwaltet)	

2. Prozesstyp:	**Dienstprozesse**
Komponenten:	**Dienste**
Laufen auf dem Win32-Umgebungs-Subssystem. Können von Anwendungen in Windows eingebunden werden. z. B. Microsoft Exchange Server, Antiviren Guard...	

3. Prozesstyp:	**Anwendungsprozesse**
Komponenten:	**Anwendungen**
Programme die auf einem Umgebungssubsystem ausgeführt werden. z. B. Word oder Excel	

4. Prozesstyp:	**Umgebungssubsystem-Prozesse**
Komponenten:	**Umgebungssubsysteme**
Stellen eine Betriebssystemumgebung zur Verfügung, auf der Programme ausgeführt werden können.	

Prozesse dürfen unter Windows wieder neue Prozesse starten. So kann ein Anwendungsprogramm (Prozess 1) ein neues Dokument (Prozess 2) öffnen. Das klassische Prozessmodell erlaubt es auch, dass das Anwendungsprogramm das neue Dokument wieder löscht oder verändert. Ein Prozess hat in der Regel die Kontrolle über seine Kindprozesse. Für Administratoren auf einem Windows Rechner gibt es jedoch bei Standardprozessen auch eine Möglichkeit auf diese zuzugreifen. Möglich macht das die Berechtigung „Debuggen von Programmen“, die einem Administrator vollständigen Zugriff auf einen Prozess erlaubt. So kann er den Adressraum des Prozesses und die im Prozess genutzten Daten auslesen oder ändern.

Seit Windows Vista:

5. Prozesstyp:	**Geschützte Prozesse**
Vista hat extra für Multimediadateien und bestimmte Systemdateien das Prozessmodell verbessert und den neuen Typ „geschützte Prozesse“ definiert. Diese Prozesse schränken auch den Zugriff auf die Prozessverwaltung für Administratoren ein. Der Kernel stellt zwar auch Diagnoseinformationen für geschützte Prozesse bereit, Direktzugriff gibt es aber nicht. Nach dieser Methode kann etwa ein Codec zum Abspielen eines Films nur unter einer Bedingung als geschützter Prozess ablaufen.	

5.10. Der Kernel

Wie bereits in 5.2 ausführlicher beschrieben wurde, handelt es sich bei Windows 7 um ein Betriebssystem mit einem Kernel, der wichtige Eigenschaften eines Mikrokernels aufweist. Die Versionen ab Windows NT 4.0 bis Windows Vista, besaßen eindeutig einen Hybridkernel, da das Grafiksystem mit im Kernel saß. Ab Windows Vista wurde der anfängliche Zustand wiederhergestellt.

Die Hauptaufgabe des Kernels die Behandlung von **Interrupts**[19] und **Exceptions**[20]. Er verteilt die CPU-Zeit der Windows-Threads in Zeitscheiben, um ein **preemptives-Scheduling**[21] zu ermöglichen. Siehe Anforderungen in 5.2. Bei Multiprozessorsystemen synchronisiert er die Aktivitäten der verschiedenen Prozessorkerne. Der Kernel in Windows 7 ist fähig bis zu 256 Prozessoren zu skalieren. → Artikel!

Kurz und knapp - Aufgaben des Kerns:

- Behandlung von Interrupt und Exceptions
- Symmetrische Multiprozessorsynchronisation
- Scheduling und Dispatching von Threads
- Andere Übersetzung für andere Hardwareplattform nötig

Die Schnittstellen, welche der Kernel bereitstellt, werden von der Executive auf einer höheren Schicht implementiert. Der Kernel ist in der Datei **ntoskrnl.exe** integriert. Des Weiteren ist in ihr die Executive enthalten. Der Kernel muss teilweise modifiziert und erneut übersetzt werden, um Windows auf einer anderen Plattform mit anderen Prozessoreigenschaften betreiben zu können. Im Gegensatz zur HAL sind aber nur sehr geringe Änderungen erforderlich. Der Umfang von Kernel + Executive hat bei jeder neuen Version deutlich zugenommen. Interessanterweise hat der Windows 7 Entwicklungsleiter Steven Sinofsky erwähnt, dass der Kernelanteil mit Windows 7 deutlich kleiner ausfällt, als der des Vorgängers Windows Vista. Trotzdem ist die ntoskrnl.exe etwas größer, was an den neuen Managern der Executive liegt.

Version:	Für Architektur:	Größe:	Beschreibung:
4.00.1381	x86	0,92 MB	Windows NT 4 SP6
5.00.2195.7133	x86	1,61 MB	Windows 2000 SP4
5.1.2600.3093	x86	2,08 MB	Windows XP SP2
5.2.3790.4035	x86	2,32 MB	Windows 2003 Server SP2
5.2.3790.4478	x86	2,33 MB	Windows 2003 Server R2 SP2
6.0.6001.18000	x86	3,38 MB	Windows Vista SP1 und Windows Server 2008
6.0.6001.18226	x64	4,47 MB	Windows Vista SP1 und Windows Server 2008
6.1.7100.0	**x86**	**3,71 MB**	**Windows 7 RC und Windows Server 2008 R2 RC**
	x64	**5,24 MB**	**Windows 7 RC und Windows Server 2008 R2 RC**

Abbildung 34: Dateigröße von ntoskrnl.exe, die den Kernel und die Executive enthält.

Ursprünglich (bis Windows 2000) war der Kernel auf die Verwendung von maximal 32 Prozessoren physikalisch beschränkt, was teilweise für verschiedene Produktausführungen künstlich eingeschränkt wurde. Windows 7 bzw. Windows Server 2008 R2 wurde bereits auf Systemen mit 256 Prozessoren öffentlich vorgeführt.

[19] Kurzfristige Unterbrechung einer Prozedur, um eine andere, meist kurze, aber zeitkritische Verarbeitung durchzuführen.
[20] Ausnahmesituation
[21] Ein Prozessor kann mehrere Threads nutzen.

Jedoch ist je nach Produktausführung auch Windows 7 bzw. Windows Server 2008 R2 wieder künstlich auf eine festgelegte Anzahl an Prozessoren beschränkt. Siehe Abschnitt: Die Überwindung des Dispatcher Locks (Lastenverteiler).

Windows ermöglicht den **symmetrischen Multiprozessorbetrieb**[22]. Abzuarbeitende Systemthreads und Benutzerthreads werden stets auf mehrere verfügbare Prozessoren verteilt. Im Gegensatz zum asymmetrischen Multiprozessorbetrieb gibt keinen Hauptprozessor, der ausschließlich Systemthreads ausführt und Nebenprozessoren, die ausschließlich Benutzerthreads ausführen. Alle Prozessoren sind in der Lage Systemthreads und Benutzerthreads auszuführen. Des Weiteren nutzen alle Prozessoren einen gemeinsamen Speicherbereich.

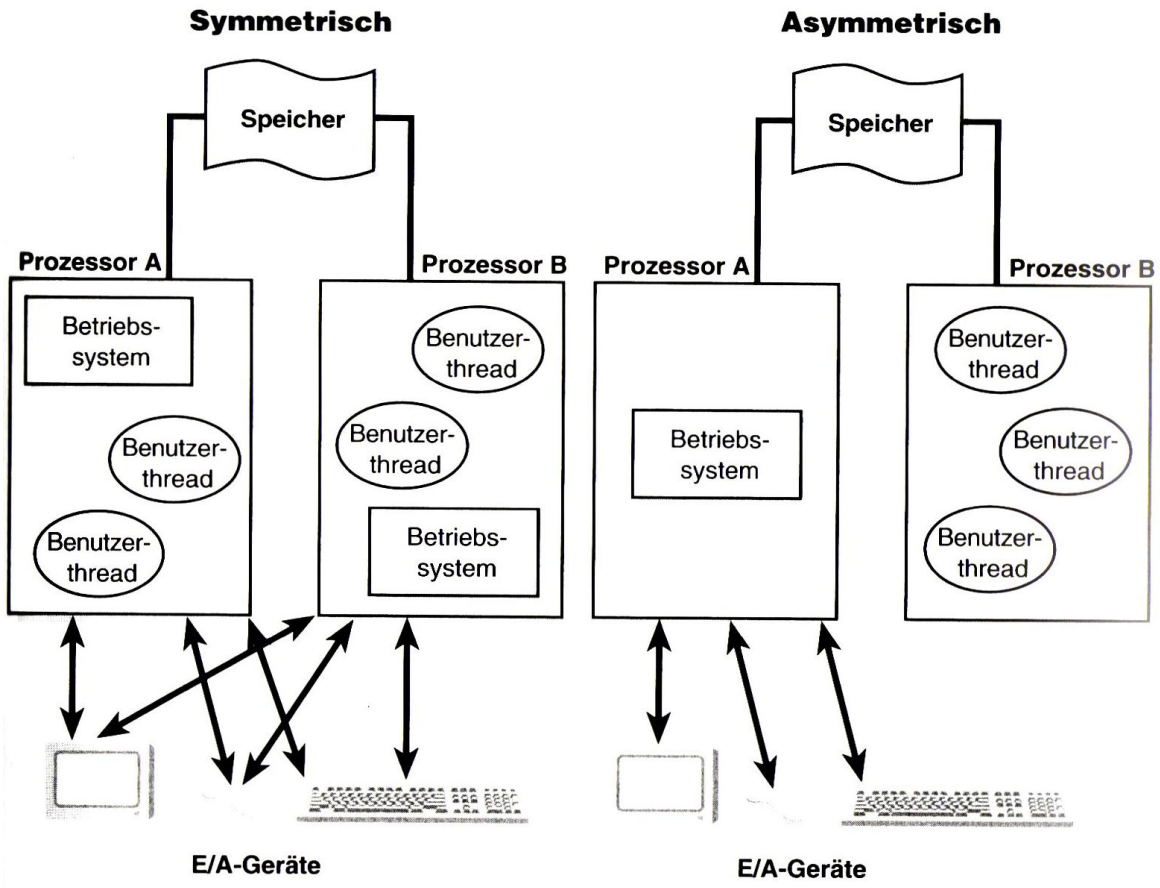


Abbildung 35: Symmetrischer- und asymmetrischer Multiprozessorbetrieb im Vergleich

Kurz und knapp - symmetrischer Multiprozessorbetrieb:

- Systemthreads können auf einem oder mehreren Prozessoren gleichzeitig ausgeführt werden.
- Benutzerthreads können auf einem oder mehreren Prozessoren gleichzeitig ausgeführt werden.
- Ein Prozess (Container) bestehend aus mehreren Threads, kann auf mehreren Prozessoren gleichzeitig ablaufen.
- Herausforderung für das Betriebssystem: Synchronisation innerhalb des Kernels und auf Benutzerebene.

[22] Threads können gleichzeitig von mehreren Prozessoren abgearbeitet werden.

Die Überwindung des Dispatcher Lock (Lastenverteiler) in Windows 7

Die 64-Bit-Versionen von Windows 7 und Windows Server 2008 R2 unterstützen nun dank Änderungen im Kernel, mehr als 32 logische Prozessoren (LP) auf einem einzelnen Computer mit Non-Uniform Memory Access (NUMA) Hardware-Architektur. Windows 7 wird zukünftig bis zu 256 Prozessoren (bzw. 256 Kerne) unterstützen. In Windows 7 konnte der Microsoft Entwickler Arun Kishan den **Dispatcher**[23] **Lock (Lastenverteiler Beschränktheit)** überwinden, welcher seit der ersten NT Version bestand. Zuvor sah es jahrelang so aus, dass diese Möglichkeit mit dem NT-Design definitiv nicht möglich sein würde, was aber glücklicherweise nicht stimmt. Mehr als 32 logische Prozessoren zu betreuen war das Ziel, das bereits seit der Einführung von Windows NT 4.0 immer wieder erneut angestrebt wurde, jedoch nie erreicht wurde. Auch David Cutler, der einst den Weg für die Windows-NT-Technologie ebnete, hatte sich daran die Zähne vergeblich ausgebissen. Erreicht wurde das durch eine starke Verfeinerung der Granularität, der vielen Synchronisationstechniken die im Kernel verwendet werden. Diese Veränderung im Kernel gilt als einer der größten Eingriffe, den es seit Jahren gegeben hat. Er hatte eine zusätzliche Anpassung der Speicherverwaltung und eine Anpassung der Threads (siehe 5.8) zur Folge. Somit wurde definitiv Code im Kernel seit Windows Vista und Windows XP verändert. Außerdem wurde dadurch der Kernel für die Zukunft ausgerüstet und soll auch im Windows 7 Nachfolger angeblich verwendet werden.

Die einzige bisher öffentliche umfangreichere Quelle zum Windows 7 Kernel ist ein Video von Mark Russinovich, das unter dem folgenden Link zu finden ist:
http://channel9.msdn.com/shows/Going+Deep/Mark-Russinovich-Inside-Windows-7/

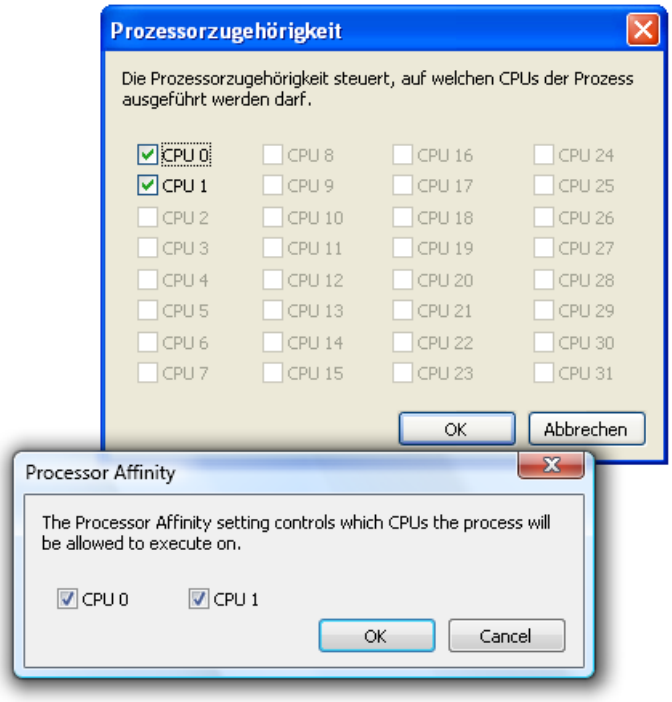


Abbildung 36: Aufgrund des Dispatcher Locks unter Windows XP können maximal 32 logische Prozessoren an einen Prozess gebunden werden.

[23] Im Rahmen der Prozessverwaltung dient der Dispatcher dazu, bei einem Kontextwechsel dem derzeit aktiven Thread die CPU zu entziehen und anschließend dem nächsten Thread die CPU zuzuteilen.

5.11. Die Speicherverwaltung

Die Architektur der Speicherverwaltung wurde maßgeblich von Rick Rashid festgelegt, der inzwischen Senior Vice Präsident bei Microsoft Research ist. Er hatte bereits bei der Entwicklung des UNIX-Mach-Kernels mitgewirkt und brachte von dieser Seite Einflüsse mit in die NT-Entwicklung.

Im Folgenden wird die Speicherverwaltung der 32-Bit-Versionen beschrieben. Diese verfügen über einen **linearen Adressraum von 32-Bit**. Das ergibt **einen 4 GB großen virtuellen Speicher,** der für jeden einzelnen Prozess bereitgestellt wird. Dieser ist in zwei Hälften zerlegt:

- Untere Hälfte von 0x00000000 bis 0x7FFFFFFF für **Anwendungsprozesse** (= 2 GB)
- Obere Hälfte von 0x80000000 bis 0xFFFFFFFF für **geschützte Systemoperationen** (= 2 GB)

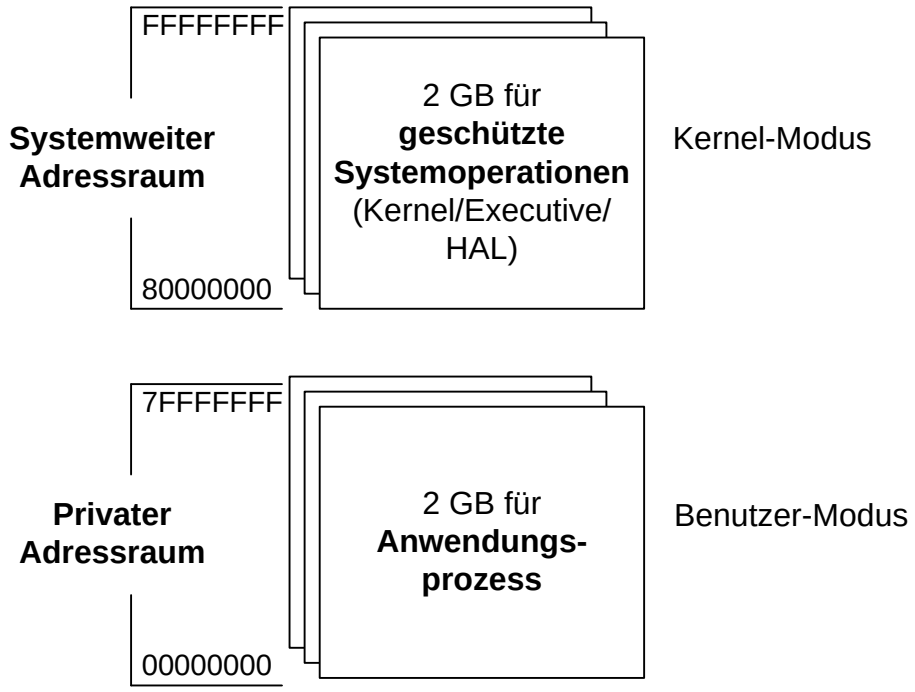


Abbildung 37: Der 4 GB große virtuelle Adressraum

Der virtuelle Speicher bietet eine logische Ansicht des Speichers, die nicht mit dem physikalischen Layout übereinstimmt. Der virtuelle Speicher ist nicht vom vorhandenen physischen Speicher auf dem Computer abhängig. Ein Prozess kann 2 GB Speicher nutzen. Während der Laufzeit werden die virtuellen Adressen von der Speicherverwaltung mit Unterstützung der Hardware in physikalische Adressen abgebildet, in denen die Daten tatsächlich abgelegt werden.

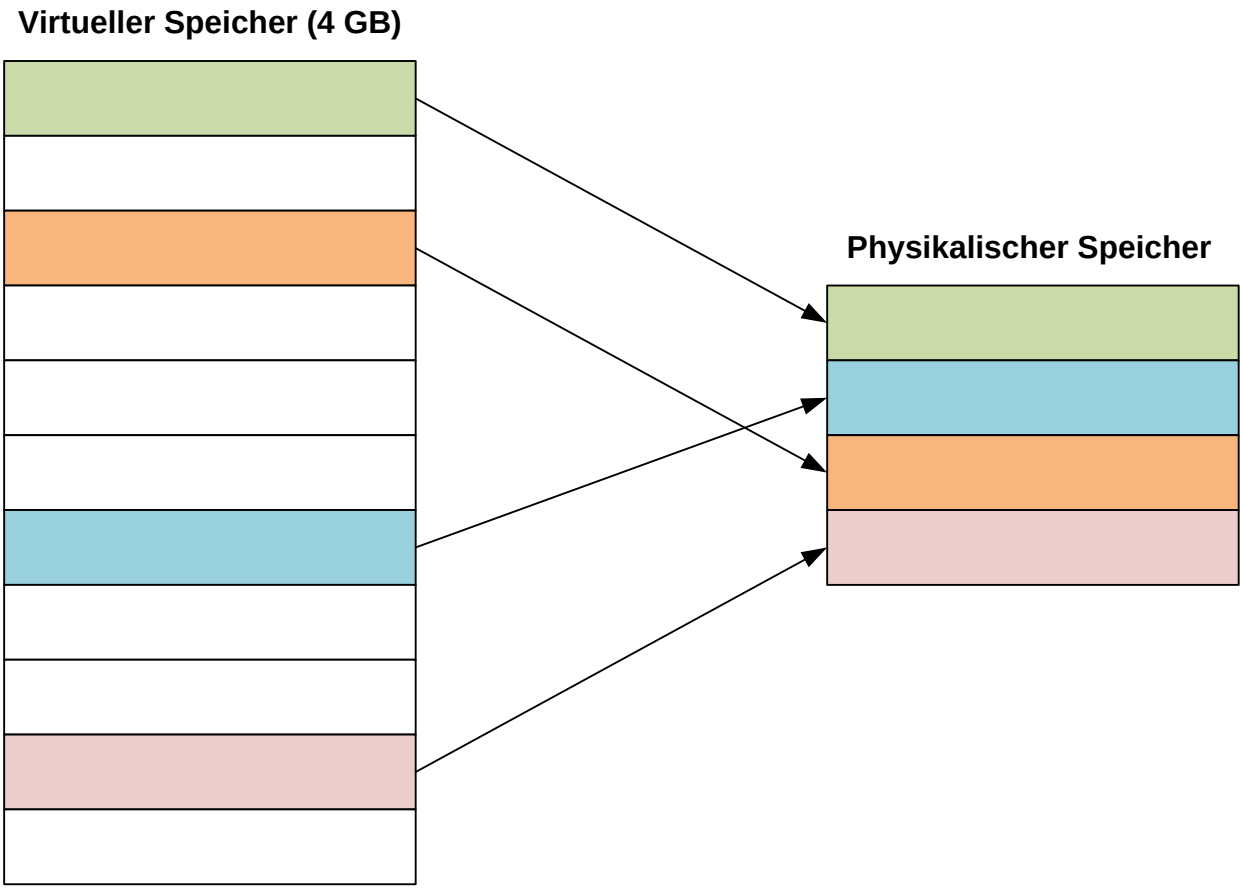


Abbildung 38: Abbildung des virtuellen Speichers auf den physikalischen Speicher

Damit Anwendungen einen größeren Speicherbereich als 2 GB nutzen können, wurde mit Windows 2000 eine Erweiterung, die AWE (Address Windowing Extension) eingebaut. AWE erlaubt es einem Programm, physische Adressbereiche zu reservieren und diese in den virtuellen Adressraum des Prozesses als Fenster einzublenden. Diese als "Overlay" oder "Windowing" bekannte Technik ist vom Prinzip her ähnlich zu dem unter MS-DOS angebotenen EMS-Speicher. Das Overlay muss jedoch dann von den Anwendungsprogrammen selbstständig verwaltet werden. Bis zu maximal 64 GB Speicher können von 32-Bit-Anwendungen in ihrem 2 GB großen virtuellen Adressraum eingeblendet werden.

Ein großer Nachteil, der Windows NT Speicherarchitektur ist jedoch die Tatsache, dass in der oberen Hälfte (beinhaltet geschützte Systemoperationen) keine Schutzmechanismen verwendet werden, wie das bei der unteren Hälfte (beinhaltet Anwendungsprozesse) der Fall ist, sodass aller dort ausgeführter Code uneingeschränkt Zugriff auf den gesamten Adressbereich hat. Die Folge ist, dass Gerätetreiber welche in der oberen Hälfte liegen, bei falscher Programmierung das gesamte System verletzen und somit zum Absturz bringen können. Dieses Manko wurde mithilfe eines speziellen Treibersignaturmechanismus, der den Benutzer warnt, wenn er nicht autorisierte Treiber hinzufügt, gelöst. In den 64-Bit-Ausgaben geht das soweit, dass ein nicht autorisierter Treiber nicht ohne spezielle Maßnahmen eingebunden werden kann.

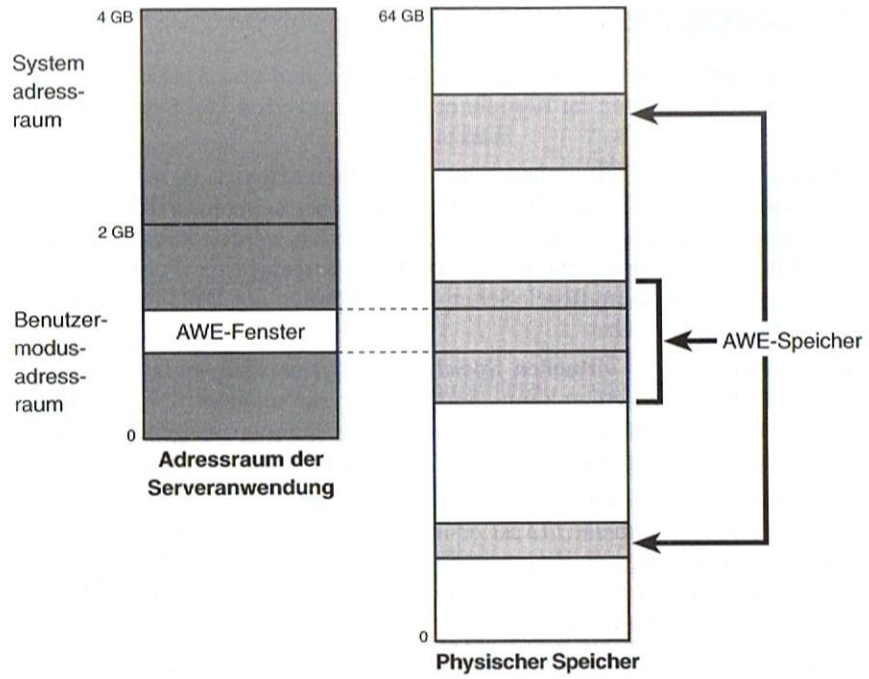


Abbildung 39: AWE (Address Windowing Extension)

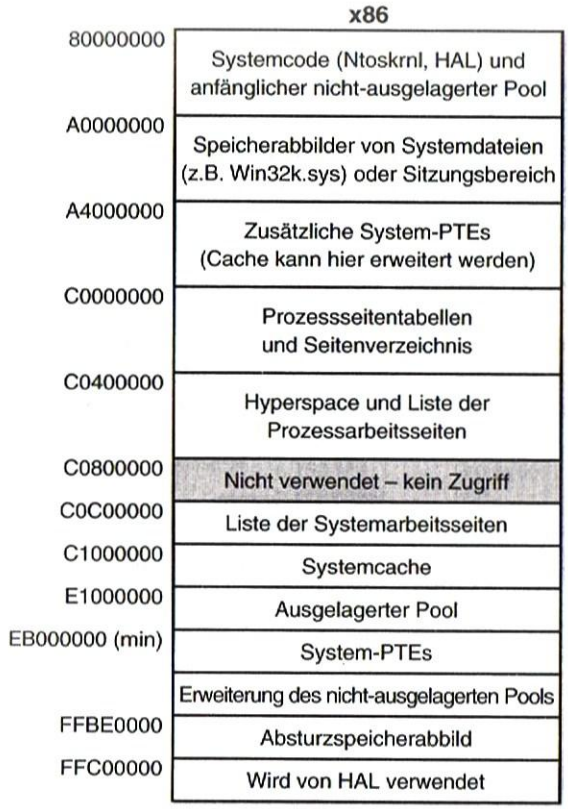


Abbildung 40: Layout des Systemadressraums

5.12. Das Paging in die Auslagerungsdatei

Auf vielen Systemen ist der virtueller Speicher größer als der physikalische Speicher. Daher wird ein Teil des Speicherinhalts von der virtuellen Speicherverwaltung auf die Festplatte ausgelagert. Dort befindet sich eine Auslagerungsdatei, die standardmäßig 1,5 fach so groß wie der physikalische Arbeitsspeicher ist. Wenn ein Thread auf eine virtuelle Adresse zugreift, die auf der Festplatte ausgelagert wurde, lädt der Speicher-Manager diese Daten von der Festplatte wieder in den Arbeitsspeicher. Für Anwendungen läuft dieser Vorgang völlig transparent ab. Dabei ist bei Windows für jede Seite im virtuellen Speicher ausgewiesen, in welchem Zugriffsmodus sich der Prozessor befinden muss, um die Seite zu lesen oder zu schreiben. Die Anwendungsprozesse (unter Hälfte) beanspruchen den Benutzer-Modus, die geschützten Systemoperationen (obere Hälfte) erfordern den Kernel-Modus.

5.13. Die Hardwareabstraktionsschicht

Die Hardwareabstraktionsschicht (HAL = Hardware Abstraction Layer) ist die unterste Schicht der Windows NT-Architektur und befindet sich unmittelbar über der Hardware. Sie trennt die Executive von der Hardwareumgebung ab und sorgt dafür, dass die oberen Schichten nicht die genaue Spezifikation der Hardware einhalten müssen, indem sie die von einer bestimmten Hardwareklasse bereitgestellte Funktionalität von der konkreten Implementierung abstrahiert und dafür allgemeine HAL-Routinen anbietet. Die konkrete Umsetzung auf die Hardware erfolgt dann über einen hardwarespezifischen Treiber. Sie übernimmt für alle Komponenten die Kommunikation mit der Hardware. Nur ein paar Bestandteile des Kernels und Treiber können, wenn nötig direkt mit der Hardware kommunizieren, also ohne die HAL zu benutzen. Die HAL ist der größte Teil, der von Windows in reinem Assembler-Code geschrieben ist.

Mithilfe der HAL konnte das Hauptziel „volle Portabilität“ erreicht werden. Um Windows auf einer anderen Hardwareplattform betreiben zu können, bedarf im günstigsten Fall ausschließlich einer Anpassung der HAL an diese andere Plattform. In extremeren Fällen (z. B. andere Rechnerarchitektur) ist eine erneute Übersetzung des Kernels nötig, einschließlich weitere Dateien der Executive (siehe Abbildung 58: Wichtige Systemdateien von Windows 7). Die erste Version Windows NT 3.1 konnte auf x86-kompatibler Hardware und auf einigen RISC kompatiblen Prozessoren ausgeführt werden. Ab Version 3.51 arbeitete es zusätzlich auf PowerPC-, MIPS-RISC- und Alpha-RISC-Systemen. Die Unterstützung dieser zusätzlichen Systeme wurde jedoch bereits mit Service Packs (Updatepakete) für Windows NT 4 teilweise wieder aufgegeben und Windows XP erschien schließlich nur noch für x86-kompatible Hardware. Derzeit werden mit Windows 7 Gerüchte verbreitet, dass eine Version für die ARM-Plattform geplant sei. Entwickler haben bereits mehrfach betont, dass es aufgrund der Portabilität und HAL möglich wäre, Windows 7 auf dieser Hardware zu portieren. Intel konnte sich jedoch mit seiner Plattform am Markt sicher etablieren. Der Konkurrent AMD baute selbst x86-kompatible Prozessoren. Dennoch benötigen verschiedene x86-kompatible Systeme teils eine verschiede Windows HAL, diese hängen von der Anzahl der Prozessor-Kerne und den Power-Management-Funktionen (ACPI: Advanced Configuration and Power Interface) des Bios ab.

Windows 2000 und Windows XP wird mit folgenden HALs ausgeliefert:

- Standard-PC, Nicht-ACPI-PIC-HAL (Hal.dll)
- ACPI-PIC-HAL (Halacpi.dll)
- MPS-Uniprozessor-PC, Nicht-ACPI-APIC-UP-HAL (Halapic.dll)
- MPS-Multiprozessor-PC, Nicht-ACPI-APIC-MP-HAL (Halmps.dll)
- ACPI-Uniprozessor-PC, ACPI-APIC-UP-HAL (Halaacpi.dll)
- ACPI-Multiprozessor-PC, ACPI-APIC-MP-HAL (Halmacpi.dll)

Die für den jeweiligen PC benötigte HAL (C:\Windows\System32\hal.dll) wird bei der Windows Installation vom Setup-Programm automatisch erkannt und installiert. Eine passende Dateiversion wird eingespielt und Registry-Einträge werden geschrieben. Ein beachtlicher Nachteil vor Windows Vista war, dass die HAL nach der Installation offiziell nicht mehr geändert werden konnte, was bei bestimmten Hardwareaufrüstungen (z. B. Prozessor-Upgrade oder neue Hauptplatine) eine Neuinstallation erforderlich machte. Auch das Klonen eines ganzen Systems auf andere Hardware wird durch diese **HAL-Abhängigkeit** zur ernsthaften Tortur und ist nicht ohne spezielles Eingreifen einer meist teuren Deployment Software machbar.

Das Problem wurde jedoch bei Windows Vista und Windows 7 endlich gelöst
Eine neue Integration der HAL in diese Betriebssysteme, ermöglicht es in der ersten Startphase HAL-unabhängig zu sein. Das Betriebssystem reagiert selbstständig auf die Hardwareänderungen und kann noch vor dem eigentlichen Windows Start die HAL passend austauschen. Diese neue Funktionalität ist einer der größten Vorteile von Windows Vista und Windows 7. Sie hilft Kunden, welche Windows auf mehreren inhomogenen Rechnern bereitstellen möchten, enorm.

5.14. Die Systemdienste (Executive) und ihre Manager

Die Executive mit ihren **Managern** läuft im **Kernelmodus** und verwaltet die Systemressourcen. Sie stellt mit Hilfe festgelegter Schnittstellen diverse Systemdienste den oberen Schichten bereit, also den Subsystemen und ist in einzelnen Managern organisiert, von denen einige im folgenden Teil dargestellt werden. Das Managerprinzip erleichtert die Austauschbarkeit von Komponenten.
Bei den einzelnen Managern handelt es sich jedoch nicht um separate Prozesse oder Threads sondern um **Module**.

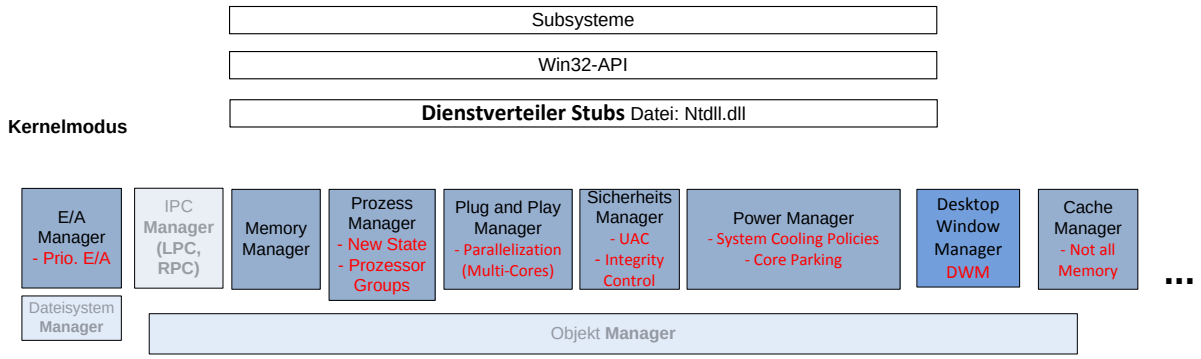


Abbildung 41: die Executive unter Windows 7

Die Verbindung zwischen den Subsystemen (öffentliche Win32-API) und der Executiven erfolgt über **Dienstverteiler Stubs**. Diese sind in der Lage die Systemdienste in Anspruch zu nehmen. Deren Aufruf ist undokumentiert und direkt nicht möglich. Ein konkretes Beispiel des Ablaufes ist unter Abbildung 57: Dienstverteiler Stubs um Zugang zu den Systemdiensten zu bekommen, zu finden.

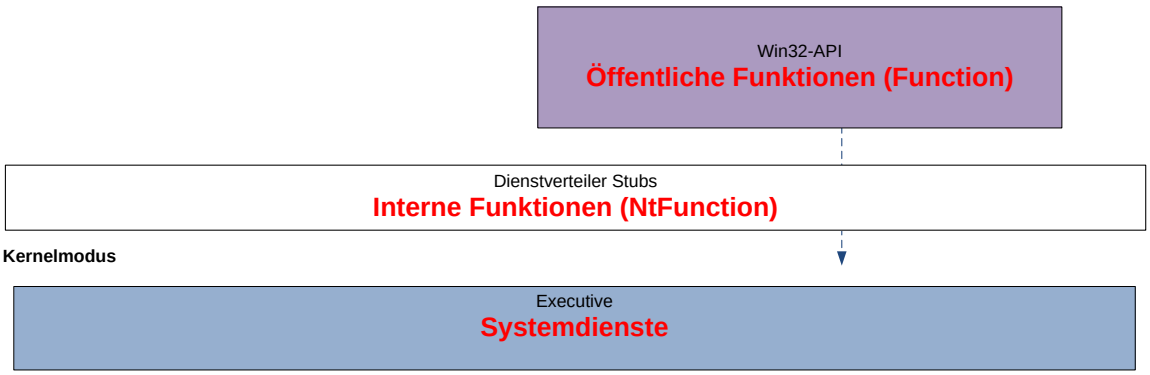


Abbildung 42: Dienstverteiler Stubs

5.15. E/A Manager und Treiber

Das **E/A System** besteht aus einzelnen Systemkomponenten, welche die Ein- und Ausgaben der Hardware steuern. Das E/A System wurde für Uni- als auch auf Multiprozessorsystemen optimiert. Eine Komponente des E/A Systems ist der **E/A Manager**. Er ist zuständig für die Organisation von Ein- und Ausgabe auf verschiedene Geräte. Eine Unterfunktion ist der Dateisystem Manager, der Zugriffe auf Speichermedien wie Festplatten, Bandlaufwerke oder Netzwerk-Freigaben verwaltet.

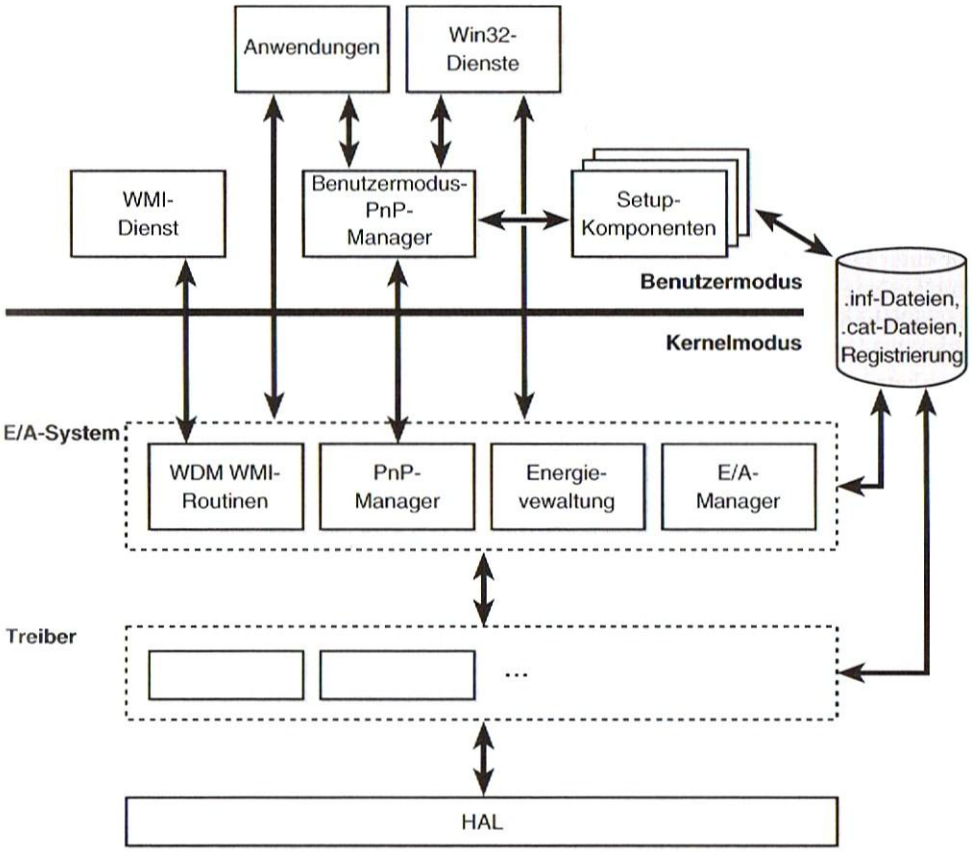


Abbildung 43: Komponenten des E/A Systems

Die Aufgabe des E/A Managers ist es Anwendungen und Systemkomponenten zu virtuellen, logischen und physikalischen Geräten zu verbinden. Er definiert die zentrale Infrastruktur zur Unterstützung der **Gerätetreiber**. Gerätetreiber verwenden den E/A Manager, um E/A-Befehle an andere Gerätetreiber weiterzuleiten.

Gerätetreiber sind systemnahe Programme, die für die Kommunikation zwischen dem System und der jeweiligen Hardware verantwortlich sind. E/A-Funktionsaufrufe von Benutzermodus-Programmen werden in E/A-Anforderungen für ein bestimmtes Gerät übersetzt. Treiber könne direkt auf die Hardware zugreifen (Kernmodus) und sind daher in der Lage, das komplette System abstürzten, zu lassen. Daher werden von Windows nur zertifizierte Treiber automatisiert installiert. Bei nicht zertifizierten muss der Benutzer explizit zustimmen, da er somit ein Risiko eingeht, sofern der Treiberhersteller einen Bug implementiert hat.

Eine detaillierte Funktionsbeschreibung des E/A Systems findet man im Buch Inside Microsoft Windows 2000 von David A. Solomon und Mark Russinovich ab Seite 447.

Priorisierte E/A

Neben den normalen Funktionen zum Öffnen, Schließen, Lesen und Schreiben stellt der E/A Manager auch weitere Funktionen bereit. Eine neue Funktion, die erstmals mit Windows Vista vorgestellt wurde, nennt sich Priorisierte E/A.

E/A Hintergrundaktivitäten wie die Indexierungen von Dateien, Virenüberprüfungen und die Defragmentierung bekommen jeweils eine niedrigere E/A Priorität zugewiesen, damit sie von dem E/A Manager weniger stark als normale Aktivitäten berücksichtigt werden. Das führt dazu, dass die durch den Anwender ausgelöste Festplattenaktivität seines Anwendungsprogramms bevorzugt abgearbeitet wird und er flüssiger arbeiten kann, da nicht auf Hintergrundaktivitäten mit niedrigerer E/A gewartet werden muss.

Windows 7 unterstützt 5 Prioritätsstufen und somit mehr als Windows Vista, das nur 4 unterstützt:

- Kritisch: Wird vom Speicher Manager für das Auslagern von Dateien verwendet.
- Hohe: Kann erst ab Windows 7 für kritische Anwendungen verwendet werden.
- Normal: Standard Priorität für Anwendungen und Dienste
- Minimal: Priorität für geplante Aufgaben, die im Hintergrund ausgeführt werden.
- Sehr niedrig: Hintergrund Aktivitäten wie Indexdienste und Defragmentierungs-Software.

Der Windows-Taskplaner verwendet die Priorität Minimal. Alle Windows 7 Hintergrundaktivitäten verwenden die Priorität Sehr niedrig. Dazu gehört der Windows Defender, die Defragmentierung und die Windows Suche. Implementiert ist die Steuerung des priorisierten E/As in der Datei Classpnp.sys. Sie setzt die Prioritäten und übergibt diese dem E/A Manager. Weitere Informationen findet man unter dem Link: http://technet.microsoft.com/en-us/magazine/cc162494.aspx .

5.16. Dateisystem Manager

- Unterfunktion des E/A Managers
- Steuert den Zugriff auf die Speichermedien (Festplatten, Wechseldatenträger, Netzwerkfreigaben).
- Benötigt die Festplatten- und Netzwerkkartentreiber.

Auf die Funktionsweise wird hier nicht näher eingegangen. Eine detaillierte Funktionsbeschreibung des Dateisystem Managers findet man im Buch Inside Microsoft Windows 2000 von David A. Solomon und Mark Russinovich ab Seite 447. Das Dateisystem NTFS ist des Weiteren unter dem folgenden Link beschrieben: http://wwwbs.informatik.htw-dresden.de/svortrag/ai95/Zschiedrich/v010.htm .

5.17. IPC[24] Manager

- Verarbeitet die gesamte Kommunikation zwischen den Prozessen.
- Lokal: LPC (Lokal Procedure Call)
- Remote: RPC (Remote Procedure Call)

[24] Unter Interprozesskommunikation (englisch inter-process communication, IPC) versteht man Methoden zum Informationsaustausch, informatisch gesprochen Datenübertragung, von nebenläufigen Prozessen oder Threads.

Er verarbeitet die gesamte Kommunikation zwischen den Prozessen. Diese Kommunikation kann lokal über den LPC (Lokal Procedure Call) erfolgen oder mit Prozessen auf anderen Rechnern via RPC (Remote Procedure Call). Auf die Funktionsweise wird hier nicht näher eingegangen. Eine detaillierte Funktionsbeschreibung des IPC Managers findet man im Buch Inside Microsoft Windows 2000 von David A. Solomon und Mark Russinovich.

5.18. Memory Manager

- Virtual Memory Management (VMM)
- Bereitstellung eines eigenen virtuellen Adressraum für jeden Prozess
- Gegenseitiges Absichern der verschiedenen Adressräume
- Verwaltet das Paging (Auslagerungsdatei).

Die Aufbau der Speicherverwaltung wird im Kapitel 5.11 beschrieben. Das Paging wird in Kapitel 5.12 beschrieben.

5.19. Prozess Manager

- Erzeugt Prozesse und Threads
- Verwalten der Prozesse im System
- Überwachen der Prozesse

Prozesse und Threads werden in Kapitel 5.8 erklärt.

5.20. Plug-and-Play Manager

- Erkennen von PnP-Geräten
- Einleiten einer Treiberinstallation
- Steuern der Dienste für die Treiber

Auf die Funktionsweise wird hier nicht näher eingegangen. Eine detaillierte Funktionsbeschreibung des Plug-and-Play Manager s findet man im Buch Inside Microsoft Windows 2000 von David A. Solomon und Mark Russinovich ab Seite 447.

5.21. Sicherheits Manager

Aufgaben des Managers:

- Security Reference Monitor (SRM)
- Zentrale Überwachung aller Sicherheitsmechanismen
- Authentifizierung durchführen
- Zugriffsrechte verwalten und überwachen
- Besitzrechte verwalten und überwachen
- Objekte, die überwacht werden: Benutzer, Dateien, Registry-Schlüssel, Prozesse, Threads, Pipes und weitere.

Die ursprüngliche Implementierung ist unter http://www.s-line.de/homepages/trac/wissen/datensicherheit/nt-sicherheitssystem.html oder http://wwwbs.informatik.htw-dresden.de/svortrag/ai95/Apfelboeck/ntsicherheit.htm beschrieben. Seit Windows 2000 wurden große Änderungen durchgeführt, die ab der 3. Auflage in Inside Microsoft Windows 2000 von David A. Solomon und Mark Russinovich erklärt werden.

Access Token (Prozesse)
Jeder Prozess (ein Container aus Threads) in Windows verfügt über einen eigenen **Access Token**. In der folgenden Abbindung ist der Aufbau des Tokens dargestellt:

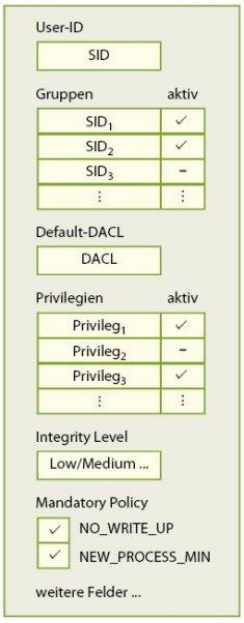


Abbildung 44: Das Access Token in Windows

Das Token enthält Informationen darüber, was der Prozess alles tun darf. Es enthält einen Kontext, der festlegt, unter welchem Benutzer der Prozess und die darin enthaltenen Threads ausgeführt werden. Der Benutzer und eventuelle Gruppenzugehörigkeiten werden über **Security-IDs (SIDs)** zugeordnet. Des Weiteren ist ein Satz von Privilegien definiert, die bestimmte Rechte einräumen. Mit dem Befehlsprogramm **whoami /all,** kann das Token des aktuellen Kontextes (aktueller Benutzer) ausgelesen werden.

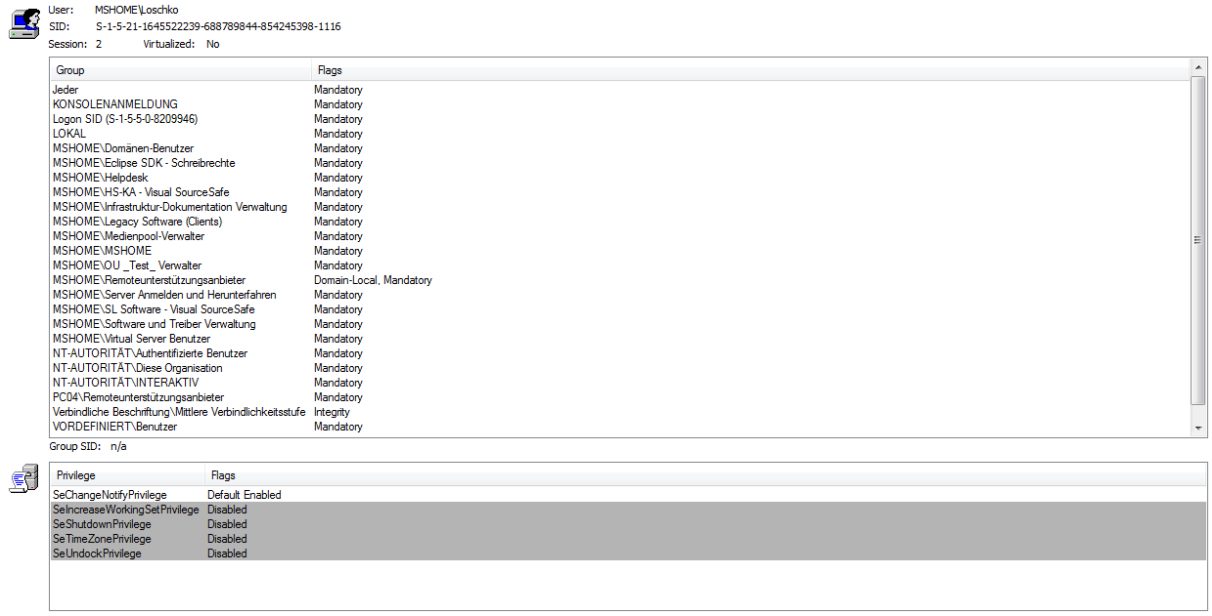


Abbildung 45: Der Prozess von Microsoft Word unter dem Benutzerkontext MSHOME\Loschko

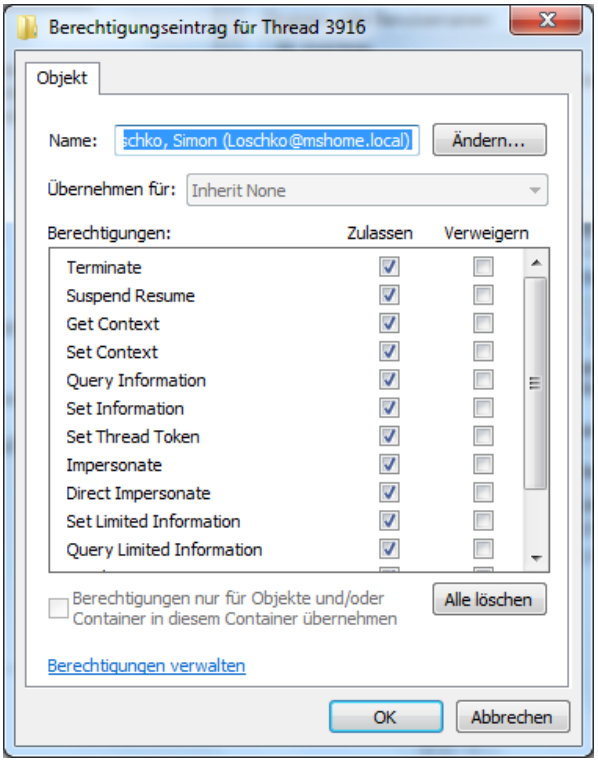


Abbildung 46: Ein einzelner Thread von Microsoft Word und seine Berechtigungen

Security Descriptor (Objekte)
Jedes der Zugriffskontrolle unterliegende Objekt im System verfügt über einen **Security Descriptor**, der darüber bestimmt, wer was mit diesem Objekt tun darf. Er besteht aus der SID des Besitzers und eine Liste mit Zugriffsrechten, die **Discretionary Access Control List (DACL)**. Die DACL besteht aus mehreren Einträgen, die jeweils einer SID (Security-ID) bestimmte Rechte zugestehen (Erlauben) oder entziehen (Ablehnen). Wenn ein Prozess auf ein Objekt zugreifen möchte, geht Windows die DACL im Security Descriptor des Objekts Eintrag für Eintrag durch und vergleicht die darin aufgeführten SIDs mit den im Access Token des Prozesses gespeicherten. Sobald es eine Ablehnenregel findet, die zu einer der SIDs im Access Token passt, lehnt Windows den Zugriff ab. Wenn nach der kompletten Abarbeitung der ganzen Liste keine Erlaubenregel gefunden wurde, wird auch kein Zugriff gestattet.

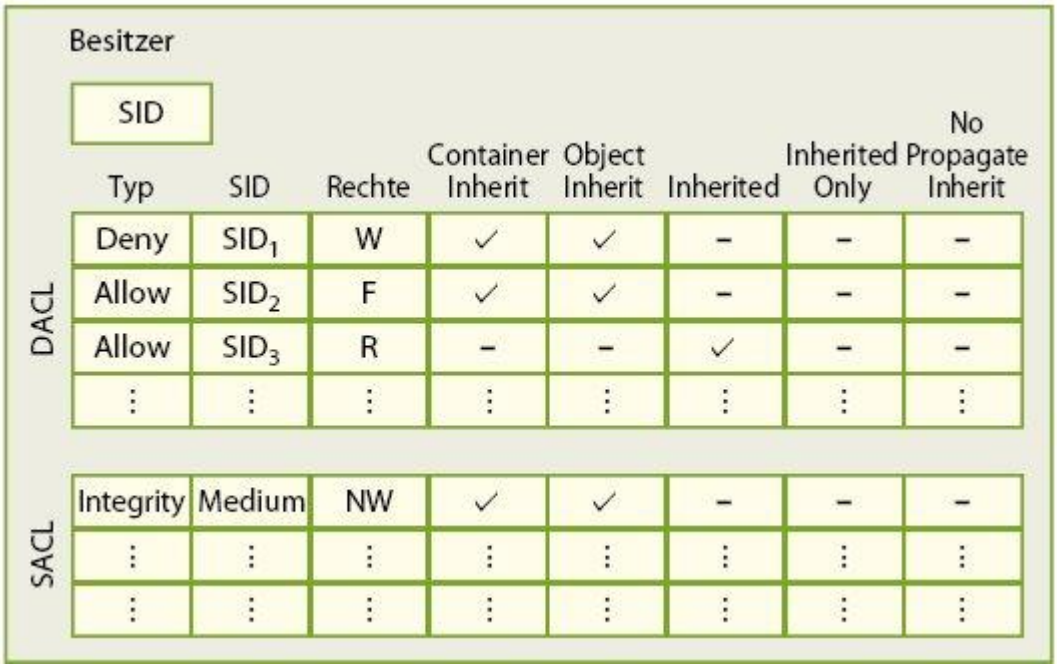


Abbildung 47: Der Security Descriptor in Windows 7

Ein Administrator ist unter Windows in der Lage die DACL jederzeit zu verändern, da er den Besitz des Objekts übernehmen kann. Er kann Änderungen an der Rechtevergabe durchführen, da er in seinem Access Token das Privileg **SeTakeOwnership** mitführt und dieser eine Besitzübernahme von Objekten ermöglicht.

Mandatory Integrity Control (Verbindlichkeitsstufe)
Das Modell aus Access Token und Security Descriptor eignet sich nicht gut, um zur Vorbeugung gegen Sicherheitslücken gezielt die Rechte eines Prozesses wie des Internet Explorer einzugrenzen. Denn solange der Prozess unter der User-ID des angemeldeten Benutzers läuft, lässt sich der Zugriff auf Objekte die diesem Benutzer zugeordnet sind nicht verhindern. Daher gibt es seit Windows Vista und auch in Windows 7 einen weiteren Sicherheitsmechanismus, der Vorrang vor dem Access Token und Security Descriptor hat. Folgende vier Stufen wurden mit Windows Vista eingeführt:

Low:	Internet Programme
Medium:	Programme mit normalen Benutzerrechten
High:	Administrative Tätigkeiten
System:	Betriebssystemdienste

Die Stufe eines Objekts ist in dessen Security Descriptor in der **System Access Control List (SACL)** festgelegt. Prozesse, die auf einer niedrigeren Stufe laufen, können Objekten einer höherer Stufe nicht beeinflussen. Zugriffe von unten nach oben sind daher beschränkt, während auf gleicher Ebene oder von oben nach unten alles erlaubt ist – solange die **Discretionary Access Control List (DACL)** das nicht in noch blockiert. Bei der Access Control List (SACL) genügt es nicht, der Besitzer eines Objekts zu sein, um sie zu ändern.

Die SACL und DACL können mit dem Befehlsprogramm **icacls.exe** verwaltet werden. Die Flags haben folgende Bedeutung:

Einfache Rechten:
N - Kein Zugriff
F - Vollzugriff
M - Änderungszugriff
RX - Lese- und Ausführungszugriff
R - Schreibgeschützter Zugriff
W - Lesegeschützter Zugriff
D - Löschzugriff

Bestimmten Rechten:
DE - Löschen
RC - Lesesteuerung
WDAC - DAC schreiben
WO - Besitzer schreiben
S - Synchronisieren
AS - Systemsicherheitszugriff
MA - Maximal zulässig
GR - Allgemeiner Lesezugriff
GW - Allgemeiner Schreibzugriff
GE - Allgemeiner Ausführungszugriff
GA - Allgemeiner Zugriff (alle)
RD - Daten lesen/Verzeichnis auflisten
WD - Daten schreiben/Datei hinzufügen
AD - Daten anfügen/Unterverzeichnis hinzufügen
REA - Erweiterte Attribute lesen
WEA - Erweiterte Attribute schreiben
X - Ausführen/Durchsuchen
DC - Untergeordnetes Element löschen
RA - Attribute lesen
WA - Attribute schreiben

Vererbungsrechte
Werden nur auf Verzeichnisse angewendet:
OI - Objektvererbung
CI - Containervererbung
IO - Nur vererben
NP - Vererbung nicht propagieren
I - Vom übergeordneten Container vererbte Berechtigung

Abbildung 48: Flags der Berechtigungen in Windows 7

```
c:\
   VORDEFINIERT\Administratoren:(F)
   VORDEFINIERT\Administratoren:(OI)(CI)(IO)(F)
   NT-AUTORITÄT\SYSTEM:(F)
   NT-AUTORITÄT\SYSTEM:(OI)(CI)(IO)(F)
   VORDEFINIERT\Benutzer:(OI)(CI)(RX)
   NT-AUTORITÄT\Authentifizierte Benutzer:(OI)(CI)(IO)(M)
   NT-AUTORITÄT\Authentifizierte Benutzer:(AD)
   Verbindliche Beschriftung\Hohe Verbindlichkeitsstufe:(OI)(NP)(IO)(NW)

c:\Program Files
            NT SERVICE\TrustedInstaller:(F)
            NT SERVICE\TrustedInstaller:(CI)(IO)(F)
            NT-AUTORITÄT\SYSTEM:(M)
            NT-AUTORITÄT\SYSTEM:(OI)(CI)(IO)(F)
            VORDEFINIERT\Administratoren:(M)
            VORDEFINIERT\Administratoren:(OI)(CI)(IO)(F)
            VORDEFINIERT\Benutzer:(RX)
            VORDEFINIERT\Benutzer:(OI)(CI)(IO)(GR,GE)
            ERSTELLER-BESITZER:(OI)(CI)(IO)(F)
```

Abbildung 49: Beispiele für Berechtigungen bei Windows 7

5.22. Power Manager

- Kommunikation mit der Hardware um Energiesparfunktionen zu aktivieren bzw. zu deaktivieren
- ACPI
- Batterieüberwachung

5.23. Cache Manager

Aufgaben des Managers:

- Zuletzt gelesene Dateien werden im Systemspeicher zwischengespeichert, um diese bei einem erneuten Zugriff schneller bereitstellen zu können
- Enge Zusammenarbeit mit dem Memory Manager

Der Cache Manager spielt eine zentrale Rolle, ohne die das System überhaupt nicht arbeiten würde. Er ist besonders essenziell, damit andere Manager ihren Dienst verrichten können. Dateioperationen im lokalen Dateisystem auf den Wechseldatenträger und im Netzwerk über das CIFF bzw. SMB-Protokoll laufen mit seiner Hilfe ab und bringt enorme Geschwindigkeitsvorteile. Grund hierfür ist, dass ständig benötigte Daten aus dem schnellen RAM schneller gelesen werden, anstatt sie von der langsamen Festplatte zu holen. Unter Windows Vista war eine wesentliche Neuerung, dass fast der gesamte nicht durch Threads bzw. Prozesse genutzte Arbeitsspeicher dem Cache Manager zur Verfügung stand, solange der Speicher nicht durch einen Thread bzw. Prozess angefordert wurde. Der Cache Manager nutze diesen freien Arbeitsspeicher. Dieses Prinzip wurde jedoch mit Windows 7 total verworfen. Die Anwender beklagten sich unter Vista, dass das Betriebssystem, den ganzen Arbeitsspeicher an sich reißen würde. Diverse Statusprogramme zeigten an, dass kein freier Speicher verfügbar ist. Grund hierfür ist der Cache Manager, der alles nutzte, da es andere Programme ja nicht anforderten. Da dieses Verhalten für Kritik sorgte und für den Endanwender nicht nachvollziehbar war, wurde bei Windows 7 das Verhalten geändert.

Der Cache Manager arbeitet ähnlich wie Windows XP mit einem gewissen Teil des Arbeitsspeichers und passt sich dynamisch an die Speicherverhältnisse an, nutzt aber nicht mehr den ganzen freien Speicher.

Die grundsätzliche Arbeitsweise ist relativ einfach. Beim Lesen füllt Windows den Cache. Der freie Hauptspeicher nimmt dabei stetig ab. Kommt währenddessen eine Speicheranforderung, rückt der Cache noch keinen Speicher heraus. Stattdessen lagert Windows andere Speicherinhalte aus und füllt dabei mitunter sogar die Auslagerungsdatei zusätzlich. Erst wenn das Lesen beendet ist, wird der Cache verkleinert und Arbeitsspeicher frei.

5.24. Desktop Window Manager: Win32-GDI unter Windows 7

- Fensterverwaltung- und Grafiksystem
- GUI-Funktionen (USER- und GDI-Funktionen von Win32)
- Steuerelemente, Zeichenobjekte, Buttons...

Seit Windows 7 nutzt das **Graphics Device Interface (GDI)**. Verbessert wurde die synchrone Kommunikation mehrerer Anwendungen mit der GDI, um mit Windows 7 eine schnellere Reaktionszeit zu ermöglichen. Zusätzlich wird auch der Arbeitsspeicher des Systems effizienter für die GUI-Darstellung verwendet, was besonders für Grafiklösungen mit Shared Memory ein großer Vorteil ist. In Vista verbrauchte jedes Fenster zweimal Speicher, da sowohl im Video- als auch im Systemspeicher von identischem Inhalt Platz belegt wurde. In Vista wurden die Aufgaben des **Desktop Window Manager** von der GPU abgearbeitet. Gleichzeitig wurden die Daten für die CPU dupliziert, sodass im Grunde ein Inhalt doppelt Platz benötigte. Windows 7 benötigt nun die Daten nur noch einmal und zwar im Grafikspeicher und muss kein Duplikat im Systemspeicher mehr erstellen. Der Speicherverbrauch ist nun konstant, da der Grafikkartenspeicher immer voll ausgeschöpft wird und keine Kopien im Arbeitsspeicher liegen.

Unterstützt wird diese neue Art des Speicherzugriffs von den neuen WDDM-1.1.-Treibern. Ältere WDDM-1.0.-Treiber bleiben zwar zu Windows 7 kompatibel, unterstützen die neue Technik aber nicht. Obwohl die neue Zugriffsmethode in der Praxis Vorteile hat, kann es auch zu Nachteilen auf älterer Hardware kommen, da die CPU sich nun Daten aus dem Video-RAM ziehen muss und das auf älteren System eventuell durch den langsameren Bus schwerfälliger ist.

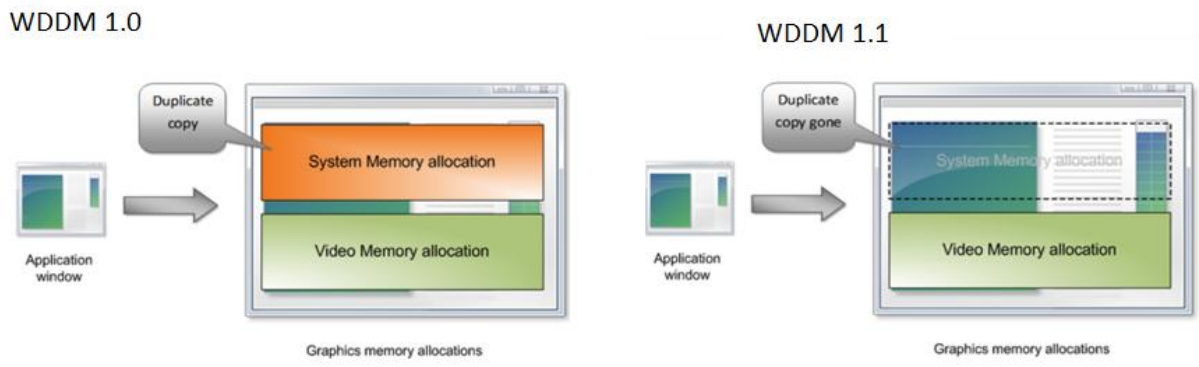


Abbildung 50: Graphics Device Interface (GDI) mit WDDM 1.1 unter Windows 7

Weitere Informationen findet man im Internet unter:
http://blogs.msdn.com/e7/archive/2009/04/25/engineering-windows-7-for-graphics-performance.aspx

5.25. Objekt Manager

Die Hauptaufgabe eines Betriebssystems besteht darin, logische und physikalische Ressourcen zu verwalten. In Windows NT repräsentieren Objekte **Systemressourcen**. So existiert z. B. zu jedem Thread ein Thread Objekt und zu jeder geöffneten Datei ein Dateiobjekt welches von außen über ein **Handle** verwendet werden kann. Einem Prozess werden Objekte zugeteilt und erst wenn der Prozess terminiert, oder der Handle von außen freigegeben wird, werden die Objekte wieder gelöscht. Das Betriebssystem muss Schutzmechanismen anwenden wenn Prozesse um Systemressourcen konkurrieren. Dabei sind diese Mechanismen für die verschiedenen Systemressourcen die identischen. Anders als in der UNIX-Welt, dort werden die verschiedenen Ressourcen in ein Dateisystem integriert, existiert unter Windows ein **zentraler Objekt Manager** der alle Systemressourcen unter einem einheitlichen Namensraum verwaltet. Die Designziele des Objekt Managers wurden erreicht: Ein einheitlicher Mechanismus für alle Systemressourcen, um diese über das gesamte Betriebssystem hinweg abbilden zu können, damit eine leichte Erweiterbarkeit und Portierbarkeit sowie Zuverlässigkeit und Robustheit gesichert ist.

Kurz und knapp:

- Der Objekt Manager verwaltet Objekte und übernimmt damit einen Großteil der Verwaltung von Systemressourcen.
- Er hat eine zentrale Funktion und befindet sich in der Executiven.
- Die abstrakten Objekte repräsentieren Systemressourcen wie z. B. Threads.
- Der Zugriff von außen erfolgt über Handles.
- Die interne Datenstruktur von Objekten bleibt verborgen.
- Dennoch ist Windows 7 kein objektorientiertes Betriebssystem.

Folgende Objekte werden vom Objekt Manager verwaltet:

- **Kernelobjekte** repräsentieren ausschließlich Ressourcen des Kernels und durchlaufen nicht den Objekt Manager um keinen Overhead, für das dafür nötige Durchlaufen der Executiven, zu erzeugen.
- **Kontrollobjekte** dienen rein der Kontrolle verschiedener Betriebssystemfunktionen. Dazu gehören Prozessobjekte, APC-Objekte (Asynchrounous Procedure Call), DPC[25]-Objekte (Deferred Procedure Call), Interrupt-Objekte)
- **Dispatcher Objekte** werden für die Realisierung von Synchronisationsfunktionen beansprucht. Dazu gehören Thread Objekte, Mutex Objekte, Semaphor Objekte, Ereignis Objekte, Ereignispaar Objekte, und weitere.

Die meisten Objekte sind nicht im Objekt Manager selbst implementiert, sondern in ihrem dazugehörigen Modul, das sich ebenfalls wie der Objekt-Manager selbst in der Executiven befindet. Der Objekt-Manager verfügt ausschließlich über ein Grundgerüst zur Erzeugung, Löschung, Schutz, Aufzeichnung und anderen grundsätzlichen Verwaltungsaufgaben von Objekten. Weitere Objektmethoden werden durch das zum Objekt gehörende Modul bzw. Subsystem ergänzt.

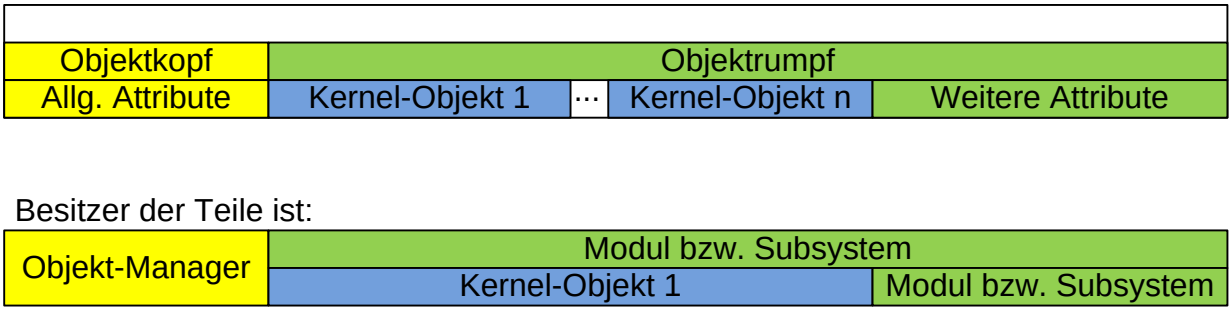


Abbildung 51: Aufbau von Objekten

[25] DPCs sind Interrupts, die mit niedrigerer Priorität als Standardinterrupts ausgeführt werden

Der Objektkopf speichert allgemeine Attribute, auf die nur der Objekt-Manager einen Zugriff hat. Der optionale gefüllte Objektrumpf kann Kernel-Objekte enthalten, die ausschließlich von Kernelfunktionen manipuliert werden können. In den weiteren Attributen können spezifische Informationen bestimmter Module oder Subsysteme abgelegt werden.

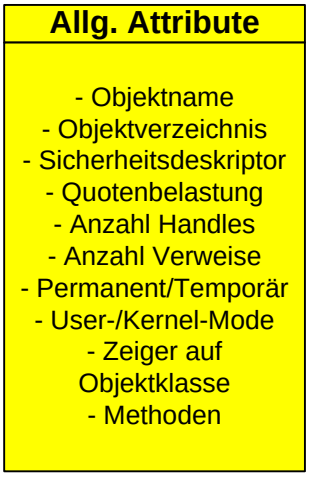


Abbildung 52: Die allgemeinen Attribute eines Objekts

Alle Objekte mussten an einer zentralen Stelle, dem Objekt-Manager verwaltet werden damit die NT-Architektur die hohen Anforderungen für die C2-Zertifizierung erfüllen konnte. C2 verweist an eine Reihe Sicherheitsrichtlinien, die definieren, wie ein sicheres System funktioniert um die Sicherheitsanforderungen des amerikanischen Verteidungsministeriums zu erfüllen.

Viele objektorientierte Ansätze sind in der Windows NT Architektur enthalten. Beispielsweise greifen die meisten Komponenten nicht auf die Datenstrukturen anderer Kernel-Komponenten zu, wenn sie Daten benötigen, die von einer andern Komponente verwaltet werden. Sie verwenden formale Schnittstellen, um Parameter zu übergeben und auf Datenstrukturen zuzugreifen und diese zu verändern. Es handelt sich dennoch nicht um ein objektorientiertes Betriebssystem, da das Prinzip der Datenkapselung[26] in Teilen verletzt wurde und zum Anderen unterstützt die Programmiersprache C, aus der Windows NT überwiegend besteht, keine objektorientierten Konstrukte. Grund für wenige direkte Zugriffe war, dass diese die Performance ungemein steigerten im Gegensatz zu den indirekten Zugriffen.

Der Objekt-Manager kennt verschiedene **Typenobjekte** (welche die gleichen Systemressourcen repräsentieren) und packt diese in Objektklassen. Alle Objekte dieser Klasse besitzen bestimmte identische Attribute.

[26] Datenkapselung ermöglicht das Abschotten der internen Implementierung vor direkten externen Zugriffen

Modul bzw. Subsystem	Objektklasse / Typenobjekte	Repräsentierte Ressource
Objekt Manager	Object Type	Typobjekt
	Directory	Verzeichnis im Namensraum
	Symbolic Link	Verweis im Namensraum
Prozess- und Thread Manager	Process	Aktiver Prozess
	Thread	Aktiver Thread
	Token	Sicherheitsprofil eines Prozesses
VMM	Section	Speicher-Mapping
LPC Facility	Port	Kommunikationskanal
E/A Manager	File	Geöffnete Datei
	IO Completion	"Beendet"- Mitteilung
	Adapter	DMA Ressource
	Controller	DMA Contoller
	Device	Logisches oder physisches Gerät
	Driver	Gerätetreiber
Executive	Event	Synchronisationsprimitive
	Event Pair	
	Mutant	
	Semaphore	
	Timer	

Abbildung 53: Typenobjekte des Objekt-Managers

Was sind genau die Handles?

Die Umgebungssubsysteme wie z. B. die Win32-API von Windows stellen teils eigene Objekte zur Verfügung, die auf den Objekten der Executiven aufbauen. Der Zugriff auf die Objekte der Executiven durch den Anwendungsentwickler erfolgt indirekt über **Handles**. Falls das Objekt der Executiven im Objektrumpf ein Kernel-Objekt enthält, wird diesen an den Kernel exportiert. Dieser wird vom Objekt-Manager veranlasst.

Zugeordnete Handles — Handles durchsuchen

Gefiltert von "notepad.exe"

Abbild	PID	Typ	Handlename
notepad.exe	1272	Desktop	\Default
notepad.exe	1272	Directory	\KnownDlls
notepad.exe	1272	Directory	\Sessions\1\BaseNamedObjects
notepad.exe	1272	File	C:\Users\Loschko
notepad.exe	1272	File	C:\Windows\winsxs\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.7000.0_none_a67ecbb245429e4d
notepad.exe	1272	File	\Device\KsecDD
notepad.exe	1272	File	C:\Windows\Fonts\StaticCache.dat
notepad.exe	1272	File	C:\Windows\winsxs\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.7000.0_none_a67ecbb245429e4d
notepad.exe	1272	Key	\REGISTRY\MACHINE\SYSTEM\ControlSet001\Control\Nls\Sorting\Versions
notepad.exe	1272	Key	\REGISTRY\MACHINE
notepad.exe	1272	Key	\REGISTRY\MACHINE\SYSTEM\ControlSet001\Control\SESSION MANAGER
notepad.exe	1272	Key	\REGISTRY\MACHINE\SYSTEM\ControlSet001\Control\Nls\Locale
notepad.exe	1272	Key	\REGISTRY\USER\S-1-5-21-1645522239-688789844-854245398-1116
notepad.exe	1272	Key	\REGISTRY\MACHINE\SYSTEM\ControlSet001\Control\Nls\Locale\Alternate Sorts
notepad.exe	1272	Key	\REGISTRY\MACHINE\SYSTEM\ControlSet001\Control\Nls\Language Groups
notepad.exe	1272	Mutant	\Sessions\1\BaseNamedObjects\MSCTF.Asm.MutexDefault1
notepad.exe	1272	Section	\Sessions\1\BaseNamedObjects\windows_shell_global_counters
notepad.exe	1272	WindowStation	\Sessions\1\Windows\WindowStations\WinSta0
notepad.exe	1272	WindowStation	\Sessions\1\Windows\WindowStations\WinSta0

Abbildung 54: Handles des Editors (notepad.exe) unter Windows 7

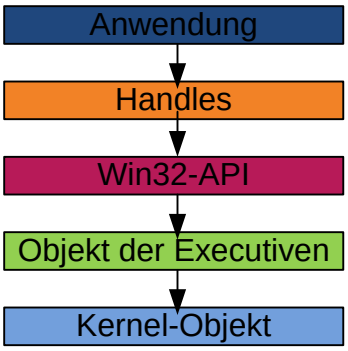


Abbildung 55: Zugriff über Handles auf die Executive

6. Anlagen

6.1. Nutzung der Systemdienste (Executive) durch eine Anwendung

Im folgenden Beispiel (Abbildung 56: Eine Anwendung erstellt eine Datei und Abbildung 57: Dienstverteiler Stubs um Zugang zu den Systemdiensten zu bekommen) erstellt eine Windows-Anwendung eine Datei auf der Festplatte und muss dazu das Betriebssystem auffordern, diese Datei zu erstellen und Zugriff darauf zu gewähren:
Die Anwendung verwendet dazu eine Funktion der Win32-API. Beim Aufruf der Win32-API-Funktion **CreateFile** erfolgt ein Sprung (Trap) in den Kernel-Modus. Die Systemdienste rufen die interne Systemroutine **NTCreateFile** auf, die daraufhin dem Festplattentreiber die Anweisung übergibt, die Datei auf die Festplatte zu schreiben. Es erfolgt erneut ein Trap. Nun erhält der Anwendungsprozess ein Dateihandle zurück und hat somit für weitere zukünftige Aufrufe Zugriff auf die Datei. Die interne Systemroutine NTCreateFile ist für die Öffentlichkeit undokumentiert.

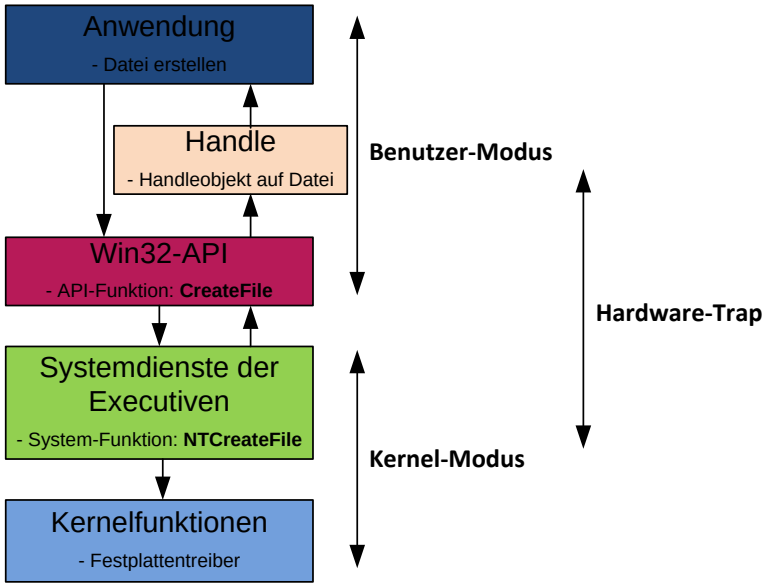


Abbildung 56: Eine Anwendung erstellt eine Datei

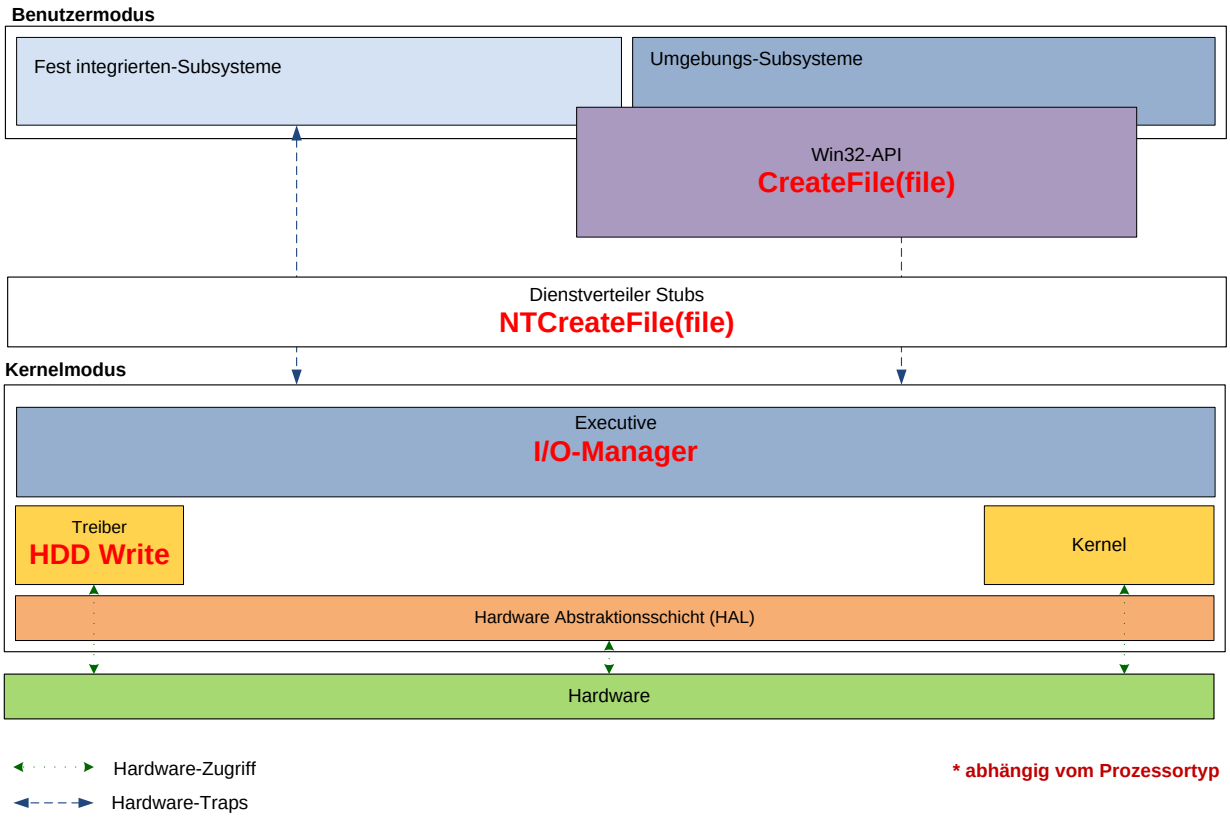


Abbildung 57: Dienstverteiler Stubs um Zugang zu den Systemdiensten zu bekommen

6.2. Die wichtigsten Windows 7 Systemdateien

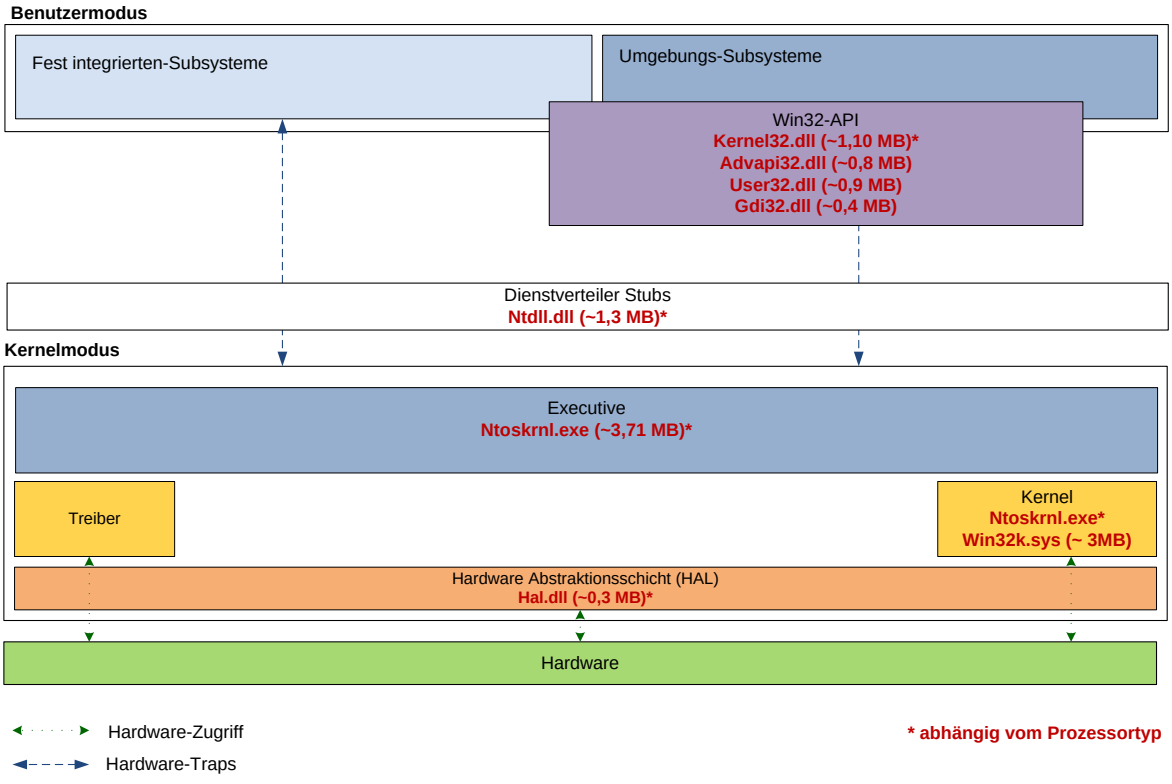


Abbildung 58: Wichtige Systemdateien von Windows 7

6.3. Die Registrierung

In Windows wurde als Nachfolger der Konfigurationsdateien (INI-Dateien) eine zentrale Datenbank eingeführt die Registrierung oder auch Registry genannt wird.

Informationen in der Registrierung:

- Startkonfiguration
- Systemweite Einstellungen
- Systemweite Softwarekonfigurationen
- Sicherheitsdatenbank
- Benutzerspezifische Einstellungen
- Geladene Gerätetreiber
- Zugriffbasis auf Leistungsindikatoren

Value Type	Description
REG_NONE	No value type.
REG_SZ	Fixed-length Unicode string.
REG_EXPAND_SZ	Variable-length Unicode string that can have embedded environment variables.
REG_BINARY	Arbitrary-length binary data.
REG_DWORD	32-bit number.
REG_DWORD_LITTLE_ENDIAN	32-bit number, with low byte first. This is equivalent to REG_DWORD.
REG_DWORD_BIG_ENDIAN	32-bit number, with high byte first.
REG_LINK	Unicode symbolic link.
REG_MULTI_SZ	Array of Unicode NULL-terminated strings.
REG_RESOURCE_LIST	Hardware resource description.
REG_FULL_RESOURCE_DESCRIPTOR	Hardware resource description.
REG_RESOURCE_REQUIREMENTS_LIST	Resource requirements.
REG_QWORD	64-bit number.
REG_QWORD_LITTLE_ENDIAN	64-bit number, with low byte first. This is equivalent to REG_QWORD.
REG_QWORD_BIG_ENDIAN	64-bit number, with high byte first.

Abbildung 59: Werte in der Registry

Root Key	Description
HKEY_CURRENT_USER	Stores data associated with the currently logged-on user
HKEY_USERS	Stores information about all the accounts on the machine
HKEY_CLASSES_ROOT	Stores file association and Component Object Model (COM) object registration information
HKEY_LOCAL_MACHINE	Stores system-related information
HKEY_PERFORMANCE_DATA	Stores performance information
HKEY_CURRENT_CONFIG	Stores some information about the current hardware profile

Abbildung 60: Die Wurzelschlüssel in der Registry

6.4. Übersicht: Die Windows 7 Architektur

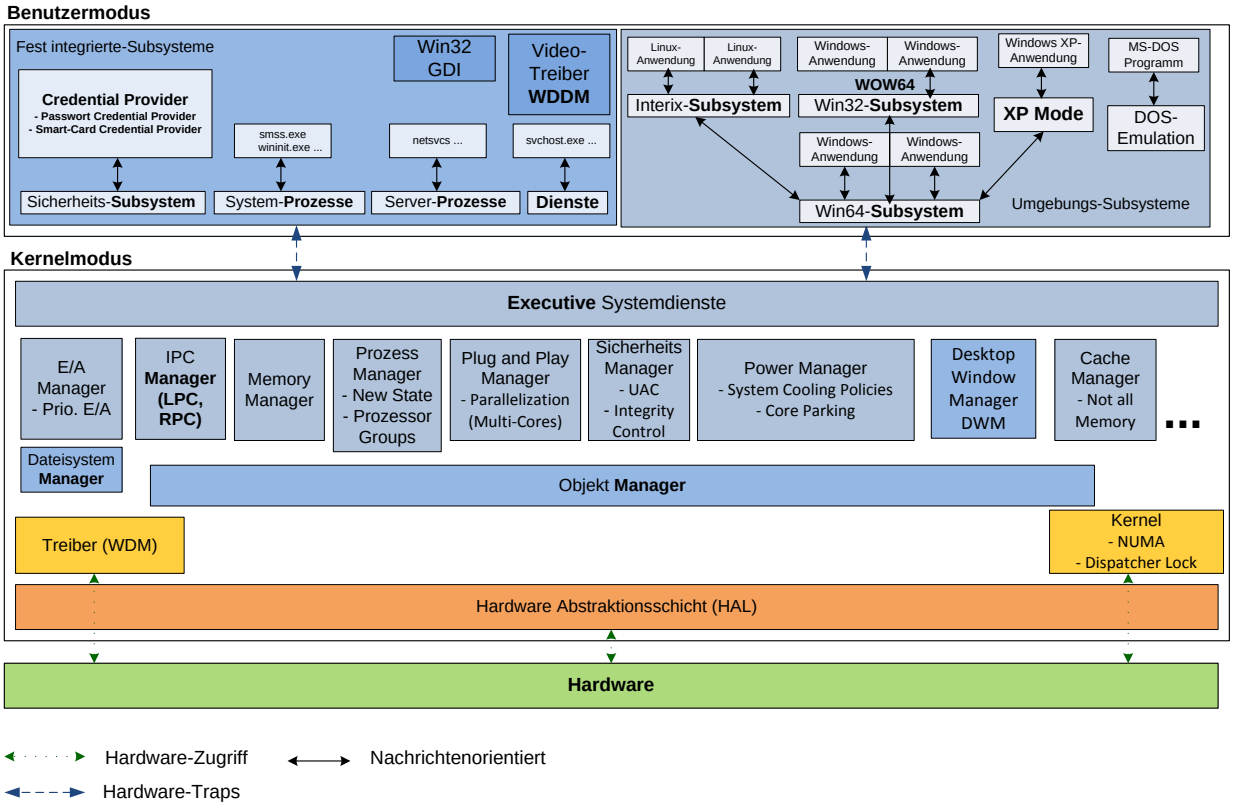


Abbildung 61: Windows 7-Architektur basierend auf dem Windows NT-Betriebssystemmodell

6.5. Übersicht: Die Windows 2000 Architektur

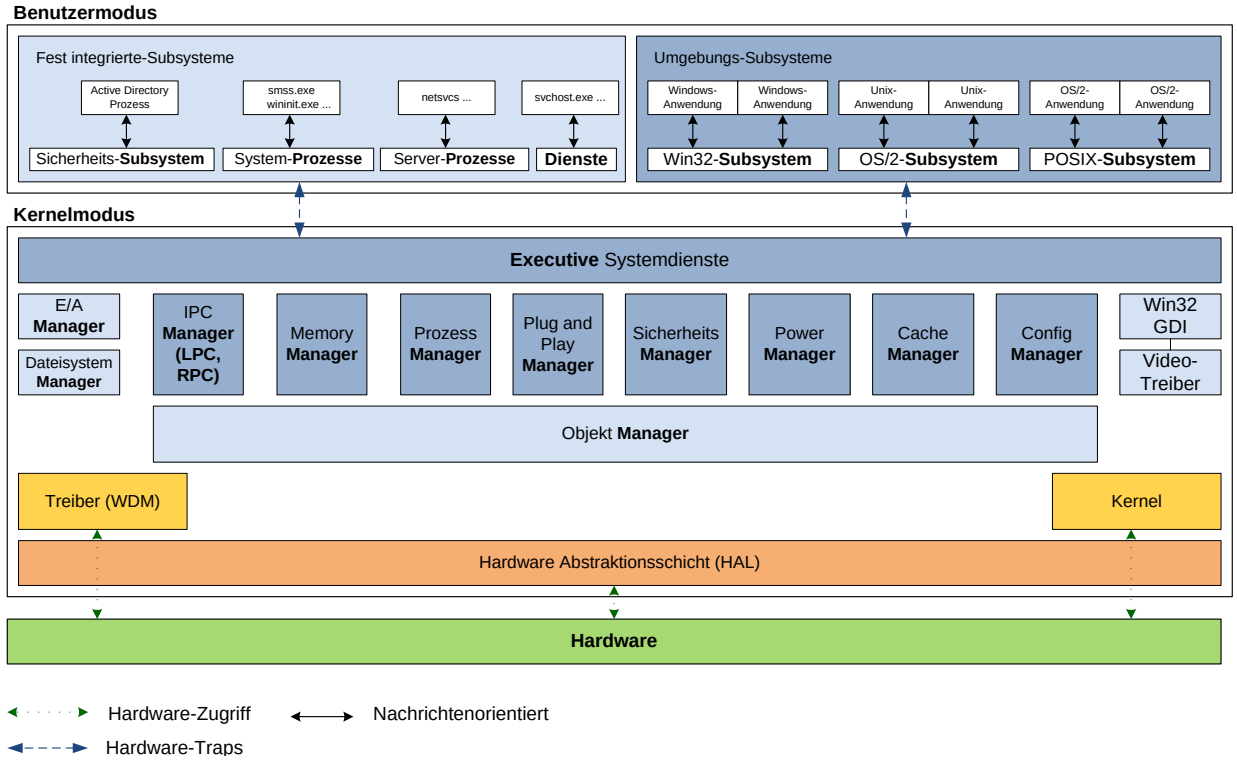


Abbildung 62: Windows 2000-Architektur basierend auf dem Windows NT-Betriebssystemmodell

Abbildungsverzeichnis

Die Abbildungen, die aus einer fremden Quelle entnommen wurden, sind mit einer Quellenangabe versehen:

Literaturverzeichnis

In der Ausarbeitung wurden keine Zitate (wörtlich übernommenen Stellen aus anderen Texten) verwendet.

Ausnahme: Die Fußnoten, welche technische Begriffe erläutern, wurden aus der Wikipedia teilweise wörtlich übernommen. http://www.wikipedia.de

---

Die in den Kapiteln 1, 2 und 3 und 5.3 verwendete Informationen stammen aus den Microsoft Entwickler-Blogs. Die deutschsprachigen Informationen werden von Microsoft Deutschland GmbH - Niederlassung Berlin, Katharina-Heinroth-Ufer 1, D-10787 Berlin bereitgestellt.

http://windowsteamblog.com/blogs/

http://blogs.technet.com/

http://channel9.msdn.com/Default.aspx?wa=wsignin1.0

Viele Informationen basieren zudem auf dem von Daniel Melanchthon (Microsoft Deutschland) veröffentlichen Wissen.

Die in den Kapiteln 5.2, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 5.12, 5.13, 5.14, 5.15, 5.16, 5.17, 5.18, 5.19, 5.20, 5.21, 5.22, 5.23, 5.24, 5.25, 6.1, 6.3 und 6.5 enthaltenen Informationen stammen aus folgender Literatur:

David A. Solomon, M. R. (2000). *Inside Microsoft Windows 2000.* Microsoft Press Deutschland; Auflage: 3. A.

Solomon, D. A. (1998). *Inside Microsoft Windows NT, deutsche Ausgabe.* Microsoft Press Deutschland.

Des Weiteren stammen die aktuellen Informationen zu Windows 7, von Mark Russinovich der über die Microsoft Technet regelmäßig aktuelle Informationen veröffentlicht.

http://blogs.technet.com/markrussinovich/

Das im gesamten Kapitel 4 und Kapitel 5.1 und 5.4 enthaltenen Wissen stammt aus folgenden Quellen:

http://und0cumented.ntinternals.net/tmp/ww/msxenix/pics/

http://toastytech.com/guis/indexwindows.html

http://www.operating-system.org/

http://windowsteamblog.com/blogs/
