Unit OS2:
Operating System Principles

2.5. History of the Windows NT/2000/XP/2004 OS

Copyright Notice

- These materials are part of the *Windows Operating System Internals Curriculum Development Kit,* developed by David A. Solomon and Mark E. Russinovich with Andreas Polze
- Microsoft has licensed these materials from David Solomon Expert Seminars, Inc. for distribution to academic organizations solely for use in academic environments (and not for commercial use)

Roadmap for Section 2.5.

- History of NT
- Windows Release History and Versions
- New features in Windows XP/2003
- Original Windows Design Goals/Culture
- How Development Processes evolved throughout Windows NT 3.1 Development and Windows 2000 Development

NT Timeline first 10 years

- 2/89 Coding Begins
- 7/93 NT 3.1 Ships
- 9/94 NT 3.5 Ships
- 5/95 NT 3.51 Ships
- 7/96 NT 4.0 Ships
- 12/99 NT 5.0 a.k.a. Windows 2000 ships

Unix Timeline first 20 years

- ’69 Coding Begins
- ’71 First Edition – PDP 11/20
- ’73 Fourth Edition – Rewritten in C
- ’75 Fifth Edition – Leaves Bell Labs, basis for BSD 1.x
- ’79 Seventh Edition – One of the best
- ’82 System III
- ’84 4.2 BSD
- ’89 SVR4 Unification of Xenix, BSD, System V
  - NT development begins

History of NT

- Team forms November 1988
- Six guys from DEC
- One guy from Microsoft
- Build from the ground up
  - Advanced PC Operating System
  - Designed for desktops and servers
  - Secure, scalable SMP design
  - All new code
- Schedule: 18months (only missed the date by 3 years)

History of NT (cont.)

- Initial effort targeted at Intel i860 code-named N10, hence the name NT which doubled as N-Ten and New Technology
- Most development done on i860 simulator running on OS/2 1.2 (took about 30 minutes)
- Microsoft built a single board i860 computer code named Dazzle including the supporting chipset and actually ran a full kernel, memory management, etc on the machine.
- Compiler came from Metaware with weekly UUCP updates sent to Marc Lucovsky's Sun-4/200.
- Microsoft wrote a PE/Coff linker as well as a graphical cross debugger

Windows NT Origins

- **Design began in late 1988/early 1989 after Dave Cutler and a handful of Digital employees started at Microsoft**
  - Dave Cutler—legend in the operating system world
    - Project leader for Digital VMS
  - Internally, many similarities to Digital's VMS (scheduling, memory management, I/O and driver model)
  - VMS+1=WNT just a coincidence
- **Original goal was replacement for OS/2**
- **Later goal changed to be the replacement for Windows 3.0**
  - The name "Windows NT" was born
  - NT="New Technology"
    - But at a high level, the architecture and user interface are not really that "new" (as compared to most 32-bit OS's)
- **Interesting book on the early years of NT:**
  - Show-stopper!: The Breakneck Race to Create Windows NT and the Next Generation at Microsoft
  - By G. Pascal Zachary, ISBN: 0029356717

Release History

- Although product name has varied, internally, each version identified by a “build number”
  - Internal identification - increments each time NT is built from source (5-6 times a week)
  - Interesting timeline: http://windows2000.about.com/library/weekly/aa010218a.htm

Build#	Version	Date
297	PDC developer release	Jul 1992
511	NT 3.1	Jul 1993
807	NT 3.5	Sep 1994
1057	NT 3.51	May 1995
1381	NT 4.0	Jul 1996
2195	Windows 2000 (NT 5.0)	Dec 1999
2600	Windows XP (NT 5.1)	Aug 2001
3790	Windows Server 2003 (NT 5.2)	Mar 2003
4051	Longhorn PDC Developer Preview	Oct 2003

Windows XP

- Replacement for Windows 2000 Professional, Windows 9x/ME
- Six 32-bit packages:
  - Windows XP Professional
  - Windows XP Home Edition
  - Windows XP Embedded
    - Same kernel as regular 32-bit XP
    - Configurable to remove unnecessary components
    - Boot and execute from ROM (OS runs from RAM, apps from ROM)
  - Windows XP Tablet PC Edition
  - Windows XP Media Center Edition
  - Windows Preinstall Execution Environment (WinPE)
- Two 64-bit packages:
  - Windows XP 64-Bit Edition For 64-Bit Itanium-Based Systems
  - Windows XP 64-Bit Edition For x64
    - NOTE: based on Server 2003 SP1 kernel

Windows Server 2003

- Replacement for Windows 2000 Server family
  - Superset of Windows XP
  - More kernel changes than in Windows XP, but still relatively modest as compared with Windows NT 4.0->Windows 2000
  - Internal version number is 5.2
    - Not the same kernel as Windows XP (5.1)
- Four 32-bit packages:
  - Windows Server 2003, Web Edition (new package)
  - Windows Server 2003, Standard Edition (was "Server")
  - Windows Server 2003, Enterprise Edition (was "Advanced Server")
  - Windows Server 2003, Datacenter Edition (no name change)
- Six 64-bit packages:
  - Windows Server 2003 for 64-Bit Itanium-based Systems
    - Standard, Enterprise, and Datacenter Editions
  - Windows Server 2003 for 64-Bit Extended Systems
    - Standard, Enterprise, and Datacenter Editions

64-bit Windows

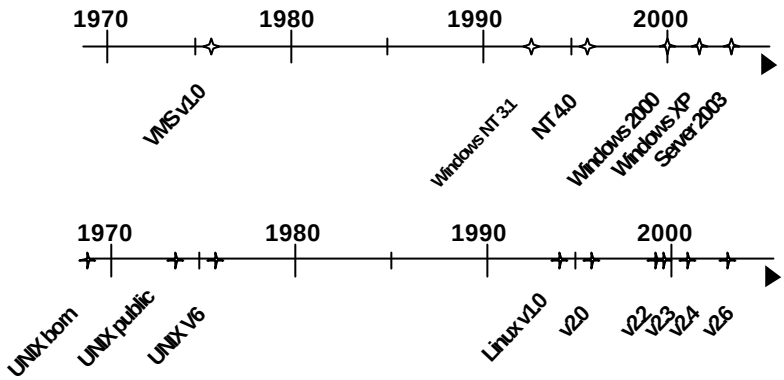
- Windows XP 64-bit edition for Itanium is based on XP kernel
- Windows XP 64-bit edition for x64 is based on Server 2003 kernel

Windows Server 2003 Features

- IIS 6.0
  - HTTP in the kernel
  - Connection failover, process recycling
- Headless server support (no keyboard, video, mouse)
  - Remote Installation Service
  - EMS (Emergency Management Service) allows remote disaster recovery/control via serial port or network
- Active Directory enhancements
- Terminal Services enhancements (scalability, sound, etc)
- Many new group policies
- Bundles .NET Framework
- Hot plug memory, hot plug PCI

Windows And Linux Evolution

- Windows and Linux kernels are based on foundations developed in the mid-1970s



(see http://www.levenez.com for diagrams showing history of Windows & Unix)

Windows Kernel Evolution

- Basic kernel architecture has remained stable while system has evolved
  - Windows 2000: major changes in I/O subsystem (plug & play, power management, WDM), but rest similar to NT4
  - Windows XP & Server 2003: modest upgrades as compared to the changes from NT4 to Windows 2000
- Internal version numbers confirm this:
  - Windows 2000 was 5.0
  - Windows XP is 5.1
  - Windows Server 2003 is 5.2
  - Windows XP 64-bit Edition for x64 is 5.2 (based on the Windows Server 2003 SP1 kernel)
  - Longhorn is 6.0
- But, many interesting kernel changes in XP/2003
  - Will be described through the class in the appropriate sections

Windows XP (& 2003) Kernel Changes

- **Scalability improvements**
  - Increased physical memory support
  - Bigger multiprocessor systems
  - New types of multiprocessor systems
  - Improved kernel synchronization
  - Increases in system virtual memory limits
- **Reliability**
  - System Restore, Driver rollback, Verifier improvements
- **64-bit support**
- **Logical Prefetcher (speeds boot, app startup time)**
- **File system enhancements**
  - FAT32 on DVD-RAM, UDF 2.01, Volume shadow copy
- **Terminal services ships with client**

Windows Server 2003 Kernel Changes

- More security & stability fixes
- Many small usability/manageability improvements
  - Many more command line utilities
- More scalable
  - Per-CPU scheduling queues
  - 8 node clusters
  - 32GB max RAM for Enterprise Edition (was 8 GB)
  - 512GB max RAM for 32-bit Datacenter Edition (was 64 GB)
  - 64 processor support for 64-bit Datacenter Edition

Retrospective: Design Longevity

- OS Code has a long lifetime
- You have to base your OS on solid design principles
- You have to set goals, and not everything can be at the top of the list
- You have to design for evolution in hardware, usage patterns, etc.,
- Only way to succeed is base your design on a solid architectural foundation
- Development environments never get enough attention…

Retrospective: Goal Setting

- First job was to establish high level goals.
  - Portability – Ability to target more than one processor, avoid assembler, abstract away machine dependencies. We purposely started the i386 port very late in order to avoid falling into a typical, Microsoft, x86 centric design.
  - Reliability – Nothing should be able to crash the OS. Anything that crashes the OS is a bug. Very radical thinking inside of Microsoft considering Win16 was cooperative multi-tasking in a single address space, and OS/2 had many similar attributes with respect to memory isolation
  - Extensibility – Ability to extend the OS over time
  - Compatibility – With DOS, OS/2, POSIX, or other popular runtimes. This is the foundation work that allowed us to invent windows two years into NT OS/2 development.
  - Performance – All of the above are more important than raw speed!

NT OS/2 Design Workbook

- Design of executive captured in functional specs
- Written by engineers, for engineers
- Every functional interface was defined and reviewed
- Small teams can do this efficiently,
  - making this process scale is an almost impossible challenge
  - Senior developers are inundated with spec reviews and the value of their feedback becomes meaningless
  - You have to spread review duties broadly, and everyone must share the “culture”

Retrospective: Developing a Culture

- To scale a development team, you need to establish a culture
  - Common way of evaluating designs, making tradeoffs, etc.
  - Common way of developing code and reacting to problems (build breaks, critical bugs, etc.)
  - Common way of establishing ownership of problems
- Goal setting can be the foundation for the culture
- Keeping a culture alive as a team grows is a huge challenge

The NT Culture

- Portability, Reliability, Security, and Extensibility ingrained as the teams top priority
  - Every decision was made in the context of these design goals
- Everyone owns all the code, so whenever something is busted anyone has a right and a duty to fix it
  - Works in small groups (< 150 people) where people cover for each other
  - Fails miserably in large groups
- Sloppiness is not tolerated
  - Great idea, but very difficult to nurture as group grows
  - Abuse and intimidation gets way out of control, can't keep calling people stupid and expect them to listen
- A successful culture has to accept that mistakes will happen

Development Environment

- NT 3.1 vs. Windows 2000
  - Development Teams
  - Source Code Control System
  - Process Management
  - Serialized Development
  - Defects

Development Team

- NT 3.1
  - Starts very small (6), grows very slowly to 200 people
  - NT Culture was commonly understood by all
- Windows 2000
  - Mass assimilation of other teams into the NT team
  - NT 4.0 had 800 developers, Windows 2000 had 1400
  - Original NT culture practiced by the old timers in the group, but keeping the culture alive was very difficult due to growth, physical separation, etc.
    - Diluted culture leads to much conflict
      - Accountability: I don't "own" the code that is busted, see Markl
      - reliability vs. new features
      - 64-bit portability vs. new features

Source Code Control System (NT 3.1)

- Internally developed, maintained by a non-NT tools team
  - No branch capability, but with small team, it was not needed
- 10-12 well isolated source “projects”, 6M LOC
- Informal project separation worked well
  - minimal obscure source level dependencies
- Small hard drive could easily hold entire source tree
- Developer could easily stay in synch with changes made to the system

Source Code Control System (Windows 2000)

- Windows team takes ownership of source code control system which at this point is on life support
- Branch capability sorely needed, tree copies used as substitute, so merging is a nightmare
- 180 source “projects” 29M LOC
- No project separation, reaching “up and over” was very common as developers tried to minimize what they had to carry on their machines to get their jobs done
- Full source base required about 50Gb of disk space
- To keep a machine in synch was a huge chore (1 week to setup, 2 hours per-day to synchronize)

Process Management (NT 3.1)

- Safe synch period in effect for ~4 hours each day, all other times the rule is check-in when ready
- Build lab synchs during morning safe synch period, and starts a complete build.
  - Build breaks are corrected manually during the build process (1-2 breaks was normal)
- Complete build time is 5 hours on ~486/50
- Build is boot tested with some very minimal testing before release to stress testing
  - Defects corrected with incremental build fixes
- 4pm, stress testing on ~100 machines begins

Process Management (Windows 2000)

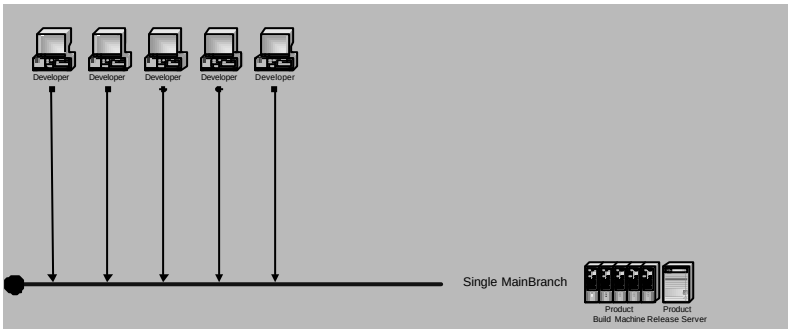
- Developers are not allowed to change the source tree without explicit, email/written permission
  - Build lab manually approves each check-in using a combination of email, web, and bug tracking database
- Build lab approves about 100 changes each day and manually issues the appropriate synch and build commands
  - Build breaks are corrected manually, and when they occur, all further build processing is halted
  - A developer that mistypes a build instruction can stop the build lab, which in turn stops over 5,000 people
- Complete build time is 8 hours on 4 way PIII Xeon 550 with 50Gb disk and 512k RAM
- Build is boot tested and assuming we get a boot, extensive baseline testing begins
  - Testing is a mostly manual, semi-automated process
  - Defects occurring in the boot or test phase must be corrected before build is “released” for stress testing
- 4pm, stress testing on ~1000 machines begins

Team Size

Product	Dev Team Size	Test Team Size
NT 3.1	200	140
NT 3.5	300	230
NT 3.51	450	325
NT 4.0	800	700
Win2k	1400	1700

Serialized Development

- The model from NT 3.1 -> Windows 2000
- All developers on team check-in to a single main line branch
- Master build lab synchs to main branch and builds and releases from that branch
- Checked in defect affects everyone waiting for results



Defect Rates and Serialization

- Compile time or run time bugs that occur in a developers office only affect that developer
- Once a defect is checked-in, the number of people affected by the defect increases
- Best developers are going to check-in a runtime or compile time mistake at least twice each year
- Best developers will be able to cope with a checked-in compile time or run time break very quickly (about 20 minutes end-to-end)
- As the code base gets larger, and as the team gets larger, these numbers typically double

Defect Rates Data

- With serialized development:
  - Good, small teams operate efficiently
  - Even the absolute best large teams are always broken, and always serialized

Product and Team Size	Defects: Per year Per Dev	Time to Fix: Per Defect	Defects: Per Day	Total Defect Fix Time
NT 3.1, 200	2	20 minutes	1	20 minutes
NT 3.5, 300	2	25 minutes	1.6	41 minutes
NT 3.51, 450	2	30 minutes	2.5	1.2 hours
NT 4.0, 800	3	35 minutes	6.6	3.8 hours
Win2k, 1400	4	40 minutes	**15.3**	**10.2 hours**

Development Environment Summary

- NT 3.1
  - Fast and loose development, lots of fun, lots of energy
  - Few barriers to getting your work done
  - Defects serialized parts of the process, but didn't stop the whole machine, minimal down time
- Windows 2000
  - Source code control system bursting at the seams
  - Excessive process management serialized the entire development process, 1 defect stops 1400 devs, 5000 team members!
  - Resources required to build a complete instance of NT were excessive giving few developers a way to be successful

Focused Fixes

- Source Code Control System
- Source Code Restructuring
- Make the large team work like a set of small teams
  - Windows is already organized into reasonable size development teams
  - Goal is to allow these teams to work as a team when contributing source code changes rather than as a group of individuals that happen to work for the same VP
    - Parallel Development, Team Level Independence
- Automated Builds

Source Code Control System

- New source code control system identified 3/99 (SourceDepot)
- Native branch support
- Scalable, high speed, client server architecture
- New machine setup 3 hours vs. 1 week
- Normal synch 5 minutes vs. 2 hours

Summary

- The initial NT development environment and culture worked well for the first few years
- Ten years of team and code growth forced a major re-design of the development environment and culture
- With the new environment in place, the team is working a lot like they did in the NT 3.1 days with a small, fast moving, development team

Further Reading

- Mark E. Russinovich and David A. Solomon, Microsoft Windows Internals, 4th Edition, Microsoft Press, 2004.
  - Historical Perspective (pp. xix ff.)
- G. Pascal Zachary, Show Stopper! The Breakneck Race to Create Windows NT and the Next Generation at Microsoft, ISBN: 0029356717, Free Press, 1994.