

Operating Systems

Athabasca University, COMP 601
Arman Kanooni, September 29, 2002

Abstract:

In this report, we will review a brief history of computer operating systems, from early legacy systems to the new and emerging operating systems such as Linux and Windows XP. We shall see how the existing operating systems are using some of the features from older legacy operating systems. We will examine succinctly other types of OS for real-time and distributed operating systems.

Key Words:

Batch System, BSD, CTSS, DIGITAL MVS, Free Software Foundation (FSF), FreeBSD GNU project, GNU/Linux, IBM 701, IBM 704, IBM 709/7090, IBM OS/360, IBSYS, Linux kernel, MCP, Microsoft MS-DOS, Microsoft Windows NT, Motorola 68000 family of chips, MULTICS, OpenBSD, OS/2, OS/360, Solaris, TCP/IP protocols, Unix

Introduction

In this document we will look at the evolution of operating systems from the first beginning 1954 to today's modern operating systems such as Windows XP and GNU/Linux. We look at the major characteristics of well know operating systems like OS/360, UNIX, DIGITAL MVS, OS/2 and NT, Linux.

In software tools, we shall layout a chronological view of different operating systems and their major capabilities. Finally, we compare two operating systems VMS versus UNIX in term the utilization of operating system commands.

We reviewed a study done in subject of robustness of the Microsoft operating system in compare to Linux.

When Fortran started to become popular, certain element of the computer operating systems has been developed. These services were the automatic sequencing of the applications, Input / Output management, call to assembler or compiler and the load and the execution of the program and its routine libraries. For example, a rudimentary operating system has been developed conjointly between the General Motors and North American Aviation in 1956 for IBM 704 computer. [9]

In the beginning of 60's the **IBSYS** operating system was introduced for the IBM 7090 computers. These systems were based on a **batch system** in which jobs are bundled together with the instructions necessary to allow them to be processed without

intervention. Often jobs of a similar nature can be bundled together to further increase economy. In certain case the results could take couple of hours or days. The programmers specified the tasks to be executed via job control cards.

Then, the batch system became richer by using the capacity of multi-processing and multiprogramming allowing among other things the development of timesharing system.

For example, Burroughs was offering in 1963 the **MCP** (Master Control Program) operating system, developed in high level language and offering multiprocessing and multiprogramming (always in batch mode). Another innovation was the Compatible Time-Sharing System (**CTSS**), developed in MIT under IBM709/7090 system between 1961 and 1963. CTSS had the following characteristics:

1. Several terminals could be connected to the mainframe and be used simultaneously
2. Each user had the impression to possess the computer and its software to him/her self.
3. The execution of smaller jobs was sufficiently short for all the users that they had the impression that the computer was reserved for them.

Near 1970, the most powerful, general and interactive operating system was **MULTICS** (**Multiplexed Information and Computing System**) [5], developed jointly by General Electric, MIT and AT&T Bell laboratories. This was the precursor of **UNIX** and was heavily inspired from MIT experience with CTSS.

Finally, the third concept to do its path in certain operating system is the **real time**, developed in the framework of **SAGE** project. In the same vein, the project **SABRE**, near the real time, was the primary transactional operating system and it has even influenced the **OS/360**.

At the end of the 60's, the three concept of interaction, batch processing and real time started to converge because all have a common objective: the management of computer resources. The modern operating system such as UNIX and Windows XP could operate in batch, interactive or real time mode, based on the user's need. Finally, divers functionality has been added to the operating systems to protect against program failure or accidental manipulations done by the users. The command and control languages have been developed to allow users to communicate more easily with the operating system.

OS/360

1. Developed on the middle of the 60's for IBM System 360, OS/360 became the archetype of the batch operating systems even though it was conceived for the larger use such as real time processing. The OS/360 allowed managing the computer resources (memory, peripherals) and the data, the batch jobs, the debugging (via the core dump) and multiprogramming.
2. In addition, OS/360 allowed the clients to increase their processing power (by acquiring another model from 360 family and/or other software) without writing

additional codes. So the results can be obtained in few minutes instead of couple of hours.

UNIX [6],[7],[8]

1. Since the beginning of the 70's, UNIX has become a model of a small but powerful interactive operating system. Ken Thompson and Dennis M. Ritchie from AT&T Bell Labs developed the original version of UNIX. They had participated in the previous mentioned MULTICS project. They successfully wrote an operating system, an assembler, and couple of utility programs. They allowed two users to work at the same time under a PDP-7 system. They called this operating system **UNIX (Uniplexed Operation and Computing System)**.
2. The Assembly language, although fast, is limited to a single type of processor and is relatively hard to master. That's why Thompson and Ritchie created the B language (slow since interpreted) and then the C language. They rewrote UNIX in C in 1973 to make it portable. At this time, UNIX needed 0.5 Mega Bytes of disk space and supported 5 users.
3. The first specification of UNIX was published in 1974 by its authors entitled "**The UNIX Time Sharing System**" through the famous ACM (Association for Computer Machinery) Communications. The article became the basic description of UNIX system. It specified the UNIX structure and working methods. The underlying concepts did not change since.
4. Since 1956, the AT&T telephone monopoly couldn't, by law, enter in the Computer sector. The publication of the article in the ACM Communications excited lots of interest. Therefore, in 1974 AT&T started to sell the UNIX licenses very cheap (\$ 150) to the government and universities. This version became the version 5, even thou AT&T continued to improve its own version. A license provided the source code for UNIX without neither guaranties nor support. Each buyer could modify the source code and add new commands and its own utilities.
5. This was the beginning of the proliferation of various versions of UNIX. Since AT&T preserved UNIX name trademark, the other manufacturers started to give different name to their operating system: **AIX** (IBM), **HP-UX** (HP), **XENIX** (Microsoft), **SunOS** (Sun), **ULTRIX** (DEC) and recently **Linux**. UNIX is very popular in the colleges and the universities.
6. AT&T continued to make UNIX available for the universities up to the version 7, available from 1978. For some people, this is the "true" last version of UNIX. After a restructuring at AT&T in 1981, UNIX was commercialized by Unix Systems Labs (USL, who belongs to Novell since 1993). This version, inspired by the version 7 and another internal version at Bell of UNIX (PWB - Programmer's Workbench), was called System III. Other versions were produced up to System V Release 4 (SVR4) in 1988.

7. The University of California At Berkeley (UCB) has always used UNIX as a research system and added many improvements such as network capability using TCP/IP protocol thanks to the DARPA project. The integration of the version 4.2 of Berkeley to System V Release 3 of USL and the one of Sun (SunOS) in 1987 produced System V Release 4. From this version, Sun produced **Solaris**.
8. Another agreement between AT&T and Microsoft in January 1988 replaced XENIX by UNIX System V/386.
9. In 1988, observing that the three manufactures (AT&T, Sun and Microsoft) trying to establish a control over UNIX, seven major manufactures regrouped to oppose their proper strategy and for writing a version of UNIX alternative to SVR4. They are Digital, Apollo, Bull, Seimen, HP, IBM and Nixdorf. They form the **OSF** (Open Software Foundation) and created a version of UNIX based on the IBM (AIX), itself based on **SVR3** of AT&T and not based on SVR4.
10. In response to OSF, AT&T forms UNIX International which is comprised of AT&T, Microsoft, Amdahl, Informix, Motorola, Unisys, Sun, Control Data, Intel, and NCR. UNIX International product is based on **SVR4**. At the end of 1980's, they were only 2 kind of UNIX system available on the market.
11. Novell bought UNIX form AT&T in 1993. Under the new management, USL produces a new version of UNIX called **Unixware**, offering better interoperability with the Novell Netware product.
12. Face to the medium popularity of Unixware and the immediate menace of Windows 95 and NT, Novell gave in 1994 the rights of the name UNIX to X/Open Foundation to which belong the two clans. Novell actions constitute the biggest unifying effort of the two versions of UNIX.

DIGITAL VMS [9]

1. Dave Cutler has entered at Digital Equipment (DEC) in 1971 for writing an operating system in real time for the 16 bits PDP-11 product line. The operating system could run with 32 Kilobytes with a hierarchy of files (like DOS and Windows), do the multi-tasking and being fast enough to execute the applications in real time. This operating system was called RSX-11M.
2. At the end of the 1970's, the line of 32 bits VAX computers had to replace the popular PDP-11. Also they had to assure the compatibility with PDP-11 and all the applications wrote it. This is in this context that Dave Cutler had to write a new 32 bits operating system called VMS which assured this compatibility by emulation of operating system RSX-11M. VMS is still utilized today among the Digital computers but they have tendency to adopt Microsoft NT, for example among processor Alpha.

OS/2 and NT [9]

1. With the arrival of the Intel 386 Microcomputers in the middle of 1980, Microsoft called upon the talent of Dave Cutler to write an operating system that could take advantage of the new type of processor. In 1988, Dave Cutler left Digital for Microsoft and worked the central part of the operating system OS/2 (kernel) which Microsoft and IBM used to call it OS/2 at that time. The version 1.1 with a graphic interface (Presentation Manager) of OS/2 was launched in October 1988. IBM insisted the use of a different graphic interface from Windows (the version 1 of OS/2 has been launched in 1987 without a graphic interface).
2. Eventually, Microsoft and IBM went their separated ways. This gave birth to two different operating systems: OS/2 of IBM and OS/2 of Microsoft where the user interface was adapted to the Window 3.x. Microsoft revised its version of OS/2 with Dave Cutler and all the subsequent versions were called NT (“New Technology”) from 1993. NT can be perceived like a rework of OS/2 and a fusion with “LAN Manager” that Microsoft had developed with 3COM. IBM did the same with its own version of OS/2. NT and OS/2 of IBM are therefore the operating systems capable of managing a single computer as well as a network of computers. Microsoft sold a “Workstation” version as well as a “Serve” version of NT.

Linux [1]

Prehistory

- **1969** Linus Torvalds Born in Helsinki, Finland.
- **1983.** Richard Stallman created the Free Software Foundation (GNU project).
- **1986** Design of the Unix Operating System by Marice J. Bach was published.

Minix period (1988-1991)

- **1988.** Admitted to the University . The same year Minix emerged.
- **1990.** Takes his first C programming class.
- **1991.** The start of infamous Novell vs U.C. Berkeley lawsuit and Linux development.

The beginning of Linux development (1991-1992)

Hermit-like work during early kernel releases

- **September 1991.** Using Marice J Bach book Design of the Unix Operating System and Minix released the first (0.01) version of Linux kernel (at the age of 22).
- **October 1991.** Announced the first "official" version of Linux, which was version 0.02. At that point, Linux was able to run `bash` (the GNU Bourne Again Shell) and `GCC` (the GNU C compiler), but not much else.
- **January, 1992.** More or less stable version 0.12 released. **License was changed to GPL.** Due to stability this version was soon renamed to 0.9

- **March 1992.** Linux v.95 was released.
- **June 1992.** 386BSD 0.1 was released. A CD-ROM version of 386BSD has been announced in Dr. Dobbs's Journal. All of the distributions and compilation files would fit onto 180Meg of hard drive.
- **1992.** Yggdrasil released the first CD-ROM distribution. "Linux wave" started.
- **1992.** Web began Internet commercialization wave.

Successful Fight with FreeBSD (1993-1997)

- **December 1993** 386BSD 1.0 was released on CD ROM.
- **December 1993. FreeBSD 1.0 was released.** FreeBSD, which originally started life as 386bsd 0.1 with the patch kit applied, has since evolved into an entirely separate BSD lineage in its own right and incorporates many important innovations.
- **1994.** Version 0.99pl15 aka v.1.0 was released via Internet. WEB revolution started with Linux as one of the major beneficiaries. At least five CD Rom distributors already exist selling ~50,000 CD ROM a month. In October Caldera was founded by Bryan Sparks as a start-up venture funded by Ray Noorda, former CEO of Novell. Still very weak networking support limited its role as a workstation.
- **May 1994.** A very successful FreeBSD 1.1 was released.
- **1994** Novell and U.C. Berkeley settled their long-running lawsuit over the legal status of the Berkeley Net/2 tape
- **Digital** invested money into two porting projects to bring Linux to DEC Alpha. Professionals from DEC started contributing to Linux. Quality of the kernel is improved. A pretty decent Ext2 files system was added. Networking started to look acceptable.
- **January 1995.** FreeBSD 2.0 was released.
- **1995.** Red Hat merged with ACC -- Robert Yong of ACC (former founder of Linux Journal) became a CEO.
- **1996.** Linus' first daughter was born. Minor disruptions of kernel development.
- **August 1996.** FreeBSD 2.1.5 released.
- **December 1996.** Linux 2.0 was released.
- 1997 Linus Torvalds Receives 1997 Nokia Foundation Award.
- **March 1997.** Linus Torvalds receives Lifetime Achievement Award at Uniforum.
- **1997. *Linux meets Microsoft: end of the Finland period and of the academic career*** (1988-1997 -- he spent 10 years as a student and researcher at the University of Helsinki, coordinating development of the kernel since 1992). Now he decided to become rich and moved to the Bay Area (Santa Clara) to work for Transmeta (Microsoft's Paul Allen is one of major investors).
- **November 1998.** FreeBSD 3.0 released.
- **1999:** The Linux IPO gold rush, and the raise of the Red Hat.
- **January 1999.** Version 2.2 of the kernel released despite bugs -- Linus probably needed to respond to the announcement of the FreeBSD 3.0 ;-)
- **February 1999.** IBM invested in Red Hat and several other distributors.

- **March 1999.** Novell invested in Red Hat not in Caldera with which it has strong historic ties. That was a confirmation of RH prima donna status among Linux distributors.
- **August 1999.** Red Hat IPO -- *semiofficial end of the Linux history*. Now there is a multibillion commercial structure to care about it and shareholders. Key people in the Linux camp became rich, some probably very rich ;-). Other enthusiasts will increasingly find themselves either "incorporated" or seduced by the commercial rush to capitalize financially on the Linux trend. Enormous emotional effort that so many volunteers put into developing Linux is probably over and most volunteers already ended in full staff or executive positions in various Linux-related companies. It's now official -- Linux turned into corporate supported, corporate funded and corporate developed OS.

Software Tools [9]

We use Alta Vista search engine to find the web site link to explore more information about a specific operating system. For example:

- *search IBM & 701*
- *Search IBM & 7090*
- *Search XDS-940*

In the context of different operating system, we review a historic exploration of different available software tools proper to each operating system. [2]

- **1950 No Operating System**

No Programming Languages

IBM 701 was operator controlled.

(http://www-1.ibm.com/ibm/history/exhibits/701/701_intro.html)

1 job

- load
- run
- dump
- collect results

IBM 704 was most powerful scientific processor.

(<http://www.columbia.edu/acis/history/704.html>)

IBM share group users developing OS. The basic idea was to reduce idle time. Every job had own boot strap and ran alone until processing complete.

- **1960 IBM 709/7090 (IOCS, FMS, SOS)**

(<http://www.frobenius.com/7090.htm>)

Standard file usage routines (SYSIN, SYSOUT, SYSUT1, SYSUT2, SYSUT3)

Programming Languages

IOCS is the I/O control system

SAGE Future real time systems

SABRE On line interaction (American Airlines)

ATLAS Device drivers

System calls

Batch - spooling

Scheduling of jobs by availability of peripherals
Memory Management
Core memory
Drum memory
Demand paging

- **1963 MCP Multiprogramming**

Multiprocessing (2 CPU master/slave)

- Virtual storage
- High level language
- Source level debugging

XDS-940 Paging memory management

<http://burks.brighton.ac.uk/burks/foldoc/40/129.htm>

- Time sharing system
- Virtual memory 16K
- Physical memory 64K (many users at a time)

RC 4000 - Brinch Hansen

<http://web.syr.edu/~pbhansen/html/biography.html>

OS nucleus (Kernel) layered OS

Kernel - collection of concurrent processes

- Round robin CPU scheduler
- Processes sharing memory
- Message processing system

CTTS (Compatible Time-Sharing System)

<http://www.multicians.org/thvv/7094.html>

supported 32 users interactive

- compile
- manipulate file
- run through terminal

1965 MULTICS (MIT)

Virtual address

First virtual computer

Programs could use more memory than actually existed

OS/360 Designed to span small to large computers.

All used same OS allowed porting programs between machines.

DOS/VSE Limited lower overhead for small machines

1972 OS/VS1 IBM - small to medium machines

VM Virtual machine, Medium to large machines Host OS to one or more "guest" operating systems of different or same types.

1974 MVS IBM's primary OS for large computers with complex environments

1972 UNIX Mini-computer,

Interactive
Time-sharing

VAX/VMS 32-bit extension to PDP-11 line of 16-bit mini Virtual memory system

IBM OS/400 IBM's popular business minicomputer

CP/M (Control Program Microcomputers)

- Low overhead
- Standard for 1st generation of microcomputer

1980 - 1990 Microsoft - IBM

MS-DOS (Microsoft Disk Operating System)

- More compatibility than CP/M
- Standard for 2nd generation of microcomputer

Window3.1 (released spring 1992)

- 32 bit applications
- New graphical user interfac
- Extended memory

IBM OS/2 (1987)

- Multi-tasking
- Multi-user

Windows NT (1993)

- LAN manager
- Multi-tasking
- Multi-user

2000 - Modern OS

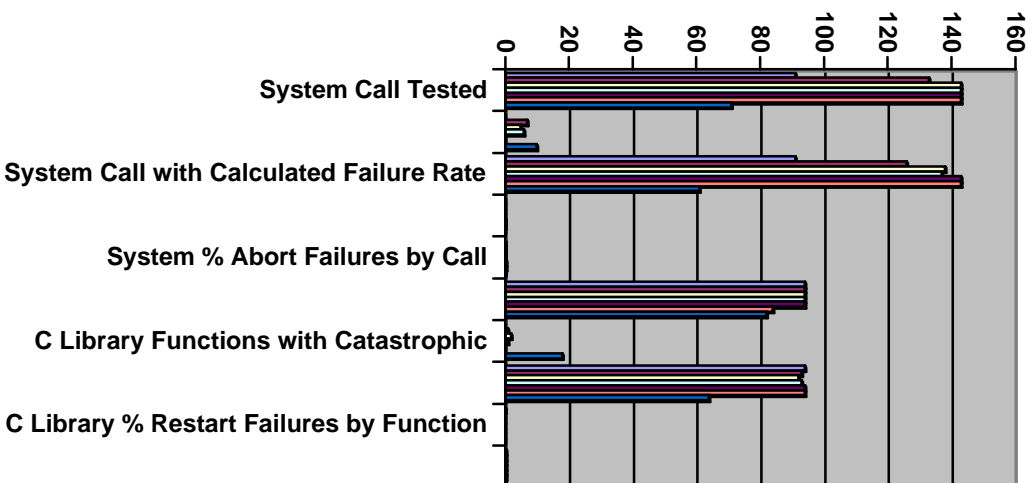
Windows 2000 (2000)

- Multi-tasking
- Multi-processing
- Virtual Memory
- Long filenames

Robustness Testing of the Microsoft Win32 API [4]

Although Microsoft Windows is being deployed in mission-critical applications, little quantitative data has been published about its robustness. We present the results of executing over two million Ballista-generated exception handling tests across 237 functions and system calls involving six Windows variants, as well as similar tests conducted on the Linux operating system. Windows 95, Windows 98, and Windows CE were found to be vulnerable to complete system crashes caused by very simple C programs for several different functions. No system crashes were observed on Windows NT, Windows 2000, and Linux. Linux was significantly more graceful at handling exceptions from system calls in a program-recoverable manner than Windows NT and Windows 2000, but those Windows variants were more robust than Linux (with glibc) at handling C library exceptions. While the choice of operating systems cannot be made solely on the basis of one set of tests, it is hoped that such results will form a starting point for comparing dependability across heterogeneous platforms.

	System Call Tested	System Call with Catastrophic Failure	System Call with Calculated Failure Rate	System % Restart Failures by Call	System % Abort Failures by Call	C Library Functions Tested	C Library Functions with Catastrophic Failure	C Library Functions with Calculated Failure Rates	C Library % Restart Failures by Function	C Library % Abort Failures by Function
Linux	91	0	91	0.2%	7.1%	94	0	94	0.8%	34.9%
Windows 95	133	7	126	0.1%	11.6%	94	1	93	0.02%	24.7%
Windows 98	143	5	138	0.1%	13.3%	94	2	92	0.0%	24.6%
Windows 98E	143	6	137	0.1%	12.9%	94	1	93	0.0%	25.0%
Windows NT	143	0	143	0.3%	23.5%	94	0	94	0.01%	24.6%
Windows 2000	143	0	143	0.4%	22.7%	84	0	94	0.05%	24.1%
Windows CE	71	10	61	0.1%	13.3%	82	18	64	0.0%	14.0%



Robustness Failure rates by Module under Test

Common VMS Commands and their Unix Equivalents

The following table shows some of the commands VMS users use frequently, together with the Unix equivalent. In all cases, full information about the Unix command can be obtained by using **man (manual command)**. **Boldface** is used for actual Unix commands you type; normal typeface for parameters you must supply following the command. Some parameters are optional; these are enclosed in brackets. Numbers in parentheses after the Unix command refer to notes below: [3]

VMS Command	Unix equivalent
-----	-----
COPY fromfile tofile	cp fromfile tofile
CREATE/DIRECTORY directory	mkdir directory
DELETE file	rm file
DELETE directory	rmdir directory
DELETE [...] *.*;*	rm -r directory (1)
(Delete all files in a directory, including subdirectories and their files.)	
DIRECTORY [directory or file]	ls [path] (2)
DIRECTORY/FULL [directory or file]	ls -l [path] (2)
PRINT filename	lp filename (3)
RENAME oldname newname	mv oldname newname
RUN program	program (4)
SET DEFAULT directory	cd path
SHOW DEFAULT	pwd
SET PASSWORD	passwd
SET PROTECTION =protection file	chmod protection file (5)
SET PROTECTION =protection/default	umask protection (5)
SHOW PROTECTION	umask (5)
SHOW USERS	who
SHOW SYSTEM/SUBPROCESSES	ps (6)
TYPE file	cat file

Notes on the above:

1. The Unix command also deletes the directory - not just all the subdirectories and files.
2. The Unix path may name either a directory or a file. If it is a directory, its contents are listed; if a file, just information on that file is listed. If no path is specified, the contents of the current default directory are listed.
3. There are two different printing systems in the Unix world. The version used by IRIX was originally supplied with the AT&T version of Unix uses the **lp** command to print files. The other system, supplied with Berkeley Unix systems (and used in Linux) uses the "**lpr**" command to print.
4. Note that almost all Unix commands do, in fact, run a program of the name specified by the command - e.g. **ls** runs the program `/bin/ls`; **cat** runs `/bin/cat` etc. The shell maintains a list of directories it searches to try to find a file with the name specified on the command line in order to run it. This list includes a number of system directories, plus the bin sub-directory of your home directory - which is where you should put all of your private executable programs to facilitate access to them regardless of what your current default directory is.
5. See the documentation for **chmod** for the format for specifying protections. Note that **umask** always shows a protection in numeric form, and requires you to specify it in this form as well. See the manual page for `chmod` for details, and note that there is no manual page for `umask` as a command (though there is one for the `umask` system service.) since it is built in to `bash` (the command interpreter), see the manual page for `bash` for more information.
6. By default, the **ps** command lists only sub-processes belonging to you; this can be overridden by command switches described on the manual page for `ps`.

Conclusions

We had a brief overview of different operating systems throughout the computer history. It is evident that each new generation of an operating system has built upon the knowledge and discoveries of the previous one. The evolution is closely tied to the state of the current underlying hardware and technology allowing more features and more processing power.

To test your understanding of this fascinating topic you are invited to do a Self Test Quiz online:

<http://io.acad.athabascau.ca/~kanooni/comp601/tme1/quiz.htm>

To learn more about the operating systems, you can visit the online document:

<http://io.acad.athabascau.ca/~kanooni/comp601/tme1/index.htm>

References

1. Chapter 4: A Slightly Skeptical View on Linus Torvalds (unauthorized biography and the first ten years of Linux), Nikolai Bezroukov.
2. Sinclair Community College
Why Operating Systems are Needed - Not What They DO.
<http://www.sinclair.edu/classenhancements/cis225e-dmk/histlect.htm>
3. These notes are based on a set of notes by Prof. R. Bjork, Gordon College and the textbooks Operating System Concepts by Silberschatz and Galvin, Addison-Wesley, 1998 and Operating Systems: Design and Implementation by Tanenbaum and Woodhull, Prentice-Hall, 1997.
4. Philip Koopman
ECE Department
Carnegie Mellon University
Pittsburgh, PA, USA
koopman@cmu.edu
<http://www.ece.cmu.edu/~koopman/ballista/./dsn2000/index.html>
5. Multics--The first seven years
<http://www.multicians.org/f7y.html>
by F. J. CORBATÓ and J. H. SALTZER
Massachusetts Institute of Technology
Cambridge, Massachusetts
and
C. T. CLINGEN
Honeywell Information Systems Inc.
Cambridge, Massachusetts
6. A Brief History of Unix
By Charles Severance
7. An Introduction to Unix
John Montgomery
8. The Evolution of the Unix Time-sharing System
Dennis M. Ritchie
Bell Laboratories, Murray Hill, NJ, 07974
9. Operating System Technical Comparison
<http://www.osdata.com/kind/summary.htm>

10. Nikolai Bezroukov

www.softpanorama.org

Softpanorama: (slightly skeptical) Open Source Software Educational Society

Appendix

Using the UNIX utilities such as grep, awk extract pattern matching keywords to create the following chronological table of operating system:

```
grep -i "IBM 701" *.htm > ostable.txt
grep -i "IBM 704" *.htm >> ostable.txt
grep -i "IBM 709" *.htm >> ostable.txt
grep -i "IBM 701" *.htm >> ostable.txt
grep -i "Univac 1107" *.htm >> ostable.txt
grep -i "IBM 709x" *.htm >> ostable.txt
grep -i "OS/360" *.htm >> ostable.txt
grep -i "VM/370" *.htm >> ostable.txt
grep -i "DOS/360" *.htm >> ostable.txt
```

```
gawk "print $1}" ostable.txt > os.txt
```

Chronology of some Operating System

Operating System	Description	Who	Year
Batch processing monitor (IBM 701)	I/O Routines, Batch Monitor	Group of IBM 701 users during the " Eastern Joint Computer Conference"	1953
GM/NAA I/O System (IBM 704)	I/O Routines, Batch Monitor	Joint effort of General Motors (Bob Patrick) and the North American Aviation (Owen Mock) for rewrite the Batch Processing Monitor of IBM 701 for the 704	1956
UMES (University of Michigan Executive System, IBM 704)	I/O Routines, Batch Monitor	Derived from the research center of General Motors	1957
FMS (Fortran Monitor System, IBM 709)	I/O Routines, Batch Monitor	North American Aviation and IBM	1958
EXEC I-II (Univac 1107)	I/O Routines, Batch Monitor, Multiprogramming	Group of System Programming for the UNIVAC and CSC (Computer Science Corporation) whom was writing compiler for Algol, Fortran and Cobol for UNIVAC	1960
SOS (Share Operating System, IBM 709)	I/O Routines, Batch Monitor	The group of users SHARE	1960
MCP (Master Control Program, Burroughs)	I/O Routines, Batch Monitor, Multiprocess	Developed using high level language by Burroughs	1963
IBSYS (IBM 709x)	I/O Routines, Batch Monitor	IBM decides to support the SOS and renamed it as IBSYS. It will be used by 2nd generation of computer 7090 et 7094. The command language of IBSYS has become the JCL used today on MVS batch and on MVS TSO (terminal)Predecessor of I'OS/360.	1963
CTSS (Compatible Time Sharing System, IBM 7094)	Supervisor of multiprogramming in time sharing	MIT Ancestor of TSO (Time Sharing Option)	1961
MULTICS (Multiplexed Information and Computing Service)	Supervisor of multiprogramming, I/O Routines, Batch Monitor, Files Hierarchical System	MAC Project (Multiple Access Computers) of MIT, GE et Bell Labs (up to 1969). Multics has been built on the principal of CTSS	1961
UNIX (Uniplexed Operating and Computing System)	multi-users complet operating system	Bell Labs separated itself from MAC project and developed UNIX.	1974
OS/360	Princiapl operating system of System/360	IBM	1967
	Versions: PCP (Monitor of batch processing) MFT (multiprog.with fixed tasks) MVT (multiprog.with variable tasks)	Batch system for 360 Job Control Language (JCL) TSO : Time Sharing Option	1971
	Extended version OS/VS	Need of capable OS to manage the virtual memory OS/VS1: replaced MFT, unique virtual memory OS/VS2 replaced MVT	1972
		OS/VS2 SVS: Unique Virtual Memory (16M)OS/VS2 MVS: Multiple Virtual Memory	1974
	Extended Versions OS/MVS SP and MVS/370	Principal OS for 370	1979
	Extended Version OS/MVS XA	For System/370 with architecture XA, Virtual Memory 2GB	1981
	Extended Version OS/MVS ESA	For -IBM ES/3090 - IBM ESA/390 -IBM ES/9000	1988
	OS/390	For IBM System/390	1995
VM/370 ou VM/CMS	OS with virtual machine	Projct at IBM Lab in Cambridge Adopted par IBM on System/370 using the command language CMS (Conversational Monitor System)	1972
DOS/360	Batch processing	IBM System/360	1965
	DOS/VS	Version of DOS/360 with Virtual Storage	1970
	DOS/VSE	Extended version of DOS/VS for IBM 43XX	1980