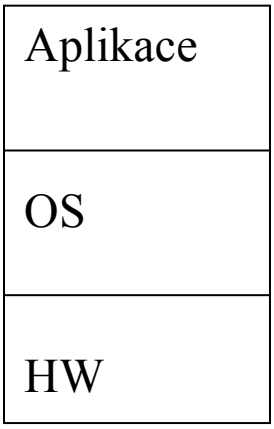
Na co je operační systém?

Funkce vylepšení HW

- sjednocení různosti zařízení
- ulehčení programování (např. časové závislosti)
- přiblížení k potřebám aplikací
  - soubory namísto diskových bloků
  - více „procesorů“ (*time sharing*)
  - větší hlavní paměť (*virtual memory*)

rozšířený stroj (stroj = HW) (*extended machine*)



Funkce správy prostředků (zdrojů) (*resources*)

- cílem je zprostředkovat sdílení prostředků
  - prostředky jsou nákladné
  - pro spolupráci nutno sdílet informace

- úkoly
  - samotné zabezpečení sdílení
  - efektivnost sdílení
    - využití zařízení (*utilization*)
    - výkonnost (*throughput*) – práce za jednotku času
    - doba odpovědi (*response*) – čas vykonání individuálního požadavku
  - správnost
    - neúmyslné poškození
    - zabránit neautorizovaný
      - přístup k datům
      - přístup k prostředkům

správce prostředků (*resource manager*)

Pohled zdola nahoru

HW

- jeden nebo více procesorů (*central processing unit*, CPU)
- hlavní paměť (*main memory*, RAM, *core*)
- V/V zařízení (*I/O devices*)
- sběrnice

CPU má počítadlo instrukcí (PC, *program counter*)

- vybírá instrukce na vykonání podle hodnoty PC
  - zvětší PC o délku vybrané instrukce
  - mění PC při instrukcích skoku a volání procedury
  - při přerušení pokračuje programem pro jeho zpracování (interrupt handler) – neřízené volání procedury

Hlavní paměť reaguje na instrukce „load“ a „store“

V/V zařízení

- vstup a výstup dat
- ukládání dat

THE ABSTRACTIONS OF OPERATING SYSTEMS

No	Name	Objects	power of ten
9	Graphics service	windows, events	+3
8	Shell (GUI)	application programs	+3
7	Directories	directories	0
6	Streams	file, device, pipe	-2
5	Processes	processes	-2
4	Inter-process communication (IPC)	message, port	-3
3	Memory management	address, page, segment	-4
2	Threads	thread, ready list, semaphore	-4
1	Low-level I/O	device, driver	-4
0	Hardware levels	interrupts, procedures, call stacks, instruction set, logic gates	-12 to -9

NOTE: numbers in time scale column are powers of 10 orders of magnitude on the times between events at that level.

Levels 1-5 make up the **microkernel**, i.e., the smallest set of functions that must be executed in privileged (supervisor) mode.

© Copyright 1994-99 P. J. Denning, George Mason University

Problém času

Intenzita (rychlost) vzniku událostí při sdílení prostředků výpočetního systému je velice různá.

Jak moc?

Zdroj: P.J.Denning: The Art of Operating Systems.
//cne/gmu/pjd/ArtOS

Příklady:

- osobní počítač ~ 100 miliónů instrukcí/s
- čtení z disku ~ 10 ms
  - vykoná se 1 000 000 instrukcí
- vstup z klávesnice ~ 200 ms
  - vykoná se 20 000 000 instrukcí

Řešení?

```
začni diskovou operaci;
vykonej 1 000 000 instrukcí jiného
výpočtu;
čekej na skončení diskové
operace(jestliže skončila pokračuj);
```

Dobré řešení

```
P1::
    for (;;) {
        začni V/V operaci
        čekej na skončení
        použij data
}

P2::
    for (;;) {
        nějaký užitečný výpočet
}
```

Problém prostoru

- virtuální adresní prostor programu je dán velikostí lineární adresy
  - 32 bitů - 2^{32} = 4GB
  - 64 bitů - 2^{64} = 16EB (Exa)

- hlavní paměť je typicky menší
  - program se nemusí vejít, zvlášť chceme-li mít v paměti více programů současně

- typické programy
  - pracují v nějakém podprogramu, zbytek nemusí být v paměti
  - pracují s částí dat, ostatní nemusí být v paměti
  - čekají na dokončení V/V operace, nemusí být v paměti vůbec

Řešení

- virtuální paměť
  - program a data jsou na disku
  - části potřebné na výpočet jsou na požádání přenášena do hlavní paměti

Pohled shora dolů

- uživatel aplikace
  - požaduje vykonání nějaké činnosti
  - nemusí vědět na co je OS

- programátor aplikace
  - vytváří aplikace, využívá vylepšený počítač
    - pojmenované soubory
    - „neomezená“ paměť
    - transparentní vykonání V/V operací
  - někteří na úrovni vyšších programovacích jazyků a knihoven
    - ```
      main()
      {
          printf("OS")
      }
      ```
  - jiní využívají programové rozhraní OS, systémová volání, API – application programming interface
    - ```
      main()
      {
        char buf[2] = "OS";
        write(stdout,buf, sizeof(buf);
      }
      ```
- systémový programátor/administrátor
  - potřebuje vědět jak operační systém pracuje

Definice OS není ustálená

Zdroj: L. Petrlík

Pokus o definici

„OS je soubor programů, které minimalizují úsilí uživatele při maximálním využití výpočetního systému“

A.S. Tanenbaum

… a layer of software (called operating system), whose job is to manage all these device and provide user programs with a simplex interface to the hardware.

P.J. Denning

Operating systems fulfill two functions: managing the resources of computing among the competing demands of the system’s users, and providing a high-level environment for programming and program execution.

Silberschatz, Galvin, Gagne

An operating system is similar to a *government*. Like a government, it performs no useful function by itself. It simply provides an *environment* within which other programs can do useful work.

Velikost operačních systémů

- **Linux** – 2 000 000 řádků
- **Windows 2000** – 30 000 000 řádků

Rozdělení OS

- Mainframe OS
  - zpracovávají současně množství úloh
    - dávkové zpracování
    - zpracování transakcí
    - sdílení času
  - OS/390 (IBM)
- Server OS
  - poskytují služby (souborový, webový) mnoha uživatelům
  - UNIX, Linux, Windows 2000
- Multiprocesorové OS
  - paralelní počítače
  - multipočítače
  - multiprocesory
  - varianty serverových OS

- OS osobních počítačů
- OS reálného času
  - hard real-time
  - soft real-time (multimédia)
- OS pro vnořené systémy
  - Windows CE
- Smart Card OS
  - vlastnické (proprietary) OS
  - Java Virtual Machine

Uživatelské rozhraní OS

- je tvořeno souborem aplikačních (uživatelských) programů
  - interpret příkazů (shell)
  - příkazy pro práci se soubory a adresáři
  - filtry (sort)
  - nástroje pro vývoj programů (editory, překladače,…)
  - správa systému
  - různé

- ovládání
  - příkazový řádek
  - okno, GUI

Programové rozhraní OS

- je tvořeno souborem systémových volání, které poskytuje jádro OS
- instrukční soubor je tedy rozšířen o vykonání dalších operací – rozšířený stroj
  - na vykonání systémových volání využívá všechny instrukce CPU
  - některé z nich nemůžou používat aplikační (uživatelské) programy – privilegované instrukce
    - práce s některými registry – PC, stavový registr
    - práce s HW
- CPU musí „vědět“, jestli vykonává instrukci aplikačního programu nebo instrukci jádra

Řešení:

- CPU musí mít možnost vykonávat instrukce v různých stavech, režimech – např. procesory Intel 80x86 mají čtyři různé stavy
- privilegovaný režim (režim jádra (*kernel mode*), režim supervizoru)
  - přepnutí do privilegovaného režimu
    - program žádá systémové volání
    - výjimka (exception)
    - přerušení od zařízení
- neprivilegovaný režim (uživatelský režim)
  - přepnutí do neprivilegovaného režimu, při návratu z privilegovaného, instrukcí, která je privilegovaná
  - všechny aplikační programy, tedy i programy tvořící uživatelské rozhraní, se vykonávají v neprivilegovaném režimu

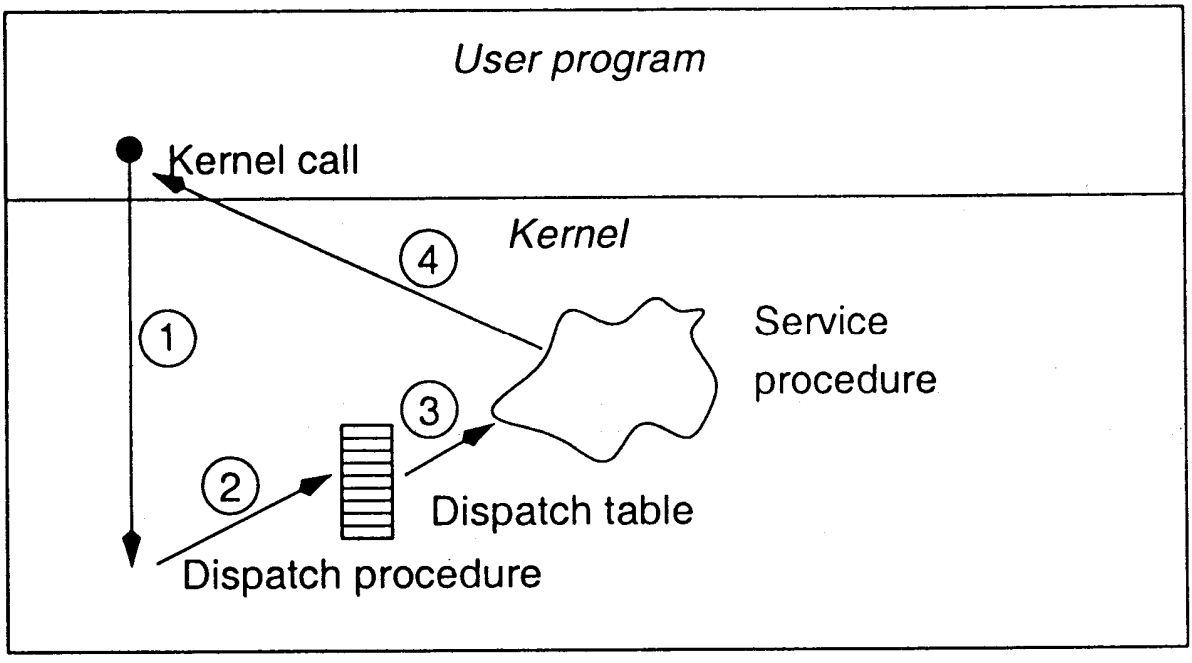
UNIX	WinNT	Popis
fork	CreateProcess	Vytvoří nový proces
waitpid	WaitForSingleObject	Čeká na ukončení procesu
execve	ShellExecute+ WaitForSingleObject	Spuštění procesu a čekání na ukončení
exit	ExitProcess	Ukončení proces
open	CreateFile	Otevřít/vytvořit soubor
close	CloseHandle	Zavřít soubor
read	ReadFile	Čtení ze souboru
write	WriteFile	Zápis do souboru
lseek	SetFilePointer	Nastavení pozice v souboru
stat	GetFileAttributesEx	Získání atributů souboru
mkdir	CreateDirectory	Vytvoří adresář
rmdir	RemoveDirectory	Smaže adresář
link	CreateHardLink	Vytvoří odkaz na soubor
unlink	DeleteFile	Smaže odkaz
mount	SetVolumeMountPoint	Připojí diskový oddíl jako adresář (nebo disk – jen Win)
umount	DeleteVolumeMountPoint	Odpojí diskový oddíl z adresářové struktury
chdir	SetCurrentDirectory	Nastaví aktuální adresář procesu
chmod	SetSecurityDescriptorOwner	Práva a zabezpečení
kill	TerminateProcess	Násilné ukončení procesu
time	GetLocalTime	Aktuální čas

- Srovnání API Unixu a WinNT.
- K jedné funkci lze přiřadit i více alternativ.
  - zpráva `WM_QUIT` ke `kill`
  - `GetSystemTime` k `time`
  - příznak `COPY_FILE_COPY_SYMLINK` k `link`

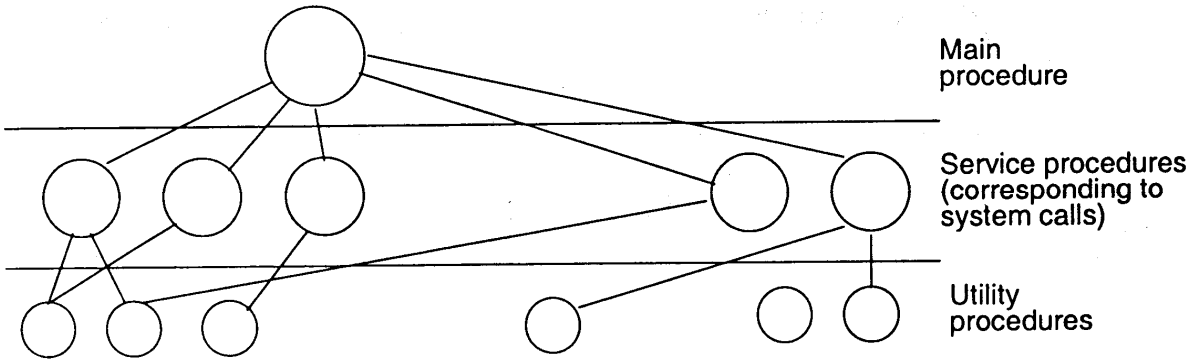
Struktura OS

- **Monolitické systémy**
  - pro jednotlivé funkce jsou definovány moduly
  - modul může volat jakýkoli jiný modul
  - všechny moduly jsou spojeny do vykonatelného souboru s operačním systémem

- Systémové volání (služba jádra)
  - volání vstupního bodu jádra OS s přepnutím do privilegovaného režimu
  - zjištění čísla požadované služby
  - volání obslužné procedury
  - návrat s přepnutím do neprivilegovaného režimu



- Model struktury monolitického systému
  - hlavní program, který spouští obslužnou proceduru
  - množina obslužných procedur pro systémová volání
  - podpůrné procedury pro vykonání obslužných procedur



- Monolitické systémy
  - mají tendenci extrémně narůstat
  - monolit akumuluje moduly, které by potenciálně mohli být potřebné
  - těžce se ladí

Příklady: Linux, MacOS, Windows *

* Win32 a Win64 jsou stejně jako např. POSIX subsystémy mimo jádro OS

Systémy založené na mikrojádře

- Vrstvené systémy
  - hierarchie vrstev poskytujících služby
  - programy vyšší vrstvy využívají služeb nižších vrstev
  - holý počítač je nejnižší vrstva
  - aplikační program je nejvyšší vrstva
  - princip vrstev umožňuje systematickou tvorbu programů a jejich testování
  - princip vrstev je možné použít pro monolitický model i pro systémy založené na mikrojádře

- Mikrojádro
  - vrstva nad holým strojem, která obsahuje minimální množinu abstrakcí, tak aby ostatní funkce OS mohli být implementovány nad ním
  - tyto funkce OS nemusí být vykonávány v privilegovaném režimu
  - jenom mikrojádro musí být vykonáváno v privilegovaném režimu
  - typická množina abstrakcí implementována mikrojádrem (zdola nahoru)
    - přerušení, nízkoúrovňový V/V
    - vlákna
    - správa paměti (JavaOS)
    - meziprocesovou komunikaci
    - procesy
  - ostatní funkce - soubory, adresáře, síťové služby - jsou programy vykonávané v uživatelském režimu

Příklady: Mach, Chorus Amoeba

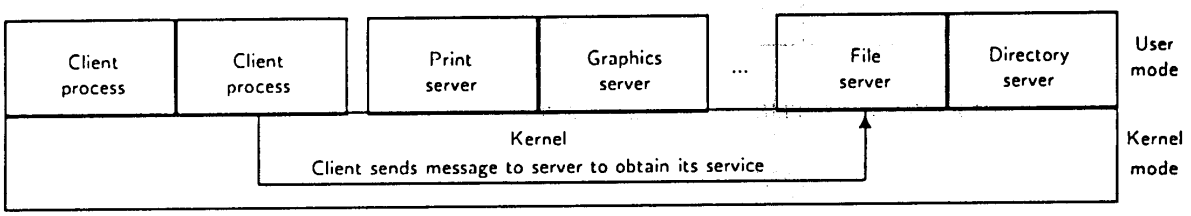


Figure 1: The client-server model on a single machine.

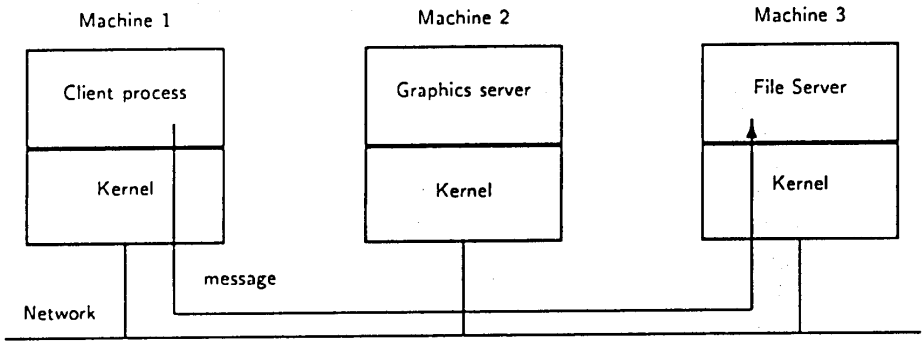


Figure 2: The client-server model in a distributed system.

Plán přednášek

- **OS**
- **UNIX**
- **Proces a jádro**
- **Vlákna**
- **Signály**
- **Plánování procesů**
- **Meziprocesová komunikace**
- **Multiprocessing**
- **Souborový systém**
- **Implementace souborového systému**
- **Přidělování paměti**
- **V/V podsystém**
- **Mikrojádra**

Cvičení

- **Projekt (skupina)**