1. Unix/Linux – historie, koncepce, shell, souborový systém (i-nod, …), procesy

* 50. léta
  + první **počítače bez OS**
  + měli unikátní program přímo pro něj
  + na konci **náznaky OS → tzv. monitor** **→** základní systém ochrany paměti, IO operace a kontrola nekonečné smyčky
* 60. léta
  + **vyšší rychlost** počítačů => **čekání na IO operace**
  + vznikají **počítačová centra** (hlavní rychlý počítač pro práci s mg. páskami a několik vedlejších pro přepis programů z děrných na magnetické)
  + vznik spoolovacích systémů (**SPOOL** = Simultaneus Peripheral Operation OnLine)
    - běh **více programů zároveň** a **přepínání** **mezi** **nimi** podle čekání a na dokončování **IO operací**
* 1964 – projekt **MULTICS**, což je rozsáhlý počítač s velkým množstvím uživatel
* 1969
  + **UNICS** napsaný v **Assembleru**
    - maličké jádro **řídící spouštění a běh programů**, přístup k **I/O prostředkům** atd
    - **hierarchický** souborový systém a **využití pipy**
    - **periferie** a **logické** části systému jako soubor
* 1970 – název UNIX
* 1973 – vytvoření jazyka C
* 1975 – první komerční verze **UNIXu**
* 80. léta – standardizace normou **IEEE POSIX** (rozhraní, přenositelnost)
* 1983
  + projekt **GNU**, za účelem **vyvinout OS se svobodnou licencí založený na původním UNIXu**, ale **nevyužívající** jeho kód (vytvoření General Public Licence)
* 1991 – Linus Torvalds – první **LINUXové** **jádro**
* 1992 – grafické prostředí **X Windows System** (snadné propojení textového a graf.)
* 1996 – vytvoření prostředí **KDE**
* 1997 – vytvoření prostředí **GNOME**

**Operační systém UNIX**

* Jádro (kernel)
  + v **privilegovaném režimu** (režim jádra) a **často jediný soubor**
  + využívá **souvislý adresový prostor** (nazýváno monolitické, ale s přesně rozvrženou vnitřní strukturou)
  + spravuje hardwarové zdroje, jako je **paměť** (alokaci a dealokaci paměti), **procesory** (vytváření, plánování a ukončování procesů) a **zařízení** (device drivers)
* Vrstva HAL (Hardware Abstraction Layer)
  + **základní rozhraní k** **zařízením**
  + HAL je vrstva **softwaru**, která **abstrakuje detaily hardwaru**, aby operační systém a aplikace **nemusely** pracovat přímo s konkrétním hardwarem.
  + skrytí technických detailů zařízení patřících do různých tříd
  + načítání ovladačů a provozování abstraktního modelu HW
  + **vytváření** a **odstraňování** **přípojných bodů** pro bloková (paměťová) zařízení
  + v novějších systémech se od používání upouští, nahrazují jiné moduly jádra
* Ovladače
  + **bloková** a **znaková** **zařízení**, **síťová a virtuální zařízení** + další specializované
  + **souborové systémy** také jako ovladače
* Souborový systém
  + **rozhraní** často mezi **ovladačem vnějšího paměťového média** a **vyššími vrstvami jádra**, ale **nemusí** **s paměťovými médii souviset**
  + -“všechno je soubor“ → souborové systémy **pro vnější paměťová média**, pro **přístup k informacím o stavu systému**, konfiguraci, apod.
  + **VFS (Virtual File System)**
    - **nenáleží k žádnému paměťovému médiu**
    - nejdůležitější virtuální souborový systém
    - všechny souborové systémy v jednom stromu

**Operační systém LINUX**

* Architektura vrstvená
  + jádro (**kernel**) - vlastní základ OS
  + příkazový interpret (**shell**) - **textový příkazový režim**, **spuštění programu**, programovací jazyk, např. GNU Bash
  + (základní) **programy** – **operace s daty** (soubory) a **úlohami** (procesy), “tiché chování”
* Charakteristika
  + **Svobodný** software (**GPL**) a koncepce **UNIXU** (Unix != Linux)
  + **Přenositelnost** a **škálovatelnost** + široká **hardwarová a softwarová podpora**
  + Distribuce **GNU/Linux**
    - “**balení**” systému pro snadnou instalaci a správu
    - další **administrátorské programy** a **specializovaný software**
    - programy (i jádro) ve **formě balíčků**, závislosti mezi balíčky
    - **Ubuntu**, **Debian**, Mandriva, Fedora, **SuSE**, Gentoo

**Shell**

* Označení programu, **který vytváří v počítači rozhraní pro práci uživatele**, zpravidla v prostředí **příkazového řádku**
* Umožňuje uživateli **využívat služby operačního systému**, zejména **spouštět programy, zajišťovat pro ně vstupy, zobrazovat, uchovávat a přesměrovávat jejich výstupy, spojovat jednotlivé programy do kolon** a podobně
* Shell dělíme na dvě skupiny:
  + **řádkové** (textové)
    - vytvářejí příkazový řádek **(CLI)** a jsou využívány **zejména pro administraci počítačových systémů**
    - Almquist shell (ash), **Bourne-Again shell (bash),** Bourne shell (sh)
    - Dělíme na dva režimy:
      * **interaktivní** režim
        + vložené **příkazy ihned po stisknutí klávesy enter** na konci zadaného řádku zpracovány -> interpretace příkazů -> interpret
      * **dávkový** režim
        + shell provádí příkazy **předem zapsané v textovém souboru** -> příkazové soubory - **shellové skripty**
  + **grafické** (vizuální)
    - vytvářejí grafické uživatelské rozhraní **(GUI**), které je pro běžné uživatele **uživatelsky přívětivější**
    - příkazy **nezadávány** psaním do řádky,
    - akce uživatele **v rámci grafického prostředí na pozadí převedena** do „řádku příkazů“ a provedena (spuštěna)

**Jádro**

* **Řízení provádění úloh** (vytváření, ukončení, suspendování, komunikace, přístup k periferiím, …)
* **Správa systému souborů** (organizace disku, vytváření a mazání souborů, řízení práv, udržování konzistence, …)
* **Správa paměti** (přidělování, uvolňování, ochrana, swapping/paging, …)
* **Plánování procesů pro sdílení času CPU** (plánovací algoritmus, přidělování časových kvót, priority, …)

**Souborový systém**

* hierarchický systém souborů
* / – kořen souborového systému
* **/bin** – **základní spustitelné soubory pro použití všemi uživateli**
* **/boot –** **jádro systému**, initrd, soubory zavaděče (boot loader) GRUB
* **/dev** – jednotlivá **fyzická zařízení** nebo **pseudozařízení systému**
* **/etc** – globální **konfigurační soubory systému**
* **/home** – **domovské adresáře** **uživatelů**
* **/lib** – základní **sdílené knihovny systému**, **mapování klávesnice**, konzolové fonty, moduly pro kernel
* /lost+found – ztracené a opravené soubory po chybách FS (ext2,3)
* /mnt – podadresáře, ke kterým se připojují další soubor. zařízení
* **/opt** – **SW** (aplikace) jež **nejsou součástí distribuce**
* **/proc** – soubory **nastavení** a **stavu systému** a jednotlivých procesů
* **/root** – **domovský adresář superuživatele** (root)
* /sbin – systémové privilegované spustitelné soubory (práva má root)
* /sys – virtuální adresář
* /tmp – odkládací a pomocné soubory
* /usr – další důležité podadresáře (knihovny, zdrojové kódy, konfigurační soubory, …)
* **/var** – **soubory**, jejichž obsah se **během chodu systému mění**
* Obsah obrázku snímek obrazovky, řada/pruh, Vykreslený graf

  Popis byl vytvořen automaticky

**Procesy**

* **Program** = **spustitelný soubor** (aplikace prezentována pasivně)
* **Proces** = **spuštěný soubor** (aplikace prezentována aktivně)
* **Multitasking** -> cyklické přidělování **CPU plánovačem procesů**
* **PID** (Process IDentifier) = **číselný identifikátor procesu** (stromová hierarchie)
  + ps -e: Zobrazí seznam všech procesů v systému spolu s jejich PID
  + PID 1 - Je to **první proces**, který je spuštěn **při startu systému** a je rodičem všech ostatních procesů. Ve většině moderních Linuxových systémů je to systémový správce **systemd**
  + systémové programy jako potomci
* Ukončení proběhne buď doběhnutím nebo násilně (ukončí i potomci)

# Souborové systémy (FAT, NTFS, UFS, ext2, ext3, ext4, RieserFS, Btrfs) – charakteristika

### FAT (File Allocation Table)

* systém vytvořený pro DOS
* flash disky a paměťové disky s menší kapacitou

**Logická struktura disku**

* systémové oblasti pevného disku
  + master boot record (MBR) = hlavní zaváděcí záznam disku
  + partition table = tabulka oblastí
  + boot record = lokální zavaděč OS
  + FAT = alokační tabulka, záznam o obsazení disku (clusterů)
  + root directory = kořenový adresář, informace o souborech a složkách
* datová oblast pevného disku
* Možné chyby:
  + fragmentace souborů = soubor ve více clusterech na různých místech
  + ztracené fragmenty souborů = data se zapíší, ale neuloží se jméno (chkdsk)
  + překřížené soubory = více záznamů FAT ukazuje na stejný cluster
  + neplatná podsložka = složka zapsána jako soubor
  + poškození FAT = chyby, pády, odpojení média

### NTFS

* vytvořený pro Windows NT (od verze Win NT 3.x a vyšší)
* starší specifikace 32 bitové adresy (232 - 1 clusterů), novější 64 bitové adresy (264 - 1)
* maximální velikost souboru 16TB
* **GPT** (GUID Partition Table) - náhrada za MBR
  + pro oddíly s větší kapacitou
  + až 128 primárních diskových oddílů
  + nutno použít UEFI
  + opouští se od dělení na cylindry, hlavy a sektory -> práce pouze s logickým blokovým adresováním (LBA)
* podpora všech velikostí clusterů (512B - 64Kb) - 4KB standard

**Systémová oblast diskového oddílu**

* souborový systém řešený jako databáze => jeden záznam = jeden soubor
* při naformátování diskového oddílu vznik 11 systémových souborů (metadat)
* $MFT = master file table, hlavní část NTFS, informace o rozložení všech dat na disku
* $MFTMIRR = kopie nejdůležitějších metasouboru MFT, uprostřed disku
* $BITMAP = pole bitů, kde každý odpovídá clusteru na disku; bit = 0 = cluster prázdný
* Vlastnosti:
  + žurnálování = $LOGFILE, zaznamenávání všech zápisů na disk, při selhání systému možnost operace dokončit nebo stornovat
  + přidělování práv = přístupová práva uživatelům a skupinám
  + komprese = zmenšení objemu dat na úrovní OS, zpomalení systému
  + šifrování = převedení dat do formátu pro ostatní uživatele nečitelného
  + diskové kvóty = nastavení maximálního místa pro konkrétní uživatele
  + dlouhé názvy souborů = až 255 pomocí UNICODE kódování

### UFS (Unix File System)

* základ pro filesystemy Linuxu

**Struktura logického disku:**

* boot blok
* skupina
  + superblok = metadata souborového systému (identifikace, volné místo, …)
  + blok s i-nody (tabulka i-nodů)
  + datové bloky (alokační jednotky)
* další skupina
* další skupina …

### ext2 (second extended filesystem)

* vychází ze systému souborů UFS z unixových systémů

**Struktura logického disku:**

* boot sektor
* skupiny bloků
  + superblok (počet i-node, počet bloků, počet fragmentů ve skupině atd.)
  + deskriptor skupiny
  + bitmapa bloku
  + bitmapa inodů
  + tabulka inodů
  + datové bloky

-typy souborů = běžný soubor, adresář, pevný odkaz, symbolický odkaz, blokové a znakové zařízení, pojmenovaná roura, socket

-komprimace = e2compr = nevytváří nový FS, komprimace a dekomprimace za běhu

### ext3

* žurnálovací funkce
  + žurnál = metadata a obsah souborů ukládány do žurnálu, poté zapsány na disk (spolehlivé, ale pomalé)
  + writeback = žurnálování pouze metadat (nejrychlejší ale při pádu riziko zápisu na nesprávné místo)
  + ordered = jako writeback, ale vynucení zápisu souboru než je označen jako zapsán
* neexistuje defragmentace za běhu = pouze offline defragmentátor, nutnost konverze na ext2 => možnost poškození dat
* nulování ukazatelů i-nodů u smazaných souborů (neobnovitelnost smazaných souborů)
* možnost změnit velikost souborového systému za běhu
* komprimace = rozšíření e3compr

### ext4

* nástupce ext3, zpětně kompatibilní
* podpora **extentů**
  + nahrazení blokového mapování disku
  + extent = rozsah navazujících fyzických bloků
* prealkoace místa na disku
* kontrolní součet žurnálu
* online defragmentace
* komprimace nepodporována

### ReiserFS

* první FS se **žurnálováním**
* struktura vyváženého B+stromu (každý list obsahuje odkaz na následující list)
* maximální velikost oddílu = 16 TiB
* maximální velikost souboru = 1 EiB (32bit systémy 8 TiB)

### Btrfs

* Copy-on-Write B-tree filesystem
  + při kopírování nedochází k okamžité duplikaci dat (až do první změny)
* subvoluming
  + jeden FS rozprostřen přes více fyzických zařízení
* snapshot (snímky disků)
  + ukládání snímků aktuálního stavu a změn oproti tomuto stavu
* maximální velikost souboru = 16 EiB
* podpora SSH disků
* možnost konverze z ext2, ext3, ext4, ReiserFS

# Druhy souborů v Linuxu, i-node

## Soubory

* obsahují vlastní **data** a **programy**
* **atributy**: jméno, velikost v bajtech, číslo i-uzlu, přístupová práva, majitel a skupina, datum poslední změny (posledního přístupu)

### Speciální soubory

* reprezentují I/O zařízení, v adresáři **/dev**
* soubory znakových zařízení => čtení a zápis po jednom znaku (serial port, USB)
* soubory blokových zařízení => čtení a zápis najednou celého bloku (HDD, DVD)
* další speciální soubory => socket a roura

### Názvy souborů a adresářů

* case sensitive
* skryté začínají “.”
* přípony vůbec, násobné nebo dle dohody

## I-node

* datová struktura uchovávající informace o souborech a adresářích
  + metadata
  + odkazy na datové bloky (uložení samotných dat)
* jeden soubor = právě jeden i-node
* v souborových systémech vycházejících z UFS (ext2, ext3, …)
* každý má své číslo (indexové číslo)
* **NEobsahuje** jméno souboru (jméno v adresáři, pro jeden soubor lze více jmen -> hardlink)
* počet určen při formátování, později nelze změnit
* počet určuje maximální počet adresářů a souborů (i když je ještě místo nelze vytvořit nový soubor nebo adresář)

### Uchovávané informace o souboru:

* typ souboru (file type)
* oprávnění pro přístup (permission)
* vlastník (owner)Obsah obrázku text, mapa, diagram, Plán

  Popis byl vytvořen automaticky
* vlastnická skupina (group)
* čas posledního přístupu a změny (file access and modification time)
* čas smazání souboru (file deletion time)
* počet odkazů (number of links)
* Access Control List (ACL)
* pouze pro přidání (append only)
* nesmazatelný soubor (immutability)
* **NEuchovává** se informace o času vytvoření souboru
* odkaz na data
* 12x přímý odkaz
  + na malé soubory = rychlé načtení
* nepřímé odkazy na datové bloky
  + neobsahují data, ale odkazy
  + zmenšení i-node
  + zpomalení přístupu k datům

1. Systémové programování – Shell, Bash, skripty

**Shell**

* **Řídící struktura** (strukturovaný příkaz, **control flow statement**)
  + konstrukce **pro zápis počítačového programu**,
  + používány ve **vyšších programovacích a skriptovacích jazycích**,
  + rozhodují o **dalším provádění programu**: větví jeho běh, vytváří cykly nebo jinak mění běh programu.
  + 3 druhy řídící struktury
    - **posloupnost příkazů** (sekvence)
    - **větvení**
    - **cyklus**
* **Posloupnost příkazů**
  + nevyžaduje **existenci speciálních řídících struktur**
  + tvořena **sledem klasických příkazů**, které se **vykonávají postupně jeden po druhém** (lineárně)
* **Větvení** - podmínka if-then, if-then else (GOTO, ENDIF), switch a case
  + **podmínka if-then (else)**
    - **podmíněný příkaz**, podmíněná konstrukce
    - prostředek **programovacího jazyka**, umožňující **rozdílné chování programu v závislosti na specifikované logické podmínce**, která je vyhodnocena jako **pravda, či nepravda**
    - IF..GOTO
      * Forma vyskytující se v **nestrukturovaných programovacích jazycích**. **Napodobuje** typickou strojovou instrukci **GOTO**, která umožňuje **skok na určitý řádek kódu**
    - IF..THEN..(ENDIF)
      * Pokud je podmínka v části **IF vyhodnocena jako pravda**, je vykonán kód specifikovaný **v části THEN**. V případě, že je podmínka vyhodnocena jako **nepravda**, je kód specifikovaný v části THEN vynechán a program **pokračuje dále za částí ENDIF**.
    - IF..THEN..ELSE..(ENDIF)
      * Oproti předchozímu výrazu, je v případě vyhodnocení podmínky v části **IF jako nepravda** vykonán kód **specifikovaný v části ELSE**
  + **switch a case**
    - porovnává **předanou hodnotu s předem specifikovanými konstantami**
    - když shoda předané hodnoty s definovanou konstantou, **vykoná příkaz**, nebo příkazy**, které jsou definovány za ní**,
    - pro případ nenalezení schody možnost **ELSE, nebo OTHERWISE**,
    - může umožnit kompilátoru optimalizace, jako například **vyhledávací tabulky**

Soubory systém (minulý semestr)

* **cyklus**
  + skládá z **posloupnosti příkazů** a **podmíněného skoku**, pomocí kterého **se cyklus ukončuje při splnění podmínky**
  + *nekonečný cyklus* – za normálních okolností **není vůbec ukončen**,
  + *cyklus while-do* – cyklus s podmínkou **na začátku posloupnosti příkazů**
  + *cyklus do-while* – cyklus s podmínkou **na konci posloupnosti příkazů**, v případě splnění podmínky ukončen
  + cyklus for – speciální případ cyklu s **podmínkou na začátku**, obvykle užívaný pro výčet prvků z množiny prvků

**Bash**

* Jeden z unixových shellů, který **interpretuje příkazový řádek**
* **POSIX** shell s řadou rozšíření a je koncipován **pro OS založené na projektu GNU**
* Používán jako **implicitní příkazový interpret v systémech postavených na linuxovém jádře**, stejně jako i v Mac OS X nebo v systému Darvin
* Možné **použít i v systému Microsoft Windows** za použití **subsystému** pro unixové aplikace **(SUA**), nebo **emulace POSIX** pomocí softwaru **Cygwin a MSYS**

**Scripty**

* Chceme-li posloupnost jistých **příkazů používat opakovaně**, případně **z různých míst adresářové struktury**, můžeme tuto posloupnost uložit do souboru, který **necháme zpracovat interpretem příkazů** (cesta zadána v prvním řádku: #! /bin/bash)
* Musí mít **právo execute** a volá se pomocí **sh** (*sh jmeno\_skriptu, nebo bash jmeno\_skriptu) (nebo bash)*
* Chceme-li, aby zapsaný skript **spouštěli všichni uživatelé stejným způsobem** jako jiný příkaz v OS Linux, nezadáváme **koncovku .sh a skript umístíme do jedné z cest** (.../bin nebo .../sbin), které se zobrazí po zadání **příkazu echo $PATH**

1. Terminálový a grafický přístup, X Windows Systém, GUI, …

**Terminálový a grafický přístup**

* **Terminál**
  + **spouštění a ovládání programů** (pořizování vstupů/výstupů do/z programu)
  + **Znakový**
    - **klávesnice + obrazovka**
    - připojeno **sériovým kabelem** (modemem, telefonní linkou)
    - řádkový
      * text se **vykresloval z levého horního rohu po řádcích**
      * **editace** **textu** speciálním editorem **(ed)**
    - celoobrazovkový
      * speciální **kódy pro umístění kurzoru** – pokročilejší programy
    - systémová konzole
      * **základní ovládání počítače**,
      * typicky **monitor + klávesnice** přímo **připojené k počítači**
  + **Grafický**
    - využití **grafického uživatelského rozhraní** a **ovládání výstupů na úrovni jednotlivých bodů**
    - dnes emulátory terminálů (**PuTTY**)
  + **CLI** (**příkazový řádek**)
    - Na rozdíl od grafického uživatelského rozhraní (GUI), kde uživatelé používají myš a ikony, CLI vyžaduje zadávání příkazů prostřednictvím klávesnice
* **Grafické rozhraní** – **G**raphic **U**ser **I**nterface (GUI)
  + ovládání počítače **pomocí grafických ovládacích prvků**, kdy na monitoru jsou okna (výstupy programů)
  + oproti CLI **urychlení**
  + UNIX/Linux – základ CLI, **GUI jako nástavba**
  + MS Windows – **základ GUI**, CLI jako doplněk
  + 1973 první rozhraní **WIMP**
  + 1984 **X windows System**

**X Windows systému**

* Navržen jako **nezávislý na platformě a je síťově transparentní**
* Několik základních **komponent** (**X server, X protokol, knihovna Xlib**)
* model klient-server
  + **klient** – **jednotlivé aplikace**
  + **X server** – **zajištění vstupů a výstupů** (ovládání HW – myš, klávesnice, grafická karta, …)
  + komunikace **protokolem IPC, TCP/IP**
  + **knihovna Xlib** – **interakce s X serverem**
* Původně **referenční implementace dle XFree86** a od roku **2004** referenční implementace **dle X.Org**
* **aplikace v podobě oken** => možnost **více aplikací na obrazovce najednou**
* **správa zobrazených oken** -> samostatný správce oken (**window manager**)
* **defaultní implementace** -> správce oken **TWM** (Timeless Window Manager)
* **další správci oken** => IceWM, Window Maker, Fluxbox, Compiz

**GUI**

* **Gnome**
  + Unixové/Linuxové **prostředí pracovní plochy s instalovaným X serverem**
  + původně napsané **pro bitmapový editor GIMP**
  + aplikace pro základní potřeby uživatelů GUI
    - správa souborů (**GNOME Files**)
    - kancelářské aplikace (**GNOME Office**)
    - práce s **internetem** (**GNOME Web, Evolution**)
    - práce s **dokumenty** (**Evince**)
  + distribuce Gnome – Fedora, Red Hat, Ubuntu, Debian, **OPenSUSE**
* **KDE**
  + 1996 bez **otevřené licence** (knihovna Qt) a v 1998 **licence změněna na QPL**
  + z důvodu rozsáhlosti KDE rozděleno **do několika balíčků** (oficiální schéma) a každá **distribuce si vytváří své libovolné vlastní schéma**
  + aplikace pro základní potřeby uživatelů GUI
    - hudební přehrávač (**Amarok**)
    - program na správu fotografií (**digiKam**)
    - program vypalující CD a DVD (**K3b**)
  + distribuce KDE – Mandriva, Debian, **Kubuntu**, **OpenSUSE**
* **Xfce**
  + odlehčené **desktopové prostředí psané na GTK** (malá náročnost)
  + některé **ovládací prvky přejaty** (v light verzi) **z GNOME**
  + 1999 – Xfce 3.0 licencováno pod GNU General Public Licence (GTK+)
  + 2003 – Xfce 4.x (GTK+2)
  + aplikace pro základní potřeby uživatelů GUI
    - aplikace umožňující přidání programu do oznamovací oblasti (**AllTray**)
    - grafický vyhledávací nástroj (**Catfish**)
    - přehrávač hudby (**Exaile**)
    - editace Xfce menu (**Menueditor**)
    - webový prohlížeč (**Midori**)
    - textový editor (**Mousepad**)
  + distribuce - Linux Mint, Debian, **Xubuntu**, **OpenSUSE**, **Kali Linux**
* **Cinnamon**
  + desktopové **prostředí psané na GTK +3**
  + **odnož Gnome**, od r. 2013 samostatný vývoj
  + vyvíjen na (pro) **Linux Mint**, později i na jiných distribucích (**OpenSUSE,** Debian, CentOS, …)
  + aplikace pro základní potřeby uživatelů GUI
    - správce souborů (**Nemo**)
    - textový editor (**Xed**)
    - prohlížeč obrázků (**Xviewer**)
    - prohlížeč dokumentů (**Xreader**)
    - multimediální přehrávač (**Xplayer**)
* **LXDE**
  + „odlehčené“ **desktopové prostředí psané v C** na základě knihoven **GTK+**
  + malá náročnost, funkčnost i na slabých počítačích
  + **funkcionalitou podobné IceWM**
  + správce oken (**Windows manager**) – OpenBox
* **LXQt**
  + open source desktopové prostředí – fúze projektů LXDE a Razor-qt
  + malá náročnost, funkčnost i na slabších počítačích
  + aplikace pro základní potřeby uživatelů GUI
    - webový prohlížeč (**Falkon**)
    - prohlížeč obrázků (**lximage-qt**)
    - souborový manažer (**pacmanfm-qt**)

1. Distribuované systémy a systémy reálného času

**Distribuované systémy**

* Pracuje na **více než jednom procesu** a program je rozdělen **na vzájemně komunikující části**, které mohou **běžet na jiném procesu**
* **Hrubá granularita**
  + části **systému větší**, **samostatnější** a s **malou vzájemnou komunikací**; pro špatnou a pomalou komunikaci
* **Jemná granularita**
  + části systému **co nejmenší s velkou vzájemnou komunikací**
* **Distribuované systémy na uživatelské úrovni**
  + systém běžící na **více propojených počítačích** a každý má **vlastní OS**
  + podpora distribuce umístěna na **sw vrstvě** nad **nedistribuovaným OS**
* **Distribuované operační systémy**
  + samostatný **OS běžící na síti procesorů**, které **nesdílejí společnou paměť**
  + pro uživatele **jako jeden počítač** (systém se tváří jako jednoprocesorový)
  + **architektura** (většina DOS) založena na modelu **klient/server**
  + asymetrický
    - **vybrané počítače určeny jako servery** (file server, print server, …); ostatní – běh uživatelských programů
  + symetrický
    - **každý počítač může být jak server pro jiné počítače, tak klientem jiných serverů** (vyšší decentralizace, lepší využití prostředků, rozšiřitelnost)
* **Výhody**
  + **Ekonomika** – mikroprocesory poskytují lepší poměr ceny a výkonu
  + **Výkon** – muže dosáhnout lepšího výkonu, než potenciálně jakýkoliv sálový PC
  + **Spolehlivost** – při selhání části systému muže zbytek pokračovat v činnosti
  + **Rozšiřitelnost** – kapacita a výkon systému může být po částech zvyšovány
  + **Sdílení** – sdílení společných dat a periferií
  + **Vzdálený běh procesu** – sdílení a vyvažování výpočtové zátěže
* **Nevýhody**
  + **Software** – málo existujícího software, velké požadavky na hardware
  + **Připojení do sítí** – síť může být zahlcena
  + **Bezpečnost** – snadný přístup k datům – vyvolává nutnost utajení
* **Požadované funkce**
  + efektivní řízení přidělování prostředků dostupných na síti
  + skrytí fyzického rozmístění prostředků a umožnění bezpečné komunikace
  + poskytnutí mechanismu ochrany systémových prostředků před přístupem neautorizovanými uživateli
* **Požadované vlastnosti**
  + **Transparentnost** – **pro uživatele i pro procesy dojem jednotlivosti systému**
    - Přístupová
      * proces jednotným způsobem **přistupuje k lokálním i vzdáleným prostředkům**
    - Lokační
      * proces **nemusí znát fyzické umístění prostředku**
      * ve jménech prostředku nesmí být zakódováno jejich umístění
    - Migrační
      * prostředky **jsou beze změny jména a bez vlivu na procesy libovolně přesouvány** a **připojovány** k různým částem celého systému
      * OS musí zabezpečit, aby po **dobu přemisťování prostředku** (např. souboru na jiný server) byly **požadavky pozdrženy** a poté  **přesměrovány na nové místo**
    - Exekuční
      * procesy mohou **běžet na jakémkoliv procesoru** a při běhu **mohou být přesunuty na jiný procesor** -> např. vyrovnávání zátěže různých částí systému,
    - Replikační
      * umožňuje **vytváření kopií souboru bez vědomí uživatele**
      * velmi důležité **zejména u rozsáhlých systému**, kde by přístup např. k souboru fyzicky umístěnému na jiném kontinentu trval nepřiměřeně dlouhou dobu
    - Konkurenční
      * přístup k prostředku i **když je využíváno jiným**
    - Paralelizmová
      * OS by měl teoreticky sám **využít všech dostupných procesorů** tak, aby celkový čas provádění byl co nejmenší
  + **Flexibilita**
    - každá část systému **co nejvíce samostatná** a možnost **přemístění prováděných procesů na méně vytížené procesory či mezi částmi systému**
  + **Spolehlivost**
    - při výpadku komponent **systému si ostatní rozdělí práci**
  + **Výkonnost**
    - běh aplikace **nesmí být výrazně pomalejší než v jednoprocesorovém systému**; musí být výkonnější HW
  + **Rozšiřitelnost**
    - schopnost **rozšíření** o **jakékoliv** (teorer.) **množství procesorů**

**Systémy reálného času**

* je systém, ve kterém je správnost výstupu závislá nejen na správnosti výsledku výpočtu, ale též na čase, v němž je výsledek spočten
* “okamžitá“ odezva - tzv. horní časová hranice, ve které je třeba požadavky zpracovat
* Nevhodné pro desktopy - požadavky RTOS výkonově náročné,
* Realtimová priorita i v běžných systémech = výrazně vyšší než běžný proces
* Řízení letadel, elektráren, automobilovém průmyslu a telekomunikace
* Podle času přiblížení se “real time” dělíme:
  + **Hard Real Time OS** - požadavek času na reakci absolutní (katastrofa)
  + **Soft Real Time OS** - povolené drobné odchylky v reakcích
* Plánovač RTOS specifickým způsobem přiděluje systémové prostředky
  + minimální latence při reakci na událost a při přepínání vláken
  + minimální časových okamžiků se zákazem přerušení
  + např.: RMS a EDF
* PIke OS - speciálně vyvinutý OS, založen na mikrojádru, pro bezpečnostně kritické aplikace na převážně vestavěných systémech (letectví, automobilová technika,...)
* QNX - několik důležitých serverů je na malém mikrojádru a zbytek jako procesy, tím pádem je stabilní, rychlí i na grafickém rozhraní; podpora sítě
* RTLinux - je systém, ve kterém je správnost výstupu závislá nejen na správnosti výsledku výpočtu, ale též na čase, v němž je výsledek spočten (L. jádro jako proces)
* OS Windows - v základní konfiguraci NELZE použít jako Real-time system (moduly)
* RTX - modul rozšiřující možnosti Win směrem k RealTime systémům
* WxWorks 6.x - -nejrozšířenější RTOS v průmyslových aplikací vestavěných systémů a je bezpečný i v kritických aplikačních úlohách + možnost použití na běžných CPU + porovnat dis a real time sys.

Dopsat porovnání mezi Distribučními a real time systémy…

1. Adresářové služby – popis, modely

**Adresářová služba (Directory Service – DS)**

* aplikace (skupina aplikací) **pro ukládání a organizaci údajů** (informací) **o uživatelích a zdrojích v počítačové síti**
* centrální **autentizační autorita** (uživatelé, služby, počítače, …)
* přistupuje **k adresáři** (directory repository) -> informace **pro administrátory, uživatele, aplikace**, …
* vytváří **fyzickou síťovou topologii** -> transparentnost pro uživatele
* **součást OS nebo samostatná aplikace**

**Adresář (directory)**

* **databáze** (hierarchická struktura)
  + uložení **informací o pojmenovaných objektech na síti**
  + **organizování** a **sdružování** do skupin (např. počítač, tiskárna, služba, doména, účet uživatele, …)
  + uchovávání **dat různých typů** (text, obrázek, certifikát,…)
  + časté čtení, vyhledávání, občasný zápis => **není to relační db**
  + možnost omezení přístupu pomocí **ACL** (Access Control List)
* **X.500**
  + sada mezinárodních standardů **pro adresářové služby v počítačových sítích**
  + vyvinuty pro **systém el. pošty X.400**
  + zahrnuto do standardů **protokolů OSI** jako standard **ISO/IEC 9594**
  + definování protokolů:
    - **DAP** (Directory Access Protocol)
    - **DSP** (Directory System Protocol)
    - **DISP** (Directory Information Shadowing Protocol)
    - **DOP** (Directory Operational Bindings Management Protocol)
  + v praxi komplexní model neprosazen (velikost a složitost)
* **LDAP (Lightweight Directory Access Protocol)**
  + **zjednodušení** (odlehčení – lightweight) **standardu X.500 + zaměření na TCP/IP**
  + **aplikační** protokol pro **dotazování** a **modifikaci** adresářových služeb nad TCP/IP
  + definuje **způsoby modifikace** a **přístupy** k **datům** na adresářovém serveru
  + použití **LDAP Data Interchange Format** (LDIF)
  + data kódována pomocí **Lightweight Basic Encoding Rules (LBER)**
  + popis pomocí čtyř modelů
    - **informační model – schéma**
      * informace v adresáři **uloženy ve stromové struktuře** (Directory Information Tree – DIT)
      * **kořen stromu *rootDSE***(globání informace o adresáři)
      * další záznamy obsahují **informace o jednotlivých objektech** -> objekt – složen z atributů
      * implementace -> schéma (sada všech možných definovaných tříd objektů a všech atributů)
      * *třída objektů (objectClass)*
        + **kategorie objektů v adresáři** (user, computer, organizationalUnit, domain, container, group, …)
        + kategorie: Structural, Abstract, Auxiliary
        + **objekt může být zařazen ve více třídách najednou** => nejednoznačné vyhledávání ->
      * *kategorie objektů (objectCategory)*
        + oproti objectClass **pouze jedna hodnota** => jednodušší vyhledávání
      * *atributy objektů (object attributes)*
        + může **obsahovat jednu nebo více hodnot** (jméno, příjmení, email, …)
      * Obsah obrázku text, snímek obrazovky, číslo, Písmo

        Popis byl vytvořen automaticky
    - **jmenný model**
      * **jednoznačná identifikace objektů**
      * Distinguished Name (DN)
        + jednoznačný **identifikátor objektu**
        + **úplná cesta k záznamu** (pozice ve stromě)
      * Relative Distinguished Name (RDN)
        + když **není** třeba specifikovat celou cestu
        + **poslední část DN**

Obsah obrázku snímek obrazovky, diagram, řada/pruh, kruh

Popis byl vytvořen automaticky

* + - * jmenné atributy
        + **každá část DN je vyjádřena pomocí *typ\_atributu****=hodnota*
        + každá třída má **přiřazen jmenný atribut** -> *User* má *cn*

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automaticky

* + - **funkční model**
      * definice práce s informacemi v adresáři
      * Obsah obrázku text, snímek obrazovky, Písmo, číslo

        Popis byl vytvořen automaticky
    - **bezpečnostní model**
      * určuje**, jak se přistupuje k datům z hlediska bezpečnosti**
      * souvisí s autentizačními službami adresářových služeb
      * autentizace
        + ***anonymní*** – v rámci operace bind nejsou **serveru předávány žádné informace nesoucí identitu či heslo uživatele**
        + ***jednoduchá*** – pomocí **DN a hesla** (atributu userpassword)
        + ***proxy*** – využívá **existence definovaného uživatele**, který má **právo nahlížet na hesla ostatních uživatelů**
        + ***PKI*** – založená na **principu PKI digitálních certifikátů**, které jsou uloženy v definovaném atributu userCertificate
        + ***SASL mechanismus***– pomocí **zásuvných modulů**
      * autorizace
        + po **ukončení autentizace**
        + přístup autentizovaného uživatele **jen k datům a operacím dle jeho oprávněn**í