Operační systémy

IOS 2023/2024

Tomáš Vojnar

vojnar@fit.vutbr.cz

Vysoké učení technické v Brně
Fakulta informačních technologií
Božetěchova 2, 612 66 Brno

Unix – úvod

Historie UNIXu

❖ 1961: CTSS (Compatible Time-Sharing System):

- vyvinut na MIT, jeden z prvních OS podporujících preemptivní sdílení času (další takový OS byl Plato II z University of Illinois),
- jeden z prvních OS s emailem, zárodkem shellu, jedním z prvních programů pro formátování textu (RUNOFF, předchůdce nroff/groff).

❖ 1965: MULTICS – Multiplexed Information and Computation Service (Bell Labs, MIT, General Electric):

- v zásadě neúspěšný OS, „zhroutil se vlastní vahou“ (přílišná obecnost, přílišný rozsah služeb, přespecifikovanost), měl ale významný vliv na další vývoj OS,
- hierarchický souborový systém, paměťově mapované soubory, dynamické linkování, ACLs, dynamická rekonfigurace, předchůdce shellu, ...

❖ 1969: Začátek vývoje nového OS – PDP 7 (K. Thompson, D. Ritchie a několik jejich kolegů z Bell Labs, AT&T).

❖ 1970: Zavedeno jméno UNIX (původně UNICS).

❖ 1971: PDP-11 (24KB RAM, 512KB disk) – text processing.

- 1972: asi 10 instalací.
- 1973: UNIX přepsán do C.
- 1973/74: Článek: „Unix Timesharing System“, asi 600 instalací.
- 1977: Berkeley Software Distribution – BSD.
- 1978: SCO (Santa-Cruz Operation) – první Unixová společnost.
- 1979: UNIX Version 7 – verze UNIXu blízká současným verzím.
- 1980:
  - DARPA si vybrala UNIX jako platformu pro implementaci TCP/IP.
  - Microsoft a jeho XENIX.
- 1981: Microsoft, smlouva s IBM, QDOS, MS-DOS.
- 1982: Sun Microsystems.
- 1983:
  - AT&T UNIX System V.
  - BSD 4.2 – síť TCP/IP.
  - GNU, R. Stallman.

- 1984: X/OPEN XPG.
- 1985: POSIX (IEEE).
- 1987: AT&T, SUN: System V Release 4 a OSF/1.
- 1990: Windows 3.0.
- 1991: Solaris, Linux.
- 1992: 386BSD.
- 1994: Single Unix Specification (The Open Group).
- 1998:
  - začátek prací na sloučení základu SUS a POSIX,
  - open source software.
- 2002: SUS v3 – zahrnuje POSIX, poslední revize 2008 (SUS v4, rev. 2018).

- 2015: Spolupráce Red Hat a Microsoft v oblasti cloudových řešení a podpory .NET.
- 2018: Microsoft Azure Sphere – verze Linuxu pro IoT aplikace.
- 2019: Windows Subsystem for Linux 2 – Linuxové jádro nad Windows 10.

Příčiny úspěchu

- Mezi příčiny úspěchu UNIXu lze zařadit:
  - víceprocesový, víceuživatelský – v době vzniku velmi pokrokové,
  - napsán v C – přenositelný,
  - zpočátku (a později) šířen ve zdrojovém tvaru,
  - „mechanism, not policy“,
  - „fun to hack“,
  - jednoduché uživatelské rozhraní,
  - skládání složitějších programů z jednodušších,
  - hierarchický systém souborů,
  - konzistentní rozhraní periferních zařízení, ...

- Řada z těchto myšlenek je inspirující i mimo oblast OS.

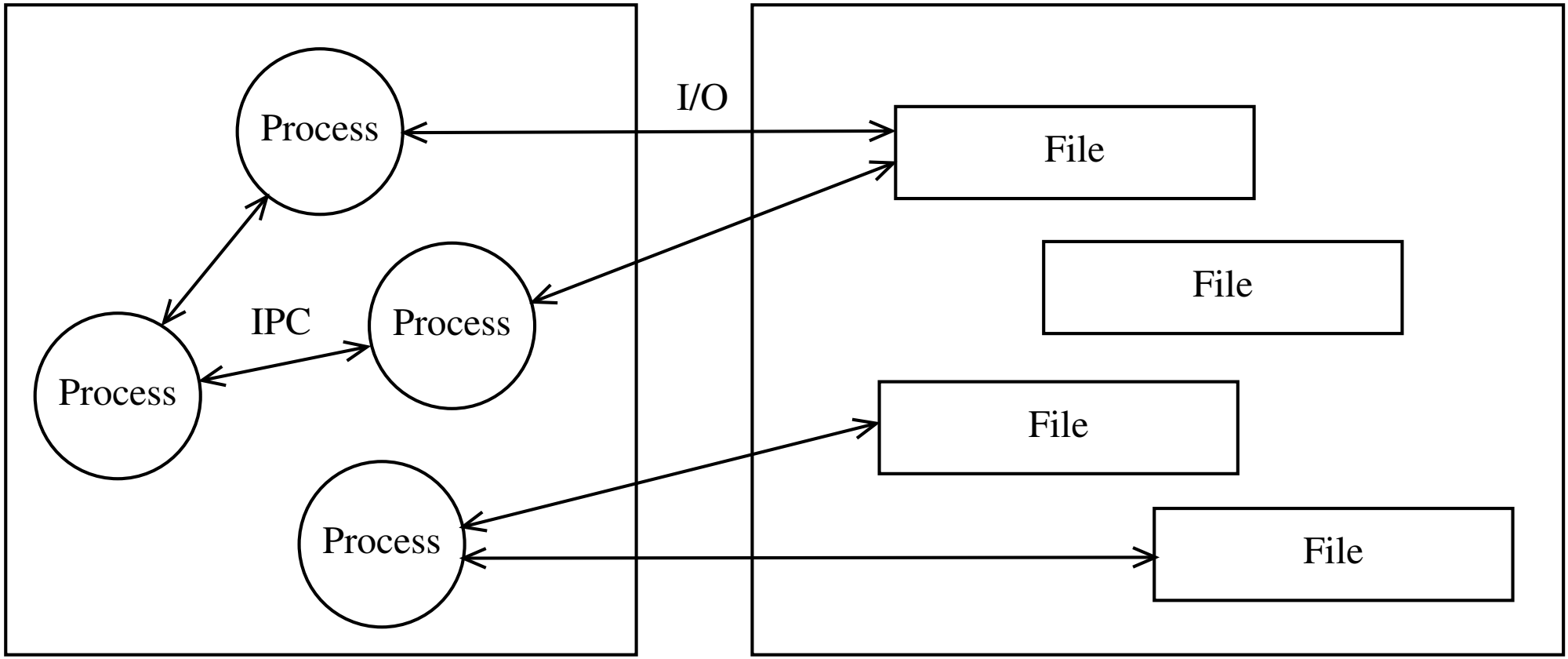
Varianty UNIXu

- Hlavní větve OS UNIXového typu:
  - UNIX System V,
  - BSD UNIX,
  - různé firemní varianty (AIX, Solaris, ...),
  - Linux.

- Související normy:
  - XPG – X/OPEN,
  - SVR4 – AT&T a SUN,
  - OSF/1,
  - Single UNIX Specification,
  - POSIX – IEEE standard,
  - Single UNIX Specification v4, edition 2018 – shell a utility (CLI) a API.

Základní koncepty

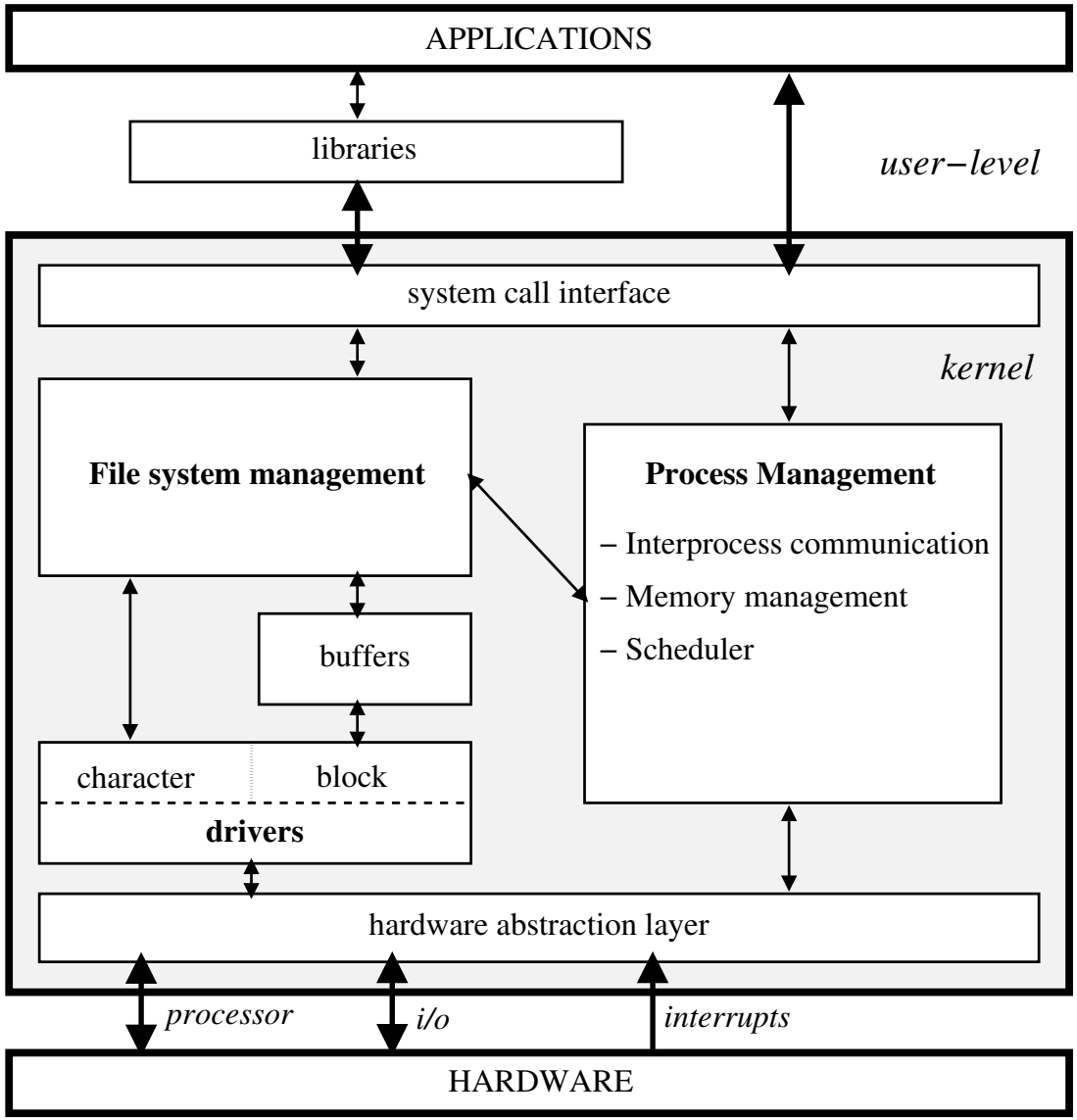
- Dva základní koncepty/abstrakce v UNIXu: procesy a soubory.



IPC = Inter-Process Communication – roury (pipes), signály, semafory, sdílená paměť, sockets, RPC, zprávy, streams...
I/O = Input/Output.

Struktura jádra UNIXu

- Základní podsystémy UNIXu:
  - Správa souborů
    (File Management)
  - Správa procesů
    (Process Management)



Komunikace s jádrem

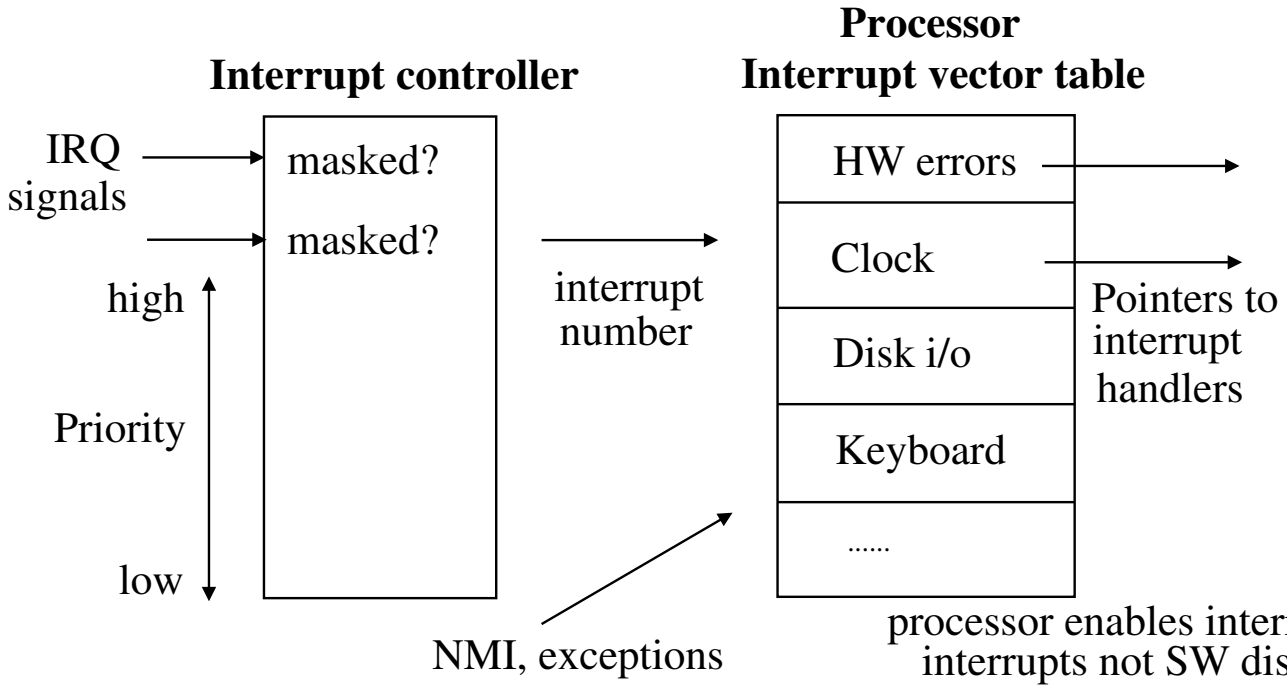
❖ Služby jádra – operace, jejichž realizace je pro procesy zajišťována jádrem. Explicitně je možno o provedení určité služby žádat prostřednictvím systémového volání („system call“).

❖ Příklady některých služeb jádra dostupných systémovým voláním v UNIXu:

služba	jaká operace se provede
open	otevře soubor
close	zavře soubor
read	čte ze souboru
write	zapisuje
kill	pošle signál
fork	duplikuje proces
exec	přepíše kód
exit	ukončí proces

❖ HW přerušení (hardware interrupts) – mechanismus, kterým HW zařízení oznamují jádru asynchronně vznik událostí, které je zapotřebí obsloužit.

- Žádosti o HW přerušení přichází jako elektrické signály do řadiče přerušení.
  Na PC dříve PIC, nyní APIC (distribuovaný systém: každý procesor má lokální APIC, externí zařízení mohou být připojena přímo nebo přes I/O APIC – součást chip setu).
- Přerušení mají přiřazeny priority, dle kterých se oznamují procesoru.
- Přijme-li procesor přerušení s určitým číslem (tzv. interrupt vector) vyvolá odpovídající obslužnou rutinu (interrupt handler) na základě tabulky přerušení, přičemž automaticky přejde do privilegovaného režimu.
- Řadič může být naprogramován tak, že některá přerušení jsou maskována, případně jsou maskována všechna přerušení až po určitou prioritní úroveň.



❖ Příjem nebo obsluhu HW přerušení lze také zakázat:

- na procesoru (instrukce CLI/STI na Intel/AMD),
- čistě programově v jádře (přerušení se přijme, ale jen se poznamená jeho příchod a dále se neobsluhuje).

❖ NMI (non-maskable interrupt): HW přerušení, které nelze zamaskovat na řadiči ani zakázat jeho příjem na procesoru (typicky při chybách HW bez možnosti zotavení: chyby paměti, sběrnice apod., někdy také pro ladění či řešení uváznutí v jádře: "NMI watchdog").

❖ Přerušení také vznikají přímo v procesoru – synchronní přerušení, výjimky (exceptions):

- trap: po obsluze se pokračuje další instrukcí (breakpoint, přetečení apod.),
- fault: po obsluze se (případně) opakuje instrukce, která výjimku vyvolala, resp. pamatuje se adresa problematické instrukce (např. výpadek stránky, dělení nulou, chybný operační kód apod.),
- abort: nelze určit, jak pokračovat, provádění se ukončí (např. některé zanořené výjimky typu fault, chyby HW detekované procesorem – tzv. „machine check mechanism").

- Při obsluze přerušení je zapotřebí dávat pozor na přerušení obsluhy přerušení:
  - Nutnost synchronizace obsluhy přerušení tak, aby nedošlo k nekonzistencím ve stavu jádra díky interferenci částečně provedených obslužných rutin.
  - Využívají se různé mechanismy vyloučení obsluhy přerušení (viz předchozí slajd).
  - Při vyloučení obsluhy je zapotřebí věnovat pozornost době, po kterou je obsluha vyloučena:
    - zvyšuje se latence (odezva systému),
    - může dojít ke ztrátě přerušení (nezaznamenají se opakované výskyty),
    - výpočetní systém se může dostat do nekonzistentního stavu (zpoždění času, ztráta přehledu o situaci vně jádra, neprovedení některých akcí v hardware, přetečení vyrovnávacích pamětí apod.).
  - Vhodné využití priorit, rozdělení obsluhy do více částí (viz další slajd).

❖ Obsluha přerušení je často rozdělena na dvě úrovně:

- 1. úroveň:
  - Zajišťuje minimální obsluhu HW (přesun dat z/do bufferu, vydání příkazů k další činnosti HW apod.) a plánuje běh obsluhy 2. úrovně.
  - Během obsluhy na této úrovni může být zamaskován příjem dalších přerušení stejného typu, stejné či nižší priority, případně i všech přerušení.
- 2. úroveň:
  - Postupně řeší zaznamenaná přerušení, nemusí zakazovat přerušení.
  - Obsluha může běžet ve speciálních procesech.
  - Vzájemné vyloučení se dá řešit běžnými synchronizačními prostředky (semafory apod. – nelze u 1. úrovně, kde by se blokoval proces, v rámci jehož běhu přerušení vzniklo).
  - Mohou zde být samostatné úrovně priorit.

❖ Mohou existovat i další speciální typy přerušení, která obsluhuje procesor zcela specifickým způsobem, často mimo vliv jádra. Např. na architekturách Intel/AMD:

- Interprocessor interrupt (IPI): umožňuje SW běžícímu na jednom procesoru poslat určité přerušení jinému (nebo i stejnému) procesoru – využití např. pro zajištění koherence některých vyrovnávacích pamětí ve víceprocesorovém systému, předávání obsluhy přerušení či v rámci plánování.
- System management interrupt (SMI): generováno HW i SW, způsobí přechod do tzv. system management mode (SMM) procesoru.
  - V rámci SMM běží firmware, který může být využit k optimalizaci spotřeby energie, obsluhu různých chybových stavů (přehřátí), rekonfiguraci HW apod.
  - Po dobu běhu SMM jsou prakticky vyloučena ostatní přerušení, což může způsobit zpožďování času v jádře a případné další nekonzistence stavu jádra oproti okolí, které mohou vést k pádu jádra.
- HW signály jako RESET, INIT, STOPCLK, FLUSH apod.

- Ovladače zařízení a přerušení:
  - Při inicializaci ovladače (v Linuxu typicky modul), nebo jeho prvním použití, se musí registrovat k obsluze určitého IRQ.
  - Příslužné IRQ je buď standardní, zjistí se přes konfigurační rozhraní sběrnic komunikací s jejich řadičem, nebo sondováním (zařízení je vyzváno ke generování přerušení a sleduje se, ke kterému dojde).
  - IRQ může být sdíleno: jsou volány všechny registrované obslužné rutiny a musí být schopny komunikací s příslušným zařízením rozpoznat, zda přerušení bylo vygenerováno příslušným zařízením.

- Základní statistiky o obsluze přerušení v Linuxu: `/proc/interrupts`.

- Příklad komunikace s jádrem:
  - synchronní: proces-jádro
  - asynchronní: hardware-jádro