

17 Procesy

Obsah hodiny



Obsahem této hodiny je popis životního cyklu procesu od vzniku procesu až po jeho ukončení.

Cíl hodiny



Po prostudování budete schopni:

- vysvětlit, co je to proces, jak vzniká, jakým způsobem se identifikuje
- popsat životní cyklus procesu a stavy, kterými proces prochází
- objasnit pojem swapování

Klíčová slova



Proces, Program, Swapping, Thread, Multithreading, Primární vlákno, Životní cyklus procesu.

17.1 Proces, stavy procesu

Proces (process, task) je běžící program¹, vzniká v okamžiku, kdy spustíme program (aplikaci, příkaz). Pro svoji realizaci potřebuje zdroje (procesor, paměť, I/O zařízení ...).

Odbornější definice říká, že proces je "určitý měřitelný adresový prostor, s n vlákny běžícími v rámci tohoto určitého adresového prostoru, a systémové zdroje pro tato vlákna"

¹ Program sám o sobě je pouze zápis algoritmu v nějakém programovacím jazyce (například ve strojovém kódu). Je statický, neměnný (neuvažujeme-li vývoj nových verzí programů). Proces je dynamický.

Procesy se v některých OS mohou rozdělit na několik souběžných částí – vláken (threads), jedná se o multithreading. Vlákno (thread) je vlastně funkce, která může volat další funkce. Jedno z vláken takového procesu je primární vlákno obsluhuje uživatelský vstup a podle potřeby vytváří další vlákna. Vlákna jednoho procesu sdílejí společný adresní prostor paměti a mohou spolu komunikovat pomocí této sdílené paměti.

Nepodporuje-li systém multithreading, znamená to, že každý proces je tvořen právě jedním threadem.

Aby proces mohl být zpracován procesorem, musí se umístit do operační paměti (RAM). Z RAM se načte do registru CPU, provádí se „výpočet“ na CPU. Po jeho ukončení nebo přerušení se obsah registrů přesune zpět do RAM. Operační paměť pak obsahuje výsledky výpočtu.

Každý proces je definován:

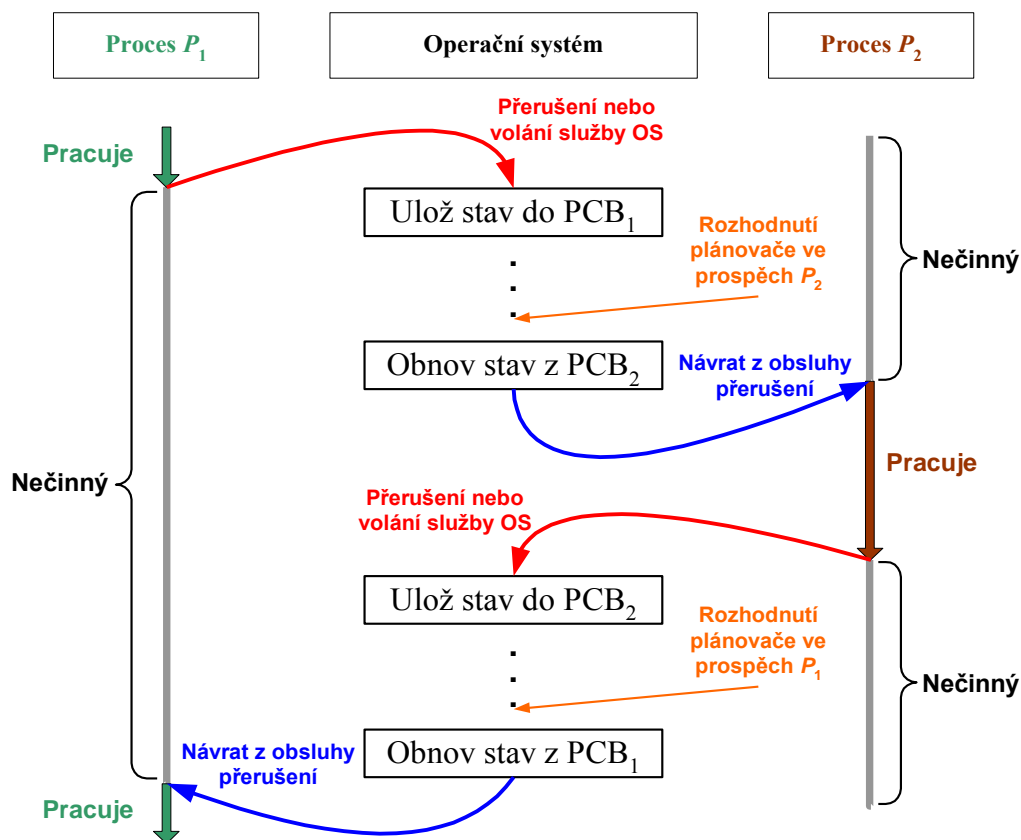
- **Stavem procesoru** (hodnoty registrů): Proces, který běží na procesoru (CPU) je uložen v registrech CPU_ registry obsahují aktuální stav právě probíhajícího výpočtu.
- **Adresovým prostorem** (obsah paměti):
 - instrukce programu, tj.kód aplikace,
 - statická data aplikace (definovaná při překladu), typicky konstanty, globální proměnné
 - heap: „natahovací“ oblast, ze které si může proces „ukusovat“ paměť za běhu podle potřeby (pomocí operátoru new, resp. funkce malloc)
 - stack: lokální úložiště pro registry CPU a lokální proměnné funkcí.
- **Prostředím** (struktury operačního systému) Proces neexistuje ve vakuu. Je např. vázán na uživatelský terminál, na otevřené soubory používané pro vstup a výstup, na komunikační kanály a sokety (spojení mezi procesy).

Informace o procesech jsou uloženy v Deskriptoru procesu – *Process Control Block* (PCB). Jedná se o tabulku, která obsahuje pro každý spuštěný proces je jeden záznam. Mezi položky záznamu patří např.

- identifikátor procesu
- stav procesu,
- priorita,
- ukazatel na místo v RAM, kde je proces umístěn,
- informace o souborech, které proces užívá,
- údaje o čase procesoru, který proces spotřeboval,
- informace potřebné pro plánování procesoru(ů),
- proměnné prostředí,

— ...

Na základě těchto informací se provádí přepínání mezi procesy:



Obrázek 17-1 Přepínání mezi procesy

17.2 Životní cyklus procesu

Proces je vytvořen příkazem uživatele, na žádost jiného procesu (rodičovského) nebo na žádost operačního systému o provedení služby

Nový proces **vzniká** jako kopie rodičovského procesu (toho, na jehož žádost vznikl). Ve vzniklé kopii/klonu je původní obsah rodičovského procesu (instrukce, data ...) nahrazen novým.

Vytváření synovských procesů je v různých OS implementováno různě. V OS UNIX má každý proces jedinečné identifikační číslo (PID). Nový proces (kopie rodiče) je vytvořen systémovým voláním jádra fork. Přepsání adresového prostoru procesu novým programem provádí jiné systémové volání jádra - exec. Oba procesy jsou schopny komunikovat a každý jde svou cestou. Mezi procesy tak zůstává zachována vazba rodič – potomek a procesy vytvářejí hierarchickou strukturu.

Z hlediska vlastníka (toho, kdo proces spustil) můžeme procesy rozdělit na:

- **uživatelské** - ty, co spustil libovolný uživatel,
- **systémové** - ty, co spustil systém při startu nebo na základě nějaké události.

Proces může být **ukončen** tak, že se vykonají všechny instrukce, proces normálně celý proběhl a skončil nebo násilně, např.

- uživatelem,
- provedením chybné instrukce,
- chybou V/V zařízení,
- vypršením časového limitu,
- na žádost rodiče.

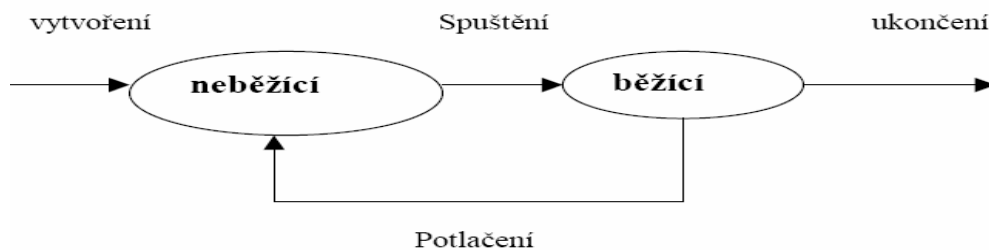
Dobu od vzniku po ukončení procesu označujeme jako životní cyklus procesu a proces během té doby prochází různými stavy:

- nový (*new*) – proces je právě vytvářen,
 - běžící (*running*) – program řídící tento proces je právě vykonáván, tj. interpretován některým procesorem,
 - čekající (*waiting, blocked*) – proces čeká na jistou událost,
 - připravený (*ready*) – proces čeká na přidělení procesoru,
 - ukončený (*terminated*) – proces ukončil svoji činnost, avšak stále ještě existuje
- Přechody mezi stavy procesu v rámci životního cyklu procesu lze graficky znázornit pomocí stavových modelů.

17.2.1 Dvoustavový model procesu

Nejjednodušší je **model dvoustavový**. Proces se nachází ve dvou stavech:

- neběžící = připravený (*ready*)
- běžící (*running*)

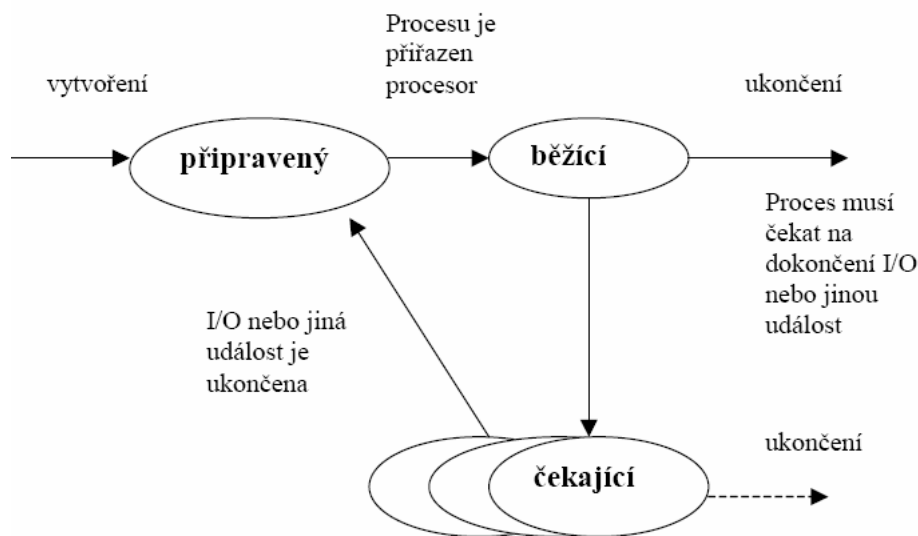


Obrázek 17-2 Dvoustavový model

Po vytvoření se nový proces řadí do fronty mezi **procesy připravené** a čeká na procesor. Když na něj přijde řada, je zpracován procesorem a stává se **procesem běžícím** (procesor může zpracovávat pouze jeden proces), po vykonání všech instrukcí je proces ukončen, pokud ne, je potlačen a vrací se mezi procesy připravené.

17.2.2 Třístavový model procesu

Model třístavový zachycuje potlačení procesu v okamžiku, kdy proces čeká na nějakou událost. Není tedy připraven ke zpracování procesorem a až dojde k dané události, bude zařazen mezi připravené procesy .



Obrázek 17-3 Třístavový model

Proces se může nacházet ve třech stavech:

- **připravený** (*ready*) proces je připravený k vykonání a čeká pouze na přidělení procesoru.
- **běžící** (*running*) procesu je přidělen procesor a právě se provádí
- **čekající** (*waiting*) proces je blokový, tj. čeká na určitou událost, např. na dokončení I/O operace.

17.2.3 Pětistavový model

Třístavový model má jeden nedostatek. Nezachycuje situaci, kdy dojde k obsazení operační paměti (RAM) a další proces už se do paměti nevejde.

Protože kapacita operační paměti je omezená a příliš mnoho procesů v operační paměti snižuje výkonnost, umožňují OS **swapping**, tj. odložení některých procesů na disk.

Stavový model se tak rozšířil o další dva stavy:

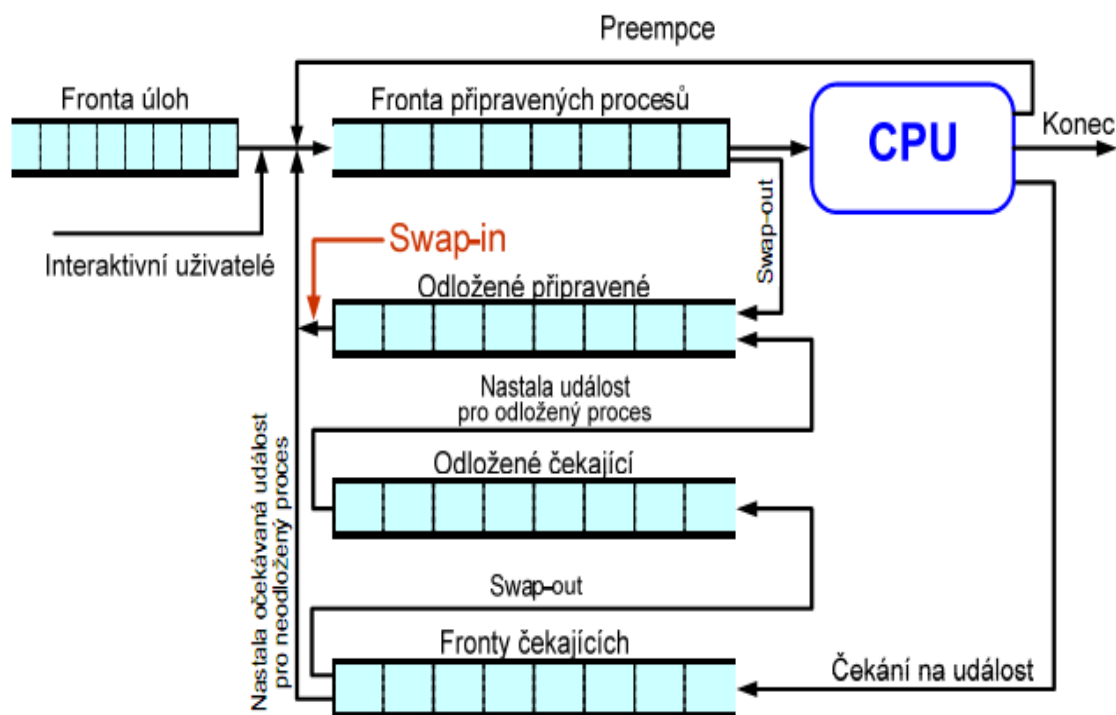
- odložený čekající,
- odložený připravený

Odložené procesy uvolňují operační paměť a přecházejí do stavu odložený čekající v případě, že

- je mnoho čekajících procesů,
- vlastník procesu si to přeje nebo to předpisuje časový plán,
- přeje si to rodič z důvodu synchronizace sourozenců.

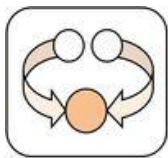
V případě, že se stala očekávaná událost, přechází proces ze stavu odložený čekající do stavu odložený připravený.

V případě, že se fronta připravených vyprázdnila (nebo alespoň téměř vyprázdnila), pak přechází proces ze stavu „odložený připravený“ do stavu „připravený“.



Obrázek 17-4 Pětistavový model

Shrnutí kapitoly



Proces (process, task) je běžící program. Může být tvořen jedním nebo více vlákny (mutithreading). Vlákno je součástí procesu, představuje konkrétní běžící výpočet – aktivitu. Proces může obsahovat jedno vlákno nebo více vláken.

U vícevláknového procesu (těžký proces) je jedno vlákno primární a vytváří podle potřeby další vlákna.

Aby proces mohl být zpracován procesorem musí se umístit do operační paměti (RAM), odtud se načte do registru CPU, provádí se „výpočet“ na CPU a po jeho ukončení nebo přerušení obsah registrů přesune zpět do RAM. Operační paměť pak obsahuje (dosavadní) výsledky výpočtu.

Proces je jednoznačně rozpoznatelný (unikátní PID - Process ID):

- Každý proces má kód (instrukce programu), data, proměnné, otevřené soubory., lokální úložiště pro registry CPU ...
- Používá systémové prostředky počítače, které mu přidělil operační systém (dočasně „vlastní“ tyto prostředky)

Jeden proces spouští další, mezi procesy vznikají vztahy rodič – potomek, procesy tak mají hierarchickou strukturu. Nový proces vzniká jako kopie procesu rodičovského. Proces zaniká tak, že normálně doběhne nebo může být ukončen násilně.

Za svého života (životní cyklus procesu) prochází proces až pěti stavy:

- běžící (*running*)
- čekající (*waiting, blocked*)
- připravený (*ready*)
- odložený čekající
- odložený připravený

Poslední dva stavy ošetřují situaci, kdy je plně vytížená operační paměť a dochází ke swapování, tj. procesy jsou odkládány na disk.

Kontrolní otázky a úkoly



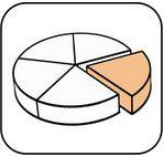
- 1) Co je to proces?
- 2) Co je multithreading?
- 3) Jak vypadá proces v režimu multithreadingu?
- 4) Jak vznikají a zanikají procesy?
- 5) Co je PID?
- 6) Jaké jsou vztahy mezi procesy
- 7) Jaké stavové modely používáme pro popis životního cyklu procesů?
- 8) Popište pětistavový model a jeho pět stavů?

Otázky k zamyšlení



- 1) Jaký je rozdíl mezi programem a procesem?
- 2) Jaké výpočetní prostředky proces potřebuje?

Použitá literatura a jiné zdroje:



- [1] KLIMEŠ, Cyril. Principy výstavby počítačů a operačních systémů. Ostrava : Kovosil, 2007. 198 s. ISBN 978-80-903694-1-2.
- [2] LAŽANSKÝ, J. Operační systémy a jejich aplikace - X33OSA: Výpočetní procesy a jejich správa. Labe.felk.cvut.cz [online]. 20.10.2010 [cit. 2011-10-09]. Dostupné z: <http://labe.felk.cvut.cz/vyuka/X33OSA/>