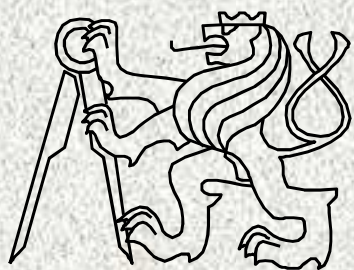


VLÁKNA



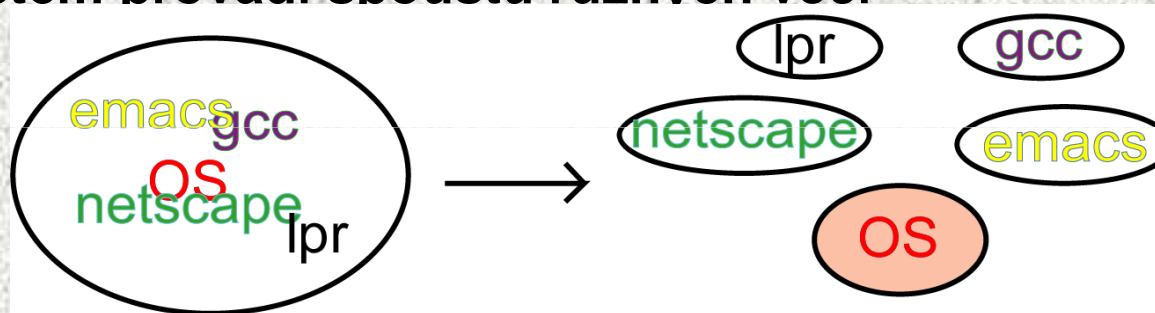
A0B36PR2-Programování 2

Fakulta elektrotechnická

České vysoké učení technické

Procesy

- Veškerý **běžící software** v systému je organizován jako **množina** “sekvenčně” běžících procesů.
- Proč procesy?
 - **Systém provádí spoustu různých věcí**



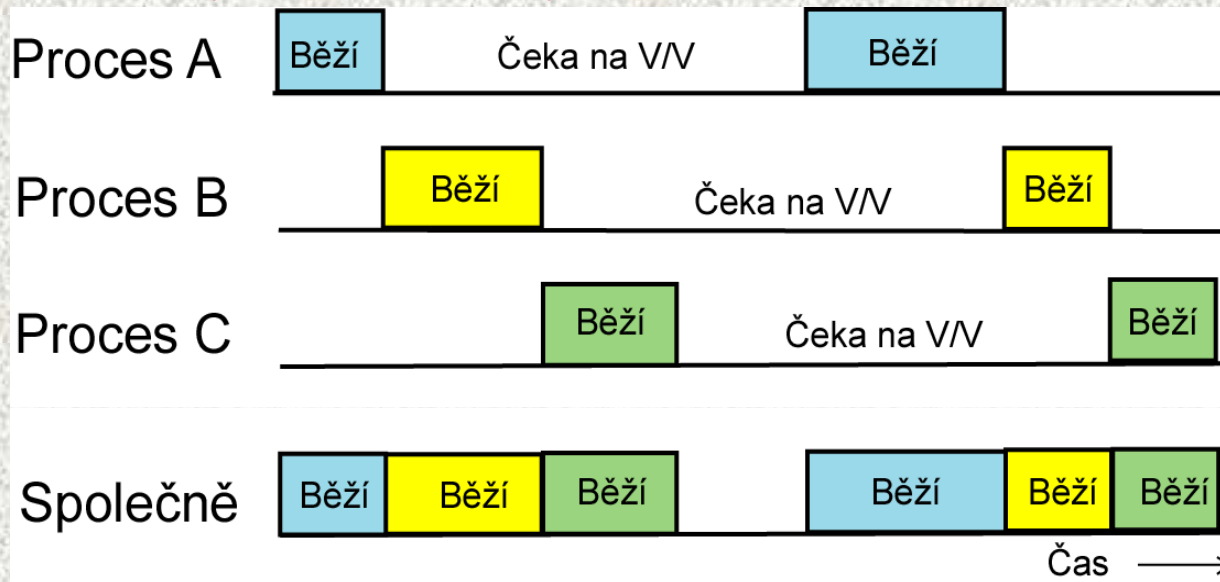
1. Jednoduchost - jak to zjednodušit?

- **Z každé jednotlivé akce udělat izolovaný proces.**
- **OS se zabývá v jednom okamžiku pouze jednou věcí.**
- **Univerzální trik pro správu složitých problémů:**
 - **dekompozice problému!**

Procesy (2)

2. Proč procesy?

Rychlost

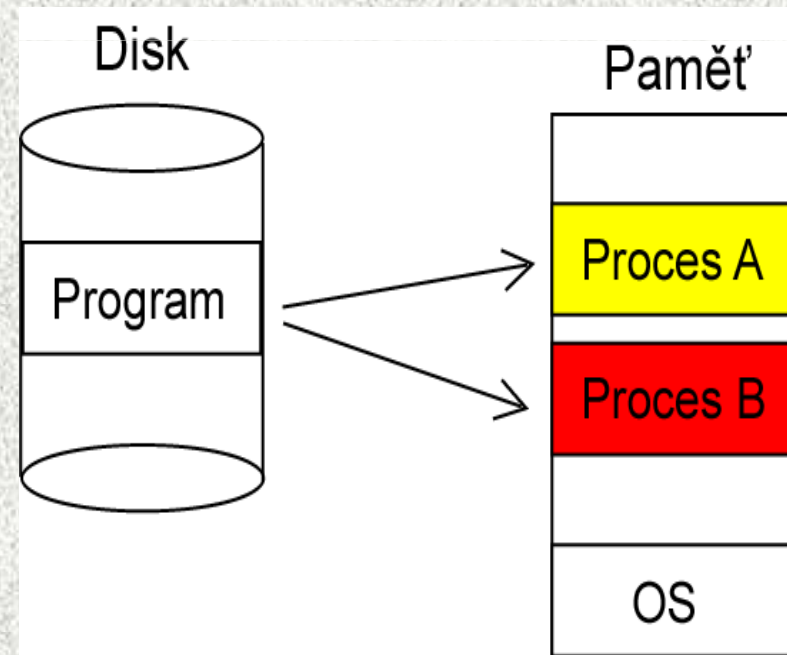


3. Proč procesy? V/V „paralelismus“

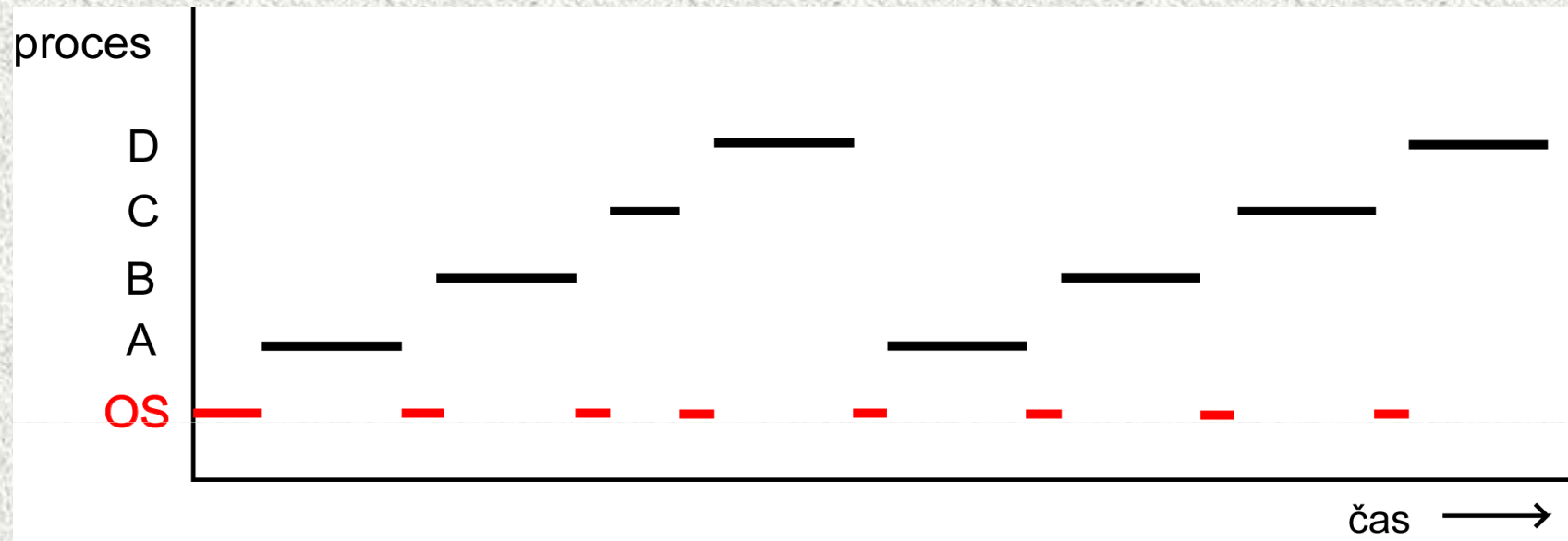
- Zatímco jeden proces čeká na dokončení V/V operace jiný proces může používat CPU.
- Překrývání zpracování: dělá z 1 CPU “více CPU”.
- Reálný paralelismus – více procesorů!

Program versus proces

- **Co je program?**
 - posloupnost instrukcí a data uložená v souboru na disku, pasivní.
- **Co je proces?**
 - abstrakce **spuštěného programu**, zahrnující aktuální hodnoty registrů a proměnných.
- Příklad: spustíme dvakrát editor.
 - **stejný program,**
 - **ale dva různé procesy.**



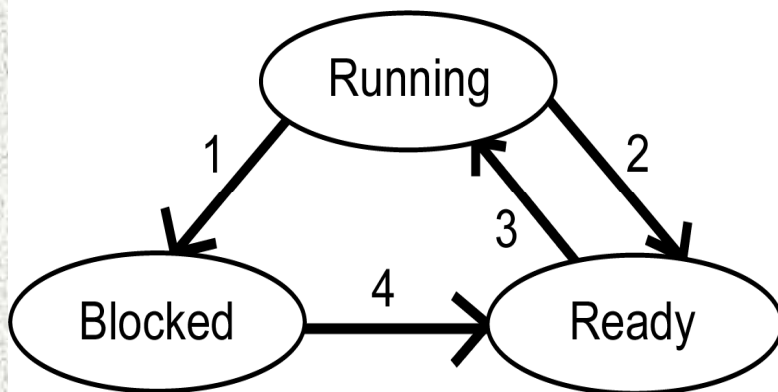
Přepínání kontextu



- CPU v krátkých časových intervalech (řádově milisekundy) přechází od vykonávání jednoho procesu k vykonávání instrukcí druhého procesu.
- Kdo **určuje**, který další proces poběží?

OS

Stavy procesu

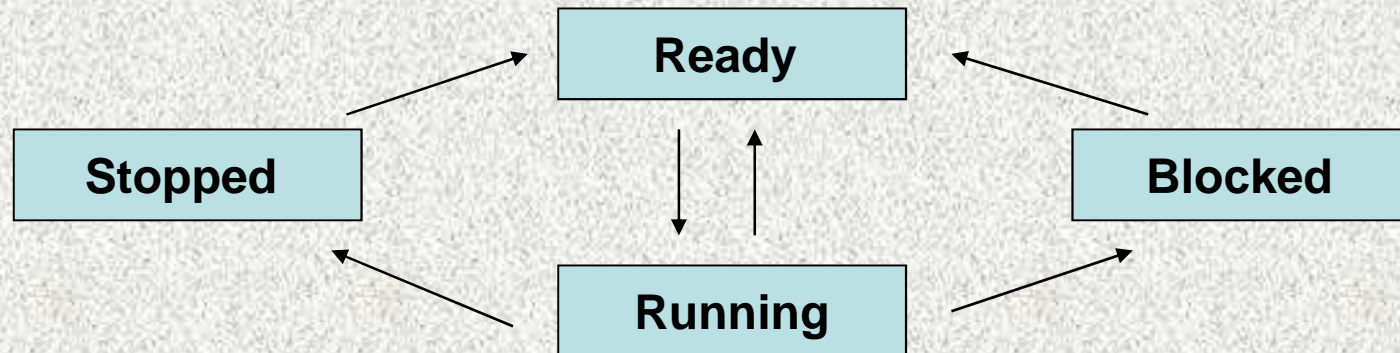


1. Čekání na událost nebo prostředek.
2. Vypršelo časové kvantum.
3. Nastal naplánovaný čas.
4. Nastala událost nebo je prostředek k dispozici.

- **Stavy procesu**

- **Running** = proces právě používá CPU.
- **Blocked** = proces čeká na externí událost nebo na prostředek.
- **Ready** = (pozastavené) proces je připraven a čeká na přidělení CPU.
- **Stoped** = proces ukončen, není možné pokračovat

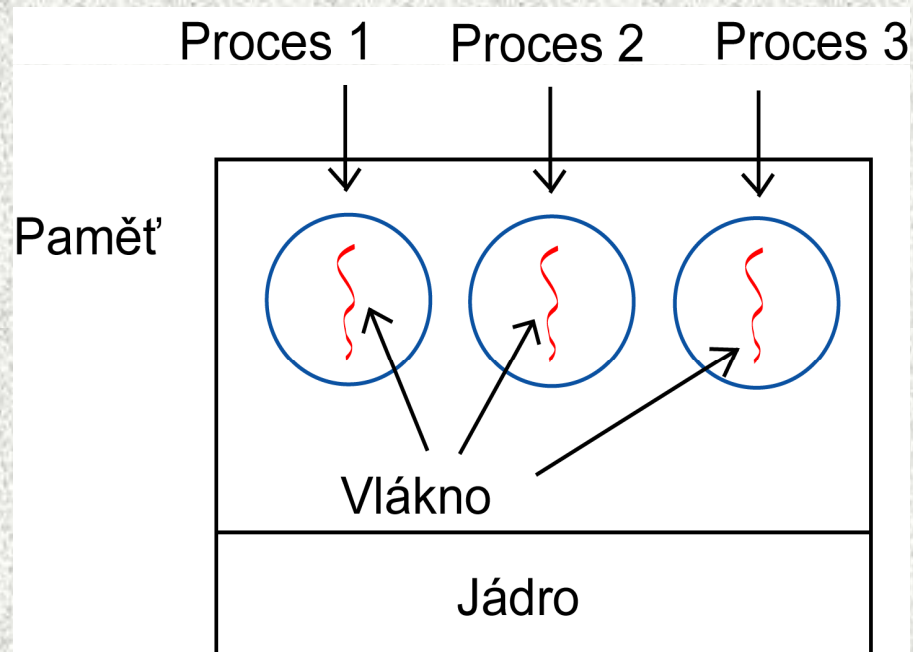
Stavy procesu



- **Stavy procesu**
 - **Running** = proces právě používá CPU.
 - **Blocked** = proces čeká na externí událost nebo na prostředek.
 - **Ready** = (pozastavené) proces je připraven a čeká na přidělení CPU.
 - **Stopped** = proces ukončen, není možné pokračovat

Proces versus vlákno

- **Proces model**
 - Každý proces **alokuje příslušné prostředky** (adresové prostor obsahující kód, data a zásobník procesu, otevřené soubory, potomky, reakce na signály, ...)
 - **Jeden proces = jedno vlákno výpočtu.**



Priority, synchronizace

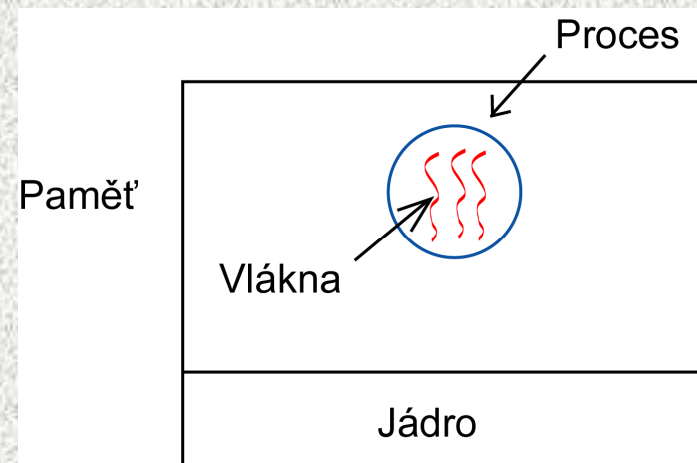
- Princip priority
 - Vlákno předává řízení dobrovolně, pak vláknu s nejvyšší prioritou
 - Vlákno s vyšší prioritou může přerušit vlákno s prioritou nižší – *preemptivní paralelní zpracování*
- Princip synchronizace
 - Vlákno čeká na dokončení jiného vlákna
 - Problém sdílení datových prostorů
 - Řeší se synchronizační metodou
 - Vlákno v synchronizační metodě nad daným objektem zabrání všem ostatním vláknům volat další synchronizační metodu nad tímž objektem

Proces versus vlákno (2)

- **Vláknový model – jeden proces – více vláken**
 - Odděluje alokaci prostředků a samotný výpočet.
 - **Proces** slouží k **alokaci společných prostředků**.
 - **Vlákna** jsou **jednotky plánované pro spuštění na CPU**.
- *Příklad: práce na pozadí, vstup/výstup, simulace, producent/konzument*
- **Vlákno** má (na rozdíl o procesu)
 - svůj vlastní **program counter** (pro uchování informace o výpočtu),
 - **registry** (pro uchování aktuálních hodnot),
 - **zásobník** (který obsahuje historii výpočtu),
 - **lokální proměnné**,
 - **ale ostatní prostředky jsou sdílené**.

Vláknový model

- Jednotlivá vlákna v daném procesu **nejsou nezávislá** tak jako jednotlivé procesy.
- Všechny vlákna v procesu **sdílí** stejný adresový prostor, stejné otevřené soubory, potomky, reakce na signály, ...
- **Multithreading**
 - **Procesy se spouští implicitně pouze s jedním vláknem.**
 - **Toto vlákno může vytvářet další vlákna** pomocí knihovnické funkce
 - **Když chce vlákno skončit, může se opět ukončit** pomocí knihovnické funkce

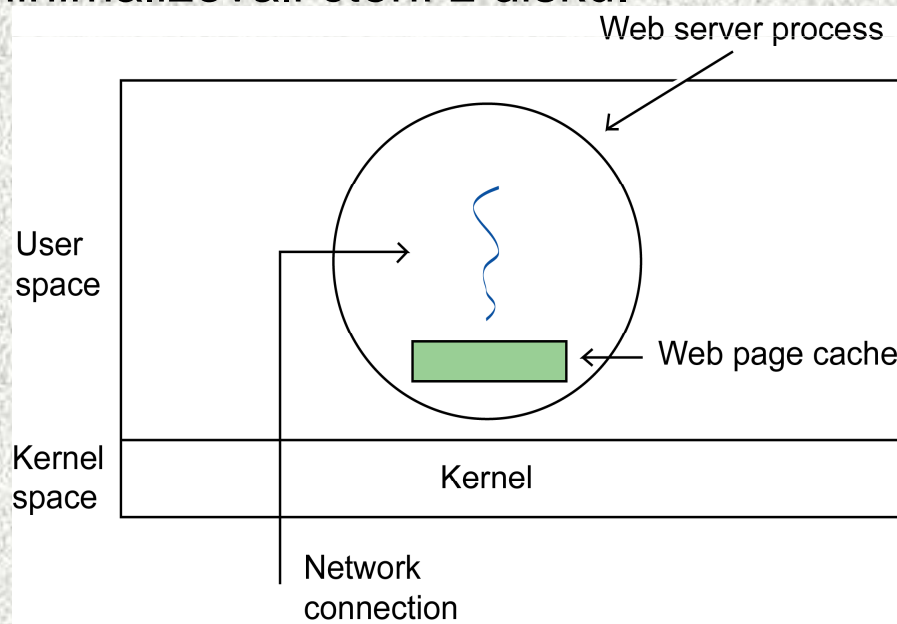


Aplikace vláken

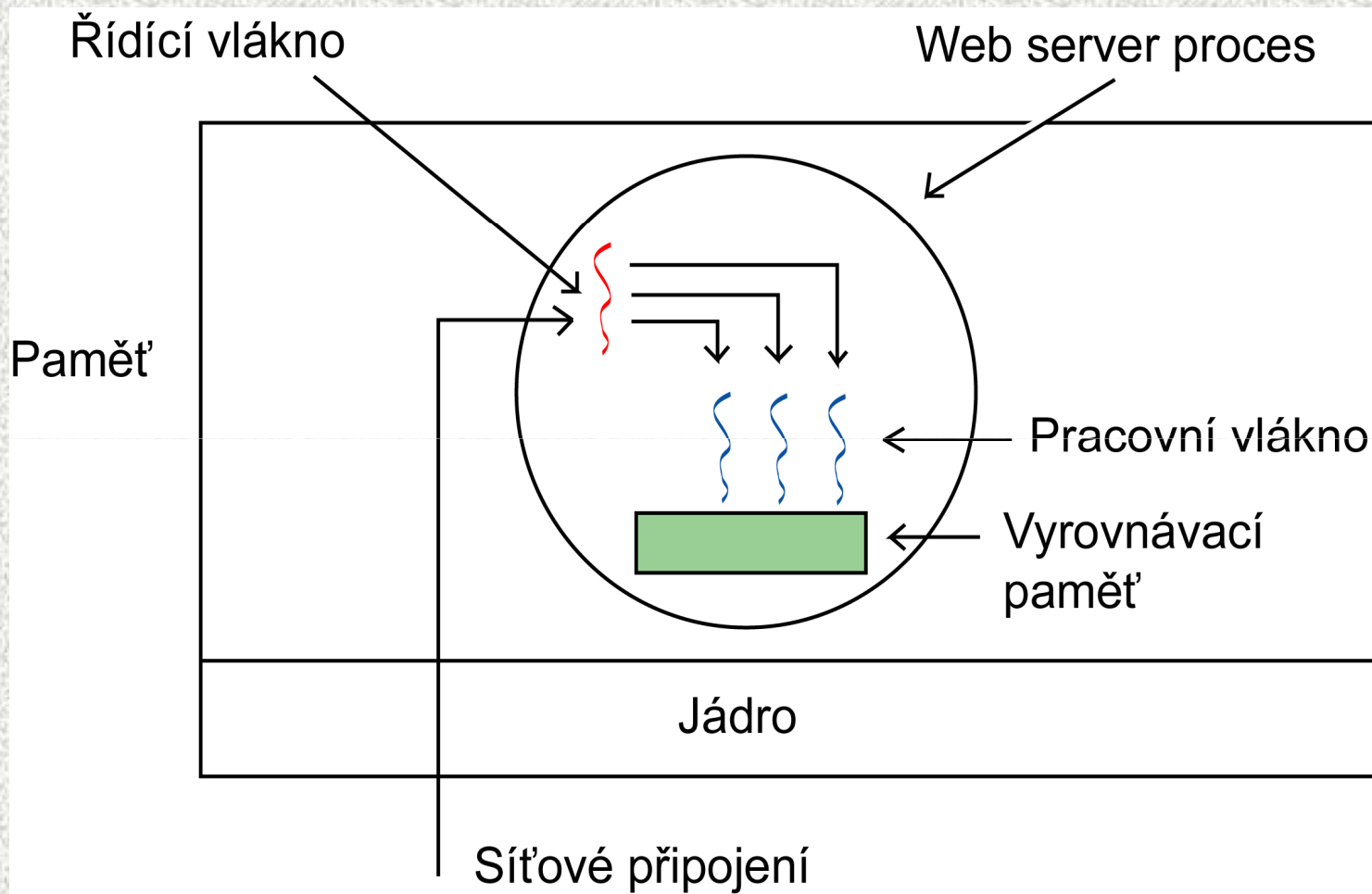
- Časově náročné akce
 - Delší než 1 vteřina
 - Obrázky
- Čekání na vstup od uživatele
 - na jednotlivé klávesy, zamyšlení, náročnější vstup
 - Kontrola pravopisu
- Opakující se výpočty
 - Simulace, počítačové hry
 - „paralelní spuštění činnosti“
- Úlohy typu producent-konzument
 - „paralelní příprava dat“

Příklad: jednovláknový Web Server

- Klient
 - pošle požadavek na konkrétní web. stránku
- Server
 - ověří zda klient může přistupovat k dané www stránce
 - načte stránku a pošle obsah stránky klientovi
- Pozn.: Často používané stránky zůstávají uloženy v hlavní paměti, abychom minimalizovali čtení z disku.



Příklad: vícevláknový Web Server



Příklad: vícevláknový Web Server (2)

- **Řídící vlákno**
 - čte příchozí požadavky,
 - zkoumá požadavek,
 - vybere nevyužitě pracovní vlákno a předá mu tento požadavek.
- **Pracovní vlákno**
 - načte požadovanou stránku z hlavní paměti nebo disku a pošle ji klientovi.

Výhody vláken

- Snadné sdílení informací
 - data uvnitř daného procesu jsou přístupná pro všechny vlákna daného procesu.
- Rychlé přepínání kontextu mezi vlákny daného procesu.

Zobrazení informací o vláknech

- MS Windows XP
 - CTRL ALT DEL → Správce úloh

Název procesu	PID	Uživatelské jméno	CPU	Čas CPU	Využití paměti	Rozdíl pa...	Velikost VP	Zákl. pr...	Objekt...
PUTTY.EXE	3564	honza	00	0:00:00	1 284 kB	0 kB	1 432 kB	Normální	19
Illustrator.exe	2972	honza	00	0:00:02	27 880 kB	0 kB	20 136 kB	Normální	749
POWERPNT.EXE	2864	honza	00	0:00:47	30 704 kB	0 kB	57 436 kB	Normální	190
SOFFICE.BIN	2796	honza	00	0:00:00	16 108 kB	0 kB	5 784 kB	Normální	21
SOFFICE.EXE	2788	honza	00	0:00:00	1 624 kB	0 kB	568 kB	Normální	1
AcroTray.exe	2504	honza	00	0:00:00	2 132 kB	0 kB	744 kB	Normální	9
BTTray.exe	2404	honza	00	0:00:00	4 920 kB	0 kB	3 600 kB	Normální	19
MSMSG5.EXE	2384	honza	00	0:00:00	3 832 kB	0 kB	3 088 kB	Normální	13
CTFMON.EXE	2376	honza	00	0:00:00	3 200 kB	0 kB	968 kB	Normální	11
ZLCLIENT.EXE	2364	honza	00	0:00:00	3 688 kB	0 kB	4 396 kB	Normální	194
WINAMPA.EXE	2348	honza	00	0:00:00	2 112 kB	0 kB	728 kB	Normální	4
AVGCC.EXE	2336	honza	00	0:00:01	10 204 kB	0 kB	3 512 kB	Normální	96
JUSCHED.EXE	2324	honza	00	0:00:00	1 856 kB	0 kB	576 kB	Normální	2
WLTRAY.EXE	2308	honza	00	0:00:00	6 012 kB	0 kB	2 892 kB	Normální	12
SOUNDMAN.EXE	2216	honza	00	0:00:00	2 688 kB	0 kB	1 972 kB	Normální	8
ATIPTAXX.EXE	2208	honza	00	0:00:00	4 180 kB	0 kB	2 972 kB	Normální	46
EPM-DM.EXE	2192	honza	00	0:00:00	3 128 kB	0 kB	2 264 kB	Normální	21
LManager.exe	2180	honza	00	0:00:00	5 716 kB	0 kB	3 816 kB	Normální	75
PDVDServ.exe	2108	honza	00	0:00:00	3 020 kB	0 kB	980 kB	Normální	7
Ktp.exe	2096	honza	00	0:00:01	3 340 kB	0 kB	1 472 kB	Normální	31
RIINDI132.FXF	2076	honza	00	0:00:00	3 332 kB	0 kB	2 348 kB	Normální	7

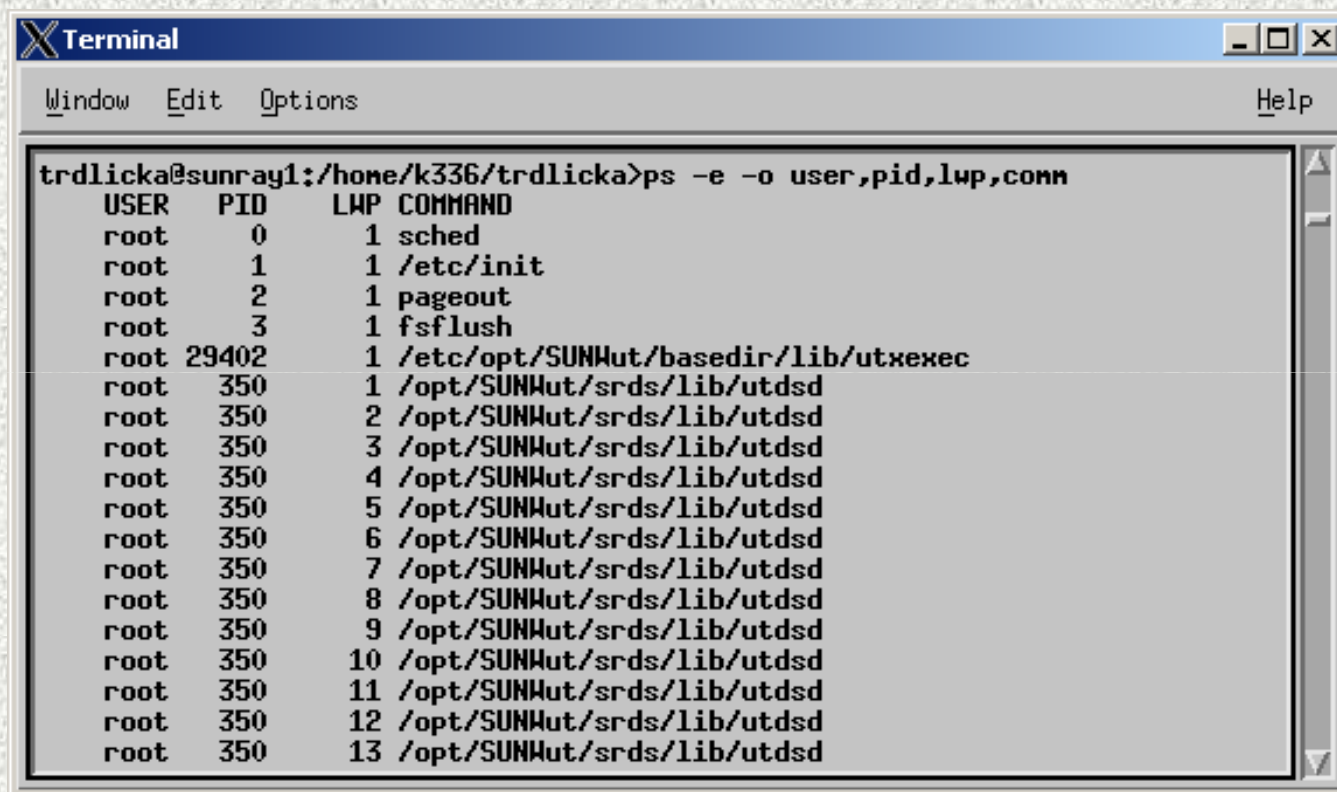
Zobrazit procesy všech uživatelů

Ukončit proces

Procesy: 52 Využití CPU: 1% Využití paměti: 384M / 1246M

Zobrazení informací o vláknech (2)

- Solaris
 - např.pomocí příkazu ps



```
trdlicka@sunray1:/home/k336/trdlicka>ps -e -o user,pid,lwp,comm
USER  PID  LWP COMMAND
root  0    1   sched
root  1    1   /etc/init
root  2    1   pageout
root  3    1   fsflush
root 29402 1   /etc/opt/SUNWut/basedir/lib/utxexec
root  350  1   /opt/SUNWut/srds/lib/utdsd
root  350  2   /opt/SUNWut/srds/lib/utdsd
root  350  3   /opt/SUNWut/srds/lib/utdsd
root  350  4   /opt/SUNWut/srds/lib/utdsd
root  350  5   /opt/SUNWut/srds/lib/utdsd
root  350  6   /opt/SUNWut/srds/lib/utdsd
root  350  7   /opt/SUNWut/srds/lib/utdsd
root  350  8   /opt/SUNWut/srds/lib/utdsd
root  350  9   /opt/SUNWut/srds/lib/utdsd
root  350 10   /opt/SUNWut/srds/lib/utdsd
root  350 11   /opt/SUNWut/srds/lib/utdsd
root  350 12   /opt/SUNWut/srds/lib/utdsd
root  350 13   /opt/SUNWut/srds/lib/utdsd
```

- V uvedeném výpisu je proces utdsd reprezentován několika vlákny.

Problémy při komunikaci mezi procesy/vláknky

- **Časově závislé chyby**

- Pokud dva nebo více procesů čte/zapíše na **sdílené prostředky** (např. sdílená paměť, proměnné, soubory,...), pak může dojít k časově závislé chybě.
- **Výsledek** výpočtu **závisí na náhodném průběhu** prokládání procesů.

- **Uvážnutí**

- Stav, kdy **množina procesů nemůže pokračovat** v činnosti, protože **každý proces z množiny čeká na uvolnění prostředku**, přiděleného jinému procesu z množiny.

- **Stárnutí**

- Stav, kdy **proces může teoreticky pokračovat** v činnosti, ale ve skutečnosti se neustále nachází ve stejném stavu (např. je neustále předbírán jinými procesy).

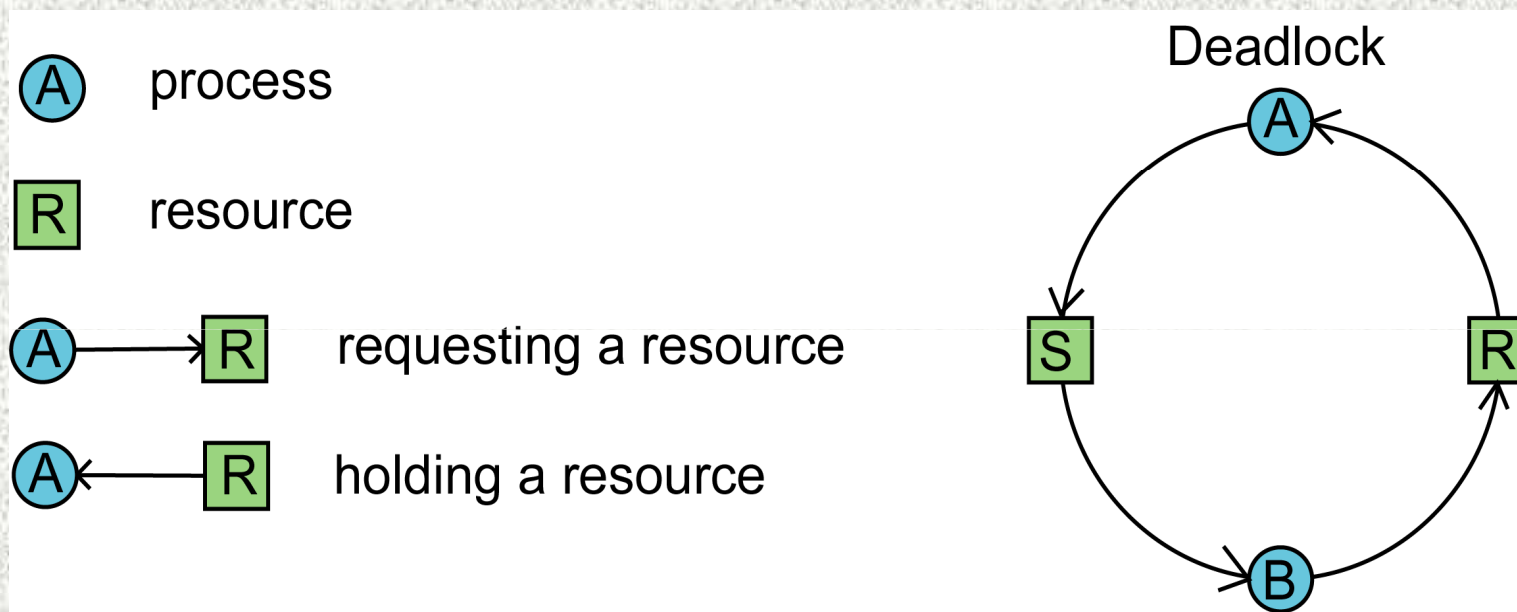
- Tyto chyby se vyskytují náhodně a proto se **velmi špatně odhalují!**

Řešení časově závislých chyb

- Definujeme kritickou sekci
 - část programu, kde procesy používají sdílené prostředky (např. sdílenou paměť, proměnnou,...).
- Časově závislým chybám předcházíme pomocí vzájemného vyloučení.
- Vzájemné vyloučení:
 - mechanismus, který znemožňuje, aby se dva nebo více procesů nacházelo v související kritické sekci současně.
- Vzájemného vyloučení dosahujeme pomocí **synchronizace procesů**.

Modelování uváznutí

- Uváznutí můžeme modelovat pomocí **alokačního grafu**.



- **Každá smyčka v grafu představuje uváznutí** (procesy ve smyčce čekají a nemohou pokračovat).

Příklad: Uváznutí procesů

proces A

Žádost o R
Žádost o S
Použití R a S
Uvolnění R
Uvolnění S

proces B

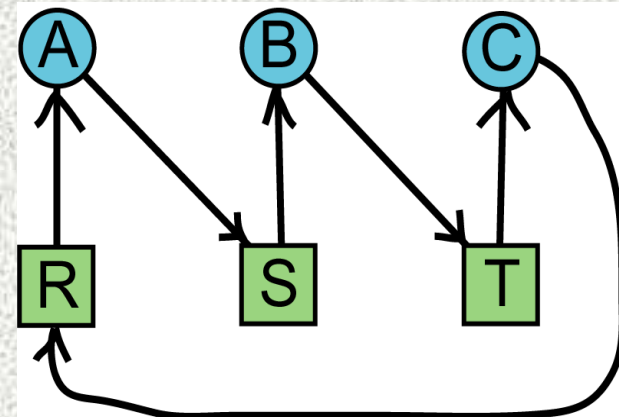
Žádost o S
Žádost o T
Použití S a T
Uvolnění S
Uvolnění T

proces C

Žádost o T
Žádost o R
Použití T a R
Uvolnění T
Uvolnění R

Alokace s uváznutím:

A: Žádost o R
B: Žádost o S
C: Žádost o T
A: Žádost o S \Rightarrow proces je uspán
B: Žádost o T \Rightarrow proces je uspán
C: Žádost o R \Rightarrow **uváznutí !!!**



Příklad: Uváznutí procesů (2)

proces A

Žádost o R
Žádost o S
Použití R a S
Uvolnění R
Uvolnění S

proces B

Žádost o S
Žádost o T
Použití S a T
Uvolnění S
Uvolnění T

proces C

Žádost o T
Žádost o R
Použití T a R
Uvolnění T
Uvolnění R

Alokace bez uváznutí:

A: Žádost o R
C: Žádost o T
A: Žádost o S
C: Žádost o R \Rightarrow proces je uspán
A: uvolnění R ... **bez uváznutí!!!**

