Distribuované systémy

pojmy, koncepty, úlohy

Obsah

1. Výpočetní modely – souhrn
2. Dnešní schémata počítačů
3. Distribuovaný systém, algoritmus
4. Aplikace distribuovaných systémů
5. Distribuovatelnost – data, řízení
6. Typy distribuovaných algoritmů
7. Distribuované úlohy

1. Výpočetní modely - souhrn

- Dávkové zpracování (z důvodu drahého HW, chybí interaktivita při zpracování)
- Host/terminál (vzdálené aplikace, přenos jen dat, umožněna interaktivnost s aplikací přes terminál)
- File server/pracovní stanice (po síti se přenáší aplikace i data – běží na stanici, přenos velkého množství dat)
- Klient / server (klientská a serverová část aplikace – menší přenos dat)

2. Schémata počítačů

Dnešní schémata počítačů 1

- von Neumann: počítač pracuje vždy nad jedním programem (velmi špatnému využití strojového času, obvyklé, že počítač zpracovává paralelně více programů zároveň - tzv. multitasking).
- Počítač může disponovat i více než jedním procesorem.
- Počítač podle von Neumannova schématu pracoval pouze v tzv. diskrétním režimu.
- Existují vstupní / výstupní zařízení (I/O devices), která umožňují jak vstup, tak výstup dat (programu).
- Program se do paměti nemusí zavést celý, ale je možné zavést pouze jeho část a ostatní části zavádět až v případě potřeby.

Dnešní schémata počítačů 2

Distribuované systémy obsahují několik CPU, existují různé způsoby organizace hardware, speciálně v přístupu k jejich vzájemnému propojení a komunikaci. Podle počtu instrukčních a datových proudů se počítače dělí na (**Flynnova taxonomie)**:

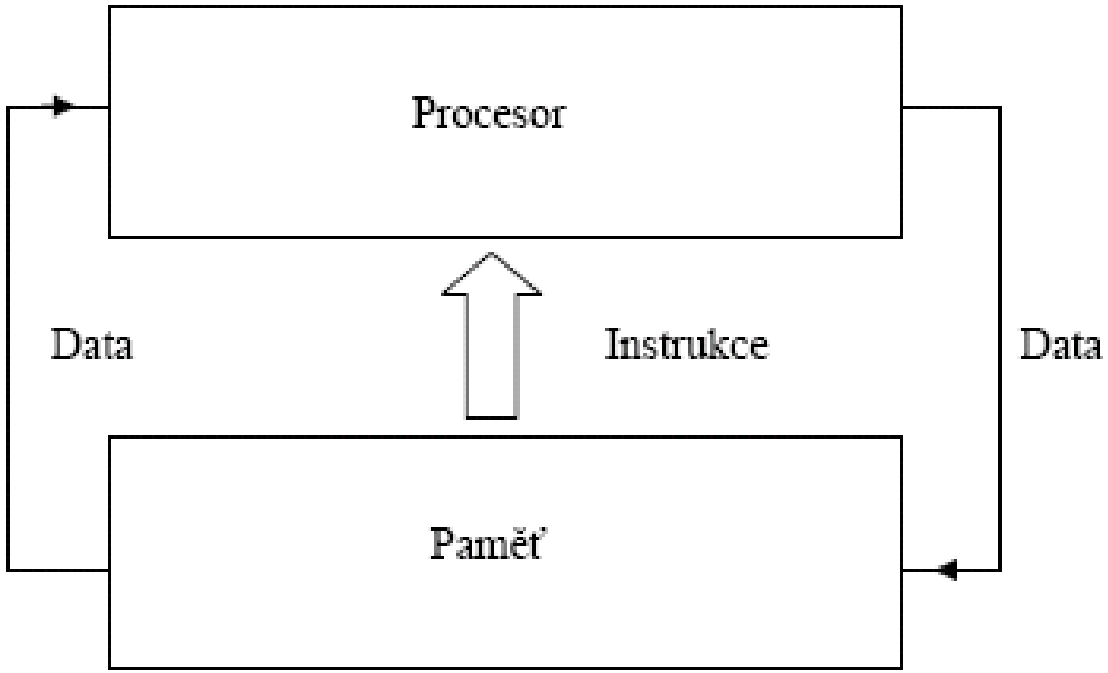
- SISD (single instruction, single data) - běžné jednoprocesorové počítače,
- SIMD (single instruction, multiple data) - jedna instrukce na větším množství dat
- MISD (multiple instruction, single data) – někdy označované jako řetězené (pipeline) procesory,
- MIMD (multiple instruction, multiple data) - víceprocesorové systémy.

SISD počítač

SISD = Single Instruction Single Data

Shodný s von Neumannovým.

Procesor postupně přebírá instrukce, dekóduje je a provádí.

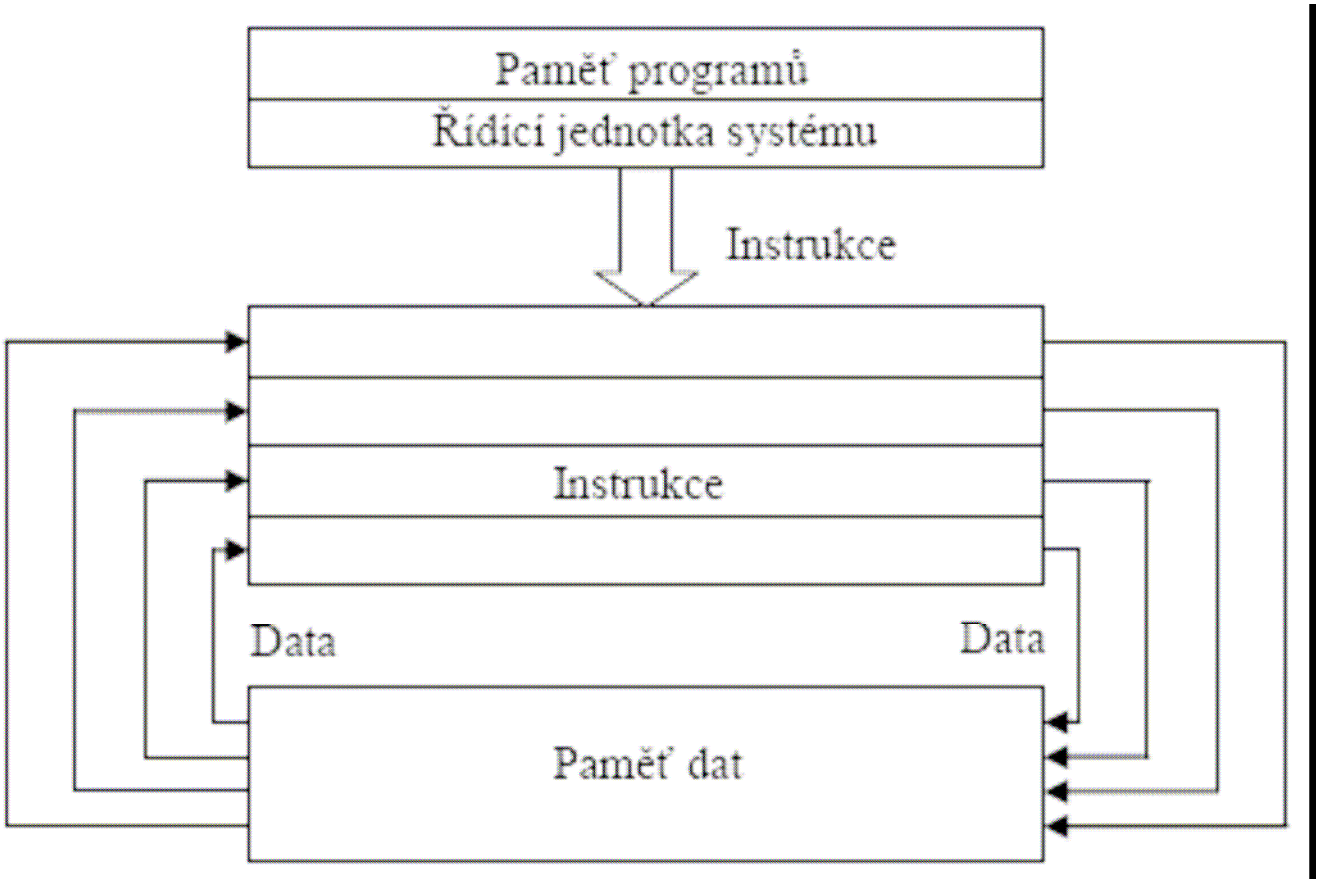


SIMD počítač

SIMD = Single Instruction Multiple Data

Kvůli efektivnímu zpracování je hlavní paměť rozdělena na dva fyzicky samostatné celky: paměť programů, která spolupracuje s řídící jednotkou maticového procesoru a paměť dat, která komunikuje se všemi procesory. Všechny procesory mohou vykonávat současně tutéž operaci s daty, uloženými v paměti. Proto je maticový procesor užitečný pro operace s maticemi a vektory.

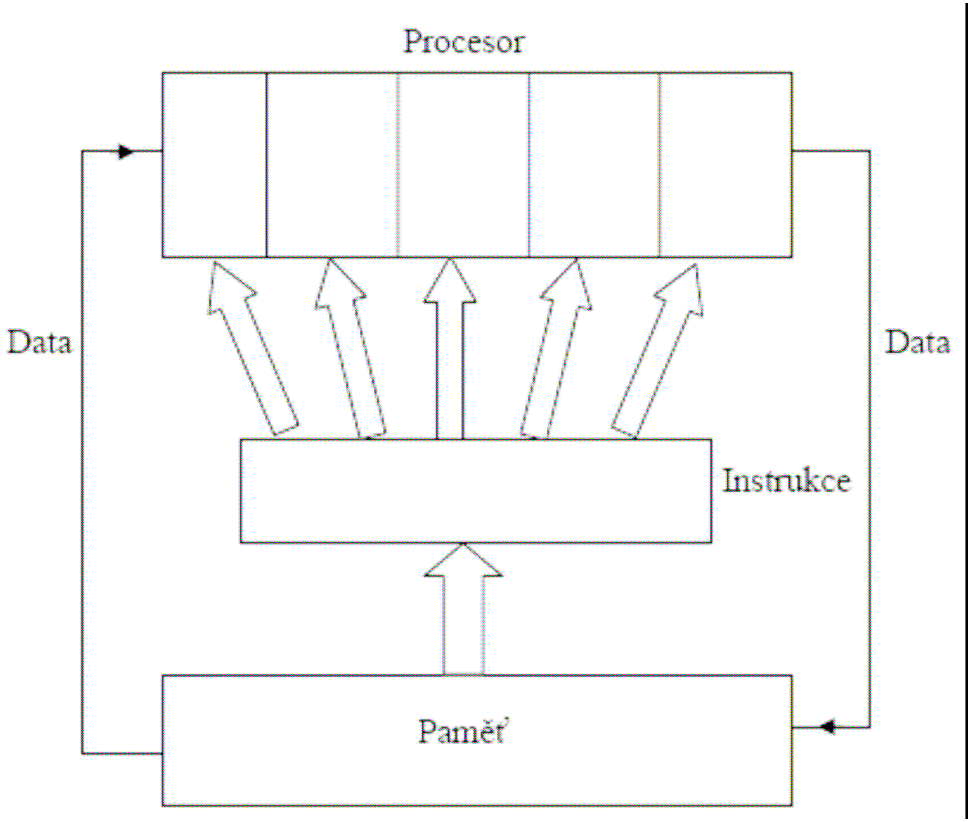
SIMD počítač – schéma



MISD počítač

MISD = Multiple Instruction Single Data

Procesor realizuje základní funkce řetězením operačních modulů, které pracují autonomně. Tím se dosahuje potřebný paralelismus.



MIMD počítač 1

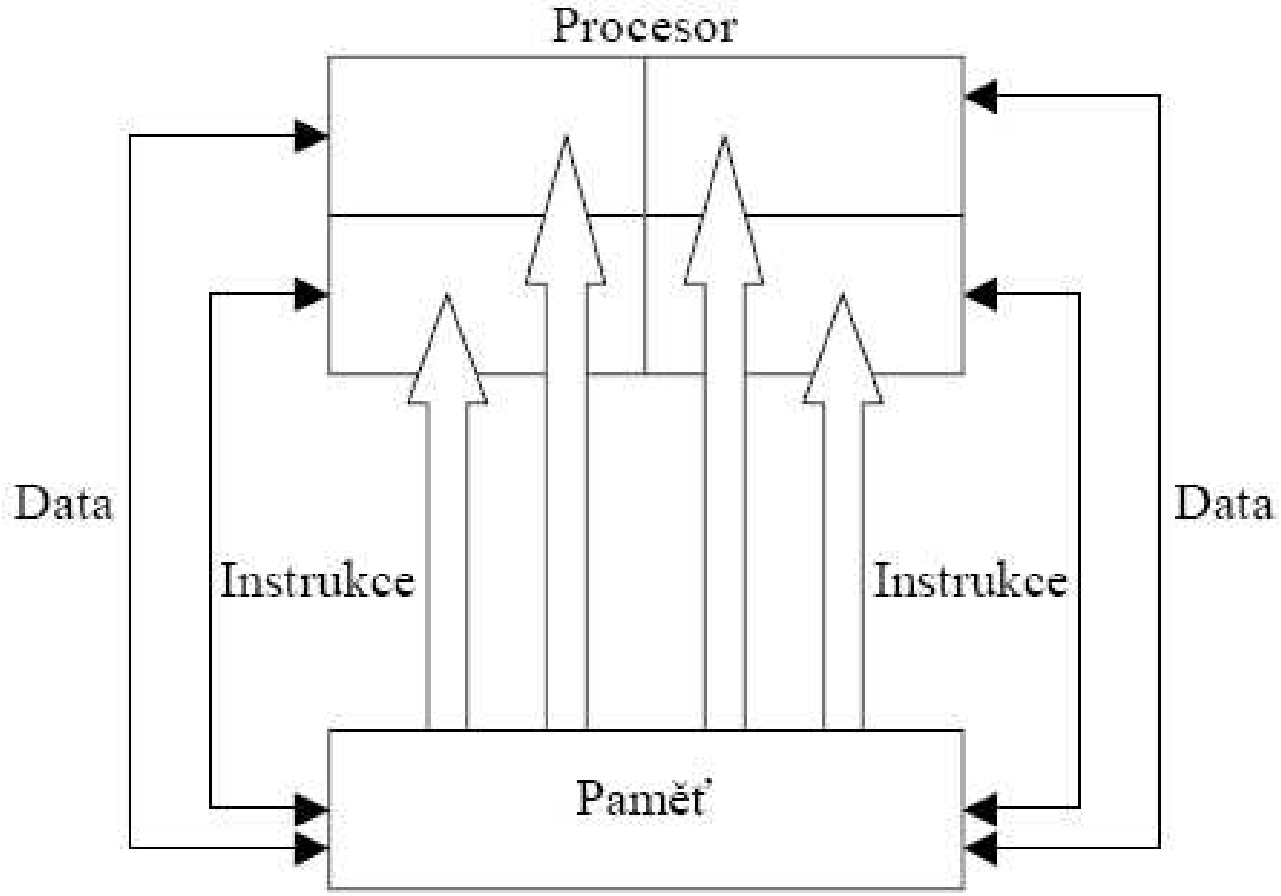
MIMD = Multiple Instruction Multiple Data

- systém obsahuje dva nebo více fyzicky úplně nezávislých procesorů, každý s jedním tokem instrukcí,
- procesory sdílejí společnou operační paměť,
- systém disponuje společným souborem vstupních/výstupních zařízení,
- celý počítačový systém je řízen jedním operačním systémem (jedna řídící autorita).

MIMD počítač 2

- Každý procesor schopen samostatně vybírat a provádět instrukce. Multiprocesor tedy může zpracovávat současně více programů (samostatných částí jednoho programu – souběžné zpracování – multiprocessing).
- Procesory nemusí být stejného typu – nehomogenní multiprocesor.
- Sdílení hlavní paměti se rozumí ve smyslu možnosti fyzického přístupu každého procesoru do libovolného paměťového místa.

MIMD počítač – schéma



3. Distribuovaný systém

Distribuovaný systém – úvod

- V distribuovaném systému probíhá distribuované zpracování. Na distribuovaném výpočtu spolupracuje nějaká podmnožina spojených procesorů.
- Každý procesor provádí svůj vlastní instrukční tok a zpracovává svá vlastní lokální data. Pokud se musí data vyměnit, děje se mechanismem zasílání zpráv.
- Výpočet realizovaný propojenou skupinou procesorů s přímým přístupem ke sdílené paměti nazýváme paralelní zpracování a takový systém se nazývá paralelní systém nebo těsně vázaný systém.

Paralelní / distribuované zpracování

Konkurentní výpočty (srovnání rychlosti a spolehlivosti meziprocesorové komunikace):

- Při velmi vysokých komunikačních rychlostech, řádově srovnatelných s rychlostí přístupu do paměti, se jedná o **paralelní zpracování**.
- Je-li rychlost komunikace nižší, potom se jedná o **distribuované zpracování**.
- Paralelním systémem je z tohoto hlediska multiprocesor a multipočítač.
- Počítačová síť je systém distribuovaný.

Distribuovaný systém – definice

„A distributed system is a collection of independent computers that appears to its users as a single coherent system“

Dva základní aspekty:

- Autonomní hardware (fyzicky oddělené stroje).
- Uživatelé si myslí, že komunikují s jedním systémem.

Distribuovaný algoritmus

Algoritmus považujeme za distribuovaný, jestliže splňuje alespoň jedno z následujících kritérií:

- přijímá, zpracovává a/nebo vysílá data distribuovaná mezi uzly,
- řízení je distribuováno mezi uzly (procesy).

Pokud jsou distribuované algoritmy použity speciálně pro výměnu informací mezi množinou kooperujících uzlů a pro řízení jejich komunikace, jsou často nazývány protokoly (protokol transportu dat, protokol zřízení spojované služby apod.).

4. Aplikace distribuovaných systémů

DS systémy a jejich využití

- Neustálý vývoj výpočetních modelů
- Hojné použití v podnicích a institucích (rostoucí počet aplikací) – pomalá obměna
- Koexistence nových a starších modelů (např. stále jsou využívané terminály)
- Složitější prostředí

Praktické použití DS

- Systémy s okny pro PC nebo pracovní stanice
- Distribuované objektové systémy (CORBA, DCOM, GLOBE)
- Zpracování transakcí ve víceuživatelských databázích (distribuované DB)
- Distribuovaný souborový systém (Coda, Plan 9, XFS)
- Distribuovaný dokument systém (WWW, Lotus Notes)
- Vědecké výpočty

Záměr distribuovaných systémů

Při konstrukci distribuovaného systému by mělo být splněno několik cílů:

1. Měl by umožnit jednoduché spojení uživatelů a zdrojů (většinou výpočetních).
2. Měl by skrýt fakt, že zdroje jsou distribuovány po síti (transparentnost).
3. Měl by být otevřený.
4. Měl by být škálovatelný.

1.Spojení uživatelů a zdrojů

- Snadné (a kontrolované) sdílení zdrojů mezi uživateli (sítě, tiskárny, superpočítače, aplikační servery, úložiště, …).
- **Důvod:** ekonomické hledisko a snadnější správa a údržba.
- Snadnější spolupráce a výměna informací (viz Internet).
- Bezpečnostní problémy (hesla přenášeny jako texty nebo v DB přístupné, sociální inženýrství).

2. Transparentnost

- Snaha skrýt, že procesy a zdroje distribuovaného systému jsou fyzicky rozprostřeny přes několik počítačů.
- Transparentnost může být aplikována na několik oblastí:

Transparentnost	**Popis**
Přístup	Ukrývání odlišné reprezentace dat a přístupu k nim (Intel, SPARC platformy; odlišné file systémy)
Lokace	Ukrytí místa, kde jsou data alokována (např. URL)
Selhání	Ukrytí selhání a zotavení zdroje (problém: selhání vs. pomalá odezva)
Migrace	Ukrytí možného přesunu dat do jiné lokace (URL, mobilní Internet)
Persistence	Ukrytí toho, zda se (softwarové) zdroje nachází v paměti nebo na disku (vlastnost OODBMS)

3. Otevřenost

- Otevřený distribuovaný systém nabízí služby podle standardních pravidel (podle jejich syntaxe, sémantiky).
- Př.: v síťovém prostředí jde o formát, obsah a význam zpráv odeslaných (sent) a přijatých (received) – ve formátu protokolu.
- V DS většinou rozhraní – popsány např. pomocí IDL (Interface Definition Language).
- **Důvod:** aby spolu mohla komunikovat různá zařízení různých výrobců.

4. Škálovatelnost

- 3 dimenze škálovatelnosti:
  - Velikost (lze přidat více users/zdrojů)
  - Geografická (users/zdroje odděleně)
  - Administrativní (snadná údržba)
- Problémy:
  - 1 centrální fyzický server (např. kvůli bezpečnosti)
  - Centralizovaná data (např. adresy v tel seznamu pro 50 mil. Lidí – kapacita OK, ale ne přístup)
  - Vzájemná komunikace může neúměrně zatěžovat síť

HW a SW koncepty

HW koncepty:

- Příkladem byly SIMD, MISD a MIMD počítače (viz dříve)
- Dále např. SAN (Storage Area Network) systémy,
- Massively Parallel Procesors,
- DAS (Distributed Computer System)

SW koncepty:

- DOS (Distributed OS) – OS pro multiprocesory a homogenní multipočítače, cílem je zakrýt a řídit HW zdroje
- NOS (Network OS) – OS pro heterogenní multipočítače (LAN a WAN), poskytuje lokální služby vzdáleným klientům
- Middleware – dodatečná vrstva nad NOS, implementuje obecné služby, poskytuje transparentnost

5. Distribuovatelnost – data, řízení

Důvody pro distribuci

Důvody pro distribuci můžeme rozdělit zhruba do dvou hlavních skupin:

1. Distribuovanost je danému problému vlastní (např. zpracovávání dat v místě jejich vzniku, resp. potřeby, spojené s centrálním zpracováním),
2. Distribuovanost je dána implementací (např. požadavkem na zkrácení doby řešení, na spolehlivost apod.).

Pokud hovoříme o distribuovaném zpracování, distribuovanost se týká buď dat nebo řízení nebo obou těchto aspektů.

Distribuce dat

Data mohou být distribuována dvojím způsobem:

1. **Duplicitou dat –** vytvořením kopie množiny dat v každém uzlu, ve kterém ji algoritmus využívá. Problémem duplicity je trvalé zajišťování konzistence všech kopií.
2. **Rozložením dat -** rozložením množiny dat na části, které jsou uchovávány pouze v určitých uzlech a ostatním uzlům jsou dostupné pouze zprostředkovaně, výměnou zpráv.

   Obě metody lze kombinovat. Celou databázi lze např. rozložit a určité její prvky přitom duplikovat.

Distribuce řízení

- O distribuovaném řízení mluvíme v případě koexistence řady souběžných procesů, mezi nimiž neexistuje žádná pevná hierarchie, neexistuje žádný hlavní algoritmus, který globálně řídí celý systém.
- Mohou existovat určité funkce, které mohou být prováděny pouze v určitých uzlech – potřebujeme distribuovaný algoritmus k identifikaci těchto uzlů
- Určitá rozhodnutí formou dosažení shody (na hodnotě, na společné akci, na vzájemném vyloučení kritických sekcí apod.) mezi skupinou uzlů.

Důvody distribuce řízení

- Algoritmus odolný vůči poruchám – nelze centrální řízení.
- Systém samotný distribuovaný – nelze centrálně řídit.

6. Typy distribuovaných algoritmů

Klasifikace algoritmů

- Podle **míry distribuce řízení** – souvisí se symetrií rolí, které mají zúčastněné procesy (asymetrie je nejnižší stupeň, silná symetrie nejvyšší stupeň rozdělení řízení).
- Podle **způsobu interakce** – určitá schémata komunikace mezi procesy, obvykle lze použít jako optimální nástroj pro řešení celé třídy problémů (používají ověřené programovací techniky).

Klasifikace podle míry distribuce řízení 1

- **Asymetrie -** každý proces v řešení distribuované úlohy je řízen jiným programem, žádné dva nelze zaměnit, asymetrické jsou aplikační systémy vybudované podle modelu C/S, ale také systémy realizované rovnocennými partnerskými procesy (např. není-li symetrický distribuovaný algoritmus znám a některý z procesů musí plnit funkci arbitra, iniciátora, koordinátora apod.).
- **Textová symetrie –** text programu prováděný všemi procesy je stejný, procesy jsou identifikovatelné, jména procesů jsou navzájem různá, procesy se mohou chovat různě, vzhledem k vlastní identitě a různým zprávám, které přijímají. Příkladem takového systému může být síť procesů (filtrů) paralelně třídící množinu dat nebo partnerské procesy spolupracující při volbě řídicího (centrálního) uzlu.

Klasifikace podle míry distribuce řízení 2

- **Slabá symetrie –** texty programů prováděných spolupracujícími procesy jsou identické, procesy jsou vzájemně neidentifikovatelné, komunikace se odehrává obvykle pouze mezi bezprostředně souvisejícími uzly. Připouští se, aby procesy vykazovaly různé chování podle přijatých zpráv. Příkladem mohou být opět sítě filtrů implementované ve speciálně propojených počítačových strukturách.
- **Silná symetrie –** texty programů prováděných spolupracujícími procesy jsou identické, procesy jsou vzájemně neidentifikovatelné a všechny se chovají v čase stejně. Se silnou symetrií lze řešit pouze speciální problémy (systolické algoritmy, třídění).

Klasifikace podle způsobu interakce klient-server

- Typický aplikační program obsahuje řadu procedur, provádění vyvolává hlavní program, procedury plní služby požadované hlavním programem (násobení vektorů, získání povětrnostní mapy z databáze). Pokud se takové procedury soustředí do nezávislého programu řídicího proces v některém uzlu, potom hlavní program žádá takový proces o provedení služby.
- Hlavní program (**klient**) řídí z hlediska vzájemné interakce aktivní proces, který zadává požadavky.
- Program soustřeďující v sobě programy služeb (**server**) řídí z hlediska vzájemné interakce pasivní proces, reaguje na iniciační aktivitu klienta.
- Klient čeká, až je jeho požadavek splněn. Server čeká na požadavek klienta a odpovídajícím způsobem na něj reaguje.
- Server je obvykle nekonečný proces a často poskytuje služby více klientům (např. souborový server – obvykle spravuje více souborů, plní požadavky všech klientů, kteří chtějí s těmito soubory pracovat).

Klasifikace podle způsobu interakce sítě filtrů

Filtr je proces transformující data:

- ze vstupního kanálu načte proud dat, provede nad nimi výpočet a vysílá proud výsledných dat na výstupní kanál.

- Filtr je modelem programů v Unixu, je tak mimo jiné implementováno mnoho příkazů jeho UI. Kaskádním a paralelním propojováním filtrů do sítí lze budovat hierarchicky složitější programové konstrukce.
- Filtry propojené sítí pracují souběžně a řešený problém řeší v kooperaci na dílčích částech. Zpracovávaná data tečou sítí jednosměrně, výstupy filtrů jsou funkcemi jejich vstupů.
- V síti filtrů nefunguje žádná zpětná vazba, která by procesu umožňovala zaslat data procesu, od kterého případně data získal.
- Řízení je distribuováno symetricky, filtry jsou replikovány.

Klasifikace podle způsobu interakce partneři

- **Partner** je jeden z několika procesů, které spolupracují na řešení určitého úkolu.
- Partnerské procesy jsou z hlediska řízení rovnocenné (např. dva partnerské procesy mohou spolupracovat při údržbě kopií datového souboru – s cílem udržet konzistentnost – nebo několik partnerských procesů může spolupracovat při paralelním řešení určité úlohy, každý z nich řeší část úlohy samostatně a o celkovém postupu se vzájemně informují zasíláním zpráv.

7. Distribuované úlohy

Distribuované úlohy

Mezi distribuované úlohy řadíme:

- problém synchronizace,
- dosažení shody,
- mnoho problémů závislých na topologii použité sítě (např. směrování toku dat, optimalizace komunikační zátěže),
- řízení distribuovaných databází.