KOMUNIKACE PROCESŮ

Komunikace procesů, známá také jako meziprocesová komunikace (IPC), je klíčovou funkcí multitaskingových operačních systémů. Tato prezentace vysvětluje různé typy komunikace mezi procesy, včetně přímé a nepřímé komunikace, zprávových front, socketů a rour (pipes). Zaměříme se také na specifické způsoby komunikace ve Windows a Linuxu a na různé mechanismy, jako jsou signály, roury, sockety a systémová volání. Představíme i použití těchto technologií v reálném světě.

PRINCIP KOMUNIKACE PROCESŮ

Definice IPC

- IPC (Interprocess Communication) umožňuje procesům vyměňovat si data nebo signály.

Typy procesů

- **Odesílatel (sender)**: Proces, který posílá data nebo signál.
- **Příjemce (receiver)**: Proces, který přijímá data nebo signál.

Typy komunikace

- **Přímá**: Odesílatel zná příjemce.
- **Nepřímá**: Data přecházejí přes třetí stranu (např. sdílená paměť).

Přenosové modely

- **Unicast**: Pro jednoho příjemce.
- **Broadcast**: Pro všechny v síti.
- **Multicast**: Pro specifickou skupinu.

PŘÍMÁ KOMUNIKACE MEZI PROCESY

Symetrická přímá komunikace

- Odesílatel a příjemce se navzájem znají; lze využít prioritní fronty.

Asymetrická přímá komunikace

- Příjemce nezná identitu odesílatele, ale odesílatel zná příjemce.

Synchronní a asynchronní přenos

- **Synchronní**: Odesílatel čeká na potvrzení.
- **Asynchronní**: Odesílatel pokračuje bez čekání na odpověď.

Funkce pro přímou komunikaci

- **`send(P, zpráva)`:** Poslání zprávy procesu P.
- **`receive(Q, zpráva)`:** Přijetí zprávy od procesu Q.

NEPŘÍMÁ KOMUNIKACE POMOCÍ PORTŮ A SOCKETŮ

Porty a sockety

- Porty slouží jako brány pro ukládání a přenos zpráv mezi procesy.

Přístup k socketům

- Vlastníkem socketu je proces, který jej vytvořil; ostatní procesy mohou pouze číst.

Funkce socketů

- **`send(ID_portu, zpráva)`** : Uložení zprávy do portu.
- **`receive(ID_portu, zpráva)`** : Přijetí zprávy z portu.

Výhody a nevýhody

- Flexibilita v síťové komunikaci, ale nutnost nastavení přístupových práv.

ZPRÁVY A SYSTÉMOVÁ VOLÁNÍ VE WINDOWS

Zprávy oknům (Window Messages)

- Komunikace v uživatelském prostoru pomocí zpráv zasílaných oknům.

Systémová volání

- Procesy komunikují s jádrem přes systémová volání.

Local Procedure Call (LPC)

- Vnitřní mechanismus pro komunikaci jádra s uživatelskými procesy.

Remote Procedure Call (RPC)

- Vzdálené volání procedury na jiném počítači či procesu.

SIGNÁLY V LINUXU

Definice signálů

- Signál je malý číselný kód, který indikuje určitý stav či událost.

Typické signály

- **SIGTERM**: Žádost o ukončení procesu.
- **SIGKILL**: Okamžité ukončení bez možnosti zachytit signál.

Možnosti obsluhy signálu

- **Ignorování**: Proces signál ignoruje.
- **Blokování**: Signál je zpracován později.
- **Vlastní obslužná rutina**: Specifická reakce na signál.

Skupiny procesů a relace

- Procesy mohou komunikovat jako skupiny, což zjednodušuje správu.

ROURY (PIPES) A JEJICH VYUŽITÍ

Pojmenované vs. nepojmenované roury

- **Pojmenované**: Chovají se jako soubory, použitelné pro trvalé spojení.
- **Nepojmenované**: Jednorázové, omezené velikostí, typické pro dočasná data.

Vytvoření a správa roury

- Funkce `pipe()` pro vytvoření nepřímého komunikačního kanálu.

Výhody a nevýhody rour

- Jednoduchá jednosměrná komunikace, ale omezená kapacita.

Ukázky příkazů

- Příkaz `ls | grep "text"` využívá rouru pro filtrování výstupu.

SOCKETY A ZPRÁVOVÉ FRONTY (MESSAGE QUEUES)

Sockety jako rozšířená roura

- Používají se v klient-server komunikaci, lokálně i v síti.

Streamová vs. datagramová komunikace

- **Streamová**: Proud dat, např. TCP.
- **Datagramová**: Bloky dat, např. UDP.

Zprávové fronty (POSIX Message Queues)

- Umožňují vytváření front zpráv s prioritami, užitečné pro komunikaci s časovou kritičností.

Sdílení a správa front

- Fronty jsou podobné souborům, mohou být vlastněny a sdíleny mezi procesy.

SHRNUTÍ

- IPC umožňuje procesům komunikovat a sdílet data různými metodami (přímé, nepřímé, sockety, zprávy).
- Ve Windows a Linuxu existují specifické mechanismy pro efektivní IPC, jako jsou signály, systémová volání a sockety.
- Při výběru vhodného IPC mechanismu záleží na požadavcích aplikace, jako jsou latence, zabezpečení a typy přenosu (streamové nebo blokové).
- Synchronní a asynchronní přenos ovlivňují výkon a reakční dobu procesu při komunikaci.

KONTROLNÍ OTÁZKY

1. Jaké jsou hlavní rozdíly mezi přímou a nepřímou komunikací procesů?
2. Kdy je vhodné použít sockety namísto roury?
3. Jaký význam mají signály v Linuxu a jak mohou být procesem obslouženy?
4. Co je to zprávová fronta a jaké jsou její výhody oproti běžným rourám?
5. Jakým způsobem probíhá komunikace mezi procesy v systému Windows?

DOPORUČENÁ LITERATURA

1. **Silberschatz, A., Galvin, P. B., & Gagne, G.** - *Operating System Concepts* - Základy IPC a správy procesů.
2. **Tanenbaum, A. S., & Bos, H.** - *Modern Operating Systems* - Podrobný popis mechanismů IPC.
3. **Stevens, W. R.** - *UNIX Network Programming* - Specifika komunikace procesů v UNIX/Linux.
4. **Love, R.** - *Linux Kernel Development* - Implementace IPC v Linuxovém jádře.
5. **Microsoft Documentation on Interprocess Communication** - Dokumentace k IPC metodám ve Windows.