prevention, but also their resolution. Hotel staff should be trained in problem solving and be able to act in stressful situations.

***Keywords**: conflicts, conflict resolution, service quality.*

Науковий керівник:
Кушнірук В. С.
канд. екон. наук, доцент,
доцент кафедри готельно-ресторанної справи
та організації бізнесу,
Миколаївський національний аграрний університет

УДК 004.383.3+519.65

ЗАСОБИ ВЗАЄМОДІЇ МІЖ ПРОЦЕСАМИ ОБЧИСЛЮВАЛЬНИХ СИСТЕМ

Звіришин Валентин Михайлович,
здобувач вищої освіти спеціальності 122 «Компютерні науки»
Миколаївський національний аграрний університет
м. Миколаїв, Україна

Анотація: *у тезах розглянуті засоби взаємодії між окремими процесами, що забезпечує концепцію багатозадачних операційних систем*.

Ключові слова: *сигнали,канали, повідомлення, семафори, сокети*

Процеси виконуються у власному адресному просторі і ізольовані один від одного. Це мінімізує можливість впливу процесів один на одного, що є необхідною умовою в багатозадачних операційних системах. Один ізольований процес малокорисний. Фактичною концепцією багатозадачних операційних систем є модульність, тобто заснована на взаємодії між окремими процесами[1].

До засобів взаємодії між процесами можна віднести:

- сигнали
- канали
- FIFO (іменовані канали)
- повідомлення (черги повідомлень)
- семафори
- пам'ять, що розділяється
- RMDA
- сокети (гнізда)

Сигнали спочатку були запропоновані як засіб повідомлення про помилки, але можуть використовуватися і для елементарного IPC, наприклад, для синхронізації процесів або для передачі найпростіших команд від одного

процесу до іншого. Проте використання сигналів в якості засобу IPC обмежена через те, що сигнали дуже ресурсомісткі. Відправлення сигналу вимагає виконання системного виклику, а його доставка - переривання процесу-одержувача і інтенсивних операцій зі стеком процесу для виклику функції обробки і продовження його нормального виконання. При цьому сигнали слабо інформативні та їх число дуже обмежене. У певному сенсі сигнали забезпечують найпростішу форму міжпроцесорної взаємодії, дозволяючи повідомляти процес або групу процесів про настання деякої події[2].

Синтаксис організації програмних каналів при роботі в командному рядку shell: cat myfile | wc.

При цьому (стандартний) висновок програми *cat (1)*, яка виводить вміст файлу myfile, передається на (стандартне) введення програми *wc (1)*, яка, у свою чергу підраховує кількість рядків, слів та символів. В результаті ми отримаємо щось на кшталт: 12 45 260, що буде означати кількість рядків, слів та символів у файлі myflle[3].

Назва каналів FIFO походить від виразу First In First Out (першим зайшов - першим вийшов). FIFO дуже схожі на канали, оскільки є односпрямованим засобом передачі даних, причому читання даних відбувається в порядку їх запису. Однак на відміну від програмних каналів, FIFO мають імена, які дозволяють незалежним процесам отримати до цих об'єктів доступ. Тому іноді FIFO також називають іменованими каналами. FIFO є окремим типом файлу у файловій системі UNIX.

Повідомлення, точніше черги повідомлень є складовою частиною операційної системи, вони обслуговуються операційною системою, розміщуються в адресному просторі ядра і є розділеним системним ресурсом. Кожна черга повідомлень має свій унікальний ідентифікатор. Процеси можуть записувати і зчитувати повідомлення з різних черг. Процес, що відправив повідомлення в чергу, може не очікувати читання цього повідомлення будь-яким іншим процесом. Він може закінчити своє виконання, залишивши в черзі повідомлення, яке буде прочитано іншим процесом пізніше.

Застосування семафорів можна пояснити на простому прикладі. Нехай, є якийсь роздільний ресурс (наприклад, файл). Необхідно блокувати доступ до ресурсу для інших процесів, коли якийсь процес виробляє операцію над ресурсом (наприклад, записує в файл). Для цього зв’яжемо з даним ресурсом якусь цілочисельну величину - лічильник, доступний для всіх процесів. Вважатимемо, що значення 1 лічильника означає доступність ресурсу, 0 - його недоступність. Тоді перед початком роботи з ресурсом процес повинен перевірити значення лічильника. Якщо воно дорівнює 0 - ресурс зайнятий, і операція недопустима - процесу залишається чекати. Якщо значення лічильника дорівнює 1 - можна працювати з ресурсом. Для цього, перш за все, необхідно заблокувати ресурс, тобто змінити значення лічильника на 0. Після виконання операції для звільнення ресурсу значення лічильника необхідно змінити на 1. У наведеному прикладі лічильник грає роль семафора.

Механізм розподіленої пам'яті дозволяє позбутися від накладних витрат передачі даних через ядро, надаючи двом або більше процесів можливість безпосереднього отримання доступу до однієї області пам'яті для обміну даними. Можна сказати, що процеси повинні попередньо "домовитися" про правила використання розподіленої пам'яті. Наприклад, поки один з процесів проводить запис даних в пам'ять, що розділяється, інші процеси повинні утриматися від роботи з нею. Завдання кооперативного використання розподіленої пам'яті вирішується за допомогою семафорів, які полягають в синхронізації виконання процесів.

RMDA (remote direct memory access) віддалений прямий доступ до пам'яті. При передачі даних за допомогою RDMA виключаються зайві копіювання між додатком і буферами операційної системи. Відповідно знижується обсяг роботи центрального процесора, навантаження на кеш-пам'ять, зменшується кількість перемикань контексту, а самі передачі можуть відбутися одночасно з іншою роботою. Коли додаток виконує запит на читання або запис в віддалену оперативну пам'ять, дані можуть доставлятися безпосередньо в мережевий адаптер, зменшуючи загальний час передачі даних. У прикладному сенсі RDMA - це механізм передачі даних між локальною мережею і оперативною пам'яттю обчислювального вузла. Уніфікується вся адресація - і привілейованої пам'яті операційної системи, і віртуальної пам'яті, що належить додаткам. Отже, два додатки, запущені на різних платформах, можуть взаємодіяти за допомогою майже загальної пам'яті.

Для позначення комунікаційного вузла, що забезпечує прийом і передачу даних для об'єкта (процесу), був запропонований спеціальний об'єкт - сокет (socket). Сокети створюються в рамках певного комунікаційного домена, подібно до того, як файли створюються в рамках файлової системи. Сокети мають відповідний інтерфейс доступу в файловій системі, і так само як звичайні файли, адресуються деяким цілим числом - дескриптором. Однак на відміну від звичайних файлів, сокети являють собою віртуальний об'єкт, який існує, поки на нього посилається хоча б один з процесів.

Для створення сокета процес повинен вказати тип сокета і комунікаційний домен, в рамках якого буде використовуватися сокет. Оскільки комунікаційний домен може підтримувати використання декількох протоколів, процес може також вказати конкретний комунікаційний протокол для взаємодії. Якщо такого не вказано, система вибере найбільш підходящий зі списку протоколів, доступних для даного комунікаційного домена. Якщо ж в рамках зазначеного домену створення сокета даного типу неможливо, тобто відсутній відповідний комунікаційний протокол, запит процесу завершиться невдало.

Колективна пам'ять є частиною адресного простору для кожного із взаємодіючих процесів, тому читання і запис в цій області не відрізняються, наприклад, від читання і запису в область власних даних процесу. У випадку використання розподіленої пам'яті необхідно забезпечити синхронізацію процесів. При використанні семафорів, необхідно брати до уваги наступні обставини:

- застосування семафорів може збільшити число процесів в черзі на виконання, оскільки кілька процесів, які очікують дозволяючого сигналу семафора, будуть одночасно розбуджені і переведені в чергу на виконання.
- застосування семафорів збільшує число перемикань контексту, що, в свою чергу, збільшує навантаження на систему.
- використання семафорів є найбільш стандартним (POSIX.lb), хоча і неефективним способом забезпечення синхронізації.

Порівняння різних систем взаємодії між процесами показує, що найбільш швидким способом передачі даних між несумісними процесами є колективна пам'ять.

Список використаних джерел

1. Шеховцов В.А. Операційні системи. Київ: Видавнича група ВНУ, 2005. 576 с.
2. Ваврук Є. Я. Цифрове опрацювання сигналів та зображень, алгоритми та реалізація: навчальний посібник / Є. Я. Ваврук, Р. Б. Попович – Національний університет “Львівська політехніка”, 2008. – 147 с
3. Операційні системи: навч. посіб./ уклад. М. Ф. Бондаренко, О. Г.Качко. Харків : Компанія СМІТ, 2012. 417 с.

***Abstract**: Theses consider means of interaction between separate processes, which provides the concept of multitasking operating systems.*

***Keywords:** signals, channels, messages, semaphores, sockets*

Науковий керівник:
Крайній В.О.,
канд. екон. наук, доцент,
доцент кафедри економічної кібернетики
і математичного моделювання,
Миколаївський національний аграрний університет

УДК 351.78

ПРАКТИЧНІ АСПЕКТИ АДАПТАЦІЇ ЗАКОНОДАВСТВА ЄС У СФЕРІ БЕЗПЕКИ І ГІГІЄНИ ПРАЦІ В УКРАЇНІ

Іваненко Валерія Сергіївна,
здобувач вищої освіти спеціальності 015 «Професійна освіта
(Аграрне виробництво, переробка сільськогосподарської продукції
та харчові технології)»
Миколаївський національний аграрний університет,
м. Миколаїв, Україна

Анотація: *проаналізовано законодавства з забезпечення безпеки праці та гігієни України в контексті євроінтеграції. Висвітлено головні проблеми та*