BIR NECHA JARAYONLARNI YARATADIGAN VA IPC (KANALLAR/SIGNALLAR) DAN FOYDALANADIGAN C/PYTHON DASTURI

Ilmiy rahbar: ***Xolmatov Abdurashid***

Talaba: ***Muhammadjonova Maftuna***

Farg'ona Davlat Texnika Universiteti 3-kurs

Ma'lumotlarni intellektual tahlili

Annotatsiya:

Ushbu maqolada operatsion tizimlarda bir nechta jarayonlarni yaratish va ular o'rtasida ma'lumot almashish (IPC — Interprocess Communication) tamoyillari yoritilgan.

Dasturlashning ikki mashhur tili — Python va C misolida kanal (Pipe) va signal (Signal) orqali jarayonlararo aloqa tashkil etish usullari ko'rsatib berildi. Har bir til uchun sodda va tushunarli kod namunalar berilib, jarayon yaratish, xabar uzatishva signallarni qayta ishlash mexanizmlari izohlandi. Maqola IPC texnologiyalarining amaliy qo'llanilishini o'rganuvchi talaba va dasturchilar uchun mo'ljallangan.

Аннотация:

В данной статье рассмотрены принципы создания нескольких процессов и организации обмена данными между ними с использованием механизмов IPC (Interprocess Communication) в операционных системах. На примерах двух популярных языков программирования — Python и C — представлены способы взаимодействия процессов через каналы (Pipe) и сигналы (Signal). Приведены простые и понятные примеры кода, демонстрирующие создание процессов, передачу сообщений и обработку сигналов. Статья предназначена для студентов и разработчиков, изучающих практическое применение технологий IPC.

Abstract:

This article discusses the principles of creating multiple processes and enabling communication between them using IPC (Interprocess Communication) mechanisms in modern operating systems. Using two popular programming languages — Python and C— it demonstrates how processes can interact through Pipes and Signals. Clear and simple mcode examples are provided to illustrate process creation, message exchange, and signal handling. This article is intended for students and developers seeking to understand the practical application of IPC techniques.

Kalit so‘zlar: Jarayon, IPC, pipe, signal, jarayonlararo aloqa, Python, C tili, fork(), multiprocessing, xabar almashinuvi, parallel hisoblash.

Zamonaviy dasturlashda samaradorlikni oshirish, vazifalarni tezroq bajarish va resurslardan optimal foydalanish bir nechta jarayonlarni bir vaqtning o‘zida boshqarishni talab qiladi. Bunday yondashuv **ko‘p jarayonli dasturlash** deb ataladi. Jarayonlar mustaqil ishlashi mumkin bo‘lsa-da, ko‘p hollarda ular bir-biri bilan ma’lumot almashishi yoki boshqaruv signallarini uzatishi zarur bo‘ladi. Jarayonlar o‘rtasidagi bunday o‘zaro ta’sir

Interprocess Communication (IPC) texnologiyalari orqali amalga oshiriladi. **Jarayon, IPC, pipe, signal, xabar almashinuvi, parallel hisoblash, multiprocessing** kabi kalit tushunchalar ushbu mavzuning asosini tashkil qiladi. Python va C dasturlash tillari

IPC mexanizmlarini qo‘llashda eng ko‘p uchraydigan platformalardandir.

Jarayon — bu operatsion tizim tomonidan mustaqil bajariladigan dasturiy obyekt bo‘lib, unda o‘zining xotirasi, registr holati va resurslari mavjud bo‘ladi. Ko‘p jarayonli dasturlashda ota jarayon fork() yoki Python’dagi multiprocessin yordamida yangi farzand jarayonlarni hosil qiladi. Jarayonlarning

afzalliklari:Ilovalar tezroq ishlaydi.Vazifalar bir biriga bog‘liq bo‘lmagan holda bajariladi.Tizim resurslari oqilona taqsimlanadi.Xavfsizlik darajasi oshadi (jarayonlar xotirasi umumiy emas). Jarayonlar bir-biri bilan aloqada bo‘lishi uchun turli IPC usullari ishlatiladi. Eng keng tarqalganlar. Pipe jarayonlar o‘rtasida ketma-ket oqim tarzida ma’lumot almashishga imkon beradi. Odatda:Parent

→Child, Yoki Child → Parent .Bir yo‘nalishda ishlaydi (half-duplex). C tilida pipe(), Python’da multiprocessing.Pipe() qo‘llaniladi.Afzalliklari: Oson, tezkor,kichik hajmdagi xabarlar uchun ideal.

Signal — jarayonni ma’lum bir hodisa haqida xabardor qiladigan qisqa boshqaruv xabari. Masalan: SIGUSR1– foydalanuvchi signali, SIGINT – klaviaturadan to‘xtatish (Ctrl+C),SIGTERM – jarayonni to‘xtatish Signal ma’lumot tashimaydi, lekin boshqaruv uchun juda muhim. Afzalliklari: Juda tez ishlaydi, Boshqaruv vazifalari uchun qulay. Python’da moduli IPC uchun qulay interfeys taqdim qiladi. Pipe orqali xabar almashish Jarayoni yengil va intuitiv bo‘lib, xotira boshqaruvi avtomatik tarzda amalga oshiriladi.

Jarayon yaratish: from multiprocessing import Process, Pipe

- Signal qabul qilish: signal.signal(signal.SIGUSR1, handler)
- Python yuqori darajadagi abstraksiyalar orqali IPC jarayonini soddalashtiradi.

C past darajadagi boshqaruv imkoniyatlari bilan ajralib turadi. fork(), pipe(), read(), write(), signal(), kill() kabi funksiyalar orqali jarayonlar va ularning aloqasi to‘liq nazorat qilinadi. C tilidagi pipe IPC usuli Tizim darajasida juda samarali va tezkor.

Signal bilan ishlash: signal(SIGUSR1, handler); kill(pid, SIGUSR1); C tili IPC’ni chuqur o‘rganish uchun eng qulay til hisoblanadi.Parallel ishlov berish: Katta hajmdagi ma’lumotlarni qayta ishlash, real vaqt tizimlarini boshqarishSun’iy intellekt va neyron tarmoqlar hisob kitoblarini tezlashtirish, server resurslaridan

optimal foydalanish uchun juda muhim. IPC parallel tizimlarning asosiy kommunikatsiya mexanizmi hisoblanadi.

Xulosa

Ko‘p jarayonli dasturlash zamonaviy dasturiy ta’minotning muhim qismidir. Jarayonlararo aloqa (IPC) esa ular o‘rtasida samarali ma’lumot almashinishini ta’minlaydi. Pipe’lar yordamida ma’lumot oqimi boshqariladi, signallar esa jarayonlarning boshqaruv jarayonlarini amalga oshiradi. Python IPC jarayonini soddalashtirsa, C tili uni to‘liq nazorat qilish imkonini beradi. Ushbu texnologiyalar parallel hisoblash, server tizimlari, avtomatlashtirilgan boshqaruv loyihalarida keng qo‘llaniladi.

Foydalanilgan adabiyotlar

1. Tanenbaum, A. S., & Bos, H. **Modern Operating Systems**. Pearson, 2015.

2. Silberschatz, A., Galvin, P. B., & Gagne, G. **Operating System Concepts**. Wiley, 2018.

3. Robbins, K. A., & Robbins, S. **UNIX Systems Programming: Communication, Concurrency and Threads.** Prentice Hall, 2003.

4. Kerrisk, M. **The Linux Programming Interface**. No Starch Press, 2010.

5. Stevens, W. R., & Rago, S. A. **Advanced Programming in the UNIX Environment**. Addison-Wesley, 2013.

6. IEEE. **POSIX Standard: Interprocess Communication**. IEEE, 2017.

7. Stallings, W. **Operating Systems: Internals and Design Principles**. Pearson, 2018.

8. Love, R. **Linux Kernel Development**. Addison-Wesley, 2010.

9. McKusick, M. K., & Neville-Neil, G. V. **The Design and Implementation of the FreeBSD Operating System.** Addison-Wesley, 2015.