# Министерство образования Республики Беларусь

УПРАВЛЕНИЕ ОБРАЗОВАНИЯ МОГИЛЕВСКОГО ОБЛАСТНОГО ИСПОЛНИТЕЛЬНОГО КОМИТЕТА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«Могилевский государственный политехнический колледж»

# Домашняя контрольная работа №1

По дисциплине: «**Системное программное обеспечение**»

Группа ПО-455

## Выполнил О. И. Любаль

Шифр 14

# 2022

**17 Дайте определение термину «Процесс» в ОС Linux**

Термин "процесс" впервые появился при разработке операционной системы Multix и имеет несколько определений, которые используются в зависимости от контекста. Процесс - это:

1. программа на стадии выполнения
2. "объект", которому выделено процессорное время
3. асинхронная работа

Для описания состояний процессов используется несколько моделей. Самая простая модель - это модель трех состояний. Модель состоит из:

1. состояния выполнения
2. состояния ожидания
3. состояния готовности

Выполнение - это активное состояние, во время которого процесс обладает всеми необходимыми ему ресурсами. В этом состоянии процесс непосредственно выполняется процессором.

Ожидание - это пассивное состояние, во время которого процесс заблокирован, он не может быть выполнен, потому что ожидает какое-то событие, например, ввода данных или освобождения нужного ему устройства.

Готовность - это тоже пассивное состояние, процесс тоже заблокирован, но в отличие от состояния ожидания, он заблокирован не по внутренним причинам (ведь ожидание ввода данных - это внутренняя, "личная" проблема процесса - он может ведь и не ожидать ввода данных и свободно выполняться - никто ему не мешает), а по внешним, независящим от процесса, причинам. Когда процесс может перейти в состояние готовности? Предположим, что наш процесс выполнялся до ввода данных. До этого момента он был в состоянии выполнения, потом перешел в состояние ожидания - ему нужно подождать, пока мы введем нужную для работы процесса информацию. Затем процесс хотел уже перейти в состояние выполнения, так как все необходимые ему данные уже введены, но не тут-то было: так как он не единственный процесс в системе, пока он был в состоянии ожидания, его "место под солнцем" занято - процессор выполняет другой процесс. Тогда нашему процессу ничего не остается как перейти в состояние готовности: ждать ему нечего, а выполняться он тоже не может.

Из состояния готовности процесс может перейти только в состояние выполнения. В состоянии выполнения может находится только один процесс на один процессор. Если у вас n-процессорная машина, у вас одновременно в состоянии выполнения могут быть n процессов.

Из состояния выполнения процесс может перейти либо в состояние ожидания или состояние готовности. Почему процесс может оказаться в состоянии ожидания, мы уже знаем - ему просто нужны дополнительные данные или он ожидает освобождения какого-нибудь ресурса, например, устройства или файла. В состояние готовности процесс может перейти, если во время его выполнения, квант времени выполнения "вышел". Другими словами, в операционной системе есть специальная программа - планировщик, которая следит за тем, чтобы все процессы выполнялись отведенное им время. Например, у нас есть три процесса. Один из них находится в состоянии выполнения. Два других - в состоянии готовности. Планировщик следит за временем выполнения первого процесса, если "время вышло", планировщик переводит процесс 1 в состояние готовности, а процесс 2 - в состояние выполнения. Затем, когда, время отведенное, на выполнение процесса 2, закончится, процесс 2 перейдет в состояние готовности, а процесс 3 - в состояние выполнения.

Диаграмма модели трех состояний представлена на рисунке 1.

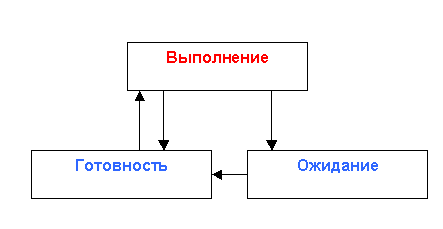


Рисунок 1. Модель трех состояний

Более сложная модель - это модель, состоящая из пяти состояний. В этой модели появилось два дополнительных состояния: рождение процесса и смерть процесса. Рождение процесса - это пассивное состояние, когда самого процесса еще нет, но уже готова структура для появления процесса. Как говорится в афоризме: "Мало найти хорошее место, надо его еще застолбить", так вот во время рождения как раз и происходит "застолбление" этого места. Смерть процесса - самого процесса уже нет, но может случиться, что его "место", то есть структура, осталась в списке процессов. Такие процессы называются зобми

Диаграмма модели пяти состояний представлена на рисунке 2.

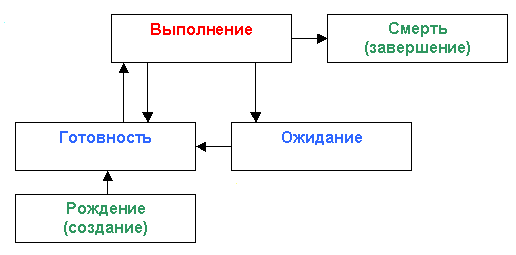


Рисунок 2. Модель пяти состояний

Над процессами можно производить следующие операции:

1. Создание процесса - это переход из состояния рождения в состояние готовности
2. Уничтожение процесса - это переход из состояния выполнения в состояние смерти
3. Восстановление процесса - переход из состояния готовности в состояние выполнения
4. Изменение приоритета процесса - переход из выполнения в готовность
5. Блокирование процесса - переход в состояние ожидания из состояния выполнения
6. Пробуждение процесса - переход из состояния ожидания в состояние готовности
7. Запуск процесса (или его выбор) - переход из состояния готовности в состояние выполнения

Для создания процесса операционной системе нужно:

1. Присвоить процессу имя
2. Добавить информацию о процессе в список процессов
3. Определить приоритет процесса
4. Сформировать блок управления процессом
5. Предоставить процессу нужные ему ресурсы

Процесс не может взяться из ниоткуда: его обязательно должен запустить какой-то процесс. Процесс, запущенный другим процессом, называется дочерним (child) процессом или потомком. Процесс, который запустил процесс называется родительским (parent), родителем или просто - предком. У каждого процесса есть два атрибута - PID (Process ID) - идентификатор процесса и PPID (Parent Process ID) - идентификатор родительского процесса.

Процессы создают иерархию в виде дерева. Самым "главным" предком, то есть процессом, стоящим на вершине этого дерева, является процесс init (PID=1).

**46 Объясните, что представляет собой индексный дескриптор в ОС Linux, какую информацию он содержит**

**inode** (произносится а́йнод или ино́д), индексный дескриптор — это структура данных в традиционных для ОС UNIX файловых системах (ФС), таких как UFS. В этой структуре хранится метаинформация о стандартных файлах, каталогах или других объектах файловой системы, кроме непосредственно данных и имени. Каждый файл ассоциирован с одним inode, хотя может иметь несколько имен в системе, каждое из которых указывает на один и тот же inode.

Индексный дескриптор содержит информацию о расположении данных файла. Поскольку дисковые блоки хранения данных файла в бщем случае располагаются не последовательно, inode, должен хранить физические адреса всех блоков, принадлежащих данному файлу. В индексном дескрипторе эта информация хранится в виде массива, каждый элемент которого содержит физический адрес дискового блока, а индексом массива является номер логического блока файла. Массив имеет фиксированный размер и состоит из 13 элементов. При этом первые 10 элементов адресуют непосредственно блоки хранения данных файла. Одиннадцатый элемент адресует блок, в свою очередь содержащий адреса блоков хранения данных файла. Двенадцатый элемент указывает на дисковый блок, также хранящий адреса блоков, каждый из который адресует блок хранения данных файла. И, наконец, тринадцатый элемент используется для тройной косвенной адресации, когда для нахождения адреса блока хранения данных файлаиспользуются три дополнительных блока. Такой подход позволяет при относительно небольшом фиксированном размере индексного дескриптора поддерживать работу с файлами, размер которых может изменяться от нескольких байтов до десятка мегабайтов. Для относительно небольших файлов (до 10 Кбайт при размере блока 1024 байтов) используется прямая индексация, обеспечивающая максимальную производительность. Для файлов, размер которых не превышает 266 кбайт (10 кбайт + 256х1024), достаточно простой косвенной адресации. Наконец, при использовании тройной косвенной адресации можно обеспечить доступ к 16777216 блокам (256х256х256).

Стандарты POSIX описывают поведение файловой системы как потомка традиционных файловых систем UNIX — UFS. Регулярные файлы должны иметь следующие атрибуты:

длина файла в байтах;

идентификатор (ID) устройства (это идентифицирует устройство, содержащее файл);

ID пользователя, являющегося владельцем файла;

ID группы файла;

режим файла, определяющий какие пользователи могут считывать, записывать и запускать файл;

Timestamp указывает дату последнего изменения инода (ctime, change time), последней модификации содержимого файла (mtime, modification time), и последнего доступа (atime, access time);

счетчик ссылок указывают количество жестких ссылок, указывающих на индексный дескриптор;

указатели на блоки диска, хранящие содержимое файла (подробнее...).

Системный вызов stat считывает номер индексного дескриптора файла и некоторую информацию из него.

Индексный дескриптор не содержит:

имени файла, которое содержится в блоках хранения данных каталога;

содержимого файла, которое размещено в блоках хранения данных.

Множество программ, используемых системными администраторами в операционной системе (ОС) UNIX, часто используют номера индексных дескрипторов для обозначения файлов. Популярная встроенная программа проверки жестких дисков fsck или команда pfiles могут послужить в данном случае примерами, так как у них есть необходимость естественным образом конвертировать номера индексных дескрипторов в пути файлов и обратно. Это может быть дополнено использованием программы поиска файлов find с ключом -inum или командой ls с соответствующим ключом (которым на большинстве платформ является -i). Иноды могут 'закончиться'. В этом случае нельзя записать информацию на устройство, даже если там достаточно свободного места.

**75 Опишите алгоритм Деккера, позволяющий разрешить проблему взаимного исключения путем использования только одной блокировки памяти**

**100 Опишите семафоры Дейкстры. Объясните, чем обеспечивается взаимное исключение при выполнении примитивов P и V.**

**Список используемых источников**

1. Артамонова, Н.В. Операционные системы для организации производства в промышленности/Н.В. Артамонова – СПб: ГУАП, 2012.
2. Гордеев, А.В. Системное программное обеспечение / А.В.Гордеев, А.Ю. Молчанов– СПб: Питер, 2003.
3. Дроздов, С.Н. Операционные системы: учебное пособие / С.Н.Дроздов. – РнД: Феникс, 2016.
4. Иртегов, Д. Введение в операционные системы / Д. Иртегов – СПб: BHV , 2012.
5. Коньков, К.А. Устройство и функционирование ОС Windows / К.А. Коньков – М: Бином, 2012.
6. Партыка, Т.Л. Операционные системы, среды и оболочки: Учебное пособие/ Т.Л.Партыка, И.И.Попов. – М: Форум, 2018.