## Концепция процессов и потоков. Задание, процессы, потоки (нити), волокна

Одним из основных понятий, связанных с операционными системами, является *процесс* – абстрактное понятие, описывающее работу программы. Все функционирующее на компьютере программное обеспечение, включая и операционную систему, можно представить набором процессов.

Задачей ОС является управление процессами и ресурсами компьютера или, точнее, организация рационального использования ресурсов в интересах наиболее эффективного выполнения процессов. Для решения этой задачи операционная система должна располагать информацией о текущем состоянии каждого процесса и ресурса. Универсальный подход к предоставлению такой информации заключается в создании и поддержке таблиц с информацией по каждому объекту управления.

Общее представление об этом можно получить из рис. 5.1, на котором показаны таблицы, поддерживаемые операционной системой: для памяти, устройств ввода-вывода, файлов (программ и данных) и процессов. Хотя детали таких таблиц в разных ОС могут отличаться, по сути, все они поддерживают информацию по этим четырем категориям. Располагающий одними и теми же аппаратными ресурсами, но управляемый различными ОС, компьютер может работать с разной степенью эффективности. Наибольшие сложности в управлении ресурсами компьютера возникают в мультипрограммных ОС.

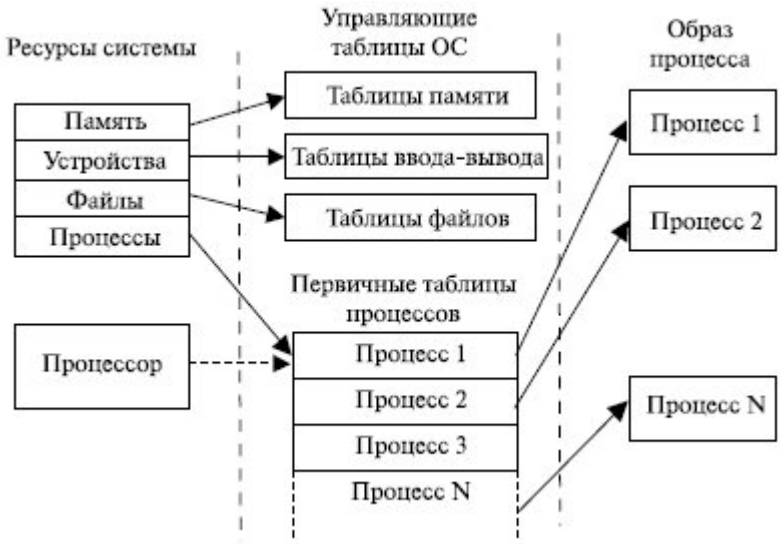


**Рис. 5.1.** Таблицы ОС

*Мультипрограммирование* (многозадачность, multitasking) – это такой способ организации вычислительного процесса, при котором на одном процессоре попеременно выполняются несколько программ. Чтобы поддерживать мультипрограммирование, ОС должна определить для себя внутренние единицы работы, между которыми будут разделяться процессор и другие ресурсы компьютера. В ОС пакетной обработки, распространенных в компьютерах второго и сначала и третьего поколения, такой единицей работы было задание. В настоящее время в большинстве операционных систем определены два типа единиц работы: более крупная единица – процесс, или задача, и менее крупная – *поток*, или *нить*. Причем процесс выполняется в форме одного или нескольких потоков.

Вместе с тем, в некоторых современных ОС вновь вернулись к такой единице работы, как *задание* (Job), например, в Windows. Задание в Windows представляет собой набор из одного или нескольких процессов, управляемых как единое целое. В частности, с каждым заданием ассоциированы *квоты* и *лимиты* ресурсов, хранящиеся в соответствующем объекте задания. Квоты включают такие пункты, как максимальное количество процессов (это не позволяет процессам задания создавать бесконтрольное количество дочерних процессов), суммарное время центрального процессора, доступное для каждого процесса в отдельности и для всех процессов вместе, а также максимальное количество используемой памяти для процесса и всего задания. Задания также могут ограничивать свои процессы в вопросах безопасности, например, получать или запрещать права администратора (даже при наличии правильного пароля).

Процессы рассматриваются операционной системой как заявки или контейнеры для всех видов ресурсов, кроме одного – процессорного времени. Это важнейший ресурс распределяется операционной системой между другими единицами работы – потоками, которые и получили свое название благодаря тому, что они представляют собой последовательности (потоки выполнения) команд. Каждый процесс начинается с одного потока, но новые потоки могут создаваться (порождаться) процессом динамически. В простейшем случае процесс состоит из одного потока, и именно таким образом трактовалось понятие "процесс" до середины 80-х годов (например, в ранних версиях UNIX). В некоторых современных ОС такое положение сохранилось, т.е. понятие "поток" полностью поглощается понятием "процесс".

Как правило, поток работает в пользовательском режиме, но когда он обращается к системному вызову, то переключается в режим ядра. После завершения системного вызова поток продолжает выполняться в режиме пользователя. У каждого потока есть два стека, один используется в режиме ядра, другой – в режиме пользователя. Помимо состояния (текущие значения всех объектов потока) идентификатора и двух стеков, у каждого потока есть контекст (в котором сохраняются его регистры, когда он не работает), приватная область для его локальных переменных, а также может быть собственный маркер доступа (информация о защите). Когда поток завершает работу, он может прекратить свое существование. Процесс завершается, когда прекратит существование последний активный поток.

Взаимосвязь между заданиями, процессами и потоками показана на рис. 5.2.

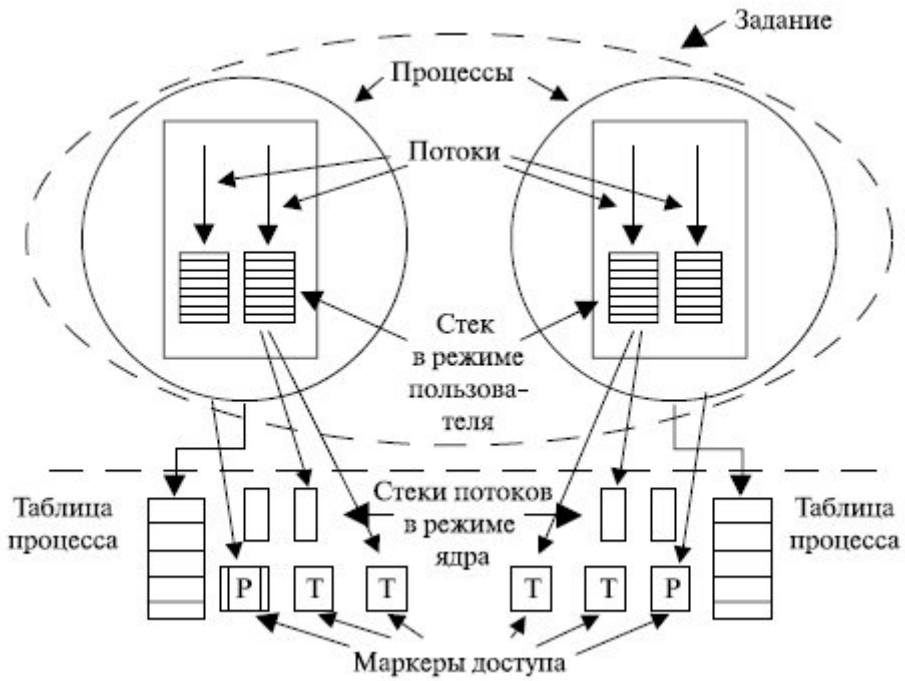


**Рис. 5.2.** Задания, процессы, потоки

Переключение потоков в ОС занимает довольно много времени, так как для этого необходимы переключение в режим ядра, а затем возврат в режим пользователя. Достаточно велики затраты процессорного времени на планирование и диспетчеризацию потоков. Для предоставления сильно облегченного псевдопараллелизма в Windows 2000 (и последующих версиях) используются *волокна* (Fiber), подобные потокам, но планируемые в пространстве пользователя создавшей их программой. У каждого потока может быть несколько волокон, с той разницей, что когда волокно логически блокируется, оно помещается в очередь блокированных волокон, после чего для работы выбирается другое волокно в контексте того же потока. При этом ОС "не знает" о смене волокон, так как все тот же поток продолжает работу.

Таким образом, существует иерархия рабочих единиц операционной системы, которая применительно к Windows выглядит следующим образом (рис. 5.3).

Возникает вопрос: зачем нужна такая сложная организация работ, выполняемых операционной системой? Ответ нужно искать в развитии теории и практики мультипрограммирования, цель которой – в обеспечении максимально эффективного использования главного ресурса вычислительной системы – центрального процессора (нескольких центральных процессоров).

Поэтому прежде чем переходить к рассмотрению современных принципов управления процессором, процессами и потоками, следует остановиться на основных принципах мультипрограммирования.

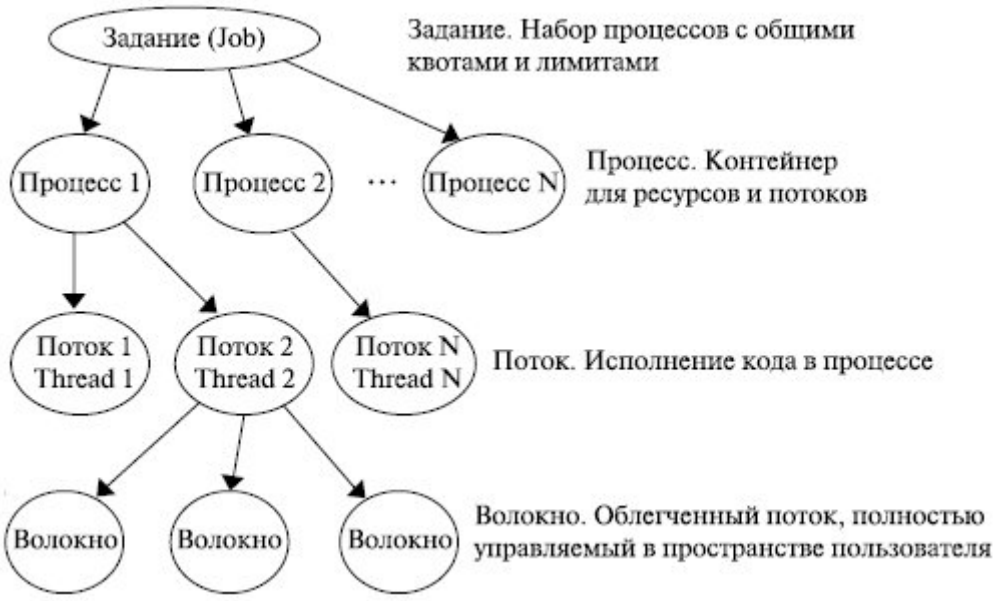


**Рис. 5.3.** Иерархия рабочих единиц ОС

## 5.2. Мультипрограммирование. Формы многопрограммной работы

Мультипрограммирование призвано повысить эффективность использования вычислительной системы. Однако эффективность может пониматься по-разному. Наиболее характерными показателями эффективности вычислительных систем являются:

- пропускная способность – количество задач, выполняемых системой в единицу времени;
- удобство работы пользователей, заключающихся, в частности, в том, что они могут одновременно работать в интерактивном режиме с несколькими приложениями на одной машине;
- реактивность системы – способность выдерживать заранее заданные (возможно, очень короткие) интервалы времени между запуском программы и получением конечного результата.

В зависимости от выбора одного из этих показателей эффективности ОС делятся на системы пакетной обработки, системы разделения времени и системы реального времени (некоторые ОС могут поддерживать одновременно несколько режимов).

Системы *пакетной обработки* предназначались для решения задач в основном вычислительного характера, не требующих быстрого получения результатов. Максимальная пропускная способность компьютера достигается в этом случае минимизацией простоев его устройств и прежде всего процессора. Для достижения этой цели пакет заданий формируется так, чтобы получающаяся мультипрограммная смесь сбалансированно загружала все устройства машины. Например, в такой смеси желательно присутствие задач вычислительного характера и с интенсивным вводом-выводом. Однако в этом случае трудно гарантировать сроки выполнения того или иного задания.

В благоприятных случаях общее время выполнения смеси задач меньше, чем суммарное время их последовательного выполнения. При этом времени выполнения отдельной задачи может быть затрачено больше, чем при монопольном ее выполнении (рис. 5.4).

В *системах разделения времени* пользователям (в частном случае – одному) предоставляется возможность интерактивной работы сразу с несколькими приложениями. Для этого каждое приложение должно регулярно получать возможность "общения" с пользователем. Эта проблема решается за счет того, что ОС принудительно периодически приостанавливает приложения, не дожидаясь, когда они "добровольно" освободят процессор.

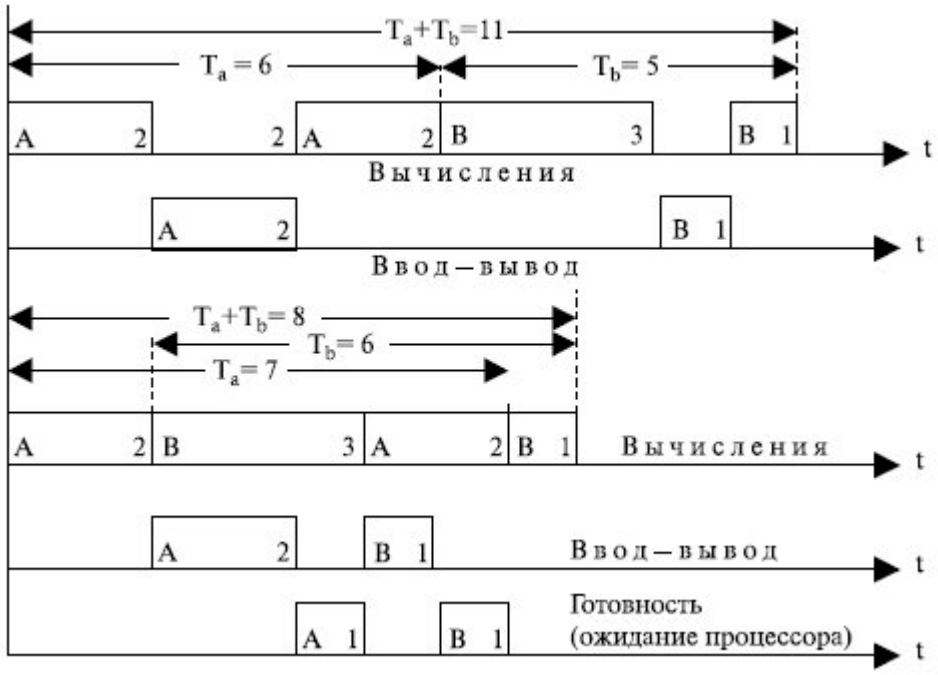


**Рис. 5.4.** Иллюстрация эффекта мультипрограммирования

Всем приложениям попеременно выделяются кванты времени процессора, таким образом, пользователи, запустившие программы на выполнение, получают возможность поддерживать с ними диалог (рис. 5.5) со своего терминала. Если время кванта выбрано достаточно небольшим, то у всех пользователей складывается впечатление единоличной работы на машине.

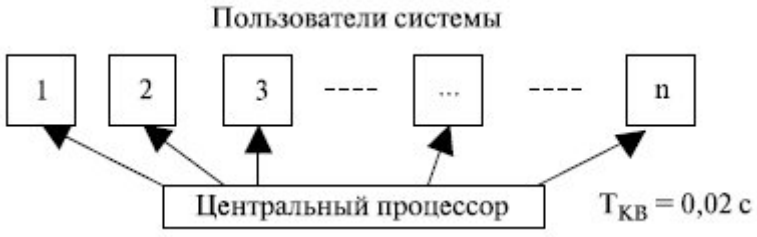


**Рис. 5.5.** Система разделения времени

Системы *реального времени* предназначены для управления техническими объектами (спутник, ракета, атомные электростанции, станок, научная установка и др.), технологическими процессами (гальваническая линия, доменный процесс и т.п.), системами обслуживания разного рода (резервирование авиабилетов, оплата покупок и счетов и др.). Во всех этих случаях существует предельно допустимое время, в течение которого должна быть выполнена та или иная программа управления объектом. В противном случае возможны нежелательные последствия вплоть до аварии.

Критерием эффективности ОС в этом случае является способность выдерживать заранее заданные интервалы времени между запуском программы и получением результата. Это время называется временем реакции системы, а соответствующее свойство – реактивностью. Требования ко времени реакции зависят от специфики управляемого объекта или процесса. В системах реального времени мультипрограммная смесь представляет собой фиксированный набор заранее разработанных программ решения функциональных задач управления объектом или процессом. Выбор программы на выполнение осуществляется по прерываниям (исходя из текущего состояния объекта) или в соответствии с расписанием плановых работ.

В системе реального времени обычно закладывается запас вычислительной мощности на случай пиковой нагрузки, а также принимаются меры обеспечения высокой надежности работы системы (резервирование, дублирование, троирование с мажоритарным элементом и др.).

Интересная форма мультипрограммной работы связана с *мультипроцессорной обработкой*. Мультипроцессорная обработка – это способ организации вычислительного процесса в системе с несколькими процессорами, при котором несколько задач (процессов, потоков) могут одновременно выполняться на разных процессорах системы. Концепция мультипроцессирования не нова, она известна с 70-х годов, однако стала доступной в широком масштабе лишь в последнее десятилетие, особенно с появлением многопроцессорных ПК (часто в качестве серверов ЛВС).

В отличие от мультипрограммной обработки, в мультипроцессорных системах несколько задач выполняется одновременно, т.к. имеется несколько процессоров. Однако это не исключает мультипрограммной обработки на каждом процессоре. При этом резко усложняются все алгоритмы управления ресурсами, т.е. операционная система. Современные ОС, как правило, поддерживают мультипроцессирование (Sun Solaris 2.x, Santa Cruz Operation Open Server 3.x, OS/2, Windows NT/2000/2003/XP, NetWare, начиная с версии 4.1 и др.).

Мультипроцессорные системы часто характеризуют как *симметричные* и как *несимметричные*. Эти термины относятся, с одной стороны, к архитектуре вычислительной системы, а с другой – к способу организации вычислительного процесса.

Симметричная архитектура мультипроцессорной системы предполагает однотипность и единообразие включения процессоров и большую разделяемую между этими процессорами память. Масштабируемость, т.е. возможность наращивания числа процессоров, в данном случае ограничена, т.к. все они используют одну и ту же оперативную память и, следовательно, должны располагаться в одном корпусе. В *симметричных* архитектурах вычислительных систем легко реализуется *симметричное мультипроцессирование* общей для всех процессоров операционной системой. При этом все процессоры равноправно участвуют и в управлении вычислительным процессом, и в выполнении прикладных задач. Разные процессоры могут в какой-то момент времени одновременно обслуживать как разные, так и одинаковые модули общей ОС. Для этого программы ОС должны быть *реентерабельными* (повторновходимыми).

Операционная система полностью децентрализована. Ее модули выполняются на любом доступном процессоре. Как только процессор завершает выполнение очередной задачи, он передает управление планировщику задач. Последний выбирает из общей для всех процессоров системной очереди задачу, которая будет выполняться на данном процессоре следующей.

В вычислительных системах с *асимметричной* архитектурой процессоры могут быть различными как по характеристикам (производительность, система команд), так и по функциональной роли в работе системы. Например, могут быть выделены процессоры для вычислений, ввода-вывода и др. Эта неоднородность ведет к структурным отличиям во фрагментах системы, содержащих разные процессоры (разные схемы подключения, наборы периферийных устройств, способы взаимодействия процессоров с устройствами и др.).

Масштабирование в таких системах реализуется иначе, поскольку отсутствует требование единого корпуса. Система может состоять из нескольких устройств, каждое из которых содержит один или несколько процессоров. Масштабирование в данном случае называют горизонтальным, а мультипроцессорную систему – кластерной. В кластерной системе может быть реализовано только асимметричное мультипроцессирование с организацией вычислительного процесса по принципу "ведущий – ведомый". Этот наиболее простой способ может быть использован и в вычислительных системах с симметричной архитектурой. В таких системах ОС работает на одном процессоре, который называется ведущим и организует централизованное управление вычислительным процессом и распределением всех ресурсов системы.