Концепции процесса

Основным понятием в любой операционной системе является **процесс**: абстракция, описывающая выполняющуюся программу. Они поддерживают возможность осуществления **(псевдо) параллельных операций** даже при наличии всего одного центрального процессора. Они превращают один центральный процессор в несколько виртуальных.

В модели процесса все выполняемое на компьютере программное обеспечение, иногда включая операционную систему, сведено к ряду **последовательных процессов,** или, для краткости, просто процессов**. Процесс — это просто экземпляр выполняемой программы**, включая текущие значения счетчика команд, регистров и переменных. Концептуально у каждого процесса есть свой, виртуальный, центральный процессор. Разумеется, на самом деле настоящий центральный процессор постоянно переключается между процессами, но, чтобы понять систему, куда проще думать о **наборе процессов**, запущенных в (псевдо) параллельном режиме, чем пытаться отслеживать, как **центральный процессор переключается** между программами. Это постоянное переключение между процессами, называется **мультипрограммированием**, или **многозадачным режимом работы.**

На рис. 2.1, а показан компьютер, работающий в многозадачном режиме и имеющий в памяти четыре программы. На рис. 2.1, б показаны четыре процесса, каждый из которых имеет собственный алгоритм управления (то есть собственный логический счетчик команд) и работает независимо от всех остальных. Понятно, что на самом деле имеется только один физический счетчик команд, поэтому при запуске каждого про- цесса его логический счетчик команд загружается в реальный счетчик. Когда работа с процессом будет на некоторое время прекращена, значение физического счетчика команд сохраняется в логическом счетчике команд, размещаемом процессом в памяти. На рис. 2.1, в показано, что за довольно длительный период наблюдения продвинулись вперед все процессы, но в каждый отдельно взятый момент времени реально работает только один процесс.

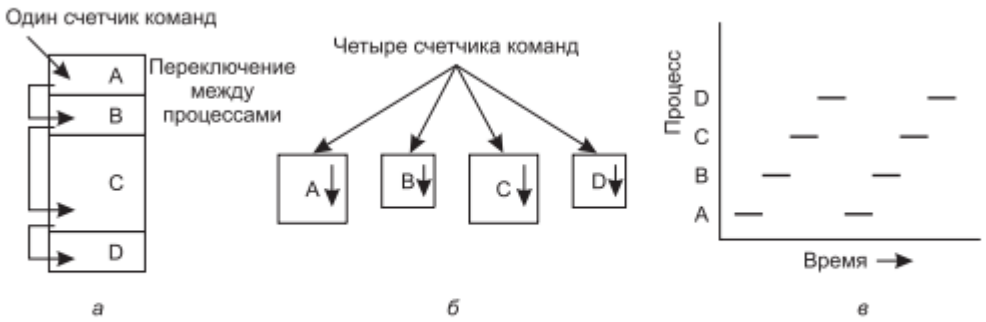

Рис. 2.1. Компьютер: *а* — четыре программы, работающие в многозадачном режиме; *б* — концептуальная модель четырех независимых друг от друга последовательных процессов; *в* — в отдельно взятый момент активна только одна программа

Операционным системам необходим какой-нибудь способ для **создания процессов**. В самых простых системах или в системах, сконструированных для запуска только одного приложения (например, в контроллере микроволновой печи), появляется возможность присутствия абсолютно всех необходимых процессов при вводе системы в действие. Но в универсальных системах нужны определенные способы **создания и прекращения процессов по мере необходимости.**

Существуют четыре основных события, приводящих к созданию процессов:

1. Инициализация системы.
2. Выполнение работающим процессом системного вызова, предназначенного для создания процесса.
3. Запрос пользователя на создание нового процесса.
4. Инициация пакетного задания.

Некоторые из процессов представляют собой **высокоприоритетные процессы**, то есть взаимодействующие с пользователями и выполняющие для них определенную работу. Остальные являются **фоновыми процессами**, не связанными с конкретными пользователями, но выполняющими ряд специфических функций

Фоновые процессы, предназначенные для обработки какой-либо активной деятельности, связанной, например, с электронной почтой, веб-страницами, новостями, выводом информации на печать и т. д., называются **демонами**.

После создания процесс начинает работать и выполняет свою задачу. Рано или поздно процессы будут завершены, обычно в силу следующих обстоятельств:

1. обычного выхода (добровольно);
2. выхода при возникновении ошибки (добровольно);

3. возникновения фатальной ошибки (принудительно);
4. уничтожения другим процессом (принудительно).

Большинство процессов завершаются по окончании своей работы. Когда компилятор откомпилирует заданную ему программу, он осуществляет **системный вызов**, сообщающий операционной системе о завершении своей работы.

Вторая причина завершения — обнаружение процессом **фатальной ошибки**.

Третья причина завершения — ошибка, вызванная самим процессом, чаще всего связанная с **ошибкой в программе**. В качестве примеров можно привести неверную инструкцию, ссылку на несуществующий адрес памяти или деление на нуль.

Четвертая причина, из-за которой процесс может быть завершен, — это выполнение процессом системного вызова, приказывающего операционной системе **завершить некоторые другие процессы.**

Несмотря на самостоятельность каждого процесса, наличие собственного счетчика команд и **внутреннего состояния**, процессам зачастую необходимо взаимодействовать с другими процессами. Один процесс может генерировать выходную информацию, используемую другими процессами в качестве входной информации.

На рис. 2.2 показана диаграмма, отображающая три состояния, в которых может находиться процесс:

1. выполняемый (в данный момент использующий центральный процессор);
2. готовый (работоспособный, но временно приостановленный, чтобы дать возможность выполнения другому процессу);
3. заблокированный (неспособный выполняться, пока не возникнет какое-нибудь внешнее событие).

1. Процесс заблокирован в ожидании ввода
2. Диспетчер выбрал другой процесс
3. Диспетчер выбрал данный процесс
4. Входные данные стали доступны

Рис. 2.2. Процесс может быть в выполняемом, заблокированном или готовом состоянии. Стрелками показаны переходы между этими состояниями

Как показано на рисунке, между этими тремя состояниями могут быть четыре перехода. Переход 1 происходит в том случае, если операционная

система определит, что **процесс в данный момент выполняться не может.**

Переходы 2 и 3 вызываются **планировщиком процессов**, который является частью операционной системы, без какого-либо оповещения самого процесса. Переход 2 происходит, когда планировщик решит, что выполняемый процесс продвинулся достаточно далеко и настало время позволить другому процессу получить долю рабочего времени центрального процессора. Переход 3 происходит, когда все другие процессы получили причитающуюся им долю времени и настал момент предоставить центральный процессор первому процессу для возобновления его выполнения.

Переход 4 осуществляется в том случае, если происходит **внешнее событие**, ожидавшееся процессом (к примеру, поступление входных данных). Если к этому моменту нет других выполняемых процессов, будет вызван переход 3 и процесс возобновится. В противном случае ему придется немного подождать в состоянии готовности, пока не станет доступен центральный процессор и не придет его очередь.

Для реализации модели процессов операционная система ведет таблицу (состоящую из массива структур), называемую **таблицей процессов**, в которой каждая запись соответствует какому-нибудь процессу. (Ряд авторов называют эти записи **блоками управления процессом**.) Эти записи содержат важную информацию о состоянии процесса, включая счетчик команд, указатель стека, распределение памяти, состояние открытых им файлов, его учетную и планировочную информацию и все остальное, касающееся процесса, что должно быть сохранено, когда процесс переключается из состояния выполнения в состояние готовности или блокировки, чтобы позже он мог возобновить выполнение, как будто никогда не останавливался.

В табл. 2.1 показан ряд ключевых полей типовой системы. Поля первого столбца относятся к управлению процессами. Поля остальных двух столбцов относятся к управлению памятью и файлами соответственно. Следует заметить, что наличие тех или иных полей в таблице процессов в большей степени зависит от системы, но в этой таблице изложено основное представление о типе необходимой информации.

Таблица 2.1. Некоторые из полей типичной записи таблицы процессов

Управление процессом	Управление памятью	Управление файлами
Регистры	Указатель на информацию о текстовом сегменте	Корневой каталог
Счетчик команд	Указатель на информацию о сегменте данных	Рабочий каталог
Слово состояния программы	Указатель на информацию о сегменте стека	Дескрипторы файлов
Указатель стека		Идентификатор пользователя
Состояние процесса		Идентификатор группы
Приоритет		
Параметры планирования		
Идентификатор процесса		
Родительский процесс		
Группа процесса		
Сигналы		
Время запуска процесса		
Использованное время процессора		
Время процессора, использованное дочерними процессами		
Время следующего аварийного сигнала		

Периодически операционная система будет принимать решения остановить работу одного процесса и запустить выполнение другого, возможно, из-за того, что первый исчерпал свою долю процессорного времени в предыдущую секунду или две. Если процесс **приостанавливается** таким образом, позже он должен возобновиться именно с того состояния, в котором был остановлен. Это означает, что на период приостановки вся информация о процессе должна быть явным образом где-то сохранена.

Во многих операционных системах вся информация о каждом процессе, за исключением содержимого его собственного адресного пространства, хранится в таблице операционной системы, которая называется таблицей процессов и представляет собой массив (или связанный список) структур, по одной на каждый из существующих на данный момент процессов. Таким образом, процесс (в том числе приостановленный) состоит из собственного адресного пространства, которое обычно называют образом памяти, и записи в таблице процессов с содержимым его регистров, а также другой информацией, необходимой для последующего возобновления процесса.