(+ в. 6.)

Понятие процесса играет ключевую роль в современных ОС. Одно из определения **процесса** - выполняющийся экземпляр программы. Важный аспект процессов с точки зрения ОС – неделимость процесса относительно выделения ресурсов. Система выделяет ресурсы процессу в целом, а не его частям.

Управление процессами, прежде всего, - создание новых процессов и контроль их выполнения. Unix: новые экземпляры программы обычно создаются с помощью вызова функции fork – возвращает PID дочернего процесса родительскому (0 – дочернему), а также функцию clone (позволяет настраивать разделение ресурсов между процессами, но не умеет раздваивать процесс в точке вызова). Windows: CreateProcess

ОС 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.

События, приводящие 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.

Причины **завершения** процессов:
1) обычного выхода (добровольно);
2) выхода при возникновении ошибки (добровольно);
3) возникновения фатальной ошибки (принудительно);
4) уничтожения другим процессом (принудительно).

Процессы

Хотя на первый взгляд кажется, что программа и процесс понятия практически одинаковые, они фундаментально отличаются друг от друга. Программа представляет собой статический набор команд, а процесс это набор ресурсов и данных, использующихся при выполнении программы.

Процесс в Windows состоит из следующих компонентов:
- Структура данных, содержащая всю информацию о процессе, в том числе список открытых дескрипторов
- различных системных ресурсов, уникальный идентификатор процесса, различную статистическую информацию и т.д.;
- Адресное пространство - диапазон адресов виртуальной памяти, которым может пользоваться процесс;
- Исполняемая программа и данные, проецируемые на виртуальное адресное пространство процесса.

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

Потоки

Поток (thread) - некая сущность внутри процесса, получающая процессорное время для выполнения.

В каждом процессе есть минимум один поток. Этот первичный поток создается системой автоматически при создании процесса. Один процесс может владеть несколькими потоками, и тогда они одновременно исполняют код в адресном пространстве процесса. Каждый поток имеет:
- Уникальный идентификатор потока;
- Содержимое набора регистров процессора, отражающих состояние процессора;
- Два стека, один из которых используется потоком при выполнении в режиме ядра, а другой – в пользовательском режиме;
- Закрытую область памяти, называемую локальной памятью потока (thread local storage, TLS) и используемую подсистемами, run-time библиотеками и DLL.

Есть несколько **причин**, почему приложения **создают потоки** в дополнение к их исходному начальному потоку:
1) процессы, обладающие пользовательским интерфейсом, обычно создают потоки для того, чтобы выполнять свою работу и при этом сохранять отзывчивость основного потока к командам пользователя, связанными с вводом данных и управлением окнами;
2) приложения, которые хотят использовать несколько процессоров для масштабирования производительности или же которые хотят продолжать работать, в то время как потоки останавливают свою работу, ожидая синхронизации операций ввода/вывода, создают потоки, чтобы получить дополнительную выгоду от многопоточной работы.