Сигналы

Сигналы — одно из традиционных средств межпроцессного взаимодействия в UNIX. Сигнал может быть отправлен процессу операционной системой или другим процессом. Операционная система использует сигналы для доставки процессу уведомлений об ошибках и неправильном поведении.

При получении сигнала исполнение процесса приостанавливается и запускается специальная подпрограмма — *обработчик сигнала*. Обработчики сигналов могут быть явно определены в исходном тексте исполняемой программы, если же они отсутствуют, используется стандартный обработчик, определённый операционной системой.

У сигнала есть только одна характеристика, несущая информацию — его номер (целое число). Иначе говоря, сигналы — это заранее определённый и пронумерованный список сообщений. Для удобства использования каждый сигнал имеет сокращённое буквенное имя.

Список сигналов и их имён стандартизован и практически не отличается в различных версиях UNIX. Для отправки сигналов процессам используется специальный системный вызов kill и одноимённая ему пользовательская утилита. Стандарт **POSIX** определяет 28 сигналов, вот несколько примеров:
SIGINT (2) — Сигнал передается активному приложению при нажатии сочетания Ctrl+C, по умолчанию завершает процесс.
SIGKILL (9) — Сигнал аварийного завершения процесса. По этому сигналу процесс завершается немедленно — без освобождения ресурсов. Этот сигнал не может быть перехвачен, заблокирован или переопределён самим процессом, всегда используется стандартный обработчик опрационной системы. Этот сигнал используется для гарантированного завершения процесса.
SIGTERM (15) — Сигнал завершения процесса, как правило используется для корректного завершения его работы.
SIGUSR1 (10) и SIGUSR2 (12) — Зарезервированные сигналы под нужды программистов.
Сигналы являются ограниченным средством межпроцесснного обмена. Они прекрасно подходят для уведомлений, но не могут использоваться для передачи информации между процессами. Сигналы передаются без каких-либо сопутствующих данных, поэтому они обычно комбинируются с другими способами обмена.
Вызовы POSIX — *signal*(), *kill*().

Сигналы являются программными прерываниями, которые посылаются процессу, когда случается некоторое событие. Сигналы могут возникать синхронно с ошибкой в приложении, например SIGFPE (ошибка вычислений с плавающей запятой) и SIGSEGV (ошибка адресации), но большинство сигналов является асинхронными. Сигналы могут посылаться процессу, если система обнаруживает программное событие, например, когда пользователь дает команду прервать или остановить выполнение, или получен сигнал на завершение от другого процесса. Сигналы могут прийти непосредственно от ядра ОС, когда возникает сбой аппаратных средств ЭВМ. Система определяет набор сигналов, которые могут быть отправлены процессу. При этом каждый сигнал имеет целочисленное значение и приводит к строго определенным действиям.

Механизм передачи сигналов состоит из следующих частей:

- установление и обозначение сигналов в форме целочисленных значений;
- маркер в строке таблицы процессов для прибывших сигналов;
- таблица с адресами функций, которые определяют реакцию на прибывающие сигналы.

Отдельные сигналы подразделяются на три **класса**:

- системные сигналы (ошибка аппаратуры, системная ошибка и т.д.);
- сигналы от устройств;

- сигналы, определенные пользователем.

Как только сигнал приходит, он отмечается записью в таблице процессов. Если этот сигнал предназначен для процесса, то по таблице указателей функций в структуре описания процесса выясняется, как нужно реагировать на этот сигнал. При этом номер сигнала служит индексом таблицы.

Известно три варианта **реакции** на сигналы:

- вызов собственной функции обработки;
- игнорирование сигнала (не работает для SIGKILL);
- использование предварительно установленной функции обработки по умолчанию.

Чтобы реагировать на разные сигналы, необходимо знать концепции их обработки. Процесс должен организовать так называемый обработчик сигнала в случае его прихода.

Передача сообщений

В *UNIX* ***сообщения*** используются для решения проблемы согласованности процессов при передаче информации, существуют специальные механизмы обмена данными, основанные на мьютексах. Это системные вызовы *send()* и *recieve()*. Первый посылает сообщение заданному адресату, второй получает сообщение от указанного источника. Если сообщения нет, второй запрос блокируется до поступления сообщения либо немедленно возвращает код ошибки.

В *Windows* ***сообщения*** используются для передачи данных. Сообщение *WM_COPYDATA* – стандартное сообщение для передачи участка памяти другому процессу. Работает однонаправленно, принимающий процесс должен расценивать полученные данные как read only. Посылать это сообщение необходимо только с помощью SendMessage, которая в отличие от PostMessage ждет завершения операции.