Командные языки и командные интерпретаторы

Как и в большинстве интерактивных систем, традиционный интерфейс с пользователем ОС UNIX основан на использовании командных языков. Выражаясь несколько тавтологично, можно сказать, что командный язык - это язык, на котором пользователь взаимодействует с системой в интерактивном режиме. Такой язык называется командным, поскольку каждую строку, вводимую с терминала и отправляемую системе, можно рассматривать как команду пользователя по отношению к системе. Одним из достижений ОС UNIX является то, что командные языки этой операционной системы являются хорошо определенными (не очень хороший русский термин, соответствующий совершенно однозначному английскому термину well-defined) и содержат много средств, приближающих их к языкам программирования.

Если рассматривать категорию командных языков с точки зрения общего направления языков взаимодействия человека с компьютером, то они, естественно, относятся к семейству интерпретируемых языков. Коротко охарактеризуем разницу между компилируемыми и интерпретируемыми компьютерными языками. Язык называется компилируемым, если требует, чтобы любая законченная конструкция языка была настолько замкнутой, чтобы обеспечивала возможность изолированной обработки без потребности привлечения дополнительных языковых конструкций. В противном случае понимание языковой конструкции не гарантируется. Житейским примером компилируемого языка является литературный русский язык. Ни один литературный редактор не примет от вас незаконченное сочинение, в котором имеются ссылки на еще не написанные части. Процесс компиляции (литературного редактирования в нашем примере) требует замкнутости языковых конструкций.

Основным преимуществом интерпретируемых языков является то, что в случае их использования программа пишется "инкрементально" (в пошаговом режиме), т.е. человек принимает решение о своем следующем шаге в зависимости от реакции системы на предыдущий шаг. В принципе, предпочтение компилируемых или интерпретируемых языков является предметом личного вкуса конкретного индивидуума (нам известны крупные авторитеты в области программирования - например, Д.Б. Подшивалов,- которые абсолютно уверены, что любая хорошая программа должна быть сначала написана на бумаге и отлажена за столом).

Особенностью командных языков является то, что в большинстве случаев они не используются для программирования в обычном смысле этого слова, хотя на развитом командном языке можно написать любую программу. По нашему мнению, правильным стилем использования командного языка является его применение в основном для непосредственного взаимодействия с системой с привлечением возможностей составления командных файлов (скриптов или сценариев в терминологии ОС UNIX) для экономии повторяющихся рутинных процедур.

Программы, предназначенные для обработки конструкций командных языков, называются командными интерпретаторами. В отличие от компилируемых языков программирования (таких, как Си или Паскаль), для каждого из которых обычно существует много различных компиляторов, командный язык, как правило, неразрывно связан с соответствующим интерпретатором. Когда ниже мы будем говорить о различных представителях командных языков ОС UNIX, относящихся к семейству shell, то каждый раз под одноименным названием мы будем подразумевать и соответствующий интерпретатор.

Общая характеристика командных языков

В этом пункте и далее в данном разделе мы будем более конкретно говорить о командных языках семейства shell. Основное назначение этих языков (их разновидностей существует достаточно много, но мы рассмотрим только три наиболее распространенные варианта - Bourne-shell, C-shell и Korn-shell) состоит в том, чтобы предоставить пользователям удобные средства взаимодействия с системой. Что это означает? Языки не даром называются командными. Они предназначены для того, чтобы дать пользователю возможность выполнять команды, предназначенные для исполнения некоторых действий операционной системы. Существует два вида команд.

Собственные команды shell (такие как cd, echo, exec и т.д.) выполняются непосредственно интерпретатором, т.е. их семантика встроена в соответствующий язык. Имена других команд на самом деле являются именами файлов, содержащих выполняемые программы. В случае вызова такой команды интерпретатор командного языка с использованием соответствующих системных вызовов запускает параллельный процесс, в котором выполняется нужная программа. Конечно, смысл действия подобных команд является достаточно условным, поскольку зависит от конкретного наполнения внешних файлов. Тем не менее, в описании каждого языка содержатся и характеристики "внешних команд" (например, find,grep, cc и т.д.) в расчете на то, что здравомыслящие пользователи (и их администраторы) не будут изменять содержимое соответствующих файлов.

Существенным компонентом командного языка являются средства, позволяющие разнообразными способами комбинировать простые команды, образуя на их основе составные команды. В семействе языков shell возможны следующие средства комбинирования. В одной командной строке (важный термин, означающий единицу информационного взаимодействия с командным интерпретатором) можно указать список команд, которые должны выполняться последовательно, или список команд, которые должны выполняться "параллельно" (т.е. независимо одна от другой).

Очень важной особенностью семейства языков shell являются возможности перенаправления ввода/вывода и организации конвейеров команд. Естественно, эти возможности опираются на базовые средства ОС UNIX (см. п. 2.1.8). Кратко напомним, в чем они состоят. Для каждого пользовательского процесса (а внешние команды shell выполняются в рамках отдельных пользовательских процессов) предопределены три выделенных дескриптора файлов: файла стандартного ввода (standard input), файла стандартного вывода (standard output) и файла стандартного вывода сообщений об ошибках (standard error). Хорошим стилем программирования в среде ОС UNIX является такой, при котором любая программа читает свои вводные данные из файла стандартного ввода, а выводит свои результаты и сообщения об ошибках в файлы стандартного вывода и стандартного вывода сообщений об ошибках соответственно. Поскольку любой порожденный процесс "наследует" все открытые файлы своего предка (см. п. 2.1.7), то при программировании команды рекомендуется не задумываться об источнике вводной информации программы, а также конкретном ресурсе, поддерживающим вывод основных сообщений и сообщений об ошибках. Нужно просто пользоваться стандартными файлами, за конкретное определение которых отвечает процесс-предок (заметим, что по умолчанию все три файла соответствуют вводу и выводу на тот терминал, с которого работает пользователь).

Что обеспечивает такая дисциплина работы? Прежде всего возможность создания программ, "нейтральных" по отношению к источнику своих вводных данных и назначению своих выводных данных. Собственно, на этом и основаны принципы перенаправления ввода/вывода и организации конвейера команд. Все очень просто. Если вы пишете в командной строке конструкцию

com1 par1, par2, ..., parn > file\_name ,

то это означает, что для стандартного вывода команды com1 будет использоваться файл с именемfile\_name. Если вы пишете

file\_name < com1 par1, par2, ..., parn ,

то команда com1 будет использовать файл с именем file\_name в качестве источника своего стандартного ввода. Если же вы пишете

com1 par1, par2, ..., parn | com2 par1, par2, ..., parm ,

то в качестве стандартного ввода команды com2 будет использоваться стандартный вывод команды com1. (Конечно, при организации такого рода "конвейеров" используются программные каналы, см. п. 3.4.4.)

Конвейер представляет собой простое, но исключительно мощное средство языков семейства shell, поскольку позволяет во время работы динамически создавать "комбинированные" команды. Например, указание в одной командной строке последовательности связанных конвейером команд

ls -l | sort -r

приведет к тому, что подробное содержимое текущего каталога будет отсортировано по именам файлов в обратном порядке и выдано на экран терминала. Если бы не было возможности комбинирования команд, до для достижения такой возможности потребовалось бы внедрение в программу ls возможностей сортировки.

Последнее, что нам следует обсудить в этом пункте, это существо команд семейства языков shell. Различаются три вида команд. Первая разновидность состоит из команд, встроенных в командный интерпретатор, т.е. составляющих часть его программного кода. Эти команды предопределены в командном языке, и их невозможно изменить без переделки интерпретатора. Команды второго вида - это выполняемые программы ОС UNIX. Обычно они пишутся на языке Си по определенным правилам (см. п. 5.2.1). Такие команды включают стандартные утилиты ОС UNIX, состав которых может быть расширен любым пользователем (если, конечно, он еще способен программировать). Наконец, команды третьего вида (так называемые скрипты языка shell) пишутся на самом языке shell. Это то, что традиционно называлось командным файлом, поскольку на самом деле представляет собой отдельный файл, содержащий последовательность строк в синтаксисе командного языка.

Базовые возможности семейства командных интерпретаторов

Здесь под командным интерпретатором мы будем понимать не соответствующую программу, а отдельный сеанс ее выполнения. Если в случае компилируемых программ следует различать наличие готовой к выполнению программы и факт запуска такой программы, то в случае интерпретируемых программ приходится различать случаи наличия интерпретатора, программы, которая может быть проинтерпретирована, и запуска интерпретатора с предоставлением ему интерпретируемой программы. Основным свойством динамически выполняемой программы (неважно, является ли она предварительно откомпилированной или интерпретируемой) является наличие ее динамического контекста, в частности, набора определенных переменных, содержащих установленные в них значения.

В случае командных интерпретаторов семейства shell имеется два вида динамических контекстов выполнения интерпретируемой программы. Первый вид контекста состоит из предопределенного набора переменных, которые существуют (и обладают значениями) независимо от того, какая программа интерпретируется. В терминологии ОС UNIX этот набор переменных называется окружением или средой сеанса выполнения командной программы. С другой стороны, программа при своем выполнении может определить и установить дополнительные переменные, которые после этого используются точно так же, как и предопределенные переменные. Спецификой командных интерпретаторов семейства shell (связанной с их ориентацией на обеспечение интерфейса с операционной системой) является то, что все переменные shell-программы (сеанса выполнения командного интерпретатора) являются текстовыми, т.е. содержат строки символов.

Любая разновидность языка shell представляет собой развитый компьютерный язык, и объем этого курса не позволяет представить в деталях хотя бы один из них. Однако в следующих подразделах мы постараемся кратко познакомить вас с особенностями трех распространенных вариантов языка shell.

Bourne-shell

Bourne-shell является наиболее распространенным командным языком (и одновременно командным интерпретатором) системы UNIX. Вот основные определения языка Bourne-shell (конечно, мы приводим неформальные определения, хотя язык обладает вполне формализованным синтаксисом):

Пробел - это либо символ пробела, либо символ горизонтальной табуляции.

Имя - это последовательность букв, цифр или символов подчеркивания, начинающаяся с буквы или подчеркивания.

Параметр - имя, цифра или один из символов \*, @, #, ?, -, $, !.

Простая команда - последовательность слов, разделенных пробелами. Первое слово простой команды - это ее имя, остальные слова - аргументы команды (имя команды считается ее нулевым аргументом - см. п. 5.2.1).

Метасимволы. Аргументы команды (которые обычно задают имена файлов) могут содержать специальные символы (метасимволы) "\*", "?", а также заключенные в квадратные скобки списки или указанные диапазоны символов. В этом случае заданное текстовое представление параметра называется шаблоном. Указание звездочки означает, что вместо указанного шаблона может использоваться любое имя, в котором звездочка заменена на произвольную текстовую строку. Задание в шаблоне вопросительного знака означает, что в соответствующей позиции может использоваться любой допустимый символ. Наконец, при использовании в шаблоне квадратных скобок для генерации имени используются все указанные в квадратных скобках символы. Команда применяется в цикле для всех осмысленных сгенерированных имен.

Значение простой команды - это код ее завершения, если команда заканчивается нормально, либо 128 + код ошибки, если завершение команды не нормальное (все значения выдаются в текстовом виде).

Команда - это либо простая команда, либо одна из управляющих конструкций (специальных встроенных в язык конструкций, предназначенных для организации сложных shell-программ).

Командная строка - текстовая строка на языке shell.

shell-процедура (shell-script) - файл с программой, написанной на языке shell.

Конвейер - последовательность команд, разделенных символом "|". При выполнении конвейера стандартный вывод каждой команды конвейера, кроме последней, направляется на стандартный вход следующей команды. Интерпретатор shell ожидает завершения последней команды конвейера. Код завершения последней команды считается кодом завершения всего конвейера.

Список - последовательность нескольких конвейеров, соединенных символами ";", "&", "&&", "||", и, может быть, заканчивающаяся символами ";" или "&". Разделитель между конвейерами";" означает, что требуется последовательное выполнение конвейеров; "&" означает, что конвейеры могут выполняться параллельно. Использование в качестве разделителя символов "&&"(и "||") означает, что следующий конвейер будет выполняться только в том случае, если предыдущий конвейер завершился с кодом завершения "0" (т.е. абсолютно нормально). При организации списка символы ";" и "&" имеют одинаковые приоритеты, меньшие, чем у разделителей "&&" и "||".

В любой точке программы может быть объявлена (и установлена) переменная с помощью конструкции "имя = значение" (все значения переменных - текстовые). Использование конструкций $имя или ${имя} приводит к подстановке текущего значения переменной в соответствующее слово.

Предопределенными переменными Bourne-shell, среди прочих, являются следующие:

HOME - полное имя домашнего каталога текущего пользователя;

PATH - список имен каталогов, в которых производится поиск команды, при ее указании коротким именем;

PS1 - основное приглашение shell ко вводу команды;

и т.д.

Вызов любой команды можно окружить одиночными кавычками (`), и тогда в соответствующую строку будет подставлен результат стандартного вывода этой команды.

Среди управляющих конструкций языка содержатся следующие: for и while для организации циклов, if для организации ветвлений и case для организации переключателей (естественно, все это специфически приспособлено для работы с текстовыми значениями).

C-shell

Командный язык C-shell главным образом отличается от Bourne-shell тем, что его синтаксис приближен к синтаксису языка Си (это, конечно, не означает действительной близости языков). В основном, C-shell включает в себя функциональные возможности Bourne-shell. Если не вдаваться в детали, то реальными отличиями C-shell от Bourne-shell является поддержка протокола (файла истории) и псевдонимов.

В протоколе сохраняются введенные в данном сеансе работы с интерпретатором командные строки. Размер протокола определяется установкой предопределенной переменной history, но последняя введенная командная строка сохраняется всегда. В любом месте текущей командной строки в нее может быть подставлена командная строка (или ее часть) из протокола.

Механизм псевдонимов (alias) позволяет связать с именем полностью (или частично) определенную командную строку и в дальнейшем пользоваться этим именем.

Кроме того, в C-shell по сравнению с Bourne-shell существенно расширен набор предопределенных переменных, а также введены более развитые возможности вычислений (по-прежнему, все значения представляются в текстовой форме).

Korn-shell

Если C-shell является синтаксической вариацией командного языка семейства shell по направлению к языку программирования Си, то Korn-shell - это непосредственный последователь Bourne-shell.

Если не углубляться в синтаксические различия, то Korn-shell обеспечивает те же возможности, что и C-shell, включая использование протокола и псевдонимов.

Реально, если не стремиться использовать командный язык как язык программирования (это возможно, но по мнению автора, неоправданно), то можно пользоваться любым вариантом командного языка, не ощущая их различий.