ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ имени академика С.П. КОРОЛЕВА»

Изучение операционной системы Linux: интерфейс и основные команды

Утверждено Редакционно-издательским советом университета
в качестве методических указаний к лабораторной работе № 8

С А М А Р А
Издательство СГАУ
2010

УДК

Составители А.М. С у х о в, Г.М. Г а й н у л л и н а

Рецензент: к.т. н., доц. Попов С.Б.

Изучение операционной системы Linux: интерфейс и основные команды: Методические указания к лабораторной работе/ Сост. А.М. Сухов, Г.М. Гайнуллина .-Самара: Изд-во Самарского государственного аэрокосмического университета, 2010. 22 с.

В настоящих методических указаниях приведен материал, необходимый для выполнения лабораторных работ по дисциплине «Информатика».

Целью лабораторных работ является изучение основных возможностей и приобретение навыков работы в операционной системе Linux.

Предназначено для студентов 010900, 140500, 160900, 150100, 200100, 080100 специальностей аэрокосмического профиля.

1. Цель лабораторной работы

- освоить основные принципы работы в операционной системе Linux:
- изучить интерфейс и основные команды
- получить практические навыки работы в операционной системе Linux

2. Основные сведения

Операцио́нная систе́ма, ОС (англ. operating system) — базовый комплекс компьютерных программ, обеспечивающий управление аппаратными средствами компьютера, интерфейс с пользователем, работу с файлами, ввод и вывод данных, а также выполнение прикладных программ и утилит.

ОС позволяет абстрагироваться от деталей реализации аппаратного обеспечения, предоставляя разработчикам программного обеспечения минимально необходимый набор функций. С точки зрения обычных пользователей компьютерной техники ОС включает в себя и программы пользовательского интерфейса.

Четыре составные части операционной системы: ядро, интерфейс пользователя, файловая система и прикладные программы и утилиты.

Ядро́ — центральная часть операционной системы, обеспечивающая приложениям координированный доступ к ресурсам компьютера, таким как процессорное время, память и внешнее аппаратное обеспечение. Также обычно ядро предоставляет сервисы файловой системы и сетевых протоколов.

Интерфе́йс по́льзователя, (UI — англ. user interface) — разновидность интерфейсов, в котором одна сторона представлена человеком (пользователем), другая — машиной/устройством. Представляет собой совокупность средств и методов, при помощи которых пользователь взаимодействует с различными машинами и устройствами.

Чаще всего термин применяется по отношению к компьютерным программам (приложениям). Но вообще под пользовательским интерфейсом подразумевается любая система взаимодействия с устройствами, способными к интерактивному взаимодействию с пользователем: меню на экране телевизора плюс пульт дистанционного управления им же, дисплей электронного аппарата (автомагнитола, часы, проигрыватель) и набор кнопок и переключателей для его настройки и управления, и так далее.

Фа́йловая систе́ма (англ. file system) — регламент, определяющий способ организации, хранения и именования данных на носителях информации. Она определяет формат физического хранения информации, которую принято группировать в виде файлов. Конкретная файловая система определяет размер имени файла, максимальный возможный размер файла, набор атрибутов файла. Некоторые файловые системы предоставляют сервисные возможности, например, разграничение доступа или шифрование файлов.

С точки зрения операционной системы, весь диск представляет из себя набор кластеров размером от 512 байт и выше. Драйверы файловой системы организуют кластеры в файлы и каталоги (реально являющиеся файлами, содержащими список файлов в этом каталоге). Эти же драйверы отслеживают, какие из кластеров в настоящее время используются, какие свободны, какие помечены как неисправные.

Прикладная программа или приложение — программа, предназначенная для выполнения определенных пользовательских задач, рассчитана на непосредственное взаимодействие с пользователем. В большинстве операционных систем прикладные программы не могут обращаться к ресурсам компьютера напрямую, а взаимодействуют с оборудованием и проч. посредством операционной системы.

Утили́та (англ. utility или tool) — компьютерная программа, расширяющая стандартные возможности оборудования и операционных систем, выполняющая узкий круг специфических задач.

Утилиты предоставляют доступ к возможностям (параметрам, настройкам, установкам), недоступным без их применения, либо делают процесс изменения некоторых параметров проще (автоматизируют его).

Начало работы с Linux

Как известно, работать в Linux можно в графической системе X Window или в текстовой консоли. Большинство пользователей после инсталляции предпочитают работать исключительно с оконным менеджером, но есть широкий ряд задач, выполнить которые можно (или значительно проще), работая в консоли.

Для начала немного о настройке консольного входа.

Добавление нового пользователя. Утилита ***adduser*** (начало задания выполняется преподавателем под пользователем с административными правами). Переход в режим суперпользователя осуществляется командой ***su.*** Студент должен придумать имя пользователя и пароль.

#adduser имя_пользователя

Далее за консолью работает студент и заполняет все поля самостоятельно. По окончанию необходимо выйти из консоли командой ***exit.*** Далее студент должен выполнить вход в систему самостоятельно, как в графическом, так и консольном интерфейсах.

Если при инсталляции Linux настроен автоматический запуск X-ов, то необходимо сначала перейти в консоль. Для этого нажмите Ctrl+Alt+F3. Вы попадете в виртуальную текстовую консоль и после ввода имени пользователя и пароля сможете давать команды shell. Для возвращения в X Window нажмите Alt+F7. Вообще говоря, по умолчанию можно работать сразу в 6-ти виртуальных консолях, что часто бывает очень удобно (переключение между ними - Alt+F1...Alt+F6).

Опытные пользователи советуют работать root-ом как можно меньше, поскольку его ошибка может вызвать самые фатальные для системы последствия, тогда как обычный пользователь может повредить обычно лишь свои собственные файлы.

Работать в консоли довольно удобно, но для перемещения по каталогам гораздо приятнее использовать Midnight Commander. После вызова команды ***mc*** на экране появляется Norton-подобный файловый менеджер, который по мощности почти ничем не уступает DN или FAR.

Работа с командной строкой

Эффективная профессиональная работа в Linux немыслима без использования командной строки. Пользователям, привыкшим работать в системах с графическим интерфейсом, работа с командной строкой может показаться неудобной: то, что можно сделать одним перетаскиванием мышью в командной строке потребует ввода с клавиатуры нескольких слов: команды с аргументами. Однако в Linux этот вид интерфейса всегда был основным, а поэтому и хорошо развитым. В командных оболочках, используемых в Linux, есть масса способов экономии усилий (нажатий на клавиши) при выполнении наиболее распространённых действий: автоматическое дополнение длинных названий команд или имён файлов, поиск и повторное выполнение команды, уже когда-то исполнявшейся раньше, подстановка списков имён файлов по некоторому шаблону и многое другое. Преимущества командной строки становятся особенно очевидны, когда требуется выполнять однотипные операции над множеством объектов. В системе с графическим интерфейсом потребуется столько перетаскиваний мышью, сколько есть объектов, в командной строке будет достаточно одной (пусть длинной и сложной) команды.

В этом разделе будут описаны основные инструменты, позволяющие при помощи командной строки решать любые задачи пользователя: от тривиальных операций с файлами и каталогами (копирование, переименование, поиск) до сложных задач, требующих массовых однотипных операций, которые возникают как в прикладной работе пользователя, при работе с большими массивами данных или текста, так и в системном администрировании.

Командные оболочки (shells)

Общая информация об оболочках

Командная оболочка (или интерпретатор команд) — это программа, задача которой состоит в том, чтобы передавать ваши команды операционной системе и прикладным программам, а их ответы — вам. По своим задачам ему соответствует command.com в MS-DOS илиcmd.exe в Windows, но функционально оболочки в Linux несравненно богаче. На языке командной оболочки можно писать небольшие программы для выполнения ряда последовательных операций с файлами и содержащимися в них данными — сценарии (скрипты).

Зарегистрировавшись в системе (введя имя пользователя и пароль), вы увидите приглашение командной строки — строку, оканчивающуюся символом **$** (далее этот символ будет использоваться для обозначения командной строки). В случае, если при установке был настроен запуск графического интерфейса при загрузке системы, то добраться до командной строки можно на любой виртуальной текстовой консоли (нажав Ctrl-Alt-F1 — Ctrl-Alt-F6) или при помощи любой программы эмуляции терминала, например xterm. Обычно доступны следующие командные оболочки:

bash

Самая распространённая оболочка под Linux. Она умеет дополнять имена команд и файлов, ведёт историю команд и предоставляет возможность их редактирования.

pdkdh, sash, tcsh, zsh

Оболочкой по умолчанию является ***bash*** (Bourne Again Shell). Чтобы проверить, какую оболочку вы используете, наберите команду: echo $SHELL.

Оболочки отличаются друг от друга не только возможностями, но и синтаксисом команд. Для начинающих пользователей рекомендуется использовать ***bash***, дальнейшие примеры описывают работу именно в этой облочке.

Командная оболочка bash

Командная строка в bash составляется из имени команды, за которым могут следовать ключи (опции) — указания, модифицирующие поведение команды. Ключи начинаются с символа - или --, и зачастую состоят из одной буквы. Кроме ключей, после команды могут следовать аргументы (параметры) — названия объектов, над которыми должна быть выполнена команда (часто — имена файлов и каталогов).

$ команда опции аргументы

Ввод команды завершается нажатием клавиши Enter, после чего команда передаётся оболочке на исполнение. В результате выполнения команды на терминале пользователя могут появиться сообщения о ходе выполнения команды или об ошибках, а появление очередного приглашения командной строки (оканчивающегося символом $) — знак того, что выполнение команды завершено и можно вводить следующую.

В bash имеется несколько приёмов, облегчающих ввод и редактирование командной строки. Например, используя клавиатуру, вы можете:

Ctrl-A

перейти на начало строки, это же можно сделать, нажав клавишу Home;

Ctrl-U

удалить текущую строку;

Ctrl-C

Прервать выполнение текущей команды.

Вы можете использовать символ ; для того, чтобы ввести несколько команд в одну строку. bash записывает историю всех выполненных команд, поэтому несложно повторить или отредактировать одну из предыдущих команд. Для этого достаточно выбрать нужную команду из истории: клавиша вверх выводит предыдущую команду, вниз — последующую. Для того, чтобы найти конкретную команду среди уже выполненных, не пролистывая всю историю, наберите *Ctrl-R* и введите какое-нибудь ключевое слово, употребленное в той команде, которую вы ищете.

Команды, присутствующие в истории, отображаются в списке пронумерованными. Для того, чтобы запустить конкретную команду, наберите:

!номер команды

Если ввести *!!*, запустится последняя из набранных команд.

Иногда в Linux имена программ и команд слишком длинны. К счастью, *bash* сам может завершать имена. Нажав клавишу *Tab*, вы можете завершить имя команды, программы или каталога. Например, предположим, что вы хотите использовать программу декомпрессии bunzip2. Для этого наберите:

$bu

Затем нажмите *Tab*. Если ничего не происходит — значит, существует несколько возможных вариантов завершения команды. Нажав клавишу *Tab* ещё раз, вы получите список имён, начинающихся с *bu*.

Например, в системе есть программы *buildhash, builtin, bunzip2*:

$ bu

buildhash builtin bunzip2

$ bu

Наберите *n* > (bunzip — это единственное имя, третьей буквой которого является n), а затем нажмите *Tab*. Оболочка дополнит имя и остаётся лишь нажать *Enter*, чтобы запустить команду!

Заметим, что программу, вызываемую из командной строки, *bash* ищет в каталогах, определяемых в системной переменной *PATH*. По умолчанию в этот перечень каталогов не входит текущий каталог, обозначаемый *./* (точка слэш). Поэтому для запуска программы prog из текущего каталога надо дать команду *./prog*.

Базовые команды

Первые задачи, которые приходится решать в любой системе: работа с данными (обычно хранящимися в файлах) и управление работающими в системе программами (процессами). Ниже перечислены команды, позволяющие выполнять наиболее важные операции по работе с файлами и процессами. Только первая из них — *cd* — является составляющей частью собственно командной оболочки, остальные распространяются отдельно, но всегда доступны в любой системе Linux. Все команды, приведённые ниже, могут быть запущены как в текстовой консоли, так и в графическом режиме (xterm, консоль KDE). Для получения более подробной информации по каждой из команд используйте команду *man*, например:

$man ls

Команда *man* (от manual) запускает чтение инструкции в редакторе *vi*. Для того, чтобы выйти из редактора необходимо набрать в командной строке *:q*

$cd

Позволяет сменить текущий каталог (перемещаться по файловой системе). Она работает как с абсолютными, так и с относительными путями. Предположим, что вы находитесь в своём домашнем каталоге и хотите перейти в его подкаталог tmp/. Для этого, введите относительный путь:

$cd tmp/

Чтобы перейти в каталог */usr/bin*, наберите (абсолютный путь):

$cd /usr/bin/

Некоторые варианты использования команды:

$cd ..

позволяет вам сделать текущим родительский каталог (обратите внимание на пробел между cd и ..).

$cd -

позволяет вам вернуться в предыдущий каталог. Команда *cd* без параметров возвращает оболочку в домашний каталог.

$ls

ls (list) выдаёт список файлов в текущем каталоге. Две основные опции: *-a* — просмотр всех файлов, включая скрытые, *-l* — отображение более подробной информации.

less позволяет вам постранично просматривать текст. Синтаксис:

$less имя_файла

Бывает полезно просмотреть файл перед тем, как его редактировать; основное же применение данной команды — конечное звено цепочки программ, выводящей существенное количество текста, которое не умещается на одном экране и в противном случае слишком быстро промелькнёт. Для выхода из *less* нажмите *q* (quit).

$grep

Данная команда позволяет найти строку символов в файле. Обратите внимание, что *grep* осуществляет поиск по регулярному выражению, то есть предоставляет возможность задавать шаблон для поиска сразу целого класса слов. На языке регулярных выражений можно составлять шаблоны, описывающие, например, такие классы строк: «четыре цифры подряд, окружённые пробелами». Очевидно, такое выражение можно использовать

для поиска в тексте всех годов, записанных цифрами. Возможности поиска по регулярному выражению очень широки, за более подробными сведениями вы можете обратиться к экранной документации по *grep* (man grep). Синтаксис:

$grep шаблон_поискафайл

$ps

Отображает список текущих процессов. Колонка команд указывает имя процесса, PID (идентификатор процесса) — номер процесса (используется для операций с процессом — например, отправки сигналов командой *kill*). Синтаксис:

$ps аргументы

Аргумент *u* предоставляет вам больше информации, *ax* позволяет вам просмотреть те процессы, которые не принадлежат вам.

$kill

Если программа перестала отвечать или зависла, используйте данную команду, чтобы её завершить. Синтаксис:

$kill PID_номер

PID_номер здесь — идентификационный номер процесса, вы можете узнать номер процесса для каждой выполняемой программы при помощи команды *ps*. Обычно команда *kill* отправляет процессу сигнал нормального завершения, однако иногда это не срабатывает и необходимо будет использовать *kill -9 PID_number* — в этом случае команда будет немедленно завершена системой без возможности сохранения данных (аварийное завершение). Список сигналов, которые команда kill может отправлять процессу можно получить, отдав команду *kill -l*.

Текстовые редакторы

В Linux часто возникает необходимость в ручном редактировании конфигурационных файлов, *nano* это единственный редактор, доступный даже на стадии инсталляции системы. Да, это не *emacs*, и даже не *joe*. Но с задачей конфигурирования справляется успешно.

Nano — немодальный редактор, и для вставки текста можно сразу начинать набор. Если вы редактируете конфигурационный файл, такой как */etc/fstab*, указывайте параметр -w, например:

 # nano -w /etc/fstab

Чтобы сохранить сделанные изменения, нажмите *Ctrl+O*. Для выхода из *nano* нажмите *Ctrl+X*. Если вы выходите из редактора, а файл изменен, *nano* предложит сохранить файл. Чтобы отказаться от сохранения, просто нажмите *N*, а для подтверждения — *Y*. Редактор запросит имя файла. Просто введите имя, а затем нажмите *Enter*.

Если вы случайно подтвердили необходимость сохранения файла, который сохранять не нужно, от сохранения всегда можно отказаться нажатием *Ctrl+C* в момент запроса имени файла.

При запуске терминала nano снизу терминала появляется подсказка, представляющая набор основных управляющих клавиш, доступных в сочитании с *Ctrl* +. Клавиша (в данном случае символ ^**)** заменяет клавишу *Ctrl*. Мета клавиша используется и в других сочетаниях, например **"*M*"** это означает клавишу ***Alt*.**

Midnight Commander

Если вы многие годы работали в MS-DOS/Windows, то, наверное, ощущаете себя немного «не в своей тарелке». Для того, чтобы попасть в привычную среду, запустите Midnight Commander командой mc. Midnight Commander — это свободный аналог Norton Commander и его популярного ныне потомка — Far. Если вы в какой-то момент сочтёте, что Midnight Commander что-то не умеет, то это, скорее всего, неверно. Ознакомьтесь с его описанием в */usr/share/doc/mc*-номер_версии или дайте команду *man mc*.

Графический интерфейс

Оконная система X и XFree86

Оконная система икс (от англ. X window system, далее — X)— один из самых больших и успешных проектов в истории компьютерной техники — восходит к 1984 г., когда разработчики двух систем компьютерной графики, претендующих на универсальность — проектов Athena (Массачусетский технологический институт) и W Windowing (Стэнфордский университет) — решили объединить свои усилия. С тех пор практически каждая компания, серьезно занимающаяся графикой, считала своим долгом внести какие-либо разработки в систему, формальным «хозяином» которой в 1987 г. стал вновь созданный X Consortium (ныне Open Group, www.X.org).

Дальнейшее изложение ориентировано на свободную реализацию X, которая называется XFree86, поддерживается одноименным партнерством (www.xfree86.org. XFree86 — самая популярная реализация X, она поставляется в составе подавляющего большинства открытых систем (как

свободных, так и несвободных) для x86-совместимых компьютеров, поддерживает беспрецедентно широкий спектр оборудования и, благодаря доступности исходных текстов и пользовательской аудитории в десятки миллионов человек, весьма устойчива и хорошо оттестирована, по крайней мере, насколько это возможно для такого разнообразия поддерживаемого оборудования. Несмотря на то, что исторически цифры «86» в названии пакета относятся к соответствующему семейству процессоров от Intel, современные версии XFree86 реализованы для большинства других популярных процессоров. XFree86 доступен и для некоторых альтернативных архитектур ОС, включая **Microsoft.**

Большинство из того, о чем будет говориться в последующих разделах, справедливо для любой реализации X на любом оборудовании и под любой ОС, список которых можно найти на www.X.org.

Цветной бутерброд

То, что пользователю, сидящему за монитором, представляется сплошной графической операционной средой, реализовано как многослойный бутерброд технологий.

Непосредственно с оборудованием (видеосистемой, устройствами ввода и динамиком) работает *X-сервер*. Эта программа захватывает оборудование и предоставляет его возможности другим программам как ресурсы (собственно, именно поэтому она и называется сервером) по особому протоколу, который так и называется, X-протокол.

Ключевой компонент графической платформы — X-сервер:

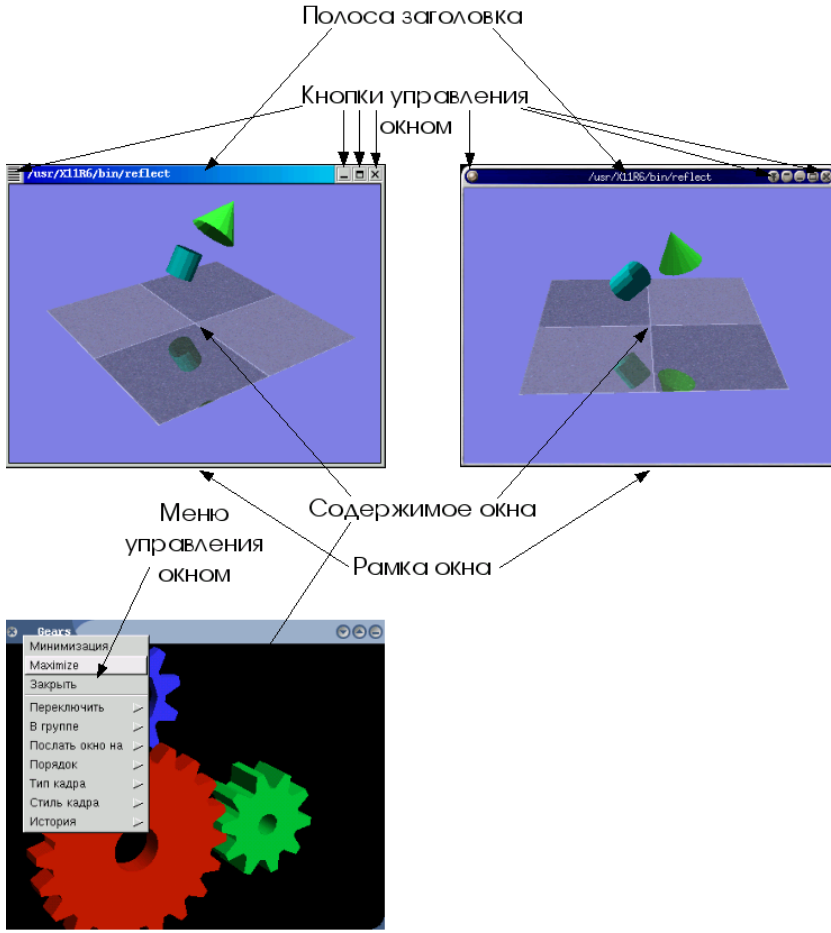
- захватывает оборудование;
- создает по запросу других программ (которые в этой терминологии называются *X-клиентами*) окна;
- предоставляет другим программам возможность работы в окнах, т.е. вывода информации в эти окна и обработки сигналов от устройств ввода (клавиатуры и мыши или другого координатного устройства), когда окно, назначенное программе, является активным. Предоставление ресурсов возможно в том числе и через сеть, когда клиент и сервер работают на разных компьютерах (узлах).

При универсальном применении компьютера характерна поочередная работа с различными программами (иногда достаточно большим их количеством), причем пользователь может отрываться, допустим, от редактирования текста, чтобы поработать с иллюстрацией при помощи другой программы, прочитать почту или заглянуть на интернет-страницу, затем возвращаться к редактированию текста и т. д. Эти возможности обеспечивает другая

программа — менеджер окон, представляющая собой следующий «слой» в графической среде пользователя.

Менеджеры окон

Рисунок 1.. Виджеты



Для одновременной и поочерёдной работы с разными программами, требуется возможность *управлять окнами* (с помощью клавиатуры или мыши), т. е. возможность изменять «на лету» их геометрию (положение и

размеры), а также (обычно не относимое к геометрии) положение в воображаемой «стопке» окон — от этого зависит, какое из окон будет «верхним» (видимым полностью), если окна перекрывают друг друга на плоскости экрана.

Управление окнами и составляет основную функцию *оконного менеджера* (устоявшийся англоязычный термин window manager).

Менеджеров окон существует превеликое множество — под любой набор задач, которые может решать графическая многооконная система. Их настолько много, что выбрать какой-нибудь в качестве «типичного представителя семейства» затруднительно.

Базовая (а также расширенная) функциональность оконных менеджеров доступна пользователю прежде всего за счет введения в интерфейс так называемых *виджетов* (от англ. widgets, сокращение от window gadgets, «оконные приспособления»). Виджеты — это рамки, кнопки, меню и пр., которые служат «органами управления» окна. Технически (в терминах оконной системы X) виджеты представляют собой отдельные окна, примыкающие к окну прикладной программы и, как правило, перемещающиеся вместе с ним.

Обрамление окна обычно составляют следующие элементы:

Рамка, окружающая окно

При «буксировке» рамки мышью окно изменяет свой размер. Иногда для изменения размера окна предназначены только выделенные «уголки» рамки, представляющие собой отдельные виджеты.

Полоса заголовка

Часто совпадает с одной из (обычно, верхней) сторон рамки. В полосе заголовка может содержаться название программы или запустившая программу команда, а также другая информация, специфичная для окна. При «буксировке» полосы заголовка перемещается все окно. Со «щелчками» различными кнопками мыши на полосе заголовка также могут быть связаны различные действия по управлению окнами.

Кнопки управления окном

Часто вынесенные на полосу заголовка или в другое место рамки кнопки позволяют выполнить с ним такие действия, как закрытие (часто сопровождающееся выходом из программы, открывшей окно), максимизация (разворачивание окна на весь экран), минимизация/сворачивание, вызов меню

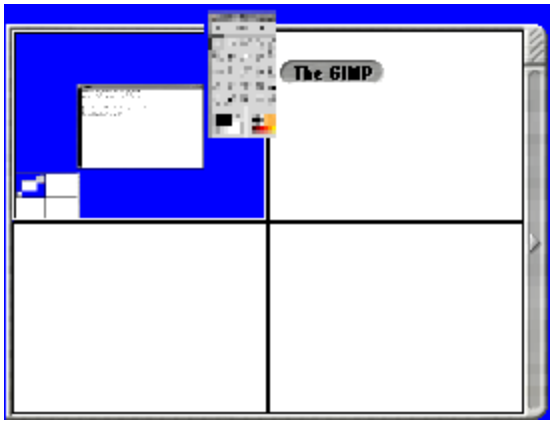
управления окном, которое может содержать весьма обширный репертуар других действий.

Детали реализации обрамления окна могут быть весьма различными в зависимости от конкретного оконного менеджера и его настроек.

Управление окнами — основная функция оконного менеджера, и на этом его функциональность может и заканчиваться. Однако большинство из них выполняют, по крайней мере, еще одну функцию.

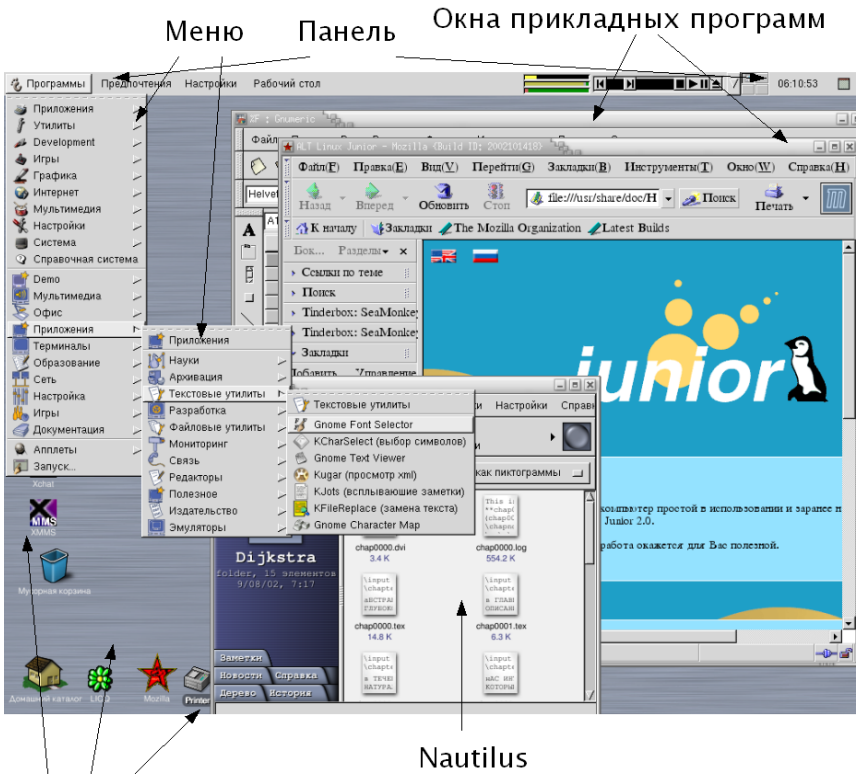
При запуске оконного менеджера на экране появляется еще одно окно. Это так называемый *пейджер* (pager), на Рисунке 2 он изображен крупным планом. На пейджере представлена миниатюрная копия экрана, обновляющаяся в режиме реального времени, причем, если подвести курсор к изображению отдельного окна, оно увеличивается и рядом высвечивается название приложения, запущенного в нем. Но почему экран занимает только четверть окна пейджера? Потому что оконный менеджер позволяет оперировать «виртуальным» столом (от англ. virtual desktop, также *рабочим столом*), по размеру превышающим физический экран, а пейджер — одно из средств перемещения физического экрана по рабочему столу.

Рисунок 2. Пейджер



Интегрированные графические среды

Рисунок 3.



Существует два подхода к тому, как можно достроить оконную систему до полнофункциональной среды, позволяющей пользователю решать все (или почти все) его практические задачи. Во-первых, можно расширить функциональность менеджера окон, добавив в него недостающие возможности. В оконном менеджере до полнофункциональной среды не хватает возможности запускать программы и утилиты. Достигается это обычно при помощи организации специального меню. Во-вторых, можно добавить в «графический бутерброд» еще один слой — *менеджер рабочего*

стола — работающий «поверх» менеджера окон и использующий функциональность последнего. Этим путем идут команды разработчиков ***GNOME*** и ***KDE***.

С точки зрения пользователя нет четкой границы между менеджерами окон с расширенной функциональностью и менеджерами рабочего стола, работающими «поверх» менеджера окон, поскольку они обеспечивают одну и ту же функциональность и нередко даже графически организованы сходным образом. Оба варианта предоставляют пользователю возможность работать в *графической среде* (desktop environment).

Интегрированная графическая среда предполагает не только единство оформления, но и трактовку объектов в рабочем пространстве (окон, файлов, пунктов меню и т. п.) как физических объектов, которые можно перемещать, выбрасывать в «корзину» и т. д.

На сегодня существуют и развиваются две свободные интегрированные графические среды общего назначения: KDE и GNOME. Они входят в поставку большинства стандартных (открытых) ОС, как свободных, так и несвободных. GNOME

GNOME (GNOME, GNU Network Object Model Environment — «Среда ГНУ, основанная на модели сетевых объектов», но также и «Образцовая среда для сетевых объектов ГНУ») — один из самых амбициозных и масштабных проектов в программистском сообществе.

С пользовательской точки зрения ***GNOME*** предстает как набор базовых компонентов интерфейса и *аплетов*, утилит и прикладных программ. К базовым компонентам относятся менеджер файлов и поверхности стола *Наутилус (Nautilus)*, панели управления и меню ***GNOME Panel*** и центр управления (*Gnome Control Center*).

Менеджер файлов ***Nautilus*** позволяет отображать содержимое файлов и каталогов в окнах и выполнять над файлами обычные действия (удаление, переименование, копирование и перемещение и т. п.), а также осуществлять предварительный просмотр многих типов данных. ***Nautilus*** эффектен, но работа с ним не более эффективна, чем с прочими браузерами файлов, включаемыми обычно в графические среды (менеджер файлов *CDE* или *Microsoft Windows Explorer*).

Панели, наряду с менеджером файлов, являются важнейшей составной частью интерфейса ***GNOME***. Панелей может быть неограниченное количество. Панель может быть двух типов: панель-меню (*menu panel*) и объектная панель (*object panel*). Первая из них содержит пункты меню и может содержать пиктограммы, а вторая — только пиктограммы.

На панелях могут присутствовать объекты пяти типов:

- *Аплет* (applet, «приложеньице») — интересный тип панельного объекта, демонстрирующий то, что он не обязан быть представлен статической картинкой. Это программа, места в панели которой достаточно, чтобы отображать какую-нибудь полезную (или забавную) информацию или даже принимать клавиатурный и/или координатный ввод. Важными аплетами являются путеводитель по столу (*Desktop Guide*) и список задач (*Task List*), позволяющие переключаться между виртуальными экранами и активизировать окна запущенных программ, соответственно.
- *Пускатель* (launcher) ассоциирован с приложением или командой, которые исполняются по щелчку на его пиктограмме в панели.
- *Выдвижной ящик* (drawer) — это кнопка, открывающая другую панель, перпендикулярно первой — некий аналог подменю в меню, который можно наполнить всевозможными аплетами.
- *Специальные объекты* — это те же аплеты, но выполняющие функции, которые другими средствами «достать» почему-либо нельзя (запереть экран, выйти из ***GNOME*** или запустить программу «вручную»).
- Наконец, *объект-меню* раскрывает меню.

У ***GNOME*** нет единой иерархии меню: кроме главного, вызываемого объектом-меню с гномьей лапой (оно же, когда вызывается щелчком правой кнопки на фоне или нажатием клавиши, почему-то называется *глобальным* (global)), пользователь может создавать *обычные* (normal) меню, связанные с объектами-меню на панелях.

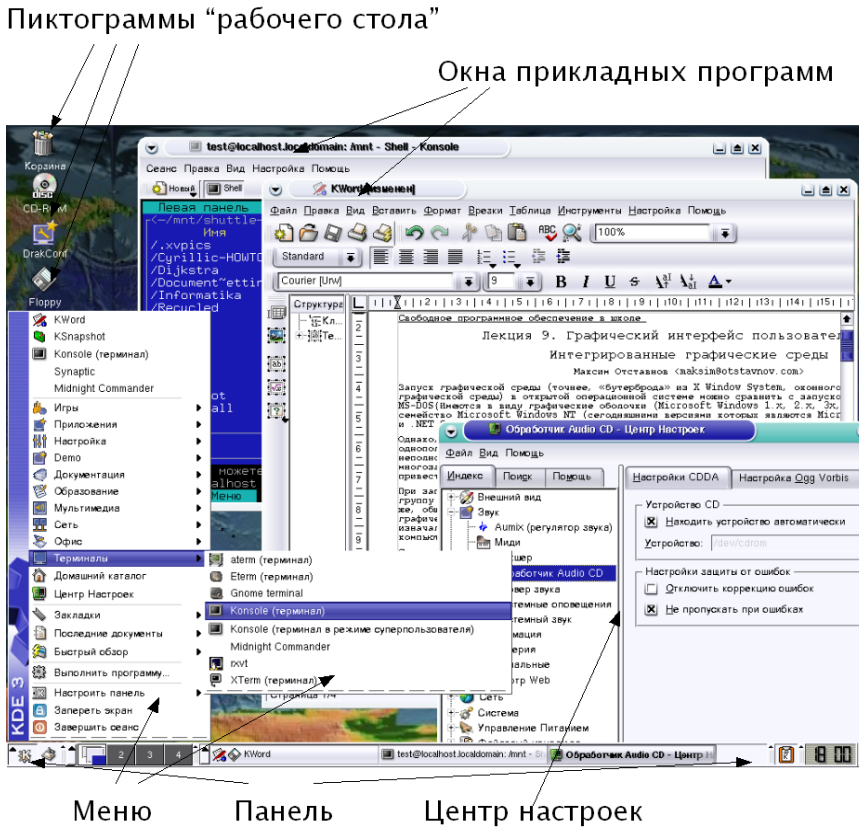
Меню настраиваются примерно так же, как и панели: пользователь может добавлять, менять и удалять пункты, создавать подменю и т. п. При этом создаваемые обычные меню изначально пусты, а главное/глобальное заполняется при установке всем, что GNOME найдет в системе, и пользователю остается только убрать лишнее и переставить пункты в соответствии со своими предпочтениями.

Также постоянно расширяется набор утилит, прикладных программ и аплетов, поставляемых с ***GNOME*** — вместе с программами, входящими в большинство дистрибутивов ОС, о которых ***GNOME*** «в курсе», их число превышает сотню. Перечислить их здесь нет никакой возможности, но среди них есть интерфейсы для администрирования системы, средства звукозаписи и воспроизведения, сетевые утилиты, игры и многое другое.

KDE

Само название ***KDE*** (KDE, K Desktop Environment — «Графическая среда К») — явная пародия на CDE (Common Desktop Environment — «Общая настольная среда»). CDE была последней попыткой отрасли стандартизовать графическую среду на несвободной основе, предпринятой в конце девяностых годов. Буква «К» в KDE ничего не означает.

Рисунок 4. Интегрированная графическая среда KDE



Несмотря на явно игривый тон, начинающийся с названия среды и продолжающийся в названии компонентов, ***KDE*** — очень серьезный проект.

В ***KDE*** любят играть со словами; например, универсальный броузер, входящий в среду, называется *Konqueror* (от англ. conqueror— «завоеватель», «покоритель»), терминал — *Konsole* (от console — «консоль»), а система помощи — вообще *Kandalf* (от имени Гэндальфа, мага из фантазийных произведений Дж. Р. Р. Толкиена).

Если единообразие и однородность графической среды считать достоинством, то ***KDE*** — несомненный лидер среди всех (как свободных, так и несвободных) интегрированных графических сред. Основное видимое средство интеграции — это универсальный броузер *Konqueror*. Функция *Konqueror* близка к той, которую приобрел Microsoft Windows Explorer в **Microsoft Windows** — он совмещает функции гипермедийного броузера WWW и браузера локальных ресурсов.

Разработчики *KDE* пошли даже дальше своих коллег из Microsoft и определили ряд дополнительных протоколов, что позволило, в частности, просматривать с помощью броузера в единообразном формате все разнообразие справочной информации, представленное в сегодняшних открытых система (традиционные страницы руководства ***man***, гипертекстовую систему ***Info*** из проекта ГНУ, разрозненные файлы документации в текстовом и гипертекстовом формате). В *Konqueror* интегрирована также возможность предварительного просмотра содержимого большого количества типов файлов.

1. ***KDE*** включает также настраиваемую систему панелей и меню и интегрированный *центр управления*, позволяющий согласованно изменять параметры среды. ***KDE*** менее гибка в настройке, чем ***GNOME***, однако ее гибкости вполне достаточно для решения любых практических задач (в том числе, имитации вида и поведения других сред). ***KDE*** работает только с собственным оконным менеджером *KWin*.

В поставку ***KDE*** входит множество «аксессуаров» и прикладных программ, к тому же рядом с проектом выросла целая группа сопутствующих, ориентированных на те или иные предметные приложения, из которых самым развитым является офисный пакет *KOffice*.

Список контрольных вопросов

1. Дайте определение операционной системы и ее составных частей
2. Какие типы командных оболочек Вы знаете?
3. Приведите формат исполнения команды в командной строке
4. Какие базовые команды shell Вы знаете?
5. Расскажите о функциях X сервера
6. Что такое менеджеры окон и виджеты

7. Расскажите о принципиальных особенностях интегрированных графических сред ***KDE, GNOME***

Задания для выполнения

1. Придумайте для себя имя пользователя и пароль, выполните вход в систему из терминальной консоли и графического интерфейса
2. Запустите команду показывающую список файлов текущего каталога
3. Получите справку по набору действий, которую можно осуществить с помощью данной команды
4. Выйдете в корневую директорию (root)
5. Посмотрите список процессов, выполняемых в данное время на Вашем компьютере
6. Завершите один из процессов
7. Создайте в домашнем каталоге файл и отредактируйте его
8. Проведите первоначальную настройку
9. Проведите первоначальную настройку ***GNOME***
10. Проведите первоначальную настройку ***KDE***

Список литературы

1. Костромин В. А. “Linux для пользователя”. БХВ - Петербург, 2002, 672 стр., ISBN: 5-94157-183-6

2. Петерсен Р., Руководство по операционной системе Linux, BHV, Москва, 2005

3. М. Уэлш. Инсталяция Linux и первые шаги. -М. МГУ, 1999

Учебное издание

Изучение операционной системы Linux: интерфейс и основные команды

Методические указания

Составители Сухов Андрей Михайлович

Гайнуллина Гелия Мухаматкамиловна

Изд-во Самарского государственного

аэрокосмического университета.

443086 Самара, Московское шоссе, 34.