FileSystems\_2006

**Введение в файловые системы**

Определение ФС

*Файловая система* – это часть ОС, назначение которой состоит в том, чтобы обеспечить пользователю удобный интерфейс при работе с данными, хранящимися на диске, и обеспечить совместное использование файлов несколькими пользователями и процессами.

Понятие ФС

В широком смысле понятие “ФС” включает:

* совокупность всех файлов на диске,
* наборы структур данных, используемых для управления файлами, такие, например, как каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске,
* комплекс системных программных средств, реализующих управление файлами, в частности: создание, уничтожение, чтение, запись, именование, поиск и другие операции над файлами.

Файлы

* ***Файл*** - это именованный набор связанной информации, записанной во вторичную память.
* *Файлы* представляют собой абстрактные объекты. Их задача - хранить информацию, скрывая от пользователя детали работы с устройствами.

Имена файлов

* Файлы идентифицируются символьными именами, которые им дают пользователи.
* Общий формат символьного имени файла:

ИМЯ.РАСШИРЕНИЕ

* Символьные имена могут быть различной длины в зависимости от типа ФС.
* Поддержка национальных алфавитов – символьные имена в формате UNICODE.
* Присваивание нескольких символьных имен одному файлу.
* Возможное различие строчных и прописных букв.

Файлы идентифицируются именами. Пользователи дают файлам символьные имена, при этом учитываются ограничения ОС как на используемые символы, так и на длину имени. До недавнего времени эти границы были весьма узкими. Так в популярной файловой системе FAT длина имен ограничивается известной схемой (8 символов - собственно имя, 3 символа - расширение имени), а в ОС UNIX System V имя не может содержать более 14 символов. Однако пользователю гораздо удобнее работать с длинными именами, поскольку они позволяют дать файлу действительно мнемоническое название, по которому даже через достаточно большой промежуток времени можно будет вспомнить, что содержит этот файл. Поэтому современные файловые системы, как правило, поддерживают длинные символьные имена файлов. Например, Windows NT в своей новой файловой системе NTFS устанавливает, что имя файла может содержать до 255 символов, не считая завершающего нулевого символа.

При переходе к длинным именам возникает проблема совместимости с ранее созданными приложениями, использующими короткие имена. Чтобы приложения могли обращаться к файлам в соответствии с принятыми ранее соглашениями, файловая система должна уметь предоставлять эквивалентные короткие имена (псевдонимы) файлам, имеющим длинные имена. Таким образом, одной из важных задач становится проблема генерации соответствующих коротких имен.

Длинные имена поддерживаются не только новыми файловыми системами, но и новыми версиями хорошо известных файловых систем. Например, в ОС Windows 95 используется файловая система VFAT, представляющая собой существенно измененный вариант FAT. Среди многих других усовершенствований одним из главных достоинств VFAT является поддержка длинных имен. Кроме проблемы генерации эквивалентных коротких имен, при реализации нового варианта FAT важной задачей была задача хранения длинных имен при условии, что принципиально метод хранения и структура данных на диске не должны были измениться.

Обычно разные файлы могут иметь одинаковые символьные имена. В этом случае файл однозначно идентифицируется так называемым составным именем, представляющем собой последовательность символьных имен каталогов. В некоторых системах одному и тому же файлу не может быть дано несколько разных имен, а в других такое ограничение отсутствует. В последнем случае операционная система присваивает файлу дополнительно уникальное имя, так, чтобы можно было установить взаимно-однозначное соответствие между файлом и его уникальным именем. Уникальное имя представляет собой числовой идентификатор и используется программами операционной системы. Примером такого уникального имени файла является номер индексного дескриптора в системе UNIX.

Типы файлов

* обычные файлы:
* специальные файлы;
* файлы-каталоги.

Обычные файлы

* *Обычные файлы* в свою очередь подразделяются на текстовые и двоичные.
  + Текстовые файлы состоят из строк символов, представленных в ASCII-коде. Это могут быть документы, исходные тексты программ и т.п. Текстовые файлы можно прочитать на экране и распечатать на принтере.
  + Двоичные файлы не используют ASCII-коды, они часто имеют сложную внутреннюю структуру. Например, исполняемый *файл* в ОС Unix имеет пять секций: заголовок, текст, данные, биты реаллокации и символьную таблицу. ОС выполняет *файл*, только если он имеет нужный формат. Другим примером бинарного *файла* может быть архивный *файл*
  + Все операционные системы должны уметь распознавать хотя бы один тип файлов - их собственные исполняемые файлы.

Специальные файлы

* *Специальные файлы* – это файлы, ассоциированные с устройствами ввода-вывода, которые позволяют пользователю выполнять операции ввода-вывода, используя обычные команды записи в файл или чтения из файла.
* Команды чтения/записи обрабатываются вначале программами файловой системы, а затем на некотором этапе выполнения запроса преобразуются ОС в команды управления соответствующим устройством. Так, например, клавиатура обычно рассматривается как текстовый *файл*, из которого компьютер получает данные в символьном формате.
* Специальные файлы, так же как и устройства ввода-вывода, делятся на блок-ориентированные и байт-ориентированные.

Каталоги

* *Каталоги* – системные *файлы*, поддерживающие структуру файловой системы.
* Каталог это, с одной стороны, группа файлов, объединенных пользователем исходя из некоторых соображений (например, файлы, содержащие программы игр, или файлы, составляющие один программный пакет), а с другой стороны – это файл, содержащий системную информацию о группе файлов, его составляющих.
* В каталоге содержится список файлов, входящих в него, и устанавливается соответствие между файлами и их характеристиками (атрибутами).

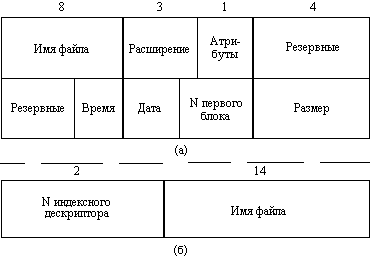
Атрибуты файлов

В разных файловых системах могут использоваться в качестве атрибутов разные характеристики:

* информация о разрешенном доступе;
* пароль для доступа к файлу;
* владелец файла;
* создатель файла;
* флаги "только для чтения", "скрытый файл", "системный файл", "архивный файл", "двоичный/символьный", "временный" (удалить после завершения процесса), флаг блокировки;
* времена создания, последнего доступа и последнего изменения;
* текущий размер файла;
* максимальный размер файла.

Структура каталогов

Каталоги могут непосредственно содержать значения характеристик файлов, как это сделано в файловой системе MS-DOS, или ссылаться на таблицы, содержащие эти характеристики, как это реализовано в ОС UNIX.



*структура записи*

*каталога*

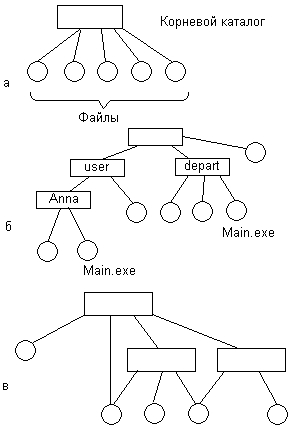
*MS-DOS (32 байта)*

структура записи

каталога

ОС UNIX

Логическая организация ФС



- одноуровневая

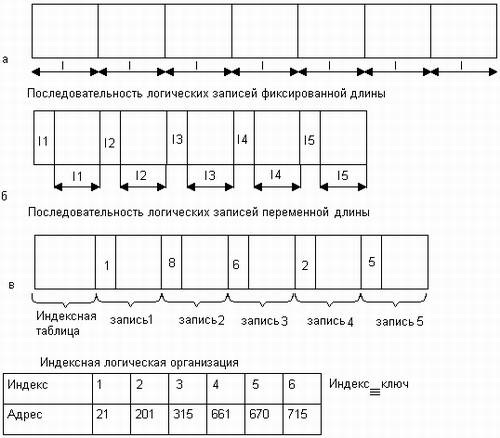
- иерархическая (дерево)

- иерархическая (сеть)

* Каталоги могут образовывать иерархическую структуру за счет того, что каталог более низкого уровня может входить в каталог более высокого уровня.
* Иерархия каталогов может быть деревом или сетью. Каталоги образуют дерево, если файлу разрешено входить только в один каталог, и сеть - если файл может входить сразу в несколько каталогов.
* В MS-DOS каталоги образуют древовидную структуру, а в UNIX'е - сетевую. Как и любой другой файл, каталог имеет символьное имя и однозначно идентифицируется составным именем, содержащим цепочку символьных имен всех каталогов, через которые проходит путь от корня до данного каталога.

Логическая организация файла

Программист имеет дело с логической организацией файла, представляя файл в виде определенным образом организованных логических записей. Логическая запись – это наименьший элемент данных, которым может оперировать программист при обмене с внешним устройством. Даже если физический обмен с устройством осуществляется большими единицами, операционная система обеспечивает программисту доступ к отдельной логической записи.

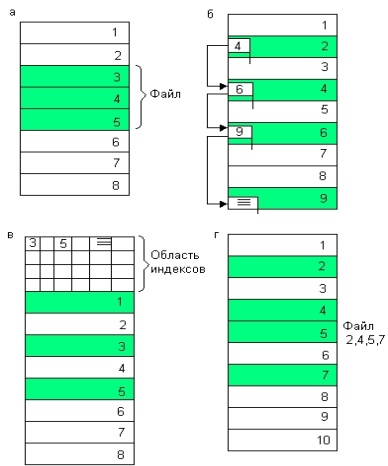
На слайде показаны несколько схем логической организации файла. Записи могут быть фиксированной длины или переменной длины. Записи могут быть расположены в файле последовательно (последовательная организация) или в более сложном порядке, с использованием так называемых индексных таблиц, позволяющих обеспечить быстрый доступ к отдельной логической записи (индексно-последовательная организация). Для идентификации записи может быть использовано специальное поле записи, называемое ключом.

В файловых системах ОС UNIX и MS Windows файл имеет простейшую логическую структуру – последовательность однобайтовых записей.

Физическая организация файла

* Физическая организация файла описывает правила расположения файла на устройстве внешней памяти (например, диске).
* Файл состоит из физических записей – блоков.
* Блок – наименьшая единица данных, которой внешнее устройство обменивается с оперативной памятью.

Способы физической организации



непрерывное

размещение связанный список блоков

связанный перечень номеров блоков

список

индексов

Непрерывное размещение

*Непрерывное размещение* – простейший вариант физической организации (рисунок а), при котором файлу предоставляется последовательность блоков диска, образующих единый сплошной участок дисковой памяти. Для задания адреса файла в этом случае достаточно указать только номер начального блока. Другое достоинство этого метода - простота. Но имеются и два существенных недостатка. Во-первых, во время создания файла заранее не известна его длина, а значит не известно, сколько памяти надо зарезервировать для этого файла, во-вторых, при таком порядке размещения неизбежно возникает фрагментация, и пространство на диске используется не эффективно, так как отдельные участки маленького размера (минимально 1 блок) могут остаться не используемыми.

Связанный список блоков

Следующий способ физической организации – *размещение в виде связанного списка блоков дисковой памяти* (рисунок б). При таком способе в начале каждого блока содержится указатель на следующий блок. В этом случае адрес файла также может быть задан одним числом – номером первого блока. В отличие от предыдущего способа, каждый блок может быть присоединен в цепочку какого-либо файла, следовательно фрагментация отсутствует. Файл может изменяться во время своего существования, наращивая число блоков. Недостатком является сложность реализации доступа к произвольно заданному месту файла: для того, чтобы прочитать пятый по порядку блок файла, необходимо последовательно прочитать четыре первых блока, прослеживая цепочку номеров блоков. Кроме того, при этом способе количество данных файла, содержащихся в одном блоке, не равно степени двойки (одно слово израсходовано на номер следующего блока), а многие программы читают данные блоками, размер которых равен степени двойки.

Связанный список индексов

Популярным способом, используемым, например, в файловой системе FAT операционной системы MS-DOS, является использование *связанного списка индексов*. С каждым блоком связывается некоторый элемент – индекс. Индексы располагаются в отдельной области диска (в MS-DOS это таблица FAT). Если некоторый блок распределен некоторому файлу, то индекс этого блока содержит номер следующего блока данного файла. При такой физической организации сохраняются все достоинства предыдущего способа, но снимаются оба отмеченных недостатка: во-первых, для доступа к произвольному месту файла достаточно прочитать только блок индексов, отсчитать нужное количество блоков файла по цепочке и определить номер нужного блока, и, во-вторых, данные файла занимают блок целиком, а значит имеют объем, равный степени двойки.

Перечень номеров блоков

В заключение рассмотрим задание физического расположения файла путем простого перечисления номеров блоков, занимаемых этим файлом. Этот способ используется в NTFS и ФС UNIX.

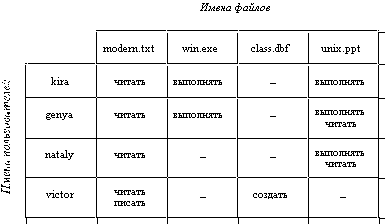
Например, UNIX использует вариант данного способа, позволяющий обеспечить фиксированную длину адреса, независимо от размера файла. Для хранения адреса файла выделено 13 полей. Если размер файла меньше или равен 10 блокам, то номера этих блоков непосредственно перечислены в первых десяти полях адреса. Если размер файла больше 10 блоков, то следующее 11-е поле содержит адрес блока, в котором могут быть расположены еще 128 номеров следующих блоков файла. Если файл больше, чем 10+128 блоков, то используется 12-е поле, в котором находится номер блока, содержащего 128 номеров блоков, которые содержат по 128 номеров блоков данного файла. И, наконец, если файл больше 10+128+128\*128, то используется последнее 13-е поле для тройной косвенной адресации, что позволяет задать адрес файла, имеющего размер максимум 10 + 128 + 128\*128 + 128\*128\*128.

Права доступа к файлу

Определить права доступа к файлу – значит определить для каждого пользователя набор операций, которые он может применить к данному файлу. В разных файловых системах может быть определен свой список дифференцируемых операций доступа.

Этот список может включать следующие операции:

* создание файла
* уничтожение файла
* открытие файла
* закрытие файла
* чтение файла
* запись в файл
* дополнение файла
* дополнение файла
* поиск в файле
* получение атрибутов файла
* установление новых значений атрибутов
* переименование
* выполнение файла
* чтение каталога

Матрица прав доступа

Строки соответствуют всем пользователям

Столбцы соответствуют всем файлам системы

На пересечении строк и столбцов указываются разрешенные операции

В некоторых системах пользователи могут быть разделены на отдельные категории. Для всех пользователей одной категории определяются единые права доступа. Например, в системе UNIX все пользователи подразделяются на три категории: владельца файла, членов его группы и всех остальных.

Различают два основных подхода к определению прав доступа:

* избирательный доступ, когда для каждого файла и каждого пользователя сам владелец может определить допустимые операции;
* мандатный подход, когда система наделяет пользователя определенными правами по отношению к каждому разделяемому ресурсу (в данном случае файлу) в зависимости от того, к какой группе пользователь отнесен.

 Кэширование диска

* Перехват запросов к внешним блочным ЗУ, промежуточным программным слоем – подсистемой буферизации (ПБ).
* ПБ представляет собой буферный пул, располагающийся в ОЗУ, и комплекс программ, управляющих этим пулом по принципу кэш-памяти.
* Каждый буфер пула равен одному блоку.

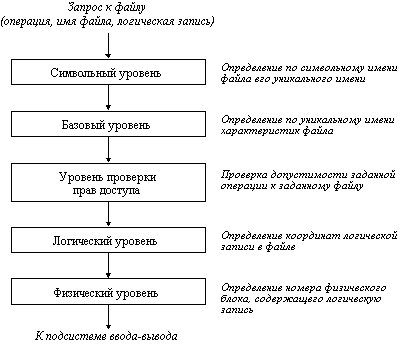
Кэширование диска – чтение

* При запросе на чтение некоторого блока подсистема буферизации (ПБ) просматривает свой буферный пул.
* Если требуемый блок находится, то ПБ копирует его в буфер запрашивающего процесса. Таким образом, операция В/В считается выполненной без физического обмена с устройством.
* При отсутствии свободного буфера на диск вытесняется наименее используемая информация.
* В результате очевиден выигрыш во времени доступа к файлу. Если же нужный блок в буферном пуле отсутствует, то он считывается с устройства и одновременно с передачей запрашивающему процессу копируется в один из буферов подсистемы буферизации.

Кэширование диска – запись

* сквозная
* отложенная (lazy commit)

Общая модель ФС

Функционирование любой файловой системы можно представить многоуровневой моделью, в которой каждый уровень предоставляет некоторый интерфейс (набор функций) вышележащему уровню, а сам, в свою очередь, для выполнения своей работы использует интерфейс (обращается с набором запросов) нижележащего уровня.

* Задачей символьного уровня является определение по символьному имени файла его уникального имени. В файловых системах, в которых каждый файл может иметь только одно символьное имя (например, MS-DOS), этот уровень отсутствует, так как символьное имя, присвоенное файлу пользователем, является одновременно уникальным и может быть использовано операционной системой. В других файловых системах, в которых один и тот же файл может иметь несколько символьных имен, на данном уровне просматривается цепочка каталогов для определения уникального имени файла. В файловой системе UNIX, например, уникальным именем является номер индексного дескриптора файла (i-node).
* На следующем, базовом уровне по уникальному имени файла определяются его характеристики: права доступа, адрес, размер и другие. Как уже было сказано, характеристики файла могут входить в состав каталога или храниться в отдельных таблицах. При открытии файла его характеристики перемещаются с диска в оперативную память, чтобы уменьшить среднее время доступа к файлу. В некоторых файловых системах (например, HPFS) при открытии файла вместе с его характеристиками в оперативную память перемещаются несколько первых блоков файла, содержащих данные.
* Следующим этапом реализации запроса к файлу является проверка прав доступа к нему. Для этого сравниваются полномочия пользователя или процесса, выдавших запрос, со списком разрешенных видов доступа к данному файлу. Если запрашиваемый вид доступа разрешен, то выполнение запроса продолжается, если нет, то выдается сообщение о нарушении прав доступа.
* На логическом уровне определяются координаты запрашиваемой логической записи в файле, то есть требуется определить, на каком расстоянии (в байтах) от начала файла находится требуемая логическая запись. При этом абстрагируются от физического расположения файла, он представляется в виде непрерывной последовательности байт. Алгоритм работы данного уровня зависит от логической организации файла. Например, если файл организован как последовательность логических записей фиксированной длины *l*, то n-ая логическая запись имеет смещение *l* \*(n-1) байт.
* Для определения координат логической записи в файле с индексно-последовательной организацией выполняется чтение таблицы индексов (ключей), в которой непосредственно указывается адрес логической записи.
* На физическом уровне файловая система определяет номер физического блока, который содержит требуемую логическую запись, и смещение логической записи в физическом блоке. Для решения этой задачи используются результаты работы логического уровня - смещение логической записи в файле, адрес файла на внешнем устройстве, а также сведения о физической организации файла, включая размер блока.
* После определения номера физического блока, файловая система обращается к системе ввода-вывода для выполнения операции обмена с внешним устройством. В ответ на этот запрос в буфер файловой системы будет передан нужный блок, в котором на основании полученного при работе физического уровня смещения выбирается требуемая логическая запись.