Рязань, 1 лекция, 02.10.19
Файловые подсистемы.

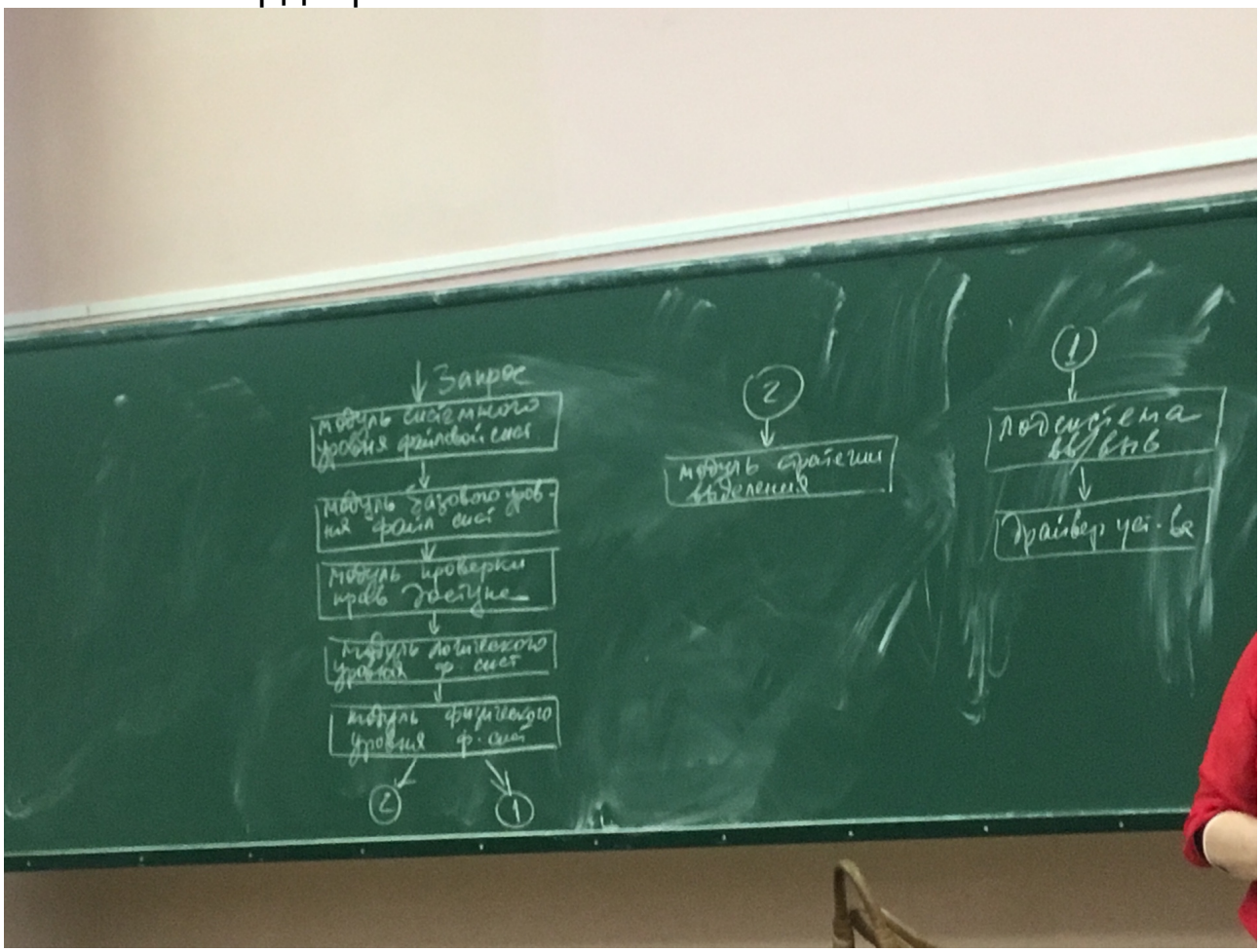
Мы рассматривали иерархическую машину Медника Донована. Она построена относительно процессов, на самом верхнем уровне находился уровень управления данными, или по-другому, это уровень файловых подсистем, уровень долговременного хранения данных, доступа.
Временные файлы называются рабочими. Все файлы рабочие. Временные файлы, создаваемые для обслуживания системы.
Существует масса вариаций в определении фпайловых систем, которые описывают одну суть. В отношении файлов такая вариация отсутствует, если понимать о чем идет речь.
Для долговременного хранения информации, существуют специальные внешние устройства.
Управление файлами осуществляется частью ос, которую принято называть файловой системой или файловой подсистемой (FileSystem). Файловая система - часть ос, которая отвечает за возможность долгого и надежного хранения и доступа инфо (создание, чтение, завись, именование, переименование, удаление, изменение прав доступа и т.д). Если файл предназначен для хранения информации (данных), то фс управляет процессом хранения и обеспечивает последующий доступ к этой информации. То есть файл, фактически, в файловой системе является единицей хранения информации. Для обычных файлов важно подчеркивать, что они хранятся во вторичной памяти. Бывают специальные файлы (программные каналы), они создаются в оперативной памяти.

Определение файла (UNIX-овое)
Файл - каждая индивидуальная индексированная совокупность информации называется файлом.
Каждая индивидуально идентифицируемая единица информации называется файлом.

Каждая поименованная совокупность данных, хранящаяся во вторичной памяти называется файлом. Речь идет об обычных файлах
(Это определение Рязановой)

Когда говорят о фс, то речь идет об обычных файлах. Файловая система определяет: а) формат сохраняемой информации и способ ее физического хранения, б) связывает формат физического хранения и API для доступа к файлам. Файловая система не интерпретирует информацию, хранящуюся во вторичной памяти. Этим занимаются соответствующие программы. Вторичную память, профессионально определяют как энерго-независимую.

Рассмотрим обобщенную модель файловой подсистемы и рассмотрим отдельно ее уровни, фс как правило имеет иерархическую организацию, в ней определены уровни, эти уровни от вышележащего к нижележащему. Верхние уровни связаны с работой с файлами пользователей. Нижние уровни связаны с хардвэром.



Это пример структурирования файловой системы. Вопросы структурирования ос при управлении информации, является едва ли не самыми важными при разработке системы. Очевидно, что такое структурирование должно учитывать задачи, которые должна решать файловая система.

Задачи фс:

1. Обеспечение удобного доступа пользователей к файлам как часть задачи именования файлов.
2. Собственно, именование файлов, то есть присвоение файлам уникальных идентфикаторов, с помощью которых можно обеспечить доступ к файлам.
3. Обеспечение программного интерфейса для работы с файлами пользователей и приложений.
4. Отображение логической модели или логического предсатвления файлов на физическую организацию хранения данных на соответствующих носителях.
5. Обеспечение надежного хранения файлов, доступа к ним и обеспечение защиты от несанкционированного доступа.
6. Обеспечение совместного использования файлов.

Системный уровень - это символьный уровень.
Развитые файловые системы (win, unix/linux), должны обеспечивать возможность существования в системе нескольких файлов с одинаковыми именами, обеспечивать возможность доступа к одному и тому же файлу по разным именам. Поэтому, в системах должна существовать соответствующая информация (справочник). Такой справочник состоит из двух частей, или состоит из двух отдельных справочников (символьные и базовые уровни)

Символьный уровень - уровень именования файлов удобный для юзера, уровень именования каталогов, уровень оперирования пользователем и приложением файла.

Удобным способом стркутуирования различных файлов, являются директории. Директории определяются как каталоги (папки).

Базовый уровень это уже уровень идентификации файла. Очевидно, что и файловая подсистема это программа, и как любая программа, она работает по определенным принципам. Понятно, что файл должен иметь уникальный id. А поскольку система должна давать вохможность называть файл разными именами (hardlink) и имеет несколько файлов с одним именем. Файл должен быть описан в системе, чтобы можно было с ним работать. Модуль проверки прав доступа - это кэп.

Модуль логического уровня. Мы говорили о том, что любая программа считает, что она начинается с нулевого адреса, что называется логическим адресным пространством, с файлами также. Когда мы создаем файл указатель стоит на начало файла, файл имеет логическое адресное пространство, которе также начинается с нуля. То есть логический уровень позволяет обеспечить доступ к данным в формате отличном от формата их физического хранения. Это непрерывная последовательность адресов.

Обычно, файловая система не накладывает никаких ограничений на структуру данных файла и не интерпретирует их. Структурой данных управляет пользователь. Существуют разные форматы хранения данных которые известны только программам, которые создают файлы и работают с ними. Например, текстоввые файлы формируются в виде последовательности символов, которые объединяются в строки произвольной длины (условно) и строка заканчивается специальным символом конца строки. Поэтому они и называются текстовыми, потому что для программиста это набор строк. Для работы с текстовыми файлами предназначены специальные программы - текстовые редакторы. Информация в текстовых файлах хранится посимвольно. Один символ - 1 байт. Можно написать программу, которая будет работать с любым тектовым файлом.

В операционных системах существует два типа файлов - байт-ориентированные и блок-ориентированные. В системе существует два типа устройств - символьные и блочные.

В байт-ор. Файлах информация хранится по байтам, соответственно в байт устройствах передача данных по байтам.

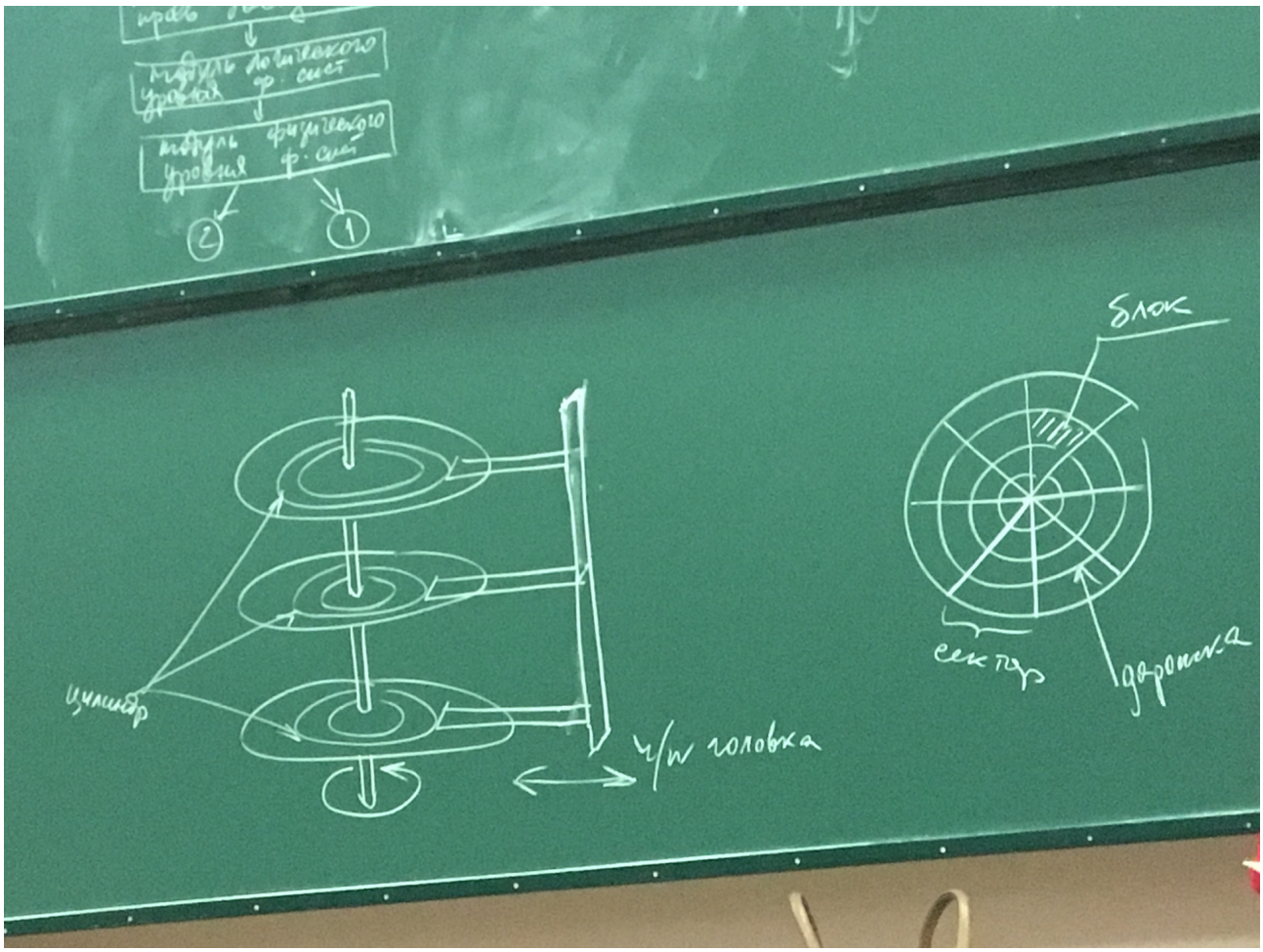
Блок-ориентированные файлы, в них информация хранится в блоках одинакового размера.

В современных системах блок-ориентированных устройствами являются только диски.

Соответсвенно, языки, такие как си и паскаль поддерживают такие файлы, в паскале это тектовые и типизированные, а в си это текстовые и бинарные.

Физический уровень хранения файлов.
БОЯН НАХУЙ, давайте глянем.
Наименьший адресуемый участок конкретной пластины определенной дорожки назвывается сектор. На пластинах концентрические дорожки. Совокупность одних и тех же дорожек разных пластин называется цилиндром.

Read-Write головка.



Если связное распределение, то файл на диске занимает непрерывное адресное пространство - непрерывные сектора.

Несвязное - сектора выделяются в разброс, но при этом система должна обеспечивать возможность адресовывать каждый отдельный блок, выделенный конкретно файлу. Позволяет более эффективно использовать адресное пространство диска.

Таким образом, данное представление о файловой системе позволяет обсепечить общий интерфейс для любой файловой системы. Все файловые системы могут быть описаны таким образом и именно это позволяет создать некоторый общий интерфейс для любого типа файловой системы. Это возможно потому, что ядро реализует слой абстракции над своим низкоуровневым интерфейсом. Именно этот подход используется в файловой подсистеме UNIX, которая реализует

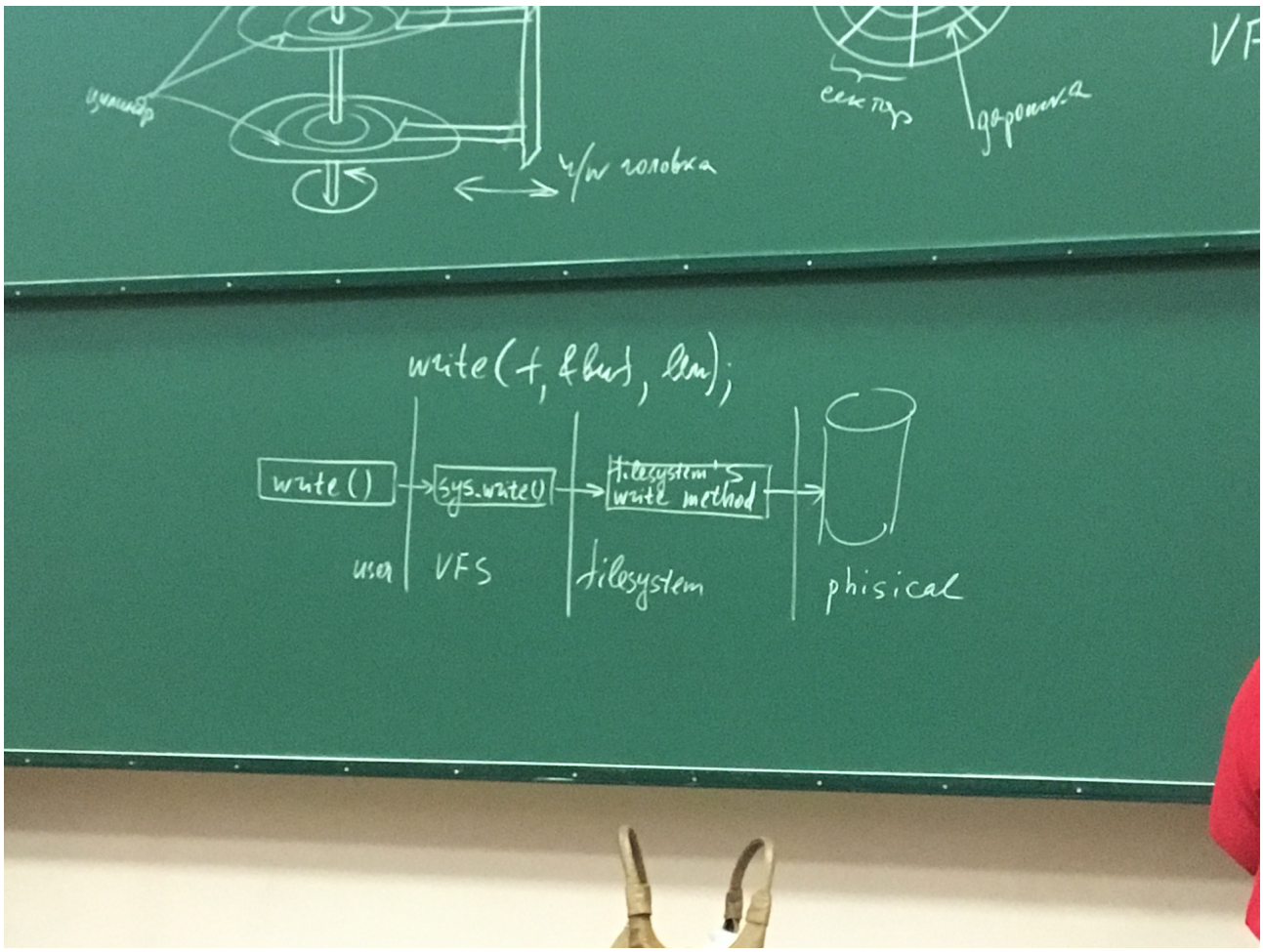
интерфейс, названный VFS/vnode (Virtual File System/ virtual node).
Linux изменил этот интерфейс, отказавшись от vnode, и называется VFS (Virtual File System.). VFS представляет общую файловую модель, которая способна отображать общие возможности и поведение любой мыслимой файловой системы.

Уровень абстракции VFS работает на основе базовые концептуальных интерфейсов и структур данных. Каждая отдельная файловая система определяет особенности того, как файл открывается, как выполняется обращение к файлу, как считается информация из файла и т. п.

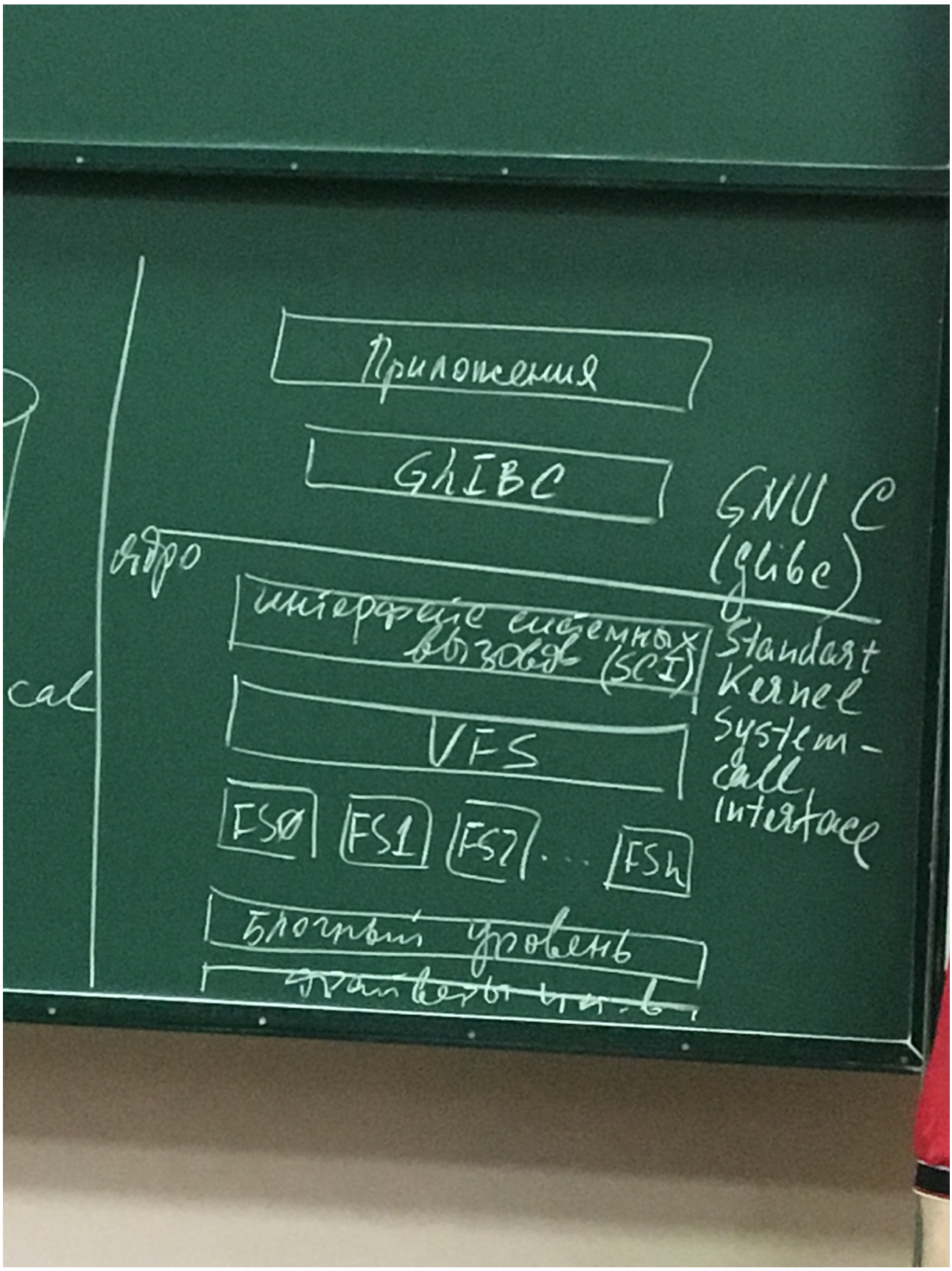
Фактический код файловых систем скрывает детали реализации, однако все фс поддерживают такие понятия как файлы, каталоги, поддерживают такие действия как создание файла, удаление файла, переименование, открытие файла, чтение/запись, закрытие файла и т. д.

Большинство файловых систем запрограммировано так, что API, которые предоставляют файловые системы рассматриваются как абстрактный интерфейс, который ожидаем и понятен VFS.

Например, рассмотрим запрос приложения write(f, buf, len);
В данном случае данное API записывает len байт, которые хранятся в buf, т. е. У приложения есть адрес, в текущую позицию файла. Речь идет о логическом адресном пространстве файла. Этот системны вызов обрабатывается в системе по следующей цепочке:



Системный вызов write сначала обрабатывается общим системным вызовом VFS sys.write(), затем, осуществляется вызов соответсвующего системного вызова конкретной файловой системы, в данном случае, для записи данных в файл.



ФС, которые поддерживает UNIX/LINUX: EXT2, EXT3, UFS, NTFS, APFS, MS-DOS, FAT, FAT32, HPFS.

Внутренняя организация файловой системы. VFS базируется на 4х структурах:
superblock, dentry (directory entry), inode (index node), file.

Безусловно, между этими структурами и объектами, которые описывают эти стрктуры, существует связь.