Тема 6

Файлові системи

План

1. Роль та рівні організації файлової системи
2. Основні поняття файлової системи
3. Приклади файлових систем

1. Роль та рівні організації файлової системи

Операційна система

Процеси та потоки

Адресні простори

Файли

? Навіщо нам файлова система?

Файлова система:

- *організовує зберігання даних на дисках та інших носіях;*
- *надає доступ до цих даних через абстракцію - файл.*

Ну або така

1. Роль та рівні організації файлової системи

Файлові системи різняться (сильно) за будовою та функціонуванням. Але загалом можна виокремити два рівні:

- **Фізичний рівень** (має справу з блоками даних на носіях)
- **Логічний рівень** (має справу з файлами)

Межі цих рівнів, їх компонентний склад - відрізняються залежно від конкретної реалізації.

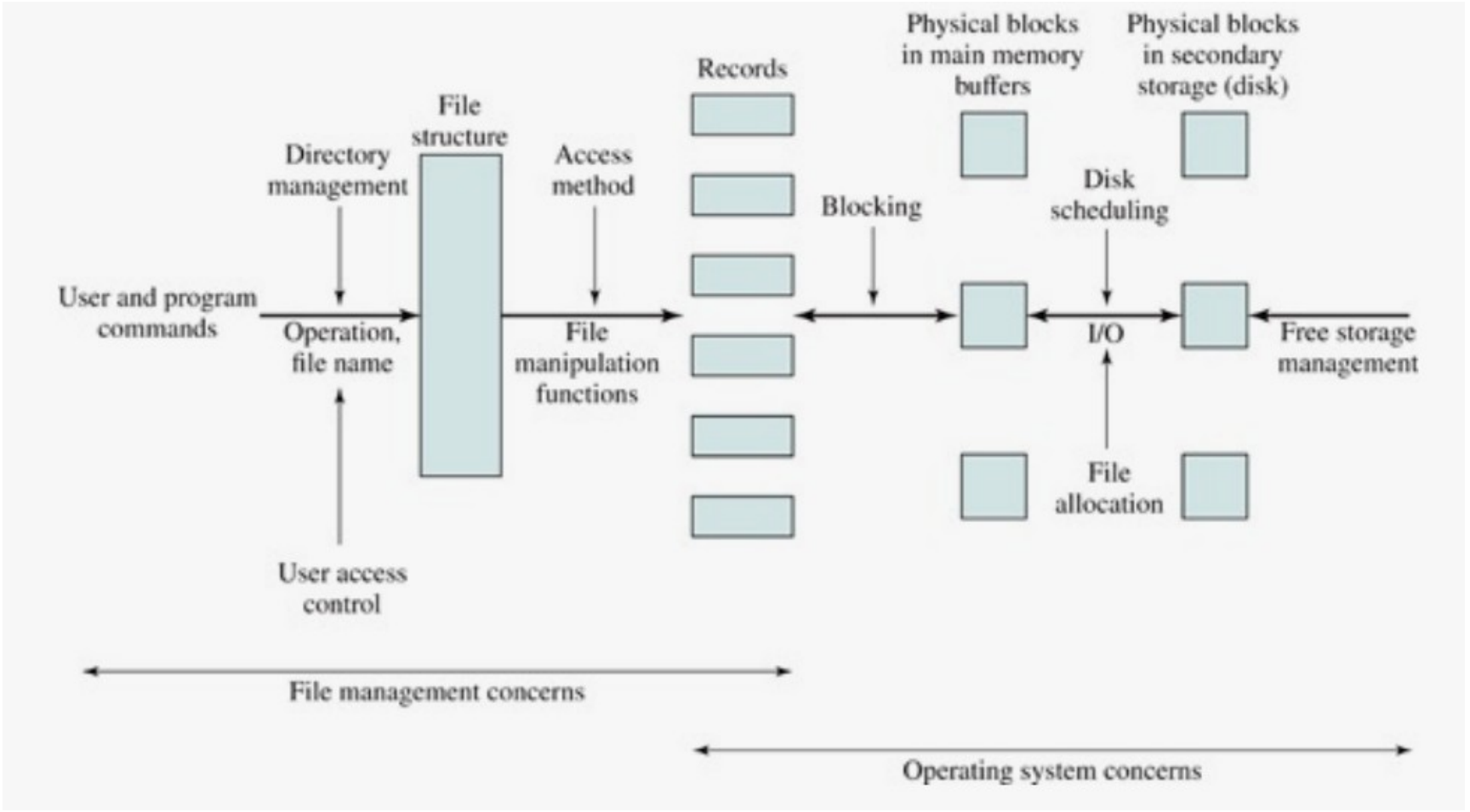
Зв'язок між рівнями - там, де розрізнені блоки інтерпретуються як один файл.

Файлова система об'єднує в собі:

- файли
- службові структури даних для зберігання відомостей про файли
- системні програмні засоби для виконання операцій над файлами (система керування файлами - ***file management system***).

1. Роль та рівні організації файлової системи

Типові елементи системи керування файлами
(file management system)



Схему взято з книги: ***W. Stallings,* Operating Systems Internals and Design Principles,** 9th ed., Pearson, 2017

2. Основні поняття файлової системи

? Що таке файл?

Файлом називають впорядкований набір даних, що зберігається у комп'ютерній системі під спільним ім'ям.

? Що таке файловий атрибут?

Атрибут - суттєва властивість.

Файлові атрибути - властивості файлів.

2. Основні поняття файлової системи

Очікувані характеристики файлів [*Stallings*]

- **Довготермінове існування.** Файли лишаються на носії після відключення живлення.
- **Спільне використання кількома процесами.** Щоб різні процеси могли звернутися - потрібне ім'я. Щоб доступ був безпечний, потрібне розмежування доступу.
- **Структурованість.** Файл може мати спеціальну внутрішню структуру залежно від типу. Файли можуть організовуватися у різноманітні структури, які показують зв'язки між цими файлами (наприклад, ієрархічна структура).

2. Основні поняття файлової системи

? Які файлові атрибути ви знаєте?

Типові атрибути файлів:

* ім'я
* тип
* розмір
* атрибути безпеки (власники, повноваження тощо)
* часові атрибути (дата і час створення, дата і час останньої модифікації, дата і час останнього доступу, ...)
* інші атрибути

2. Основні поняття файлової системи

Вимоги до ***імен файлів*** відрізняються залежно від системи.

Відмінності виявляються зокрема у:

- **Чутливості до регістру символів**

  ? ***file1.txt*** *та* ***File1.txt*** *- це один файл чи різні файли?*

  У Linux (Ext) - різні. У Windows (NTFS) - один.

  (Спробуйте створити обидва файли в одній папці)

- **Ролі розширення файлу**

  ※ Розширення важливіше в одних системах й менш важливе в інших.

  ※ Тип файлу визначається внутрішньою структурою, а не розширенням.

  ※ Файли можуть без розширення. Розширень може бути кілька.

  ※ Розширення може визначати асоційовані з файлом програми.

- **Максимально допустимій довжині імені файлу**

  Часто близько 255 символів.

  Стандарт для старіших систем - 8.3 (у сучасних системах - для сумісності).

2. Основні поняття файлової системи

? Що таке каталог?

На **каталог** можна дивитися з двох позицій:

1) каталог - це група файлів

2) каталог - це особливий файл, що зберігає відомості про деяку групу файлів.

Каталог = папка = директорія

В англомовних джерелах найпоширеніші: *directory* (директорія), *folder* (папка).

2. Основні поняття файлової системи

Абсолютне ім'я (повне ім'я) файлу

`C:\users\usr1\dir1\file5.txt` ФС NTFS (ОС Windows)

`/home/usr1/dir1/file5.txt` ФС ext4 (ОС Linux)

Відносне ім'я файлу

поточний каталог: `/home/usr1/dir1`

абсолютне ім'я файлу: `/home/usr1/dir1/file5.txt`

відносне ім'я файлу: `file5.txt`

2. Основні поняття файлової системи

Журналювання

Сучасні файлові системи часто журнальовані (*journaling file systems*).

У журнальованій файловій системі зміни, що вносяться у файлову систему, розглядаються як транзакції.

Якщо під час транзакції трапився збій, то відповідна операція потім:

а) поновлюється;

б) скасовується (відкочується);

але не виконується частково (бо це б порушило цілісність файлової системи).

Щоб поновлення або відкочування були можливими, потрібна фіксація операцій над файловою системою у *журналах*.

2. Основні поняття файлової системи

Розділи і томи

Розділ (partition, іноді - volume).

Том (volume) - набір секторів на накопичувачі, використовуваний операційною системою для зберігання даних.

Вищезгадані сектори не обов'язково розміщені на одному носії (том може бути результатом об'єднання кількох носіїв).

Може бути по-різному:

- один диск - це один том;
- один диск - це кілька томів (кожний розділ - окремий том);
- кілька дисків - це один том.

3. Приклади файлових систем

? Які файлові системи ви знаєте?

Приклади файлових систем

UFS (Unix File System)

Ext (Ext, Ext2, Ext3, Ext4) - Linux

HFS, HFS+ (Hierarchical File System), APFS (Apple File System) - macOS

FAT (FAT12, FAT16, FAT32), NTFS - Windows

ISO 9660, UDF (Universal Disk Format) - оптичні диски

ZFS - Sun Microsystems

VMFS (Virtual Machine File System) - VMware

...

3. Приклади файлових систем

FAT

FAT - сімейство файлових систем (FAT12, FAT16, FAT32, інші модифікації).

FAT = File Allocation Table (таблиця розміщення файлів)

Файлові блоки = ***кластери***

Розмір кластера:

- великий розмір кластера → внутрішня фрагментація
- малий розмір кластера → велика кількість кластерів → уповільнення операцій з файлами
- також залежить від розміру сектора
- загалом - від 4 Кб до 256 Кб

Файли - ланцюжки кластерів.

Кожному кластеру відповідає ***індексний вказівник*** у таблиці FAT.

3. Приклади файлових систем

FAT

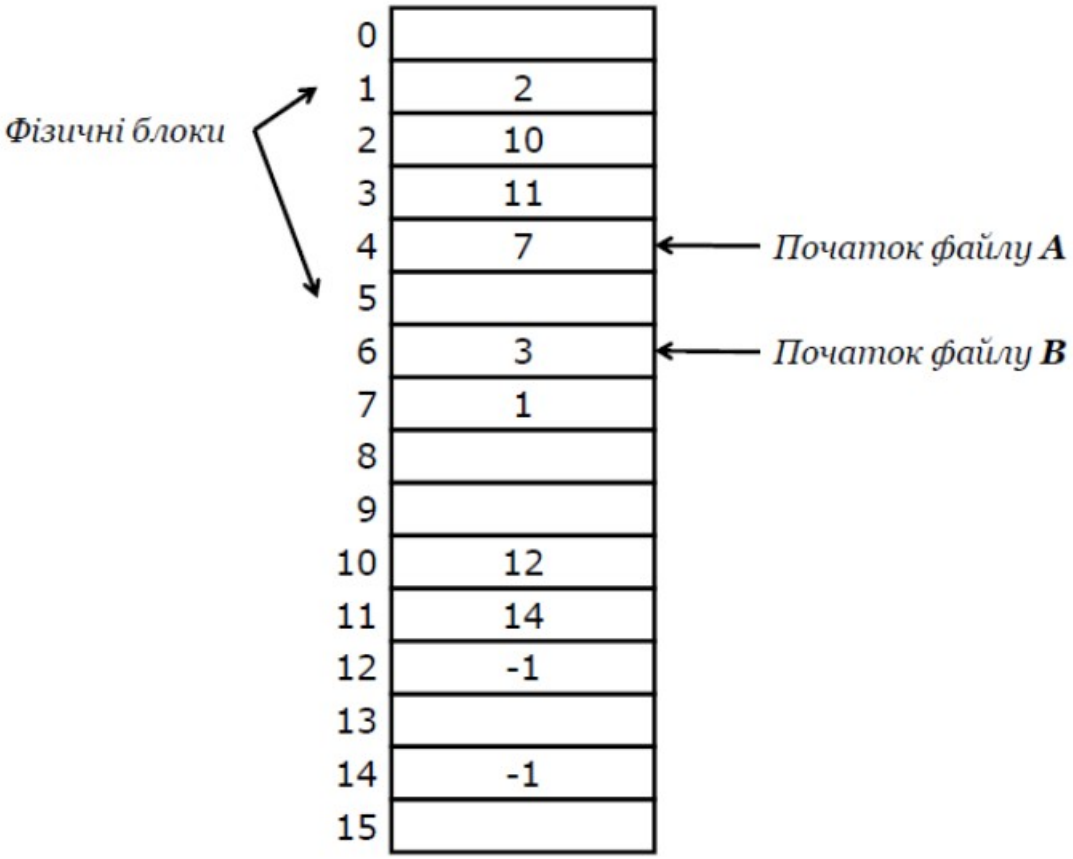
Відомості про кластер в індексному вказівнику:

- кластер вільний чи зайнятий
- чи є кластер збійним
- чи є кластер зарезервованим
- якщо кластер зайнятий не останнім фрагментом файлу - номер кластера з продовженням цього файлу
- якщо кластер містить останній фрагмент файлу - мітка кінця файлу - EOC (*End of Clusterchain* - кінець ланцюжка кластерів)

3. Приклади файлових систем

FAT

Розміщення файлів у FAT



Схему взято з книги: ***Tanenbaum, Bos.*** *Modern Operating Systems, 2014*

3. Приклади файлових систем

FAT

Запис каталогу у файловій системі FAT містить наступні поля:

- ім'я
- розширення
- атрибути
- зарезервоване поле (10 байтів)
- час модифікації (створення)
- дата модифікації (створення)
- номер першого блоку
- розмір

Довжина імені файлу

FAT12, FAT16 - формат 8.3

FAT32 - довжина 256 символів

3. Приклади файлових систем

FAT

Обмеження на розмір файлів:

4 ГіБ (можна розширювати додатковими засобами)

Недоліки файлових систем FAT:

- не передбачено розмежування доступу до файлів
- сильна схильність до фрагментації
- слабкий захист від краху (резервна копія таблиці FAT)
- уся таблиця весь час має перебувати у пам'яті

3. Приклади файлових систем

NTFS

NTFS (New Technology File System) - файлова система для Windows платформи NT.

Основні нововведення (порівняно з FAT):

- Розмежування доступу (дескриптори захисту та ACL)
- Журнальованість
- Резервні копії критично важливих системних даних
- Більші файли та томи

NTFS працює з **кластерами**.

Будь-який елемент всередині тому розглядається як файл.

Кожний файл має атрибути. Вміст файлу - теж атрибут.

3. Приклади файлових систем

NTFS

Том у NTFS:

- **Завантажувальний сектор** (*partition boot sector*). Містить відомості про розбиття тому, службові структури файлової системи, дані для завантаження файлової системи.
- **MFT** (*master file table*). Головна файлова таблиця (див. далі)
- **Системні файли** (дзеркальна копія MFT, log-файл з переліком транзакцій, відомості про зайняті кластери, відомості про підтримувані на цьому томі файлові атрибути тощо)
- **Власне файли**

3. Приклади файлових систем

NTFS

MFT

MFT має структуру таблиці.

Рядки MFT містять перелік всіх файлів та їхніх атрибутів у межах тому.

Кожному файлу в NTFS відповідає принаймні один запис у MFT. Якщо файл великий, записів може бути більше.

Розмір одного запису MFT - 1 Кб.

Запис у MFT містить:

- атрибут вмісту (якщо файл маленький - сам вміст, якщо файл більший - частина вмісту + адреси кластерів, що містять файл);
- решту атрибутів.

MFT теж є файлом.

3. Приклади файлових систем

NTFS

MFT: типові атрибути

- Стандартна інформація (доступ лише для читання, читання/запис, часові атрибути тощо)
- Список атрибутів (якщо всі атрибути не вміщуються в один запис MFT)
- Ім'я файлу
- Дескриптор захисту (англ. *security descriptor*; власник, ACL тощо)
- Дані файлу (один атрибут даних без імені за замовчуванням + додаткові пойменовані атрибути даних за потреби)
- Спеціальні атрибути для папок та томів
- Бітове поле (показує використовувані записи в MFT чи папці).

3. Приклади файлових систем

NTFS

MFT

Перші 16 записів MFT зарезервовані для службової інформації.

Цим записам теж відповідають файли – **метафайли** (їхні імена починаються з символу $).

Метафайли зберігають інформацію, яка дає змогу відновлювати коректний стан файлової системи після збою.

Деякі метафайли:

Номер	Ім'я відповідного метафайлу	Призначення
0	$MFT	Головна файлова таблиця
1	$MFTmirr	Дзеркальна копія MFT (її перших 16 записів)
2	$LogFile	Журнал
4	$AttrDef	Стандартні атрибути файлів

3. Приклади файлових систем

Ext

Ext (від *extended* - розширена) - сімейство файлових систем для Linux.
Походять від UFS (Unix Filesystem).

Представники:

* **Ext** (1992, Remy Card)
* **Ext2** (1993, Remy Card)
* **Ext3** (2001, Stephen Tweedie) + підтримка журналювання
* **Ext4** (2008, Theodore Tso) + підтримка журналювання

3. Приклади файлових систем

Ext

З кожним файлом асоціюється окремий об'єкт - **айнод** (*inode*, від *index node* - індексний вузол).

Інформація айнода

• Атрибути файлу:

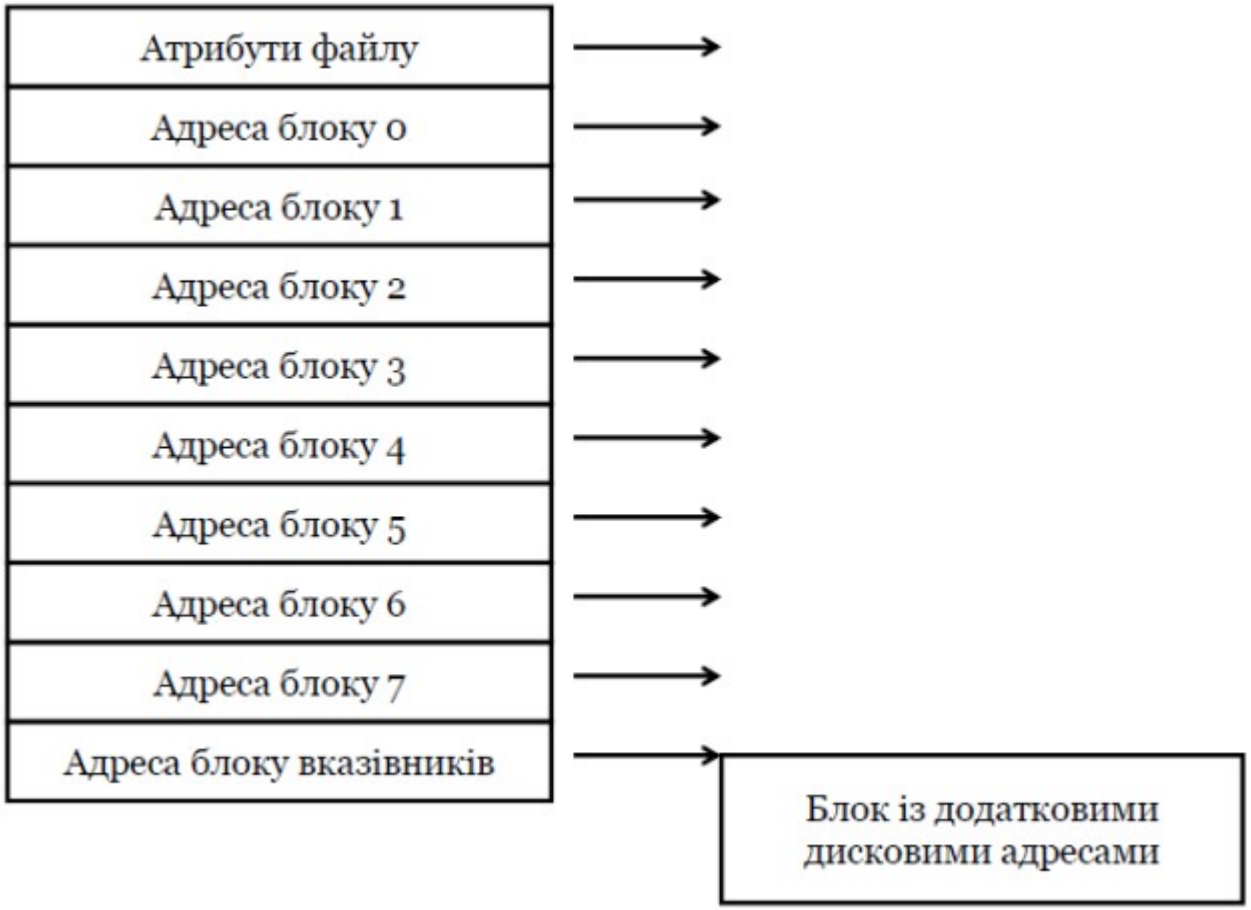
* тип файлу (звичайний файл, директорія, файл пристрою тощо);
* розмір файлу;
* дата та час останньої модифікації файлу;
* ідентифікатор користувача-власника, групи-власника;
* атрибути доступу.

• Вказівники на ті блоки диску, в яких зберігаються фрагменти цього файлу.

3. Приклади файлових систем

Ext

Айнод у попередніх версіях Ext (і в UFS):



Схему взято з книги: ***Tanenbaum, Bos.*** *Modern Operating Systems, 2014*

3. Приклади файлових систем

Ext

В Ext4 для адресації використовуються *екстенти*.

Екстент (*extent*) - діапазон суміжних фізичних блоків, відведений під зберігання файла.

Екстент визначає:

* адресу початкового блоку

* кількість блоків в екстенті

Економія пам'яті (для одного екстента лише два значення)

Але:

Фрагментованому файлу відповідатиме багато екстентів. Для боротьби з фрагментацією використовується ***відстрочене виділення пам'яті*** (allocating-on-flush).

3. Приклади файлових систем

А що як знадобиться працювати з декількома файловими системами?

Приклад: VFS (Virtual File System)

VFS

Використовується у Linux.

Суть VFS: це загальна модель властивостей і поведінки довільної допустимої файлової системи.

Файли у VFS - об'єкти у пам'яті накопичувача.

Модуль відображення (*mapping module*) перетворює характеристики деякої реальної файлової системи у характеристики, очікувані файловою системою VFS.

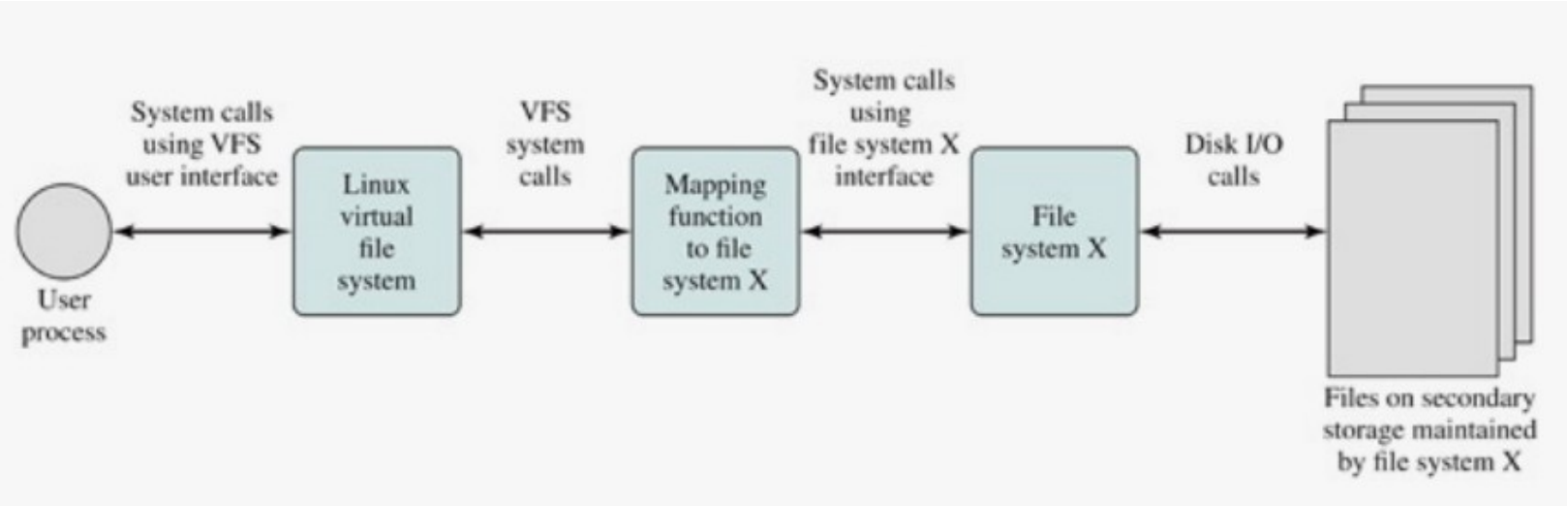
Скажімо, щоб ви могли працювати з мережними ресурсами Windows, в Linux має бути встановлено модуль відображення для SMB.

Часто такі модулі вже встановлено для більшості популярних файлових систем (зокрема, з-під Linux можна працювати з розділами на NTFS).

Розділи з іншими файловими системами **монтуються** (див. лаб. №12).

3. Приклади файлових систем

Як працює VFS



Загальна послідовність роботи VFS:

- Процес ініціює системний виклик через інтерфейс VFS
- Ядро викликає відповідну функцію VFS
- Функція VFS ініціює виклик функції цільової файлової системи (*File System X*)
- Код виклику функції цільової файлової системи передається через функцію модуля відображення. Функція модуля відображення конвертує виклик у виклик цільової файлової системи.

Схему взято з книги: ***W. Stallings,* Operating Systems Internals and Design Principles,** 9th ed., Pearson, 2017

Для самостійного читання

1. **[*Silberschatz, Galvin, Gagne, 2018*]** Chapters 10-11.
2. **[*Stollings, 2017*]** Chapter 12 (paragr. 12.7-12.10).
3. **[Tanenbaum*, 2014*]** Chapter 4.
4. **[*Шеховцов, 2009*]** Розділи 11, 12 (парагр. 12.4), 13.