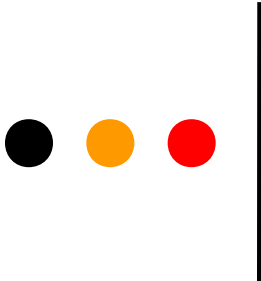


Операційні системи

Лекція 2

Архітектура операційних систем



План лекції

- Поняття архітектури операційної системи
- Ядро і системне програмне забезпечення
- Привілейований режим і режим користувача
- Монолітна архітектура
- Багаторівнева архітектура
- Мікроядрова архітектура
- Архітектура ОС UNIX і Windows
- Об'єктна архітектура



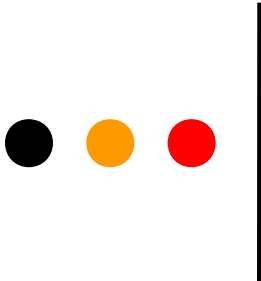
Основні функції ОС

- Керування процесами і потоками
- Керування пам'яттю
- Керування введенням-виведенням
- Керування файлами (файлові системи)
- Мережна підтримка
- Безпека даних
- Інтерфейс користувача



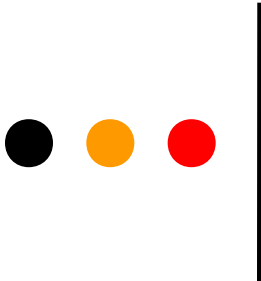
Базові поняття

- *Архітектура операційної системи* визначає набір і структурну організацію компонентів, кожний з яких відповідає за певні функції, а також порядок взаємодії цих компонентів між собою та із зовнішнім середовищем.
- Фундаментальні можливості, які надають компоненти ОС, становлять *механізм* (*mechanism*). Рішення щодо використання цих можливостей визначають *політику* (*policy*). Механізм може бути відокремленим від політики, тоді компонент, що його реалізує, називають “*вільним від політики*” (*policy-free*).
- Базові компоненти ОС, які відповідають за найважливіші функції і виконуються у привілейованому режимі (і зазвичай перебувають у пам'яті постійно), називають *ядром операційної системи* (*operating system kernel*).




Ядро і системне програмне забезпечення

- Ядро
 - Виконується в привілейованому режимі
 - Постійно перебуває в оперативній пам'яті
 - Зазвичай виконує такі функції:
 - Обробка переривань
 - Керування пам'яттю
 - Керування введенням/виведенням
- Системне програмне забезпечення
 - Системні програми (утиліти)
 - Командний інтерпретатор
 - Програми резервного копіювання та відновлення даних
 - Засоби діагностики та адміністрування
 - Системні бібліотеки

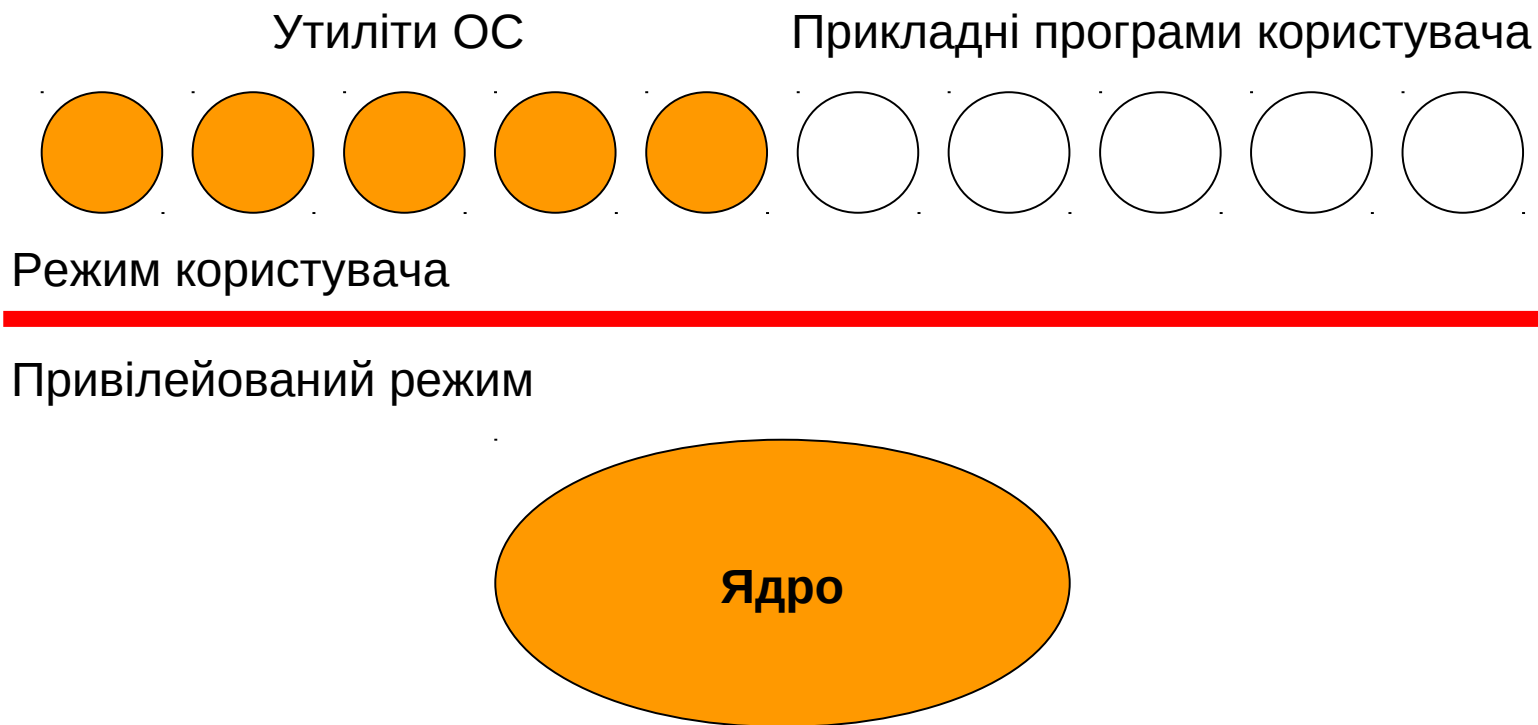


Привілейований режим і режим користувача

- Привілейований режим (режим ядра)
 - Дозволяє втручатись в роботу будь-якої програми (наприклад, для перемикання контекстів або для розв'язання конфліктів)
- Режим користувача
 - Не дозволяє критичні команди (зупинка системи, перемикання контекстів, прямий доступ до пам'яті з заданими межами та до пристроїв введення-виведення)
 - Доступ до функцій ядра здійснюється через *системні виклики*
- Необхідна апаратна підтримка з боку процесора



Типова архітектура ОС: ядро у привілейованому режимі



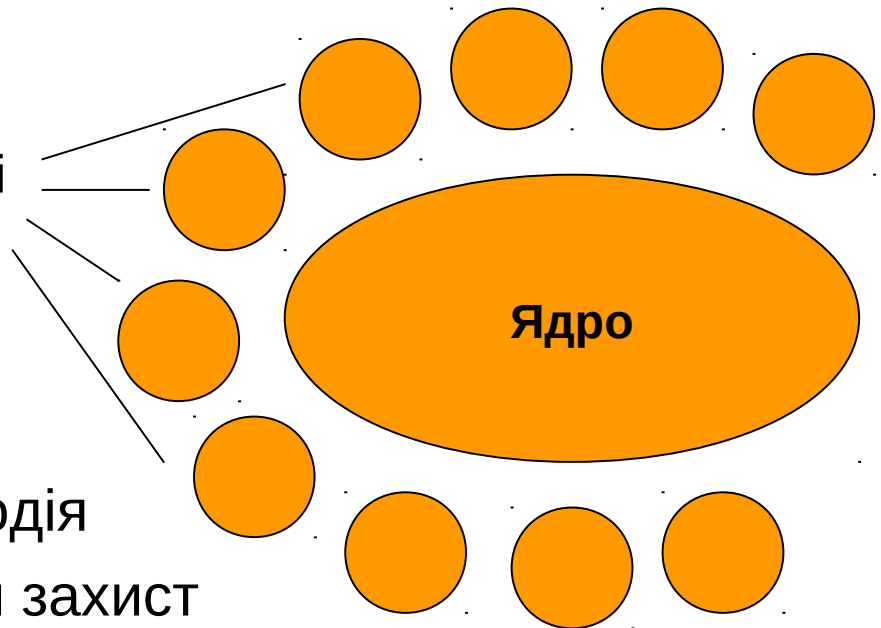
Архітектура ОС Novell NetWare: ядро і прикладні програми в одному режимі

Режим користувача

Привілейований режим

Завантажувані
модулі NLM

- Перевага – швидкодія
- Недолік – відсутній захист





Різні архітектури ОС

○ Монолітні системи

- Усі компоненти знаходяться в ядрі
- Немає чіткої ієрархії компонентів
- Єдиний адресний простір ядра

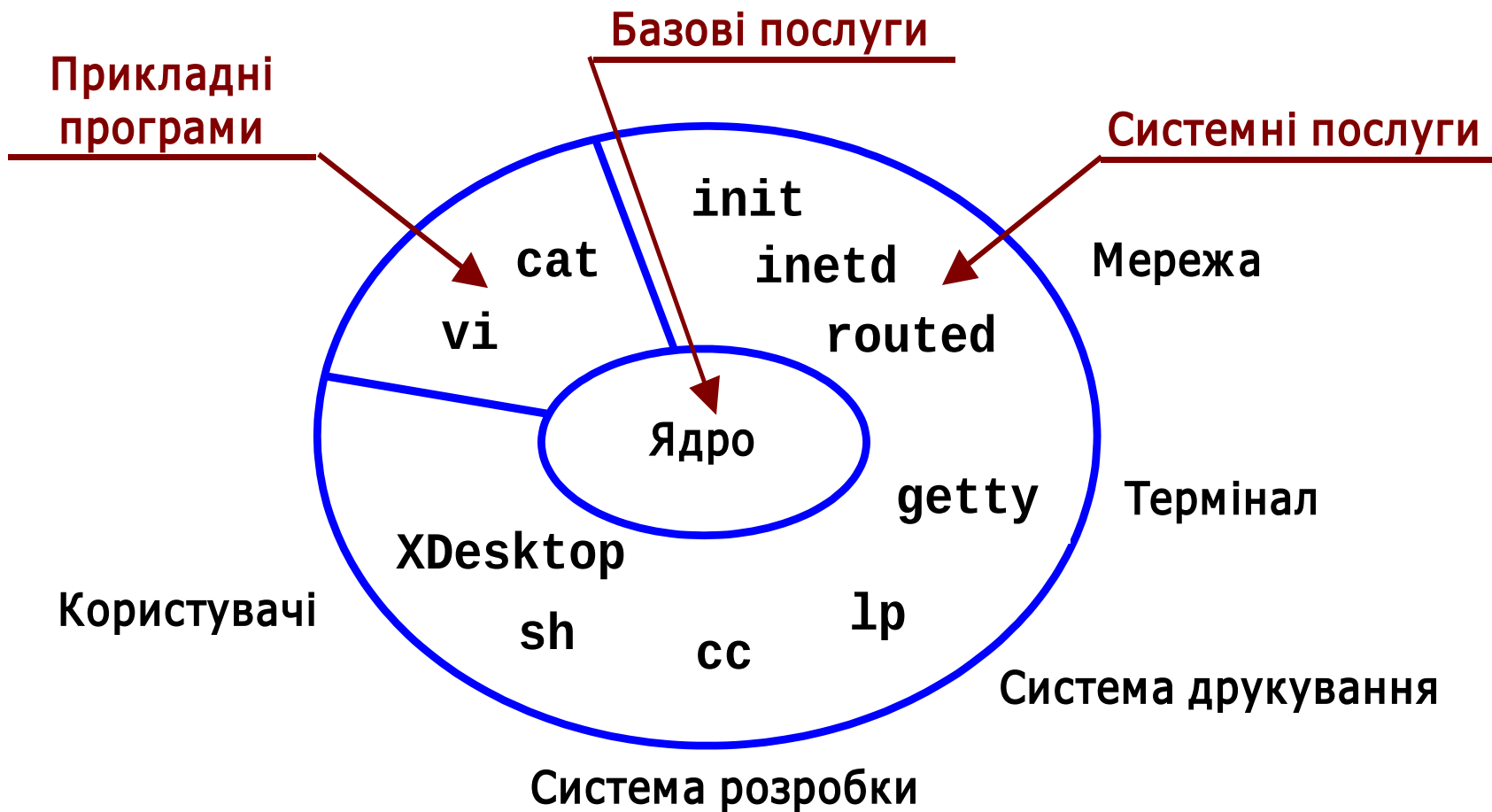
○ Багаторівневі системи

- Компоненти утворюють ієрархію рівнів (шарів)
- Кожний рівень спирається на функції попереднього рівня

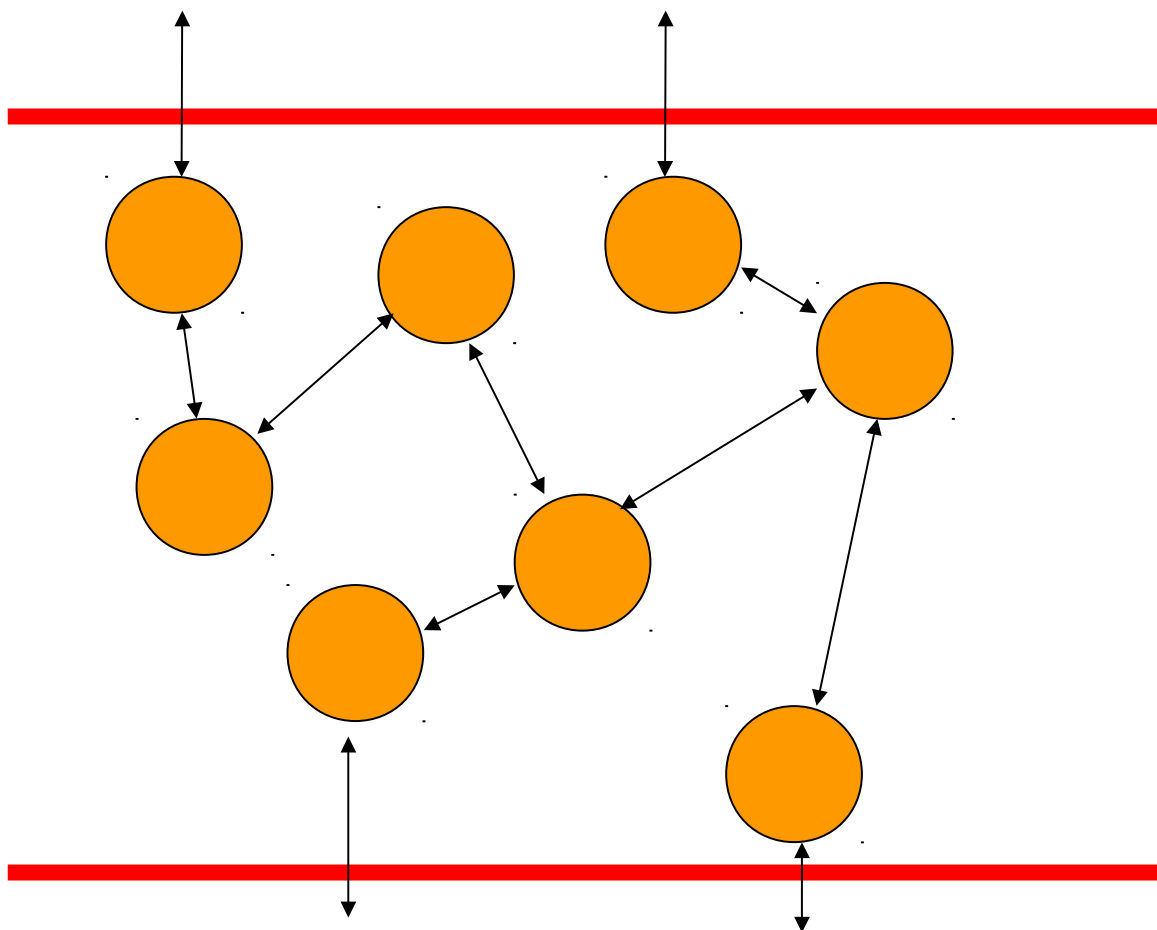
○ Мікроядрова архітектура

- Реалізація більшості функцій винесена за межі ядра у прикладні сервери
- Прикладні сервери і програми користувача взаємодіють шляхом обміну повідомленнями
- Ядро підтримує:
 - управління адресним простором оперативної і віртуальної пам'яті
 - управління процесами і потоками
 - засоби міжпроцесної взаємодії

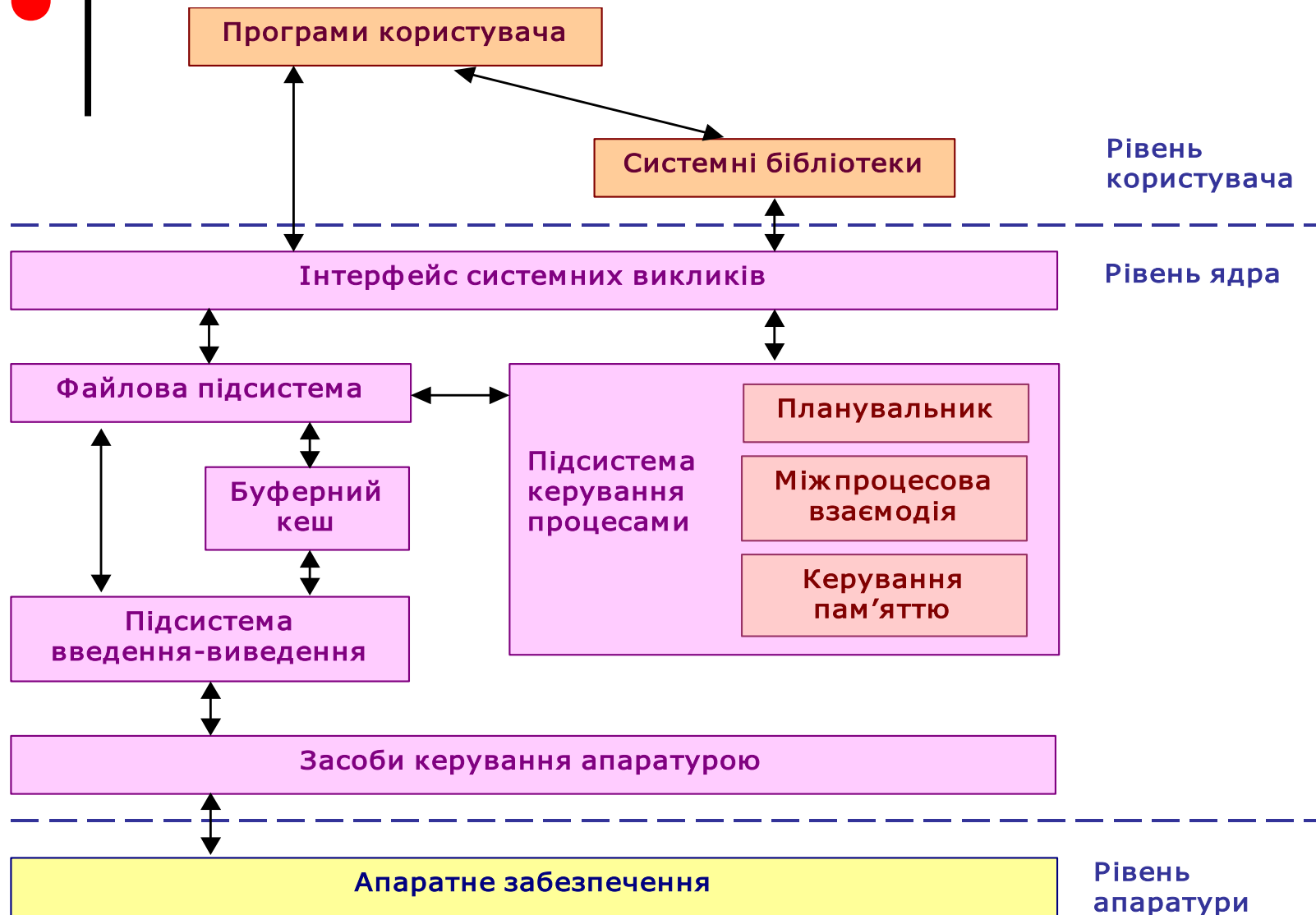
Архітектура системи UNIX (монолітне ядро)



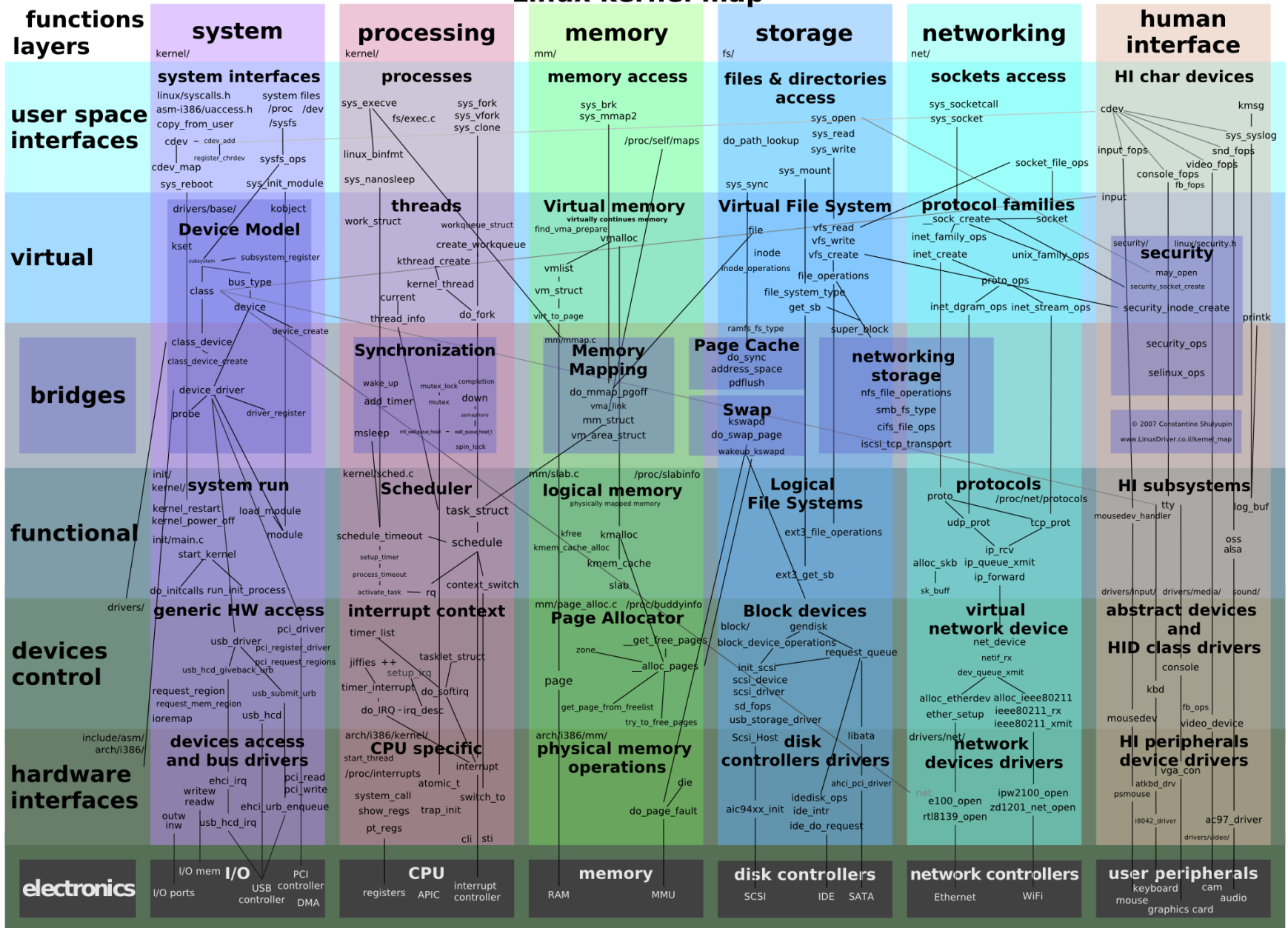
Структура монолітного ядра



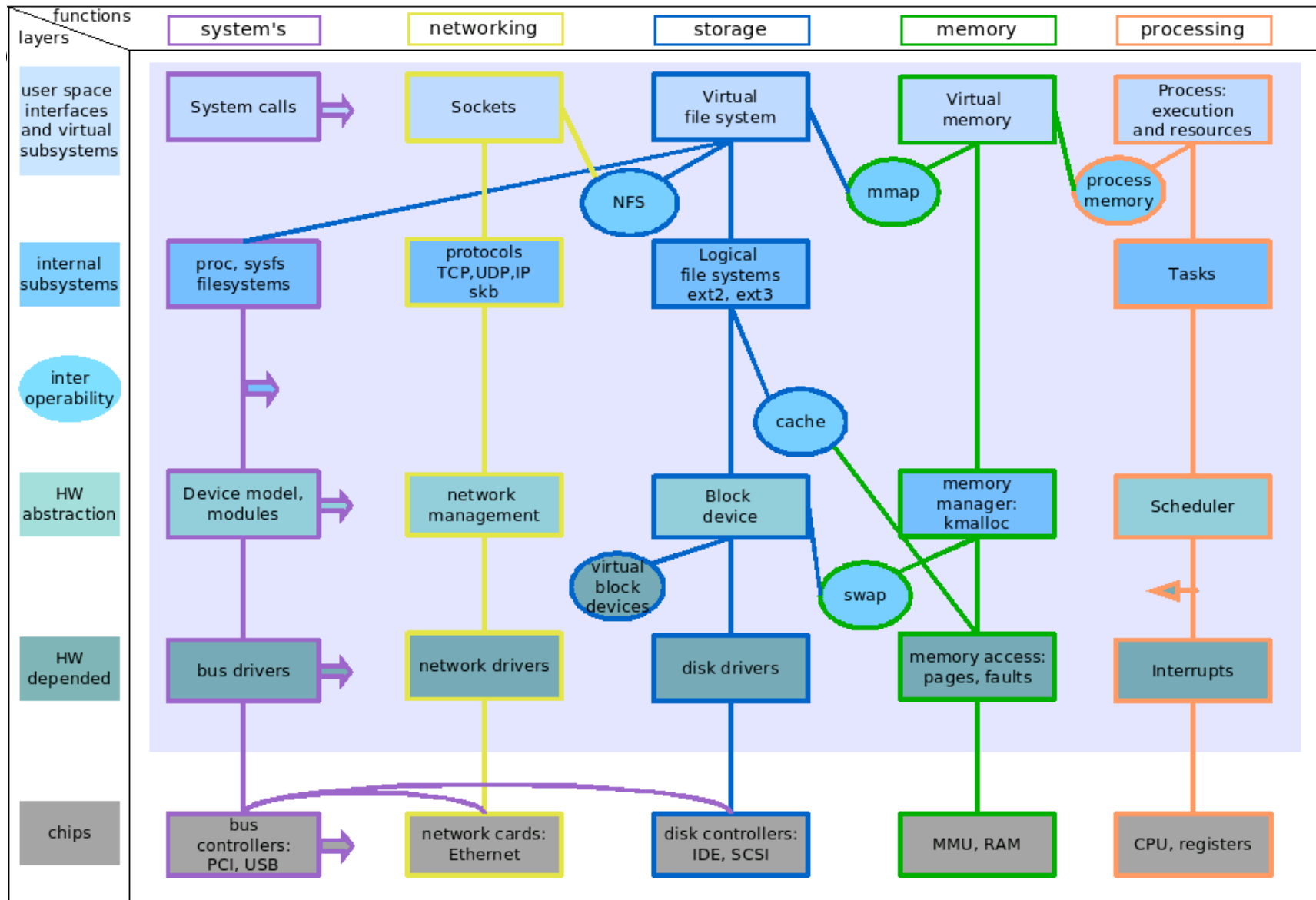
Структура ядра Linux



Linux kernel map

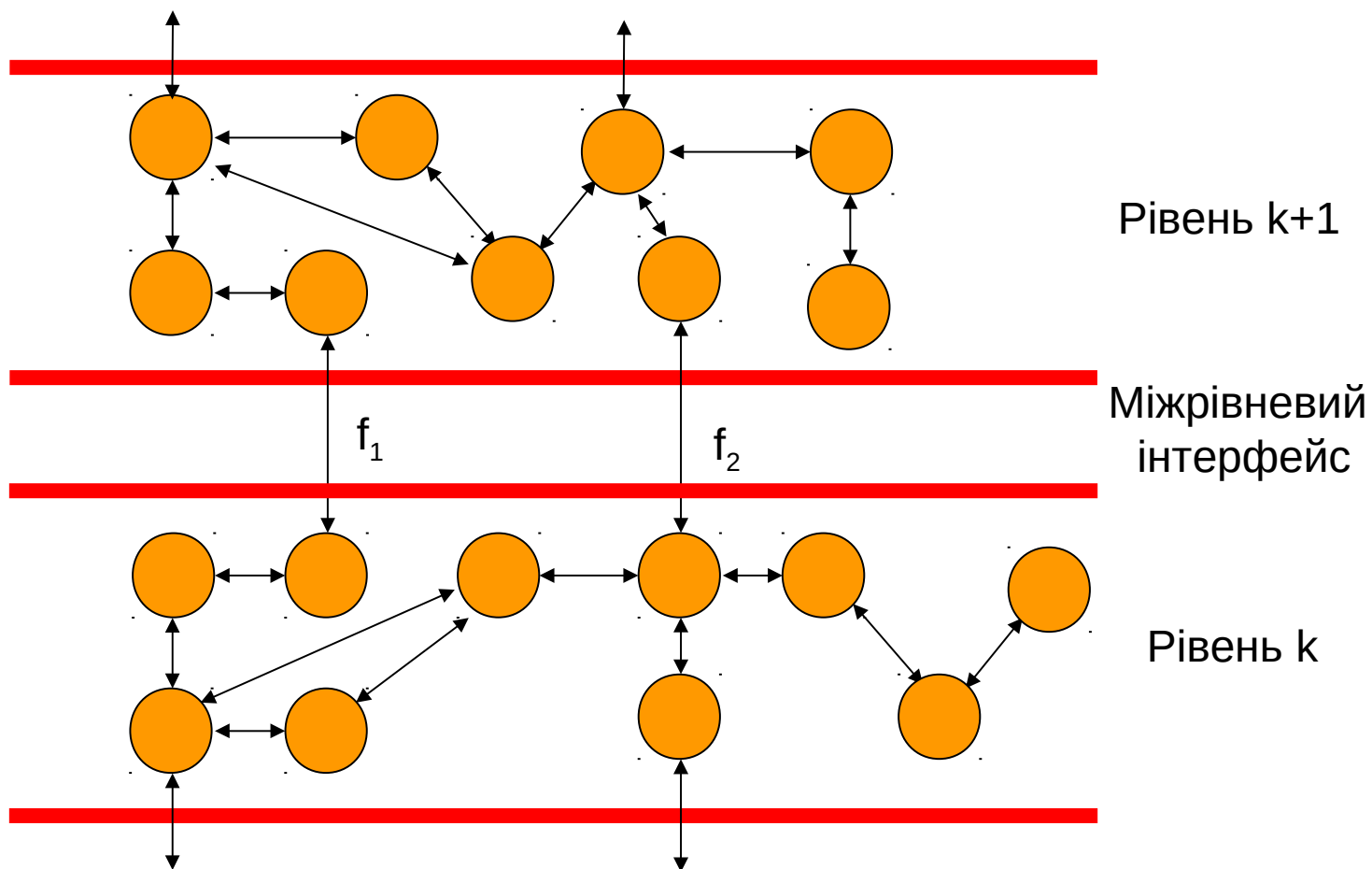


Simplified Linux kernel diagram in form of a matrix map

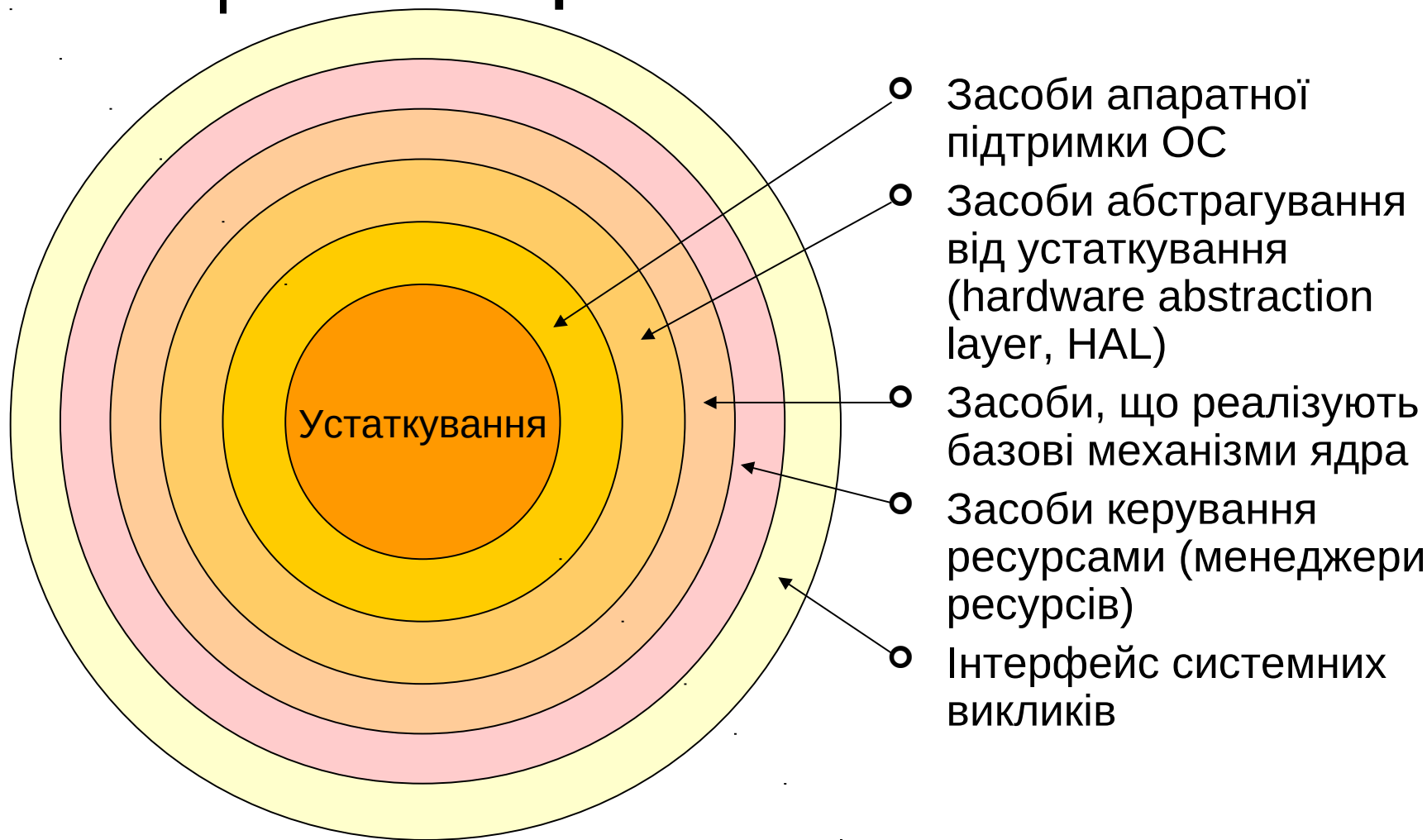


Designed with OpenOffice.org by (cc) (by-nc-sa) Constantine Shulyupin, www.linuxdriver.co.il

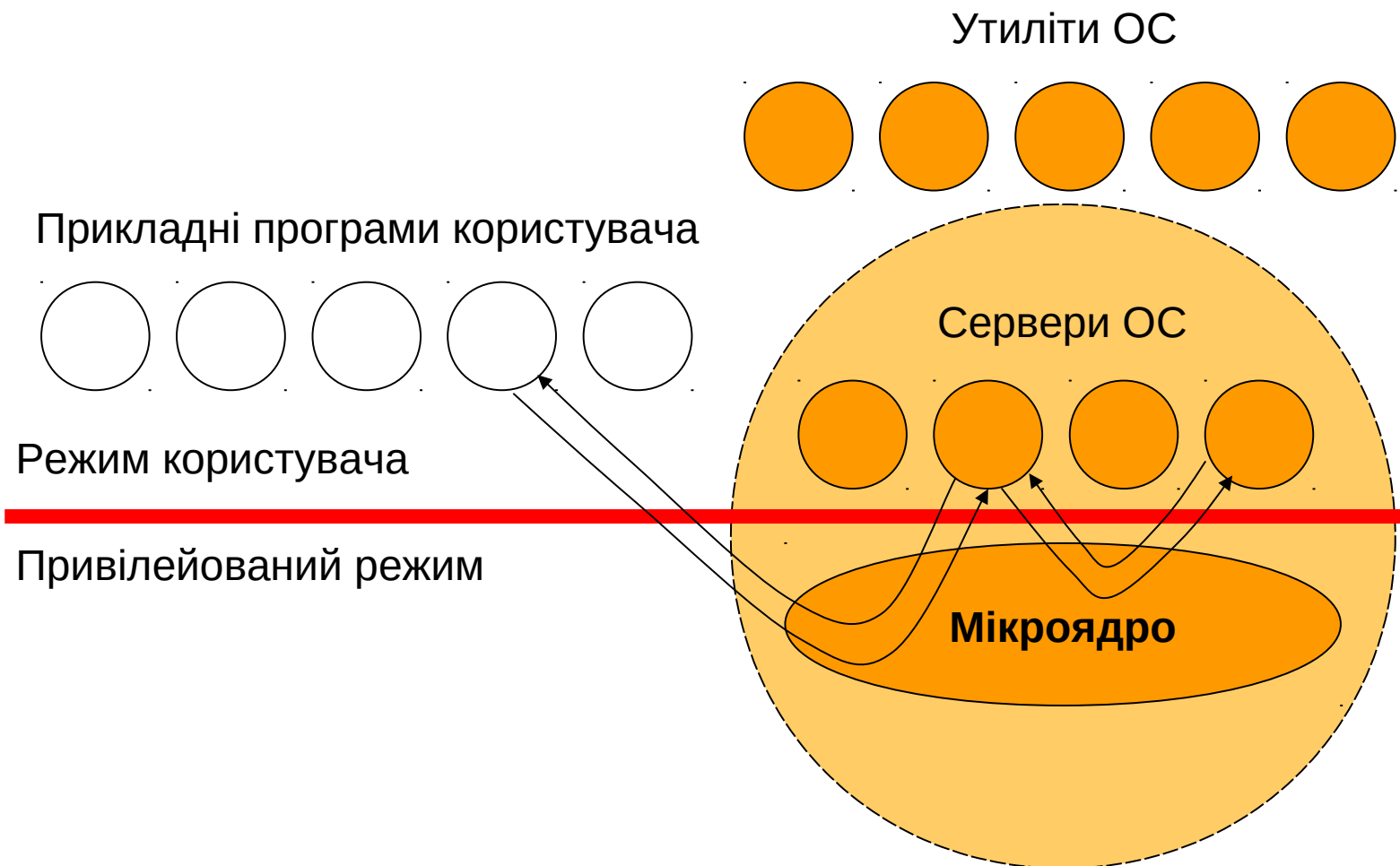
Концепція багаторівневої системи

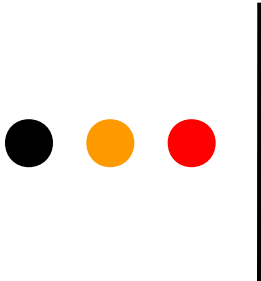


Структура ядра багаторівневої системи



Мікроядрова архітектура





Розвиток концепції мікроядра

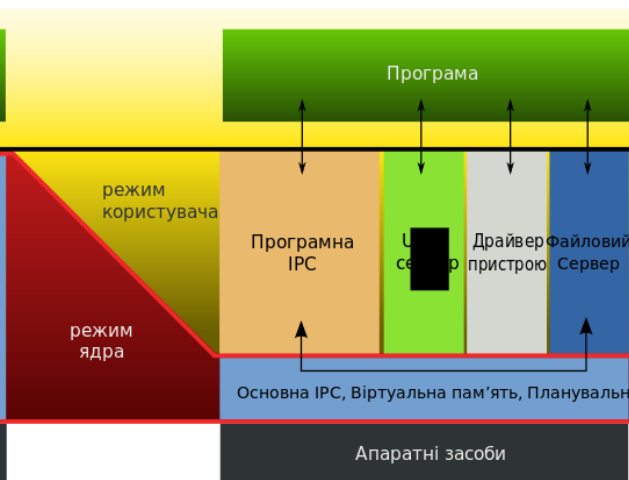
- *Гібридне ядро* (*Hybrid kernel*) — гібрид мікроядра і монолітного ядра
 - Деякі “несуттєві” процеси запускаються у просторі ядра
 - Як будь-який гібрид, комбінує як переваги, так і недоліки
 - Приклад — Microsoft Windows
- *Екзоядро* (*Exokernel*) — подальша мінімізація мікроядра
 - Надає лише функції взаємодії між процесами і безпечного виділення ресурсів
 - Не надає абстракції!
 - Приклад — MIT Exokernel Operating System
- *Наноядро* (*Nanokernel*) — подальша (максимальна) мінімізація мікроядра
 - Виконує лише одне завдання — оброблення апаратних переривань
 - Після оброблення переривання, надсилає результат прикладним програмам, використовуючи той самий механізм переривань
 - Приклад — VMware ESX Server (гіпервізор)

Порівняння систем з монолітним, мікро- та гібридним ядром

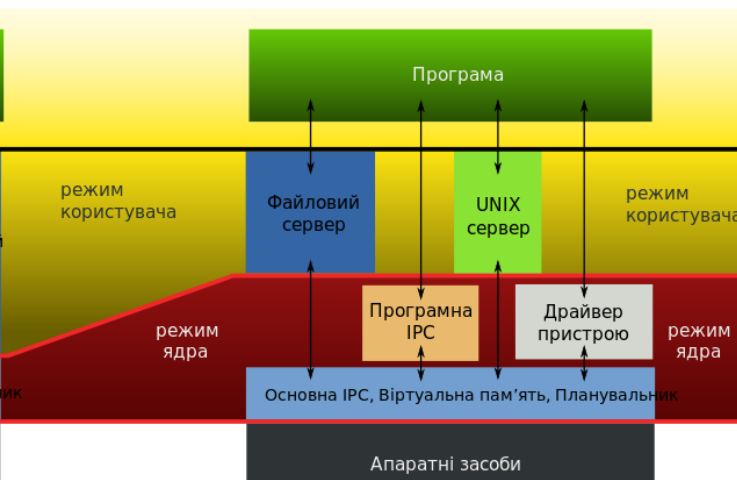
Операційна система з монолітним ядром



Операційна система з мікроядром

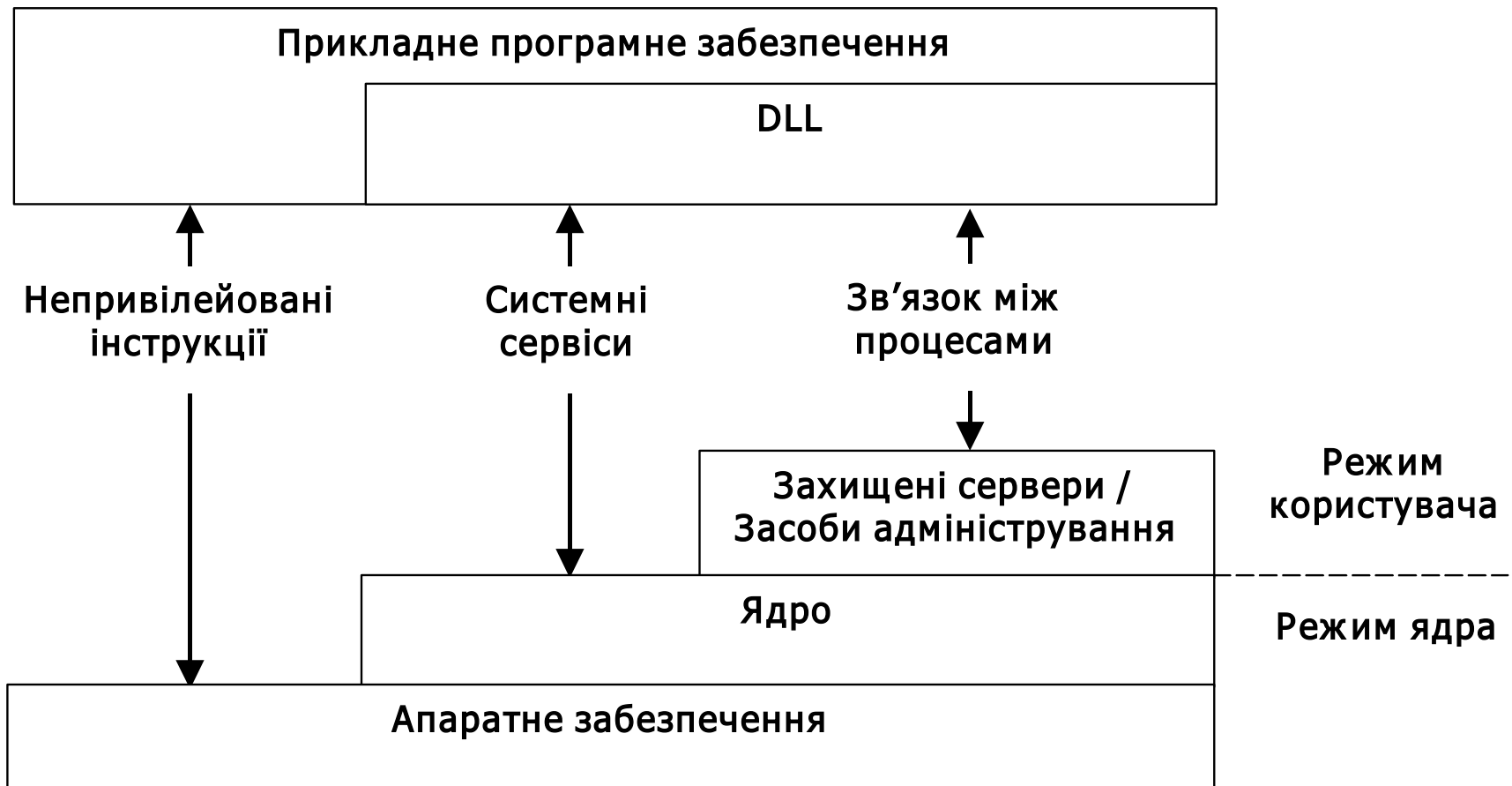


Операційна система з гібридним ядром



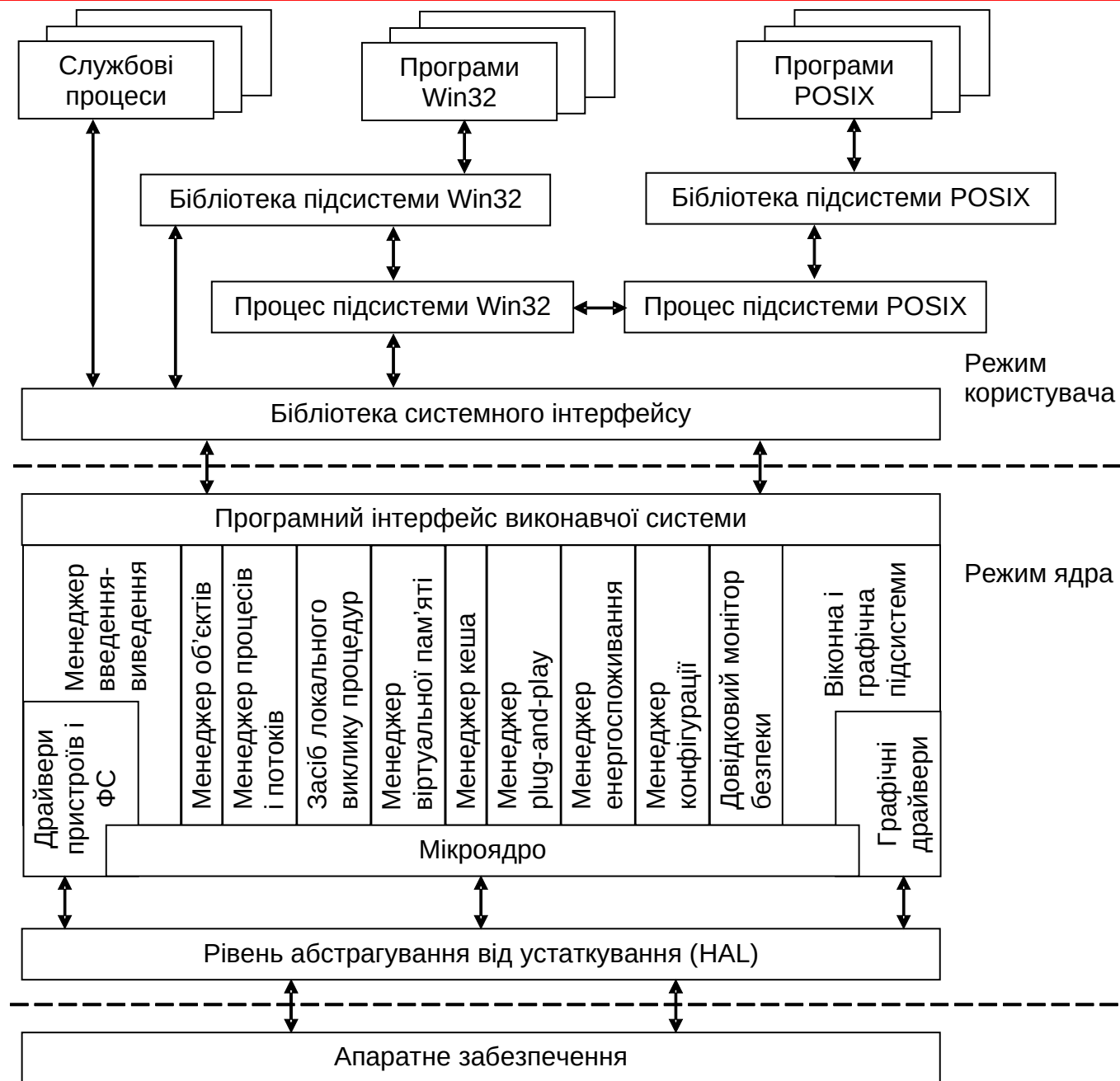
Автор: Ma)(imuM - <http://en.wikipedia.org/wiki/Image:OS-structure.svg>,
CC BY-SA 1.0, <https://uk.wikipedia.org/w/index.php?curid=1613834>

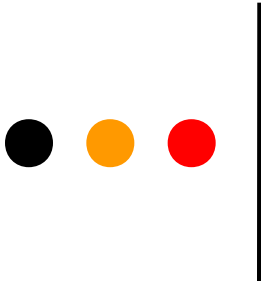
Вертикальна декомпозиція архітектури ОС Windows





Базові КОМПОНЕНТИ ОС Windows NT





Об'єктна архітектура (Windows)

- Імена об'єктів організовані в єдиний *простір імен*
- Об'єкти надають універсальний інтерфейс для доступу до системних ресурсів
 - Доступ до усіх об'єктів здійснюється однаково
 - Після створення об'єкта, або після отримання доступу до наявного, менеджер об'єктів повертає прикладній програмі *дескриптор об'єкта* (*object handle*)
- Забезпечено захист ресурсів
 - Кожну спробу доступу до об'єкта розглядає *підсистема захисту*

Об'єкт має *заголовок* і *тіло*. Структура заголовка об'єкта:

- Ім'я об'єкта, його місце у просторі імен
- Дескриптор захисту
- Витрата квоти (ціна відкриття дескриптора об'єкта)
- Список процесів, що отримали доступ до дескрипторів об'єкта