ITMO Advanced OS 2024

Aleksei Romanovskii, PhD ,
SPb Research Center (CBG OS Lab)

Aleksei Romanovskii: about myself

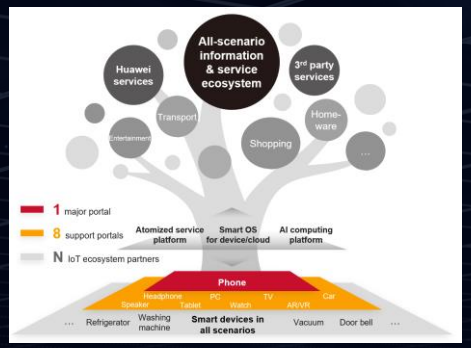
- I graduated at Novosibirsk State University, with degree in Economics and Mathematics, got PhD in Computer Science
  - I lectured on “Algorithms of real-time 3D graphics” in NSU (Faculty of Physics)
- My 3D real-time SW is in Russian Spaceship and military flight simulators
- My JPEG codecs are in Samsung consumer devices
- My compression algorithms are in EMC products
- I developed best on this planet compression and waste elimination algorithms for Huawei enterprise storage
- I have 20+ international patent applications and patents

Attitude

Only those who attempt the absurd can achieve the impossible. **Albert Einstein**

Only those who will risk going too far can possibly find out how far one can go. **Thomas Stearns Elliot**

St. Petersburg Research Center



OS lab

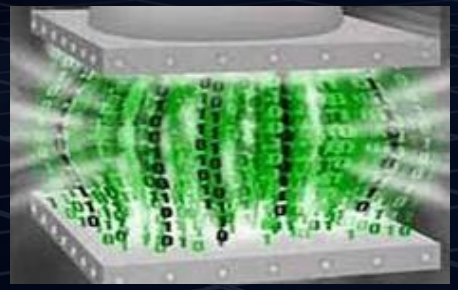
- OS kernel algorithms
- Compilers, etc.

AI lab

- Multi-lingual TTS
- NLU/ASR

Software Engineering

- IDE
- Software Defects Prediction
- software analysis

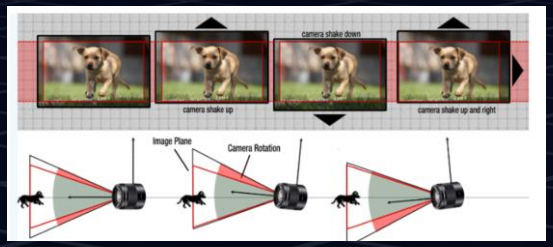


Data Algorithm Center

- Data reduction
- Video compression
- Intelligent Transport System

Cloud Service Center

- HUAWEI Mobile Services (HMS)
- Search & Ranking & Ads
- Network scheduling

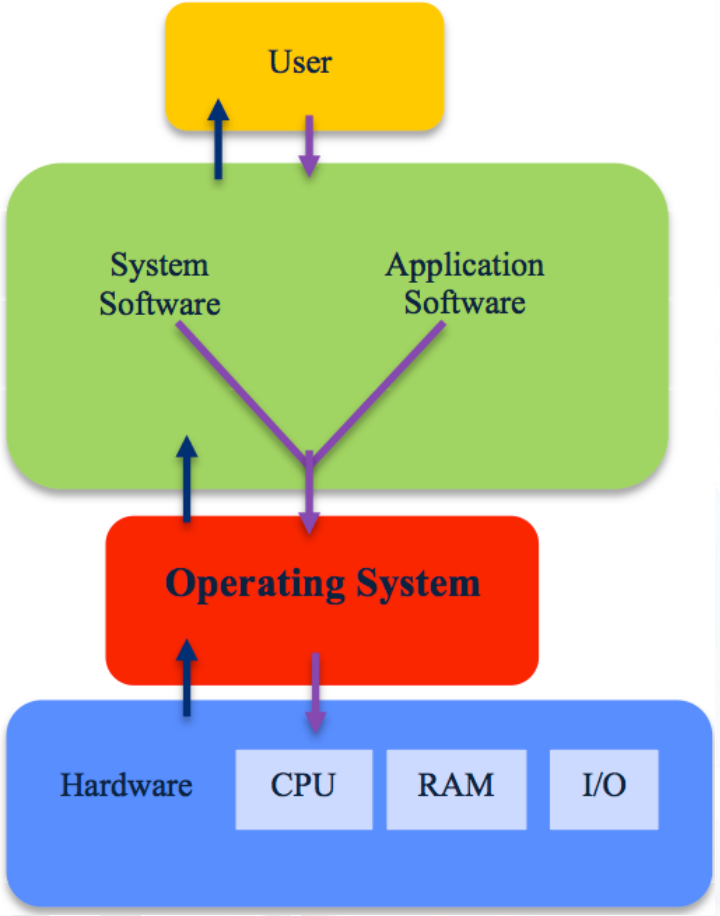


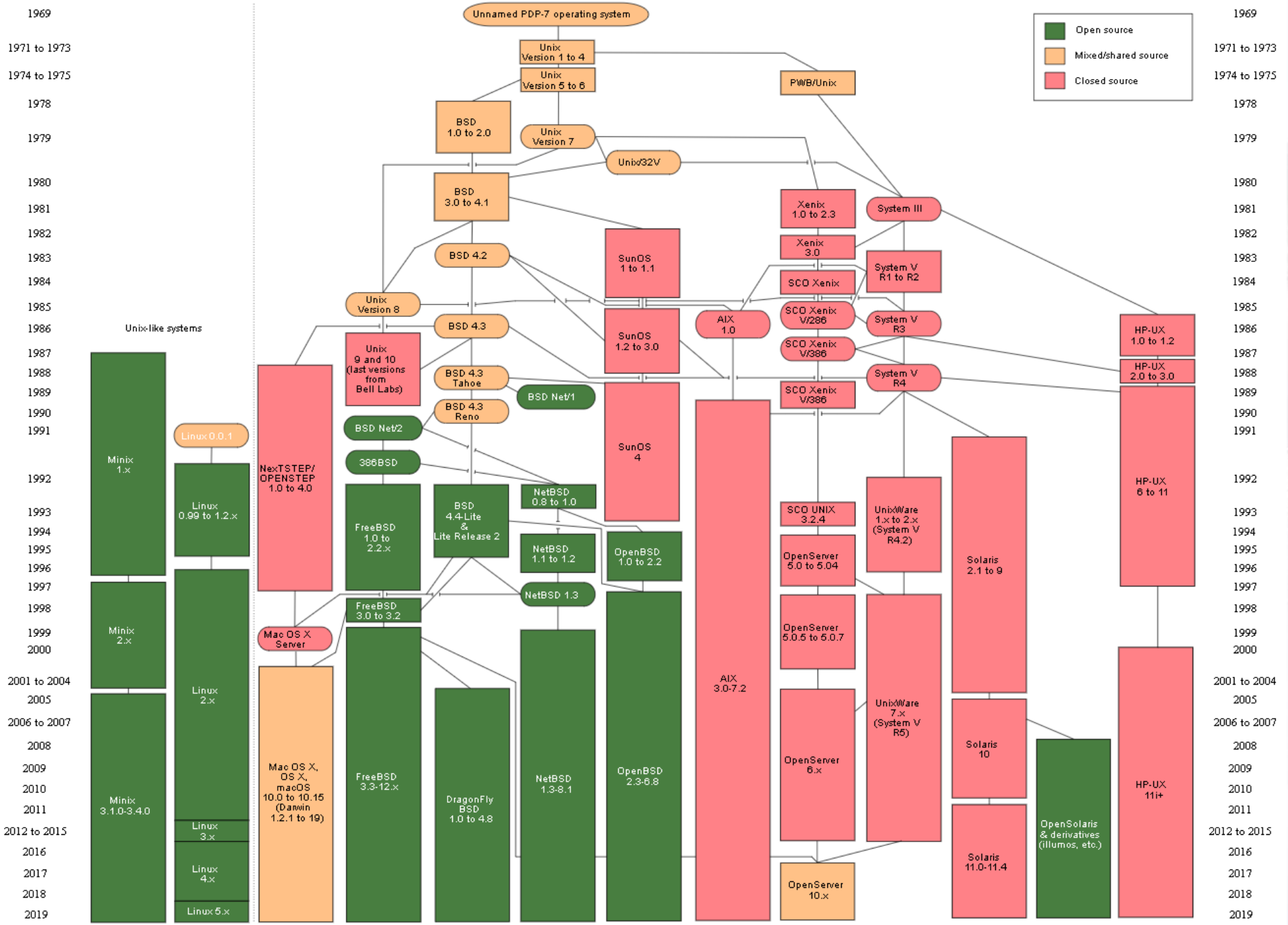
Media

- AI Draw
- Image/Video Correspondence matching

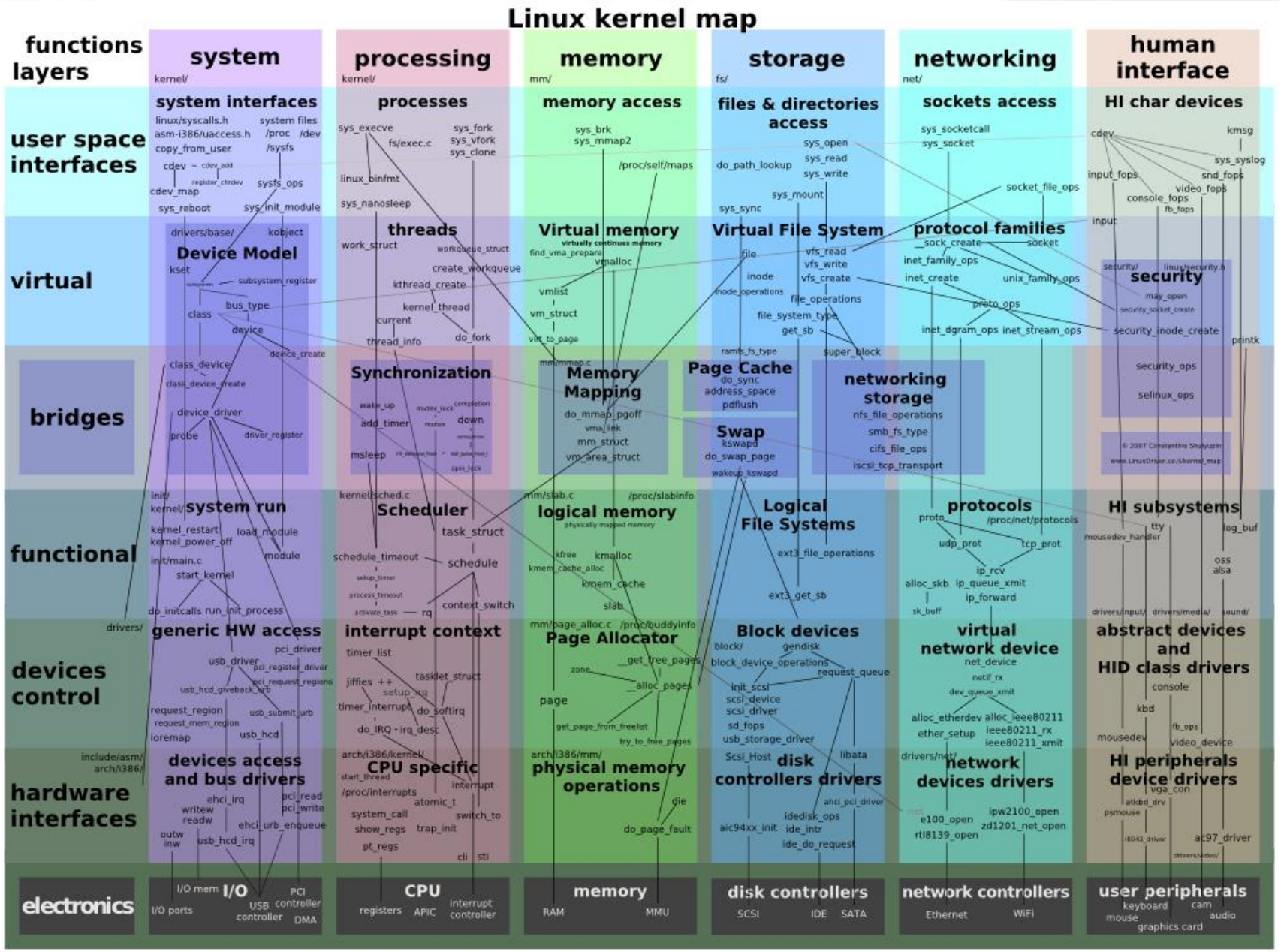
Operating System == OS

- An operating system (OS) is system software that manages computer hardware, software resources, and provides common services for computer programs.
- Unix, Linux, BSD (Free BSD, Open BSD), Android (+vendor), iOS (macOS), Zephyr, etc.
- OS == Kernel + ?

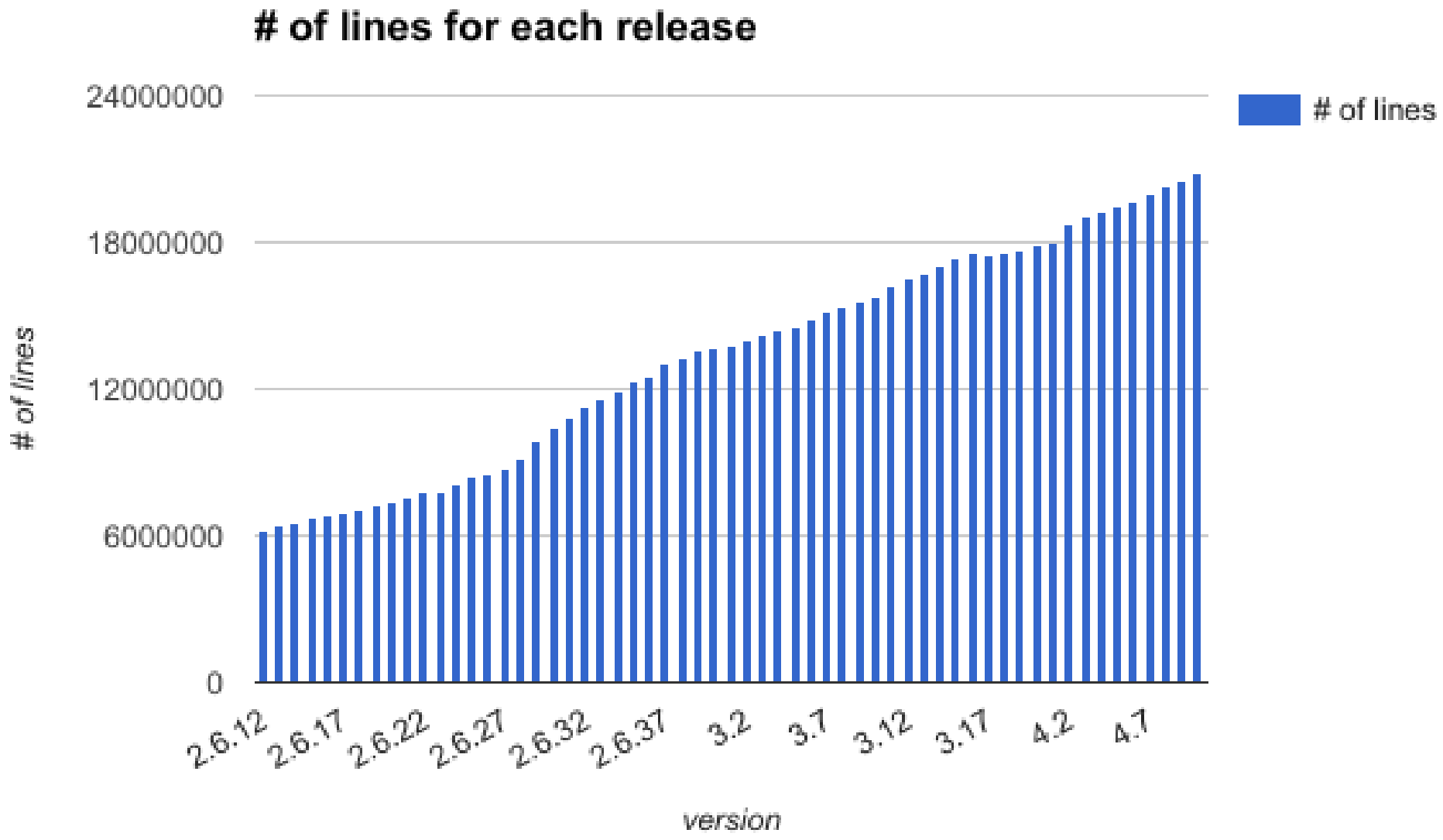



Open source
Mixed/shared source
Closed source
Unix-like systems
1969
1971 to 1973
1974 to 1975
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001 to 2004
2005
2006 to 2007
2008
2009
2010
2011
2012 to 2015
2016
2017
2018
2019
Unnamed PDP-7 operating system
Unix Version 1 to 4
Unix Version 5 to 6
Unix Version 7
PWB/Unix
BSD 1.0 to 2.0
Unix/32V
BSD 3.0 to 4.1
Xenix 1.0 to 2.3
System III
Xenix 3.0
BSD 4.2
SunOS 1 to 1.1
System V R1 to R2
SCO Xenix
Unix Version 8
SCO Xenix V/286
AIX 1.0
System V R3
BSD 4.3
SunOS 1.2 to 3.0
HP-UX 1.0 to 1.2
SCO Xenix V/386
HP-UX 2.0 to 3.0
Unix 9 and 10 (last versions from Bell Labs)
BSD 4.3 Tahoe
BSD Net/1
System V R4
BSD 4.3 Reno
SCO Xenix V/386
Minix 1.x
Linux 0.0.1
BSD Net/2
NextSTEP/ OPENSTEP 1.0 to 4.0
386BSD
SunOS 4
HP-UX 6 to 11
Linux 0.99 to 1.2.x
FreeBSD 1.0 to 2.2.x
BSD 4.4-Lite & Lite Release 2
NetBSD 0.8 to 1.0
SCO UNIX 3.2.4
UnixWare 1.x to 2.x (System V R4.2)
Solaris 2.1 to 9
NetBSD 1.1 to 1.2
OpenBSD 1.0 to 2.2
OpenServer 5.0 to 5.04
Minix 2.x
NetBSD 1.3
FreeBSD 3.0 to 3.2
OpenServer 5.0.5 to 5.0.7
Linux 2.x
Mac OS X Server
AIX 3.0-7.2
UnixWare 7.x (System V R5)
Solaris 10
OpenServer 6.x
Minix 3.1.0-3.4.0
Mac OS X, OS X, macOS 10.0 to 10.15 (Darwin 1.2.1 to 19)
FreeBSD 3.3-12.x
DragonFly BSD 1.0 to 4.8
NetBSD 1.3-8.1
OpenBSD 2.3-6.8
OpenSolaris & derivatives (illumos, etc.)
HP-UX 11i+
Linux 3.x
Linux 4.x
Linux 5.x
Solaris 11.0-11.4
OpenServer 10.x

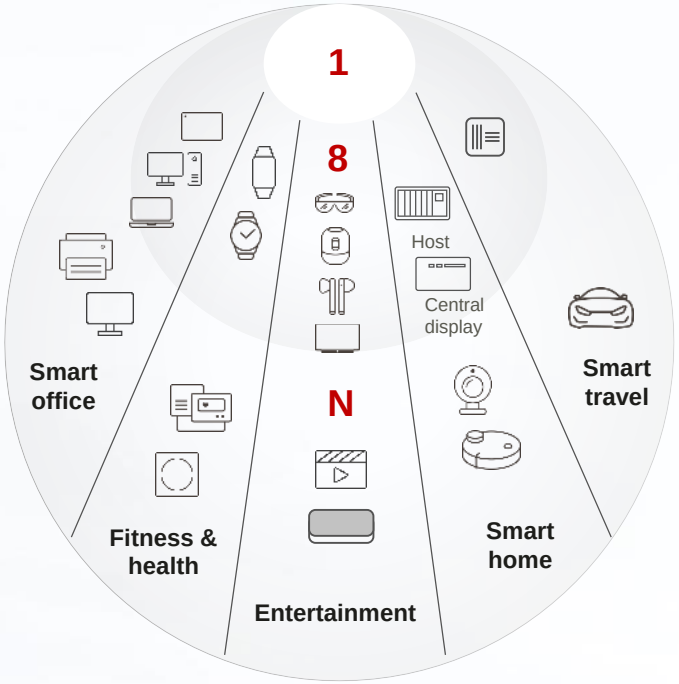
Linux kernel complexity



Linux kernel LOC



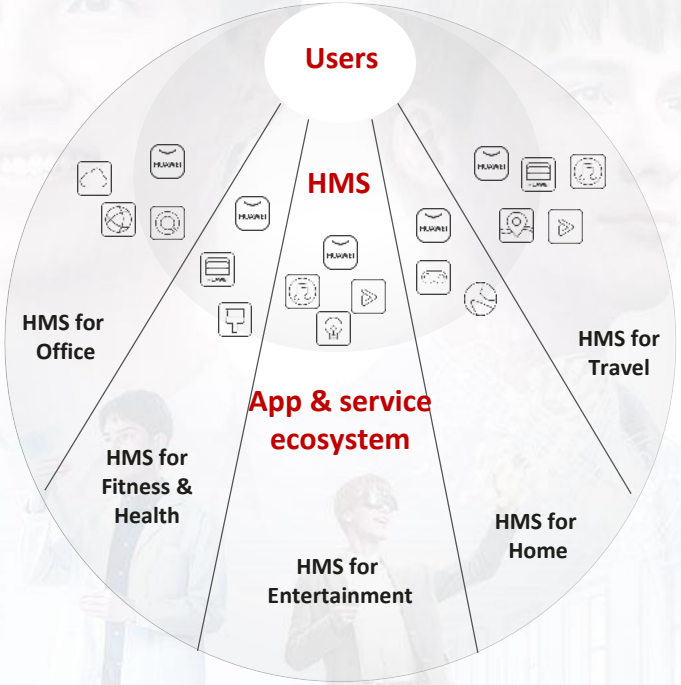
Huawei's device business strategy for the next 5 years: Building an open ecosystem around 5 major scenarios



Hardware-software integration:
Super Devices targeting 5 major scenarios

Hardware ····· Software ····· Cloud

HarmonyOS



Cloud-device synergy: Apps & services targeting
5 major scenarios

HarmonyOS ecosystem: A new all-scenario intelligent ecosystem covering HarmonyOS, HarmonyOS Connect, and OpenHarmony.

Establishing HarmonyOS

- HarmonyOS provides a common language for different kinds of devices to connect and collaborate, creating a simple, smooth, continuous, secure, and reliable experience across all scenarios.
- HarmonyOS continuously establishes itself as a leading brand in tech innovation, driving sales and a thriving ecosystem.

HarmonyOS

Different devices, same system

Flexibly deployed on devices with **128 KB to 1 GB+ RAM**

A common language for **220 mn+** major products

…Shipments of HarmonyOS Connect devices have exceeded **100 mn units**

…

Super Device integrating all hardware

Easily drag icons to connect devices and form a Super Device

Multi-cam model

Distributed file system

Task Center allows apps to travel between different devices

…

Smart services, quality ecosystem

Service Widgets: Readily accessible services

MeeTime: Seamless transfers

Atomic Services: Easily called on and shared without installation

…

hardware + software (OS&MW) == Super Device(s)

Smart office

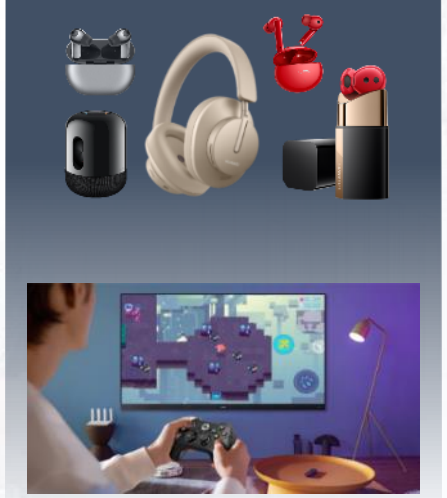
Device collaboration, ecosystem integration, free creation, and boundless communication

Fitness & health

Scientific guidance for exercises & proactive health management

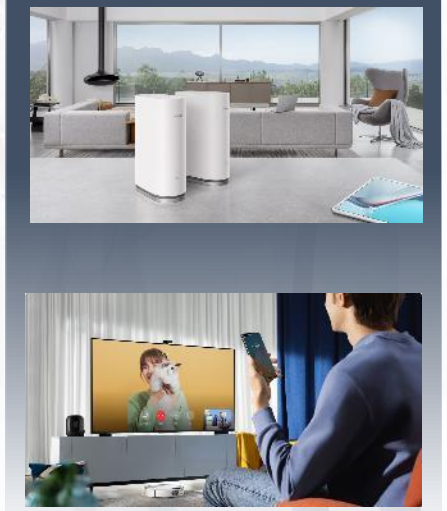
Entertainment

All-in-one subscription package and extraordinary entertainment experience that supports diverse interactions



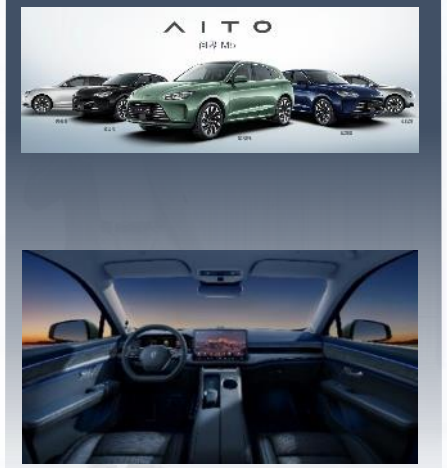
Smart home

Whole-house intelligence powered by AI and interconnectivity

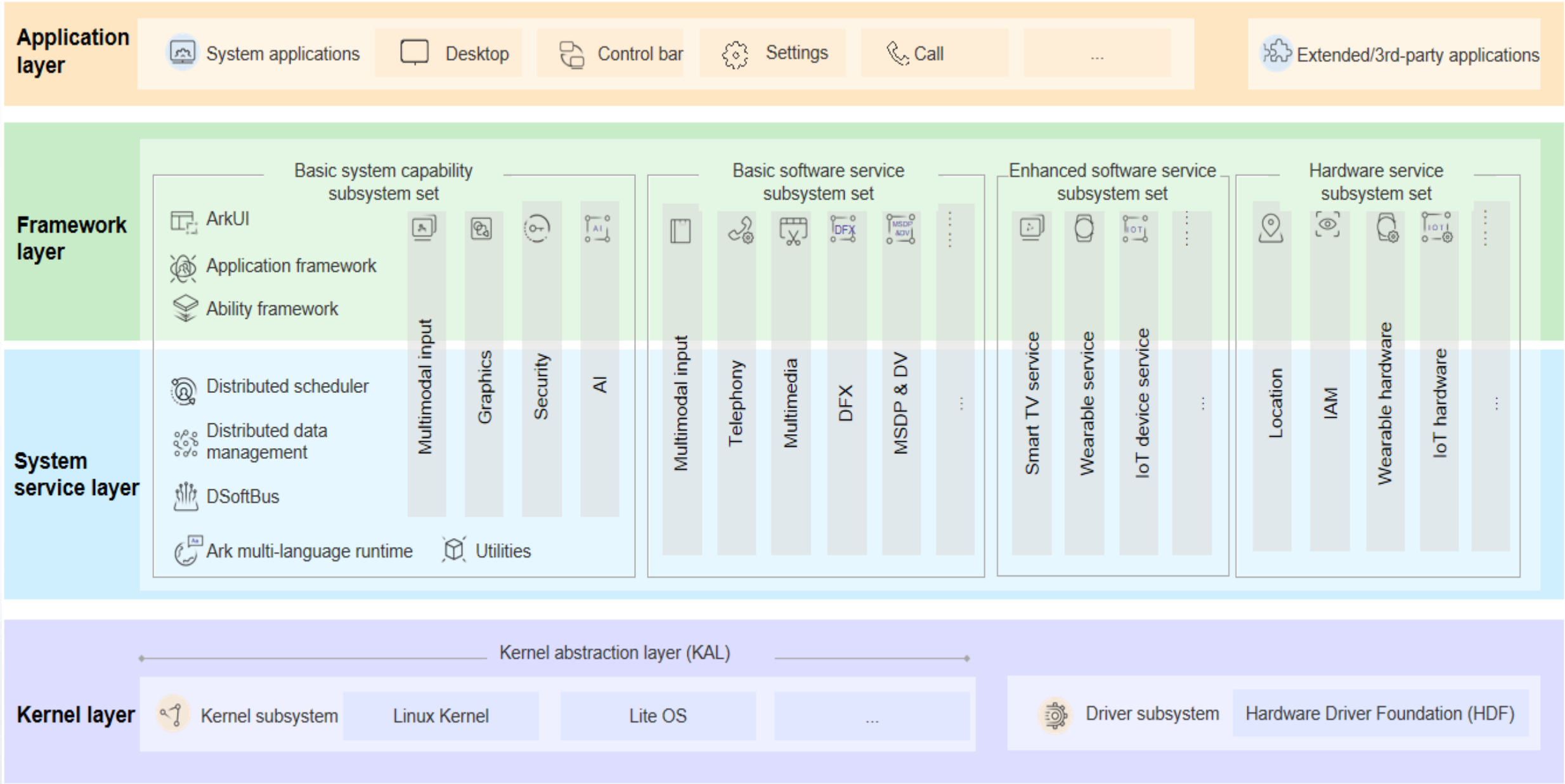


Smart travel

Multi-device collaboration enables new convenient and intelligent experiences



Open Harmony



Kolmogorov complexity

- Let a string s has description d(s) of minimal length
- the length of d(s) is the Kolmogorov complexity of s, written K(s).
- K(s) = |d(s)|

Kolmogorov complexity and data compression

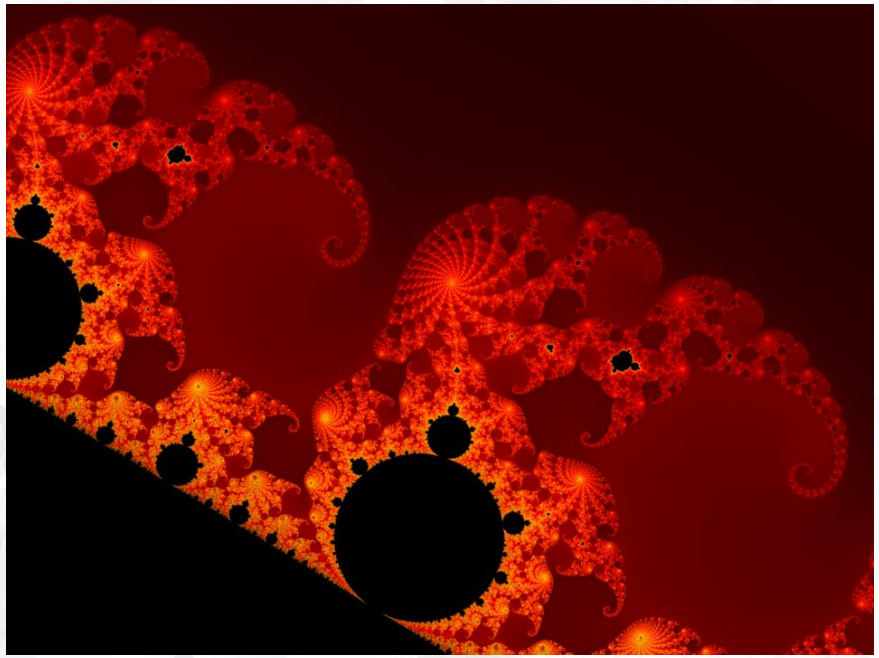
- abababababababababababababababab
- 4c1j5b2p0cv4w1x8rx2y39umgw5q85s7

- "write ab 16 times“: K(s) = 17

- "write 4c1j5b2p0cv4w1x8rx2y39umgw5q85s7“: K(s) = 38

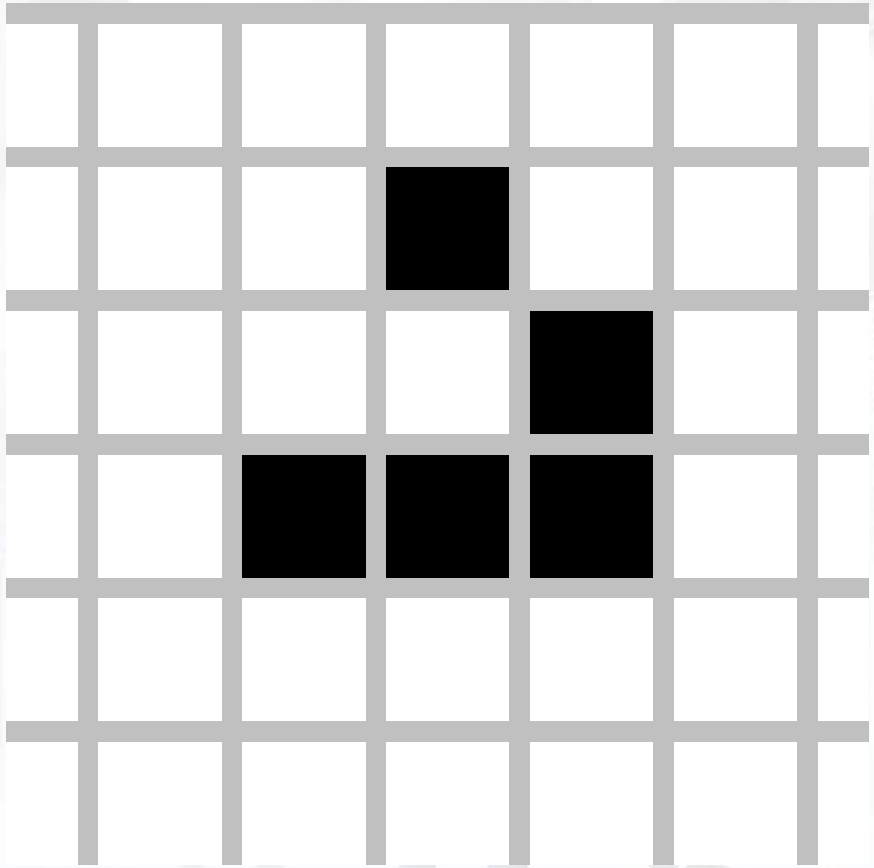
Mandelbrot set

- $z = x + iy$
- $z2 = x2 + i2xy - y2$
- $c = x0 + iy0$
- $zn+1 = zn2 + c$
- $x = Re(z2 + c) = x2 - y2 + x0$
- $y = Im(z2 + c) = 2xy + y0$



Cellular Automata: Conway's Game of Life

- Any live cell with two or three live neighbors survives
- Any dead cell with three live neighbors becomes a live cell
- Play on the web



Self-organization, chaos, complex systems (OS)

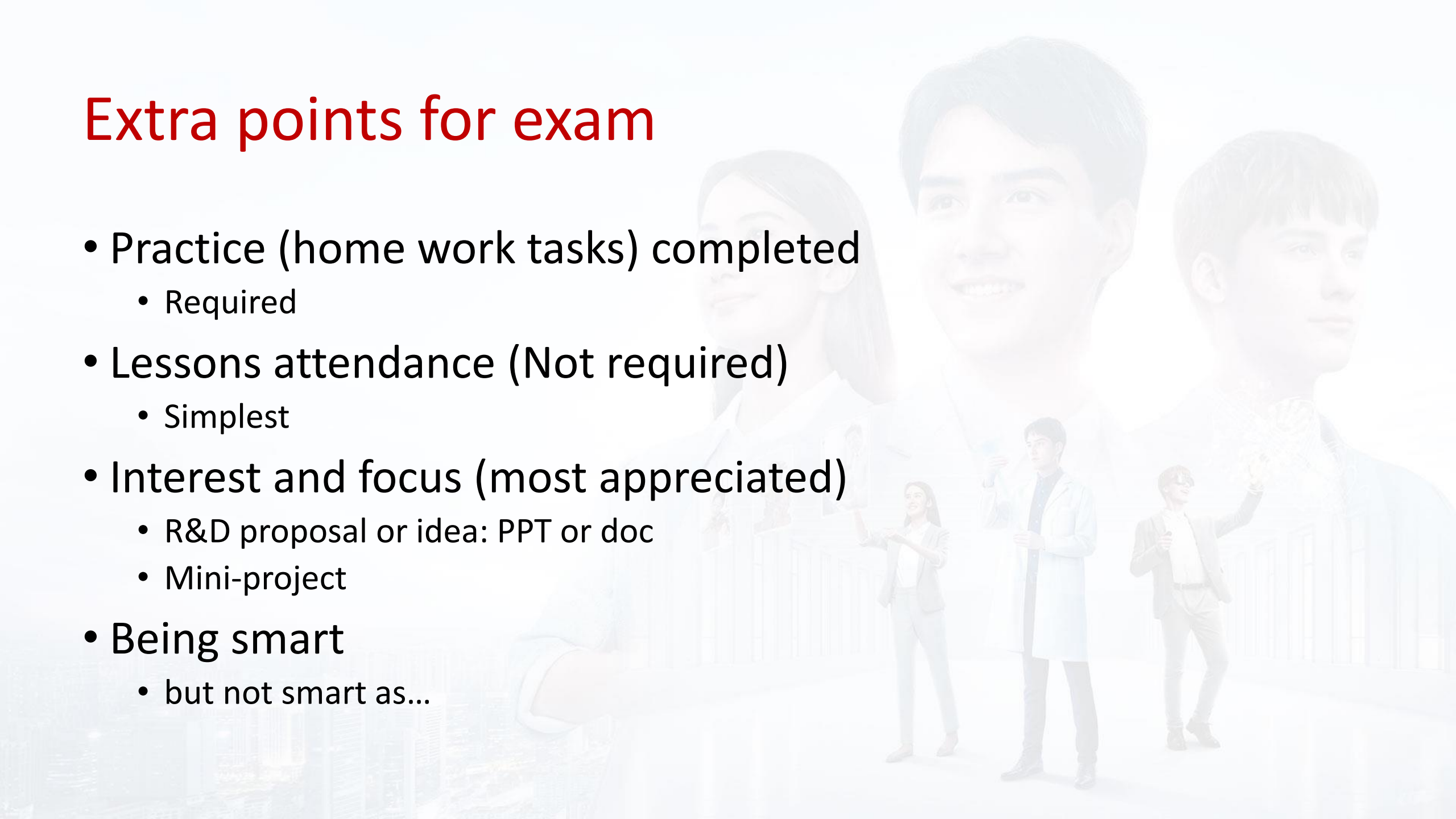
- Rayleigh–Bénard convection
- 2nd law of thermodynamics
- Entropy rearrangement

Hints to succeed in studying this course

- Get motivated
  - Emotions, goals and reasons
- Prioritize
  - What to do in what order
  - Most urgent or most important?
- Keep focus
  - Social networks and attention focus
  - Will power “battery” discharge
- Use or lose
  - repeat and practice

Extra points for exam

- Practice (home work tasks) completed
  - Required
- Lessons attendance (Not required)
  - Simplest
- Interest and focus (most appreciated)
  - R&D proposal or idea: PPT or doc
  - Mini-project
- Being smart
  - but not smart as…



Вопросы на экзамен

- Определение операционной системы. Отличие между ОС и ядром ОС.
- Базовые понятия и концепции ОС
- Общая архитектура ОС
- Системные вызовы
- Особенности параметризации системных вызовов
- Режимы (modes) исполнения в ОС
- Пространства памяти в ОС
- Монолитное и микро-ядро ОС – различия
- Модульная структура я дра ОС
- Реализации мульти-обработки в ОС
- Различие между кооперативным и вытесняющей (preemptive) мульти-обработкой
- Кооперативные ядра (на примере Линукса)
- Вытесняющие ядра (на примере Линукса)
- Ассиметричная мульти-обработка (Asymmetric multi-processing)
- Симетричная мульти-обработка (symmetric multi-processing)
- Масштабировяния ядра ОС по процессорам
- Адресные пространства памяти – физическое и виртуальное
- Таблицы трансляции виртуальных адресов
- Контекссты исполнения
- Стеки пользовательского кода, кода яядра и кода прерываний
- Страничная органицация памяти и вытеснение страниц на диск
- Общая структура кода ядра Линукса
- Управление процессами в ОС, Управление памятью в ОС
- Виртуальная файловая система и управление блочным вводом-выводом
- Сетевой стэк в Линуксе
- Драйверы устройств в архитектуре Линукса
- Процессы и потоки
- Переключение контекста и миграция задач по ядрам процесссора
- Контекст процесса
- Доступ к текущему процессу
- Блокирование и пробуждение
- Вытеснение задач в терминах контекста процесса
- Системный вызов clone()
- Пространства имен и контейнеры
- Прерывания (Interrupts)
- Исключительные ситуации (exceptions)
- Аппаратная концепция прерываний
- Программируемый контроллер прерываний
- Обработка прерываний в Линуксе
- Контекст прерывания
- Отложенные действия (deferrable actions)
- Мягкие запросы на прерывания (Soft IRQ)
- Тасклеты (Tasklets)
- Рабочие очереди и таймеры
- Симметричная мульти-обработка

Вопросы на экзамен

- Синхронизация – основные идеи и проблемы
- Проблема состояния гонки (race condition)
- Как избегать состояния гонки
- Атомарные операции
- Спинлоки
- Когерентность кэша в многопроцессорных системах
- Протоколы когерентности кэша
- Вытеснение используемых данных кэша (cache trashing)
- Синхронизация доступа к данным из контекста процесса и контекста прерывания
- Мьютексы
- Данные доступа на одном ядре (per CPU data)
- Упорядочивание доступа к памяти и барьеры
- Чтение-копирование-обновление (Read-Copy-Update на примере списка)
- Что такое управление памятью (memory management)
- Виртуальные и физические адреса
- Устройство управления памятью (MMU)
- Буфера кэширования трансляции адресов (TLB)
- Адресные пространства на примере архитектуры ARM
- Количество бит в адресе и деления адресов
- Линейное отображение адресов
- Нелинейное (произвольное) отображение адресов
- Фиксированное отображение линейных адресов
- Временное/быстрое отображение адресов (страницы в ядре)
- Постоянное отображение адресов
- Управление физической памятью
- Зоны памяти
- Неоднородрый и однородный доступ к памяти
- Кэш страниц в файловом доступе
- Выделение физической памяти
- Buddy алгоритм выделения памяти
- Подход к SLAB выделению маленьких фрагментов памяти
- Реализация SLAB
- Кэши и SLAB
- kmalloc() & kfree()
- Управление виртуальной памятью
- Анонимная память (анонимное отображение памяти)
- Переиспользование (reclaim) памяти
- Дефрагментация (Compaction) памяти
- Обработка нехватки памяти (Out Of Memory Killer)
- Обработка обращения к отсутствующес странице (page fault)
- Типы/виды page fault
- Влияние page fault на производительность

Вопросы на экзамен

- Абстракции файловой системы
- Пример простой файловой системы (структура на диске)
- Операции файловой системы
- Кэширование структур данных файловой системы
- Санитайзеры программ
- Санитайзеры от Гугла
- Санитайзеры EFENCE, KFENCE
- Санитайзер KASAN
- Теневая память в KASAN
- Красные зоны в KASAN
- Сетевая функциональность – история и концепции
- TCP/IP - коротко
- Семейство протоколов в TCP/IP
- Протокол ARP
- Сетефой стек – диаграмма и уровни
- Единицы данных в сетевых протоколах
- Блок-схема: Передача – прием данных в сети
- Сетевые устройства
- IP сервисы: роутинг
- Сокеты
- Пример использования сокетов для клиент-сервер программ
- Распределенные системы
- Оверлейные сети
- Понятие middleware
- ОС и middleware
- Цели дизайна распределенной ОС
- Ошибочные предположения в дизайне распределенной системы
- Высокопроизводительные распределенные системы
- Кластерные архитектуры
- Grid – архитектуры
- Облачные архитектуры
- Проникующие системы (pervasive systems)
- Вездесущие (Ubiquitous) системы
- Мобильные ad hoc системы
- Концепции архитектуры распределенных систем
- Архитектурные стили для распределенных систем
- Обьектные архитектуры
- Распределенные и удаленные обьекты
- Ресурсно-ориентированные архитектуры
- Архитектура Издатель-Подписчик

Вопросы на экзамен

- Структурированные peer-to-peer системы
- Неструктурированные peer-to-peer системы: обмен сообщениями
- Иерархические peer-to-peer системы
- Планирование ресурсов в ОС
- Планировщик процессов
- Типы планировщиков (по горизонту планирования)
- Диспетчер процессора/процессов
- FIFO планировщик
- Метрики планирования в мобильной ОС
- Абстракции/концепции планирования
- Абстракция/концепция аппаратуры/»железа»/устройства для планирования
- Абстракция/концепция задачи/процесса для планирования
- Абстракции управления потрелением энергии
- История планироващиков в ядре Линукс
- Планировщик CFS
- Планировщик процессоров/процессов и формулы потребления энергии
- Потребление энергии процессором/памятью
- Что такое DVFS, зачем оно нужно
- Архитектура big.LITTLE в контексте планирования процессов
- EAS – что это такое, основные принципы
- Отслеживание нагрузки для планирования процессов
- EAS PELT/WALT
- C & P, CC & PC состояния процессора
- Управление P состояниями со стороны ОС
- Масштибирование производительности в ядре Линукс
- Управление частотами в ядре Линукс
- Что такое CPUFreq и гувернеры
- Что такое CPUIdle и гувернеры
- Основы программирования для мобильной ОС
- Что такое ADB и как им пользоваться
- Измерение производительности приложения в инструкциях и циклах
- Что такое perf и как им пользоваться