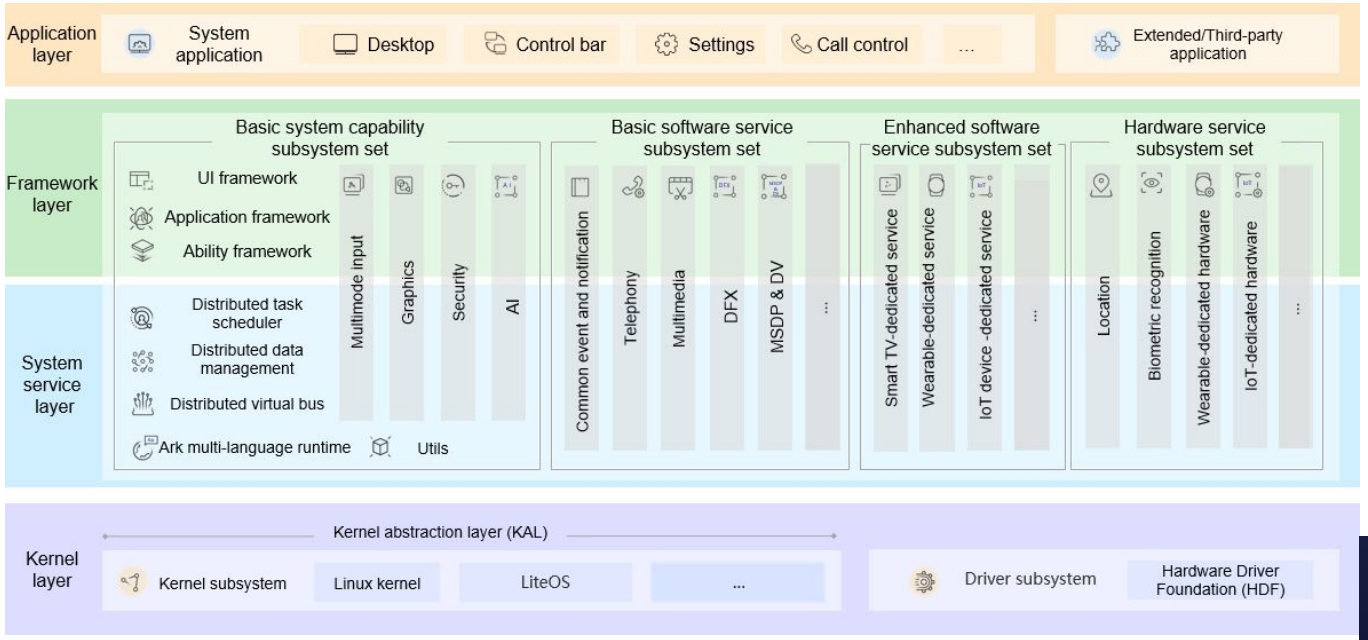
# HarmonyOS Distributed Architecture

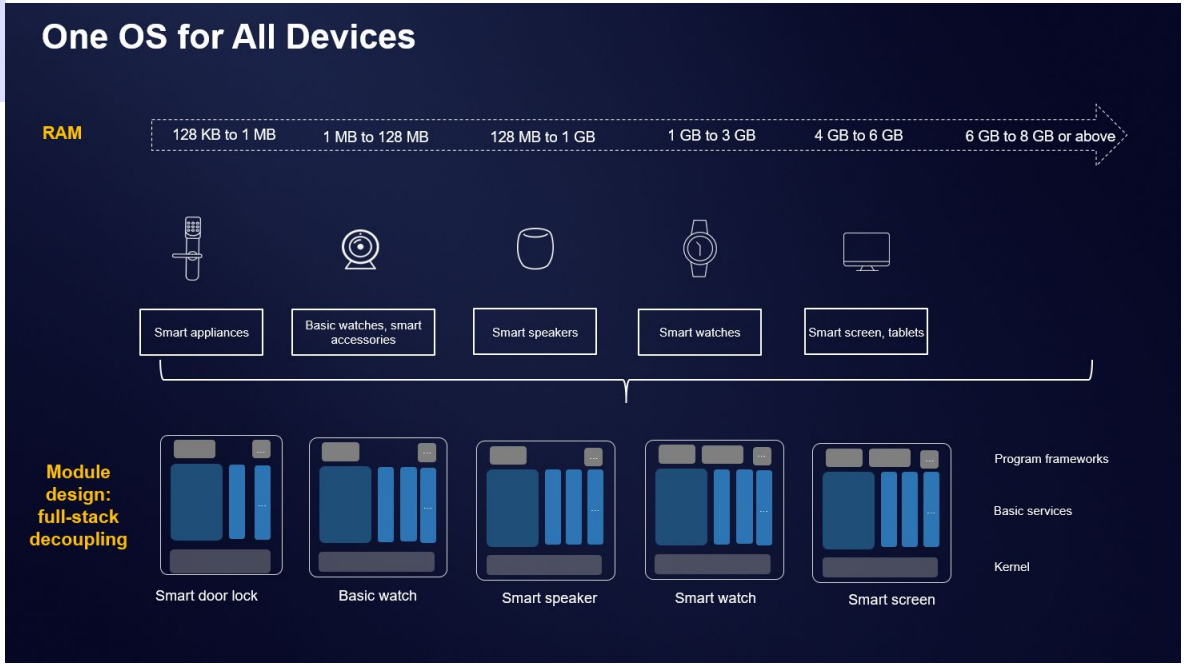
- Nagesh S

# Index

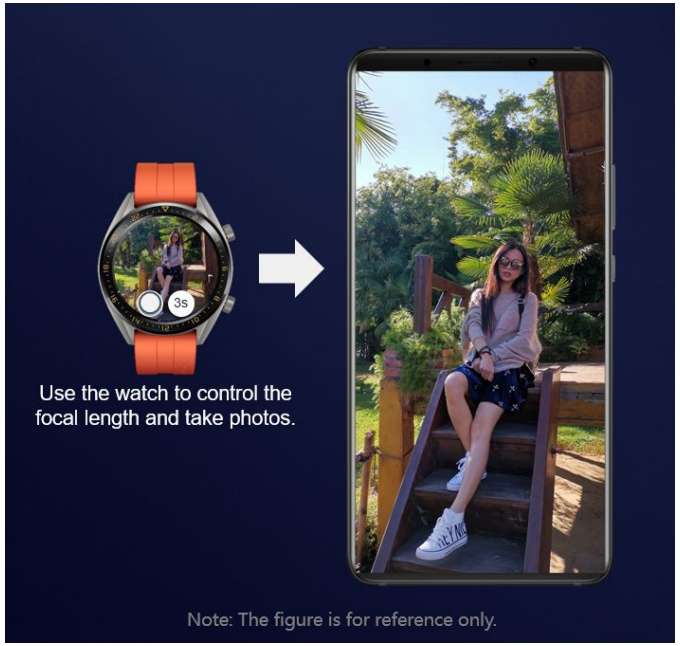
- HarmonyOS Architecture
- Distributed usecases
- Hardware Collaboration and Resource Sharing
  - Distributed Virtual Bus
  - Distributed Device Virtualization
  - Distributed Data Management
  - Distributed Task Scheduling

# HarmonyOS Architecture

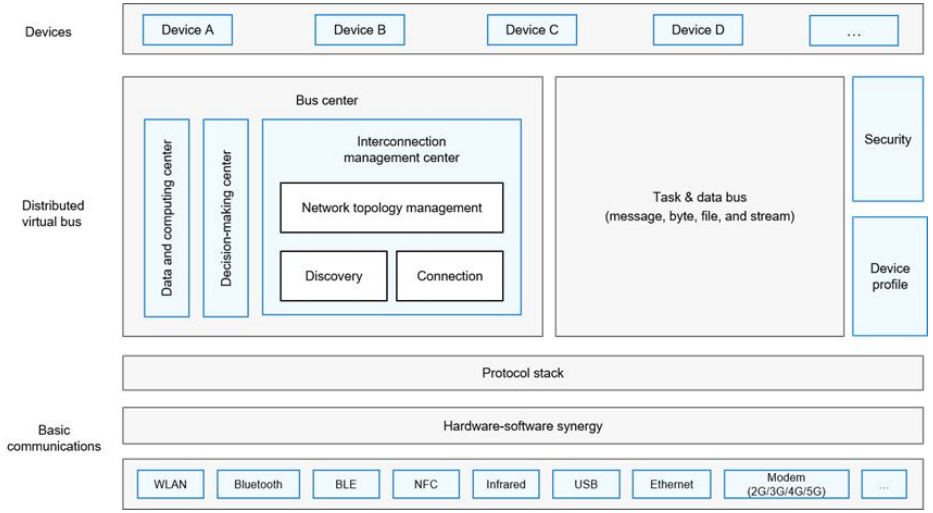


# Distributed Usecases

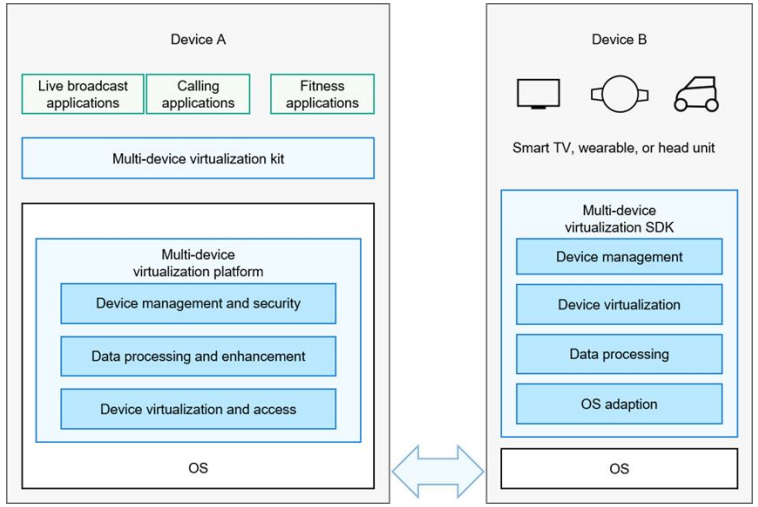



# Distributed Soft Bus Technology



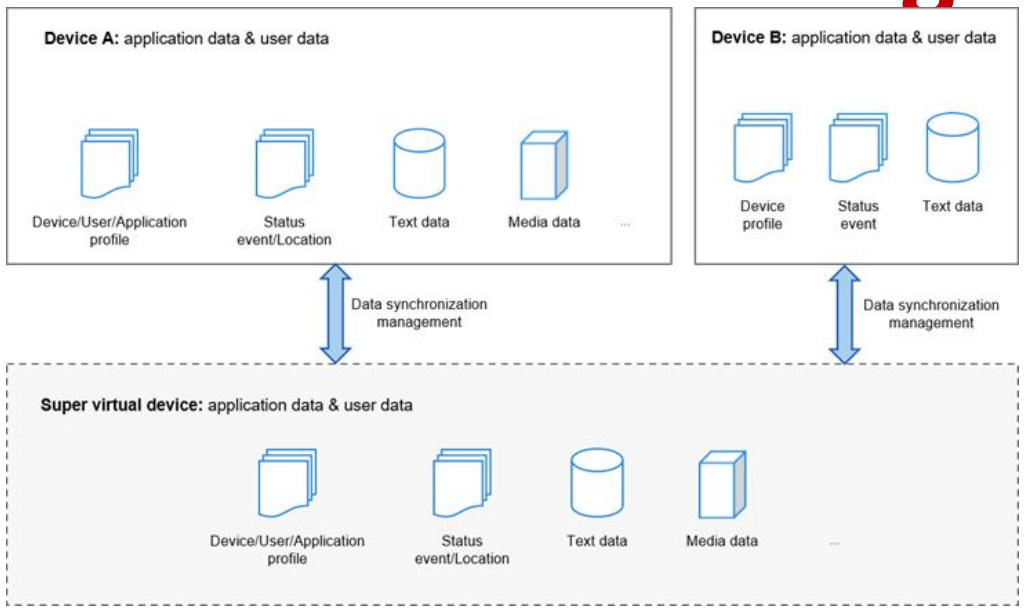

- Provides unified **distributed communication capabilities** for seamless interconnection of “1+8+N” devices.
- Enables quick discovery and connection so that programs and data can be efficiently transferred. Can achieve **zero-wait transmission** among devices.
- Provides **intra-device and inter-device communication capabilities** which are secure, reliable, high bandwidth, low latency, low power consumption.
- Has 3 main functionalities: Task bus, Data bus and Bus center.
  - **Task bus** is responsible for quickly distributing applications on multiple devices.
  - **Data bus** distributes and synchronizes data among devices with high performance.
  - **Bus Center** is used for coordination and control, automatic discovery and networking, and maintaining topology relationships between devices

# Distributed Device Virtualization Technology



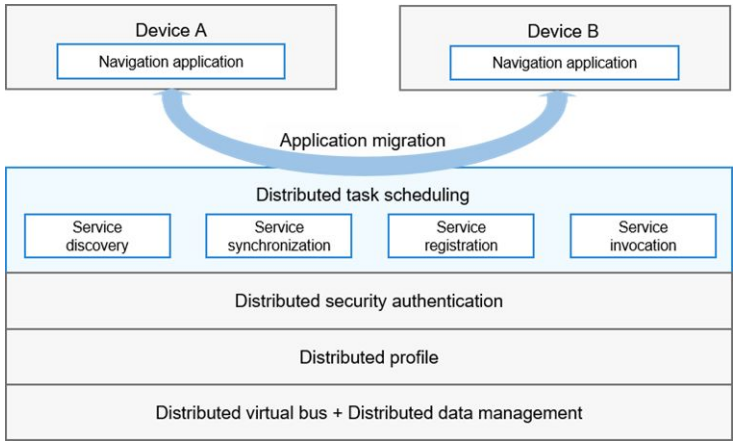

- Enables cross-device resource convergence, device management, and data processing so that multiple devices jointly function as a **super virtual device**.
- Achieves this by **virtualizing the peripherals of a hardware device as resources** that can be used on another device.
- This technology turns hardware devices into **multiple peripheral resource pools** that can be shared by multiple devices. For example, a mobile phone can use the camera of a smart screen, and the smart screen can use the phone's AI capabilities.
- This includes two key technologies: Device virtualization, and audio and video synchronization.
  - **Device Virtualization** allows remote peripheral capabilities to be used like local ones.
  - **Audio and video synchronization** ensures low latency of audio and video transmissions between two devices.

# Distributed Data Management



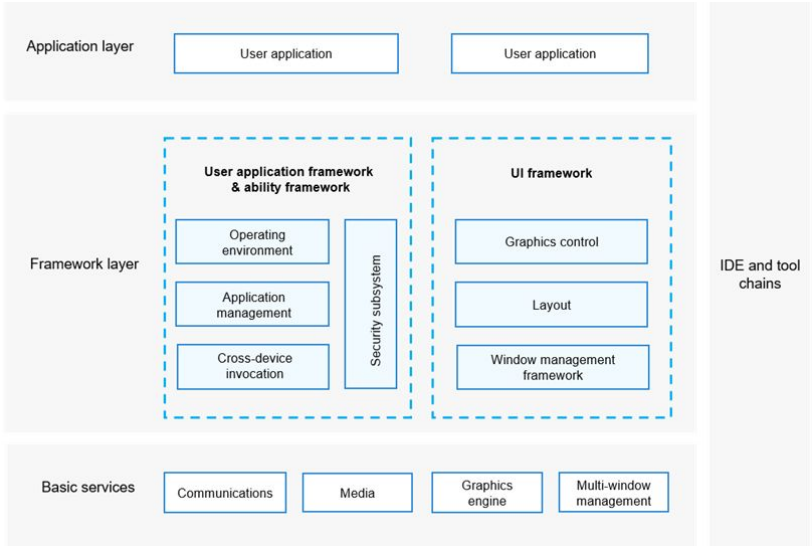

- Leverages distributed virtual bus to manage application data and user data distributed on different devices. Enables **separation** of service logic from data storage.
- Provides applications with the capability of **distributing database data among** different devices.
- Provides **distributed data interface** so that applications can save data to the distributed database.
- **Isolates applications** by account, application and database to ensure that data in one application cannot be accessed by another application through the service.
- Supports **application data synchronization** between trusted and authenticated devices, providing consistent data access experience on multiple devices.

# Distributed Task Scheduling




- Based on Distributed virtual bus, Distributed data management and Distributed profile.
- It builds a **unified distributed service management** mechanism (including service discovery, synchronization, registration and invocation)
- Implements Remote startup, Remote invocation, Remote connection and Migration of applications across devices

# One-off development for Multi-Device Deployment




- This is an **integrated application development solution** for the distributed device environment and simplifies the workload of cross-device application development.
- Applications can be seamlessly deployed to multiple running environments after **being developed for just once**.
- Provides system-level distributed capability abstraction based on an application’s service logic and UI logic, achieving hardware collaboration and ecosystem sharing at low costs.
- Consists of the distributed program & Ability framework and the UI framework.

# Adaptive UX with One-Off Development for Multi-Device Deployment

- Allows an interface logic to be reused on multiple devices, and adapts the screen shape and size with technologies such as **polymorphic control, dynamic layout, and project template**.
- Enables UI development for different displays, platforms, and interaction modes for the sake of user experience. Thus developers only need to develop once, and the UI can adapt itself to each device, and the information structure can be optimally displayed on different devices.
- The **distributed UX framework** extracts common information structures from different dimensions, such as UI elements, content frameworks, and page frameworks, and abstracts component capabilities at different granularities.
- Capabilities such **as On-demand component selection**, **Mutable function set configuration** and **Associative inter-component dependencies** can be achieved.

# Thank You