



# **Transparent Computing: Opportunities and Challenges**

**Yaoxue Zhang**

**zyx@moe.edu.cn**

**Tsinghua University**

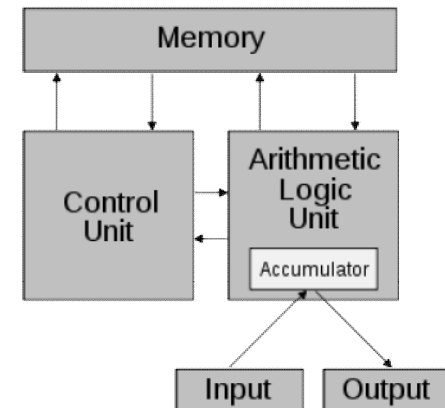


# Agenda

- What is the problem?
- What is Transparent Computing?
- Key technologies and issues
- Sample applications
- Vision

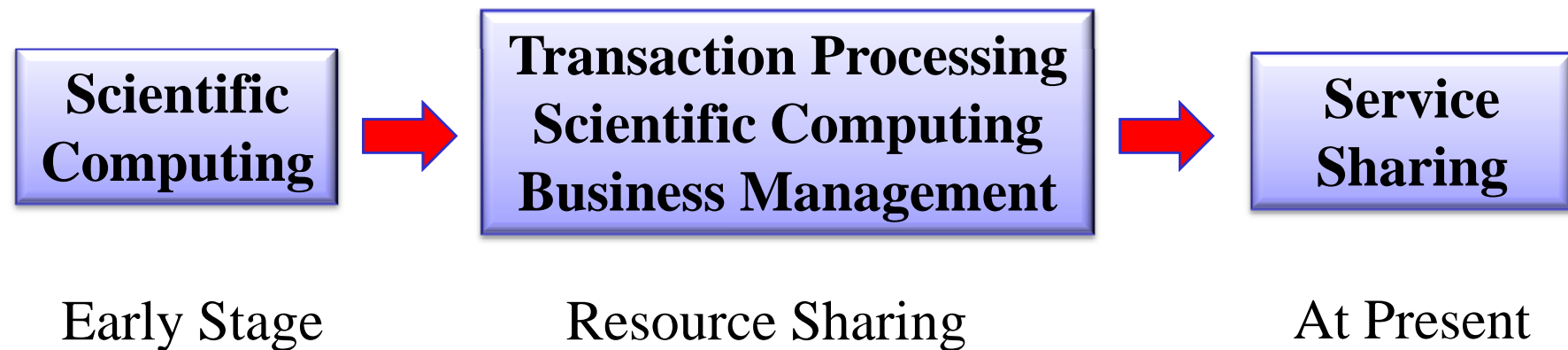
# 1. What is the problem?

- Computer architecture and Operating Systems have not been changed for many years
  - Computer Architecture
    - Von Neumann Architecture
  - Operating Systems
    - Designed for the Von Neumann architecture to manage hardware and software resources, including
      - Process/threads, storage, I/O, file, interrupt/trap ...



*“Store its instructions in its internal memory and process them in its arithmetic unit, so that in the course of a computation they may be not just executed but also modified at electronic speeds”*

# User's requirements for computers are changing rapidly

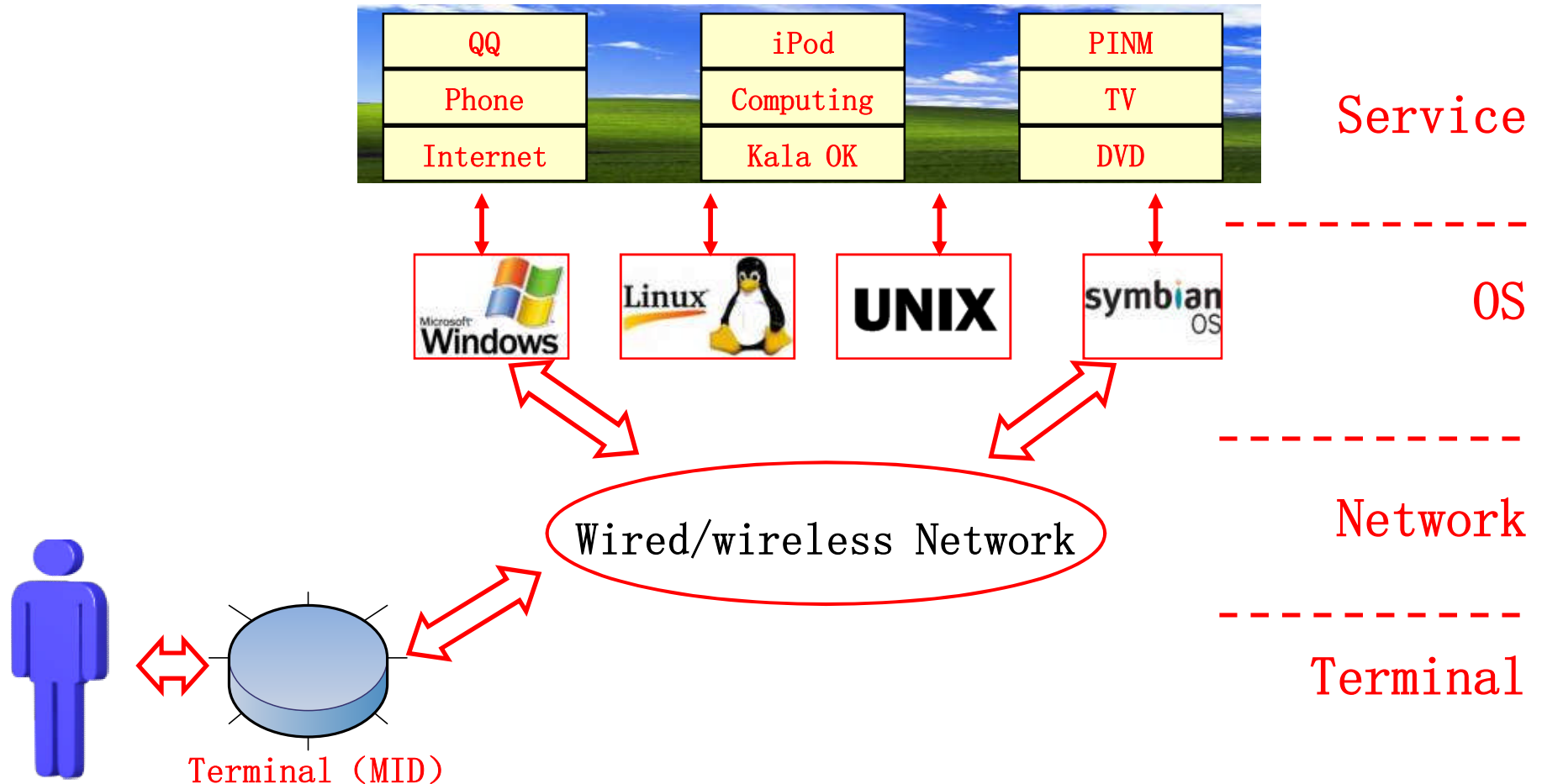




# What is Service Sharing?

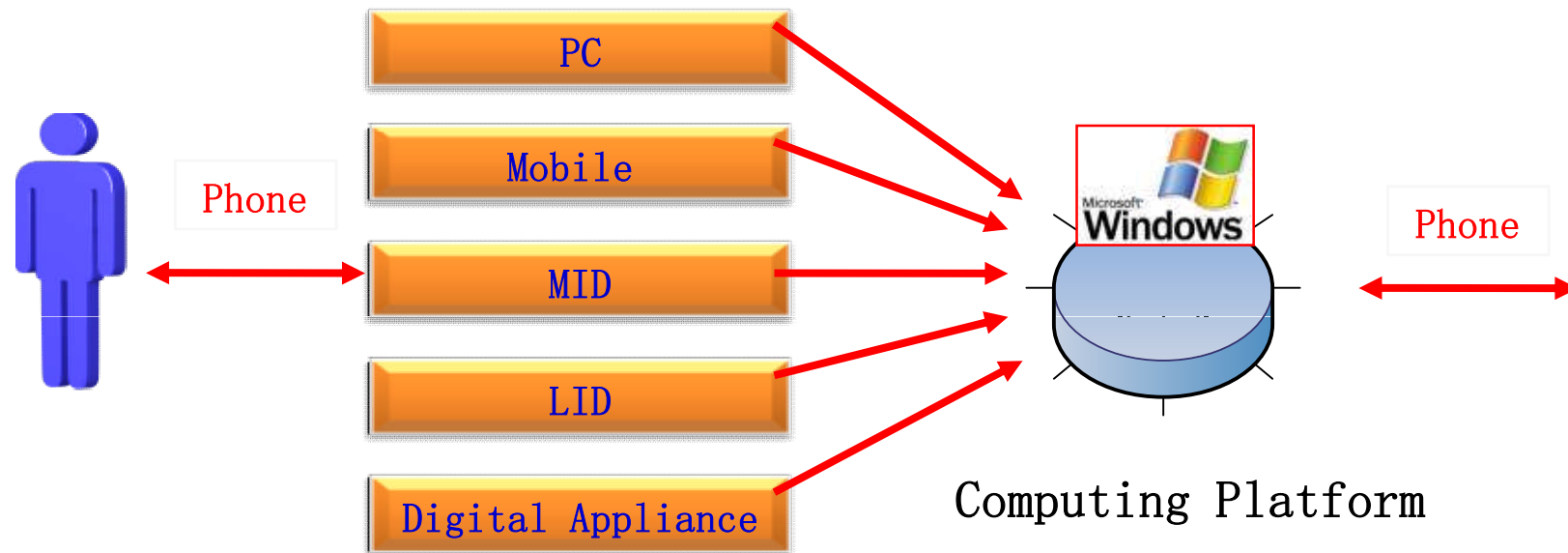
- Users don't care how computers work internally, what they care is the certain tasks (services) that they want computers deliver
  - Personalized
  - User friendly
  - Trusted
  - Environment independent

# One Hardware, Different Services



One Hardware, Different OSes, Different Services

# One Service, Different Hardware



Different Hardware, One OS, One Service

# To support Service Sharing

- Small systems
  - Infrequent or no upgrading
  - Support multi-OSes
  - Trusted
  - Support different network connections
  - Easy to maintain
- ➡ **New computing paradigm: Extend and change the architecture and OS**
- ➡ **Transparent Computing**





# Agenda

- What is the problem?
- What is Transparent Computing?
- Key technologies and issues
- Sample application
- Vision

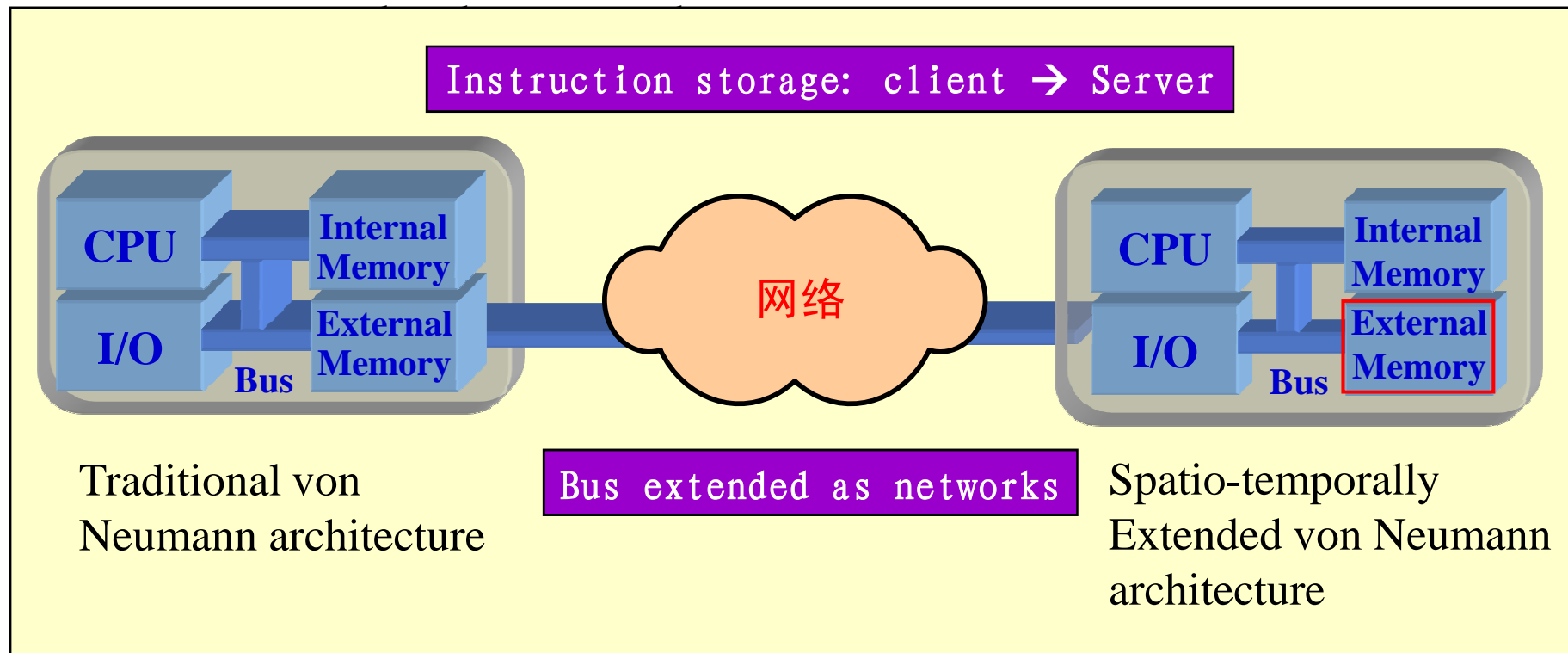


# What is Transparent Computing

- A fundamental paradigm shift from traditional solutions
  - System Architecture
    - Spatio-temporally Extended von Neumann Architecture, consider terminal, network and server together
  - Computing Model
    - Separate computing and storage: store program in server, execute program in client
    - Distributed deployment, dynamic scheduling and execution
  - Management
    - Centralization: manage terminal, network and server altogether
    - Service-oriented: OS as a service
      - Run traditional OSes compatibly
      - Choose any OS and applications according to service

# What is Transparent Computing

- A fundamental paradigm shift from traditional solutions
  - System Architecture
    - Spatio-temporally Extended von Neumann Architecture, consider terminal,



- 
- New OS underlying Extended architecture

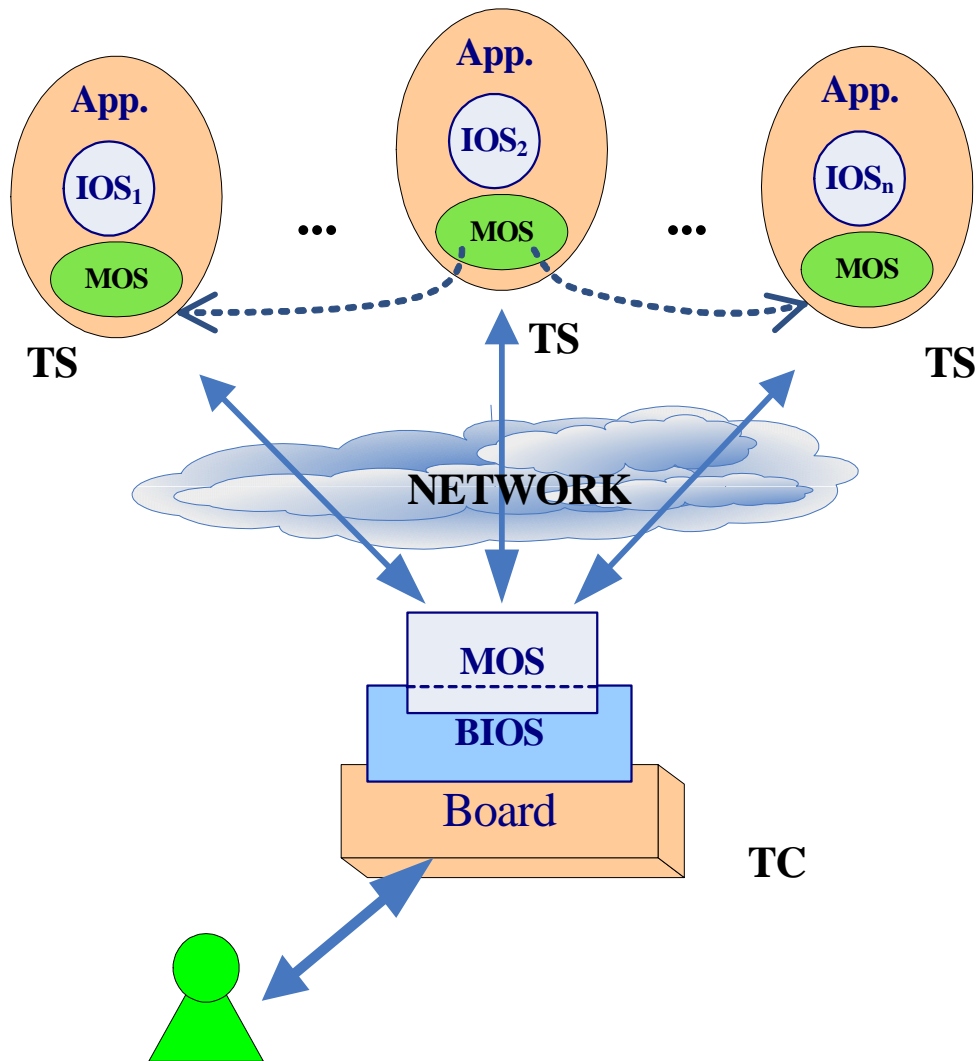
**Instance OS (IOS):** Traditional OSes: Windows, Linux, UNIX, Symbian, etc.

**Meta OS (MOS):** OS for managing IOSes and other specified networked resources

**TS:** Transparent Server

**TC:** Transparent Client

# IOS and MOS

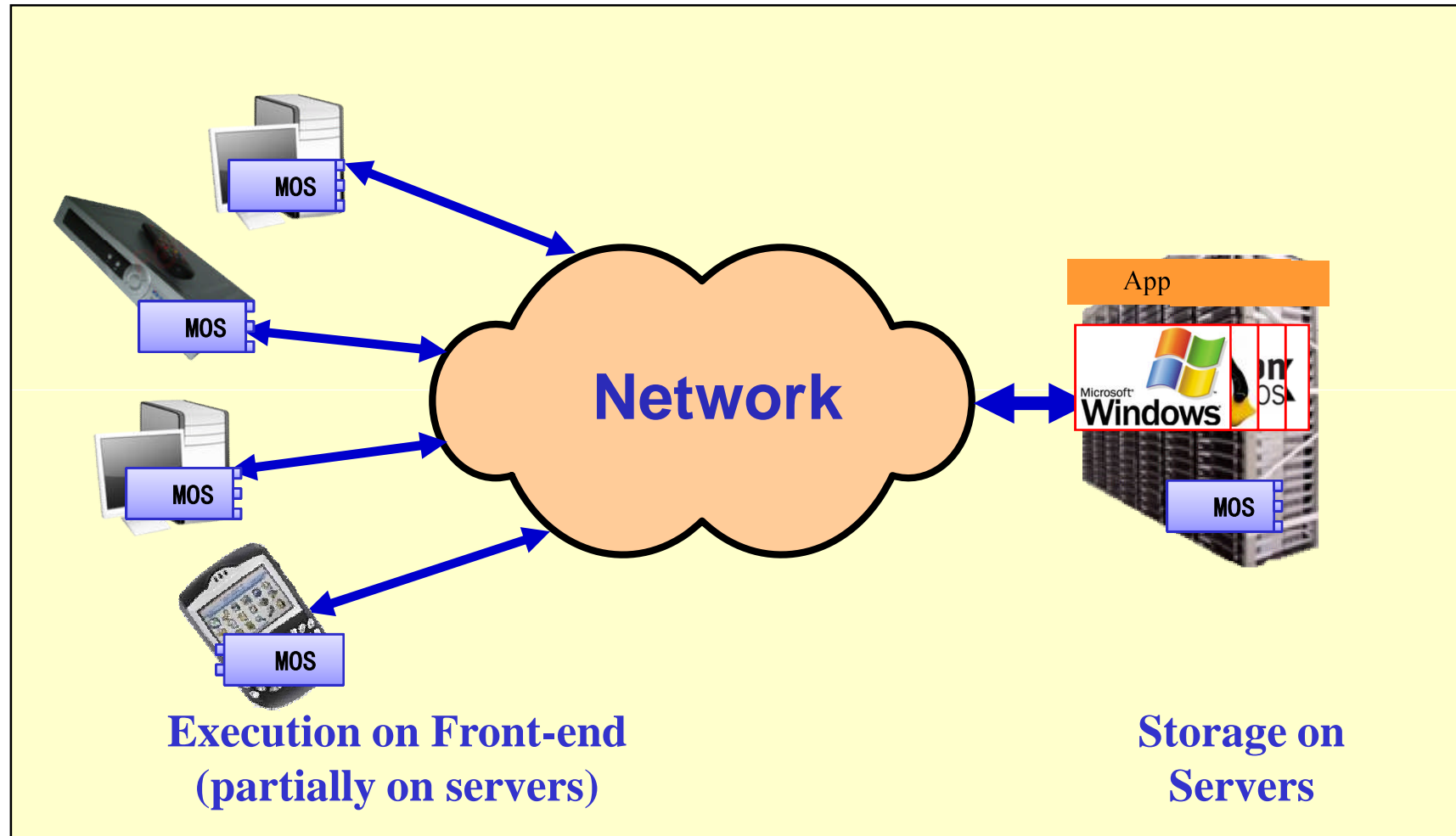




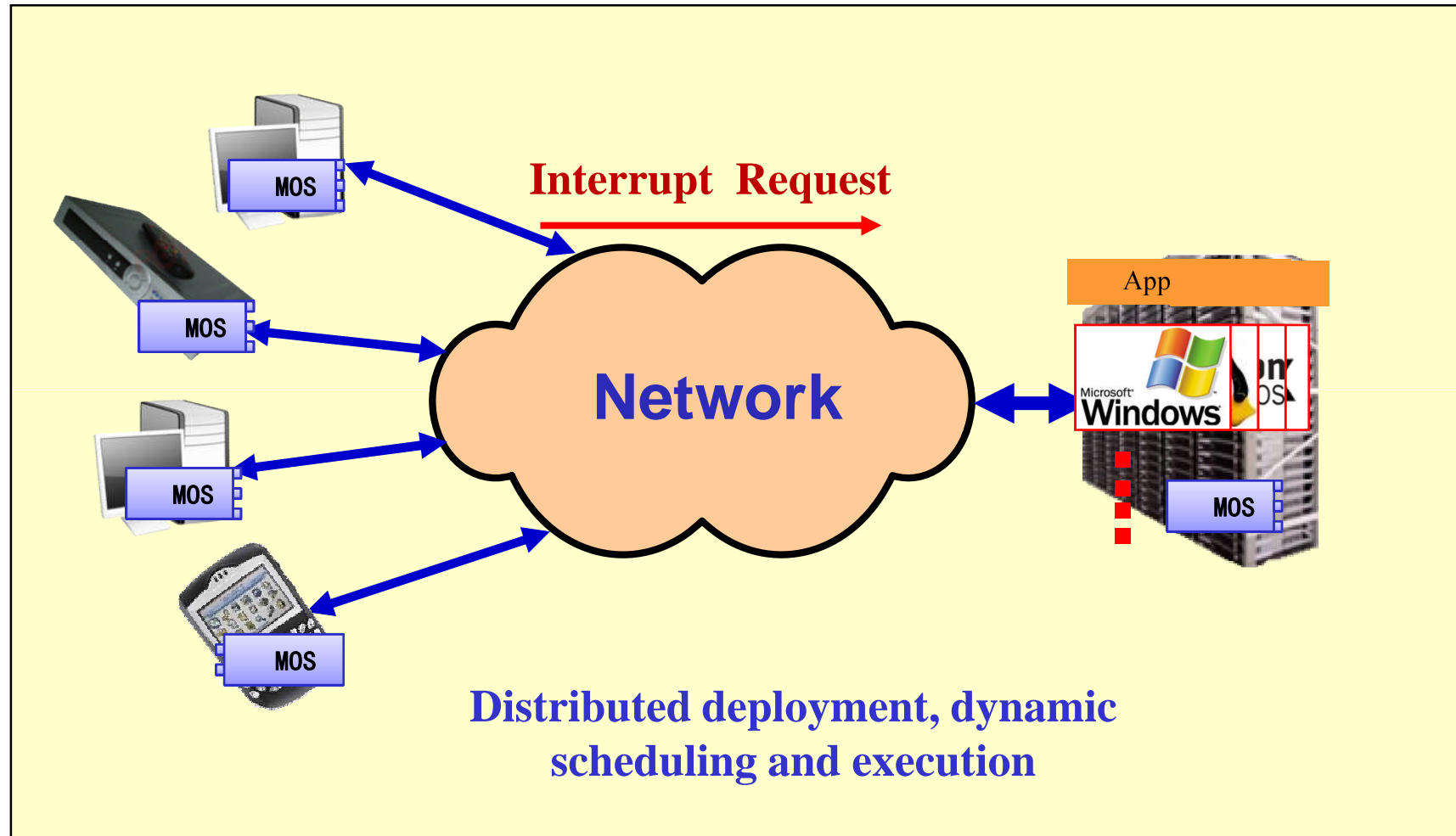
# Features of Transparent Computing

- Separated storage and execution of programs (including OS itself)
  - Store and execute program on different machine (in server or in client)
- Distributed deployment, dynamic scheduling and execution
- Compatible with different Instant OSes

# Separated Storage and Execution

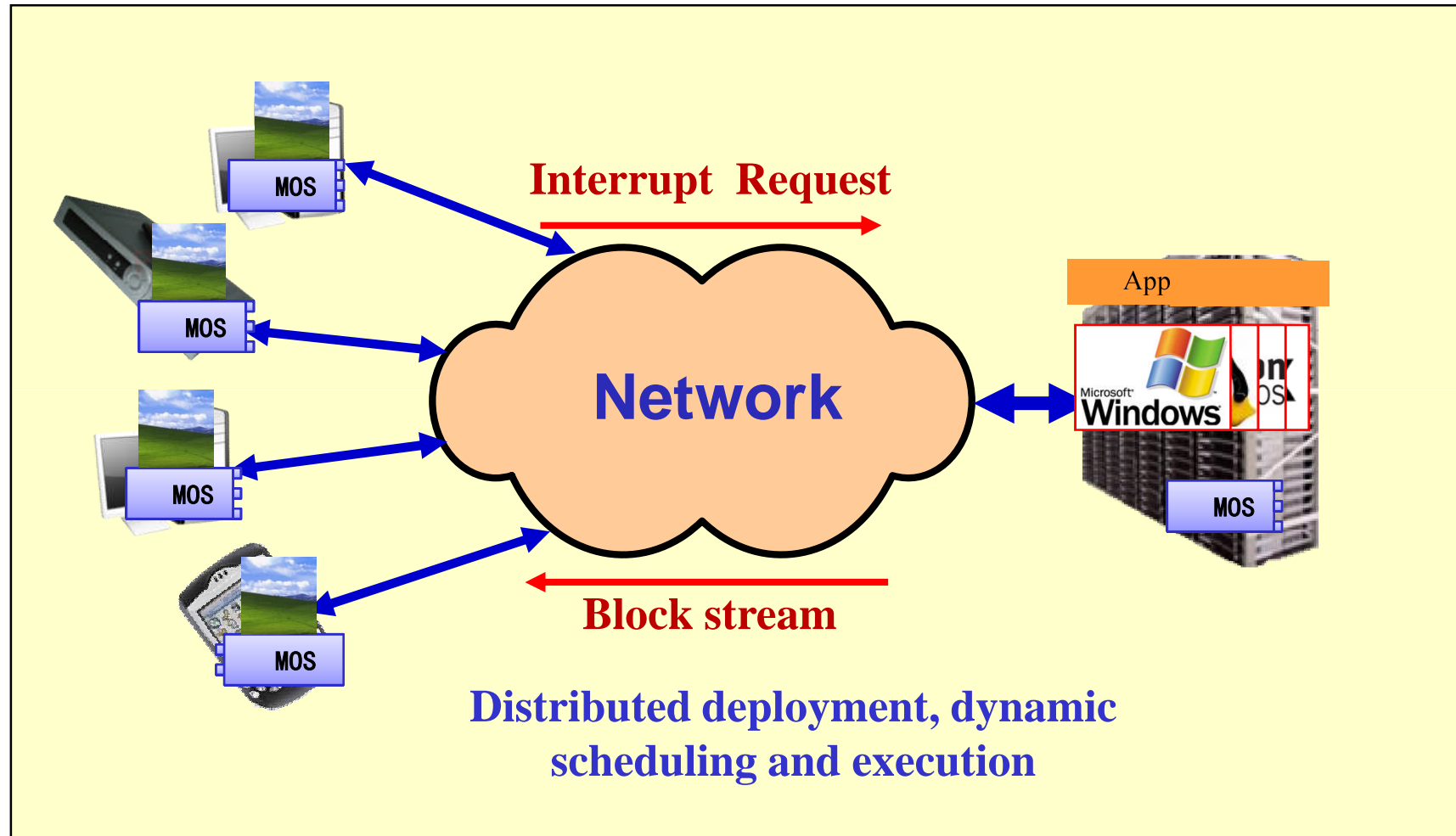


# Distributed deployment, dynamic scheduling and execution

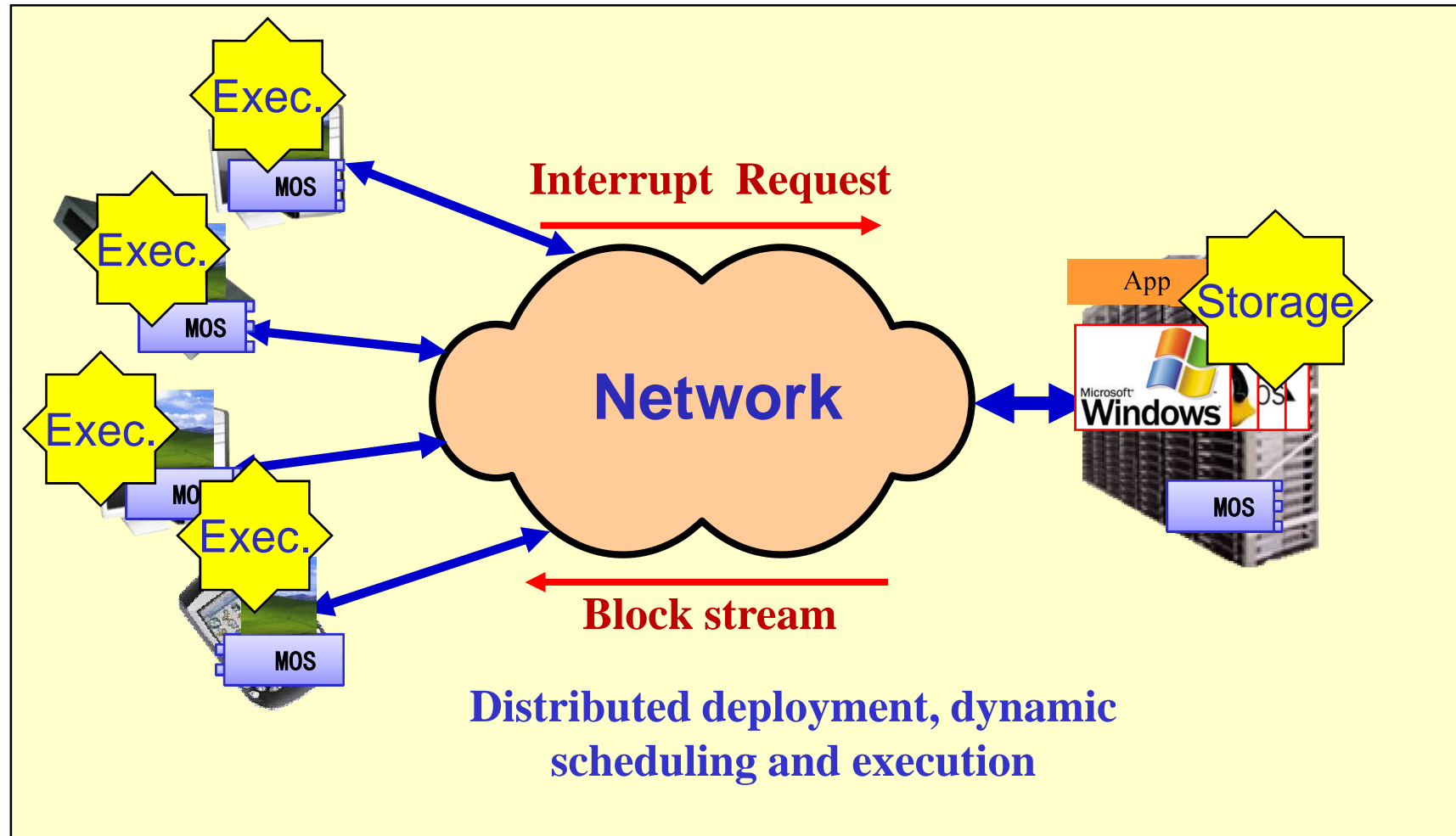




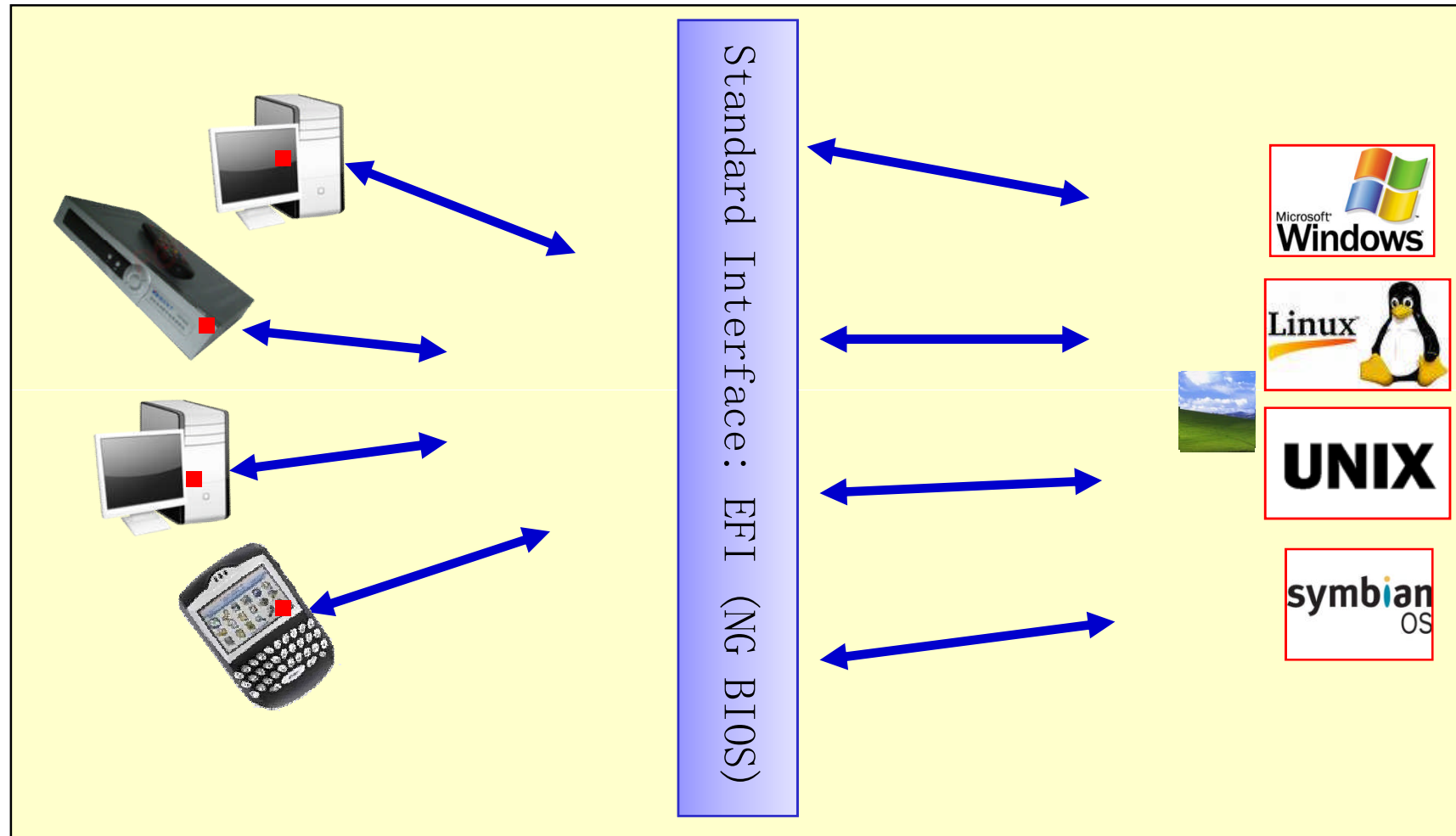
# Distributed deployment, dynamic scheduling and execution



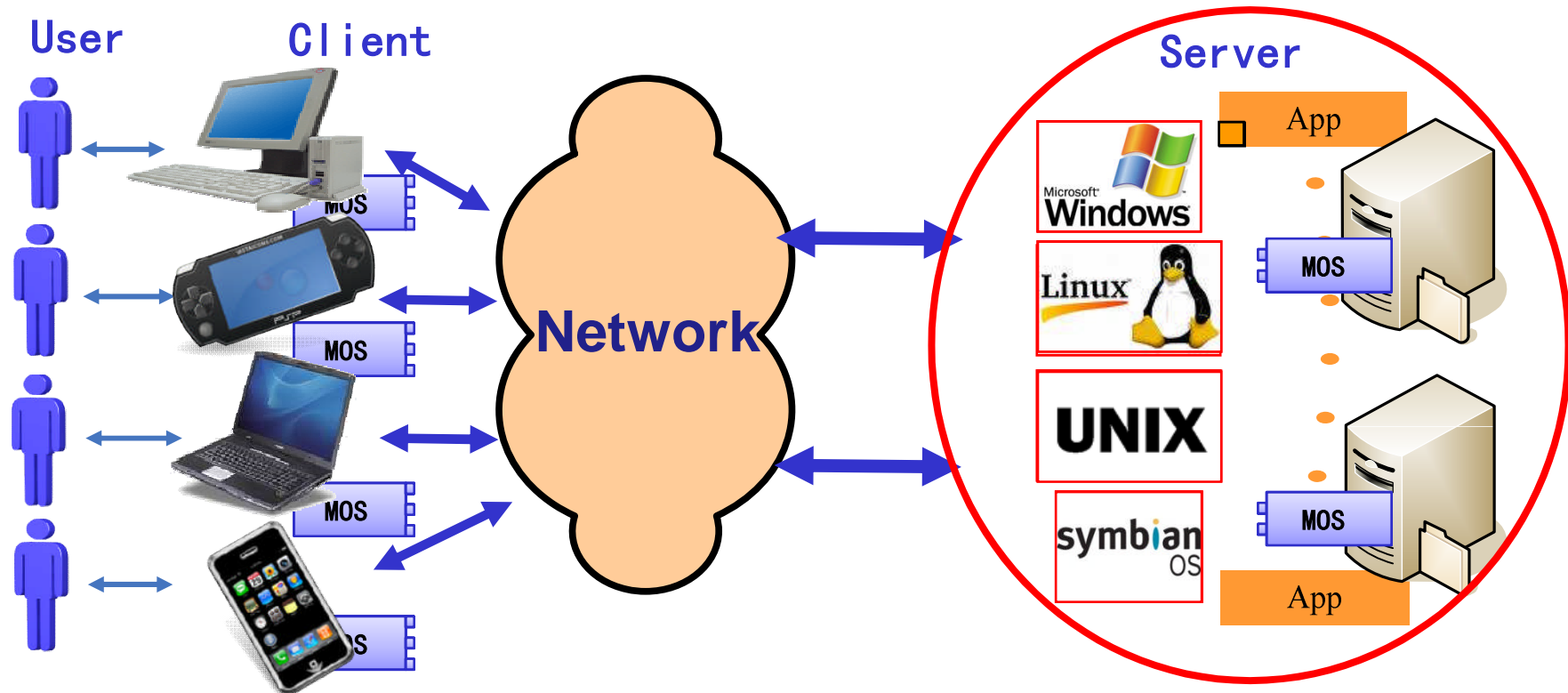
# Distributed deployment, dynamic scheduling and execution



# Compatible with different Instant OSes



# Advantages of Transparent Computing



(1) Lightweight Client (light-weight CPU/memory) → low hw cost, easy to maintain

(2) Small terminals running different OSe, apps, virtual computing → low cost

(3) Software sharing, SaaS → reduce software cost, easy to use

(4) Central management → improve manageability, security, reduce maintenance cost



# Agenda

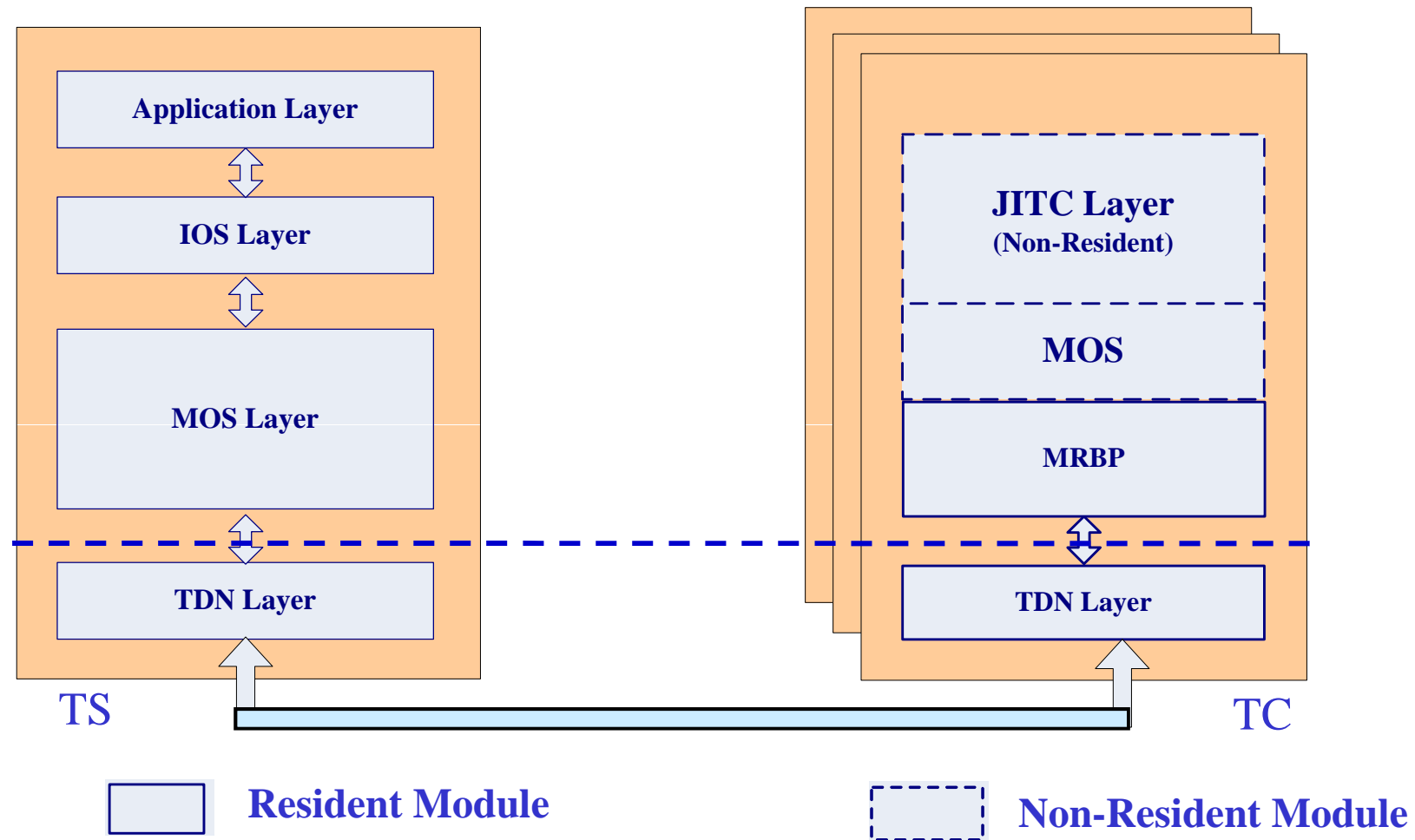
- What is the problem?
- What is Transparent Computing?
- Key technologies and issues
- Sample applications
- Vision



# Key technologies and issues

- How to control IOSes
- Build a standard OS/Mainboard interface: Support Multi-OSes
- Security checking mechanism
- Develop client systems: Support new service

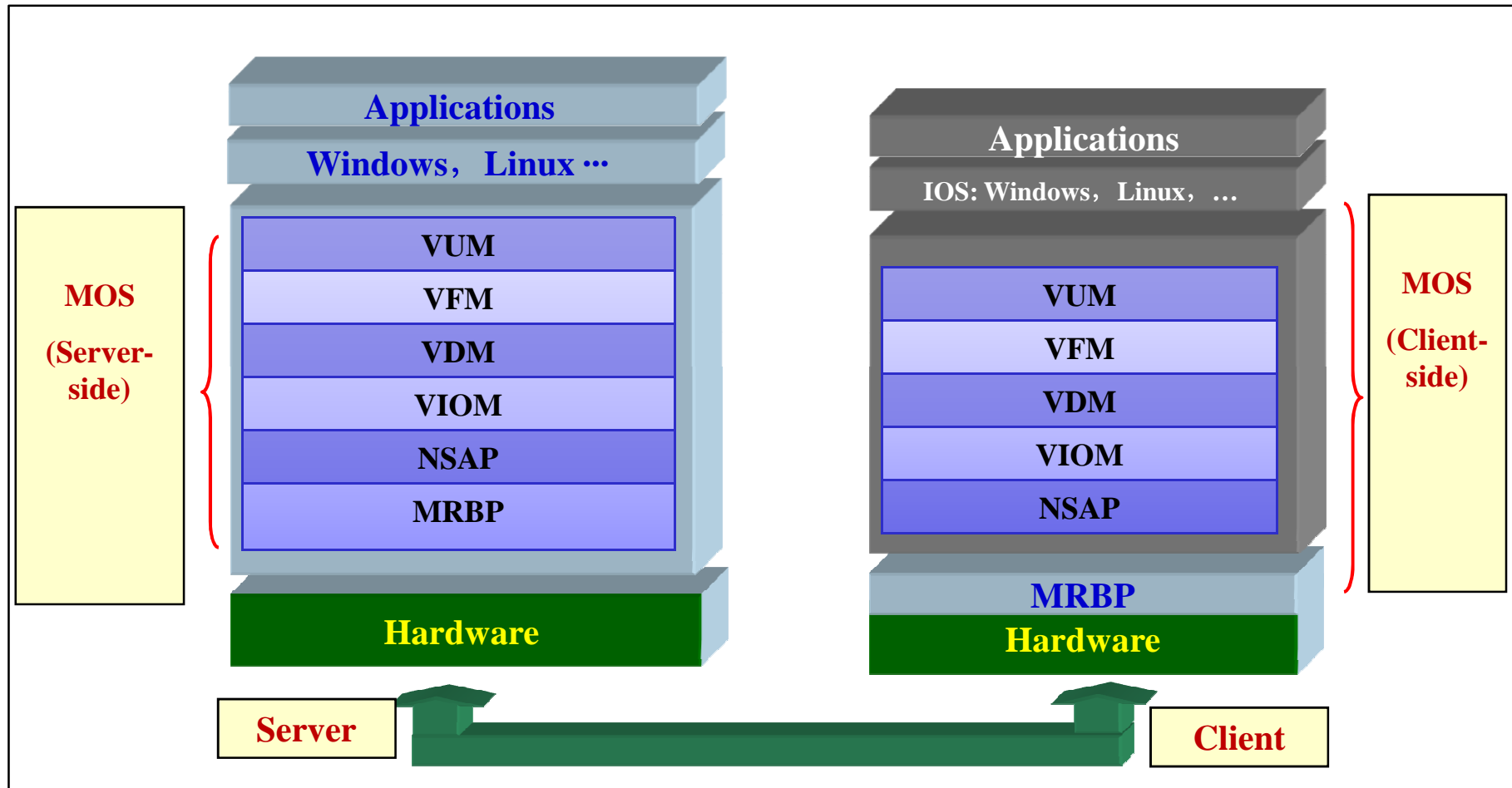
# How to control IOSes: MOS



**TDN (Transparent Delivery Network)**

**JITC (Just In Time Computing)**

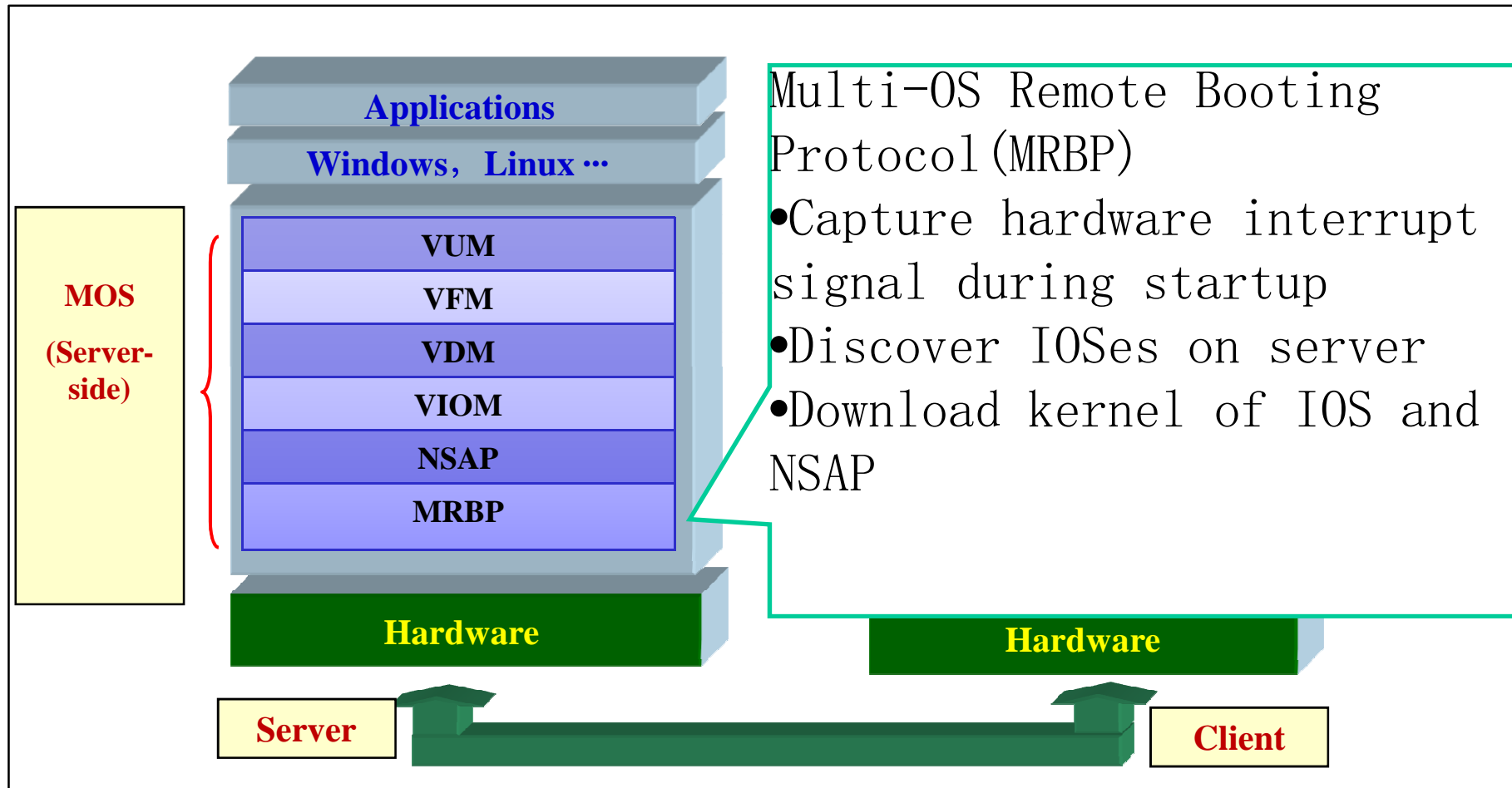
# MOS Implementation: 4VP+



 Resident  Non-resident



# MOS Implementation: 4VP+

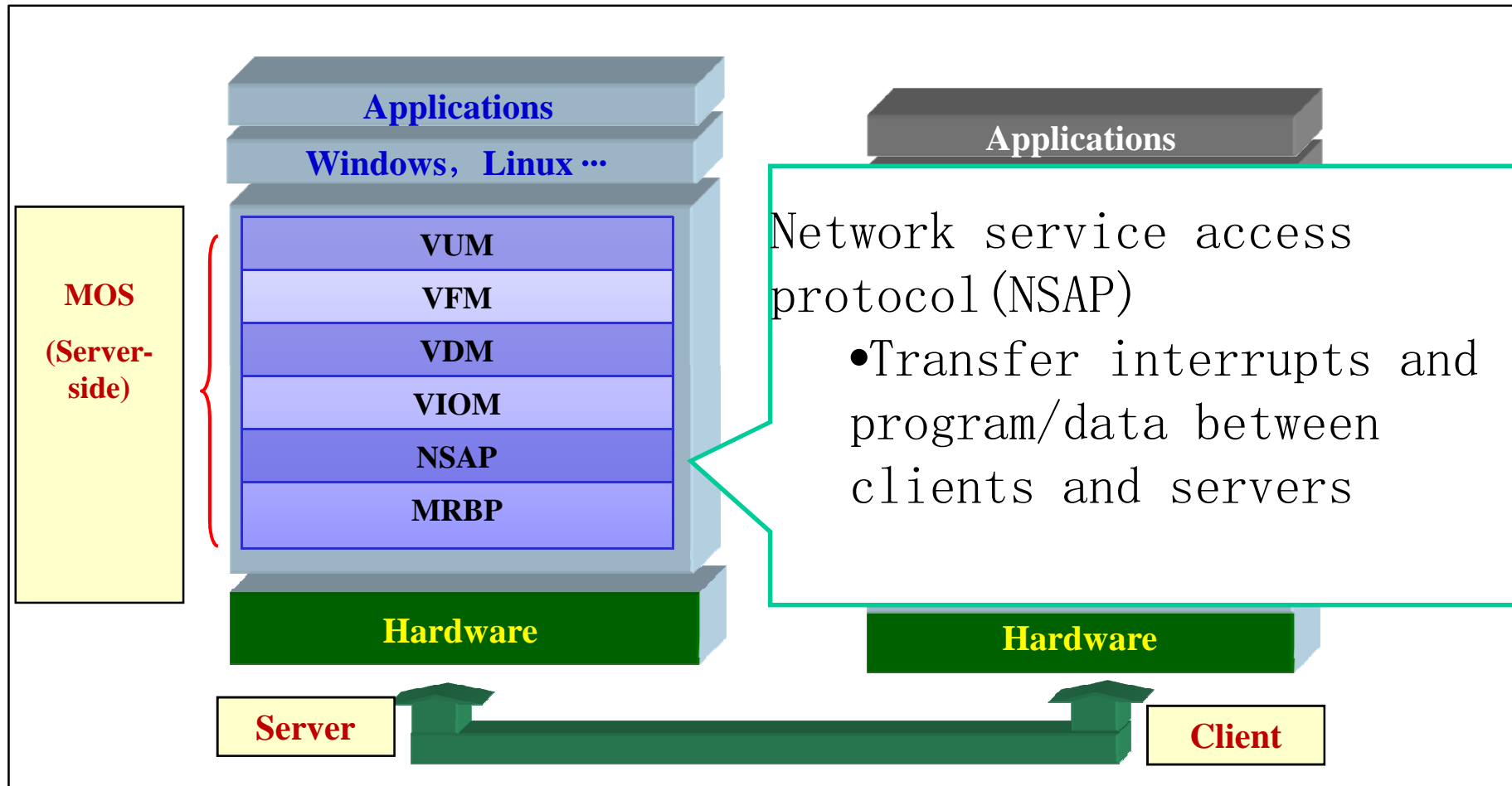


**Resident**



**Non-resident**

# MOS Implementation: 4VP+

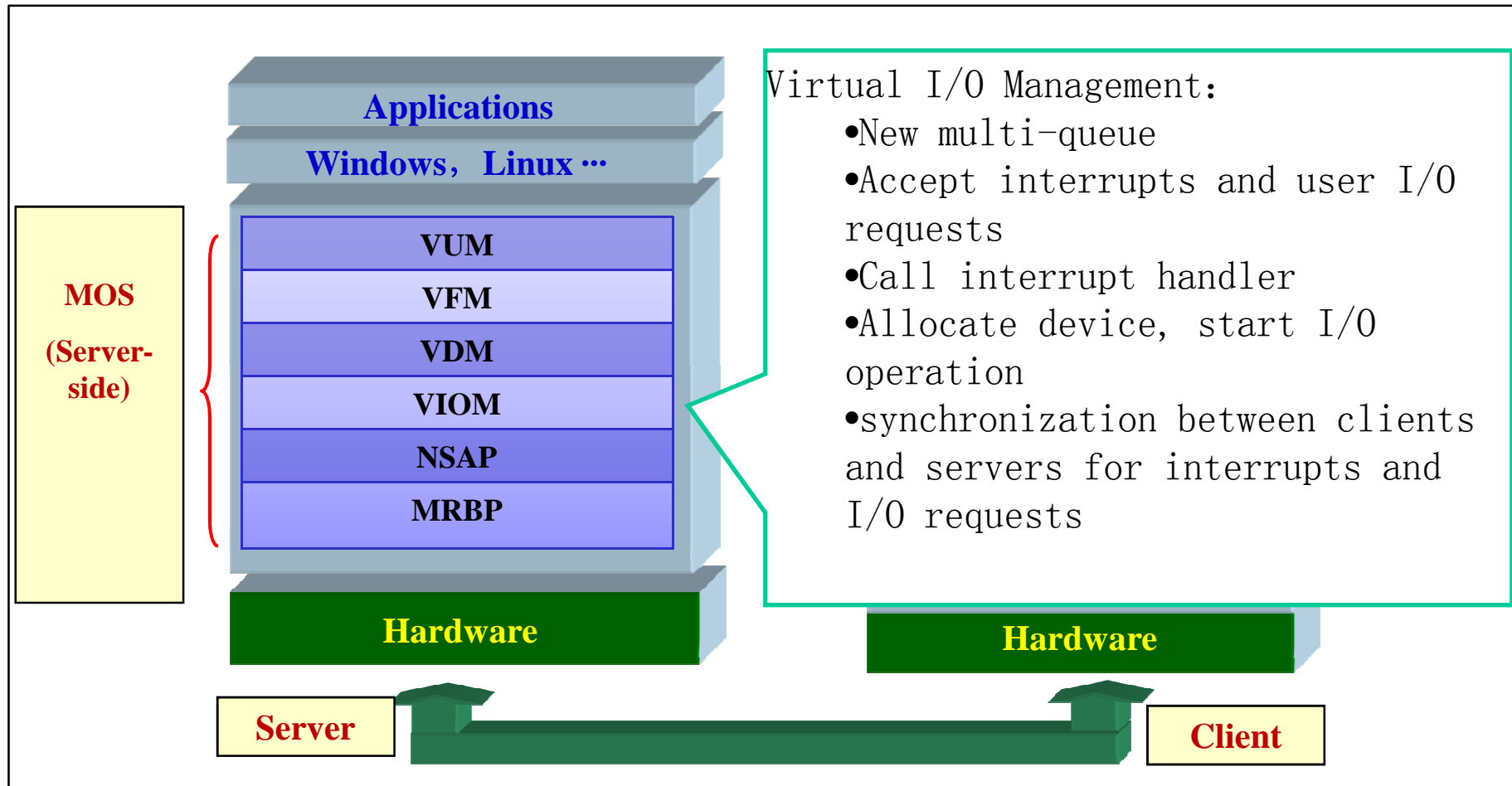


**Resident**



**Non-resident**

# MOS Implementation: 4VP+

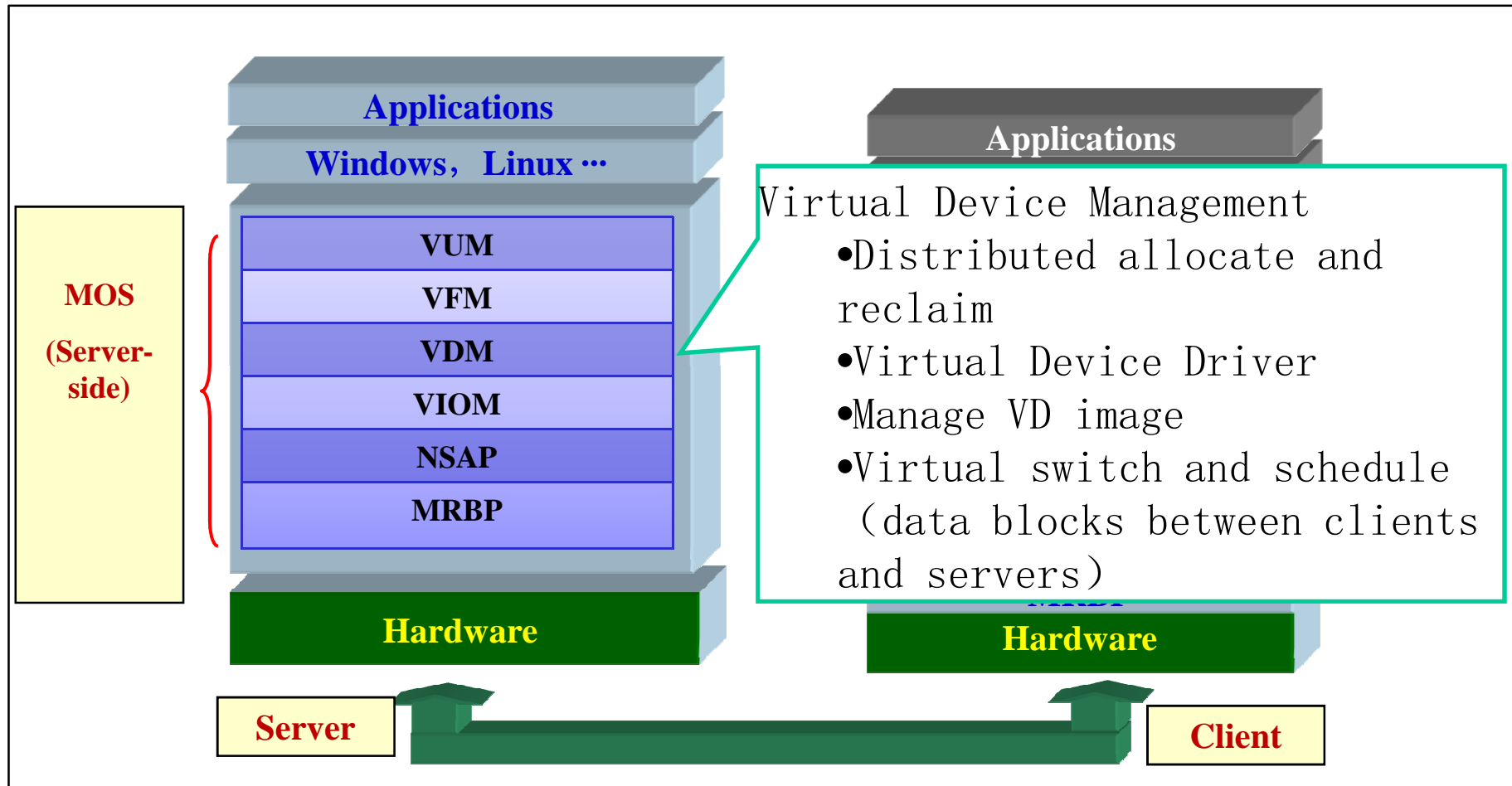


**Resident**



**Non-resident**

# MOS Implementation: 4VP+

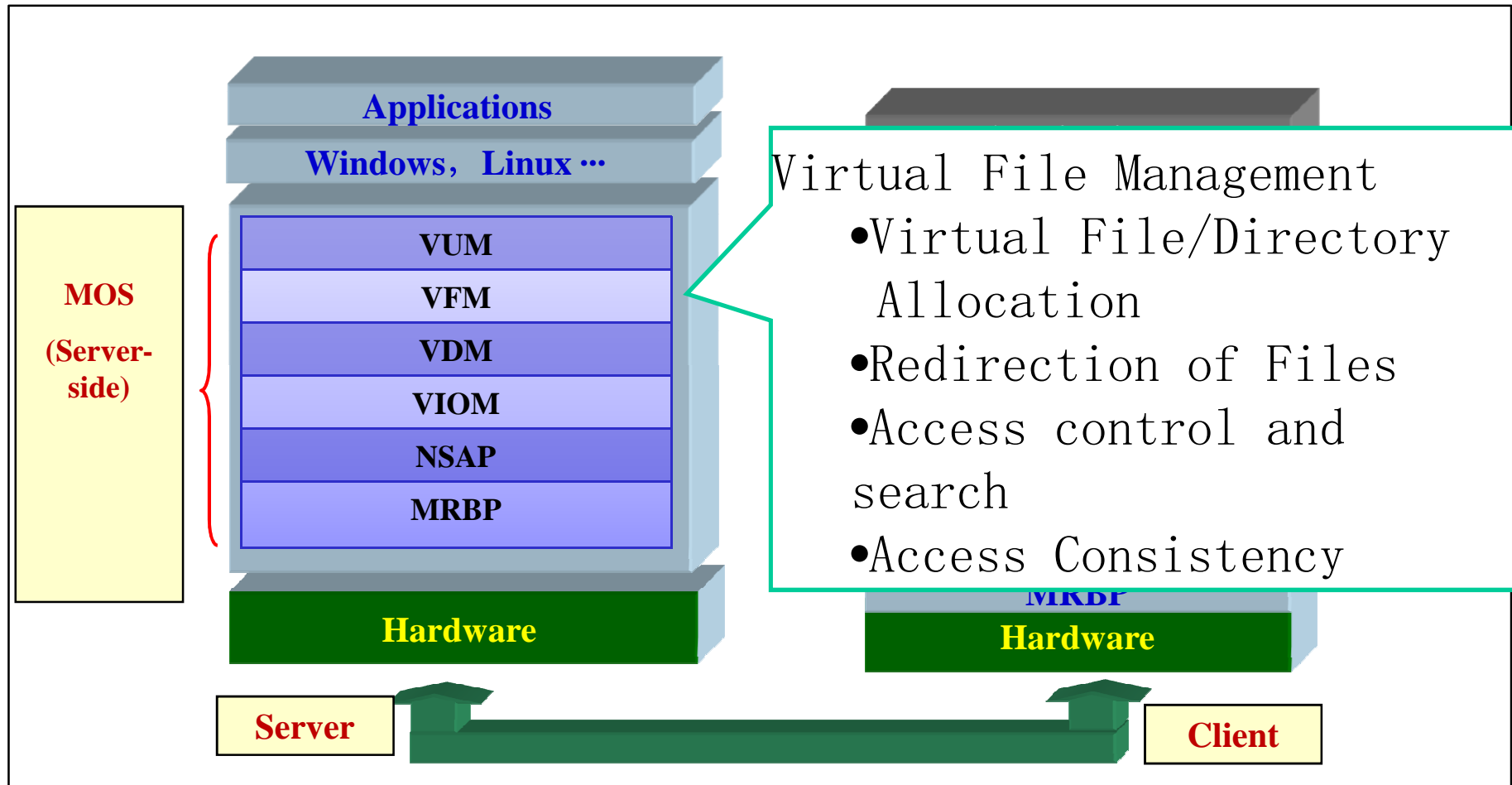


**Resident**

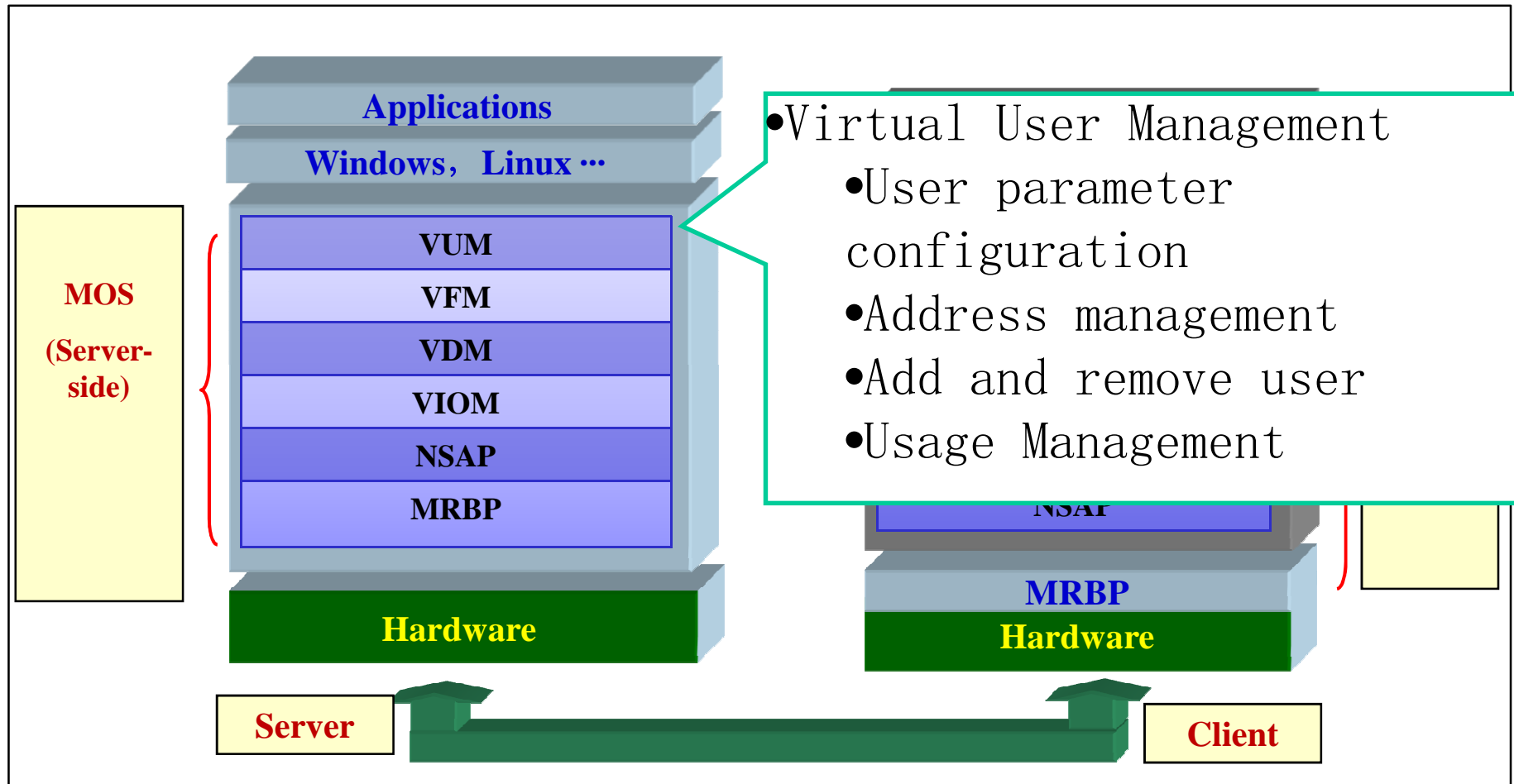


**Non-resident**

# MOS Implementation: 4VP+



# MOS Implementation: 4VP+



**Resident**



**Non-resident**



# Key technologies and issues

- How to control IOSes
- Build a standard OS/Mainboard interface: Support Multi-OSes
- Security checking mechanism
- Develop client systems: Support new service



# **Build a standard OS/Mainboard interface**

- Extensibility
  - Compatible with new software/hardware platforms?
- Efficiency
  - When we use lightweight CPU, do we need to support multiple operating systems if the interface itself is very complex?



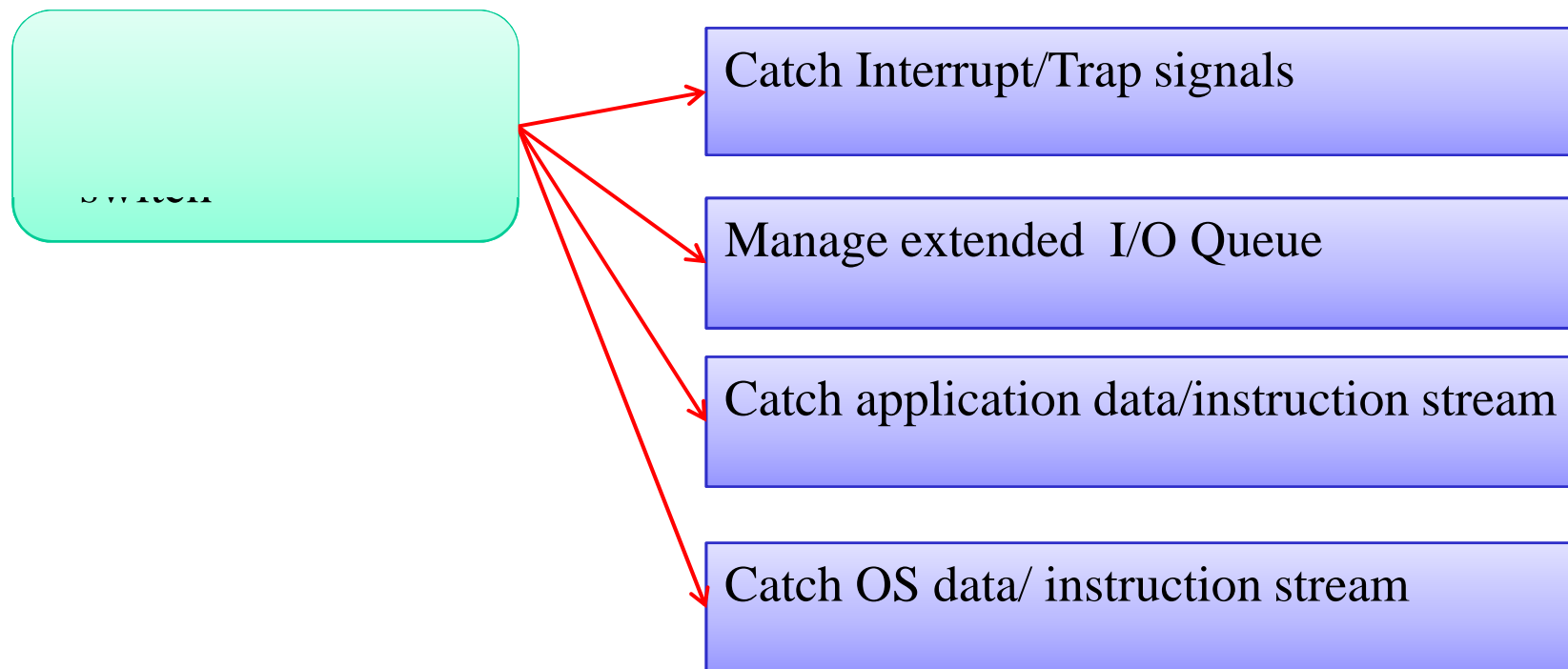


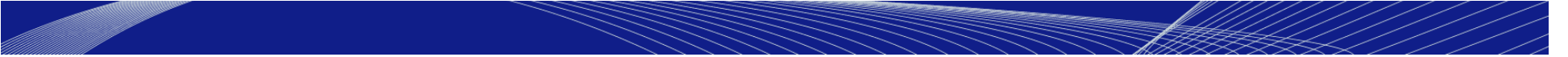
# Key technologies and issues

- How to control IOSes
- Build a standard OS/Mainboard interface: Support Multi-OSes
- **Security checking mechanism**
- Develop client systems: Support new service

# Security Checking Mechanism

- Monitor and control the data stream



- 
- Protection to virtual files
    - ReWrite and ReRead mechanism
    - Multi-boot algorithm to anti-viruses
  - Central Data Storage
    - Distributed → Centralized



# Key technologies and issues

- How to control IOSes
- Build a standard OS/Mainboard interface: Support Multi-OSes
- Security checking mechanism
- Develop client systems: Support new service

# Transparent Client Devices

- Transparent Mobile Internet Devices ( T-MID )
- Transparent Lightweight Device (T-LID )
- Transparent Digital Electronics (T-DHD )



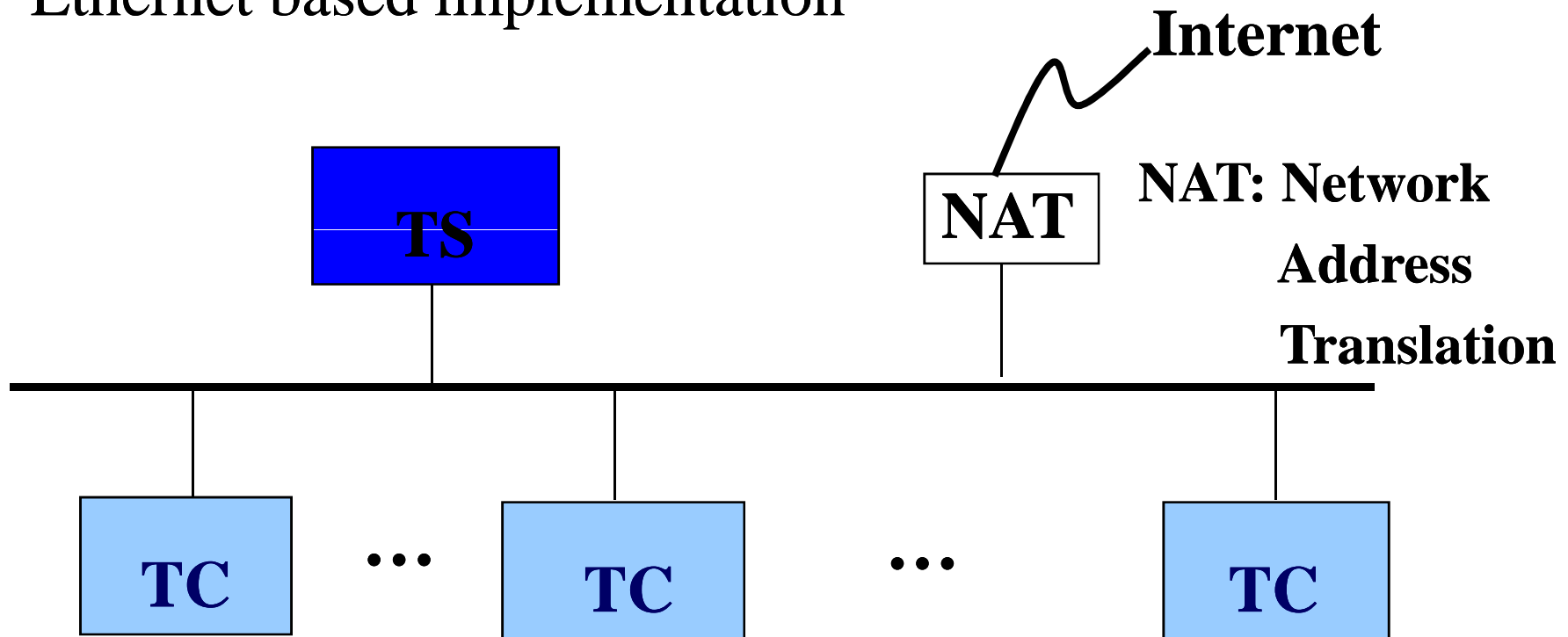


# Agenda

- What is the problem?
- What is Transparent Computing?
- Key technologies and issues
- Sample applications
- Vision

# Sample Implementation

Ethernet based implementation



TC:



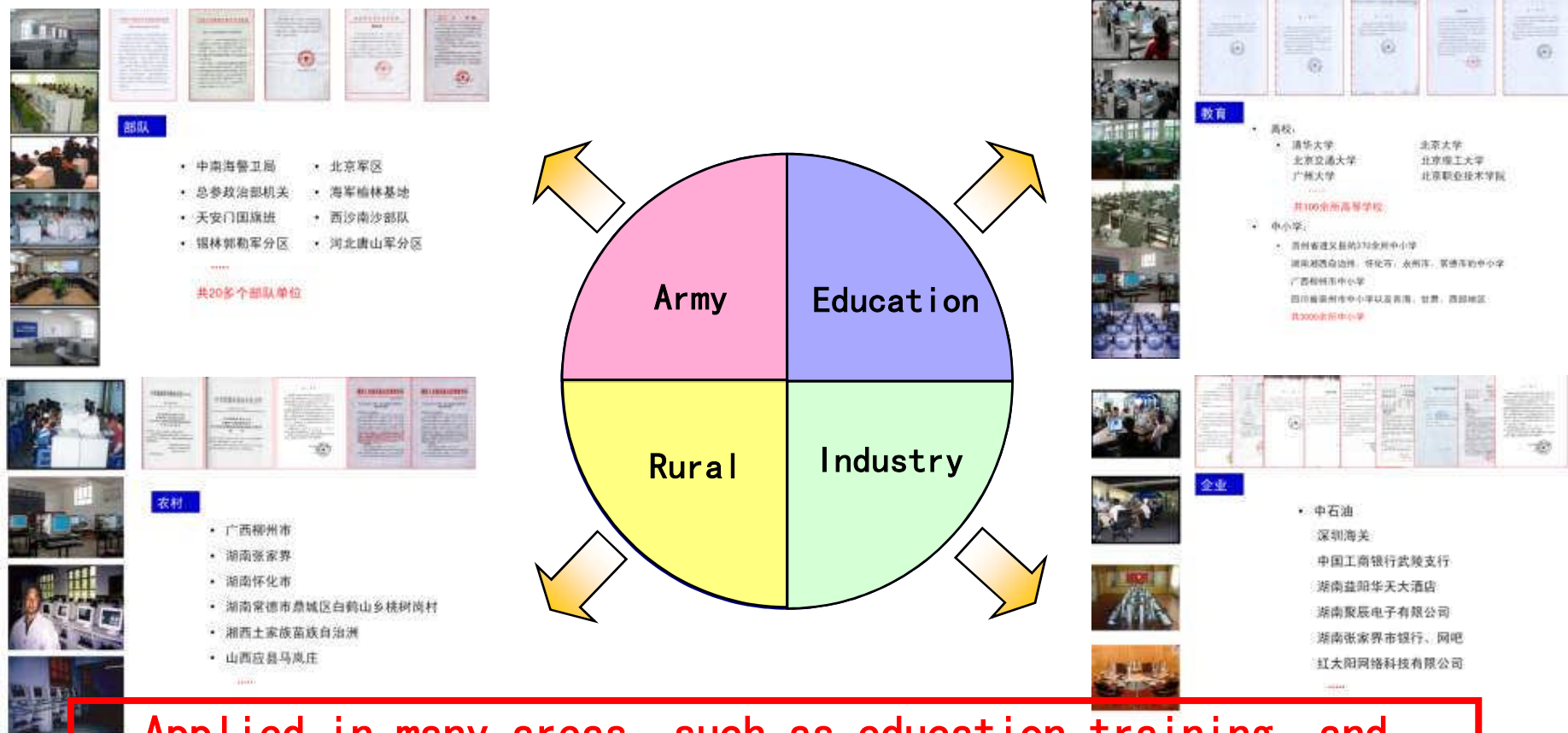
TS: Regular PC or Common Server

Software: Meta OS,  
support Linux, Windows 2000/XP



# Current Achievements

More than 160K copies of MOS has been installed. Annual sales income in related enterprises has exceeded 100M RMB.



Applied in many areas, such as education, training, and enterprise management, save cost more than 50 percent compared with PC and improve manageability and security

Demo video: 



# Agenda

- What is the problem?
- What is Transparent Computing?
- Key technologies and issues
- Sample applications
- Vision

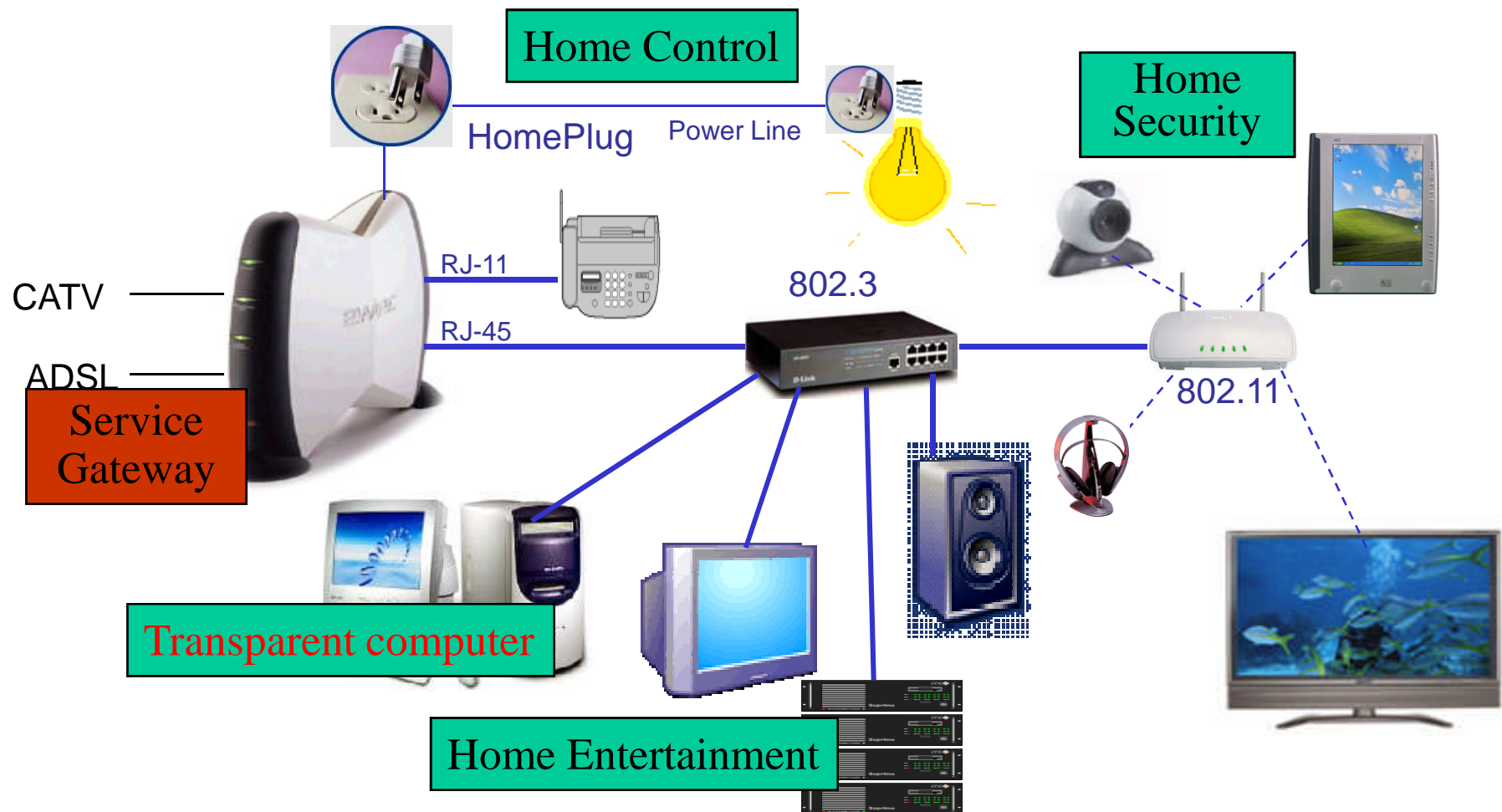


# Future Research Topics (1)

- Support Transparent Computing in mobile devices via wireless networks( WiFi, TD-SCDMA ... )
  - Driver development
  - Battery life time is limited in mobile devices
    - Reducing the network communications required by Transparent Computing
  - Network speed fluctuates
    - Caching and prefetching to provide limited feature when the system is not connected

# Future Research Topics (2)

- Support Transparent Computing in Digital Home





**Thanks!**