Open Programmable Architecture for Java-enabled Network Devices

A Revolution!

Tal Lavian
Technology Center
tlavian@NortelNetworks.com
http://openet.lab.baynetworks.com

Programmable Network Devices

Openly Programmable devices enable

new types of intelligence on the network

Agenda

- The market is changing
- Local Computation
- Architecture
- New types of applications
- ORE - Oplet Run-time Environment
- API’s
- Summary

Our Market is Changing

- Customers demand for Openness & programmability
- IEEE P1520
- Lucent’s Programmable Networks
- Intel as driving force

Lucent in Open Programmable Networks

- **Lucent's Programmable Network Conference 9/15-16**
  - Cosponsored: Sun, HP, Oracle, Novell, Compaq …….
  - Over 500 participants, enthusiastic response
- **SoftSwitch as programmable network**
  - 70 developers
  - Jun 99 - PR, Sept 99 - Prog Conference, October 99 - SDK, Jan 00 - APIs, Feb 00 - Dev Conference, May 00 - 3rd party Test Lab
- **7 R/E program**
- **$1.7B- Excel Switching acquisition (Programmable switch)**
- **Other products and directions in Openness and Programmability**
- **Marketing and PR on Lucent's Openness**

ES - Openet Center

- Create an open development community to deliver customer-valued solutions based on Nortel Networks' and partners' products and technologies
- Openet Center creates a focus to open Nortel Networks platforms
- Promotes Open Network Computing
- This potentially changes landscape and rules of the networking industry

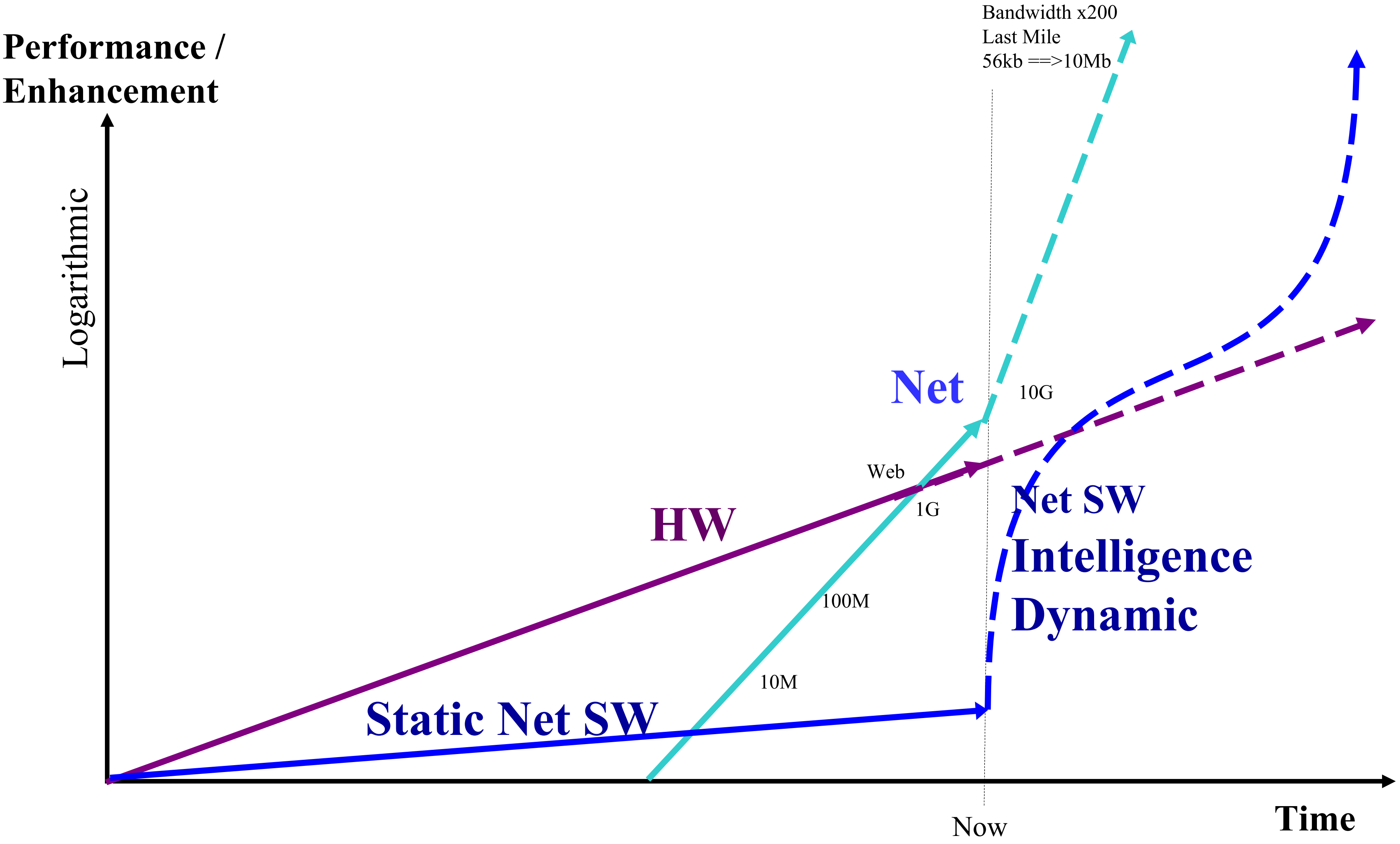
Vision: JOOSE - the OS of Choice for Programmable Network Devices

- JOOSE - Java Oplet Operation System Environment
- JOOSE is OS of choice for routers and network devices.
- Be the Microsoft for Routers
- (or better way the Linux for Programmable Network Devices)

Agenda

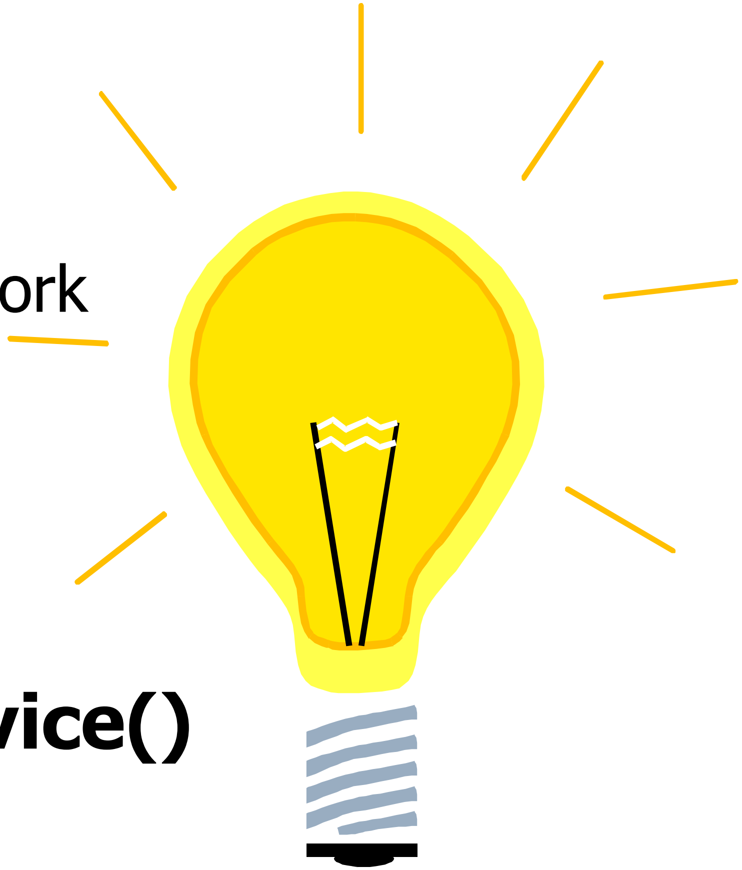
- The market is changing
- Local Computation
- Architecture
- Applications
- ORE - Oplet Run-time Environment
- API’s
- Summary

Evolution vs. Revolution



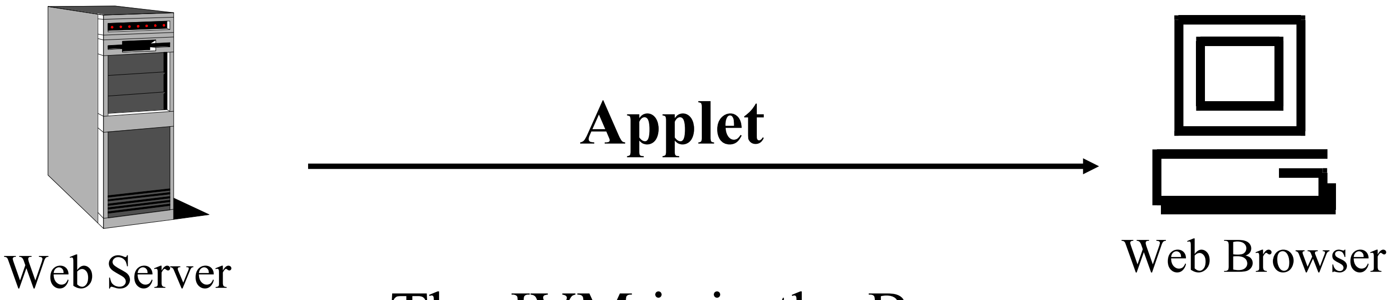
Changing the Rules of the Game

- **Move** Turing Machine **onto device**
  - Add local intelligence to network devices



```
while (true) {
   doLocalProcessingOnDevice()
   }
```

Technology Concept



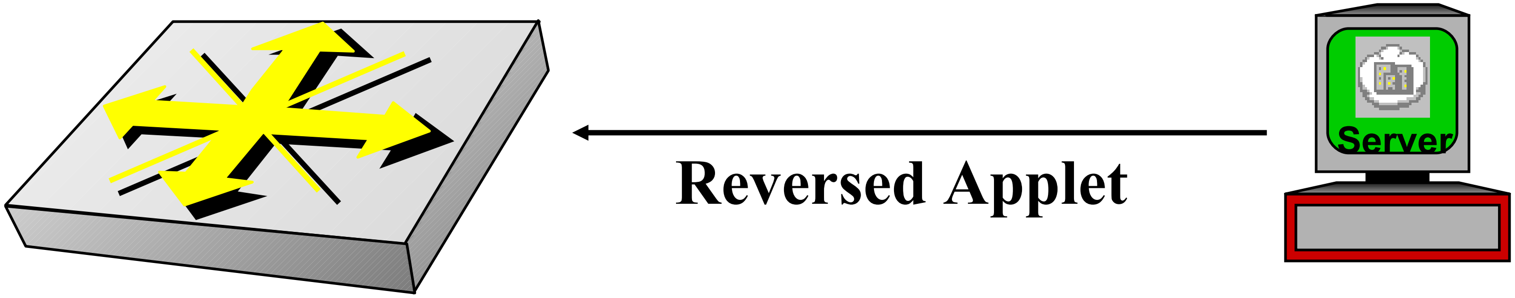
Applet

Web Server

Web Browser

The JVM is in the Browser

Download applications for local processing



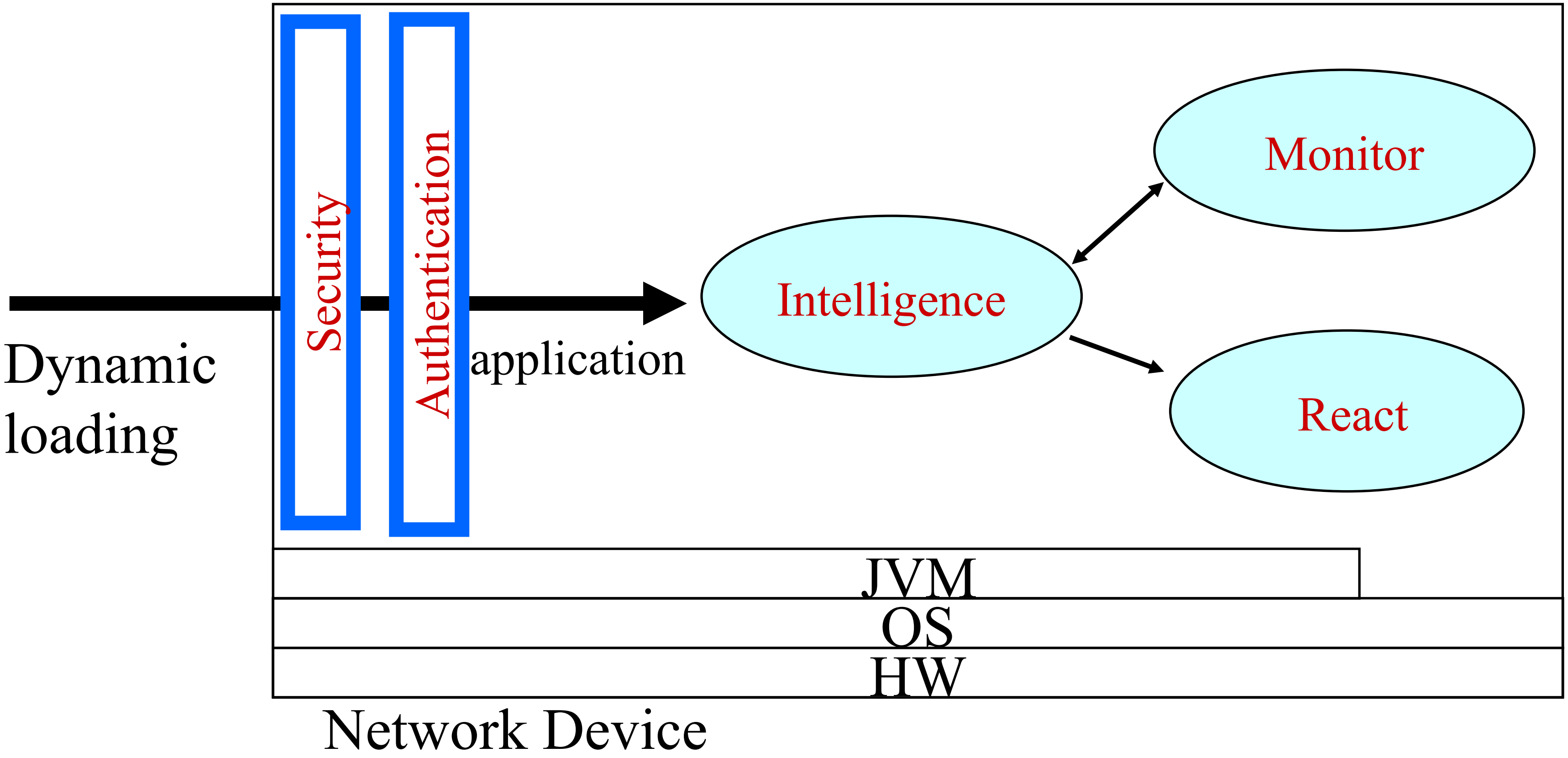
Reversed Applet

The Web Changed Everything

- **Browsers**
  - Introducing JVM to browsers allowed dynamic loading of Java ***Applets*** to end stations
- **Routers**
  - Introducing JVM to routers allows dynamic loading of Java ***Oplets*** to routers

This Capability WILL Change Everything

Example: Downloading Intelligence



Security and Stability

- **secure download of Java Applications**
- **safe execution environment**
  - insulate core router applications from dynamically loaded applications

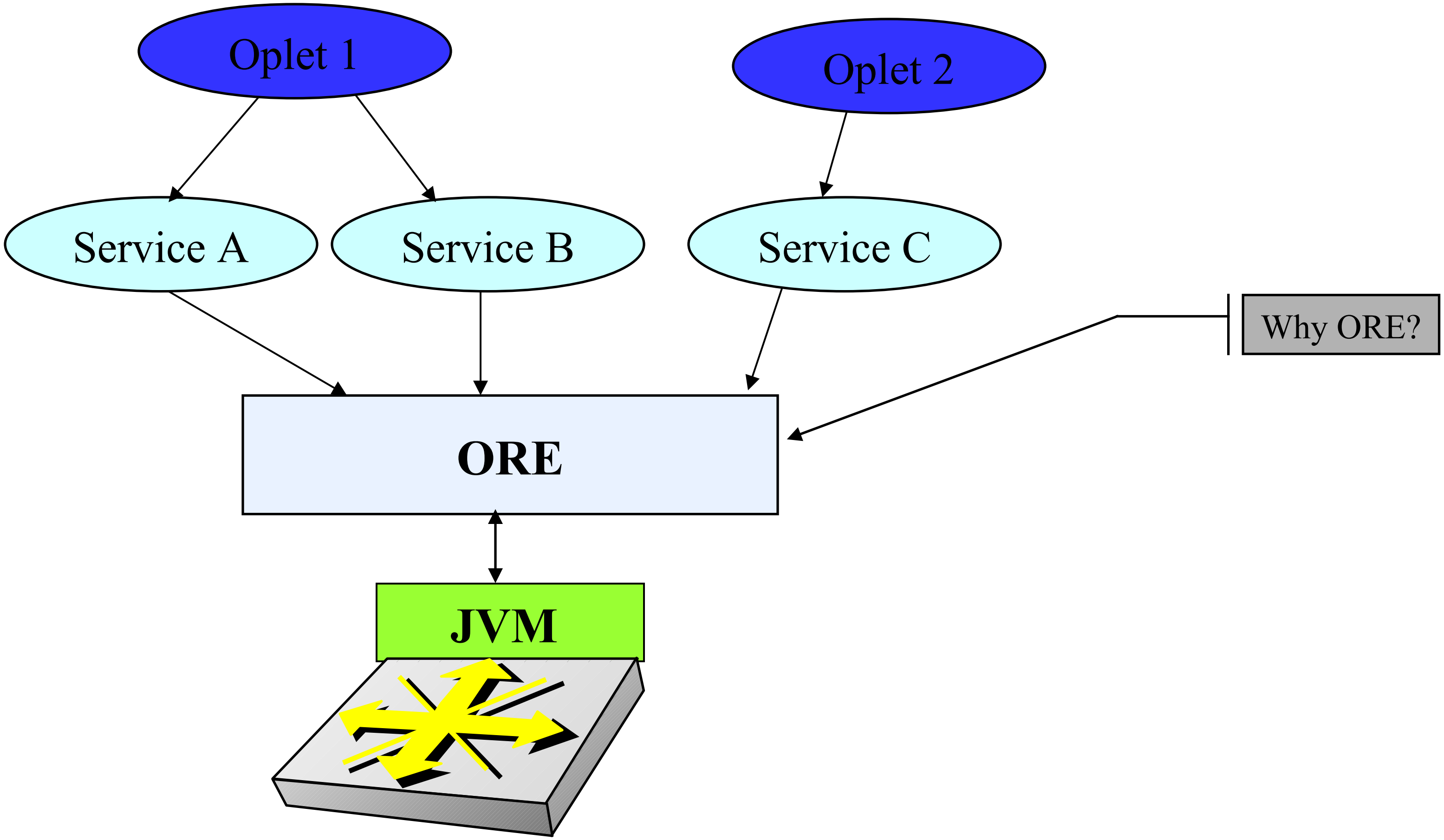
Device-based Intelligence

- **Static-vs-Dynamic Agents**
  - Static
    - SNMP set/get mechanisms
    - Telnet, User Interfaces (cli, web, etc…)
  - Dynamic closed-loop interaction on nodes
    - capable of dealing with new and difficult situations
    - autonomous and rational properties.
    - dynamically system monitoring & modification
    - report status and trends

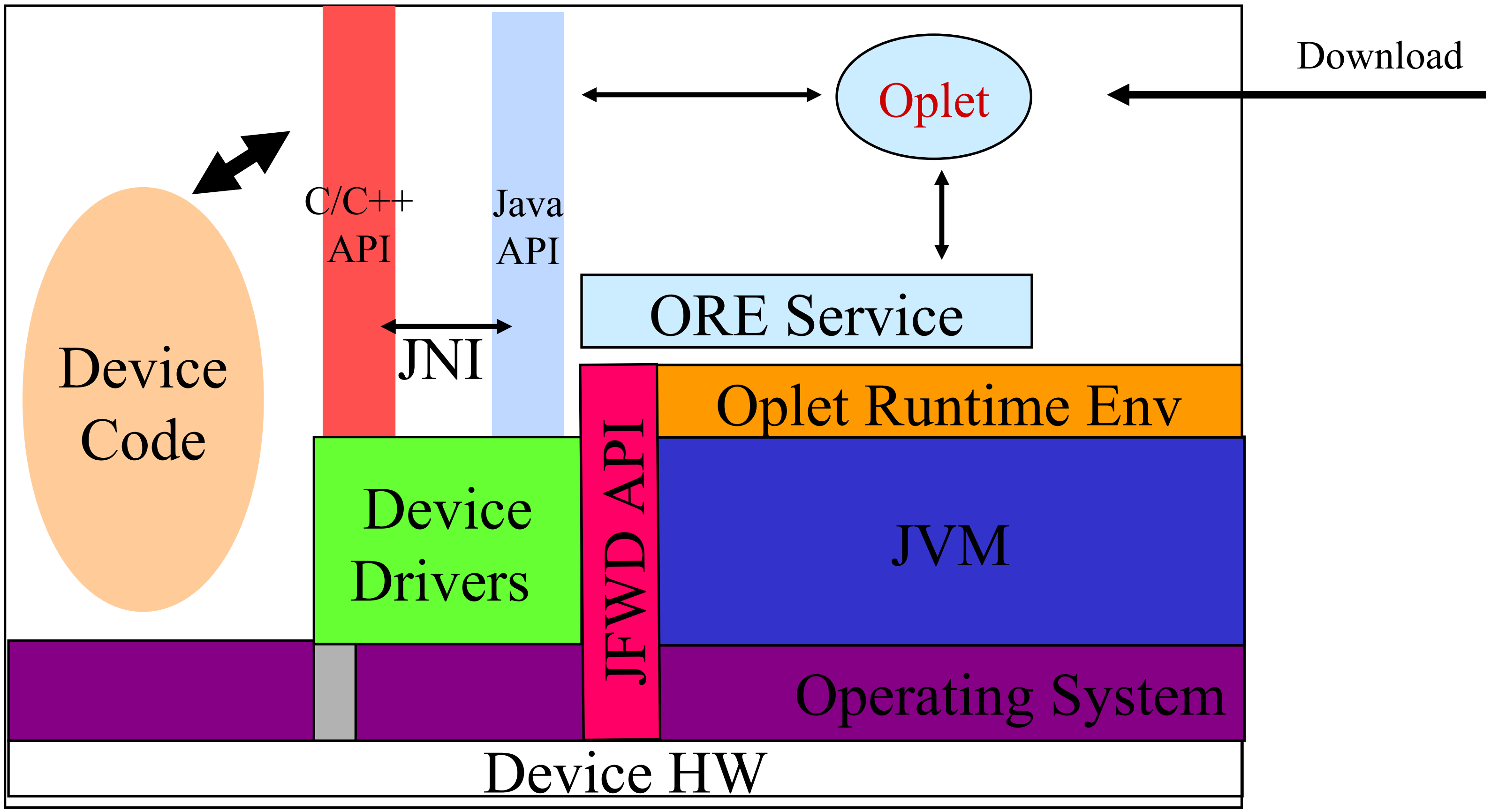
Agenda

- The market is changing
- Local Computation
- Architecture
- Applications
- ORE - Oplet Run-time Environment
- API’s
- Summary

ORE - Oplet Run-time Environment



Java-enabled Device Architecture

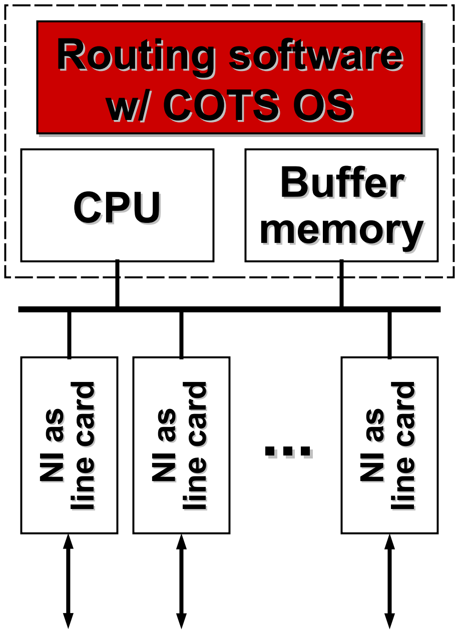


Architecture Issues

- **Green Threads -vs- Native Threads**
  - Native threads:
    - provides non-interference between Java applications
    - difficult thread-to-thread communication and sharing of data between threads
    - creates a dependency on underlying RTOS
    - multiple JVM instances consume resources
  - Green Threads
    - single JVM must manage CPU & memory resources between concurrently running threads

Evolution of Router Architecture

Centralized, CPU-based Model



Routing software w/ COTS OS

CPU

Buffer memory

NI as line card

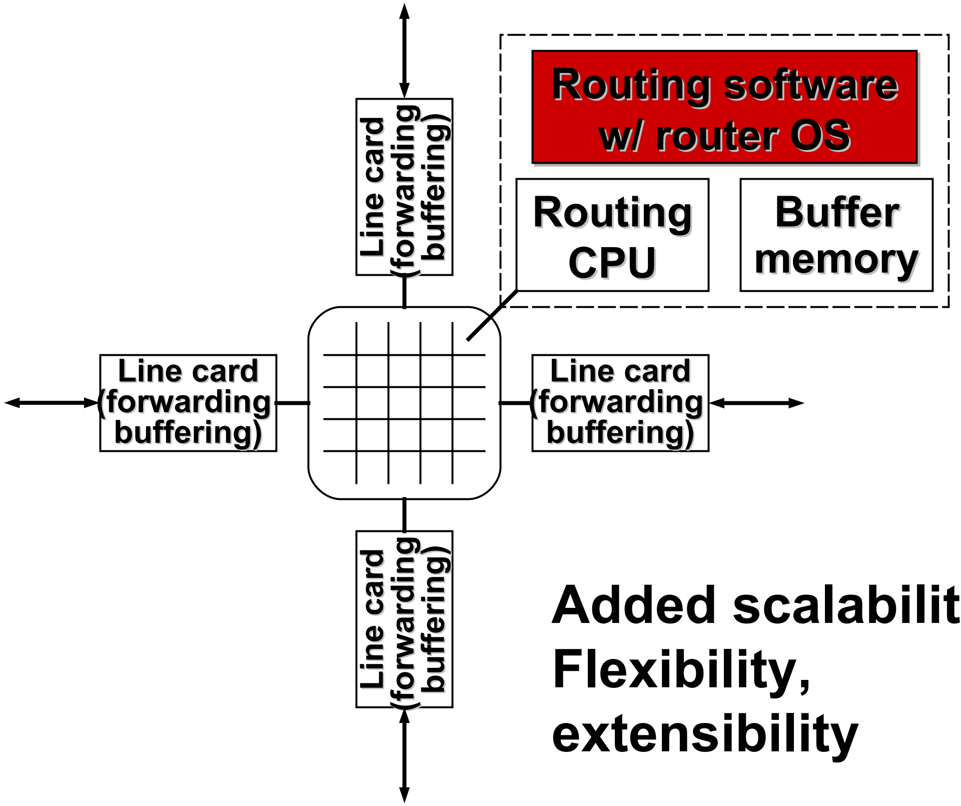
NI as line card

...

NI as line card

Control + Forwarding Functions combined

Distributed, line-card based Model



Routing software w/ router OS

Routing CPU

Buffer memory

Line card (forwarding buffering)

Line card (forwarding buffering)

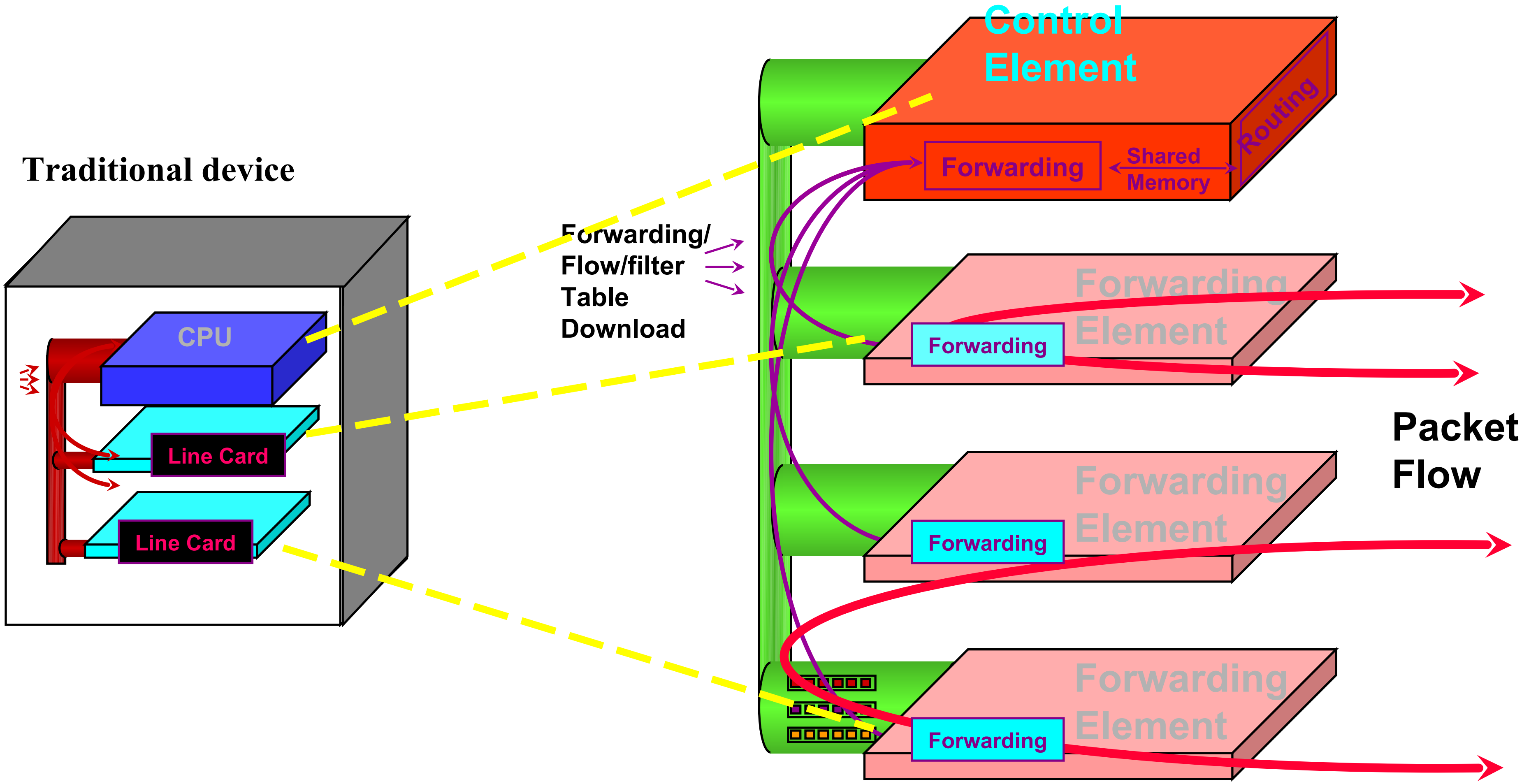
Line card (forwarding buffering)

Line card (forwarding buffering)

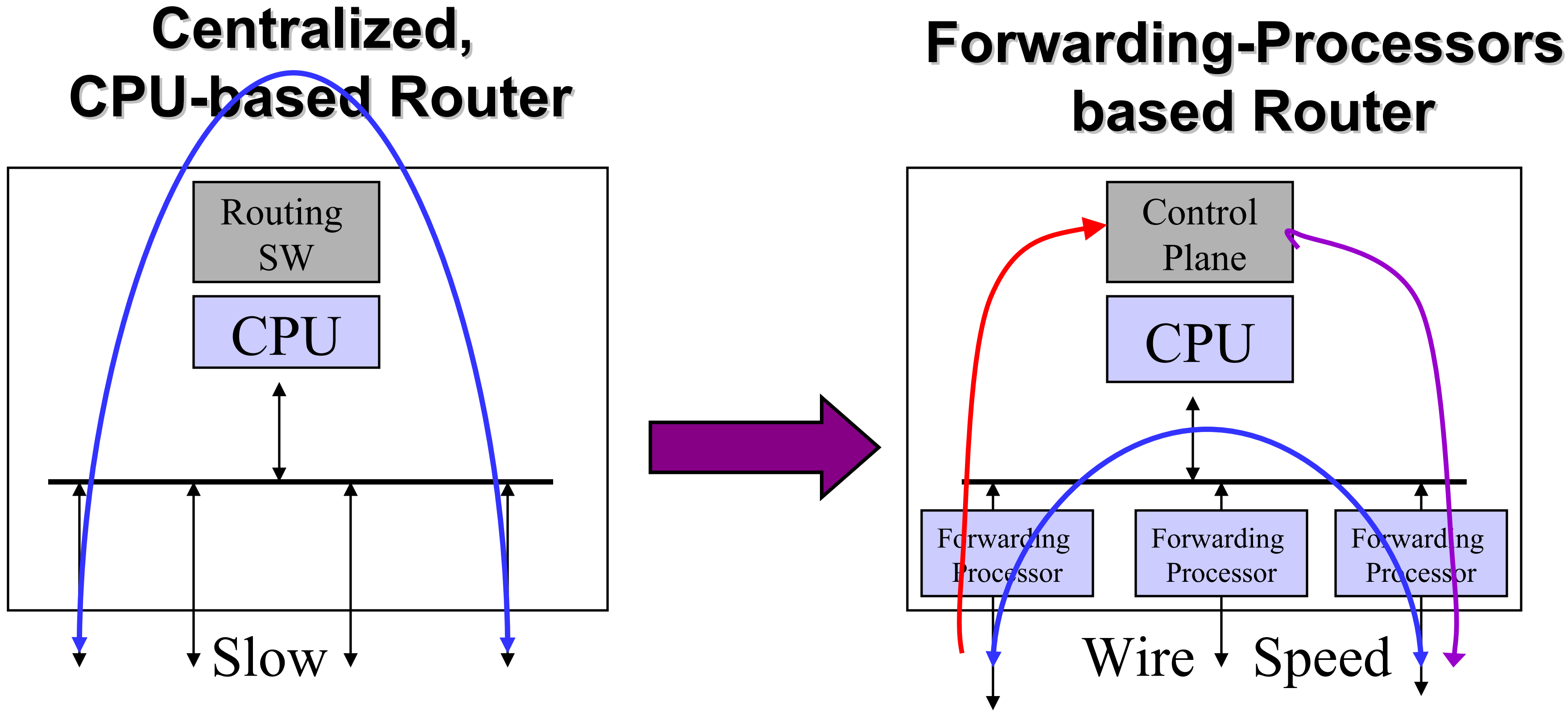
Added scalability, Flexibility, extensibility

Control separated From forwarding

Explicit Separation of Control Plane from Data Forwarding



Separation of Control and Forwarding Planes



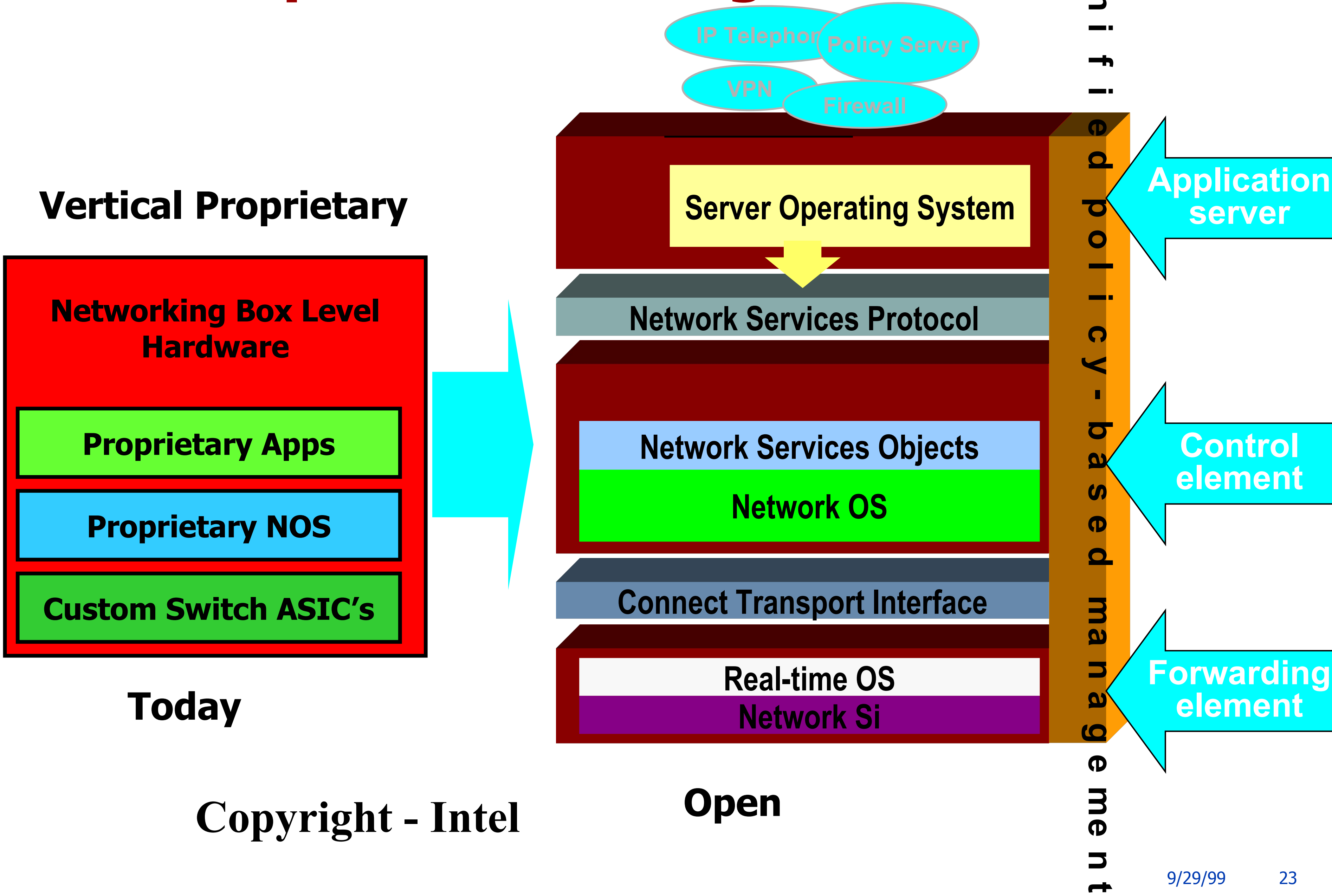
Centralized, CPU-based Router

Forwarding-Processors based Router

Control + Forwarding Functions combined

Control separated From forwarding

Open Networking Architecture



Vertical Proprietary

Networking Box Level Hardware

Proprietary Apps

Proprietary NOS

Custom Switch ASIC's

Today

IP Telephon

Policy Server

VPN

Firewall

Server Operating System

Network Services Protocol

Network Services Objects

Network OS

Connect Transport Interface

Real-time OS

Network Si

Open

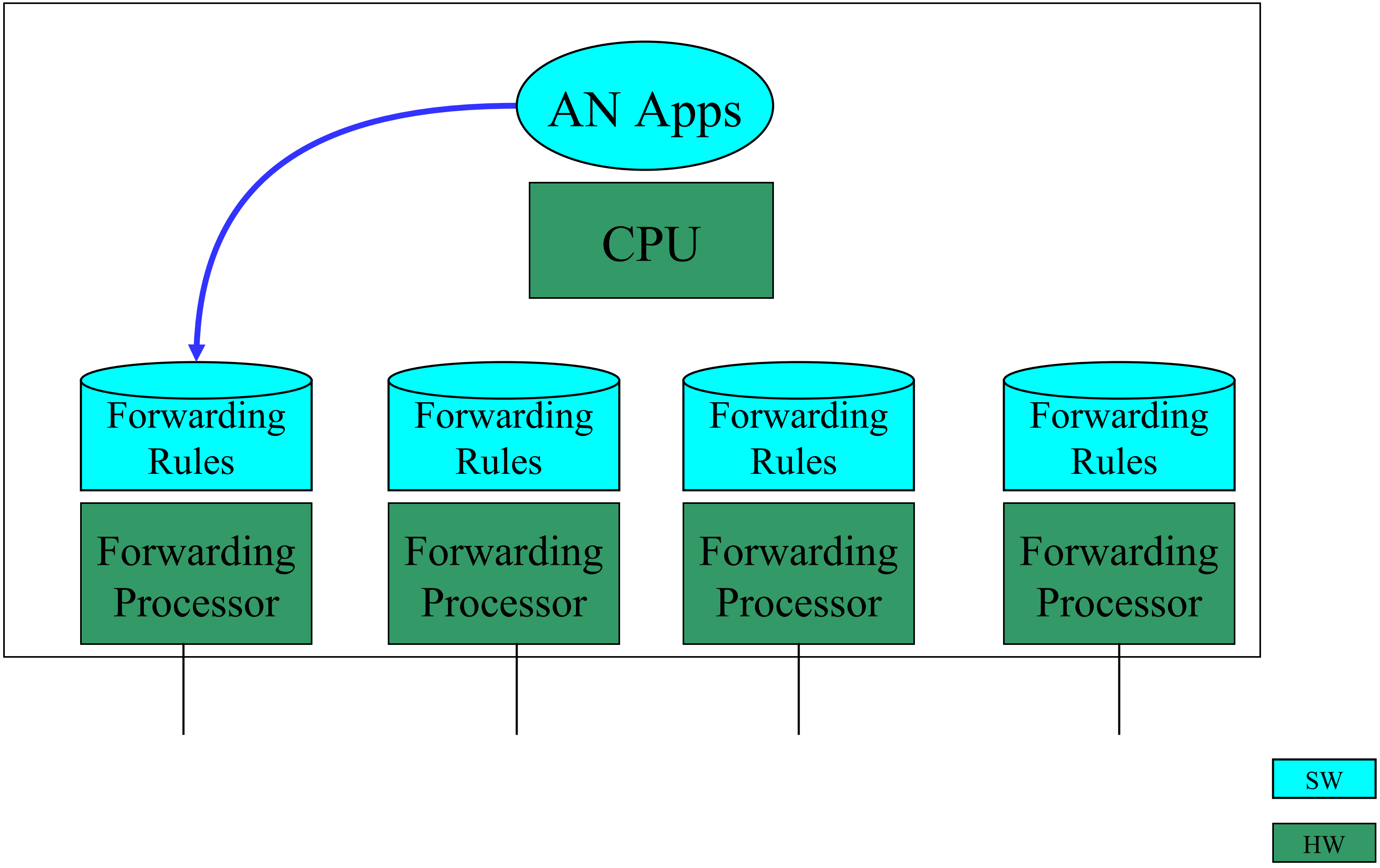
unified policy-based management

Application server

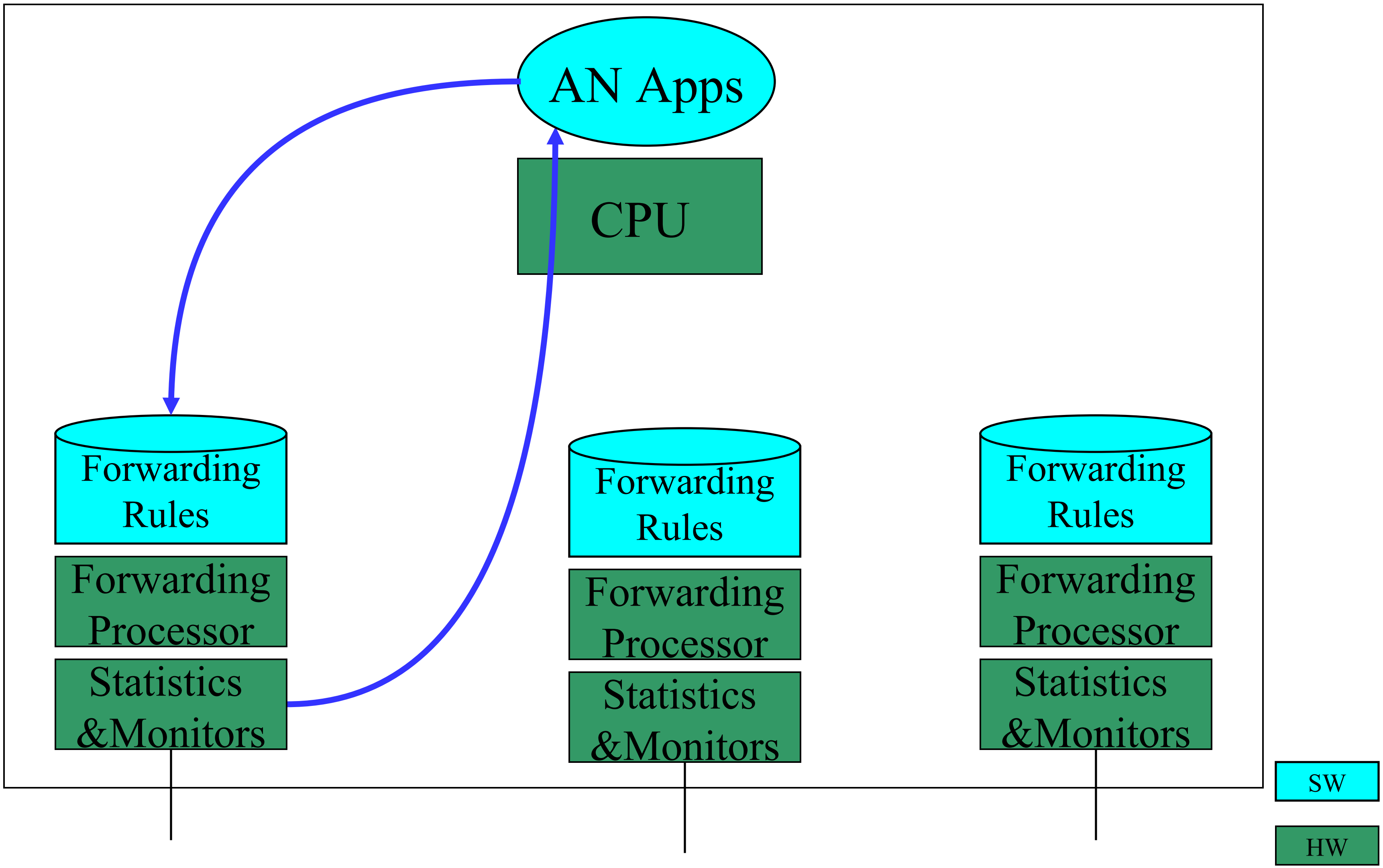
Control element

Forwarding element

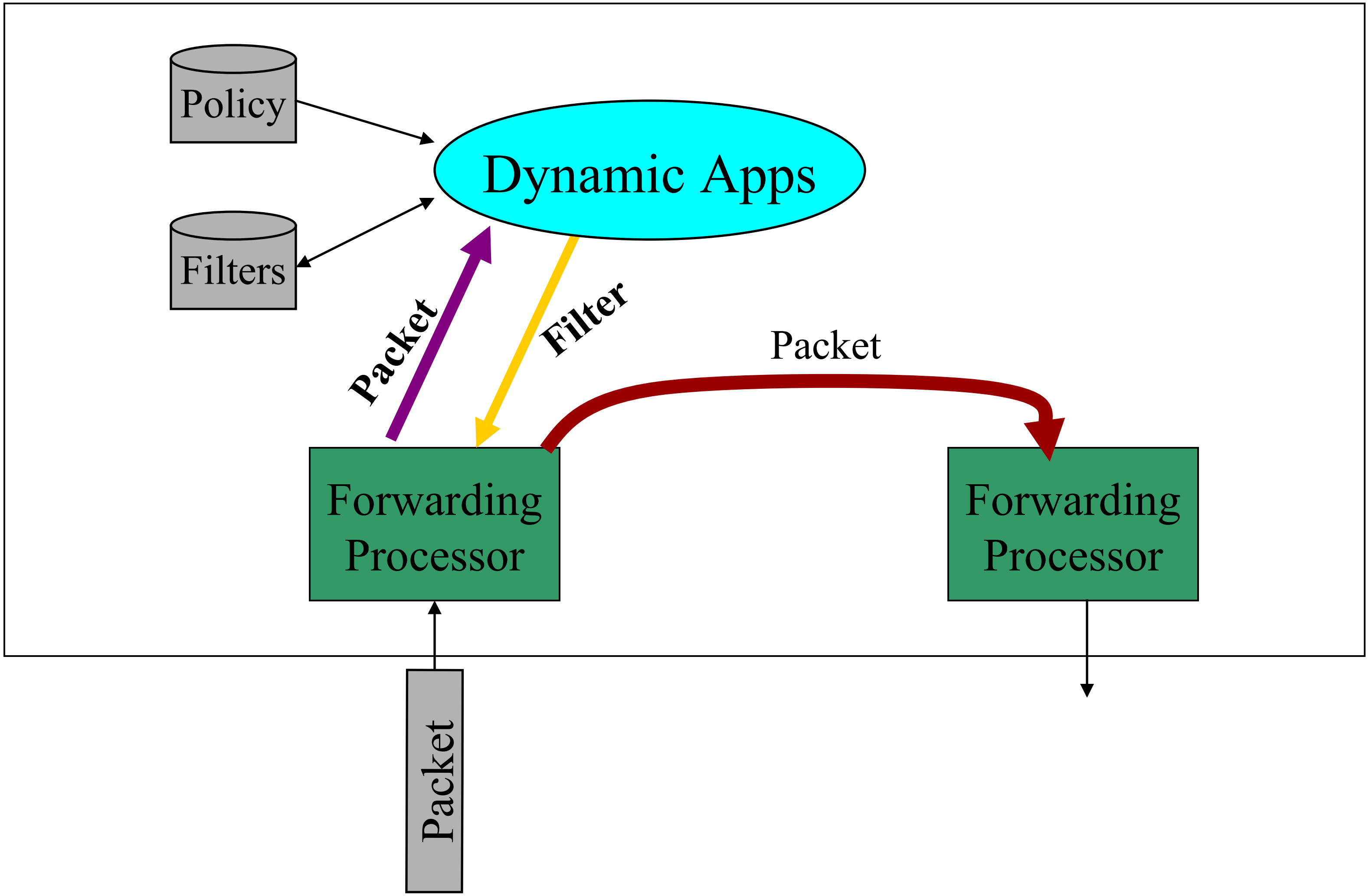
Dynamic Configuration of Forwarding Rules



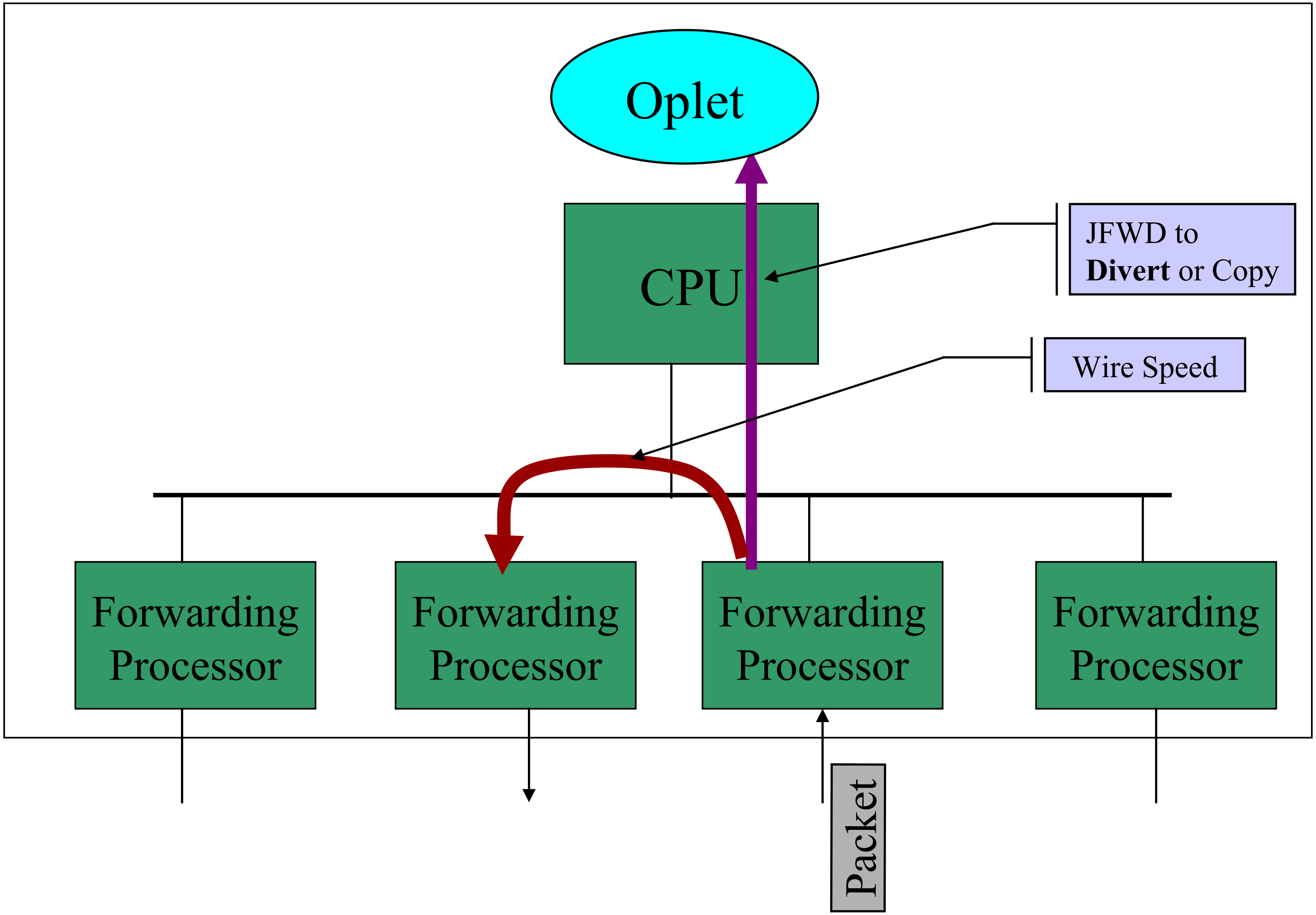
Real-time forwarding Stats and Monitors



Dynamic - On the Fly Configuration



Packet Capture



Java Environment

- **Green Threads -- Present RTOS with single unified task that includes:**
  - Java VM (JVM)
  - Java Resource Manager (JRM)
    - thread scheduling
    - manages CPU utilization
      - JVM time-slice is managed by the JRM preemptive thread scheduler
    - internal memory manager (intercepts "new")
    - garbage collection with priority based on available memory

Non-Interference w/ Single JVM

- **Multiple threads compete for resources**
  - memory
  - CPU
  - persistent storage
- **Denial-of-service attacks possible**
  - memory or CPU consumption attacks
  - trusted/untrusted service interactions

Agenda

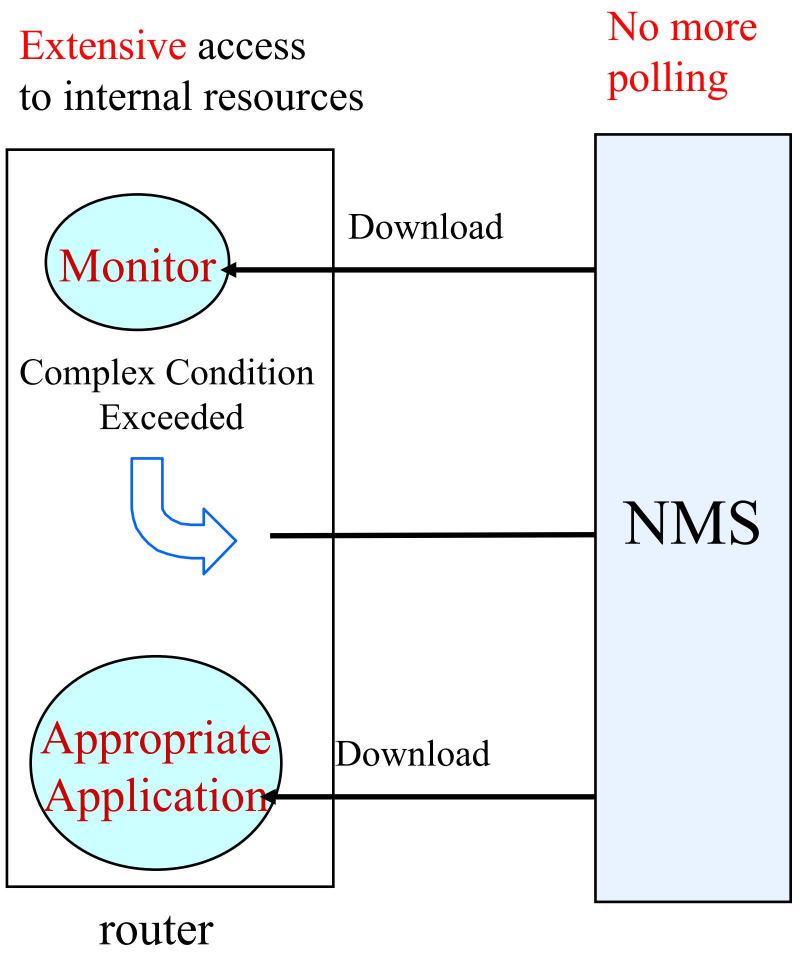
- The market is changing
- Local Computation
- Architecture
- Applications
- ORE - Oplet Run-time Environment
- API's
- Summary

Applications

- Active Network Management
  - Proactive Network Management
  - Diagnostic Agents
- Dynamic DiffServ Classifier
- Active Intrusion Detection
- Multicast Caching
- IP Accounting
- Application-Layer Router-Server Collaboration
- Pseudo Default Drop Capability

Active Network Management

- **Download Oplet Service to the device.**
- **Monitor MIB variables**
  - Might be complex conditions
  - Trend analysis
  - DiffServ, RMON-II, etc… MIBs
- **Report "events" to NMS**
  - drop rate, packets/second
- **Allow Service to take action**
- **Download application**
- **Adjust parameters based on direction from NMS**



An Open Service API Example

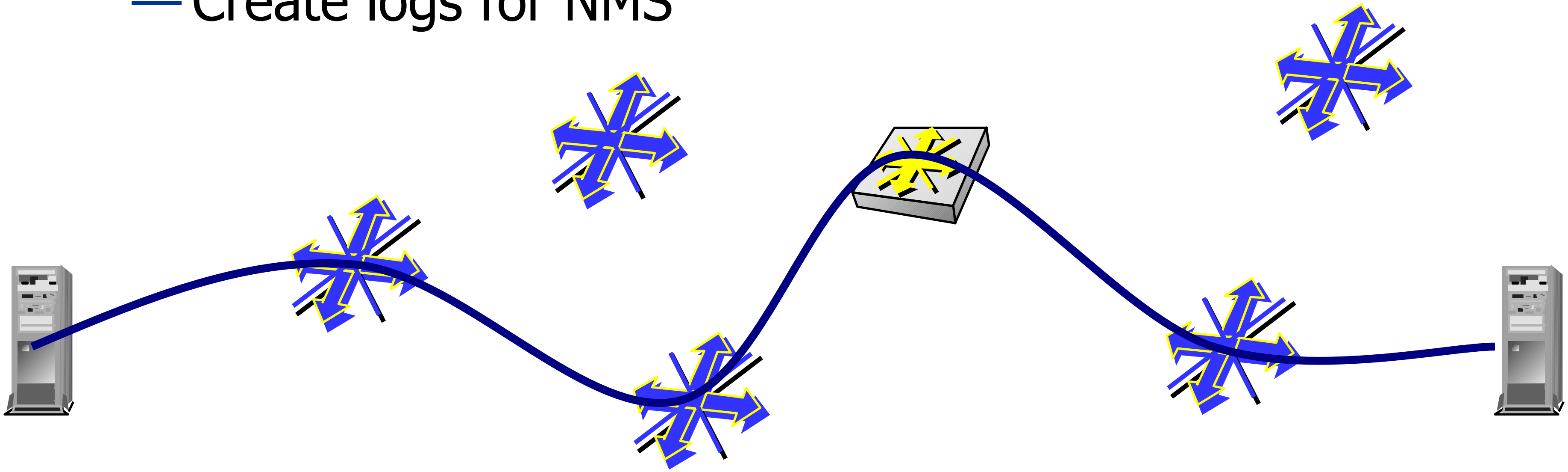
- SNMP API for Network Management
  - generated automatically
  - allows device-based applications to query MIB
  - device-based application -- query local MIB
  - report trends or significant events
  - initiate downloading of problem specific diagnostic code
  - take corrective action

Proactive Network Management

- **Device-based Intelligence is Dynamic**
  - Static Management
    - SNMP set/get mechanisms
    - Telnet, User Interfaces (cli, web, etc…)
  - Dynamic Closed-loop Management at Network Node
    - capable of dealing with new and difficult situations
    - autonomous and rational properties.
    - dynamically system monitoring & modification
    - report status and trends
  - Monitor MIB to identify poor performance and notify NMS prior to failures
  - Downloaded service can instantiated new services

Diagnostic Mobile Agents

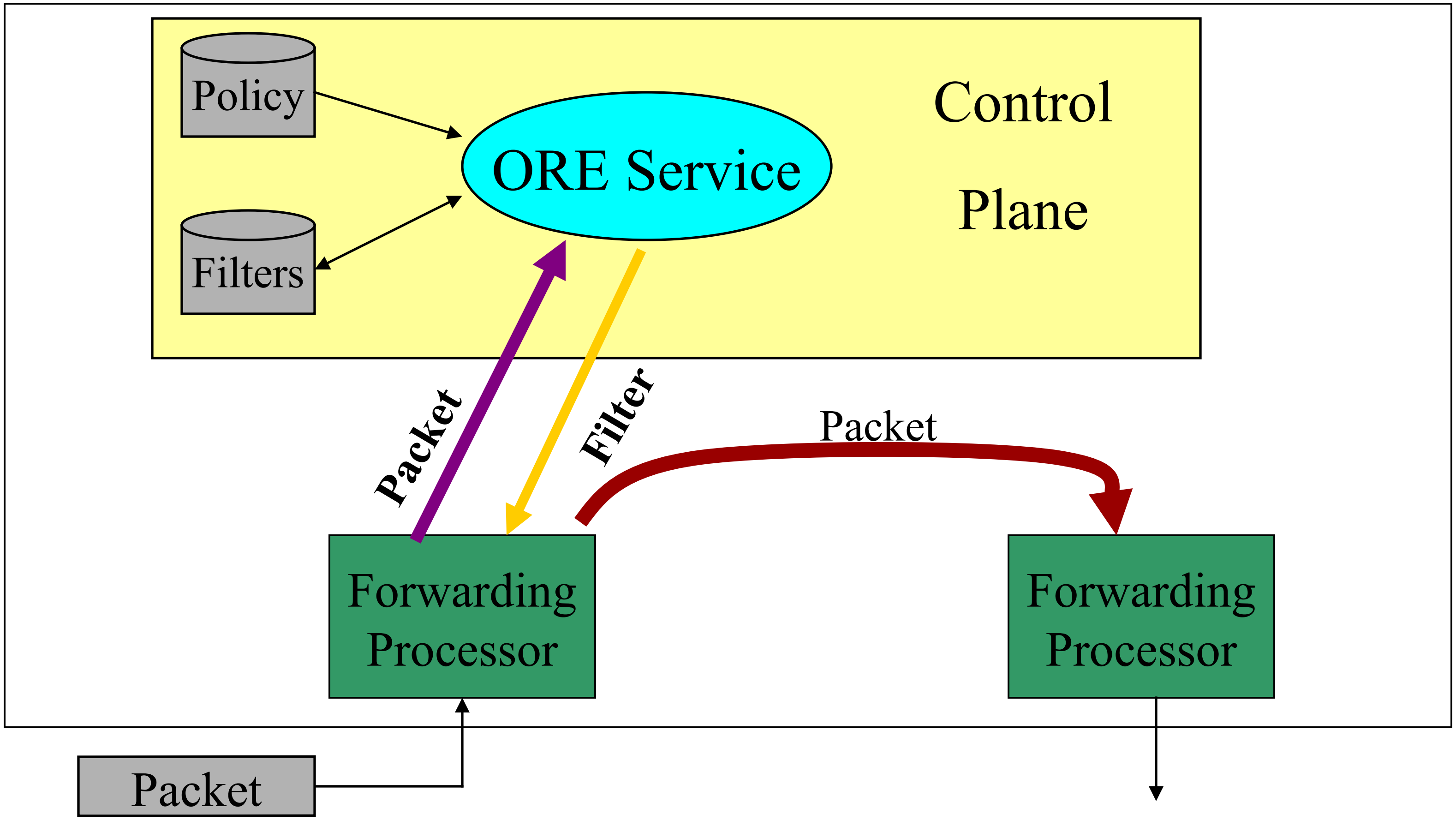
- **Automatic trace-route from edge router where problem exists.**
  - Each node reached generates a report to NMS
  - Trace-route code “moves” to next node in path
  - Mobile agents identify router health
  - Create logs for NMS



Dynamic DiffServ Classifier

- Set router filters to sample packets from edge device host ports
- Identify real-time traffic (RTP flows)
- Set filter on port to adjust DS-byte value based on policy
- Keep track of filters set
- Remove filters no longer in use

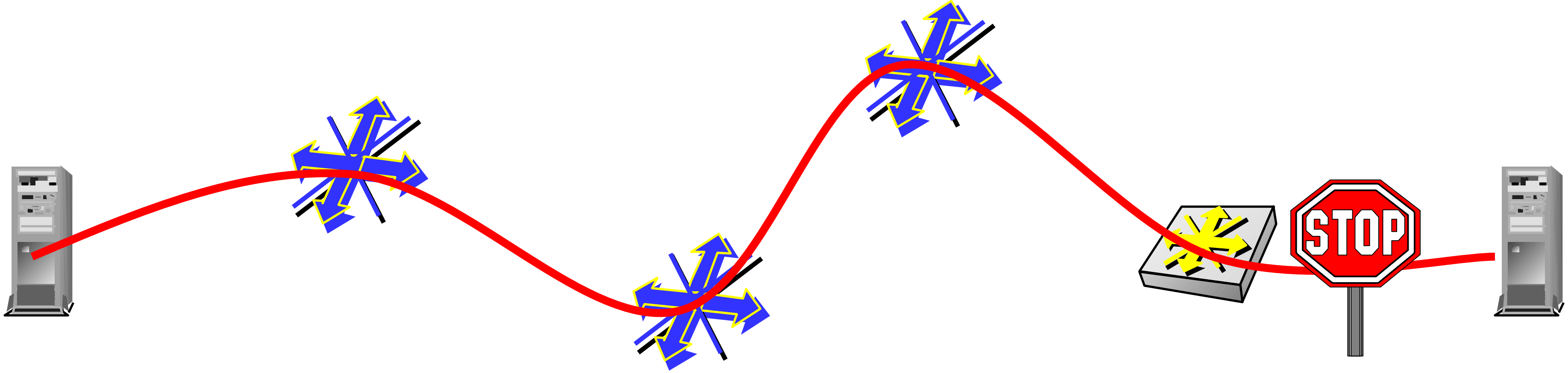
Dynamic DiffServ Classification



- **Sample packets, set filters to modify DS-byte for Per-Hop-Behavior modification**

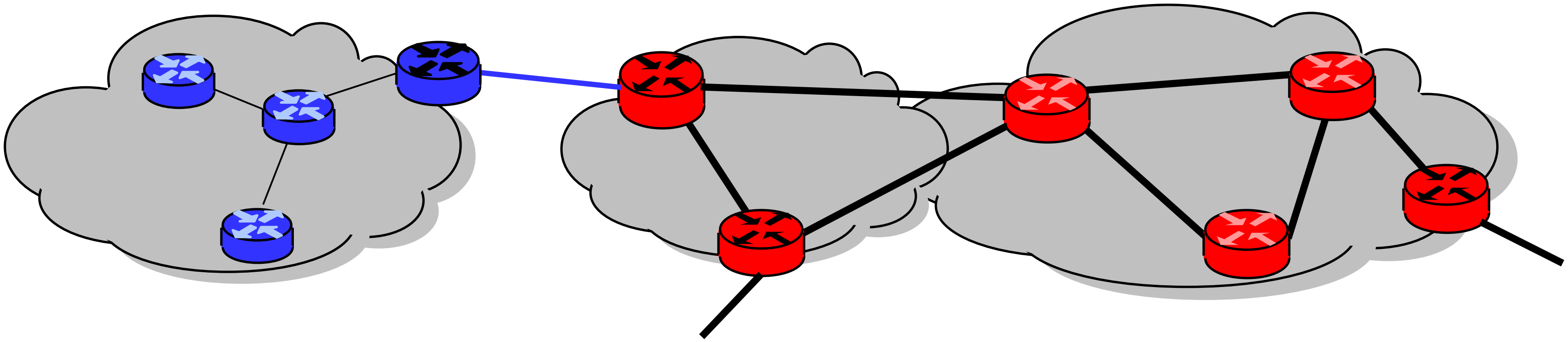
Active Intrusion Detection

- Intruder is identified by Intrusion Detection software
- Intruder signature is identified
- Mobile agent is dispatched in direction of intruder (based on physical port of entry)
- Mobile agent "chases" intruder and terminates him (shuts down link, reboot host, notify NMS)



Multicast Caching

- **Reliable Multicasting**
- **Distribute error control throughout multicast tree**
- **Retransmission a local node keeps control close to lossy links**
- **Balances processor load away from multicast source**

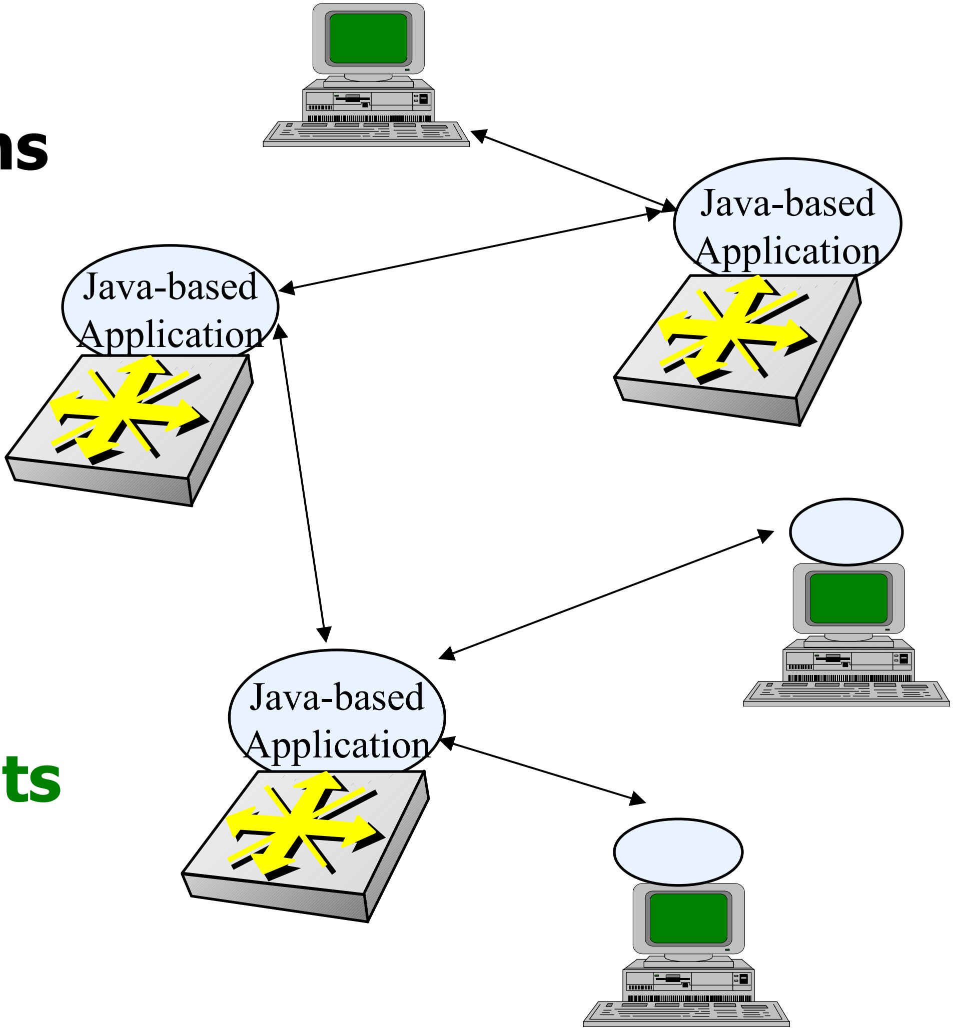


IP Accounting

- Project ABCD (Active Bean Counter in Device)
- Perform usage accounting at edge node
- PreCorrelate/aggregate/reduce accounting record on-site
- $1 rule for billing
- Real-time billing can be realized
- Customize billable resources

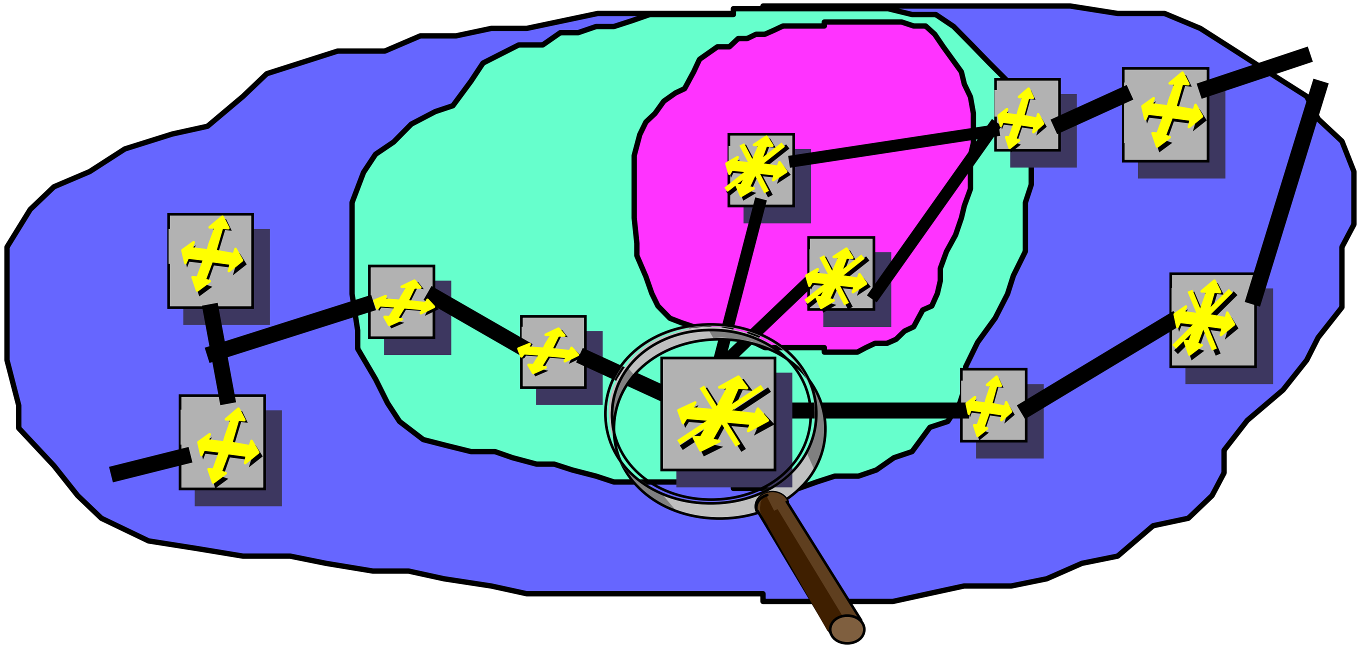
Server Collaboration

- **Supports distributed computing applications in which network devices participate**
  - router to router
  - server to router
- **Supports Intelligent Agents**
- **Supports Mobile Agents**



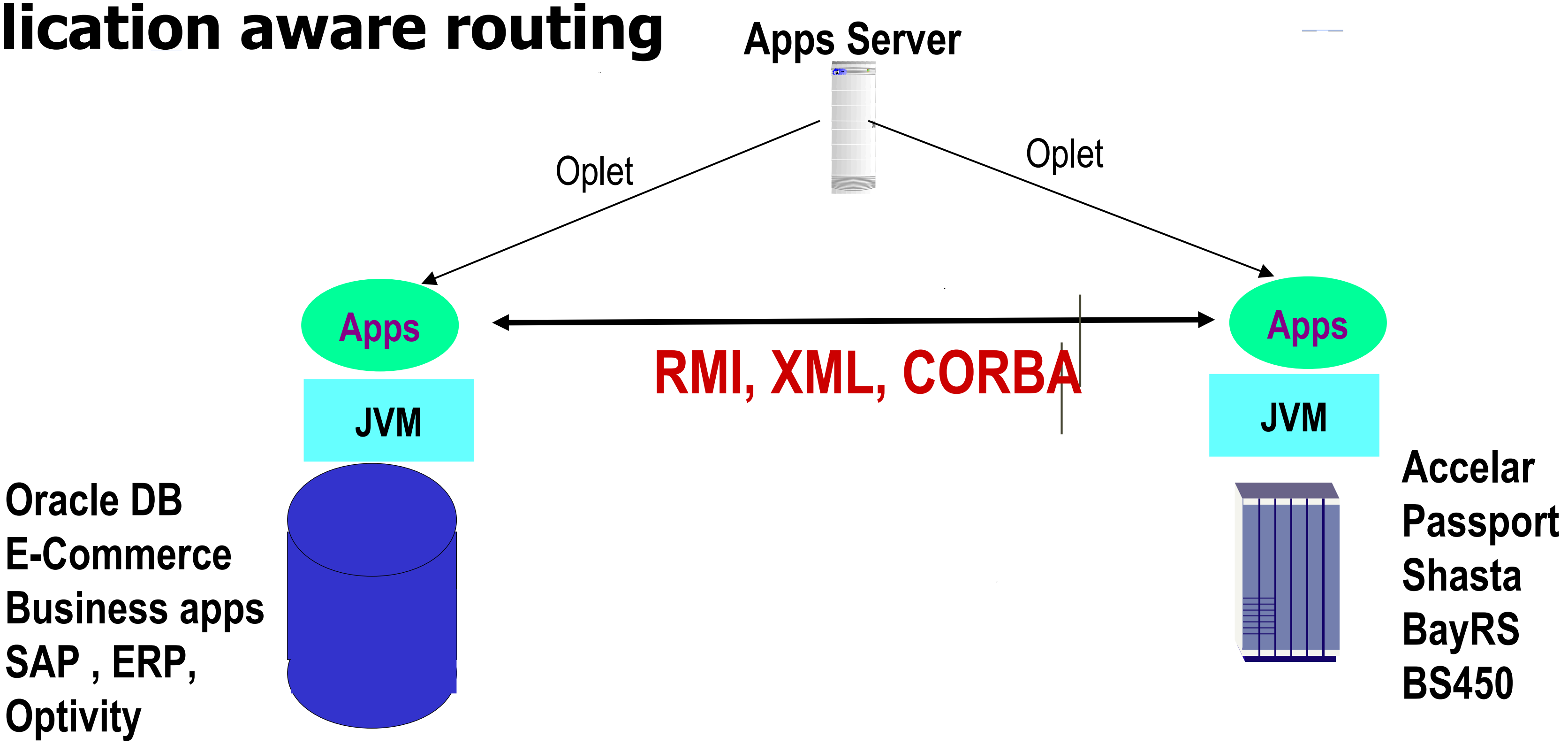
Application Layer Collaboration Among Routers and Servers

- **Server farm load balancing**
  - server state monitored; rerouting based on congestion/load
- **Auctioning Applications**
- **Bandwidth Broker**



Collaboration with Business Applications

- New paradigm of distributed applications
- Network devices collaborating with business applications
- Application aware routing



Bandwidth Broker Collaboration

- Routers Monitor RMON and DIFFSERV MIB
- Report Per-IPAddress, Per Protocol statistic to resource broker
- Adjust DS-byte and Per Hop Behavior based on Bandwidth Broker directions

Dynamic - On the Fly Configuration

- From downloadable Java application, we can modify the behavior of the ASICs

Agenda

- Our market is changing
- Local Computation
- Architecture
- Applications
- ORE - Oplet Run-time Environment
- API’s
- Summary

Oplet Runtime Environment
An Overview

- **A platform to dynamically deploy services on network elements**
- **Desirable properties**
  - Portable to many different devices
  - Secure, reliable
  - Low impact on device performance
  - Open
  - Provide a framework to structure code
    - Reusable, maintainable, robust
- **Implemented in Java**

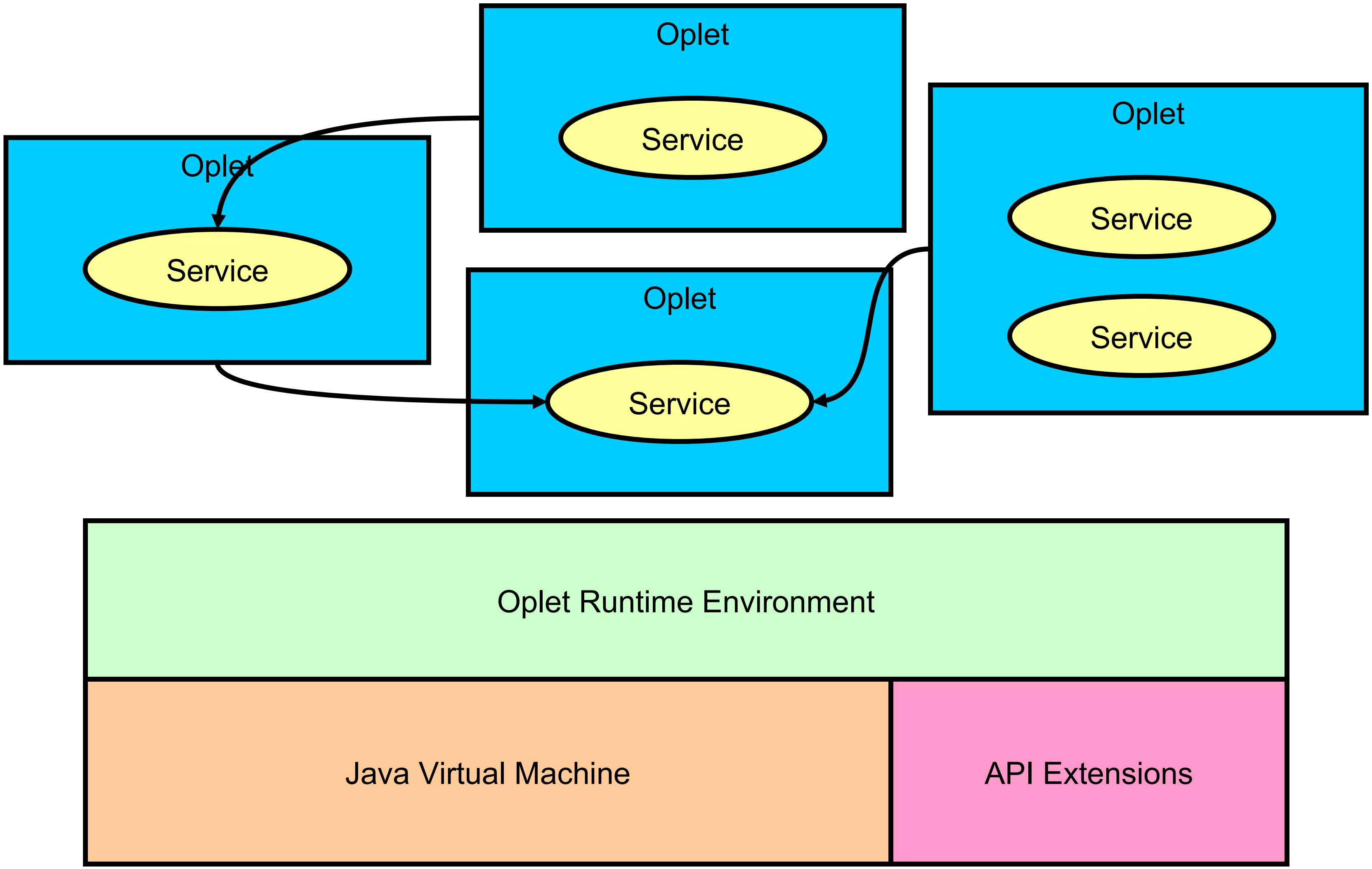
ORE - Basic Concepts

- **Oplet Runtime Environment (ORE)**
  - A kernel that manages the life cycle of oplets and services
  - Provides a registry of services
- **Service**
  - The value being added. Minimal constraints, could be anything…
  - Represented as a Java interface
- **Oplet**
  - The unit of deployment: a JAR file
  - Contains meta-data (eg signatures, dependency declarations)
  - Contains services and other resources (data files, images, properties, JAR files)

Dependencies

- A service S can use facilities provided by another service T
- This means that the oplet containing S has a dependency on service T
- Before an oplet can be started, all of its dependent services must have be started
- ORE manages dependencies and lifecycle of oplets and services

ORE Architecture



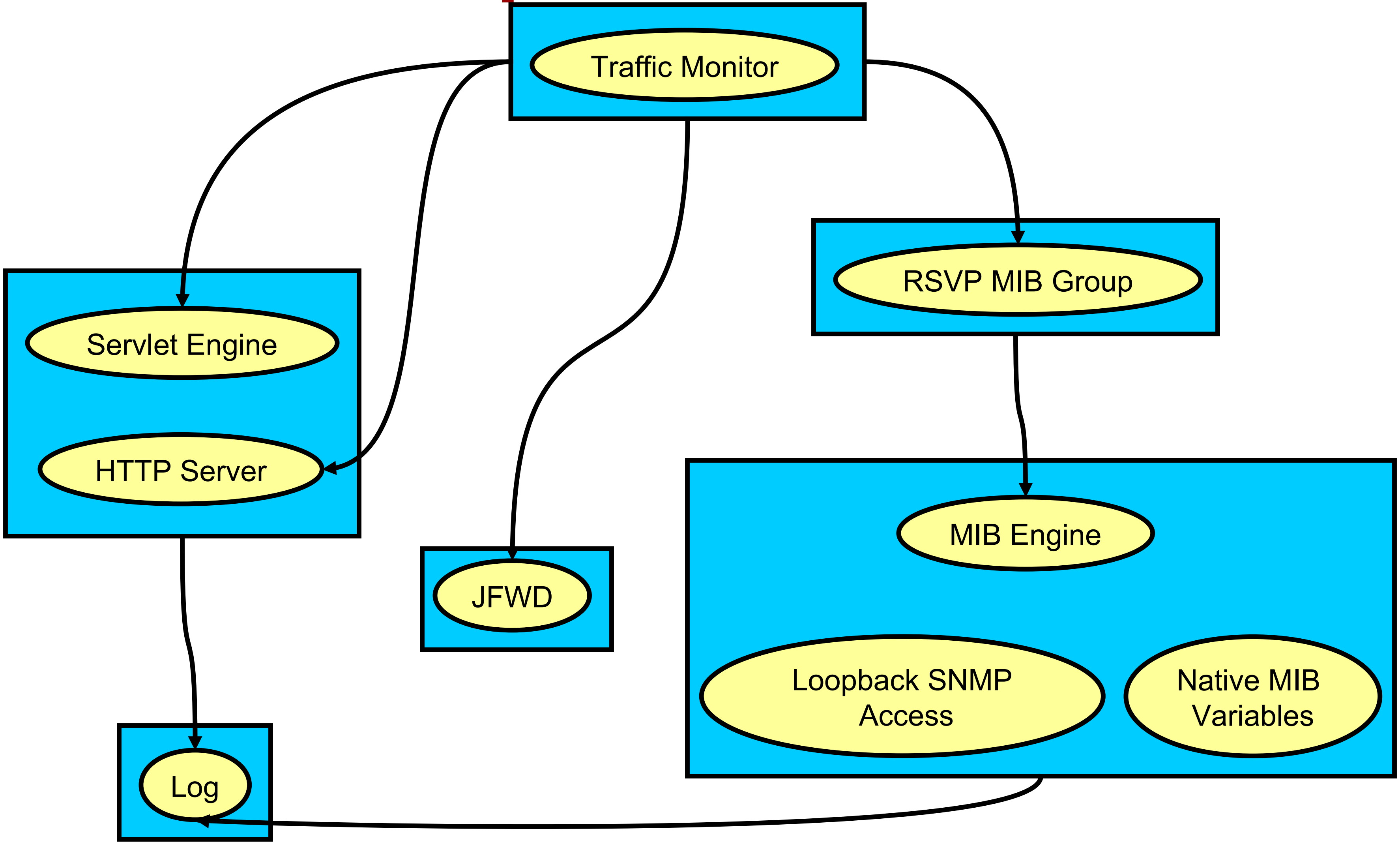
Oplet Lifecycle

- **Install**
  - Loaded from URL
- **Start**
  - Services that are depended on must already be started
- **Stop**
  - Any oplets that depend on this oplet’s services will be stopped
  - Code and data can be unloaded from ORE
- **Uninstall**

Some services

- **Bootstrap**
  - Basic configuration
- **Log**
  - Centralized logging for oplets
- **HTTP server**
  - Simple servlet support
- **Command line shell**
- **Administration commands**
  - Manage oplets and services

An Example


Traffic Monitor
Servlet Engine
HTTP Server
RSVP MIB Group
JFWD
MIB Engine
Loopback SNMP Access
Native MIB Variables
Log

Security Issues

- **Sandbox**
  - Each oplet provides a Java name space and applet-like sandbox
- **Signed oplets**
  - Oplets can be signed for assigning trust
- **Denial of service**
  - Vulnerable to DoS (memory, cycle, bandwidth, peristent storage, monitors) like all Java applications

ORE Status

- **Done now**
  - Runs on Accelar and workstations
  - First release of ORE SDK available internally
- **To be done**
  - More APIs and services (MIB, JFWD, Wrapper)
  - Security (authentication)
  - Oplet updates
  - Persistent storage

ORE Future work

- **Capabilities**
  - Revocable services
- **Security**
  - Java 2 style permissions
- **Resource limits, DoS protection**
  - Probably requires support from JVM
- **Jini, Oplet Directory**
- **Mobile Agents**
- **Open source**

Agenda

- Our market is changing
- Local Computation
- Architecture
- Applications
- ORE - Oplet Run-time Environment
- API’s
- Summary

Initial APIs

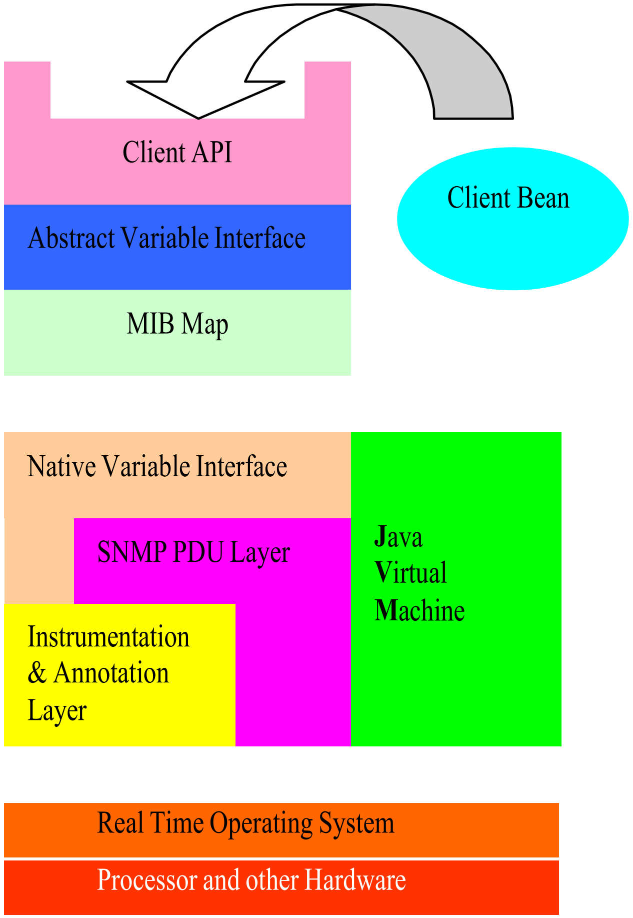
- Console Logging API
- Generic MIB Access API
- Optimized MIB Access APIs
- Trap Interception API
- Management Authentication API
- Web-Based Management Extensions API
- CLI Extensions API
- MIB Extensions API
- Pluggable Authentication API
- Network Forwarding API

Tools

- **MIB API**
  - Monitor device Management Information Base variables
    - MIB
    - RMON and RMON-II
    - DiffServ
- **Network API (JFWD)**
  - Interface to Filters
    - set packet drop filters
    - intercept packets
    - carbon copy packets while forwarding at line-speed

MIB API Example

- API uses a MIB Map to dispatch requests to variable access routines
- Different parts of the MIB tree can be serviced by different mechanisms
- Two main schemes:
- An ad hoc interface to the SNMP instrumentation layer
- A generic SNMP loopback

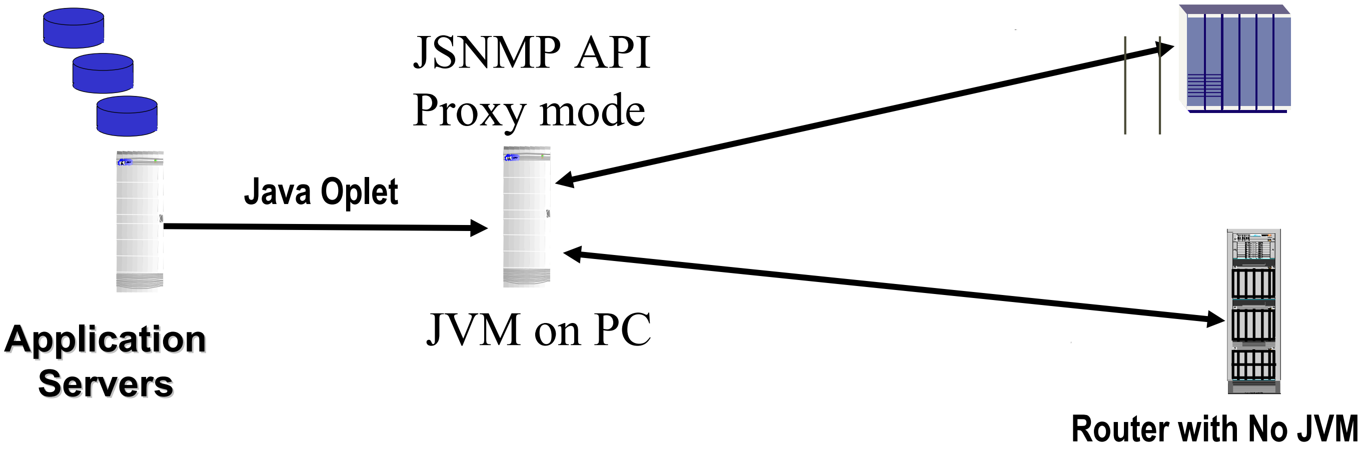

Client API
Abstract Variable Interface
MIB Map
Client Bean
Native Variable Interface
SNMP PDU Layer
Java Virtual Machine
Instrumentation & Annotation Layer
Real Time Operating System
Processor and other Hardware

An Open Service API Example

- SNMP API for Network Management
  - generated automatically
  - allows device-based applications to query MIB
  - device-based application -- query local MIB
  - report trends or significant events
  - initiate downloading of problem specific diagnostic code
  - take corrective action

Java MIB API - Proxy Mode

- **Uses SNMP loopback mechanism to target a remote network element**
- **API can be used to control devices that don't have an embedded JVM**



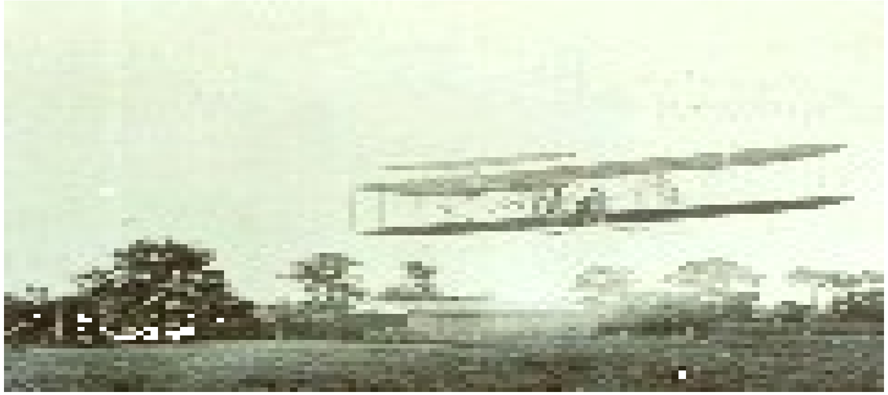
Agenda

- Our market is changing
- Local Computation
- Architecture
- New types of applications
- ORE - Oplet Run-time Environment
- API's
- Summary

Summary

- **JOOSE - OS of Choice for Open Routers**
- **Programmable**
  - Turing Machine on network devices
  - *dynamic* agents vs. *static* agents
  - dynamic loading
- **Our market is changing**
- **Openness - successfully proven paradigm**
  - Facilitates innovation
  - Domain experts - virtual development community
  - With 3rd parties we can change the networking landscape
- **Application aware routing**

This is only the first step



1903 the Wright brothers

Compare to this first flight and look where aviation is today

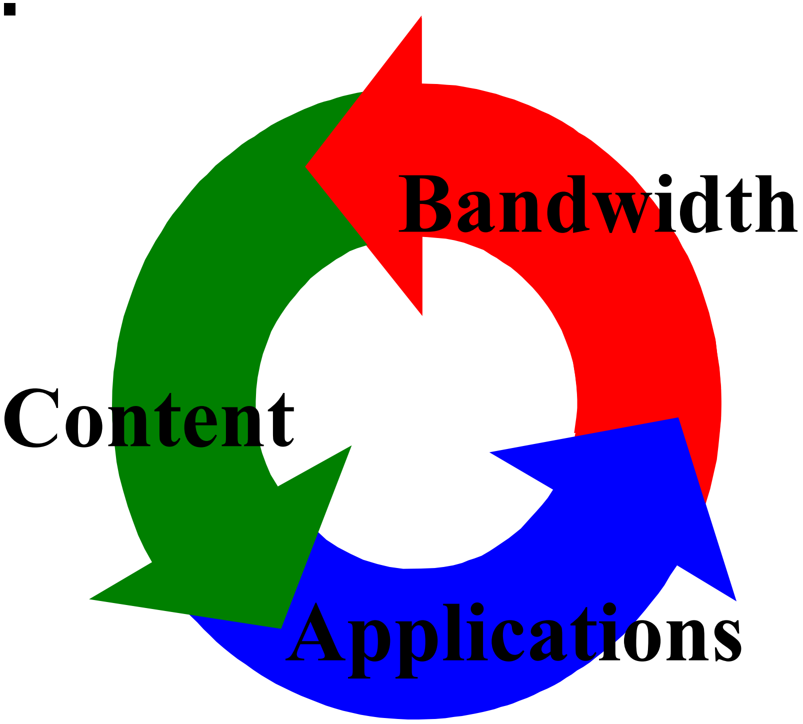
Appendix

Why Java

- **Dynamic Loading**
- **Reuse security mechanisms**
  - byte-code verifier
  - security manager
  - classloader
- **System stability**
  - constrains applications to the JVM
  - Prohibits native code applications
- **Extensible, portable, & distributable services**

Bandwidth x200 - start of new demand

- Intel web hosting - BIG pipes
- Last mile bandwidth x 200
- Multimedia and new applications will drive the demand.

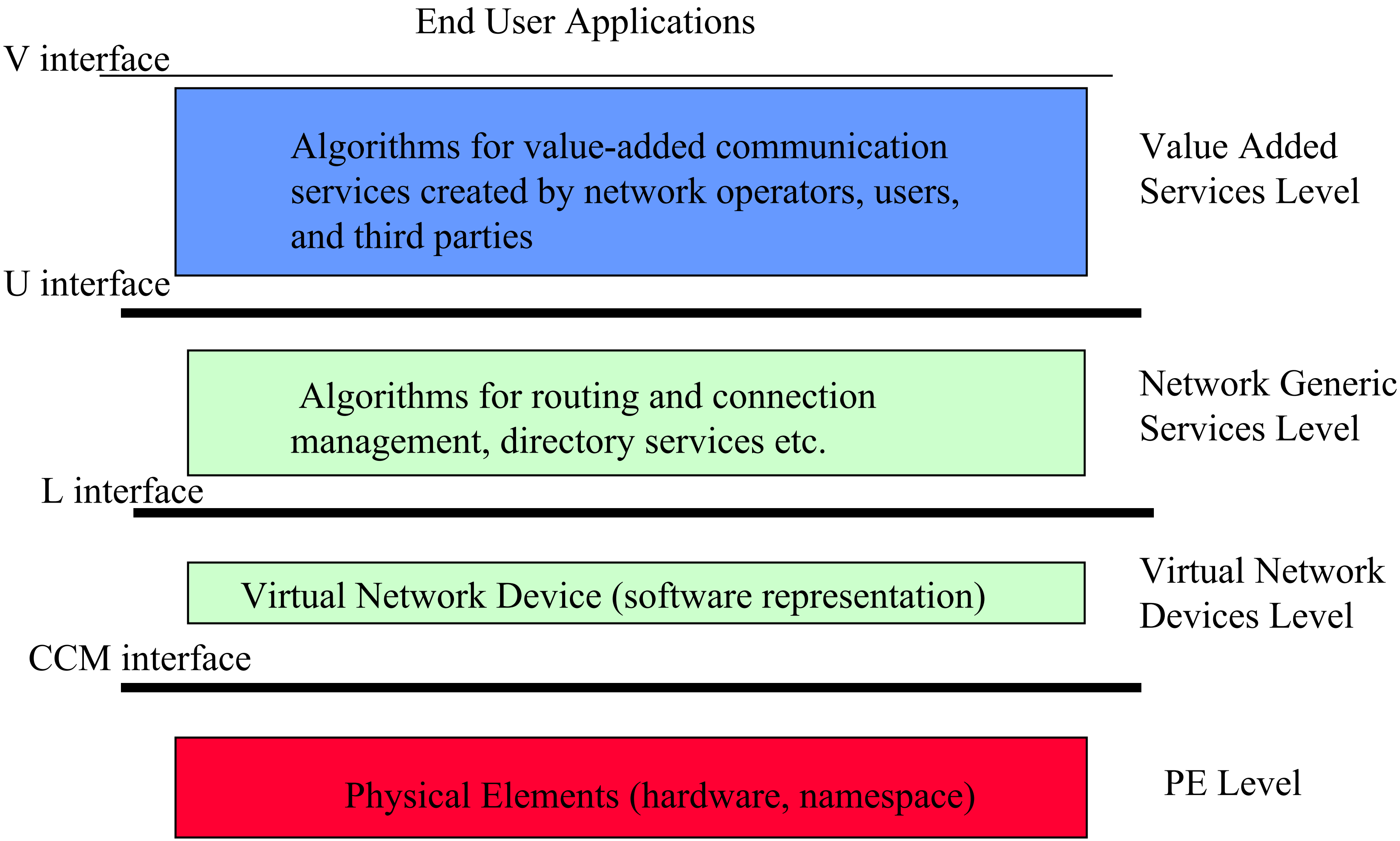


Bandwidth

Content

Applications

The P1520 Reference Model



End User Applications

V interface

Algorithms for value-added communication services created by network operators, users, and third parties

Value Added Services Level

U interface

Algorithms for routing and connection management, directory services etc.

Network Generic Services Level

L interface

Virtual Network Device (software representation)

Virtual Network Devices Level

CCM interface

Physical Elements (hardware, namespace)

PE Level

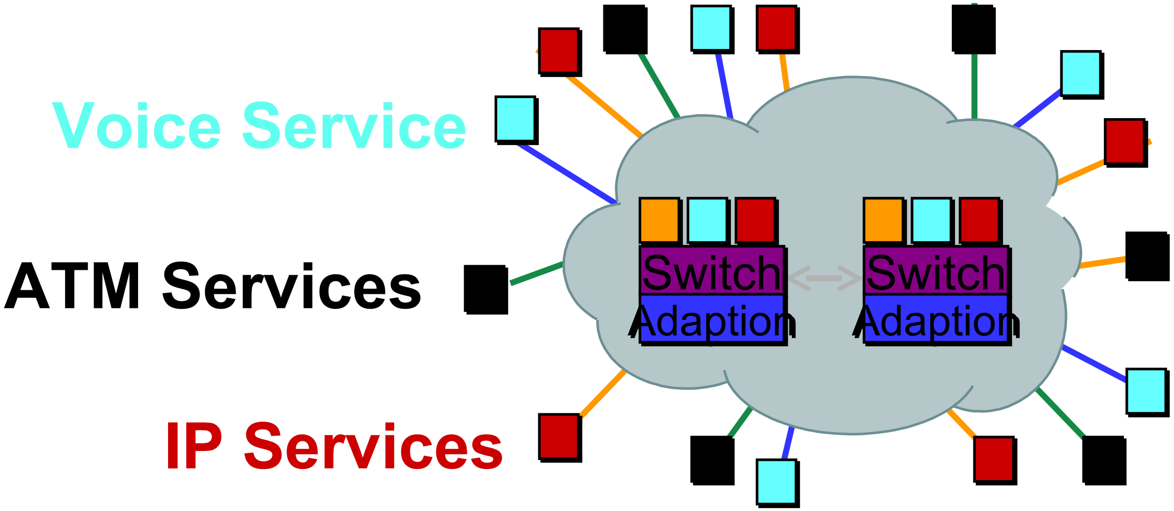
CSIX Consortium

- **Common switch interface for switch fabric independence**
  - www.csix.org
  - Detailed interface specification between port/packet processor logic and interconnect fabric logic
  - Similar to common media interface such as Utopia, but for switch fabric interface
  - Targeted at scalable switches at higher end
  - Permits mix-and-match of silicon and software components

Multi-Services Switching Forum (MSF)

www.msforum.org

- **Open Multi-service Switching**
  - Common transmission and switching infrastructure
  - Modular, layered architecture
  - Integration at a module level through open interfaces
  - Multi-vendor model with 3rd party software options



Source: MSS