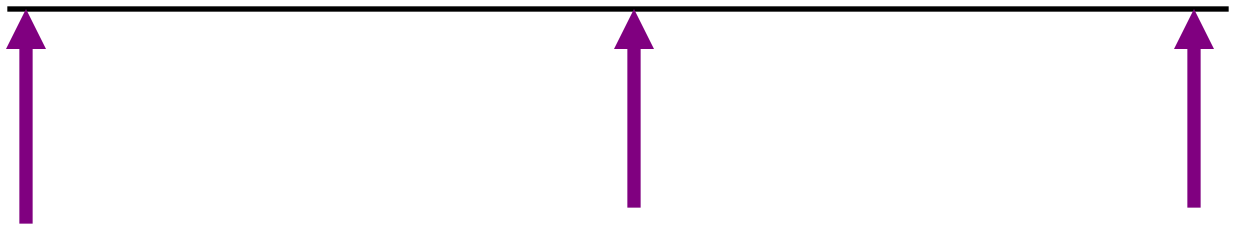
Integrating Active Networking and Commercial-Grade Routing Platforms

The University of Maryland

Rob Jaeger
(rfj@cs.umd.edu)

J.K. Hollingsworth
Bobby Bhattacharjee

The Network Paradigm Spectrum



Traditional Networks

- end-to-end connectivity
- well defined protocols
- increasingly perform forwarding in hardware

?

Active Networks

- on-the-fly service introduction
- per-flow granularity possible
- inject software in data path

Objectives

- Implement flow performance enhancement mechanisms **without** introducing software into data forwarding path
  - **Service defined packet processing in a silicon-based forwarding engine**
  - **Policy-based Dynamic packet classifier**
- Create OPEN platform for introduction of new services
  - **Specify OPEN interfaces for Java applications to control a generic, platform-neutral forwarding plane**
  - **Enable downloading of services to network node**
  - **Allow object sharing and inter-service communication**

Accomplishments

- JVM on a Silicon-Based Routing Switch
- ORE - Oplet Run-time Environment
  - Java-enabled platform for secure downloading and safe execution of services
  - Ensures required services are installed for a downloaded Oplet
- Java SNMP API (proxy mode for non Java devices)
- Implementation of Network Forwarding API (JFWD)
- RESULT: **Dynamic** Classification in Silicon-Based forwarding engine on a Gigabit Routing Switch

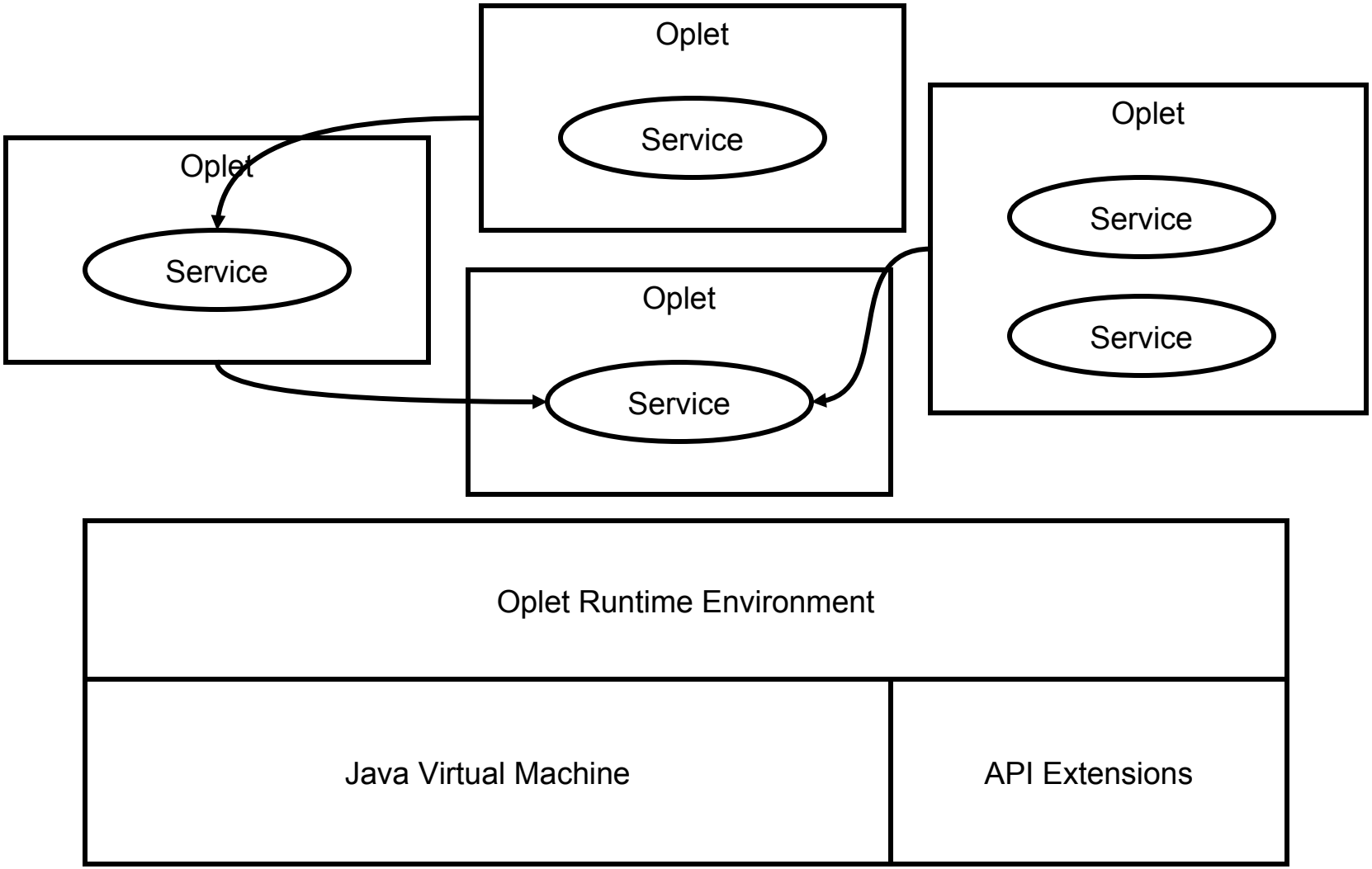
Oplet Runtime Environment Overview

- **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**

Basic Concepts

- **Oplet Runtime Environment (ORE)**
  - A kernel that manages the life cycle of oplets and services
  - Provides a registry of services
- **Services**
  - The value being added. Minimal constraints
  - Represented as a Java interface
- **Oplets**
  - 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)

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**

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 been started
- ORE manages dependencies and lifecycle of oplets and services

Some services

- **Bootstrap (ORE start time) - basic configuration**
- **Log - Centralized logging for oplets**
- **HTTP server**
  - Simple servlet support
- **Command line shell -**
  - service depends on shell to register commands
- **Administration commands -**
  - Manage oplets and services
- **Access to router resource including hardware instrumentation via JMIB**

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
  - resource management is a problem

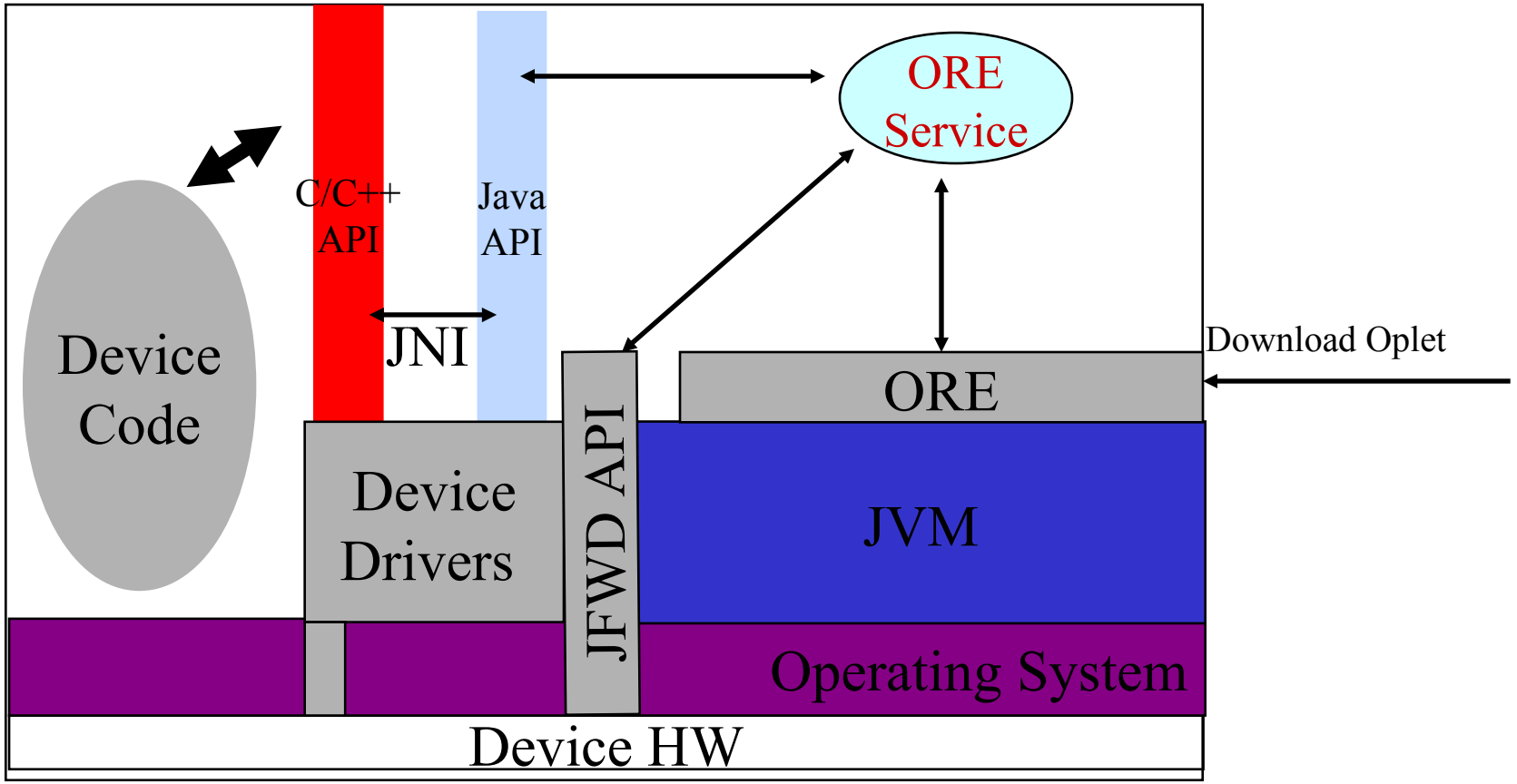
ORE Status

- **Done now**
  - Runs on several Nortel routing products
  - Run on workstations
  - First release of ORE SDK complete
  - JMIB monitor/control system through MIBs
  - JFWD

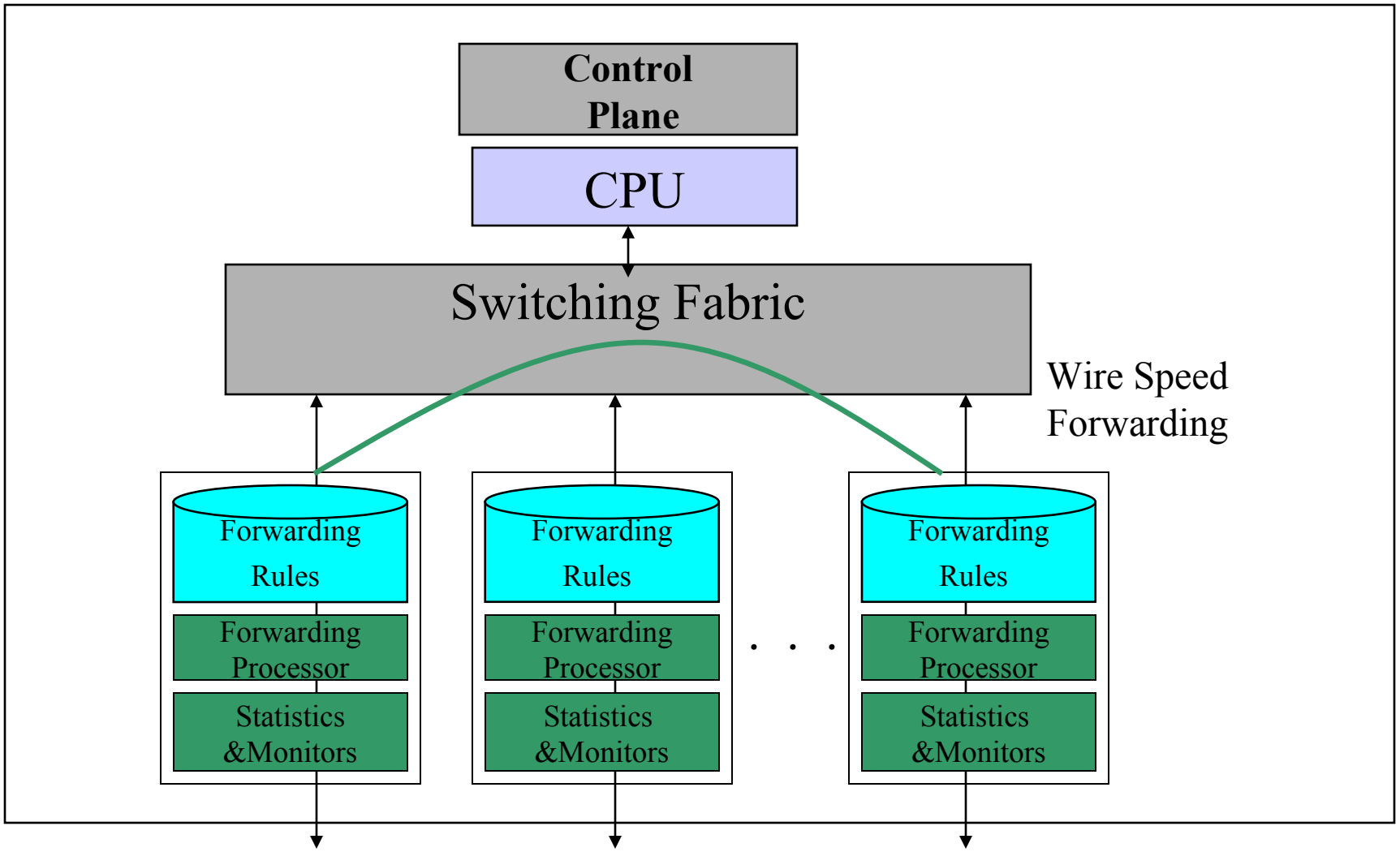
Future ORE work

- **Capabilities**
  - Revocable services
- **Security**
  - Java 2 style permissions to perform operations
- **Resource limits, DoS protection**
  - Probably requires support from JVM
- **Jini, Oplet Directory - locate and load services**
- **Agents/Services**
- **Open source**

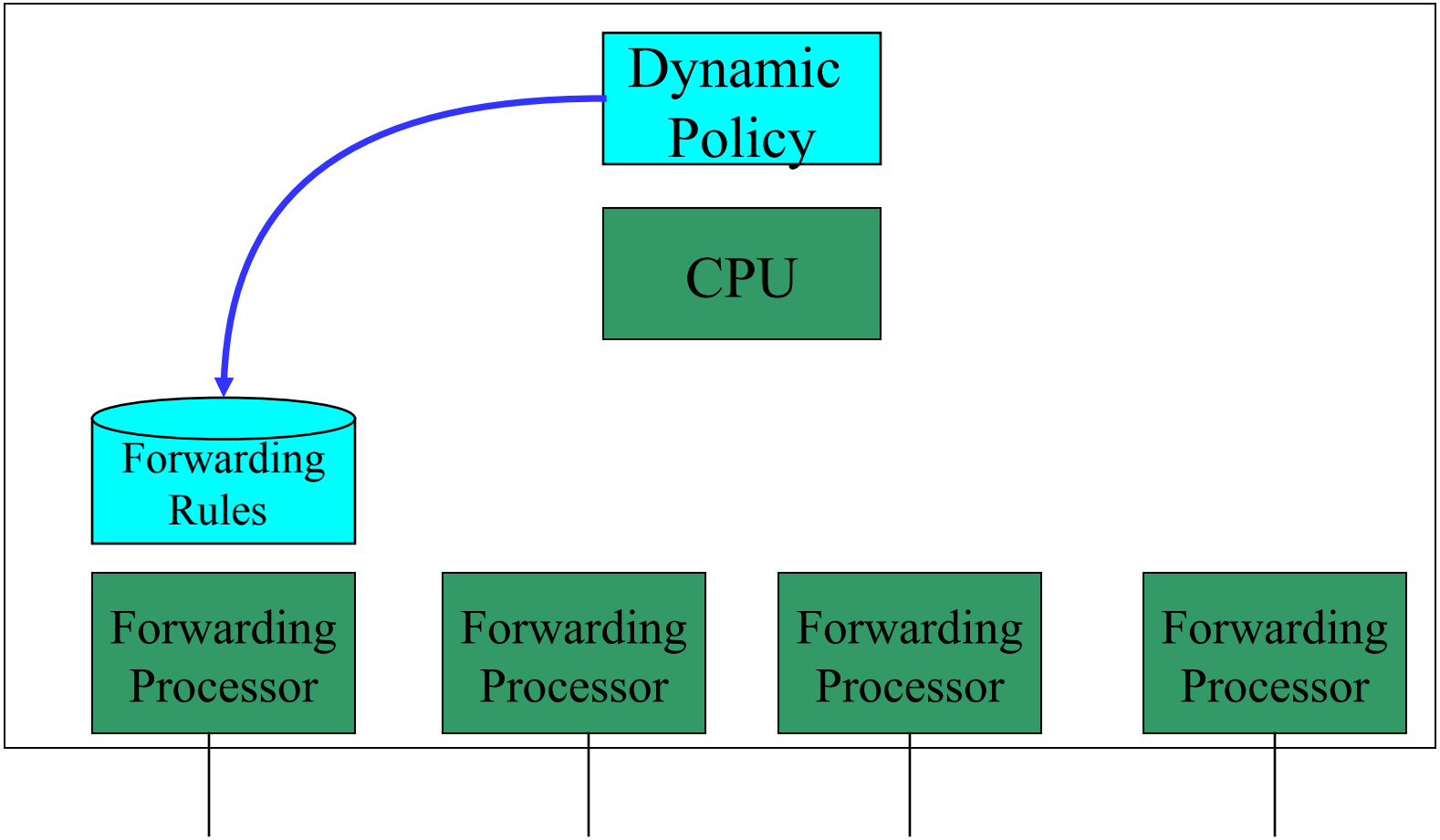
Open Device Architecture



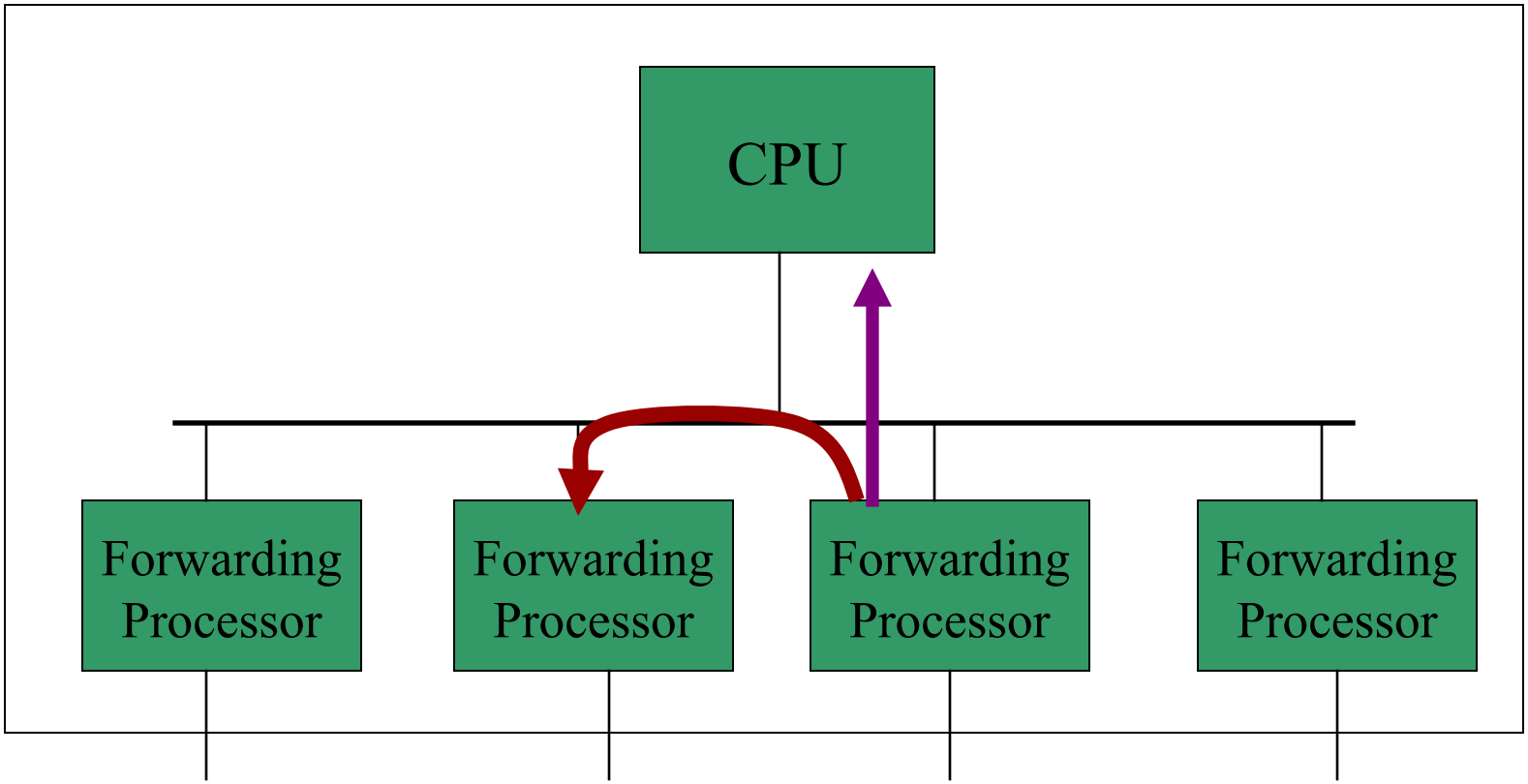
Silicon-based Forwarding Engines



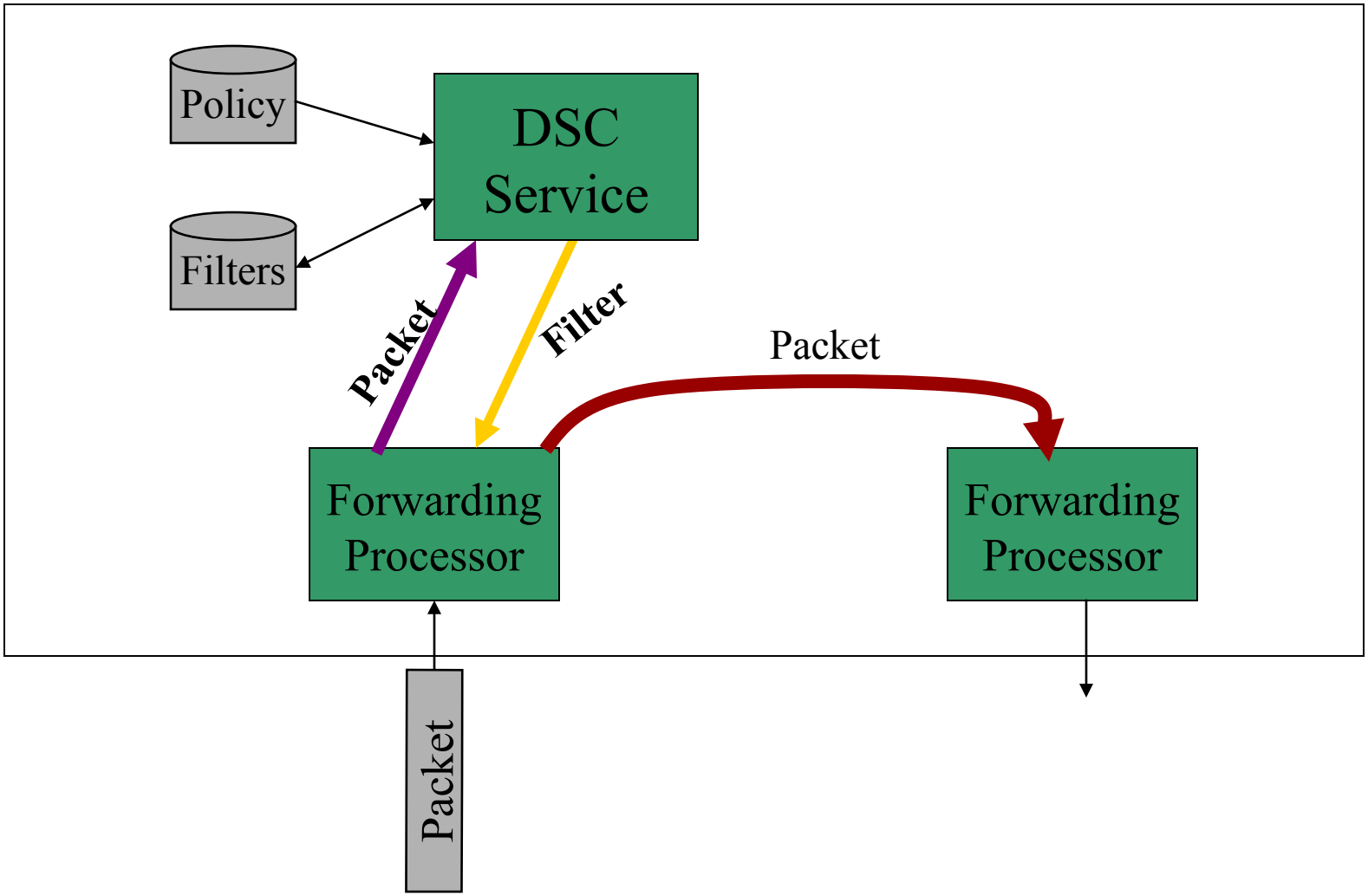
Dynamic Configuration of Forwarding Rules



CarbonCopy Capability



Dynamic Packet Configuration



Dynamic Classification

- Identify real-time flows (e.g. packet signature/flowId)
  1. Use CarbonCopy filters to deliver multimedia control protocols to control plane
     - e.g. SIP, H.323. RTSP
     - Determine dynamically assigned ports from control msgs
  2. Use CarbonCopy filters to sample a number of packets from the physical port and identify RTP packets/signature
- Set a packet processing filter for packet signature to:
  - adjust DS-byte OR
  - adjust priority queue

JFWD 5-tuple Filtering

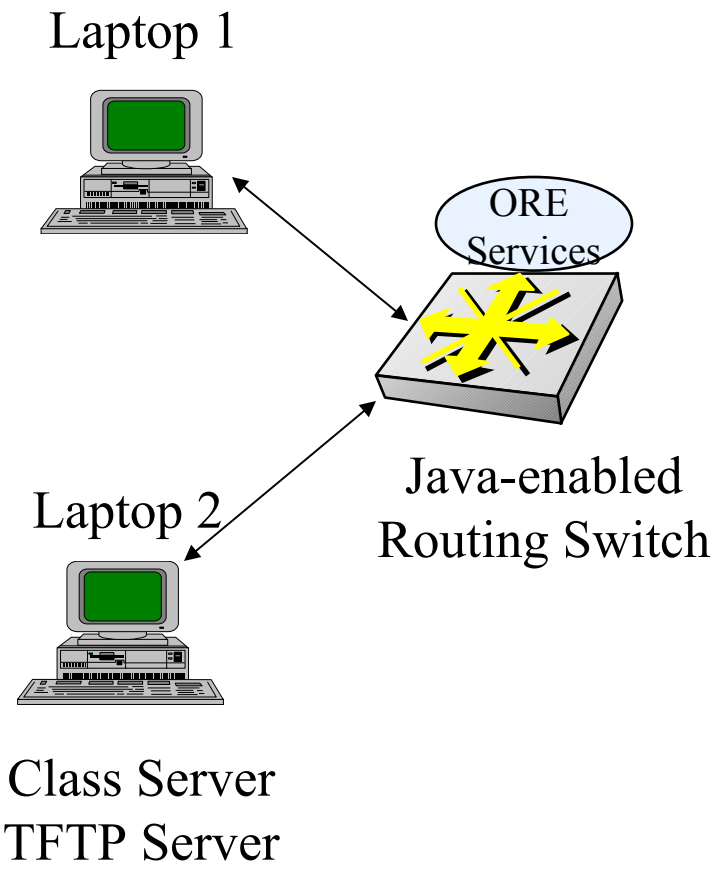
- copy the packet to the control plane
- don't forward the packet
- set TOS field
- set VLAN priority
- adjust priority queue

ANTS on Gigabit Router

Demo - 1

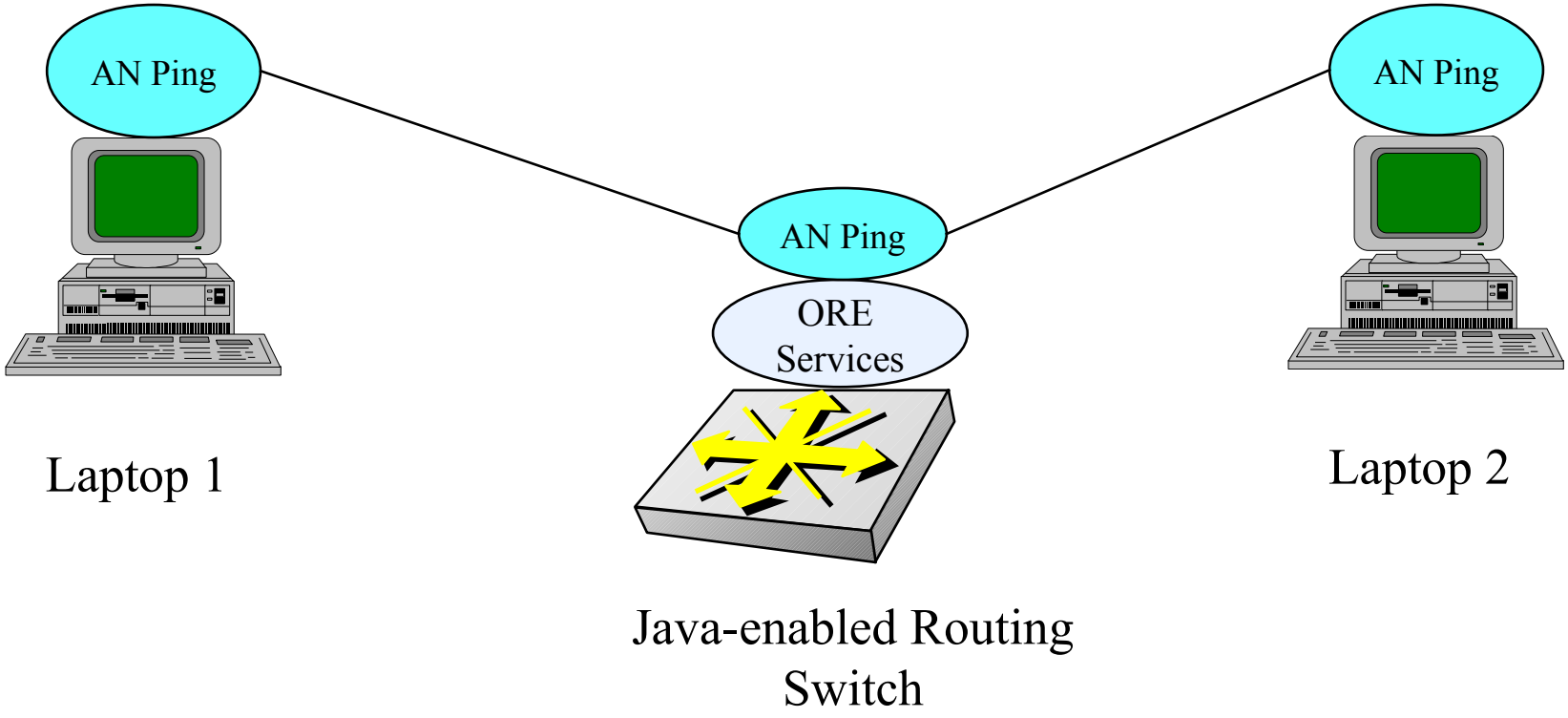
ANTS Demo Configuration

- RoutingSwitch loads boot image from TFTP server
- RoutingSwitch dynamically loads Oplets from the Class Server
    - Laptop 1 originates the ping
- Router gets Ping code from Laptop 1.
- Router "evaluates" ping
- Ping forwarded to Laptop2
- Laptop 2 requests code
- Laptop 2 perform ping reply

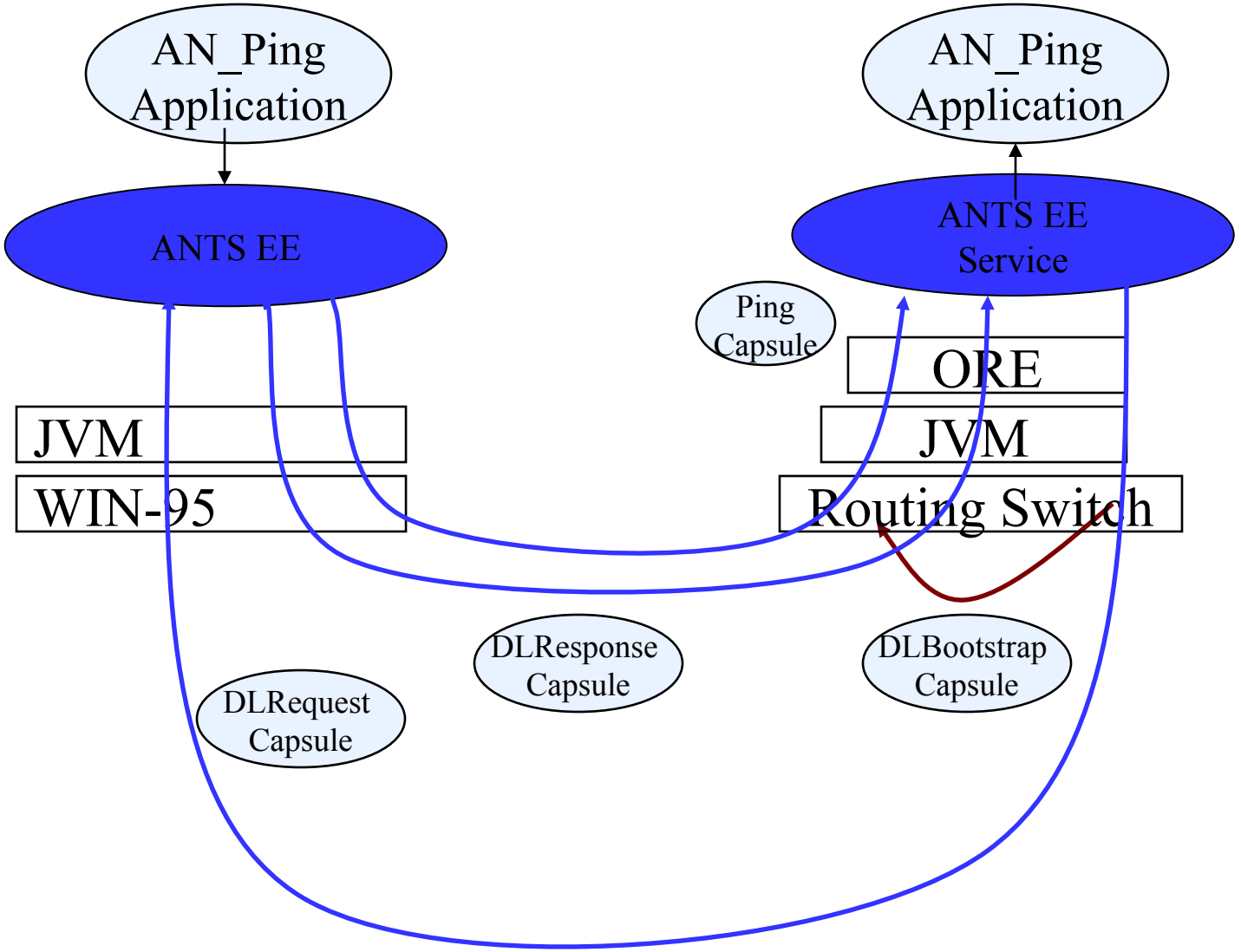


ANTS Demo

Demo 1



ANTS Demo


AN_Ping Application
AN_Ping Application
ANTS EE
ANTS EE Service
Ping Capsule
ORE
JVM
JVM
WIN-95
Routing Switch
DLResponse Capsule
DLBootstrap Capsule
DLRequest Capsule

ANTS Demo

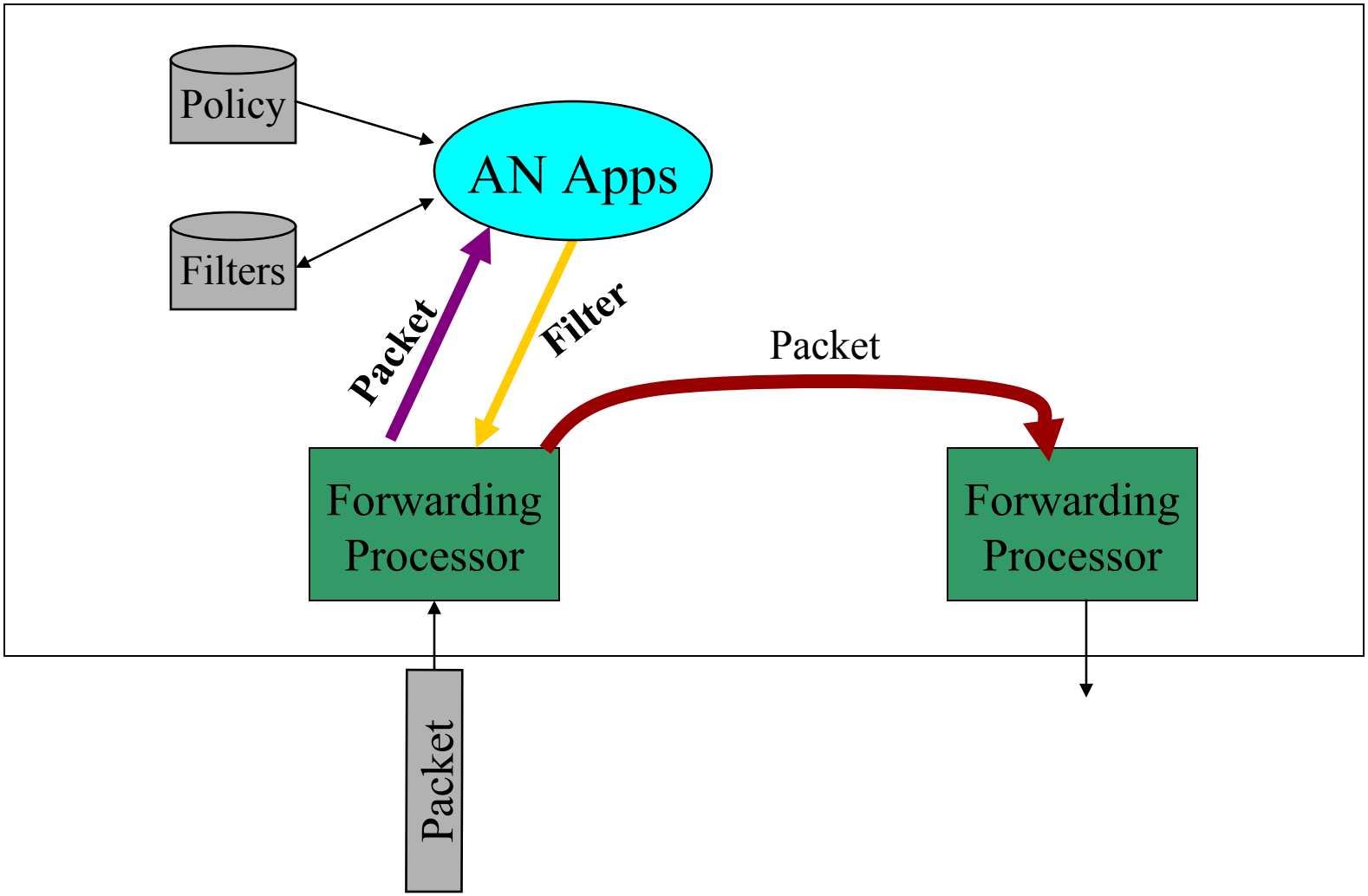
- Java application running on the router
- ORE facilitate downloading services
- Interoperable with ANTS Distribution
- Minimum changes to make it conform to ORE service specification

Dynamic Filtering & Configuring

Demo - 2

Demo 2

Dynamic - On the Fly Configuration



Dynamic - On the Fly Configuration

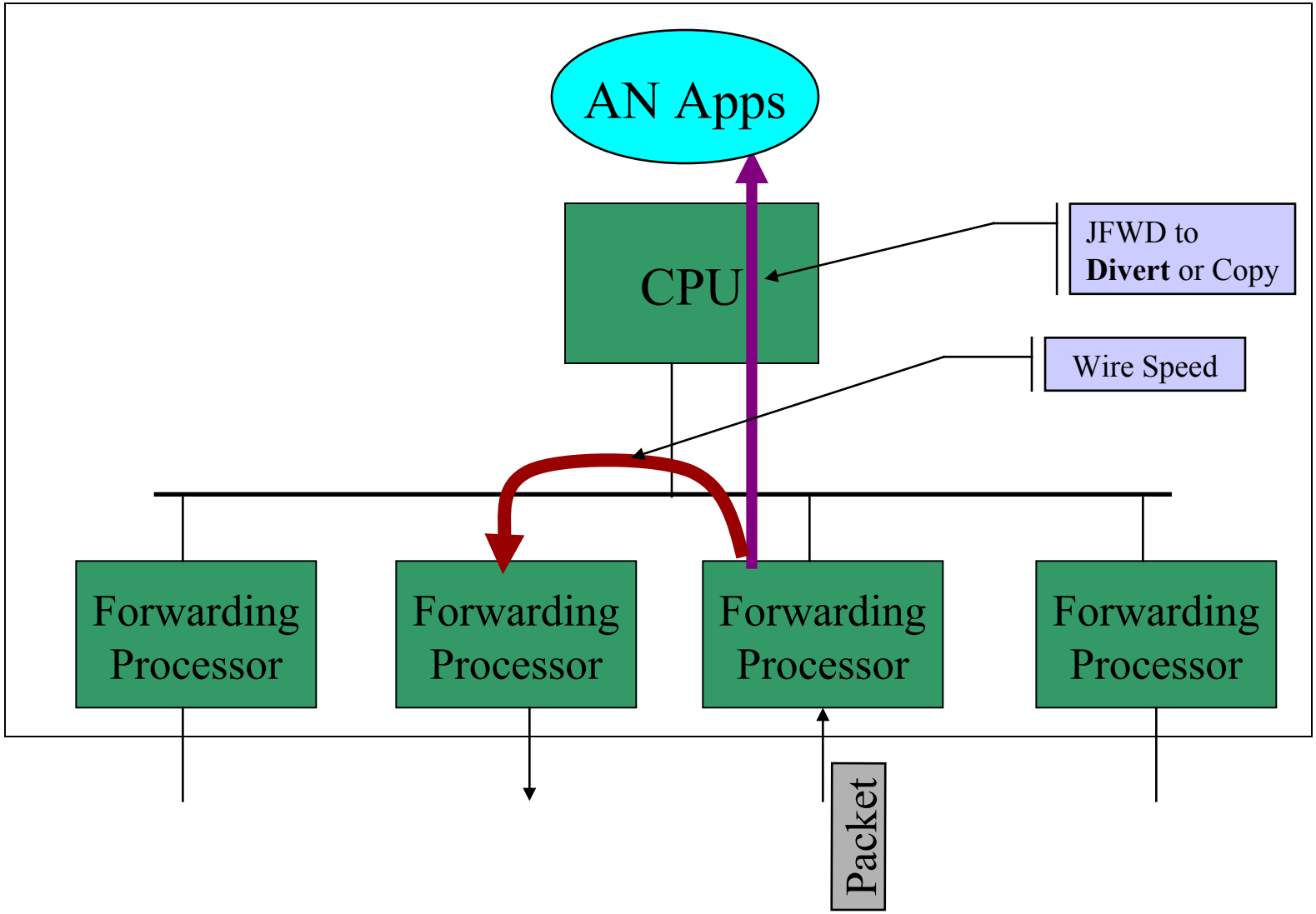
- **From downloadable Java application, we can modify the behavior of the ASICs**

Active Networks Packets Interception

Demo 3 -

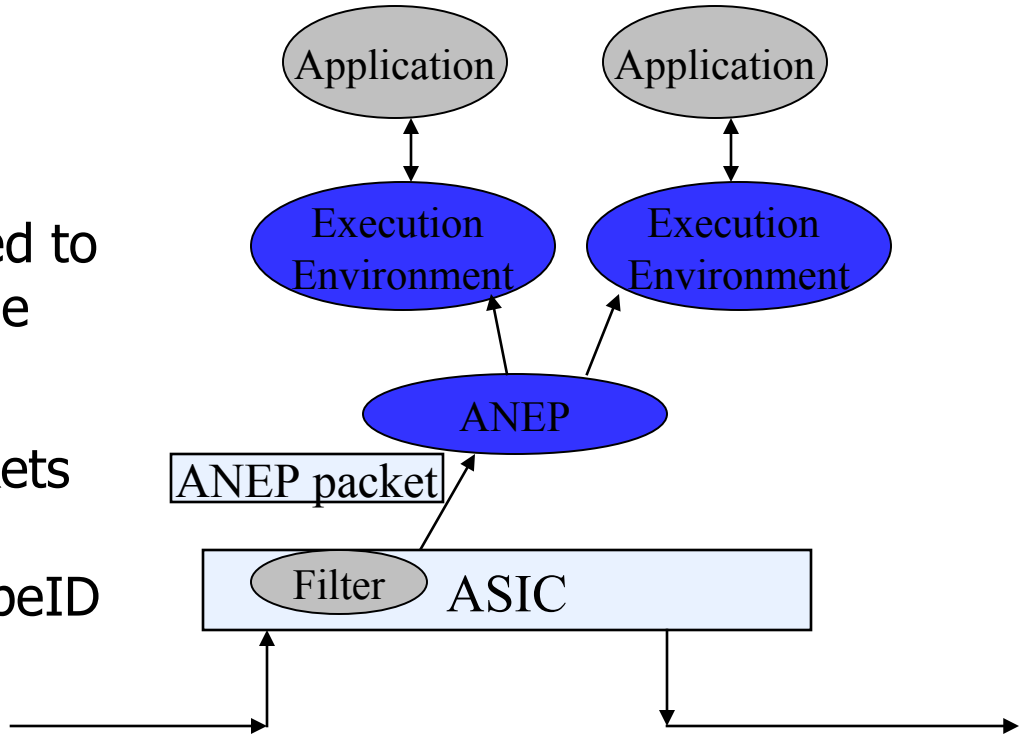
Active Networks Packet Capture

Demo 3



Packet Divert

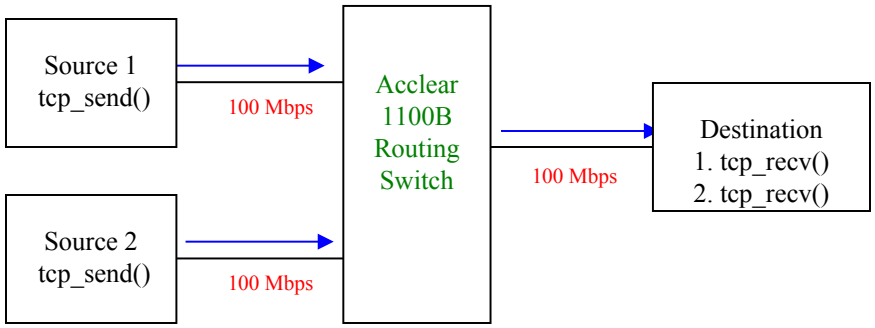
- Active Network topology is unknown
- ANEP packets NOT addressed to this node are delivered to the control plane for processing
- ANEP daemon receives packets and delivers them to the appropriate EE based on TypeID

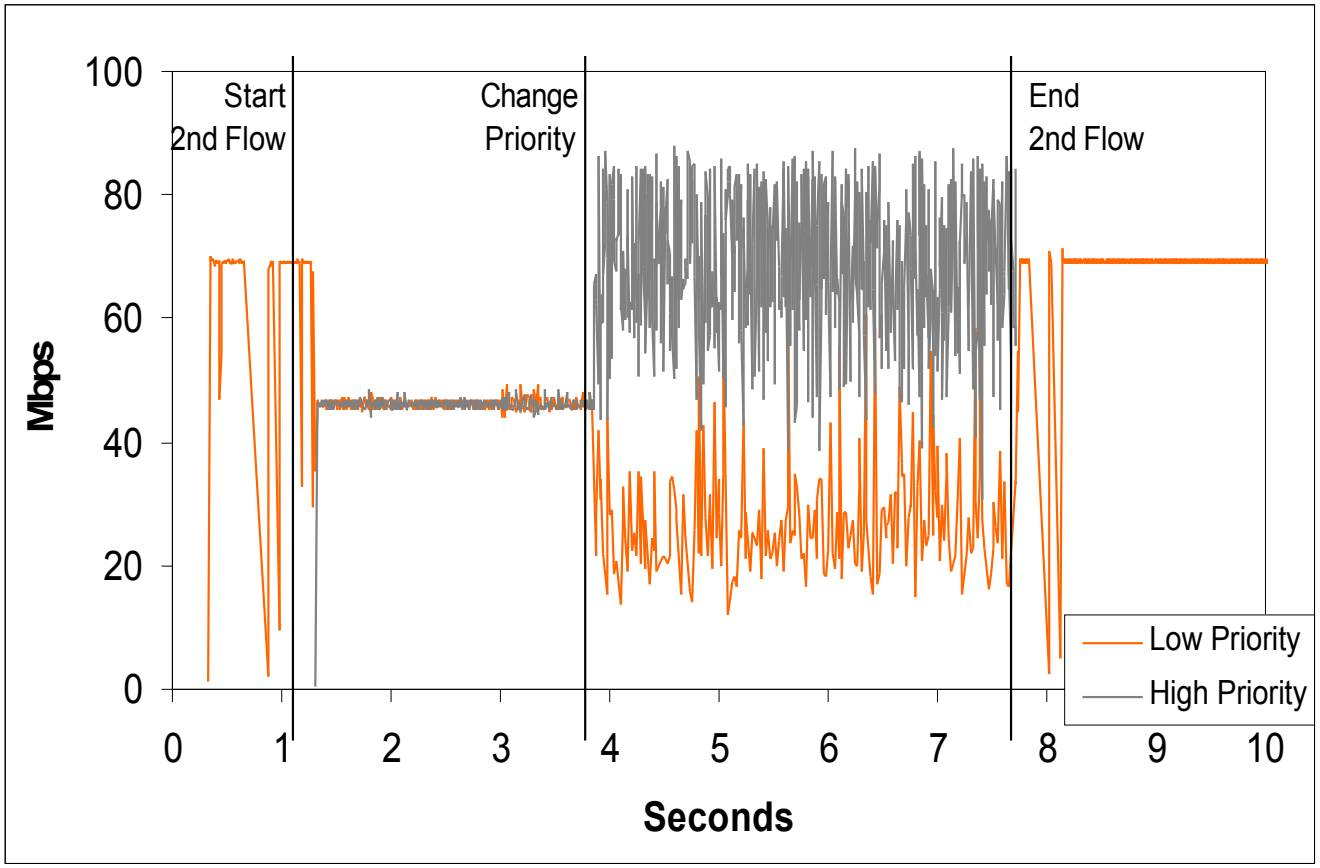


Active Networks Packet Capture

- **Be able to get the packets from the forwarding plane to the control plane**
- **Process Active Networks packets in the control plane**

Experimental Setup




100
80
60
40
20
0
Mbps
Start
2nd Flow
Change
Priority
End
2nd Flow
0
1
2
3
4
5
6
7
8
9
10
Seconds
Low Priority
High Priority

Summary

- Developed the ORE for downloading and safely running services onto network devices
- **Without** introducing software into data path we performed **Dynamic Classification of flows in a Silicon-Based Gigabit Routing Switch**
  - **Introduced a new service to a Gigabit Routing Switch**
  - **Identified real-time flows**
  - **Performed policy-based flow behavior classification**
  - **Adjusted DS-byte value**
  - **Showed that flow performance can be improved**

For more info email: rfj@cs.umd.edu