

Open Programmable Architecture for Java-enabled Network Devices

A Revolution!

**Tal Lavian
Technology Center
Nortel Networks
tlavian@NortelNetworks.com**

Programmable Network Devices

**Openly Programmable devices enable
new types of intelligence on the network**

Agenda

- **Local Computation**
- **New types of applications**
- **Architecture**
- **API's**
- **Summary**

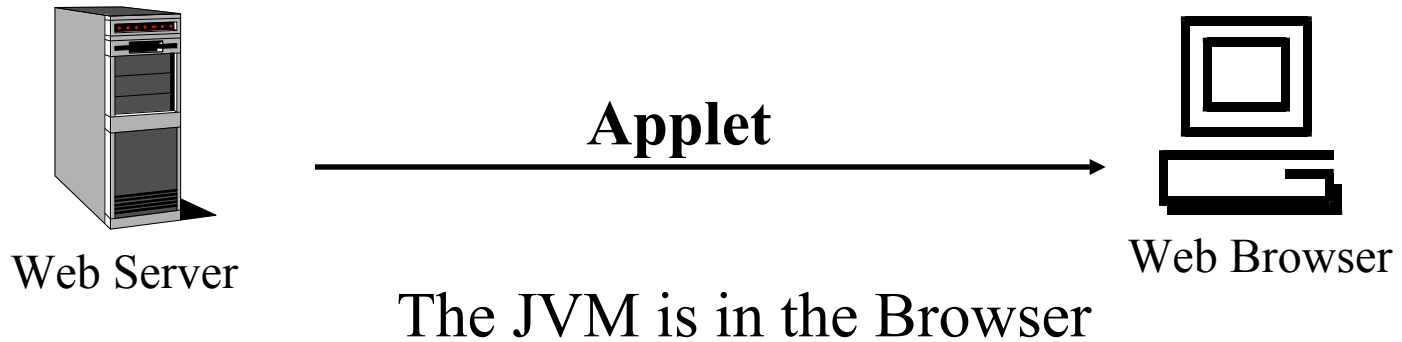
Changing the Rules of the Game

- **Move Turing Machine onto device**
 - Run non-vendor/non-bundled applications on network device

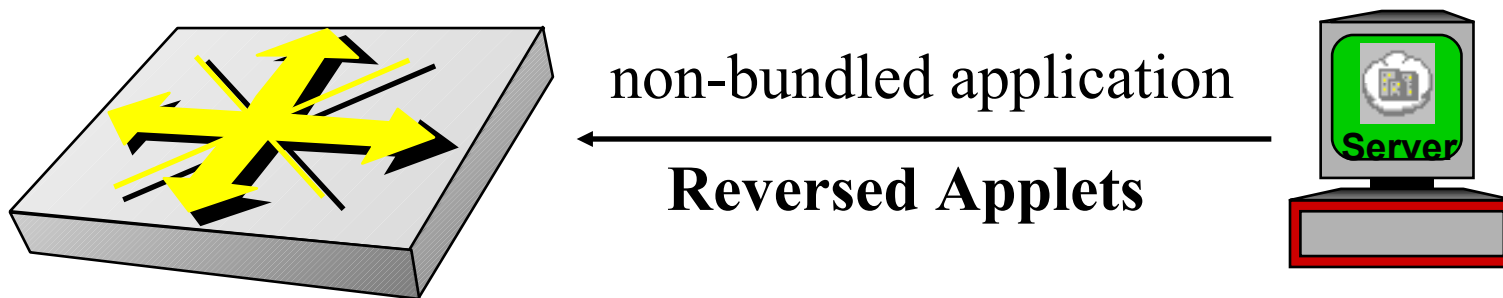
```
while (true) {  
  doLocalProcessingOnDevice()  
}
```



Non-vendor/Non-bundled Applications



Download applications for local processing



The JVM is in the Device: supports non-bundled apps

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

Architecture to Augment Vendor-Provided Software

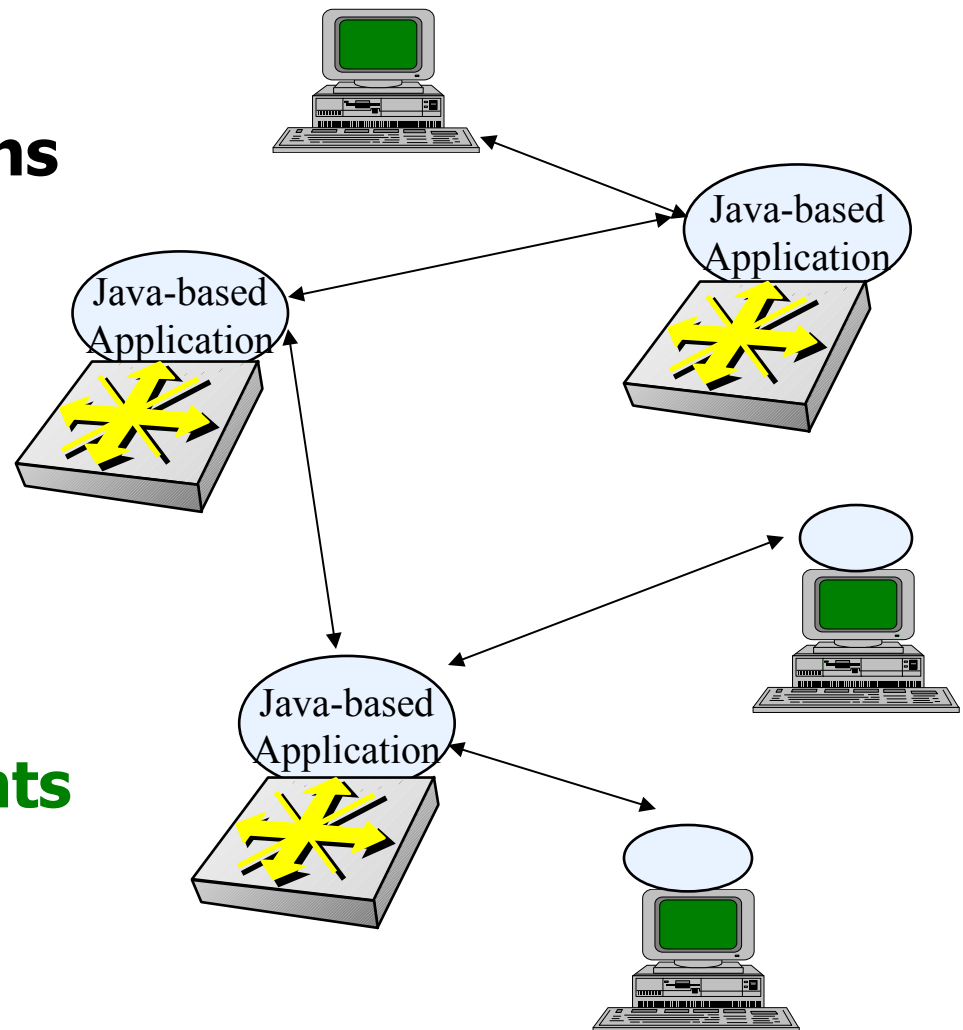
- **Supports non-vendor applications**
- **End-user custom application development**
 - Tight interaction between business applications and network devices
 - **Domain experts** who understand business goals
 - **Innovative approaches**
 - “Features on Demand”
 - download software services
 - dynamically add new capabilities

Paradigm Shift

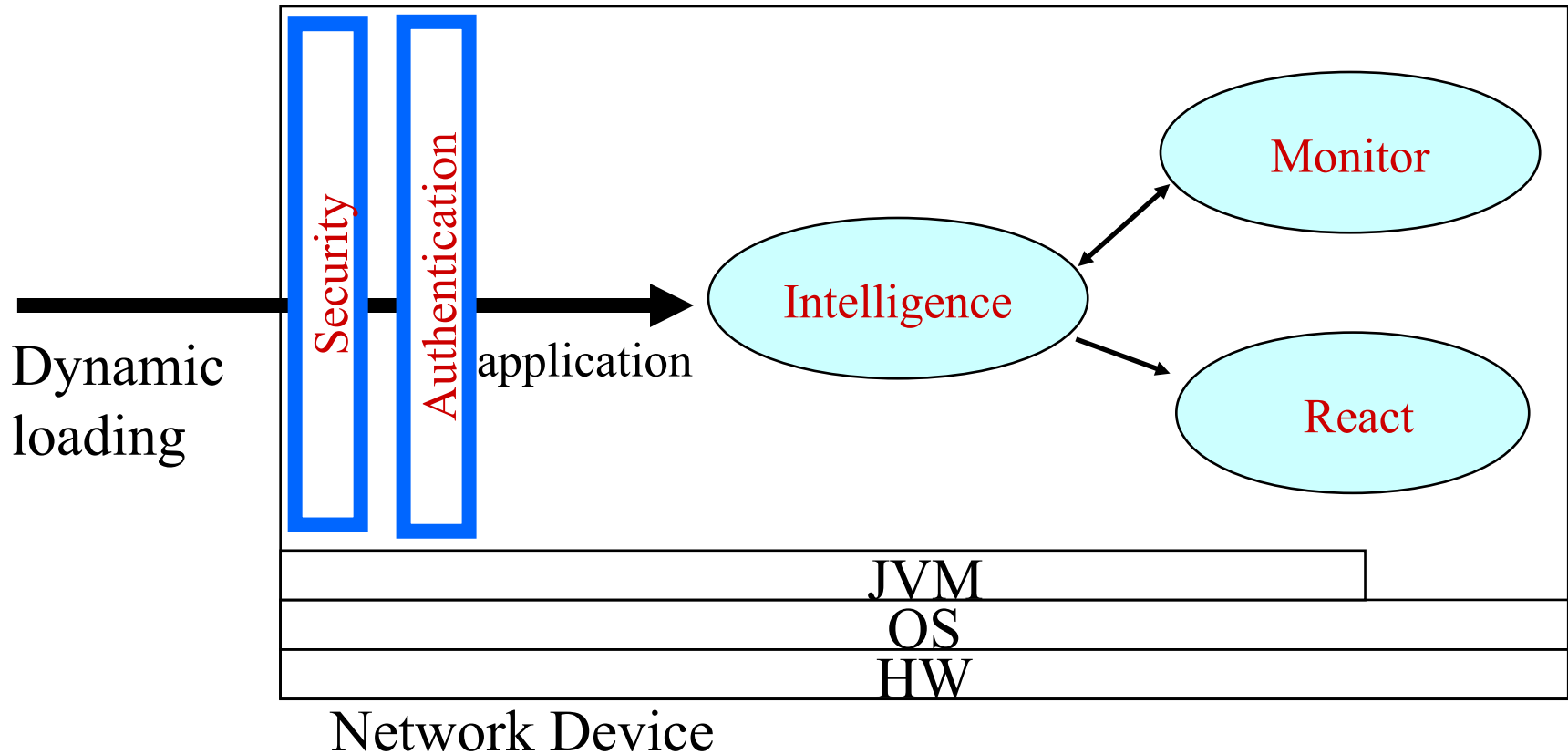
- Supports **distributed computing applications** in which network devices participate

- router to router
- server to router

- Supports **Intelligent Agents**
- Supports **Mobile Agents**



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

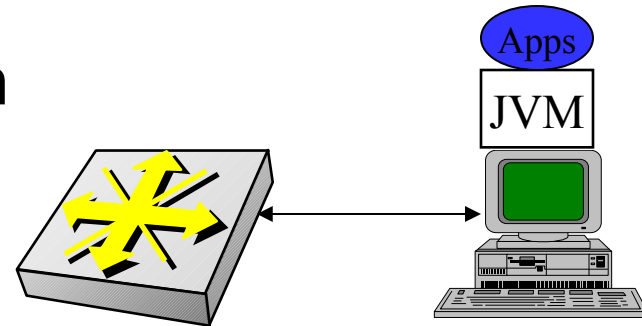
- autonomous and rational properties.

- dynamically system monitoring & modification

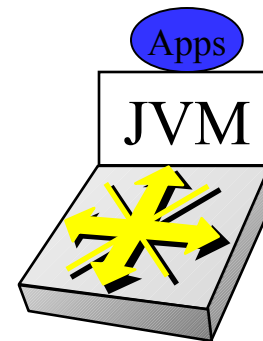
- report status and trends

Why not an external JVM?

- Isn't it easier to use an inexpensive PC with an external JVM?



- New type of apps utilizing the local computation and access to router resources.



Agenda

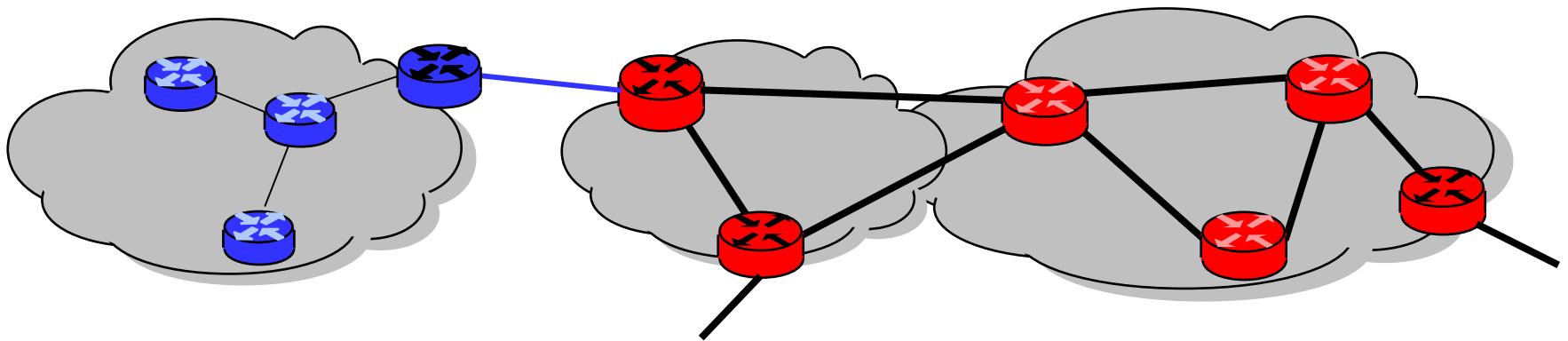
- **Local Computation**
- **New types of applications**
- **Architecture**
- **API's**
- **Summary**

New Types of Applications

- **Mobile Agents**
- **Local Intelligence for NMS**
- **Collaboration among routers**
- **Router & Server Collaboration**
- **E-commerce**

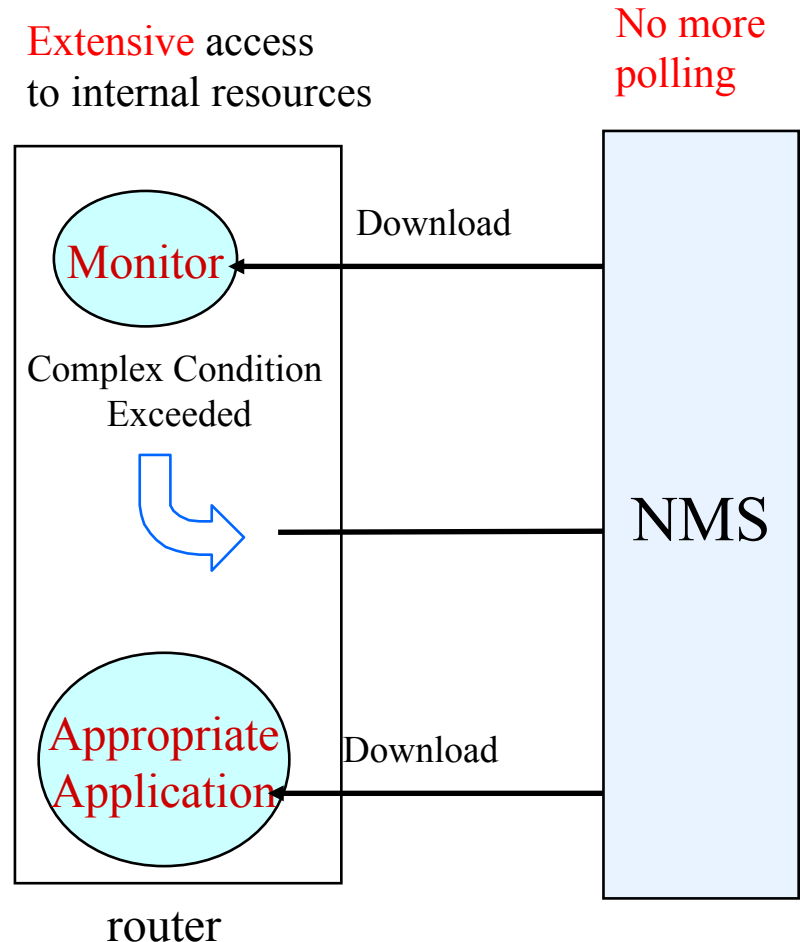
Mobile Agents

- **Intrusion Detection - Hacker Chaser**
- **Trace-route for Layer 2**
- **Mobile Connectivity Mapper**



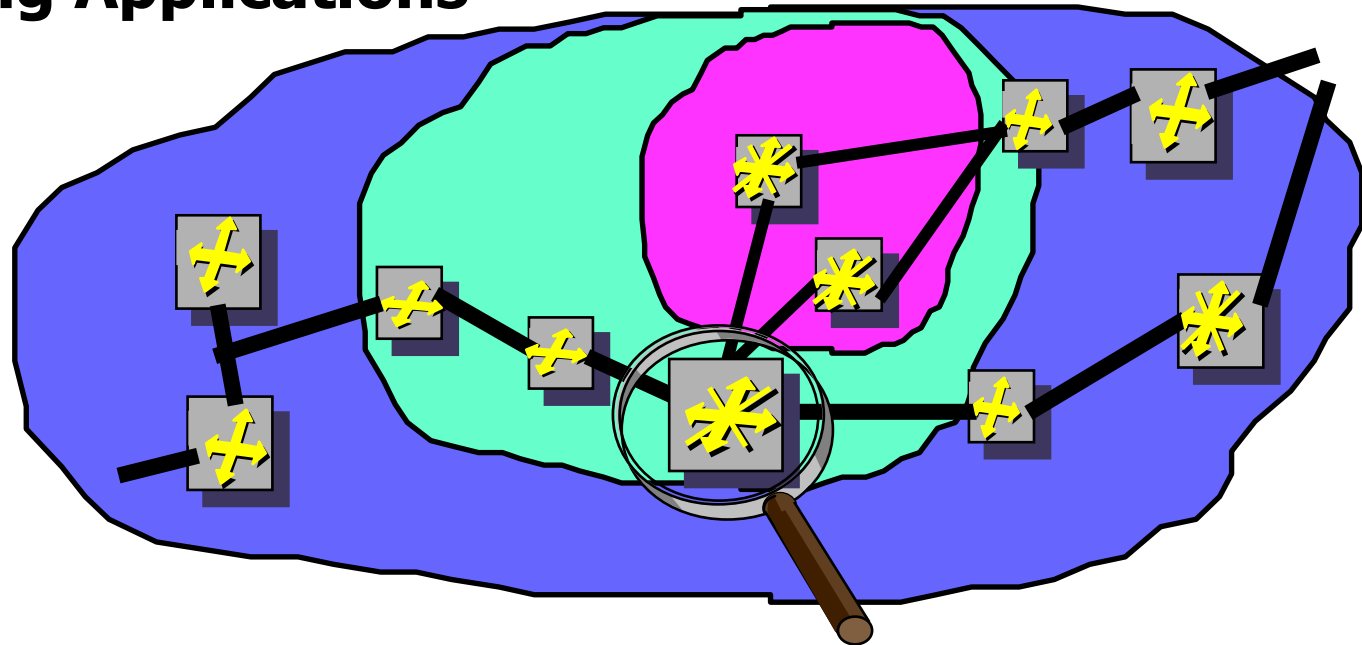
Local Intelligence for NMS: Diagnostic Agents

- **Download Intelligent Agent monitor from NMS to the device.**
- **Wait for threshold.**
 - Might be complex conditions
 - Trend analysis
- **Send “condition exceeded” event to NMS.**
- **Automatic download appropriate application**
- **Application takes action.**



Application Layer Collaboration Among Routers and Servers

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



E-Commerce Example

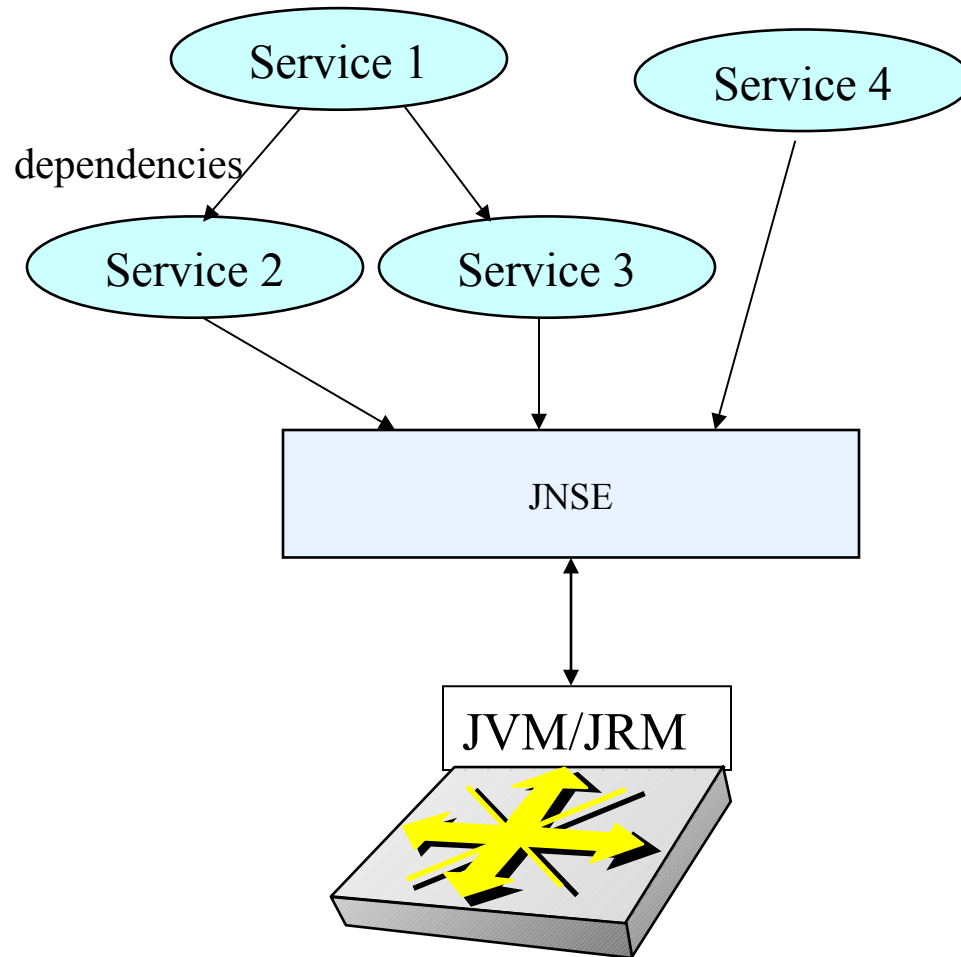
Business logic based operation changes

- **Resize forwarding queues**
- **Modify congestion control algorithm**
- **Adjust Packet Scheduling**
- **Change routing table**

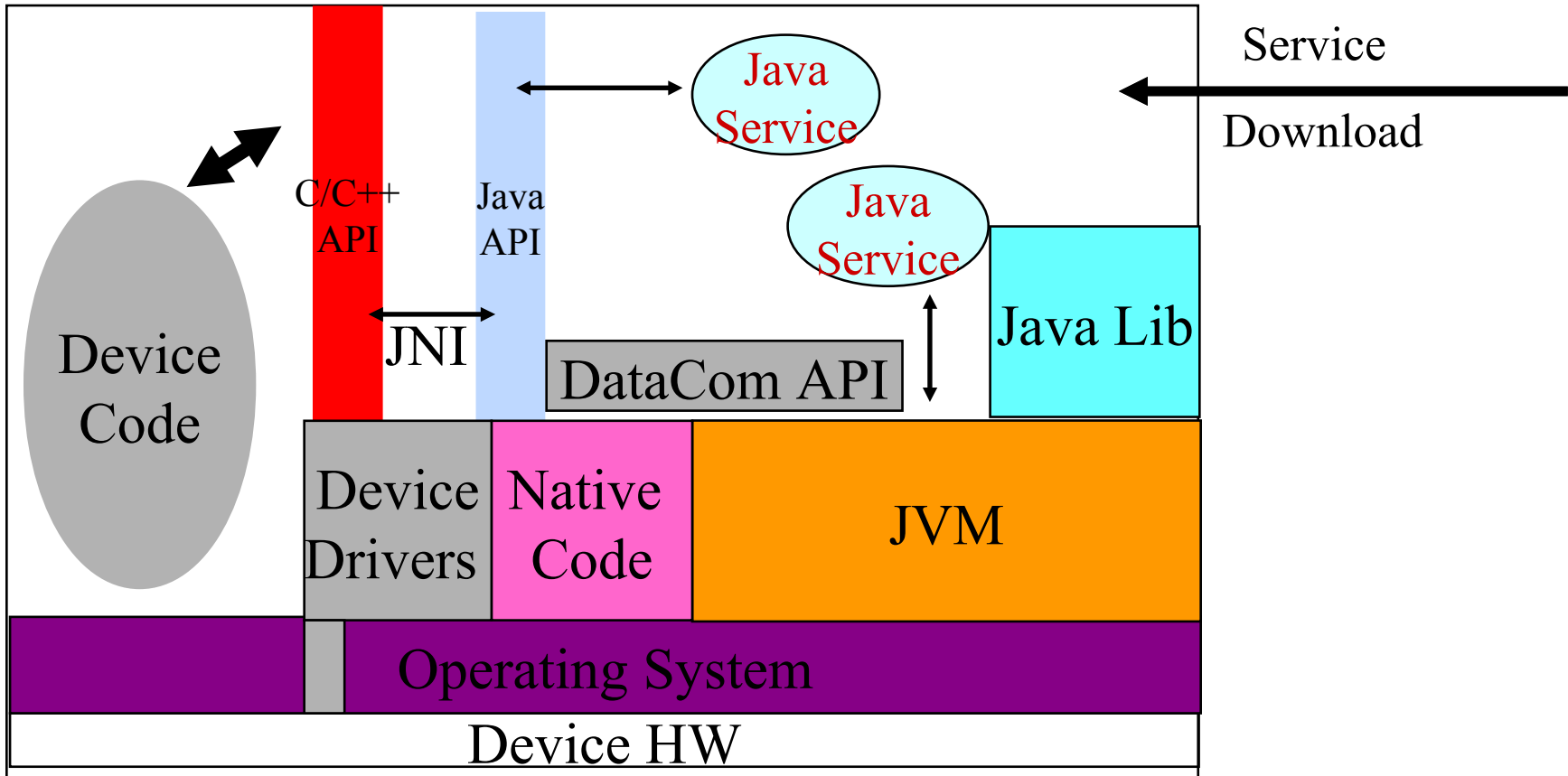
Agenda

- **Local Computation**
- **New type of applications**
- **Architecture**
- **API's**
- **Summary**

Java Network Services Environment



Open 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

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

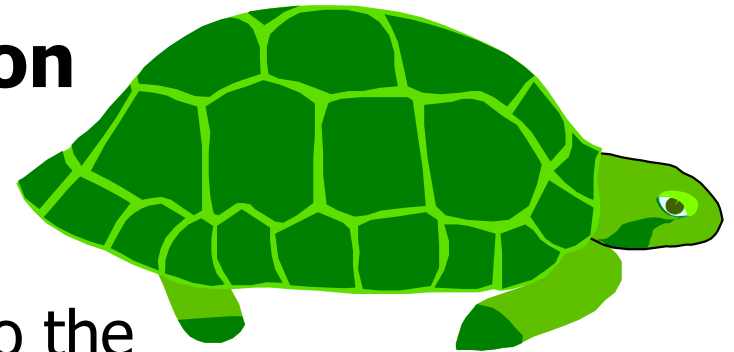
- **Multiple threads compete for resources**
 - memory
 - CPU
 - persistent storage
- **Denial-of-service attacks possible**
 - memory or CPU consumption attacks
 - trusted/untrusted service interactions

Why Java

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

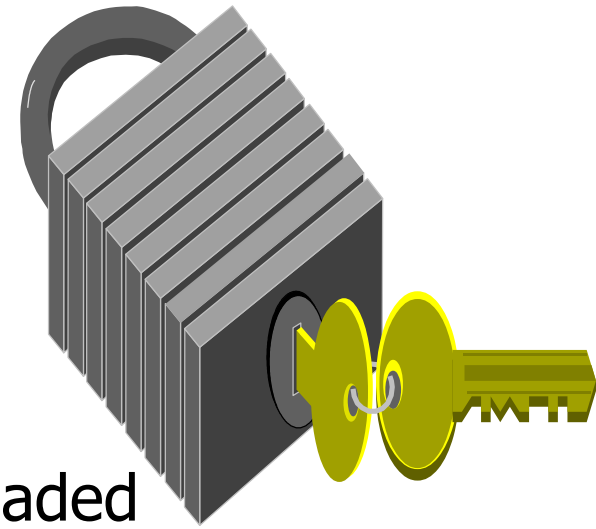
But Java is **sloooowwww**

- **Not appropriate in the fast-path data forwarding plane**
 - forwarding is done by ASICs
 - packet processing not affected
- **Java applications run on the CPU**
 - Packets destined for Java application are pushed into the control plane



Strong Security in the new model

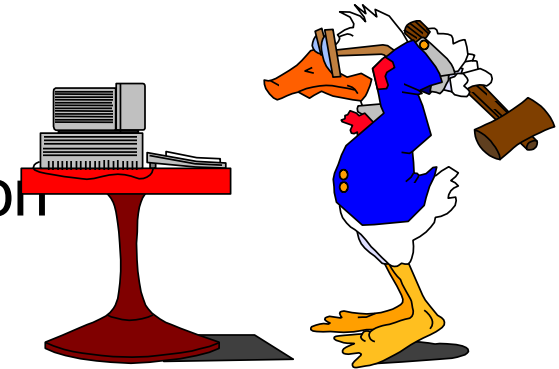
- **The new concept is secure to add 3rd party code to network devices**
 - Digital Signature
 - Administrative “Certified Optlet”
 - No access out of the JVM space
 - No pointers that can do harm
 - Access only to the published API
 - Verifier - only correct code can be loaded
 - Class loader access list
 - JVM has run time bounds, type, and execution checking



Old model Security (C/C++)

- **Old model: Not safe to add 3rd party code**

- Dangerous, C/C++ Pointers
 - Can touch sensitive memory location
- Risk: Memory allocations and Free
 - Allocation without freeing (leaks)
 - Free without allocation (core dump !!!!)



- **Limited security in SNMP**

Agenda

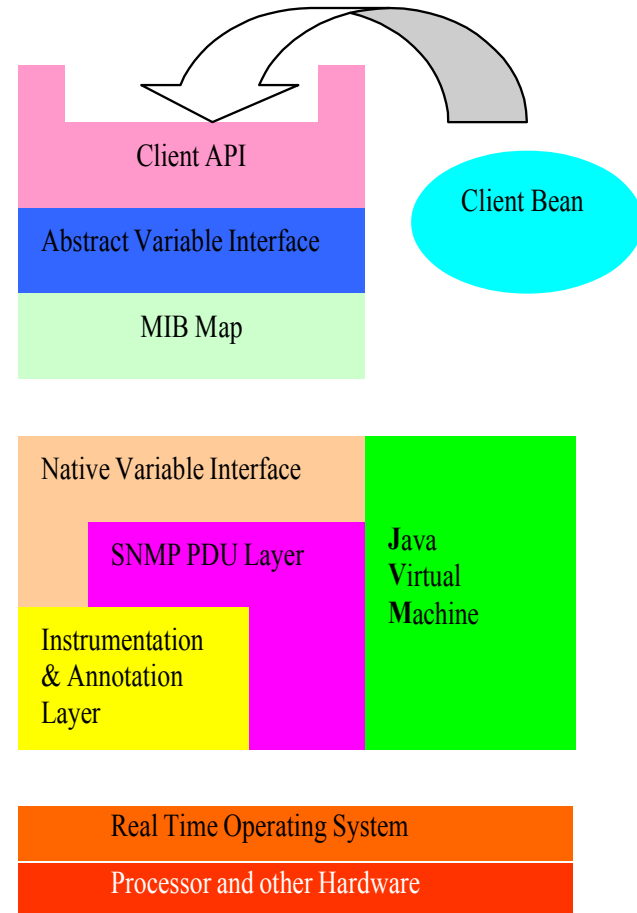
- **Openness**
- **Local Computation**
- **New types of applications**
- **Architecture**
- **API's**
- **Summary**

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

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



Agenda

- **Openness**
- **Local Computation**
- **New type of applications**
- **Architecture**
- **API's**
- **Summary**

Summary

- **Programmable**

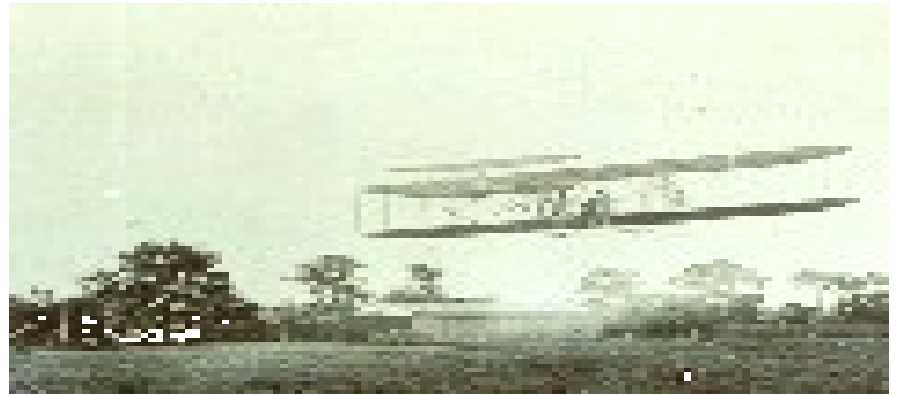
- Turing Machine on network devices
- *dynamic* agents vs. *static* agents
- dynamic loading
- strong security

- **Openness - successfully proven paradigm**

- Facilitates innovation
- Domain experts - virtual development community

- **Enabling Technology for the Revolution**

This is only the first step



1903 the Wright brothers

**Compare to this first flight and look
where aviation is today**