Unified Device Management via Java-enabled Network Devices

Tal Lavian  
Rob Duncan  
Bay Architecture Lab, Santa Clara CA

tlavian@IEEE.org  
rduncan@NortelNetworks.com
Agenda

• Unified Management for Unified Networks
• Openness - Virtual community development, Domain experts
• Open Service Interface - values
• Architecture and technology concepts
• Strong security
• Java SNMP MIB API
• Summary
Purpose

• To introduce the new Open Networking Architecture that is based on Java-enabled Network Devices

• To enable easy 3rd party integration
Unified Management for Unified Networks

**OBJECTIVE**
Unified management

**SOLUTION**
Java “Optlets” on all devices
Security and Directory

**BENEFITS**
Java-enabled Network Devices

Java on all devices
Unique value of Java

Unified Management
Community openness

- Success stories by large community of developers
- Net-based developers’ communities
  - Linux, GNU, Apache, BSD, X-Windows, Perl, Tk/Tcl
  - Netscape browser, NFS, JDK, JVM
- Linux everywhere:
  - Compaq, HP, IBM, SUN and SGI.
  - Intel, Sparc64, Alpha, PowerPC
- The Web changes everything
  - Java, XML, E-Business
Open Service Interface
- value propositions

• An open device software architecture enabler that:
  – reduces development cost by enabling cross-platform development
  – improves TTM through “feature-on-demand” capabilities
  – increases product differentiation by allowing incremental customization of products
Open Service Interface
- value propositions

• An open device software architecture enabler that:
  – enhances scalability and flexibility for distributed deployment of management and IP services
  – facilitates innovation by opening network devices to third party developers
  – provides incremental revenue through potential consulting/ customization services
Open Service Interface - levels of adoption

**Phase 1**
- Device level enabling technology
  - Selected device implementation
  - Feature-on-demand capability
  - Development efficiency

**Phase 2**
- Open systems architecture
  - Opening up of APIs for:
    - Customers
    - Consulting services
    - 3rd party ISVs

**Phase 3**
- Distributed applications framework
  - Wide adoption
  - Common distributed features
  - Distributed NMS applications “Optlets”
  - Mobile Agents

**Phase 4**
- New type of applications
  - Innovation
  - Imagination

Open Service Interface - levels of adoption
Java-enabled Network Devices

• What we have accomplished:
  – Java-enabled Device Architecture
  – JVM for Routers and Switches
  – JVM for network devices
  – Others – Optical network devices, OC-192
  – Java SNMP MIB API
    • include proxy mode for devices with no JVM,
    • Java interface to Cisco routers - COOOOOL !!
Technology concept “Reversed Applets”

The JVM is in the Device

Technology is based on the concept of Reversed-Applets

The JVM is in the Browser

Applet

Web Server

Web Browser

Optlets

Reversed Applets

Java Beans

NMS
Potential applications

• “Feature-on-demand” for devices

• New class of system level Optivity applications in the form of distributed “Optlets”
  – characterized by system applications that require intensive interaction between NMS and device and/or across multiple devices
  – potential applications are topology, design analysis, diagnostics, policy implementations
Benefits and value

• Enabling component of a new intelligent network architecture
  – distributed applications-on-demand
  – component of AI (artificial intelligence) enabling infrastructure
  – roaming diagnostics and self-healing capabilities
  – built-in support for open industry ISV support
Example - Local Intelligence
**Application example**

- Download Intelligent Agent Monitor from NMS to the device
- Wait for threshold
  - might be complex conditions
- Send “condition exceeded” event to NMS
- Automatic download appropriate application
- Application takes action
“Oplets” = Distributed Optivity Applications
Open Device architecture

Device Code

C/C++ API
Java API
JNI
DataCom API
Native Code
JVM
Operating System
Device HW

Java Bean
Java Lib

“Optlet”
Download
JSCP system diagram

Java Applications

JVM

Java Libraries

Memory Manager

JNI

Services
Reset
Clean-up
Configuration
Event Mapping
Monitor Thread

Native Applications

Scheduler

Context Switch

RTOS & Hardware

RTOS & Hardware

JSCP system diagram
Strong security in the new model

• The new concept is secure to add 3rd party code to network devices
  – digital signature
  – “Certified Optlet”
  – no access out of the JVM space
  – no pointers to harm the work
  – access only to the published API
  – verifier - only correct code can be loaded
  – class loader access list
    • different Optlets with different access levels
  – JVM has run time bounds, type, and executing checking
Old model security (C/C++)

- Old model - not secure to add 3rd party code
  - not recommended to add 3rd party code to network devices
  - dangerous, C/C++ pointers
    - can touch sensitive memory location
  - risk: memory allocations and free
    - allocation without freeing
    - free without allocation (core dump !!!! )

- Limited security in SNMP
Java SNMP MIB API

- Portable across a range of network devices
- Extensible
- Simple and convenient for client use
- Consistent with SNMP model
- Hide unnecessary SNMP details
- Permit optimized access
- Re-use MIB documentation
MIB API generation

• Most of the Java code is generated automatically

• ASN.1 MIB definitions are converted into Java classes

• Documentation and commentary in the MIB definitions is placed as Javadoc formal comments

• HTML documentation generated from Javadoc
MIB objects

- The MIB data model is structured as a tree
- API represents MIB groups with Java classes
- MIB variables are represented with accessor methods
- Conceptual tables are represented with iterators
- API converts SNMP data values into standard Java types
JSNMP MIB API architecture

• 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
Advantages of MIB map

- Allows immediate generic implementation of the entire MIB via the loopback scheme
- Enables optimized native implementation of key MIB variables for maximum efficiency
- Permits definition of pseudo-MIB variables for extending MIB dynamically
- Provides site for centralized access management
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

Diagram:

Java Server → Java "Optlet" → JVM on PC → JSNMP API Proxy mode → SNMP → Cisco Router with No JVM
Summary

• Openness - successfully proven paradigm
• Domain experts - virtual community
• Allows innovations and added value
• Dynamic agents vs. static agents
• Dynamic loading
• Strong security
• An enabling-technology

Take it, and make it work for you