# A Representation of Network Node QoS Control Policies Using Rule-based Building Blocks

Yasusi Kanada Hitachi Ltd., Central Reserach Laboratory

## Introduction

#### Policy-based QoS-control

- ◆ Policy rules:
   Administrator or Operator → Policy Server → Network Nodes
- Interfaces between Policy Servers and Network Nodes
  - Protocols: SNMP, COPS, ...
  - APIs: CORBA, ...



# A problem in the policy-control interfaces

#### ■ The grammar is limited.

- Grammar = Syntax + Semantics
- ♦ The syntax is limited.
  - E.g., Only one value can be sent at a time by SNMP.
  - E.g., Only function calls can be described by an API.
- The semantics is limited.
  - Control structures cannot be expressed.
  - Constraints cannot be expressed.

IWQoS 2000	2000-6-6	By Yasusi Kanada	(C) Hitachi Ltd.	3

# A Solution

# To use a rule-based programming language for the interface.



- ◆ A language is defined by a grammar.
  - Our problem is limitation of grammar.
- Policy-based control is a matter of programming.
  - A set of policies is a program, because policies describe the behavior of network nodes.
- ◆ Policies are *rule-based*.
  - A policy rule is (should be) an if-then rule.

## Each building block has

- One or more input ports.
- One or more output ports.
- An input and output ports are connected by a named pipe.

## ■ An example: a Diffserv router setting



# **SNAP: A Parallel Logic Language**

## An example

◆ An informal description of a rule:

```
if (Source_IP == 192.168.0.1 && Source_port == 80)
DSCP = 46;
```

◆ The rule in SNAP:

```
filter_mark(Si, So) :-
```

```
filter[Source_IP = 192.168.0.1, Source_port = 80](Si, S1) |
mark[DSCP = 46](S1, So).
```

## Building-block syntax

- ◆ Basic syntax: *bb\_name*(*Si*, *So*).
- General case:

class\_name[parameters](Si1, Si2, ..., Sin, So1, So2, ..., Som).

# SNAP: A Parallel Logic Language (cont'd)



# Conclusion

### A rule-based language SNAP has been introduced.

 SNAP is for the interface between a policy server and network nodes.

#### SNAP enables

- Definition of building blocks by users
  - Using primitive building blocks and case structures.
- Interoperable policy-based QoS control.
- Expressing wide variety of QoS functions.

#### Future work

- Detailed specification of SNAP.
- Implementation of SNAP on a policy server and routers.

## SNAP: A Parallel Logic Language (cont'd)



# **Building Blocks for Diffserv**

#### Six types of primitive building blocks

 Filtering, Metering, Marking, Discarding, Scheduling, and Merging (MUX) rules.

#### A control structure and constraints

Order of building blocks:

