Internet Management: Status and Challenges

2004 IEEE/IFIP Network Operations & Management Symposium

Seoul, Korea, 2004-04-19

Aiko Pras

a.pras@utwente.nl
University of Twente
7500 AE Enschede
The Netherlands

Jürgen Schönwälder

j.schoenwaelder@iu-bremen.de
International University Bremen
28759 Bremen
Germany

Copyright Notice

Outline of the Tutorial

1. IETF Working Group Overview (Jürgen Schönwälder, 20 min)
2. SNMP Version 3 (SNMPv3) (Aiko Pras, 45 min)
3. SNMP Improvements (Jürgen Schönwälder, 35 min)
4. XML-based Management (Jürgen Schönwälder, 35 min)
5. Web Services for Management (Aiko Pras, 45 min)

1 IETF Working Group Overview

1.1 MIB Review Guidelines

1.2 IPv6 Support and Management

1.3 Entity MIB Evolution

1.4 Distributed Management

1.5 Middlebox Management

1.6 IEEE 802 Management

1.7 ATM / MPLS / xDSL / Cable Modems

1.8 Printer, IP Storage, RMON

1.9 Extensible Provisioning Protocol

1.1 MIB Review Guidelines

- All MIB modules published by the IETF go through a review process.
- There was quite some variation what MIB doctors checked and agreed upon.
- Many discussions were needed to identify the common rules used by the MIB doctors.
- The MIB Review Guidelines draft documents some of the SMI folklore and the CLRs (crappy little rules or consistency language rules) that are checked during MIB review.
- MIB module authors are encouraged to check their MIBs against these rules before publishing them or submitting them to the IESG.
- A subset of the rules that can be automatically checked has been added to the `smilint` MIB module checker of the `libsmi` package.
- Guidelines document is of high quality and relatively stable. Should go to the IESG sometime during this year.

1.2 IPv6 Support and Management

- MIB modules under revision:
  - TCs for Internet Network Addresses (RFC 3291 update)
  - IP-MIB (RFC 2011 update)
  - IP-FORWARD-MIB (RFC 2096 update)
  - TCP-MIB (RFC 2012 update)
  - UDP-MIB (RFC 2013 update)
  - TUNNEL-MIB (RFC 2667 update)
- Original IPv6 MIB modules published in 1998 will be made historic.
- Documents being finalized, should go to the IESG soon.

1.3 Entity MIB Evolution

- The ENTITY-MIB models physical entities (e.g., fans, sensors, cpus, ports, modules, chassis) that make up a device.
- Represents the containment hierarchy of physical entities
- Very essential MIB module (comparable to the IF-MIB)
- Improvements made during the last months:
  - 3rd revision of the ENTITY-MIB (to become Draft Standard)
  - ENTITY-SENSOR-MIB extension for sensors (RFC 3433)
  - MIB module providing state objects for physical entities
- 3rd version of the ENTITY MIB waiting for publication, state extension being finalized.

1.4 Distributed Management

- Recent work was centered around the definition of a general alarm reporting mechanism, based on some of the ITU work in this space (X.733).
- An additional module provides an alarm reporting control interface, again based on some ITU work in this space (M.3100 Amendment 3).
- Other work items are concerned with the progression of DISMAN MIB modules to Draft Standard.
- Some discussion about the complexity of the event and expression MIB modules.
- Documents:
  - Alarm MIB
  - Alarm Reporting Control MIB
  - Ping, Traceroute, Lookup MIB (RFC 2925) Revision
- Documents basically ready for submission to the IESG.

1.5 Middlebox Management

- A middlebox is a network intermediate device (NAT, firewall) that needs to be configured in order to make applications work (“drilling holes into middleboxes”).
- Working group went through a detailed development process:
  - Middlebox Communication Architecture and Framework (RFC 3303)
  - Middlebox Communications (MIDCOM) Protocol Requirements (RFC 3304)
  - Middlebox Communications (MIDCOM) Protocol Evaluation
  - MIDCOM Protocol Semantics
  - Middlebox Communications (MIDCOM) Protocol Managed Objects Analysis
  - Definitions of Managed Objects for Middlebox Communication
- Most WG members wanted a simple special purpose protocol and they get an SNMP MIB which nobody really wanted. Is this a model for the future?

1.6 IEEE 802 Management

- Most of the IEEE 802 documents are doing in cooperation with the IEEE.
- Newer Documents:
  - Power Ethernet MIB (RFC 3621)
  - Ethernet-like Interface Types MIB (RFC 3635)
  - IEEE 802.3 Medium Attachment Units MIB (RFC 3636)
  - Ethernet WAN Interface Sublayer MIB (RFC 3637)
  - Port Access Control MIB
  - Ethernet in the First Mile (EFM) Common MIB
  - Ethernet in the First Mile Copper (EFMCu) Interfaces MIB
  - Ethernet Passive Optical Networks MIB
  - VLAN Textual Convention MIB
- Discussions about moving MIB development control over to the IEEE with MIB review service provided by the IETF.

1.7 ATM / MPLS / xDSL / Cable Modems

- Several working groups produce a stream of interface type specific MIB modules.
- Newer Documents:
  - Optical Interface Type (RFC 3591)
  - SONET/SDH MIB Revision (RFC 3592)
  - Supplemental Managed Objects for ATM Interface (RFC 3606)
  - DS1 / E1 / DS2 / E2 / DS3 / E3 Interface Type MIBs Revision
  - Many (perhaps too many?) MIB modules related to MPLS and traffic engineering
  - Very High Speed Digital Subscriber Lines (VDSL) MIB (RFC 3728)
  - VDSL Extensions for Single Carrier Modulation (SCM) Line Coding
  - VDSL Extensions for Multiple Carrier Modulation (MCM) Line Coding
  - Many MIB modules related to Cable Modems.
- The MPLS related MIB modules probably need to get consolidated.

1.8 Printer, Fiber Channel, iSCSI, Remote Monitoring

- Several working groups produce MIB modules related to printers, storage technologies and continue the RMON monitoring suite.

- Newer Documents:

  – Printer MIB and Finisher MIB (waiting for publication)

  – Fiber Channel MIB modules are being revised / extended

  – iSCSI MIB modules are currently being defined

  – RMON Overview Document (RFC 3577)

  – Application Performance Measurement MIB (RFC 3729)

  – Transport Performance Metrics MIB

  – Synthetic Sources for Performance Monitoring Algorithms MIB

  – Real-time Application Quality of Service Monitoring MIBs

1.9 Extensible Provisioning Protocol

- Application layer client-server protocol for the provisioning and management of objects stored in a shared central repository
- Target application area is automated interaction with registries
- Extensible Provisioning Protocol Features
  - XML based protocol (commands / responses)
  - Session management commands (login, logout)
  - Query commands (check, info, poll, transfer)
  - Object transform commands (create, delete, renew, transfer, update)
  - Mapping over TCP and BEEP defined (SCTP not detailed)
- Important piece of work done in the Applications Area and largely ignored by the Operations and Management Area...

3 SNMP Improvements

3.1 Next Generation Structure of Management Information (SMIng)

3.2 SNMP over TCP

3.3 SNMP Payload Compression

3.4 Extended SNMP Protocol Operations

3.5 AES Cipher Algorithm for USM

3.6 SNMP Uniform Resource Locators

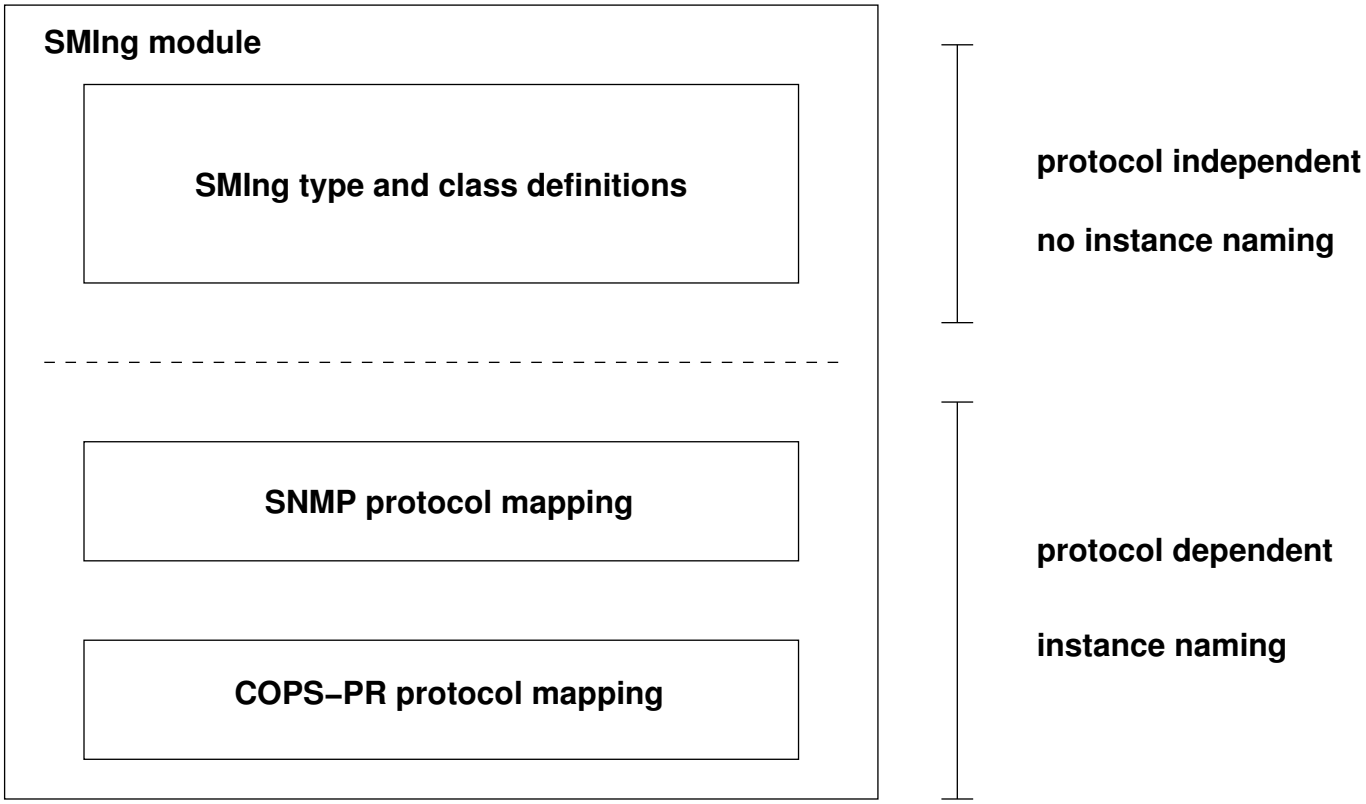
3.7 Session-Based SNMP Security Model

3.1 Next Generation Structure of Management Information

- Problems:
  - SMIv2 misses some important base types such as 64 bit numbers.
  - SMIv2 lacks reusable composite data types.
  - SMIv2 syntax depends on ASN.1 and is generally not well understood and implemented correctly.
  - SMIv2 parsers are difficult to write due to a lack of a well defined grammar.
  - SMIv2 is not extensible.
  - Desirable to use the same data definitions with SNMP and COPS-PR.

- Solution:
  - A next generation data modeling language called SMI (SMIng).
  - Detailed objectives have been documented in RFC 3216.
  - Soon to be published as Experimental RFCs.

SMIng Module Structure



- Reusable type and class definitions are seperated from protocol specific mappings.
- Abstraction of instance naming is the most difficult problem to solve.

SMIng Syntax

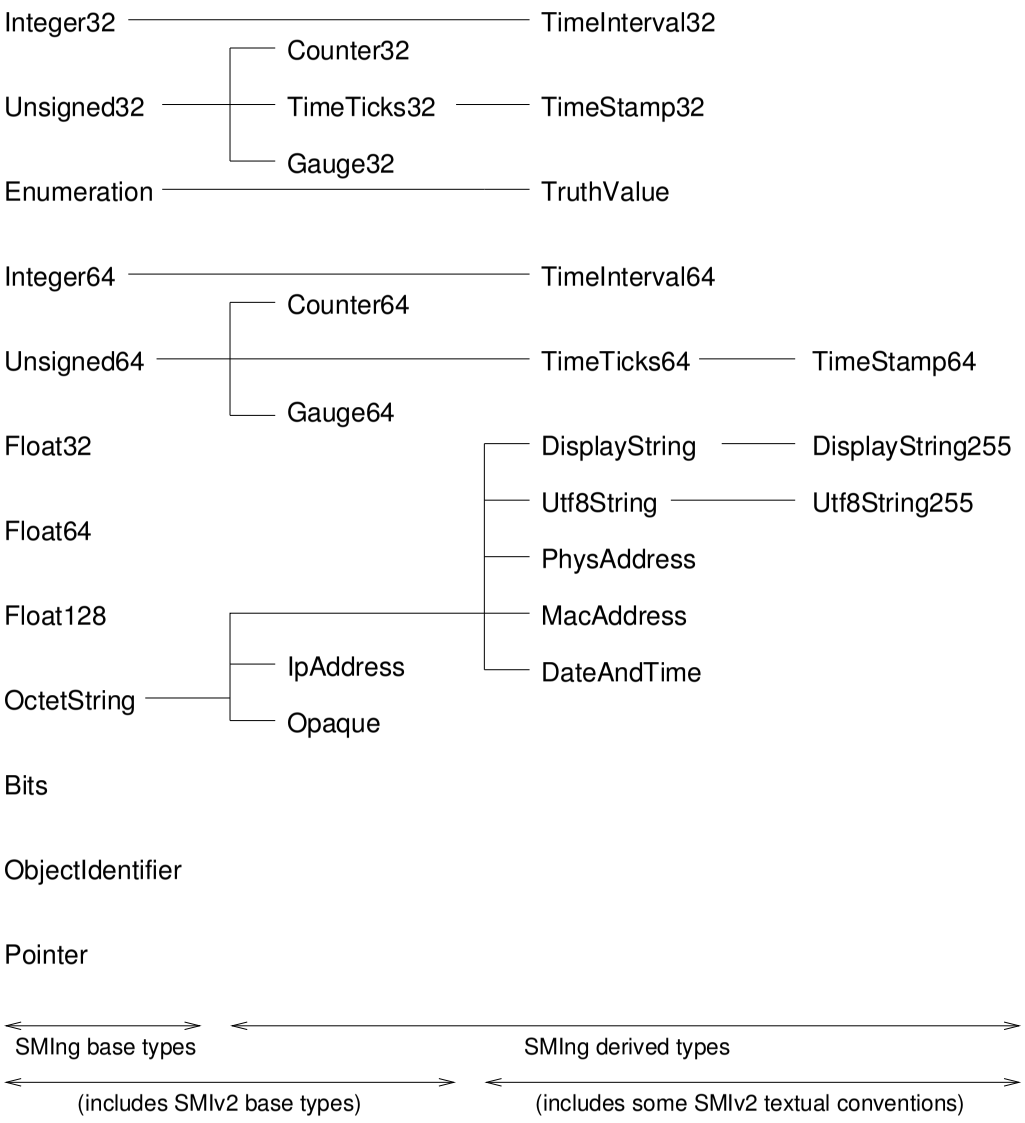
- Programmer friendly syntax:
  - look and feel similar to Java, C, C++, ...
  - consistent structure of statements (easier to memorize)

- Easy to implement and efficient to parse:
  - consistent syntactic structure simplifies grammar
  - no forward references (except in cases where they are unavoidable)
  - statement seperators help to recover from parse errors
  - complete grammar specified in ABNF (RFC 2234)

- Language extensibility:
  - declaration of new statements, parsers skip unknown statements

$\Longrightarrow$ Not everyone in the IETF agrees that the ASN.1 syntax is ugly...

SMIng Base Types and Core Derived Types



SMIng Attributes and Classes

- Classes encapsulate a set of attributes.
- Attributes have an associated type which can be
  - a base type, or
  - a derived type, or
  - a class (composite type).
- Classes can have associated events.
- Every event in SMIng is associated with a class.
- Events can be mapped to notification messages in protocol mappings.
- Methods are not supported, but might be added in the future.

SNMP Protocol Mapping

- Defines how SMIng base data types are mapped to SNMP data types.
- Uses Opaque wrapping to support new base types.
- Complex composite types are flattened and mapped to table rows or groups of scalars.
- OID names are assigned in mapping statements.
- SNMP specific derived types (e.g. RowStatus) are defined in a mapping module.

SMIng Example

```
class BasicInOutErrStats {
    attribute inOctets  { type Counter32;  ... };
    attribute inErrors  { type Counter32;  ... };
    attribute outOctets { type Counter32;  ... };
    attribute outErrors { type Counter32;  ... };
    ...
};

class Interface {
    attribute index { type InterfaceIndex; ... };
    attribute stats { type BasicInOutErrStats; ... };
    ...
};

snmp {
    table ifTable {
        oid    interfaces.2;
        index  (ifIndex);
        object ifIndex     { implements Interface.index; ... };
        object ifInOctets  { implements Interface.stats.inOctets; ... };
        object ifInErrors  { implements Interface.stats.inErrors; ... };
        object ifOutOctets { implements Interface.stats.outOctets; ... };
        object ifOutErrors { implements Interface.stats.outErrors; ... };
        ...
    };
    ...
};
```

3.2 SNMP over TCP (RFC 3430)

- Motivation:
  - Support larger message sizes to improve bulk transfers.
  - Support session-based security mechanisms.
  - No vehicle to turn unconfirmed operations into confirmed operations.

- Specification:
  - Optional transport mapping (UDP still has to be available).
  - Originator of a request-response transaction chooses the transport protocol for the entire transaction.
  - Framing relies on ASN.1/BER message length information.
  - Implementations must provide buffers to reassemble fragmented messages.
  - Piggybacking of TCP ACKs important!

3.3 SNMP Payload Compression

- Motivation:
  - Lossless compression of SNMP payloads with minimal processing overhead.
  - Improve encoding efficiency to ship more data in SNMP messages.

- Requirements:
  - Compression must happen before encryption.
  - Each SNMP message is compressed and decompressed by itself without any relation to other SNMP messages (”stateless compression”).
  - The size of a compressed SNMP message must never exceed the size of the uncompressed SNMP message (”non-expansion policy”).
  - The abstract syntax of compressed SNMP messages must be defined using ASN.1 to ensure that compressed SNMP messages have a valid ASN.1/BER encoding.
  - SNMP payload compression should be able to support multiple compression algorithms. It is desirable to define common compression algorithms in order to achieve interoperability.

OID Delta Compression (ODC)

- Reduce the OID overhead inherent in SNMP messages by encoding OIDs of variable names as deltas to the previous OID
- The deltas are expressed by a combination of the following primitives:
  1. Substitution of a single sub-identifier at a certain position
  2. Substitution of ranges of sub-identifiers at a given start position
  3. Truncation and enlargement of the OID

- Algorithm:
  1. Loop through the SNMP PDU until you find an OID name value pair (varbind).
  2. If it is the first varbind, make a copy of the OID, pass it to the output buffer and continue with the next varbind.
  3. Otherwise, compute the delta to the last OID and BER encode it into the CompOID value.
  4. If the CompOID representation is larger than the OID, pass the OID to the output buffer, else pass the CompOID to the output buffer.
  5. Update the last OID and goto step two if there are additional varbinds.

3.4 Extended SNMP Protocol Operations

- Additional protocol operations can substantially improve SNMP's capabilities:
  - `GetConfig` and `SetConfig` to read and write configuration settings.
  - `CallRequest` and `CallResponse` to invoke operations.
  - `GetTable` to retrieve complete tables with filtering and OID suppression.
  - `Create` and `Delete` to address the complexity of the `RowStatus` mechanism.
  - Object-oriented PDUs with transaction support.

$\Longrightarrow$ There is no agreement which primitives are needed in which order and complexity.

3.5 AES Cipher Algorithm for USM

- Problem:
  - The SNMP USM security model uses the DES cipher algorithm which is not considered very secure these days.
  - The Advanced Encryption Standard (AES) is widely accepted as a stronger replacement for DES

- AES Cipher Algorithm for the USM:
  - AES in Cipher Feedback Mode (CFB) with a key size of 128 bits.
  - Defines AES key localization and creation of the 128 bit initialization vector (IV) from the localized key.

$\Longrightarrow$ Internet Draft in RFC publication queue.

$\Longrightarrow$ Implementations available.

3.6 SNMP Uniform Resource Locators

- Problem:
  - No common mechanism to indicate how to contact the device for management.
  - Especially important when out-of-band IP management is used via a separate management interface
- SNMP Uniform Resource Locators
  - Use URL notation to identify SNMPv3 management communication endpoints.
  - Transport protocol selection (UDP vs. TCP) is implicit.
  - Examples:

```
snmp://snmp.example.com
snmp://tester5@snmp.example.com:8161
snmp://snmp.example.com/bridge1
snmp://snmp.example.com/bridge1;engine=0x800002b804616263

snmp://snmp.example.com//1.3.6.1.2.1.1.3.0
snmp://snmp.example.com/bridge1/1.3.6.1.2.1.2.2.1.8.*
```

$\Longrightarrow$ Internet Draft under active revision.

$\Longrightarrow$ Implementations available (fibre channel MIBs, `scli`)

3.7 Session-Based SNMP Security Model

- Problem:
  - The SNMP USM security model and VACM access control model are self-contained (following the original SNMP design goals) and do not integrate well into deployed authentication/authorization infrastructures.
  - Operators prefer to keep the number of authentication/authorization systems that must be managed to a minimum.

- Session-based security model (SSM):
  - Uses established security protocols, such as TLS/SSL or SSH.
  - SSM binds session to a connection, requiring SNMP over TCP.
  - Must map security protocol parameters into the SNMP parameters `securityName` and `securityLevel`.
  - Improved efficiency compared to USM under normal network conditions due to shared state about the security session.
  - Authentication should integrate with technologies such as Radius or SASL.

$\Longrightarrow$ Details still need to be worked out - no working group yet.

4 XML-based Management

4.1 XML Technologies Overview

4.2 NetConf Proposal based on JunoScript

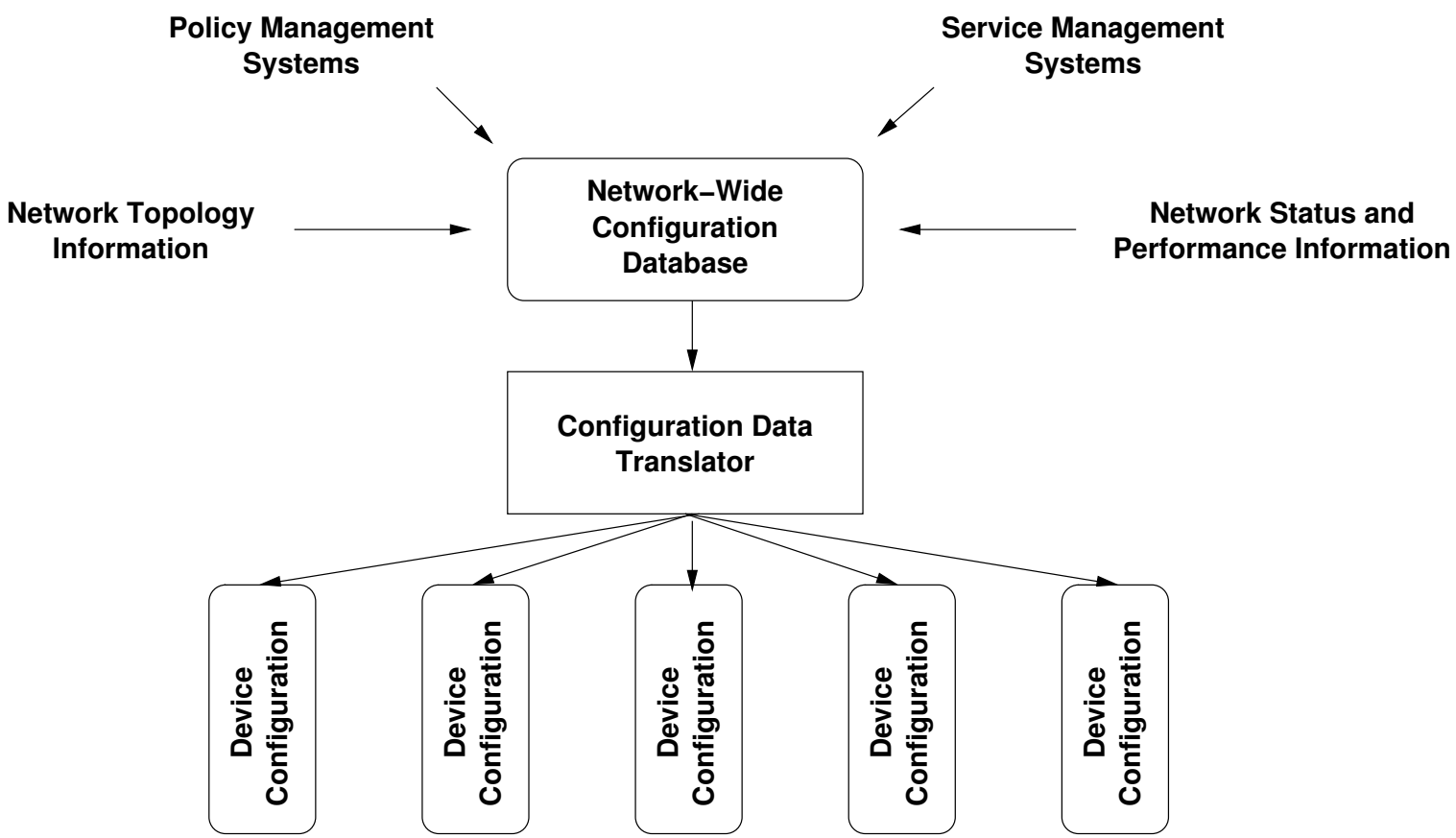
4.3 NetConf Proposal using Web Services

4.4 SMI Translations and SNMP Gateways

4.1 XML Technologies Overview

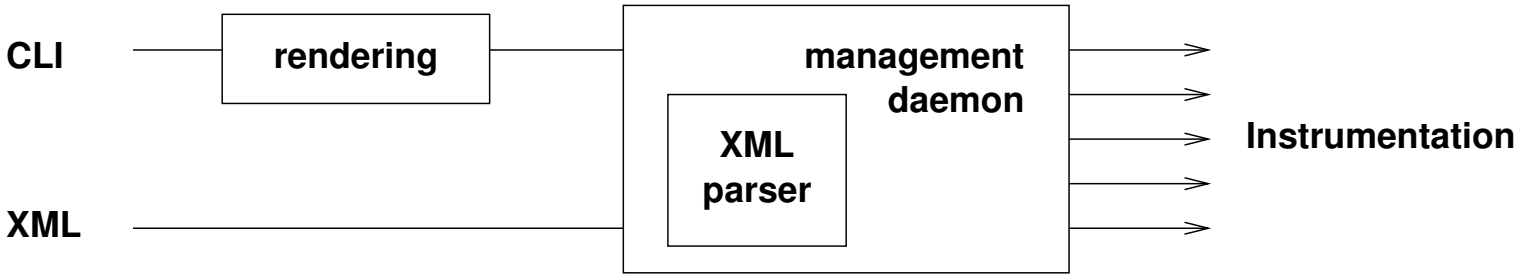
XML	The eXtensible Markup Language is a standard markup language that allows application s to exchange structured documents.
XSD	The XML Schema Definition language offers facilities for describing the structure and constraining the contents of XML documents.
XSL	The eXtensible Stylesheet Language is a family of recommendations for defining XML document transformation and presentation.
XSLT	The eXtensible Stylesheet Language Transformations is a language for transforming XML documents into other XML documents.
XPATH	The XML Path Language is a language for addressing parts of an XML document.
XQUERY	The XML Query Language is a query langauge to extract data from XML documents.
SOAP	The Simple Object Access Protocol is for exchanging XML encoded messages.
WSDL	Web Services Description Language is a language to describe the behavior of collections of XML encoded messages.
DOM	The Document Object Model is a way to represent XML documents in memory.
SAX	SAX is an event-driven API to parse and access XML documents.

Network-Wide Configuration Management Model



$\Longrightarrow$ Treating configurations as documents leads naturally to the application of XML.

4.2 NetConf Proposal based on JunoScript



- Juniper Networks developed JunoScript as a programmatic interface to (and within) their router products.
- JunoScript uses XML for data representation and the protocol messages.
- The Juniper command line interface is based internally on JunoScript.
- JunoScript is a simple RPC protocol running of Telnet or SSH.
- JunoScript provides the primitives to build robust network-wide configuration management systems (timed confirmed commits).

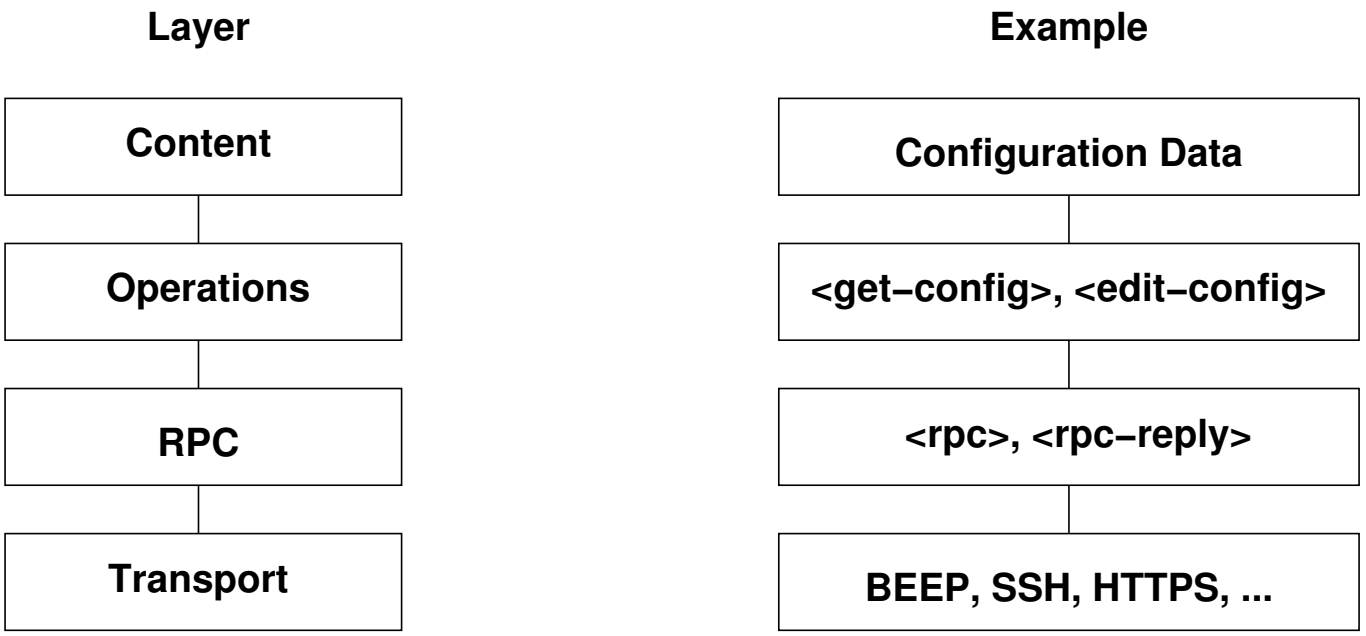
Example JunoScript RPC Interaction

```
<rpc>
  <get-interface-information>
    <statistics/>
  </get-interface-information>
</rpc>

<rpc-reply>
  <interface-information>
    <InOctets>123456</InOctets>
    <InErrors>789</InErrors>
    <OutOctets>654321</OutOctets>
    <OutErrors>0</OutErrors>
  </interface-information>
</rpc-reply>
```

- All RPC interactions over a single connection form together a single XML document.
- Filtering is based on simple subtree selection.

NetConf Layering Model



- Security has to be provided by the transport layer.
- The operations layer provides the primitives to handle configurations.
- The content layer is currently not subject to standardization.

NetConf Operations (mostly finalized)

- `get-config(source, filter)`
  Retrieve all or part of a specified configuration from a given source.
- `edit-config(target, options, config)`
  Edit target configuration, merge / replace / delete embedded in config data.
- `copy-config(source, target)`
  Create or replace an entire configuration with the contents of the source.
- `delete-config(target)`
  Delete a configuration datastore.
- `get(filter)`
  Retrieve device state information.
- `validate(source)`
  Validate the contents of the specified configuration (capability).
- `lock(source)`
  Lock a configuration source.
- `unlock(config)`
  Unlock a configuration source.
- `commit(confirmed, confirmed-timeout)`
  Commit candidate config as the new current configuration (capability).

NetConf over BEEP Request Example (under discussion)

```
MSG 1 42 . 24 291
Content-Type: text/xml; charset=utf-8
<rpc message-id="105" xmlns="http://ietf.org/xmlconf/1.0/base">
  <get-config>
    <source>
      <running/>
    </source>
    <config xmlns="http://example.com/schema/1.2/config">
      <users/>
    </config>
    <format>xml</format>
  </get-config>
</rpc>
END
```

- BEEP (RFC 3080) is a generic application protocol kernel for connection-oriented, asynchronous interactions.
- Supports exchange styles MSG/RPY, MSG/ERR, MSG/ANS, multiple channels, application layer framing and fragmentation.
- Integrates into SASL (RFC 2222) and TLS (RFC 2246) for security.

NetConf over BEEP Request Example (cont.)

```
RPY 1 42 . 24 705
<rpc-reply message-id="105" xmlns="http://ietf.org/xmlconf/1.0/base">
  <config xmlns="http://example.com/schema/1.2/config">
    <users>
      <user>
        <name>root</name>
        <type>superuser</type>
        <full-name>Charlie Root</full-name>
      </user>
      <user>
        <name>fred</name>
        <type>admin</type>
        <full-name>Fred Flintstone</full-name>
      </user>
      <user>
        <name>barney</name>
        <type>admin</type>
        <full-name>Barney Rubble</full-name>
      </user>
    </users>
  </config>
</rpc-reply>
END
```

4.3 NetConf Proposal using Web Services

- Instead of inventing a special purpose RPC protocol, use Web Service standards.

- Pros:
  - more developers available
  - more tools available
  - better integration with IT infrastructure

- Cons:
  - base technology not under control of the IETF
  - unneeded complexity
  - interoperability problems (immature technology)
  - HTTP is not a good match for a generic application protocol kernel

- Ongoing debate in the IETF how to weight the arguments.

NetConf WSDL Definition Example

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:tns="http://ietf.org/netconf/1.0/soap"
             xmlns:xb="http://ietf.org/netconf/1.0/base"
             targetNamespace="http://ietf.org/netconf/1.0/soap"
             name="http://ietf.org/netconf/1.0/soap">

  <import namespace="http://ietf.org/netconf/1.0/base" location="base.xsd"/>

  <message name="rpcRequest">
    <part name="in" element="xb:rpc"/>
  </message>
  <message name="rpcResponse">
    <part name="out" element="xb:rpc-reply"/>
  </message>

  <portType name="rpcPortType">
    <operation name="rpc">
      <input message="tns:rpcRequest"/>
      <output message="tns:rpcResponse"/>
    </operation>
  </portType>
```

NetConf WSDL Definition Example (cont.)

```
  <binding name="rpcBinding" type="tns:rpcPortType">
    <SOAP:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/
    <operation name="rpc">
      <SOAP:operation/>
      <input>
        <SOAP:body use="literal" namespace="http://ietf.org/netconf/1.0/base"/>
      </input>
      <output>
        <SOAP:body use="literal" namespace="http://ietf.org/netconf/1.0/base"/>
      </output>
    </operation>
  </binding>
</definitions>
```

- SOAP binding based on a hypothetical location for the NETCONF schema.
- Warning: This example mapping does not provide security!

NetConf WSDL Service Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:xs="http://ietf.org/netconf/1.0/soap"
             targetNamespace="urn:myNetconfService"
             name="myNetconfService.wsdl">

  <import namespace="http://ietf.org/netconf/1.0/soap" location="soap.wsdl"/>

  <service name="netconf">
    <port name="rpcPort" binding="xs:rpcBinding">
      <SOAP:address location="http://localhost:8080/netconf"/>
    </port>
  </service>
</definitions>
```

- Service definition based on a hypothetical location for the NETCONF/SOAP WSDL definitions.

NetConf over SOAP/HTTP Example

```
POST /netconf HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, text/*
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: "netconfsession:123"
Content-Length: 470

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <rpc id="101" xmlns="http://ietf.org/netconf/1.0/base">
      <get-config>
        <source>
          <running/>
        </source>
        <config xmlns="http://example.com/schema/1.2/config">
          <users/>
        </config>
        <format>xml</format>
      </get-config>
    </rpc>
  </soapenv:Body>
</soapenv:Envelope>
```

NetConf over SOAP/HTTP Example (cont.)

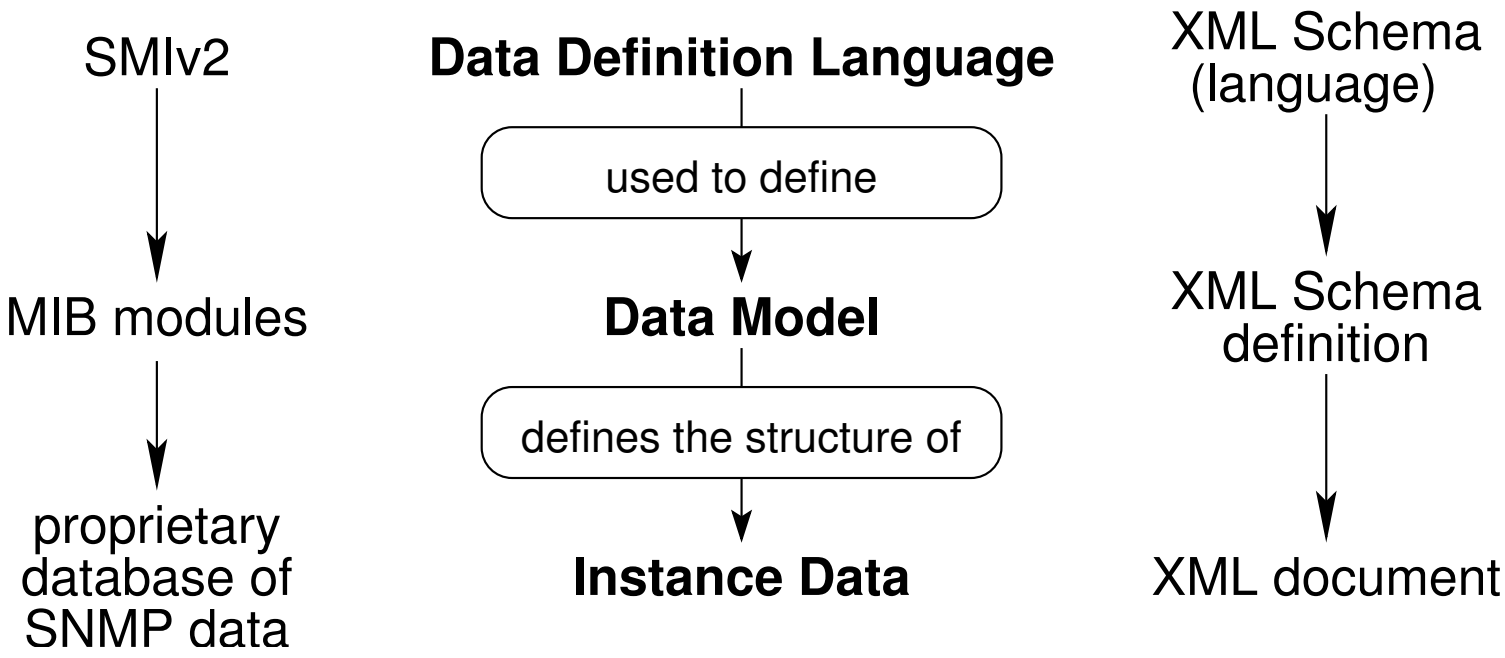
```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <rpc-reply id="101" xmlns="http://ietf.org/netconf/1.0/base">
      <config xmlns="http://example.com/schema/1.2/config">
        <users>
          <user>
            <name>root</name>
            <type>superuser</type>
          </user>
          <user>
            <name>fred</name>
            <type>admin</type>
          </user>
          <user>
            <name>barney</name>
            <type>admin</type>
          </user>
        </users>
      </config>
    </rpc-reply>
  </soapenv:Body>
</soapenv:Envelope>
```

NetConf Open Issues

- Transport: SSH (must), BEEP (may), SOAP/HTTP[S]? (may), SCTP? ...
- Filtering: ad-hoc subtree?, XPATH?, XQUERY?, XPATH light?, ...
- Protocol: primitives? locking mandatory? XML usage?
- Modeling: XML Schema?, RELAXng?, SMIng?, ...
- Integration: SNMP?, CLI?, ...

4.4 SMI Translations and SNMP Gateways



- No standard format for storing / exchanging instance data in the SNMP world.
- Automatically translate MIB modules into XML schemas to save investments.
- Such translations should follow the ”XML style” as close as possible.

Multiple ”Contexts” per XML Document

A single document may contain data

- of multiple engines (agents)
  - @ipaddr
  - @hostname
  - @port
- of multiple per-agent contexts
  - @context or
  - @community
- of multiple points in time
  - @time

```
<?xml version="1.0"?>
<snmp-data [...]>
  <context
      ipaddr="134.169.246.1"
      hostname="ciscobs.rz.tu-bs.de"
      port="161"
      community="public"
      time="2003-03-10T10:31:16Z">
    [...context data...]
  </context>

  <context [...]>
    [...context data...]
  </context>

  [...]
</snmp-data>
```

No ”deep” Element Nesting

- 1st level element `<snmp-data>`
  independent root element
  (not bound to a specific MIB, agent, or point in time)

- 2nd level elements `<context>`

- 3rd level elements e.g. `<system>`, `<ifEntry ifIndex="1">`
  - groups of scalar elements
  - table rows, identified through index attributes

- 4th level elements e.g. `<sysContact>`, `<ifInOctets>`
  - scalar elements
  - columnar elements (also of table augmentations)

- deeper level elements
  only for ”table-in-table” relationships, hence quite rare

$\Longrightarrow$ Note: The element nesting is not based on the OID tree.

Using XML Namespaces to Identify Modules

- Each MIB will be compiled to a separate XML Schema that defines an according namespace:

```
<xsd:schema
    targetNamespace="http://example.com/IF-MIB" [...]>
  [...]
</xsd:schema>
```

- Imports from MIB modules are translated to imports of namespaces:

```
<xsd:schema [...]
    xmlns:SNMPv2-MIB="http://example.com/SNMPv2-MIB" [...]>
  [...]
  <xsd:import
    namespace="http://example.com/SNMPv2-MIB" [...]/>
  [...]
</xsd:schema>
```

- Elements can be named uniquely with namespace prefixes:

```
<IF-MIB:interfaces>
  <IF-MIB:ifNumber>7</IF-MIB:ifNumber>
</IF-MIB:interfaces>
```

Value Representations and Schema Definitions

- numeric values

  XML: display hints applied, represented in decimal digits
  Schema: range restrictions (`<minInclusive>`, `<maxInclusive>`)
  display hints (`<fractionDigits>`)

- octet strings with display hints

  XML: represented as strings conforming to display hints,
  Schema: DISPLAY-HINTs converted to `<pattern>` reg-exp's (as good as possible)

- octet strings without display hints

  XML: represented as sequences of 2-digit hex values
  Schema: based on the `hexBinary` type

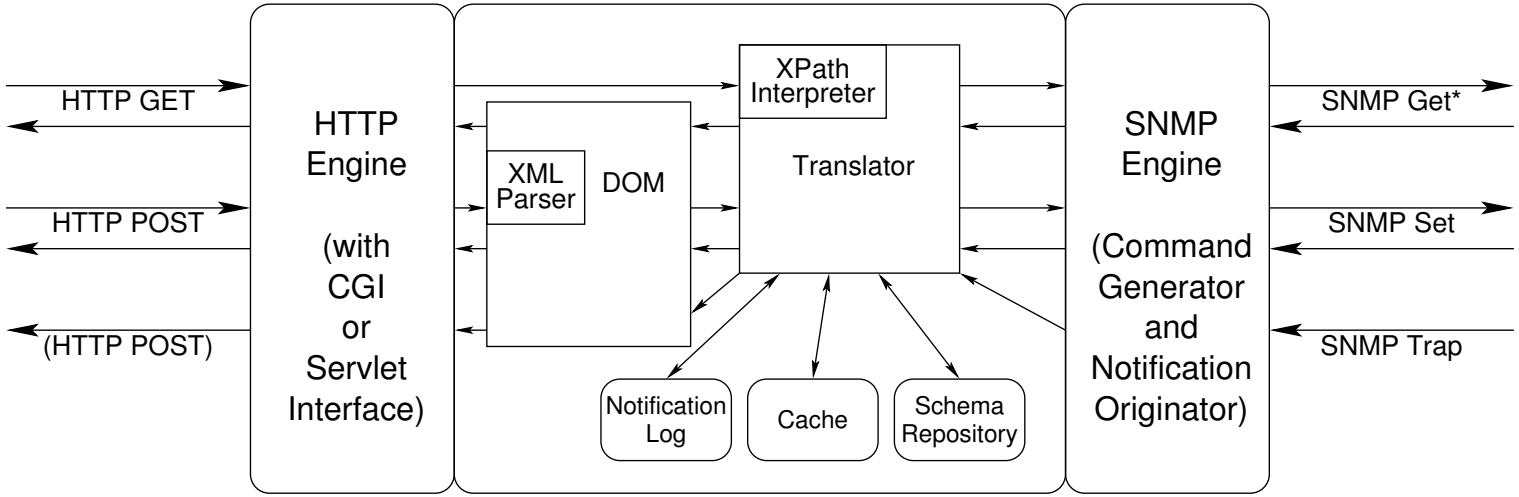
- enumeration and bit set values

  XML: represented as (sequences of) labels
  Schema: (`<list>`s of) `<enumeration>` values

Example XML Document

```
<snmp-data "xmlns=http://example.com/TCP-MIB">
  <context ipaddr="134.169.34.81" hostname="tom.example.com"
           port="161" community="public" time="2003-03-17T11:07:53Z">
    <TCP-MIB:tcp>
      <TCP-MIB:tcpRtoAlgorithm>other</TCP-MIB:tcpRtoAlgorithm>
      <TCP-MIB:tcpRtoMin>0</TCP-MIB:tcpRtoMin>
      [...]
    </TCP-MIB:tcp>
    <TCP-MIB:tcpConnEntry
        tcpConnLocalAddress="0.0.0.0" tcpConnLocalPort="9"
        tcpConnRemAddress="0.0.0.0" tcpConnRemPort="0">
      <TCP-MIB:tcpConnState>listen</TCP-MIB:tcpConnState>
    </TCP-MIB:tcpConnEntry>
    <TCP-MIB:tcpConnEntry
        tcpConnLocalAddress="134.0.0.0" tcpConnLocalPort="42077"
        tcpConnRemAddress="134.0.0.0" tcpConnRemPort="6010">
      <TCP-MIB:tcpConnState>established</TCP-MIB:tcpConnState>
    </TCP-MIB:tcpConnEntry>
  </context>
</snmp-data>
```

SNMP/XML Gateway Prototype (TU Braunschweig)



Example: Retrieve the descriptions of the interfaces at talisker.ibr.cs.tu-bs.de
that are currently in operation and transmitted or received at least one packet:

```
$ lynx -dump 'http://www.ibr.cs.tu-bs.de/snmp-xml-gw?\
  get=/snmp-data/context[@hostname="talisker.ibr.cs.tu-bs.de"]\
  /ifEntry[ifOperStatus="up" and (ifOutOctets > 0 or ifInOctets > 0)]\
  /ifDescr'
```

Online Resources

- Internet Engineering Task Force (IETF) `http://www.ietf.org/`
- IETF OPS Area `http://www.ops.ietf.org/`
- Internet Research Task Force (IRTF) `http://www.irtf.org/`
- Network Management Research Group (NMRG) `http://www.irtf.org/`
- The SimpleTimes `http://www.simple-times.org/`
- The Simple Web `http://wwwsnmp.cs.utwente.nl/`
- Tutorial Slides `http://www.faculty.iu-bremen.de/schoenw/`