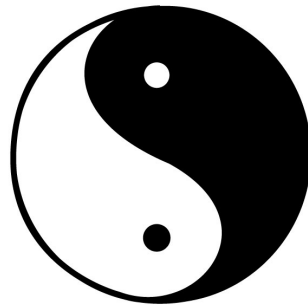


# NETCONF, YIN and YANG, BEEP, and UML



Raj Jain  
Washington University in Saint Louis  
Saint Louis, MO 63130  
Jain@cse.wustl.edu

These slides and audio/video recordings of this class lecture are at:  
<http://www.cse.wustl.edu/~jain/cse570-15/>



1. NETCONF: Network configuration protocol
2. YANG and YIN: Data modeling
3. BEEP: Transport
4. UML: Software modeling

# NETCONF

- ❑ IETF XML based Network device configuration protocol (RFC 6241, June 2011)
- ❑ Allows setting configuration parameters when the device is instantiated and changing these parameters later E.g., set IP address to 192.168.0.1
- ❑ Replacement for:
  - SNMP (Simple Network Management Protocol)
  - Command line interfaces (CLIs)
  - Scripts used by operators
- ❑ XML based ⇒ Both human and machine readable
- ❑ Also allows monitoring the device
- ❑ Uses remote procedure calls (RPCs) called “Operations”
- ❑ Runs over SSH ⇒ Secure

Ref: <https://en.wikipedia.org/wiki/NETCONF#Operations>

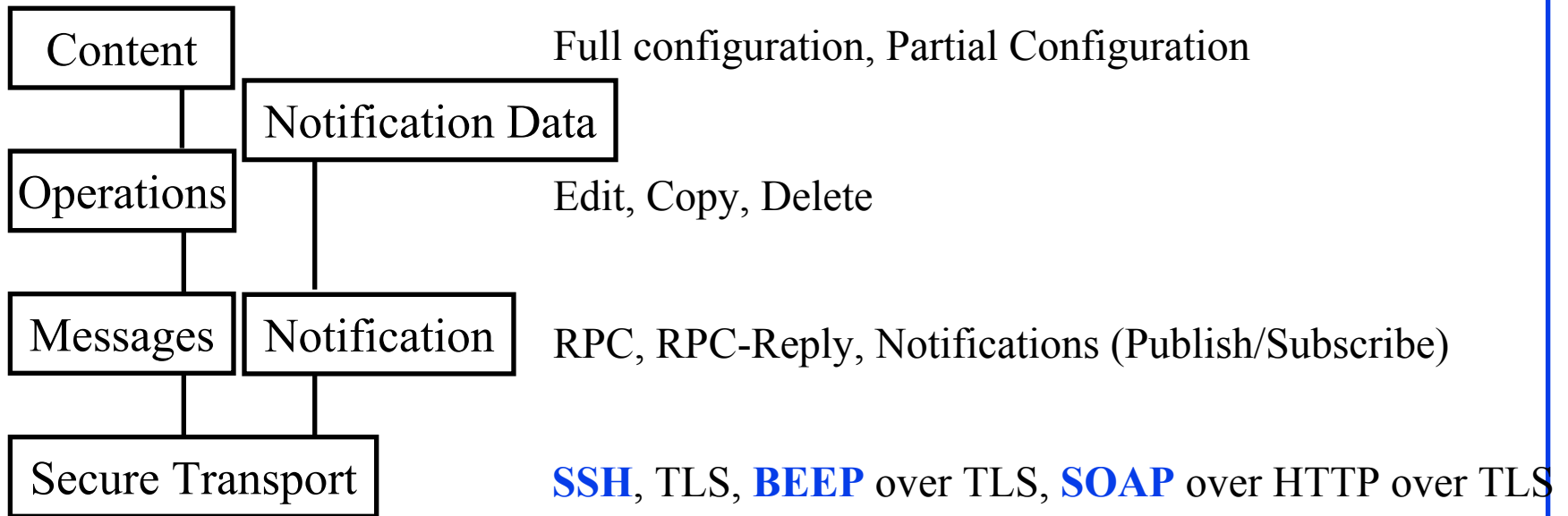
R. Enns, et al, “Network Configuration Protocol (NETCONF),” IETF RFC 6241, <https://tools.ietf.org/pdf/rfc6241>

Washington University in St. Louis

<http://www.cse.wustl.edu/~jain/cse570-15/>

©2015 Raj Jain

# NETCONF Protocol Layers



- **Notification**: Publish/subscribe mechanism to get state/alerts

Ref: netconf central, <http://www.netconfcentral.org/>

Netconf Wiki, <http://trac.tools.ietf.org/wg/netconf/trac/wiki>

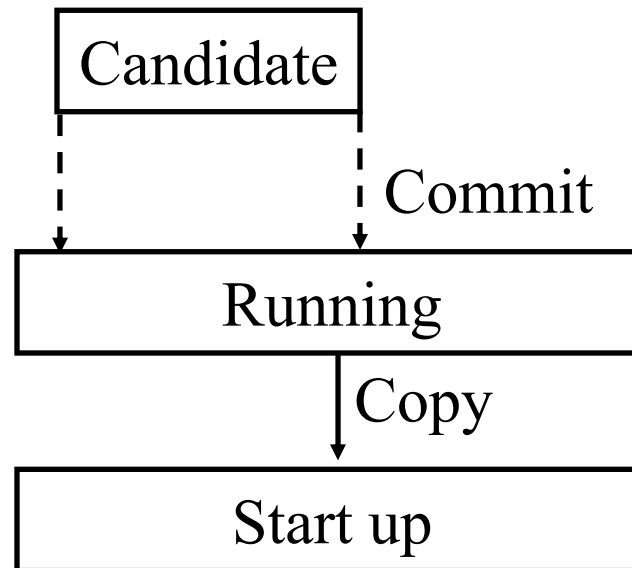
Washington University in St. Louis

<http://www.cse.wustl.edu/~jain/cse570-15/>

©2015 Raj Jain

# Configurations

1. **Running:** Complete currently running configuration
2. **Start up:** Configuration to be used on next reboot
3. **Candidate:** Part of currently running configuration. Scratch pad for configuring pieces before commit.

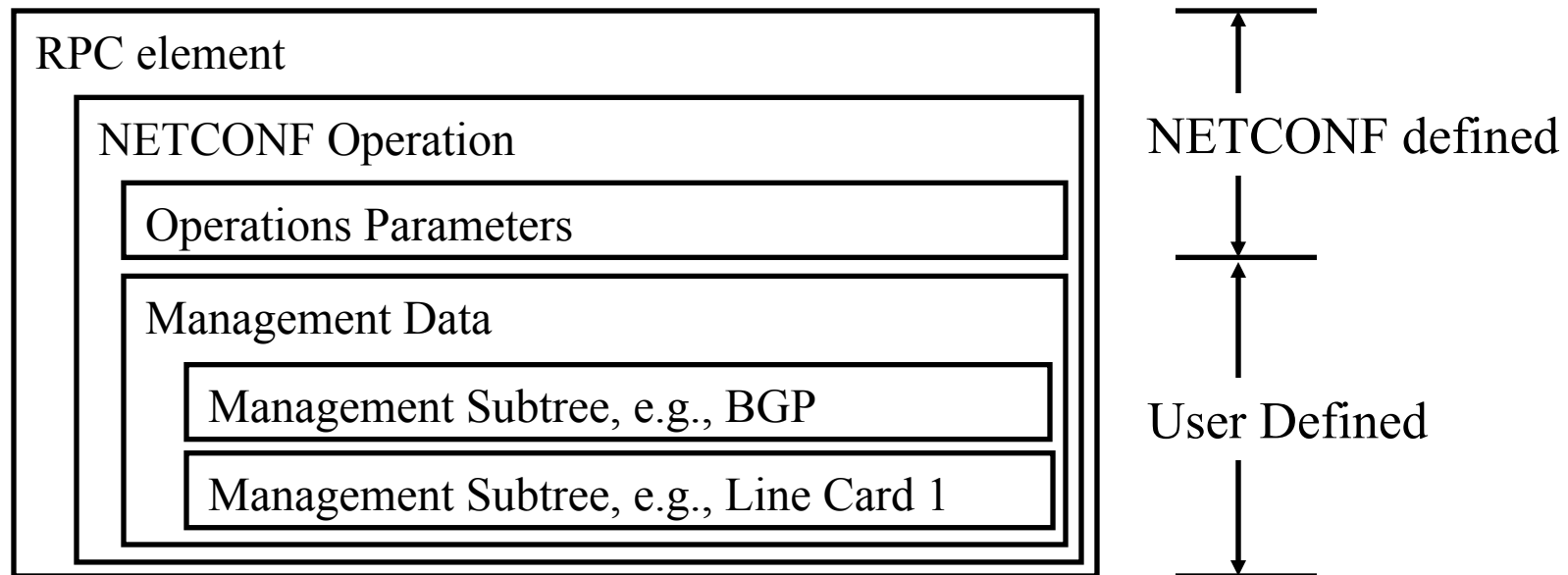


# NETCONF Operations

- ❑ Get: Get complete running configuration and state
- ❑ Get-Config: Get all or part of running configuration
- ❑ Edit-Config: Edit configuration
- ❑ Copy-Config: Copy the entire configuration store to another
- ❑ Delete-Config:
- ❑ Lock: Lock the full/partial configuration  
(so that no one else can modify)
- ❑ Unlock
- ❑ Close-Session: Graceful termination of session
- ❑ Kill-Session: Abort

# NETCONF Parameters

- ❑ Parameters are stored in a hierarchical XML file.
- ❑ Any branch or the entire tree can be over-written or retrieved



# NETCONF Example

- Show currently running BGP Configuration:

```
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">
      <top
xmlns="http://example.com/schema/1.2/config">
        <bgp/>
      </top>
    </filter>
  </get-config>
</rpc>
```

- ← Remote Procedure Call
- ← Name Space
- ← Operation
- ← Which configuration?
- ← Running configuration
- ← What subpart of configuration?
- ← User defined name space schema
- ← BGP subpart



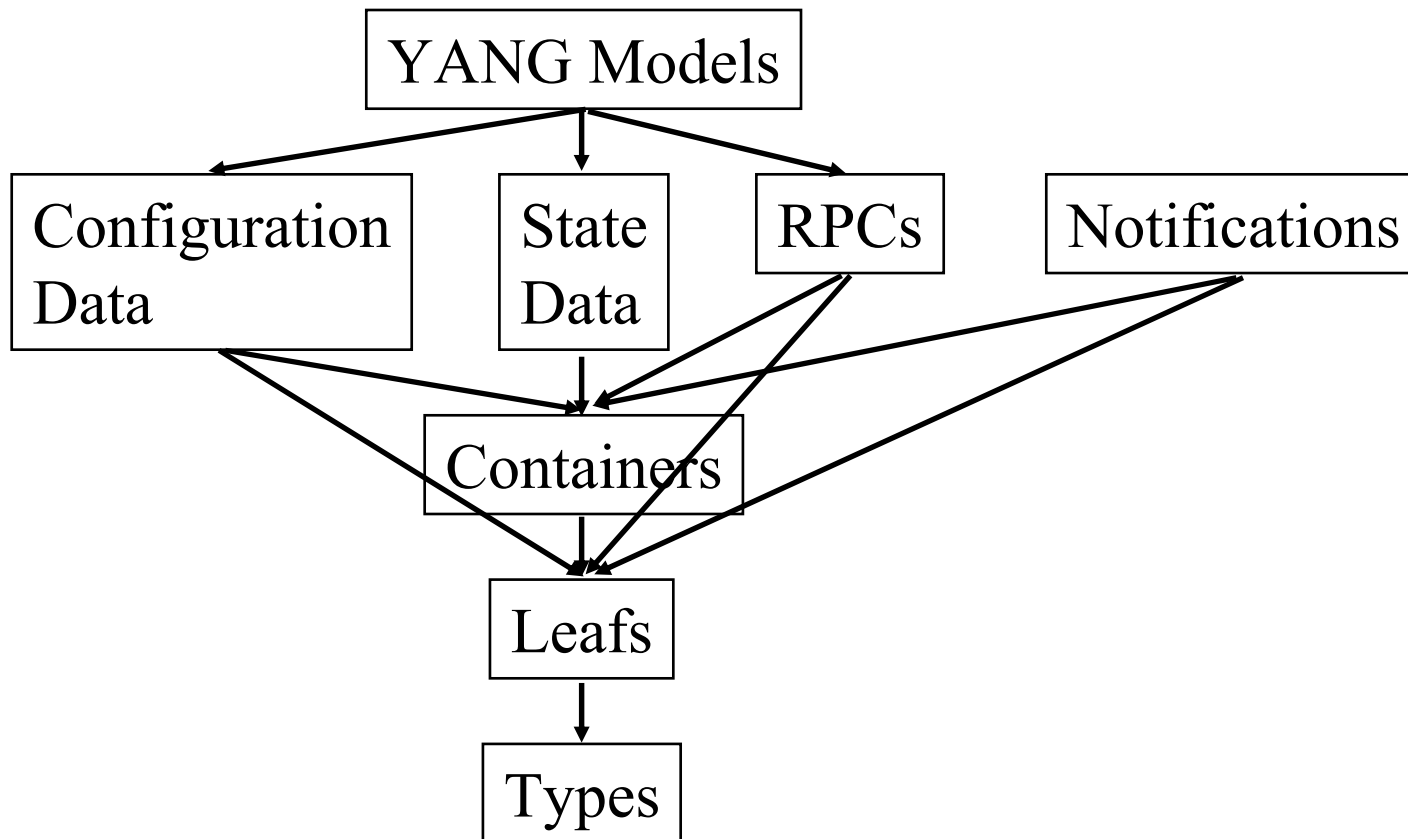
# YANG Data Model

- ❑ Yet Another Next Generation **data modeling** language.  
By IETF netmod working group
- ❑ Sequel to SMI (Structure of Management Information) used with SNMP, SMIv2 used by SNMPv2, and SMIng
- ❑ To express configuration data and state data
- ❑ **Data model**: Describes the data, its *constraints*. A.k.a., *Schema*  
E.g., address may consist of street, state, zip *within* 50 states.  
1 Brookings Dr., Saint Louis, MO 63130 is an *instance*.
- ❑ YANG defines a number of built-in data types and specifies a way to construct more complex data types.

Ref: M. Bjorklund, Ed., “YANG – A Data Modeling Language for the Network Configuration Protocol (NETCONF),”  
RFC 6020, Oct 2010, <http://tools.ietf.org/html/rfc6020>

# YANG Concepts

- YANG is used for configuration data, state data, RPCs (Operations), and event notifications.



# YANG Node Types

- ❑ **Container**: A subtree of related nodes. No data values. Only a set of child nodes. Single instance. E.g., WUSTL
- ❑ **Leaf**: Has a value and no child nodes. Can have a default value. Can be mandatory or optional. Single instance. E.g., Department of CSE
- ❑ **List**: A set of list entries. Each list entry may contain many child nodes including other lists. Uniquely identified by its key value. E.g.,  
list user {  
key login-name;  
leaf login-name { type string; }  
leaf full-name { type string; }  
}

# YANG Node Types (Cont)

❑ **Leaf-List:** A set of leafs. Used to define a sequence. Leaf nodes have only one instance, while leaf-lists may have multiple instances, e.g., “Department” is a leaf-list, while “CSE” is a leaf in “WashU” container.

❑ **Typedef:** Define new types by adding to another data type

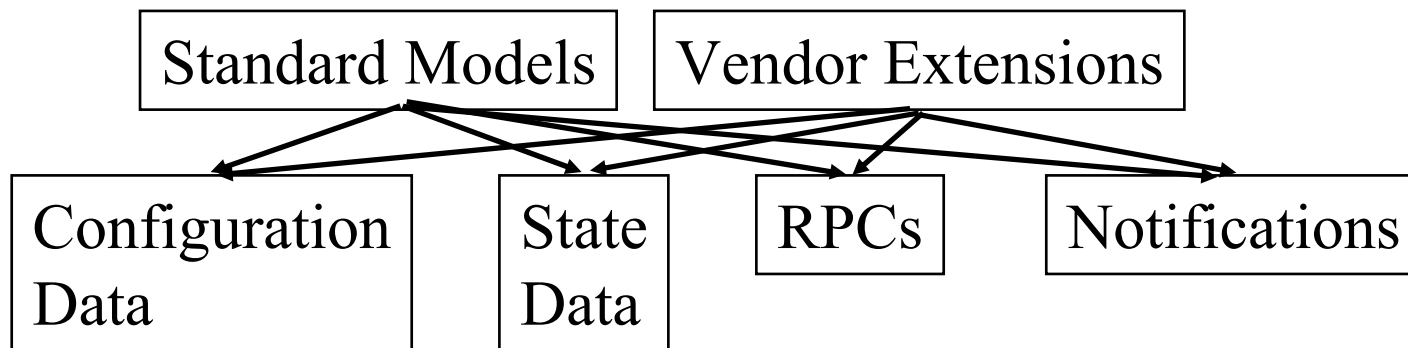
```
typedef port-number {  
    type uint16 {  
        range “1..65535”;  
    }  
}
```

❑ **Uses:** Refines and augments another data type

```
Container server {  
    Container address {  
        uses address-type;  
    }  
}
```

# YANG Node Types (Cont)

- ❑ **Anyxml**: any chunk of XML data
- ❑ **Choice**: One of  $n$  case statements. Only one is satisfied
- ❑ **Augment**: Allows vendors to add vendor-specific data to standard data modules. Should not break applications that do not understand vendor-specific data



Ref: M. Bjorklund, "YANG HighLevel Presentation, YIN, XML on the wire," IETF 72,  
<https://www.ietf.org/proceedings/72/slides/netmod-5.pdf>

# Built-in Data Types in YANG

Name	Description
binary	Any binary data
bits	A set of bits or flags
boolean	“true” or “false”
intn	n-bit signed integer, n=8, 16, 32, 64
uintn	n-bit unsigned integer, n=8, 16, 32, 64
decimal64	64-bit signed decimal number
string	Human readable string
empty	A leaf that does not have any value
enumeration	Enumerated strings
intentityref	A reference to an abstract identity
instance-identifier	References a data tree node
leafref	A reference to a leaf instance
union	Choice of member types

# YANG Examples

```
leaf host-name {  
  type string; }  
}
```

← Built-in data type

```
typedef percent {  
  type uint8 {  
    range "0..100" ; }  
}
```

← User defined data type

```
leaf-list domain-search {  
  type string; }  
}
```

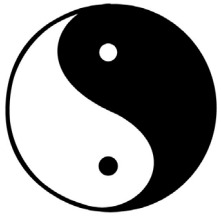
← List of domains to search

```
container food {  
  choice snack {  
    case sports-arena {  
      leaf pretzel {  
        type empty; }  
      case late-night {  
        leaf chocolate {  
          type enumeration {  
            enum dark;  
            enum milk;  
            enum first-available; } } } } }  
}
```

← No value. Only presence or absence.

← Enumeration = Set of assigned names

← Member of enumeration



# YIN

- ❑ Yin and Yang: Complementary and interrelated
- ❑ YANG is human readable  
YIN is an equivalent XML syntax for YANG.
- ❑ YANG module can be translated into YIN, manipulated by XML tools and translated back into YANG without losing any information.

$$\text{YANG}(\text{YIN}(M)) == M$$

- ❑ XML syntax useful for XML tools.  
E.g., Extensible Stylesheet Language Transformations (XSLT)
  - Extract documentation
  - Generate Code
  - Display graphically



# YIN Example

## □ YANG:

```
leaf address {  
  type inet: ip-address;  
}
```

## □ YIN:

```
<leaf name="address">  
  <type name="inet:ip-address" />  
</leaf>
```

# YANG Tools

- ❑ **Libsmi**: Generate YANG from SMIV2
- ❑ **Pyang**: Validate YANG. Translate between YANG and YIN. Generate XML schema definition (XSD) and document scheme definition language (DSDL) from YANG or YIN.
- ❑ **Yangdump**: Validate YANG. Generate XSD and HTML from YANG.

Ref: <http://www.ibr.cs.tu-bs.de/projects/libsmi>, <http://code.google.com/p/pyang>, <http://www.netconfcentral.org/download>

# Secure Transports

- ❑ SSH: Secure Shell
- ❑ TLS: Transport Level Security
- ❑ BEEP: Blocks Extensible Exchange Protocol
  - Framework for creating network application protocols
  - Provides building blocks, e.g., authentication, framing, pipelining, multiplexing, reporting, ...
  - Allows multiple parallel pipelines (channels)
  - Can define multiple profiles (sets of blocks)
  - Runs on TLS
- ❑ SOAP: Simple Object Access Protocol
  - Protocol to exchange structured information for web services
  - Can run over HTTP, SMTP, TCP, UDP, ...

# BEEP

- ❑ Blocks Extensible Exchange Protocol  
A general purpose protocol framework for exchange of data.
- ❑ Allows application developers to concentrate on their application messages and offload message exchange by using ready made BEEP code.
- ❑ BEEP implementations in C, Java, Pascal, C++, Python, JavaScript, Ruby, TCL, perl are available at [beepcore.org](http://beepcore.org)
- ❑ Like XML, BEEP is eXtensible and most applications can be implemented on the top of BEEP.
- ❑ After the success of HTTP, many applications started modifying HTTP to suite their applications, e.g., Internet Printing Protocol added a few new headers to HTTP.
- ❑ HTTP is a stateless client-server protocol.  
Not easy to use for stateful or peer-to-peer applications.

Ref: E. Dumbill, "XML Watch: Bird's-eye BEEP," Dec 2001,

<http://www.ibm.com/developerworks/webservices/library/x-beep/x-beep-pdf.pdf>

Washington University in St. Louis

<http://www.cse.wustl.edu/~jain/cse570-15/>

©2015 Raj Jain

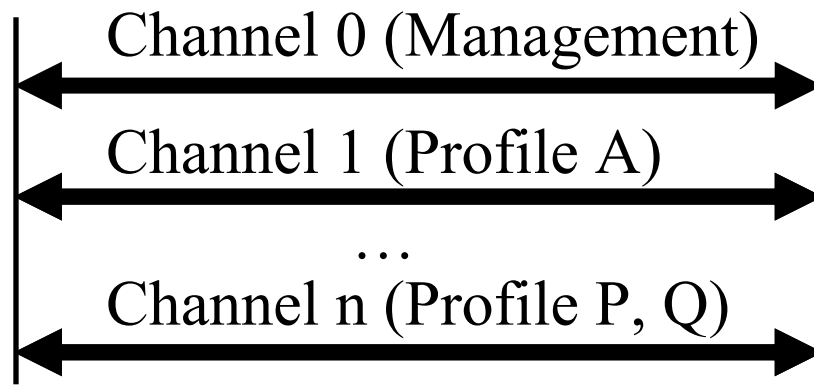
# BEEP (Cont)

- ❑ BEEP is designed for applications that are:
  - Connection Oriented: Connect, Exchange,..., Exchange, Disconnect.
  - Message Oriented: Loosely coupled peers communicating using messages
  - Asynchronous: Multiple parallel exchangesExample: HTTP, FTP, SMTP. Not good for one-shot exchanges, e.g., DNS
- ❑ BEEP Provides the following functions:
  - Separating one message from next (Framing)
  - Multiple parallel asynchronous exchanges
  - Negotiating encryption, authentication
  - Reporting errors

Ref: M. Rose, "The Blocks Extensible Exchange Protocol Core," RFC 3080, March 2001,  
<http://www.ietf.org/rfc/rfc3080.txt?number=3080>

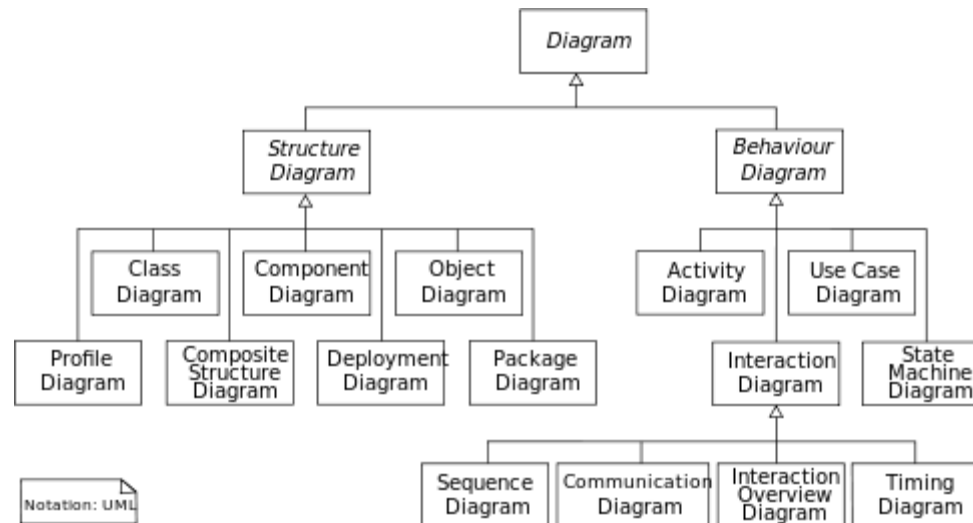
# BEEP

- ❑ *Initiator* sends a connection request to *listener* and sets up a bidirectional *session*. Multiple *channels* are then setup.
- ❑ *Channel 0* is used for managing other channels
- ❑ Some *tuning* channels are used for negotiating profiles such as encryption and authentication for other *data exchange* channels
- ❑ Libraries of standard profiles are available as XML Document Type Definitions (DTDs).
- ❑ Many application profiles use XML to encode their messages



# Unified Modeling Language (UML)

- ❑ UML is a modeling language for software engg  
Standardized by the Object Management Group (OMG) and ISO
- ❑ Structural diagrams show the static view of objects, attributes, operations, and relationships
- ❑ Behavior diagrams show the dynamic behavior in terms of collaborations among objects and state changes



# UML Diagrams

1. Class Diagram: Attributes and relationships of systems classes
2. Component Diagram: System components and their dependencies
3. Composite Structure Diagram: Internal structure of a class
4. Deployment Diagram: Hardware used in implementations
5. Object Diagram: Structure of a sample modeled system
6. Package Diagram: Logical groupings inside a system
7. Profile Diagram: profiles of various classes
8. Activity Diagram: Work flows of components in a system
9. UML State Machine Diagram: State transition diagram
10. Use Case Diagram: Actors and their goals in some use cases
11. Communication Diagram: communication between components
12. Interaction Overview Diagram: Interactions between communication diagrams
13. Sequence Diagram: Sequence of messages between objects
14. Timing Diagram: Shows timing constraints

Ref: [http://en.wikipedia.org/wiki/Unified\\_Modeling\\_Language](http://en.wikipedia.org/wiki/Unified_Modeling_Language)



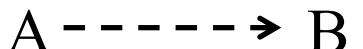



Washington University in St. Louis

<http://www.cse.wustl.edu/~jain/cse570-15/>

©2015 Raj Jain

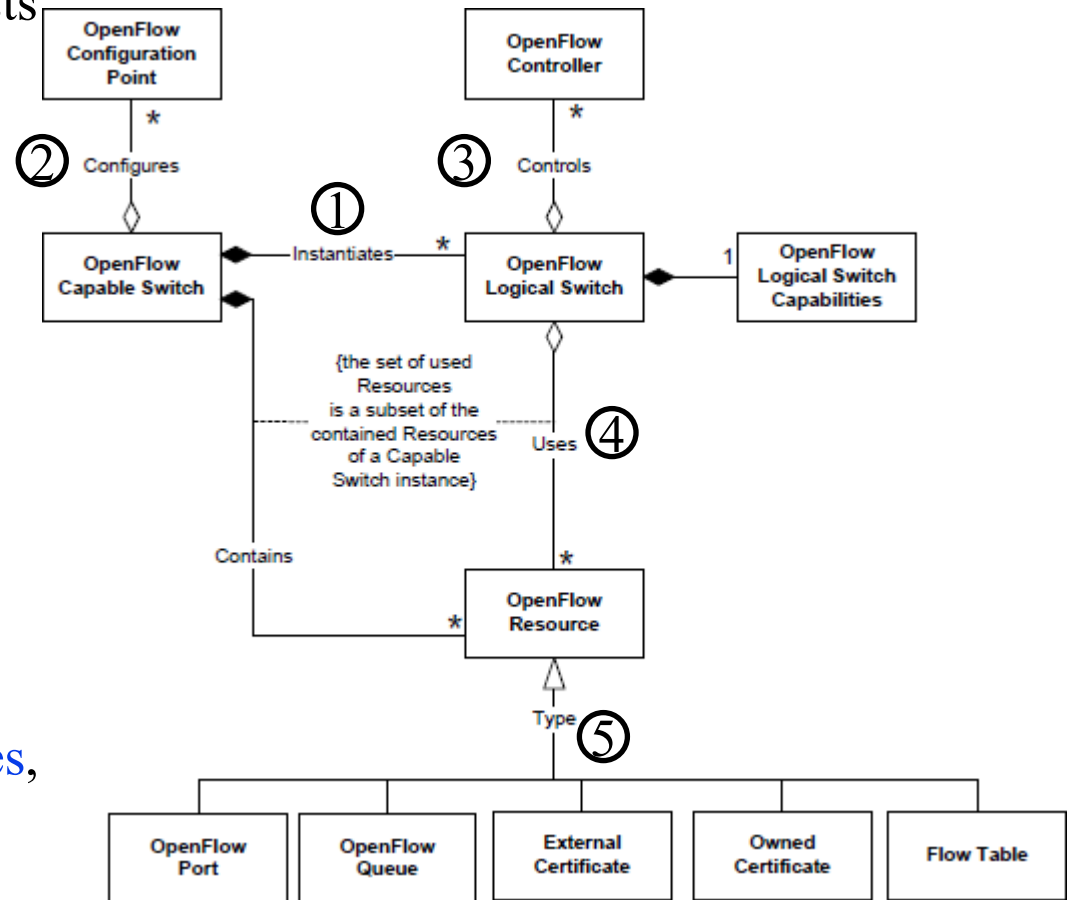


# UML Diagram Notation

- ❑ Unidirectional Association: 
- ❑ Bidirectional Association: 
- ❑ Dependency: Change to A will cause change to B 
- ❑ Aggregation: A is a part of B 
- ❑ Composite Aggregation: A (Child) and B (Parent) are tightly coupled such that child can not exist without the parent. 
- ❑ Generalization: A is a subclass of B 

# Sample UML Class Diagram for OF-Config Data Model

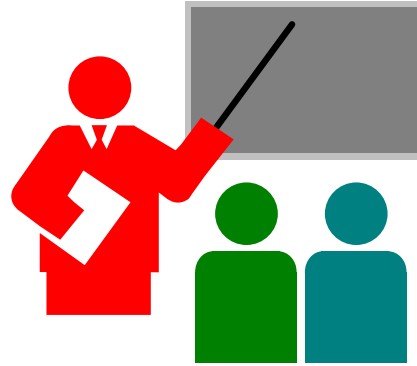
1. “OF capable switch” consists of many logical switches
2. OF capable switch is configured by many “OF configuration points”
3. “OF logical switch” is controlled by many “OF controllers”
4. Logical switch uses many “OF resources”
5. Resource types are Ports, Queues, External certificates, own certificate, and flow tables



Source: OF-Config-1-1-1

Ref: Open Networking Foundation, “OpenFlow Management and Configuration Protocol (OF-Config 1.1.1),” March 23, 2013,  
<https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1-1-1.pdf>

# Summary



1. NETCONF is the network device configuration protocol (next generation of SNMP)
2. YANG is the human-readable data modeling language (next generation of SMI)
3. YIN is the XML version of YANG modules
4. BEEP is the message exchange transport protocol
5. UML is the software modeling language

# Reading List

- ❑ A. Clemm, "Network Management Fundamentals," Section 8.1.4, Cisco Press, 2006, 552 pp., ISBN: 1587201372 (Safari Book)
- ❑ Netconf central, <http://www.netconfcentral.org/>
- ❑ M. Bjorklund, "YANG HighLevel Presentation, YIN, XML on the wire," IETF 72, <https://www.ietf.org/proceedings/72/slides/netmod-5.pdf>
- ❑ M. Bjorklund, Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)," RFC 6020, Oct 2010, <http://tools.ietf.org/html/rfc6020>
- ❑ E. Dumbill, "XML Watch: Bird's-eye BEEP," Dec 2001, <http://www.ibm.com/developerworks/webservices/library/x-beep/x-beep-pdf.pdf>
- ❑ M. Rose, "The Blocks Extensible Exchange Protocol Core," RFC 3080, March 2001, <http://www.ietf.org/rfc/rfc3080.txt?number=3080>
- ❑ R. Maksimchuk, E. Naiburg, "UML for Mere Mortals," Addison-Wesley Professional, 2004, pp. 288, ISBN:0-321-24624-1 (Safari Book)

# Wikipedia Links

- ❑ [https://en.wikipedia.org/wiki/Network\\_management](https://en.wikipedia.org/wiki/Network_management)
- ❑ <https://en.wikipedia.org/wiki/NETCONF>
- ❑ <https://en.wikipedia.org/wiki/YANG>
- ❑ <https://en.wikipedia.org/wiki/Beep>
- ❑ [https://en.wikipedia.org/wiki/Unified\\_Modeling\\_Language](https://en.wikipedia.org/wiki/Unified_Modeling_Language)
- ❑ <https://en.wikipedia.org/wiki/XSLT>
- ❑ <https://en.wikipedia.org/wiki/SOAP>

# Acronyms

- ❑ BEEP            Blocks Extensible Exchange Protocol
- ❑ BGP            Border Gateway Protocol
- ❑ CLI            Command line interface
- ❑ CSE            Computer Science and Engineering
- ❑ DNS            Domain Name System
- ❑ DSDL           Document Scheme Definition Language
- ❑ DTD            Document Type Definition
- ❑ FTP            File Transfer Protocol
- ❑ HTML           Hyper Text Markup Language
- ❑ IETF            Internet Engineering Taskforce
- ❑ IP             Internet Protocol
- ❑ ISO            Internet Standards Organization
- ❑ NETCONF      Network configuration protocol
- ❑ OF-Config     OpenFlow Management and Configuration Protocol
- ❑ OF            OpenFlow
- ❑ OMG            Object Management Group

# Acronyms (Cont)

- ❑ RPC Remote Procedure Call
- ❑ SMI Structure of Management Information
- ❑ SMIng Structure of Management Information Next Generation
- ❑ SMIv2 Structure of Management Information Version 2
- ❑ SNMP Simple Network Management Protocol
- ❑ SNMPv2 Simple Network Management Protocol Version 2
- ❑ SOAP Simple Object Access Protocol
- ❑ SSH Secure Shell
- ❑ TCL Tool Command Language
- ❑ TCP Transmission Control Protocol
- ❑ TLS Transport Layer Security
- ❑ UDP User Datagram Protocol
- ❑ UML Unified Modeling Language
- ❑ WashU Washington University in Saint Louis
- ❑ WUSTL Washington University in Saint Louis
- ❑ XML eXtensible Markup Language

# Acronyms (Cont)

- ❑ XSD XML Scheme Definition
- ❑ XSLT Extensible Stylesheet Language Transformations
- ❑ YANG Yet Another Next Generation Data Modeling Language
- ❑ YIN Complement of Yang