YANG/NETCONF Support in ONOS and ODL

박세형

ETRI

2018-04-24

목차

- 목표
  - YANG/NETCONF의 이해와 활용 방안에 대한 고찰
- 내용
  - What is YANG?
  - What is NETCONF?
  - YANG/NETCONF in OpenDaylight and ONOS
  - 환경 설정
  - MD-SAL 프로그래밍
  - 패킷광전송 SDN 컨트롤러 구조 설명

목표

- YANG 에 대한 이해
- NETCONF의 동작에 대한 이해
- Transport SDN Controller에 대한 이해

LFN (LF Networking Fund)

- 장비벤더와 서비스 프로바이더 등 강력한 산업계 지원
- 2018년 4월 기준의 플래티넘/골드/실버 멤버:

AT&T HUAWEI ZTE NEC/Netcracker SDN/NFV Solutions vodafone

SAMSUNG kt LG U+ SK telecom

- SDN/NFV의 새로운 오픈 프레임워크를 제공하는데 집중
  - OpenFlow에 제한된 프레임워크가 아니라 다양한 프로토콜이 독립적으로 사용할 수 있는 프레임워크
  - https://www.linuxfoundation.org/projects/networking/membership/

YANG 모델링

So What is YANG?

- NETCONF is a data modeling language for network configuration.
  - Human readable, and easy to learn representation
  - Hierarchical config for data models
  - Reusable types and groupings
  - Extensibility through augmentation
  - Support for RPCs
  - Formal constraints for config validation
  - Data modularity though modules and sub-modules
  - Well defined versioning rules

YANG

- 데이터 모델링 언어이며 NETCONF 설정 프로토콜
- 참고자료:
  - YANG introductory tutorial
  - RFC 6020 - YANG - A data modeling language for NETCONF

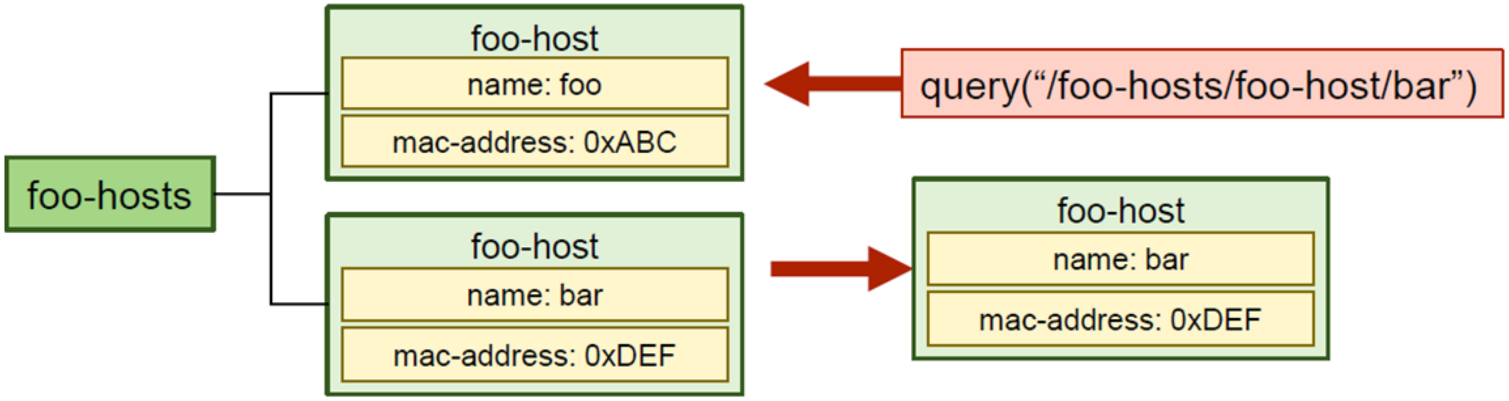
```
module hello {
     yang-version 1;
     namespace "urn:opendaylight:params:xml:ns:yang:hello";
     prefix "hello";
     revision "2015-01-05" {
          description "Initial revision of hello model";
     }
     rpc hello-world {
          input {
               leaf strin {
                    type string;
               }
     }    output {
               leaf greating {
                    type string;
          }
     }
}
```

```
package kr.re.etri.tsdn.yang.gen.v1.urn.opendaylight.params.xml.ns.yang.hello.rev150105;
```

Yang to model 1/2

```
container foo-hosts {
      list foo-host {
            key “name”;

            leaf name {
                  type string;
            }
            leaf mac-address {
                  type yang:mac-address;
            }
      }
}
```



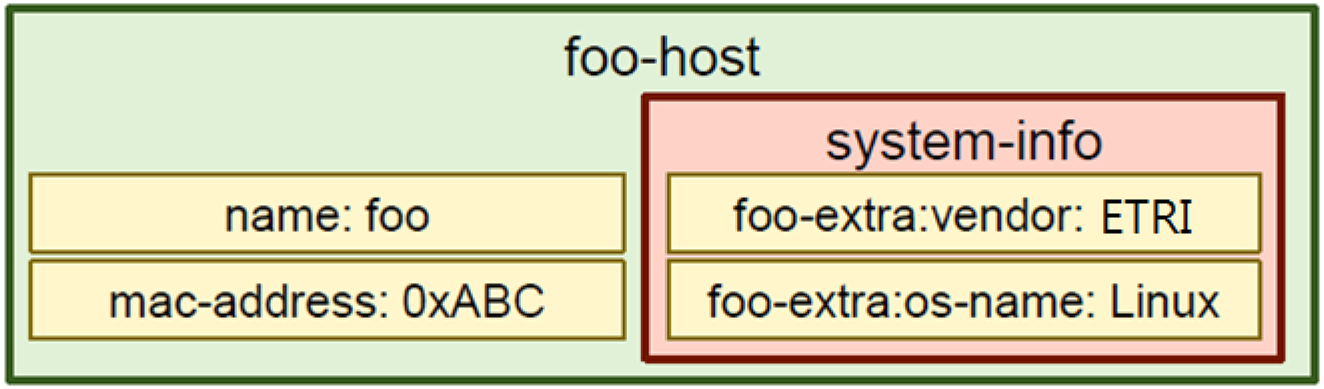
Yang to model 2/2

```
Module foo-extra {
      yang-version 1;
      namespace “urn:opendaylight:odlug:food:extra”;
      prefix fooext;

      import yang-ext { prefix etx; revision-date 2013-07-09; }
      import foo-model { prefix foo; revision-date 2015-01-24; }
      revision 2015-01-24 {}

      augment “/food:foo-hosts/foo:foo-host” {
      // yang-ext
      ext:augment-identifier “system-info”;

      leaf vendor {
            type string;
      }
      leaf os-name {
            type string;
      }
}
```



Yang: Java Bindings (1)

1) CamelCase

```
typedef foo-info {
   ......
}
```

```
public class FooInfo{
   ......
}
```

1) getter/setter

```
leaf foo-value {
  type string;
}
```

```
public String getFooValue() {
   ......
}
```

Yang: Java Bindings (2)

3) Grouping

```
grouping foo-host-info {
  leaf name {
    type string;
  }
  leaf mac-address {
    type yang:mac-address;
  }
}
```

```
package org.opendaylight.yang.gen.v1.urn.opendaylight.
  odlug.foo.rev150124;

import org.opendaylight.yangtools.yang.binding.DataObject;
import org.opendaylight.yang.gen.v1.urn.ietf.params.xml.ns.
  yang.ietf.yang.types.rev100924.MacAddress;

public interface FooHostInfo extends DataObject {
  HostName getName();
  MacAddress getMacAddress();
}
```

Yang: Java Bindings (3)

3) Container

```
container foo-hosts {
  list foo-host {
    key “name”;
    uses foo-host-info;
  }
}
```

```
import org.opendaylight.yang.gen.v1.urn.opendaylight.odlug.foo.
  rev150124.foo.hosts.FooHost;
import org.opendaylight.yang.gen.v1.urn.opendaylight.odlug.foo.
  rev150124.foo.hosts.FooHostBuilder;

// Builder
HostName name = new HostName(“foo”);
MacAddress mac = new MacAddress(“00:00:00:aa:bb:cc”);
FooHostBuilder builder = new FooHostBuilder().
  setName(name).setMacAddress(mac);

// FooHost
FooHost host = builder.build();
```

Yang: Java Bindings (4)

```
rpc add-int32 {
  input {
    leaf augend { type int32; }
    leaf addend { type int32; }
  }
  output {
    leaf sum { type int32; }
  }
}
```

```
package org.opendaylight.yang.gen.v1.urn.opendaylight.odlug.
  foo.rev150124;

import java.util.concurrent.Future;
import org.opendaylight.yangtools.yang.binding.RpcService;
import org.opendaylight.yangtools.yang.common.RpcResult;

public interface FooModuleService extends RpcService {
  Future<RpcResult<AddInt32Output>> addInt32(AddInt32Input input);
}
```

Yang: Java Bindings (5)

```
notification add-int32-called {
   leaf augend { type int32; }
   leaf addend { type int32; }
   leaf sum { type int32; }
}
```

```
package org.opendaylight.yang.gen.v1.urn.opendaylight.odlug.foo.
  rev150124;

import org.opendaylight.yangtools.yang.binding.NotificationListener;

public interface FooModuleListener extends NotificationListener {
  void onAddInt32Called(AddInt32Called notification);
}
```

YANG

```
module hello {
      yang-version 1;
     namespace
"urn:opendaylight:params:xml:ns:yang:hello";
      prefix "hello";
      revision "2015-01-05" {
            description "Initial revision of
hello model";
     }
     rpc hello-world {
            input {
                  leaf strin {
                        type string;
                  }
      }     output {
                  leaf greating {
                        type string;
            }
      }
      container helloworld {
            leaf counter {
                  type uint32;
                  config true;
            default 100;
            }
            leaf value {
                  type string;
                  config false;
            mandatory false;                    }
    }
}
```

```
module model1 {

    namespace "urn:model1";
    prefix model1;
    yang-version 1;

    revision 2015-04-06 {
        description "Initial revision";
    }

    grouping A {
        list B {
            key id;
            leaf id {
                type uint32;
            }
            leaf D {
                type uint32;
            }
        }
    }

    container C {
        uses A;
    }
}
```

NETCONF

So What is NETCONF?

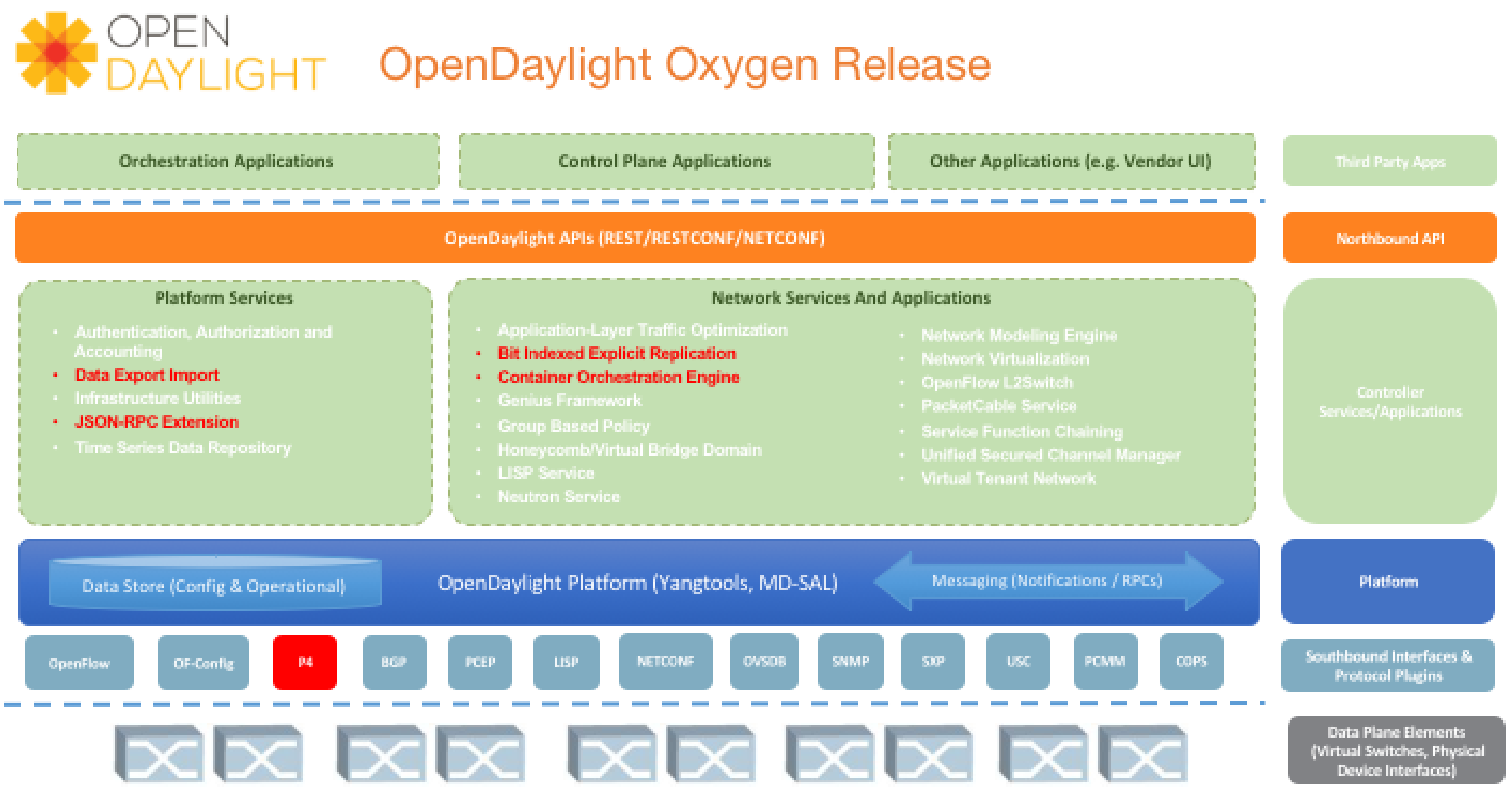
- NETCONF is an IETF network management protocol.
  - NETCONF messages are encoded in XML
  - NETCONF m
- NETCONF is an IETF network management protocol.
  - Separation of config and state data
  - Multiple data store (candidate, running, startup)
  - Transactions
  - Network-wide operation

NETCONF Base Operations

- <get>
- <get-config>
- <edit-config>
  - Test-option
  - Error-option
  - Operation
- <copy-config>
- <commit>
- <discard-changes>
- <cancel-commit>
- <delete-config>
- <lock>
- <unlock>
- <close-session>
- <kill-session>

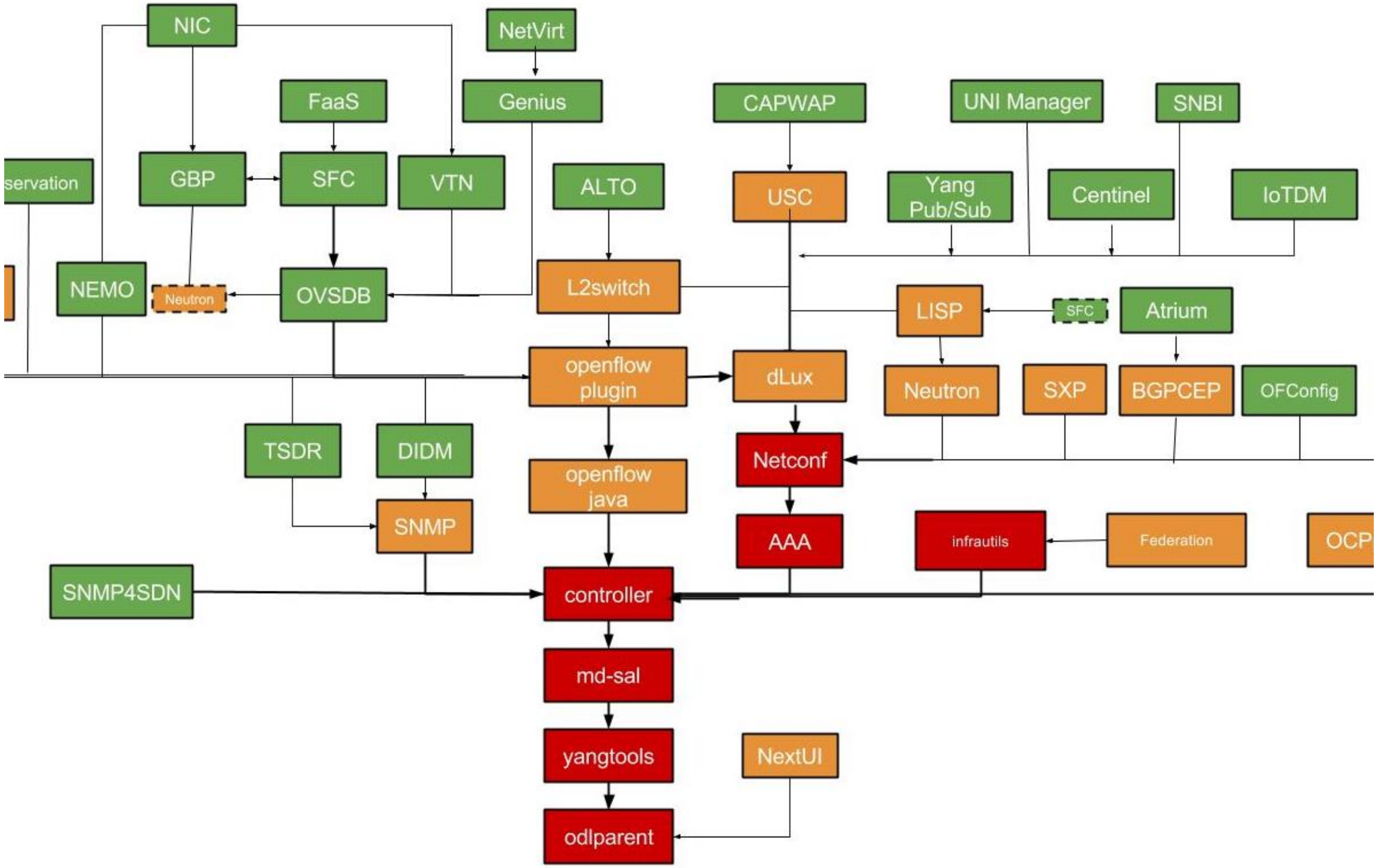
OpenDaylight

Oxygen Release (March 22, 2018)



OPEN DAYLIGHT
OpenDaylight Oxygen Release
Orchestration Applications
Control Plane Applications
Other Applications (e.g. Vendor UI)
Third Party Apps
OpenDaylight APIs (REST/RESTCONF/NETCONF)
Northbound API
Platform Services
Authentication, Authorization and Accounting
Data Export Import
Infrastructure Utilities
JSON-RPC Extension
Time Series Data Repository
Network Services And Applications
Application-Layer Traffic Optimization
Bit Indexed Explicit Replication
Container Orchestration Engine
Genius Framework
Group Based Policy
Honeycomb/Virtual Bridge Domain
LISP Service
Neutron Service
Network Modeling Engine
Network Virtualization
OpenFlow L2Switch
PacketCable Service
Service Function Chaining
Unified Secured Channel Manager
Virtual Tenant Network
Controller Services/Applications
Data Store (Config & Operational)
OpenDaylight Platform (Yangtools, MD-SAL)
Messaging (Notifications / RPCs)
Platform
OpenFlow
OF-Config
P4
BGP
PCEP
LISP
NETCONF
OVSDB
SNMP
SXP
USC
PCMM
COPS
Southbound Interfaces & Protocol Plugins
Data Plane Elements (Virtual Switches, Physical Device Interfaces)

OpenDaylight Project Dependencies

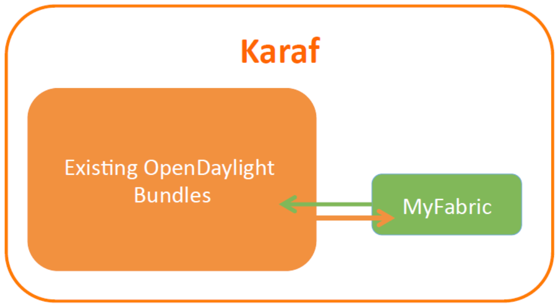


ODL 기반 프로그래밍 이란?

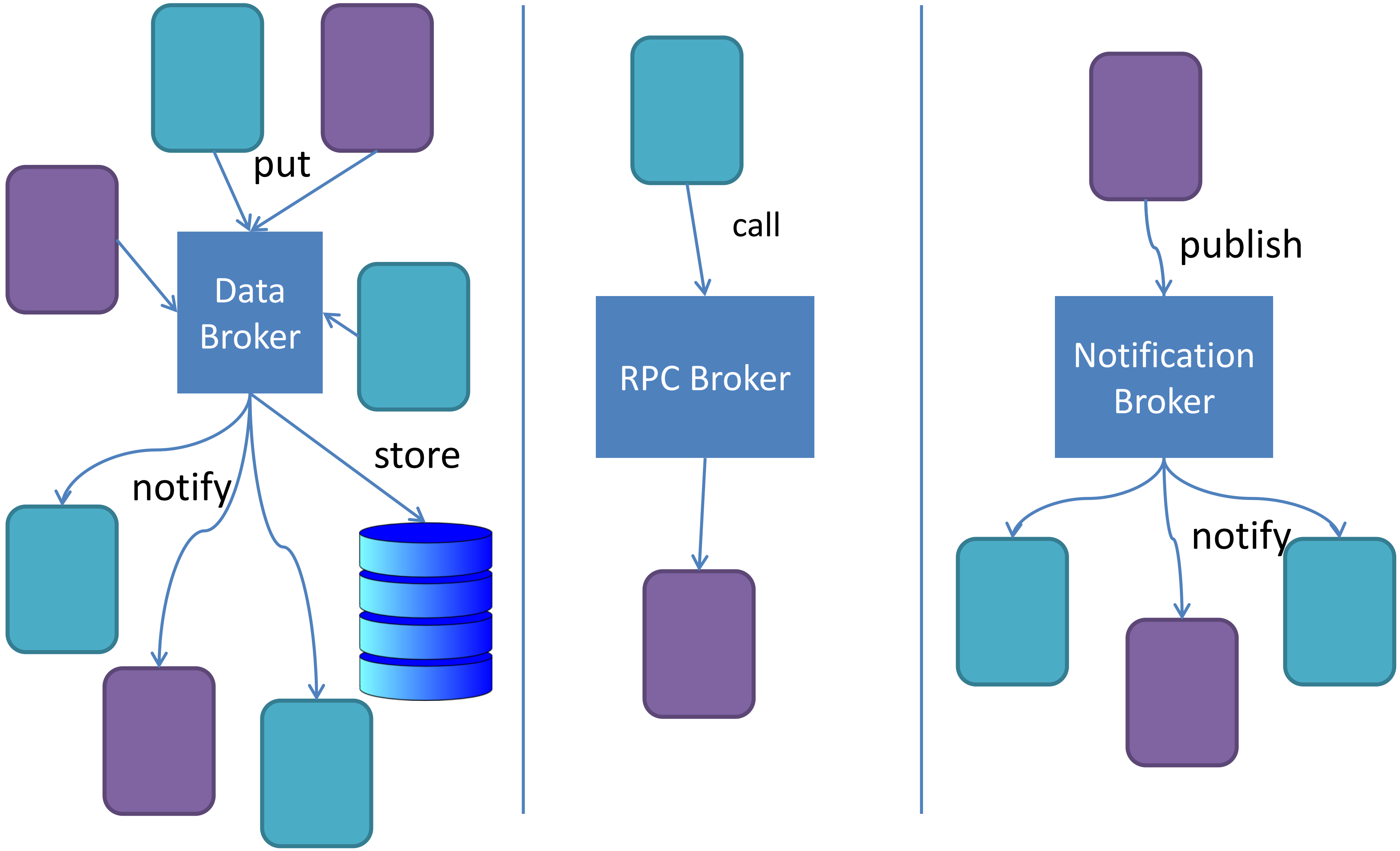
1. Model-View-Control 기법에 익숙해 지는 것
   - YANG Model 은 데이터, RPC, Notification 을 모델링
   - RESTconf는 View를 Yang모델을 기반으로 자동 생성
   - 실제로 Java를 이용하여 필요한 Code를 구현하는 부분

2. 플랫폼에 꼭 필요한 유틸에 익숙해 지는 것 (maven, git, Callable Interface, dependencies) and useful tools

3. 기존 프로젝트와 모듈의 재사용을 배우는 것
   - 다른 프로젝트의 코드를 고쳐서 사용하는 것이 아니라 바이너리 번들로 로딩하여 사용함



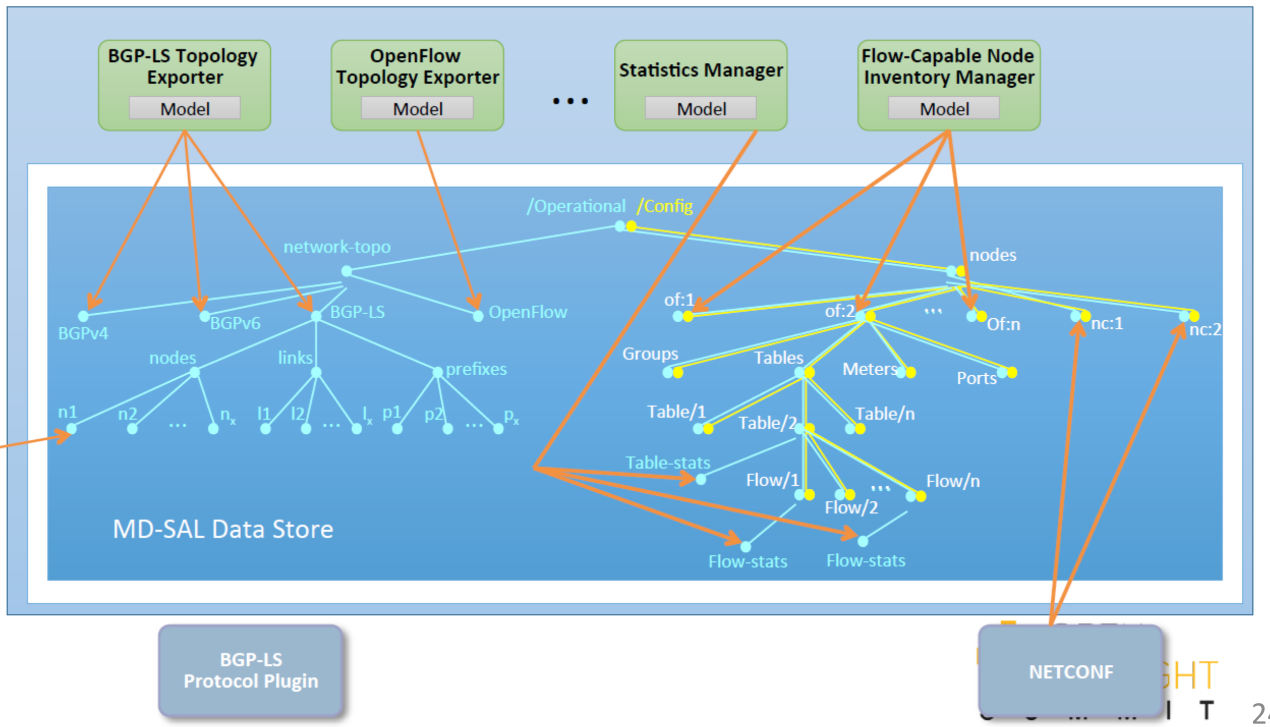
MD-SAL – 3 Brokers



data Broker

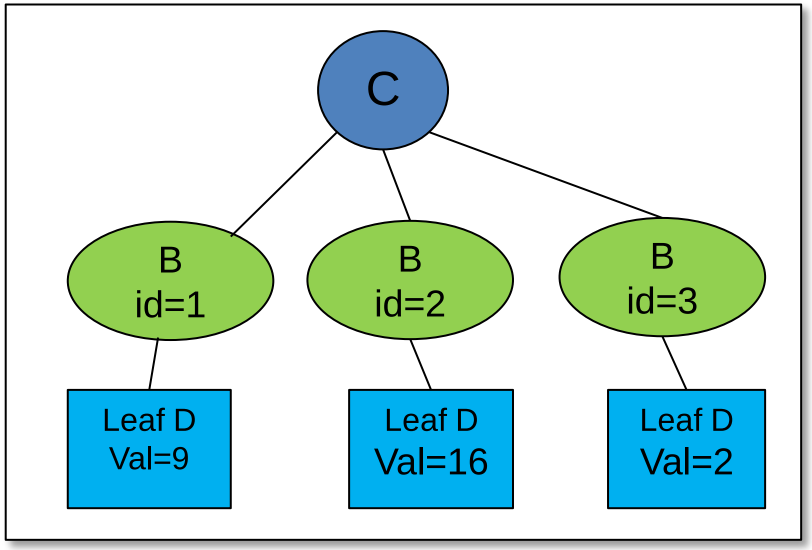
- MD-SAL은 동일 모델에 대하여 2 종류의 트리 존재
  - Config tree: 주로 응용개발자에 의해서 input 값으로 사용된다. RESTConf로 통해서 입력할 수 있다.
  - Operational tree: 주로 운영상에서 발생하는 값들을 기록하는 용도로 사용된다. 이 값은 RESTconf를 통해서 변경될 수 없다.

- Yang data is a tree
- Two Logical Data Stores:
  - Config
  - Operational
- Unified View
- InstanceIdentifier:
  - Pointer to a node



MD-SAL Data Access

- Model-driven SAL은 OpenDaylight의 커널
- Model-driven SAL은 번들간의 모든 상태 교환을 중앙 집중적으로 관여함.
- YANG 모델을 런타임 시 로딩하여 트리 구조를 만듦



MD-SAL Data Access (contd.)

- 해당 모델을 maven으로 컴파일하면 다음과 같은 클래스들이 생성 된다. Model1Data. java, B.java, and C.java

- InstanceIdentifier 자식을 가리키는 포인터로 아래의 InstanceIdentifier는 첫 번째 노드를 가리킴

```
InstanceIdentifier iid = InstanceIdentifier.builder(C.class)
     .child(B.class, new BKey((long)1))
     .build();
```

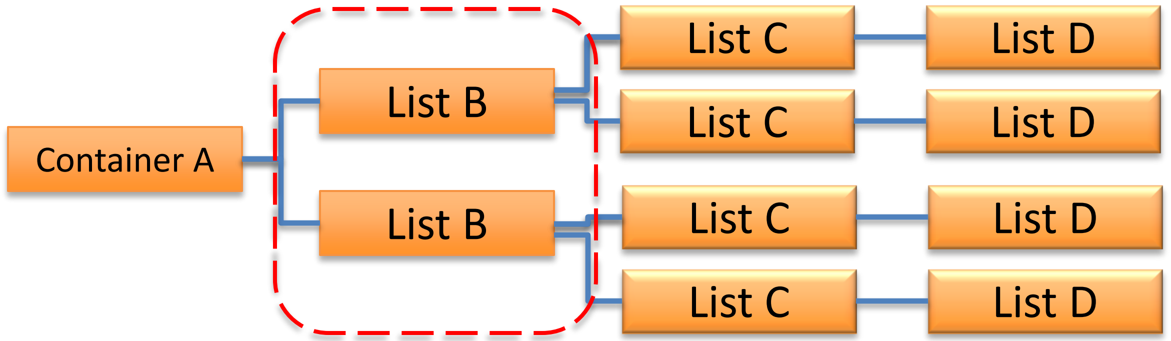
- 저장소를 읽고 쓸 때는 ReadOnlyTransaction 또는 WriteTransaction 필요

```
B updatedB = new BBuilder().setD((long)19).build();
WriteTransaction modification = dataBroker.newWriteOnlyTransaction();
modification.merge(LogicalDataStoreType.CONFIGURATION, iid, updatedB, true);
modification.submit();
```

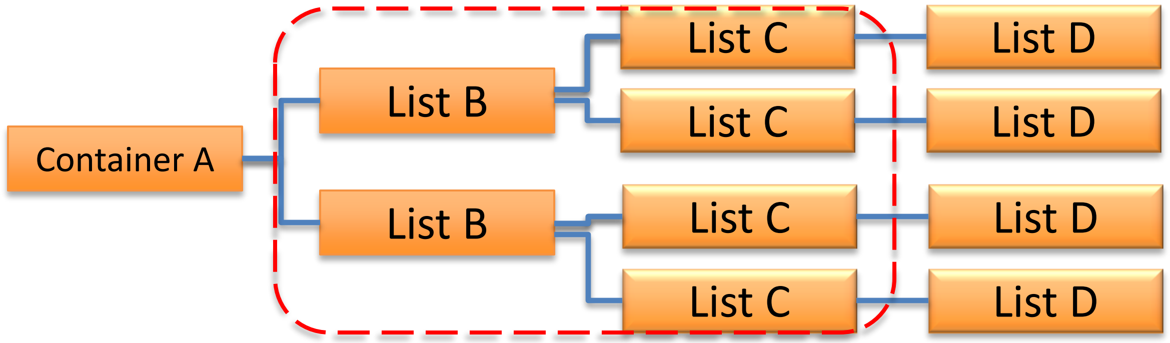
- Transaction은 배치처리 가능함

Data Broker DataChangeListener

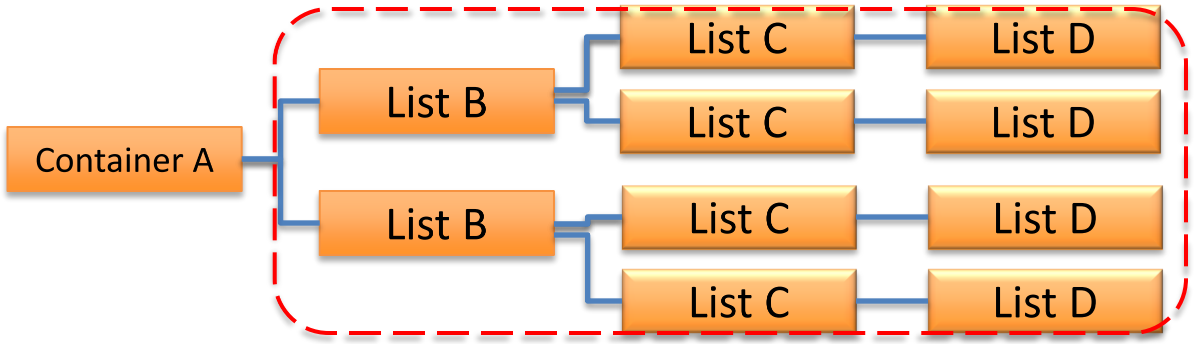
Listen("/A/B",BASE)



Listen("/A/B",ONE)

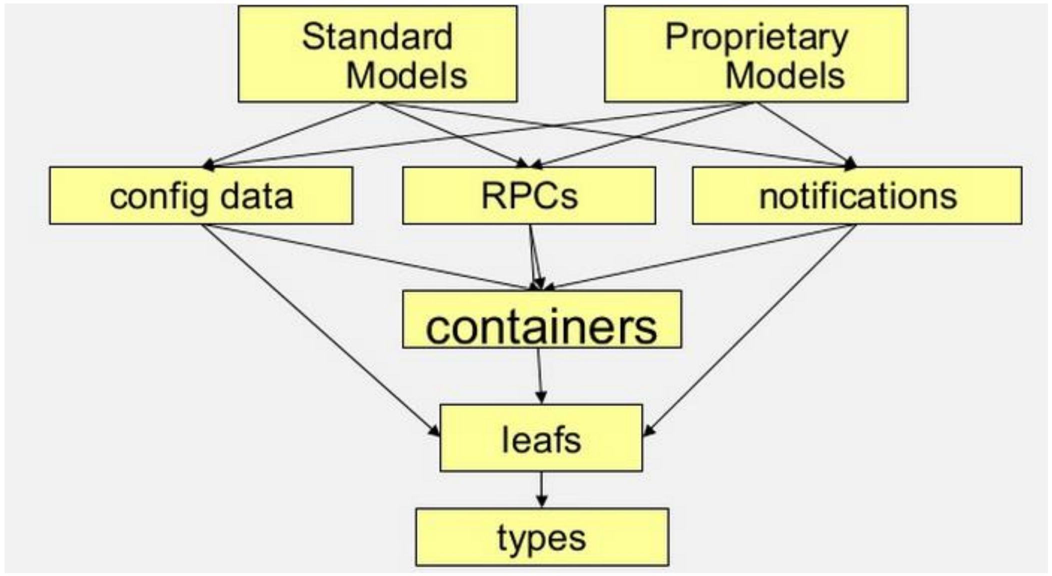


Listen("/A/B",SUBTREE)



YANG not restricted to Just Data Store

- Notifications:
  - Publish one or more notifications to registered listeners
- RPC:
  - Perform procedure call with input/output,
    without worrying about actual provider for that procedure



Using Notifications and RPCs

- Application developer usage for receiving notification and making RPC calls.

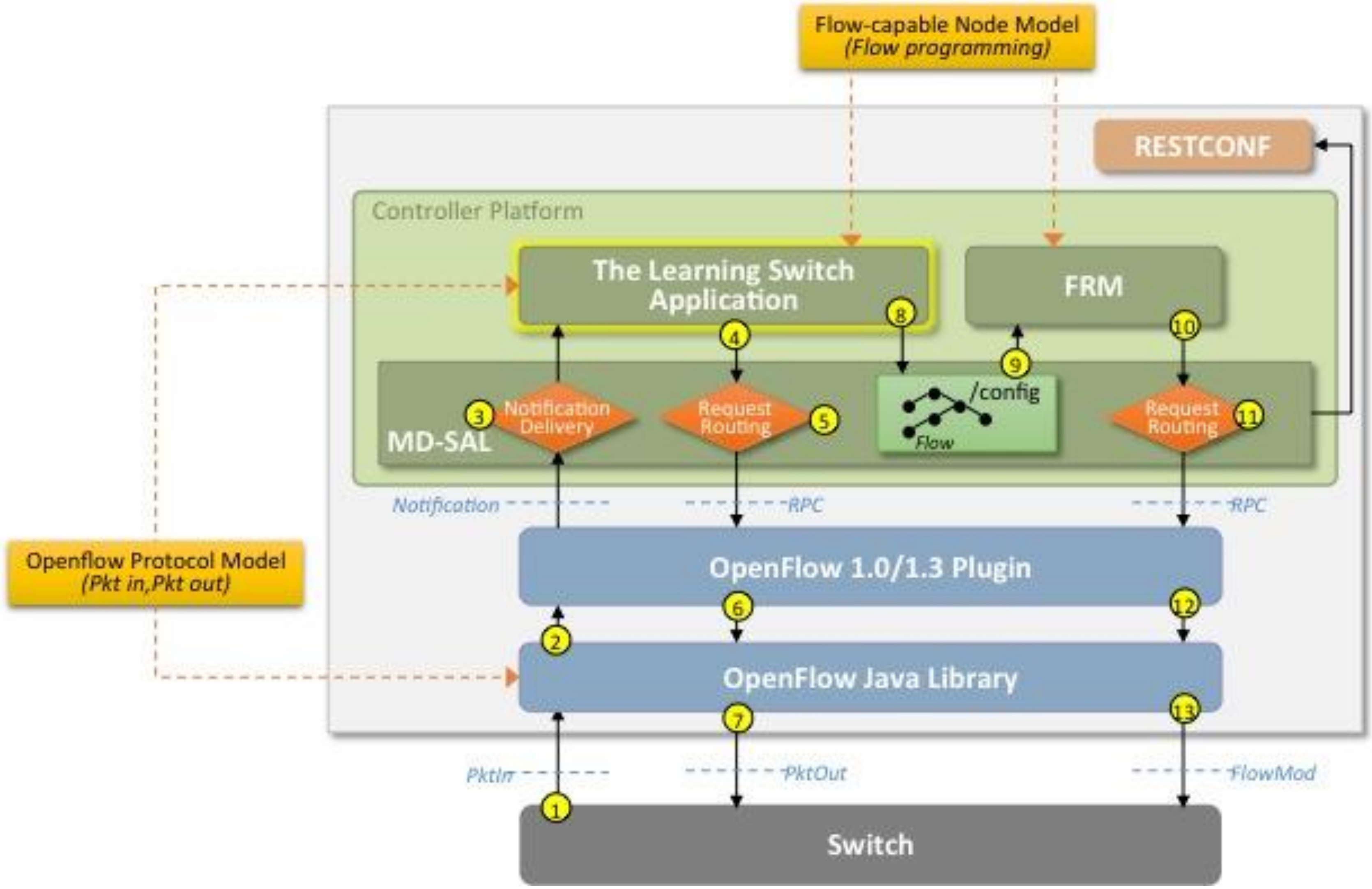
```
public class ConsumerImpl implements OpendaylightInventoryListener

     public ConsumerImpl(..) {
             notificationService.registerNotificationListener(this);
             this.salFlowService = rpcProviderRegistry.getRpcService(SalFlowService.class)
     }

     @Override
     public void onNodeUpdated(NodeUpdated nodeUpdated) {
             RemoveFlowInputBuilder flowBuilder = new RemoveFlowInputBuilder();
             flowBuilder.setBarrier(true);
             flowBuilder.setNode(NodeUtils.createNodeRef(nodeUpdated.getId()));
             salFlowService.removeFlow(flowBuilder.build());
     }
}
```

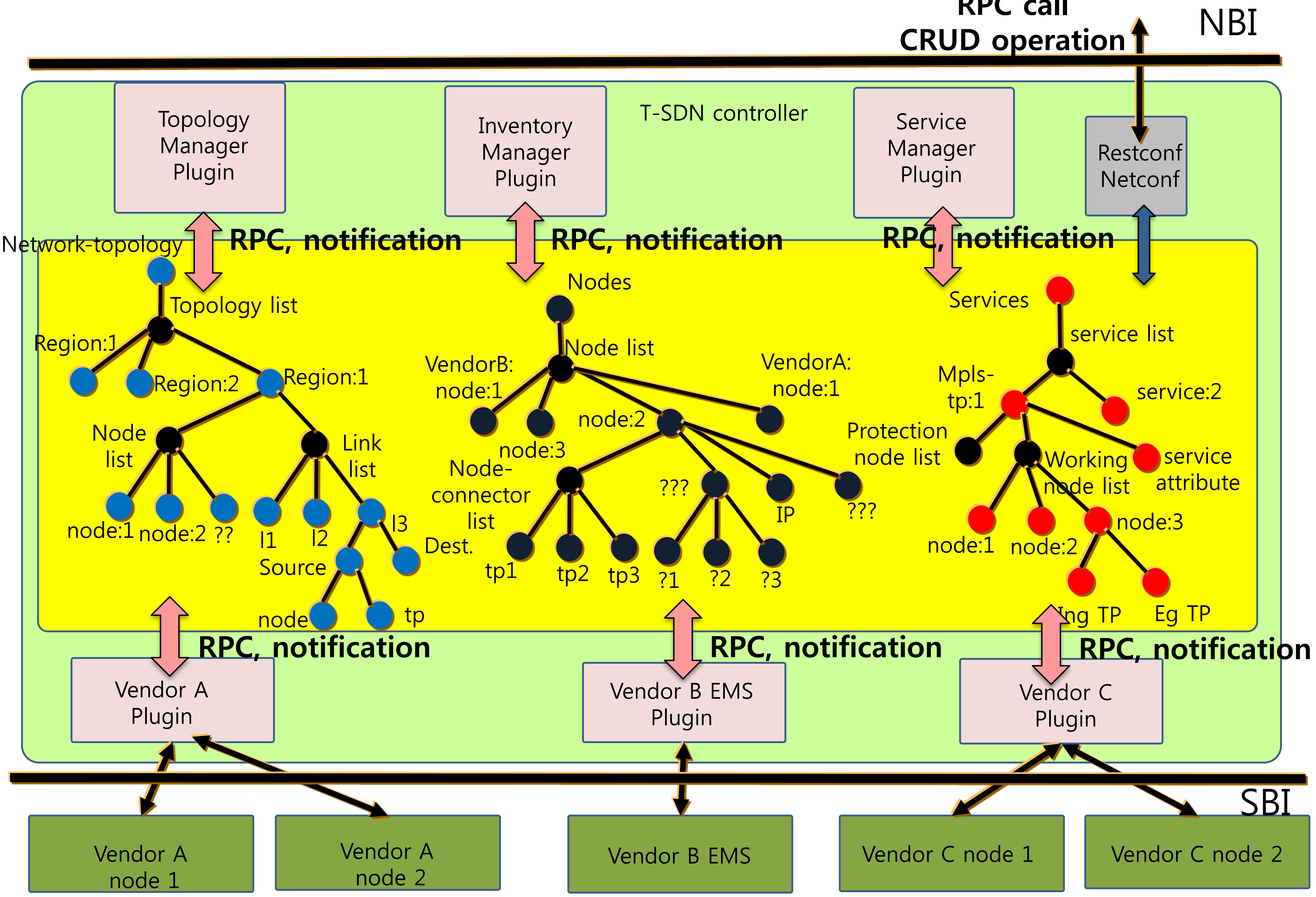
- **<u>Note</u>**: Whenever there is a change in the MD-SAL data store, you can receive a notification similar to the YANG defined notifications by implementing the DataChangeListener interface in a provider module

Example of Learning Switch
(Uses Data + Notifications + RPC)

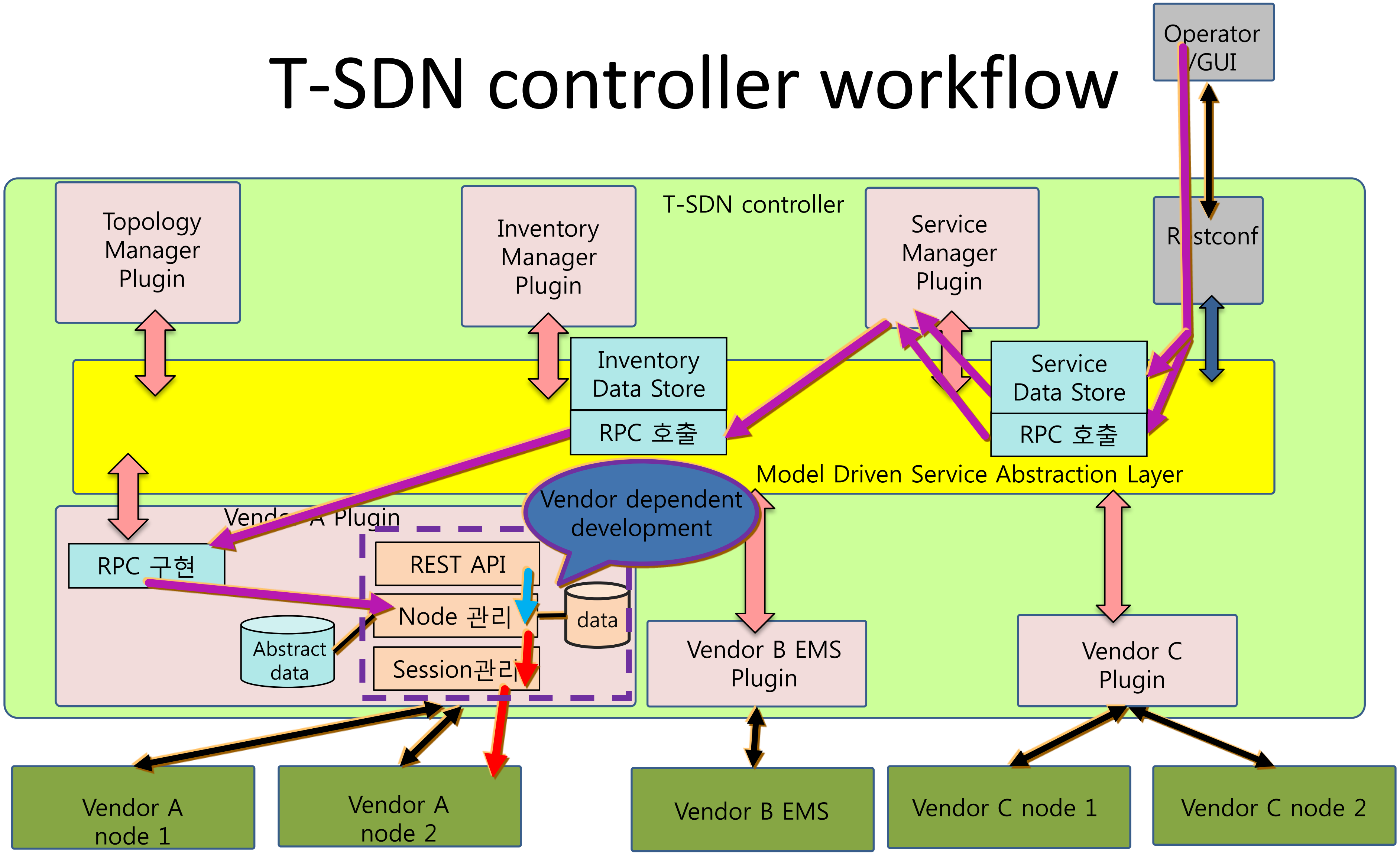


Transport SDN Controller 이해하기!

<Architecture of Calamari T-SDN Controller by ETRI> (4/1)

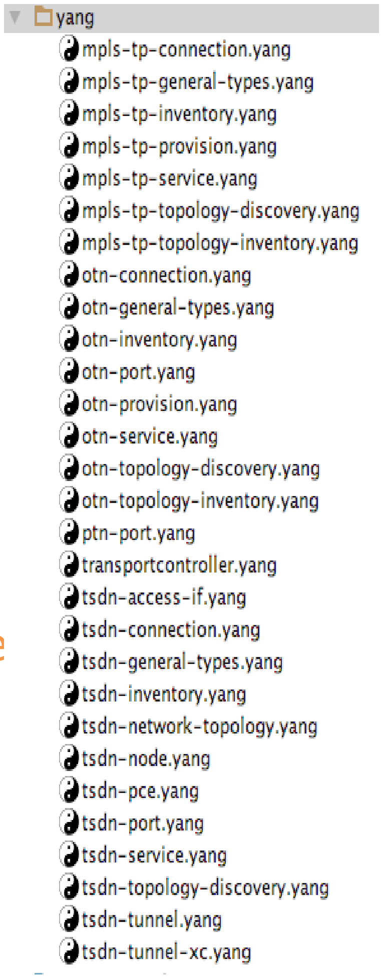

RPC call
CRUD operation
NBI
T-SDN controller
Topology Manager Plugin
Inventory Manager Plugin
Service Manager Plugin
Restconf Netconf
RPC, notification
RPC, notification
RPC, notification
Network-topology
Topology list
Region:1
Region:2
Region:1
Node list
Link list
node:1
node:2
??
l1
l2
l3
Source
Dest.
node
tp
Nodes
Node list
VendorB: node:1
VendorA: node:1
node:2
node:3
Node-connector list
tp1
tp2
tp3
???
?1
?2
?3
IP
???
Services
service list
Mpls-tp:1
service:2
Protection node list
Working node list
service attribute
node:1
node:2
node:3
Ing TP
Eg TP
RPC, notification
RPC, notification
RPC, notification
Vendor A Plugin
Vendor B EMS Plugin
Vendor C Plugin
SBI
Vendor A node 1
Vendor A node 2
Vendor B EMS
Vendor C node 1
Vendor C node 2

T-SDN controller workflow



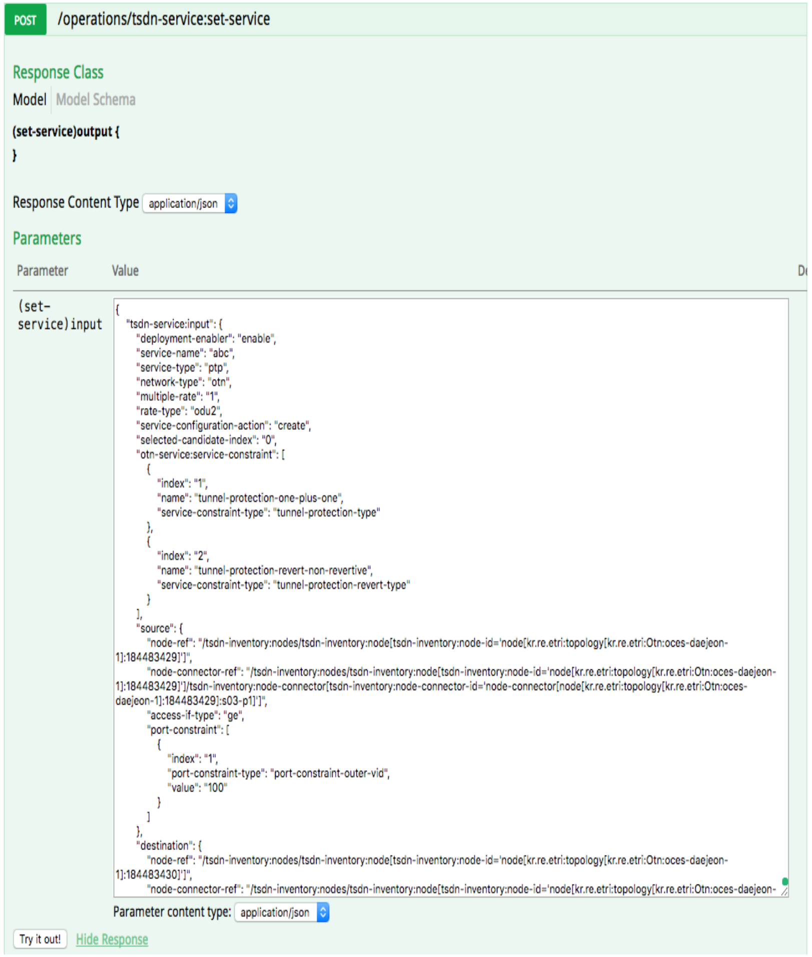
Implementation Details

- Modeling
  - 1. common idea
    - tsdn-prefix, node, node-connector, access-if, inventory, port-type
  - 2. difference
    - OTN specific – otn-prefix, tributary slots, ODU0-ODU4
    - MPLS-TP specifics – mplstp-prefix, mplstpif, psedowire
  - 3. Data Model
  - 4. RPCs – set-accessIf, set-mplsif, set-tunnel, set-tunnelXc
  - 5. Notification – tunnelUpdated, pwUpdated, mplsIfRemove



```
yang
    mpls-tp-connection.yang
    mpls-tp-general-types.yang
    mpls-tp-inventory.yang
    mpls-tp-provision.yang
    mpls-tp-service.yang
    mpls-tp-topology-discovery.yang
    mpls-tp-topology-inventory.yang
    otn-connection.yang
    otn-general-types.yang
    otn-inventory.yang
    otn-port.yang
    otn-provision.yang
    otn-service.yang
    otn-topology-discovery.yang
    otn-topology-inventory.yang
    ptn-port.yang
    transportcontroller.yang
    tsdn-access-if.yang
    tsdn-connection.yang
    tsdn-general-types.yang
    tsdn-inventory.yang
    tsdn-network-topology.yang
    tsdn-node.yang
    tsdn-pce.yang
    tsdn-port.yang
    tsdn-service.yang
    tsdn-topology-discovery.yang
    tsdn-tunnel.yang
    tsdn-tunnel-xc.yang
```

Implementation Details



POST /operations/tsdn-service:set-service

Response Class

Model | Model Schema

(set-service)output {
}

Response Content Type: application/json

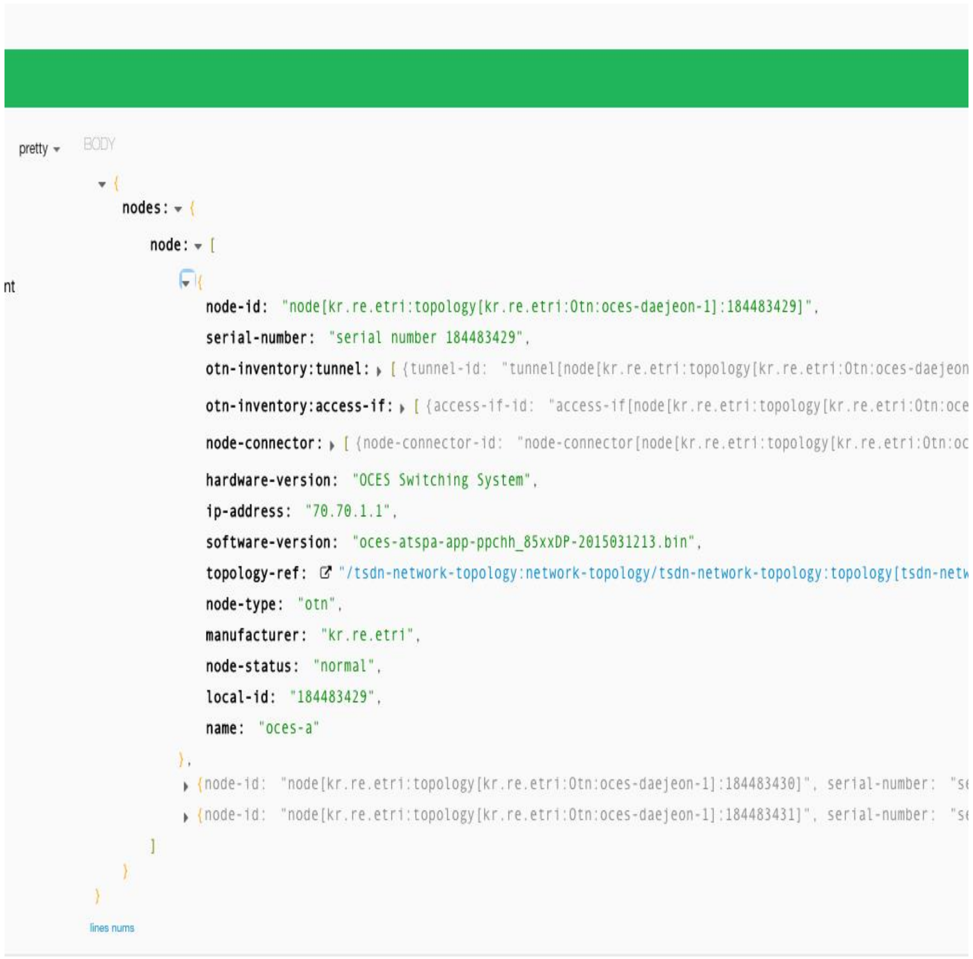
Parameters

Parameter	Value
(set-service)input	(see below)

```
{
  "tsdn-service:input": {
    "deployment-enabler": "enable",
    "service-name": "abc",
    "service-type": "ptp",
    "network-type": "otn",
    "multiple-rate": "1",
    "rate-type": "odu2",
    "service-configuration-action": "create",
    "selected-candidate-index": "0",
    "otn-service:service-constraint": [
      {
        "index": "1",
        "name": "tunnel-protection-one-plus-one",
        "service-constraint-type": "tunnel-protection-type"
      },
      {
        "index": "2",
        "name": "tunnel-protection-revert-non-revertive",
        "service-constraint-type": "tunnel-protection-revert-type"
      }
    ],
    "source": {
      "node-ref": "/tsdn-inventory:nodes/tsdn-inventory:node[tsdn-inventory:node-id='node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon-
1]:184483429]']",
      "node-connector-ref": "/tsdn-inventory:nodes/tsdn-inventory:node[tsdn-inventory:node-id='node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon-
1]:184483429]']/tsdn-inventory:node-connector[tsdn-inventory:node-connector-id='node-connector[node[kr.re.etri:topology[kr.re.etri:Otn:oces-
daejeon-1]:184483429]:s03-p1]']",
      "access-if-type": "ge",
      "port-constraint": [
        {
          "index": "1",
          "port-constraint-type": "port-constraint-outer-vid",
          "value": "100"
        }
      ]
    },
    "destination": {
      "node-ref": "/tsdn-inventory:nodes/tsdn-inventory:node[tsdn-inventory:node-id='node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon-
1]:184483430]']",
      "node-connector-ref": "/tsdn-inventory:nodes/tsdn-inventory:node[tsdn-inventory:node-id='node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon-
```

Parameter content type: application/json

<Service Input for ServiceManager>

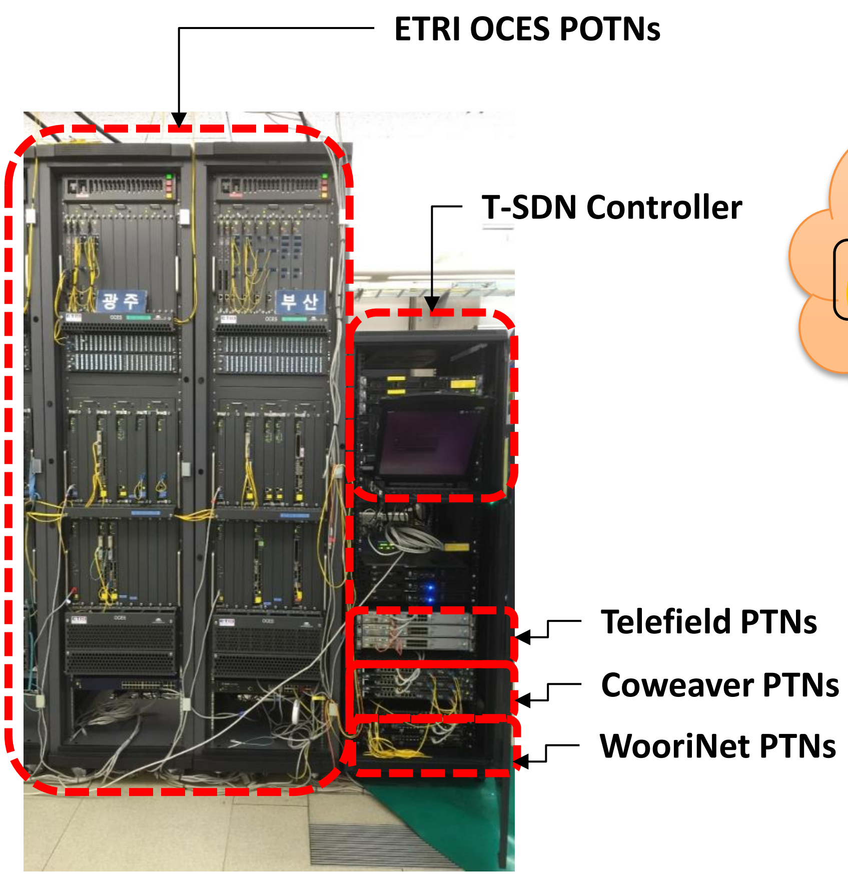


pretty BODY

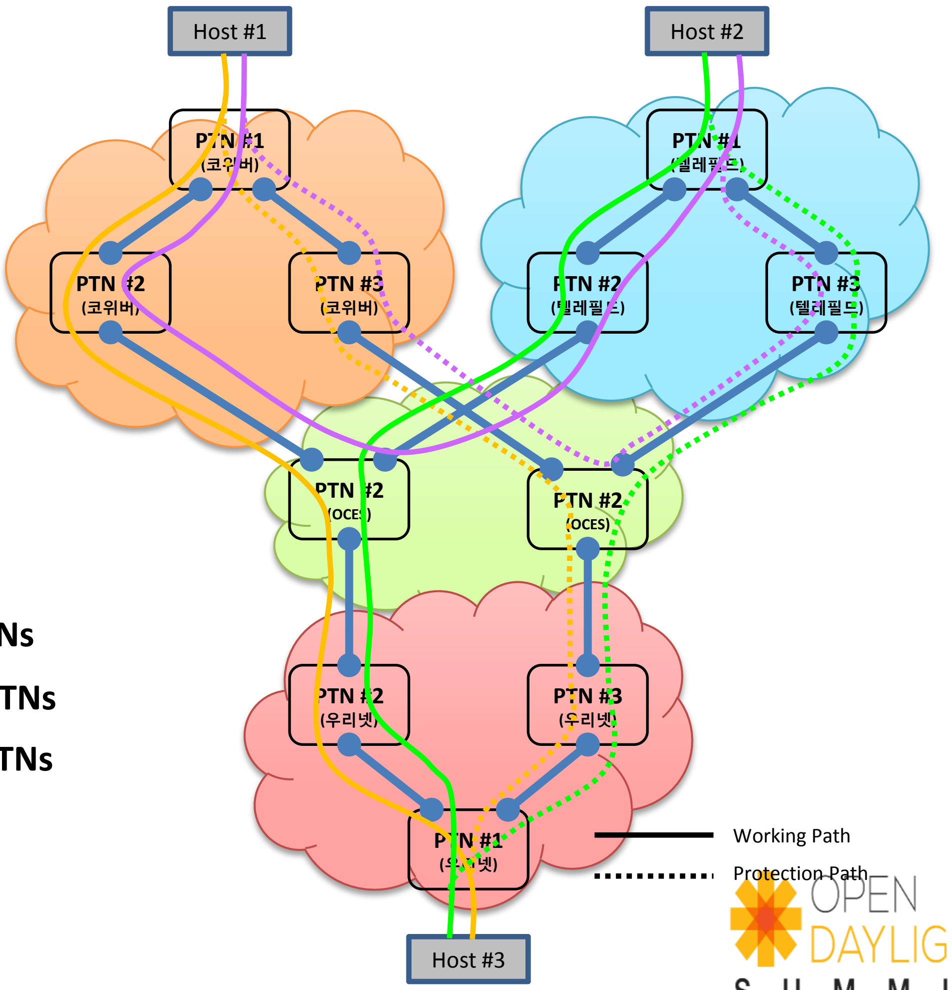
```
{
  nodes: {
    node: [
      {
        node-id: "node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon-1]:184483429]",
        serial-number: "serial number 184483429",
        otn-inventory:tunnel: [ {tunnel-id: "tunnel[node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon
        otn-inventory:access-if: [ {access-if-id: "access-if[node[kr.re.etri:topology[kr.re.etri:Otn:oce
        node-connector: [ {node-connector-id: "node-connector[node[kr.re.etri:topology[kr.re.etri:Otn:oc
        hardware-version: "OCES Switching System",
        ip-address: "70.70.1.1",
        software-version: "oces-atspa-app-ppchh_85xxDP-2015031213.bin",
        topology-ref: "/tsdn-network-topology:network-topology/tsdn-network-topology:topology[tsdn-netw
        node-type: "otn",
        manufacturer: "kr.re.etri",
        node-status: "normal",
        local-id: "184483429",
        name: "oces-a"
      },
      {node-id: "node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon-1]:184483430]", serial-number: "se
      {node-id: "node[kr.re.etri:topology[kr.re.etri:Otn:oces-daejeon-1]:184483431]", serial-number: "se
    ]
  }
}
```

<Result with APIDocs>

Testbed Information

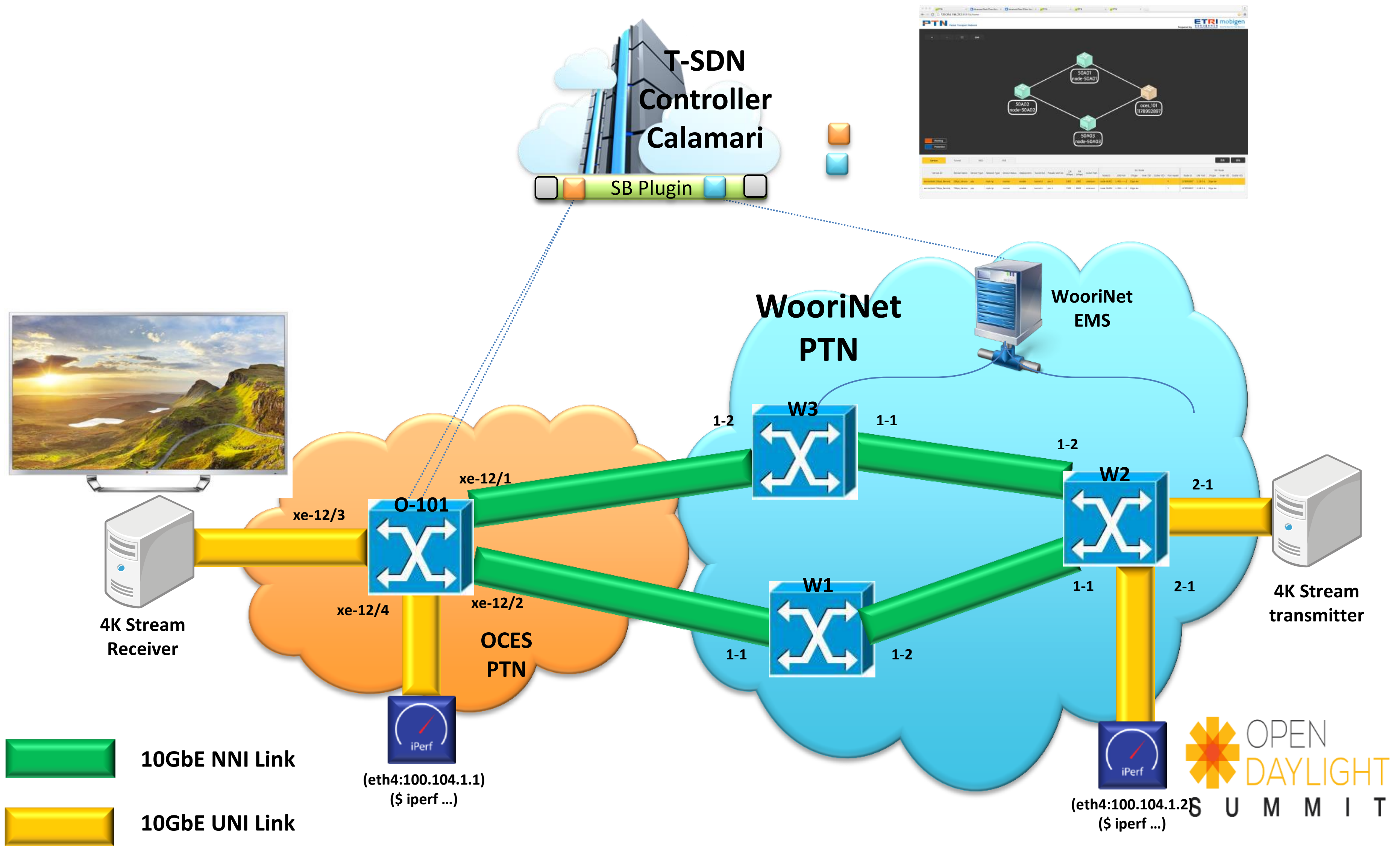


Transport Devices (ETRI Build #7 3rd floor)



OPEN DAYLIGHT SUMMIT

Accomplishments



<Calamari TSDN Controller Testbed in 2015: multi-vendor devices>

환경설정

- Ubuntu 14.04 LTS 추천
- Java 1.8 설치
- Maven 3.31 이상
- 쉘설치
  - https://github.com/t-sdn/
- OpenDaylight Gerrit 계정 생성 및 repository 설정
- Eclipse 설치
  - https://wiki.opendaylight.org/view/GettingStarted:_Eclipse
- 실습 Google docs
  - http://bit.ly/1HG1QcO

Useful ODL sites

- MD-SAL API
- https://developer.cisco.com/site/openSDN/documents/java-extension-services-api/sal-binding-api/#
- twitts
- https://twitter.com/OpenDaylightSDN
- gerrit
- https://git.opendaylight.org/gerrit/
- ask
- http://ask.opendaylight.org
- Bugzila
- https://bugs.opendaylight.org/buglist.cgi?bug_status=__open__&content=&no_redirect=1&order=Importance&product=&query_format=specific
- Wiki
- https://wiki.opendaylight.org/view/Main_Page
- ODLUG
- http://www.meetup.com/OpenDaylight-Korean-User-Group/
- IRC
- http://webchat.freenode.net/?channels=opendaylight
- Bitly
- https://bitly.com/

Reference

- http://www.slideshare.net/sdnhub/opendaylight-app-development-tutorial
- https://wiki.opendaylight.org/view/Main_Page
- Model Driven Service Abstraction Layer (MD-SAL) by NEC