ネットワーク仮想化基盤における仮想ネットワーク管理制御技術

山田　一久
NTT未来ねっと研究所

2012.07.20

Outline

- ネットワーク仮想化基盤の概要
- ネットワーク仮想化基盤の管理モデル
- 仮想ネットワーク制御管理
- 仮想ネットワークとトランスポートネットワークの連携制御管理
- まとめ

Outline

- ネットワーク仮想化基盤の概要
- ネットワーク仮想化基盤の管理モデル
- 仮想ネットワーク制御管理
- 仮想ネットワークとトランスポートネットワークの連携制御管理
- まとめ

ネットワーク仮想化基盤技術

NTT

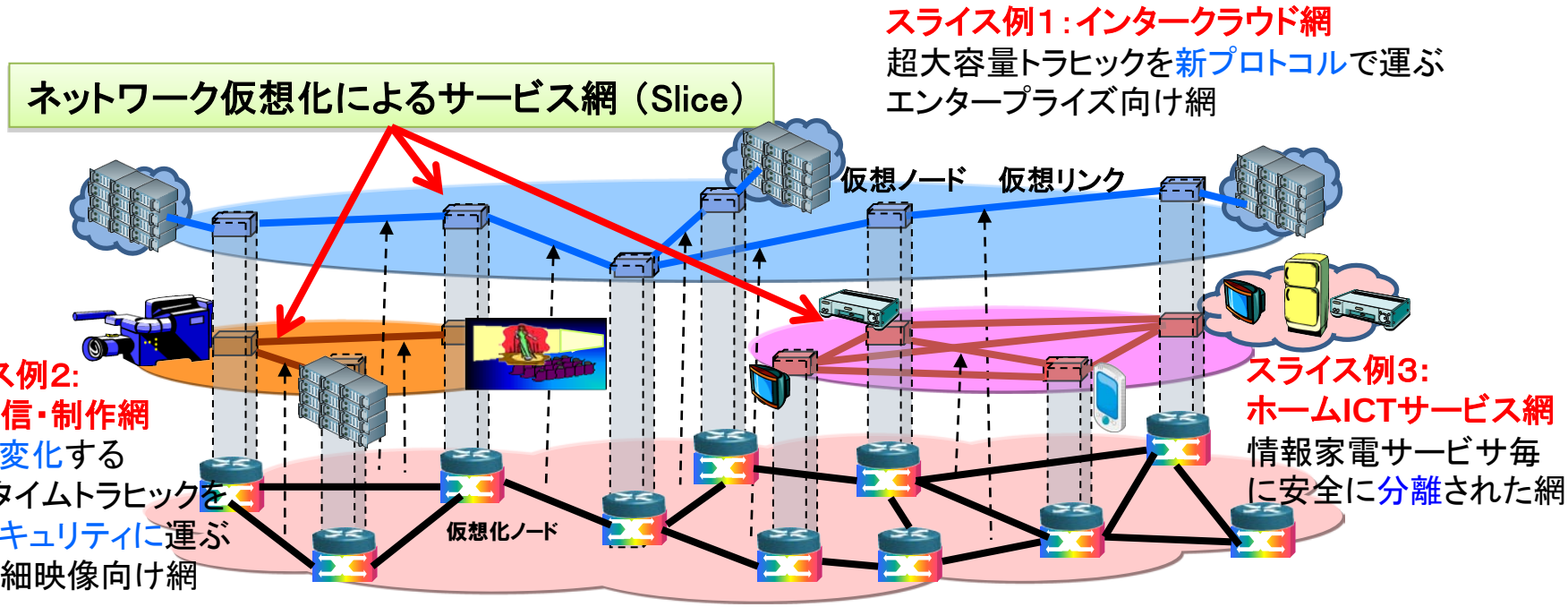
アプリケーションごとに最適なサービスの創造

機能の異なるプライベートサービス網（Slice）を迅速・動的・安価に実現

※ 現状のIP-VPNは、指定された複数の拠点間を固定プロトコル(Ether/IP)で接続

利点

- 用途に応じて「品質、セキュリティ、プロトコル」などの異なるプライベート網を提供
- 単一のインフラ上に実現する事による経済化



ネットワーク仮想化基盤網が実現する将来NW像

NTT

アプリケーションごとに最適なネットワークサービスの創造

スライス1 スライス2 スライスN

パケットキャッシュNW | センサー集約・分配NW | クラウドアクセスNW | ID/Loc NW | 動画配信NW | 放送配信NW | レガシー（従来）NW | 新世代NW

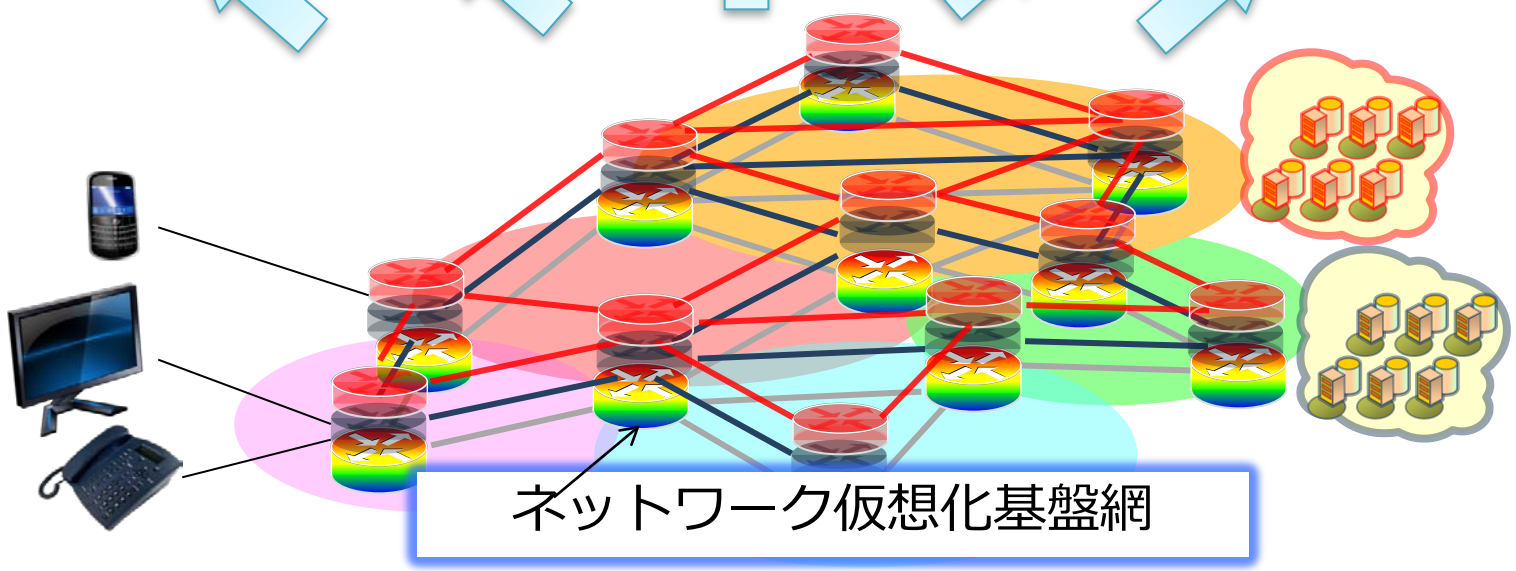
プログラマビリティとスケーラビリティの両立

資源の組合せによるアプリ要求の実現

スライス間の確実な分離、干渉抑止

測定との連携による資源利用最適化

スライスで分離されたコンピュータ・ネットワーク資源

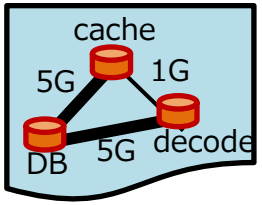


仮想化ノードとマネジメントシステム

NTT

スライス開発者（Developer）

スライス設計図
＝プライベート網の要求条件



仮想化NWマネジメントシステム
資源管理と探索・割当て、装置への制御指示を行う NMS

課題③: スライス要求条件表現法、資源の組合せ最適化

課題④: 精緻な測定との連携による資源利用最適化

仮想化ノード

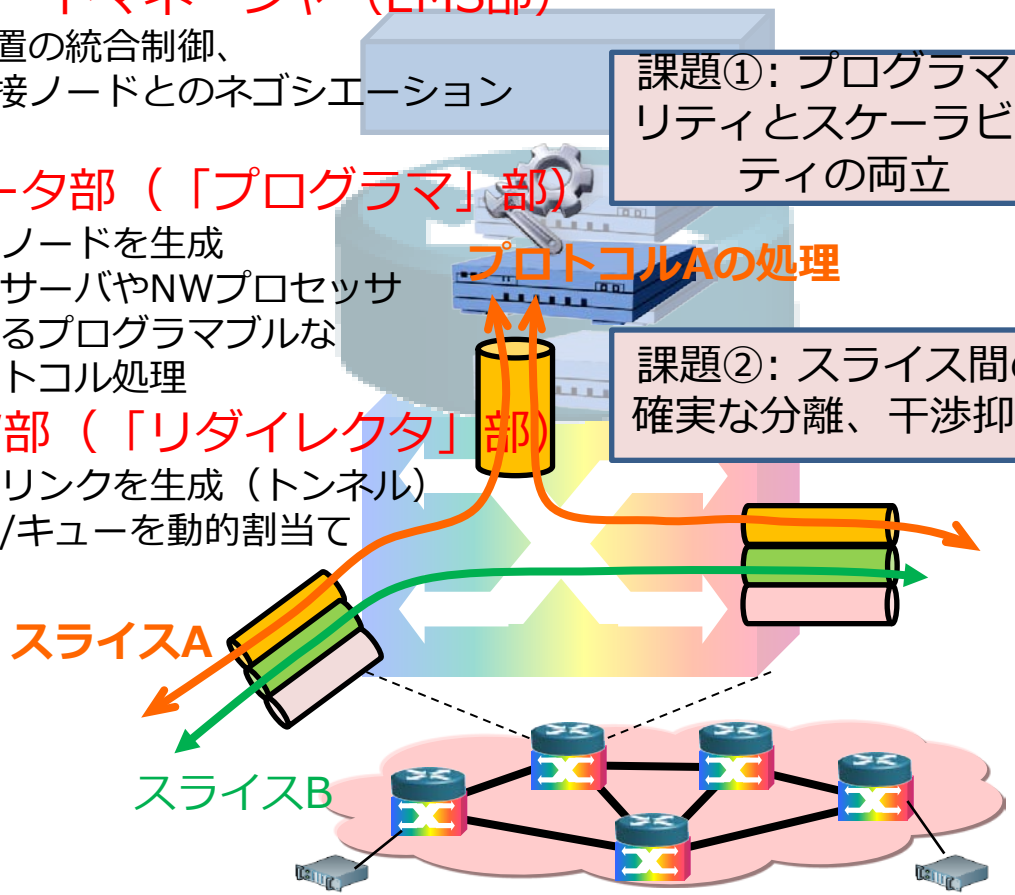
ノードマネージャ（EMS部）
装置の統合制御、
隣接ノードとのネゴシエーション

課題①: プログラマビリティとスケーラビリティの両立

ルータ部（「プログラマ」部）
仮想ノードを生成
仮想サーバやNWプロセッサによるプログラマブルなプロトコル処理

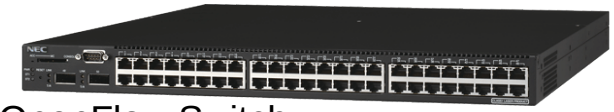
SW部（「リダイレクタ」部）
仮想リンクを生成（トンネル）
帯域/キューを動的割当て

課題②: スライス間の確実な分離、干渉抑止



仮想化ノードの構成

Programmer Part (仮想ノードの生成)
(IA Serverx4 +ATCAx2+OpenFlow SWx2)



OpenFlow Switch

vNM (仮想化ノードManager部, EMS相当)

* vNM：virtualization Node Manager
* EMS：Element Management System

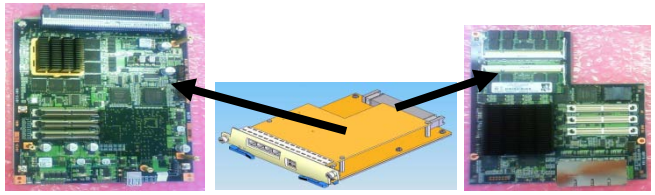
Redirector Part (仮想リンクの生成)
(AX6700+SMCx2)

AX6708S

Service Module Card

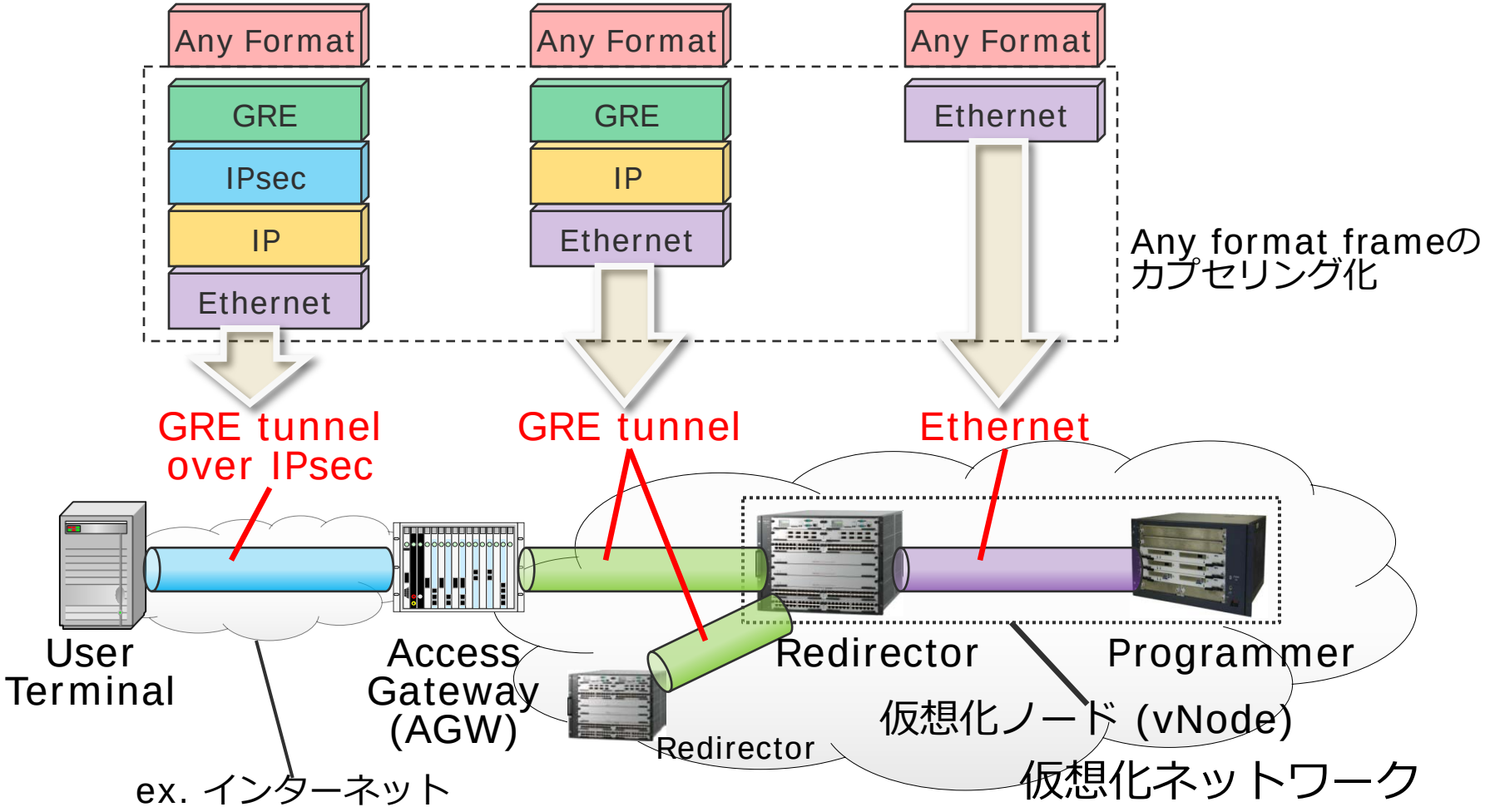
下側

上側



データプレーン構成

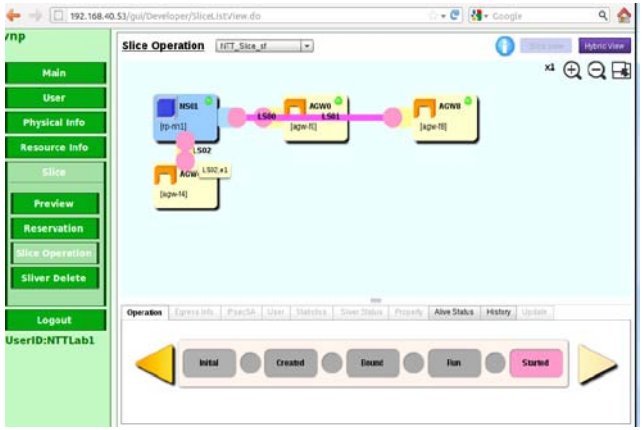
- IPsec, GREによるカプセリング化によって任意のPacket/Frameを転送可能



テストベッド展開による有効性の確認

NTT

- 要素技術を実証しながら、将来ＮＷでのサービス検討を行うために、ノードベンダとの連携、実証環境の構築を推進
  - JGN-X に展開しテストベッドとして利用開始
  - 仮想化ノード装置とマネジメントシステムの高機能化を検討



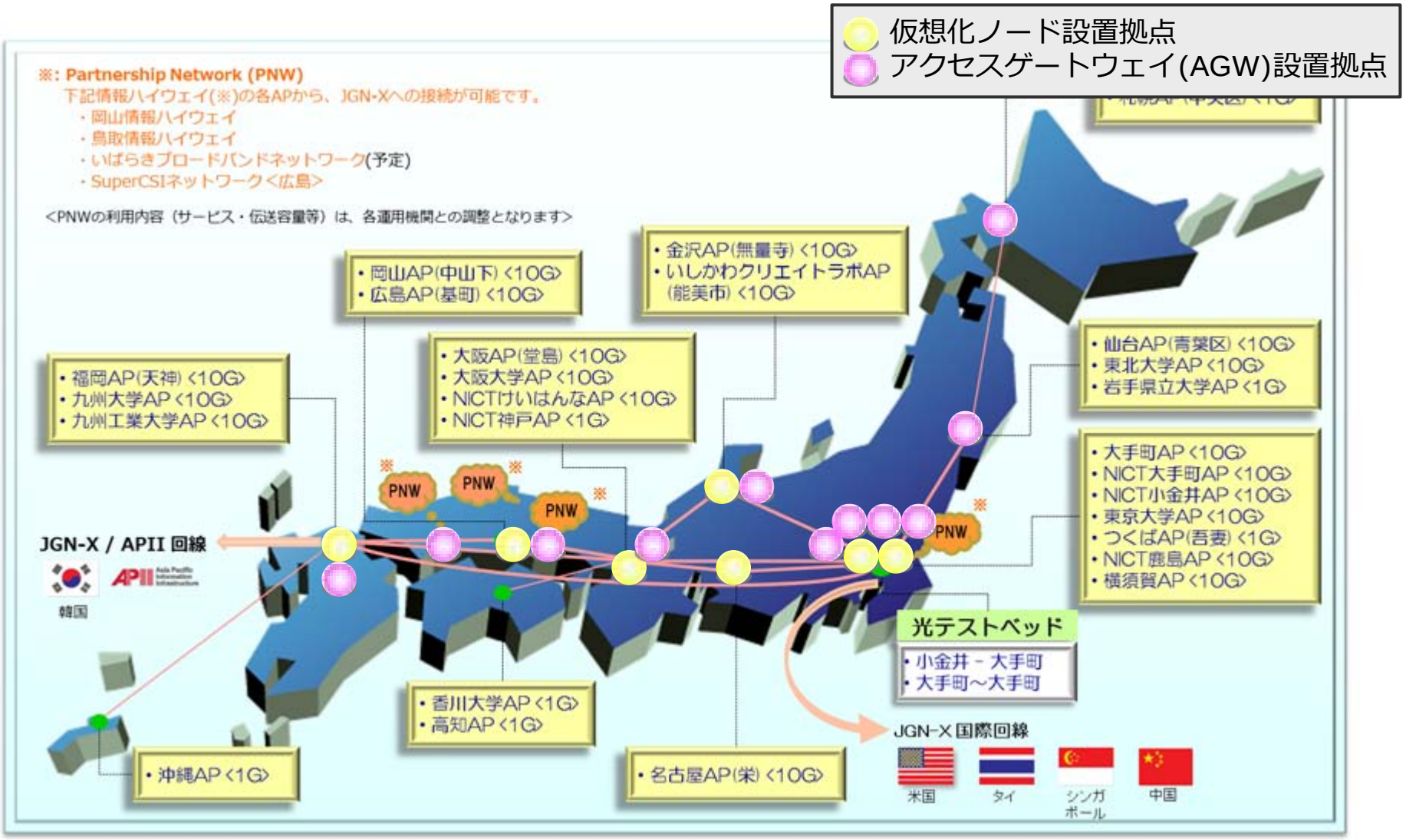
仮想化NWマネジメントシステム

仮想化ノード装置プロトタイプ

JGN-Xへの展開状況

NTT

➢ 仮想化ノード７ノード、AGW１１ノードをJGN-X上へ展開



http://www.jgn.nict.go.jp/ja/info/images/jgn-x-accesspoints.png

Deeply Programmable Network (DPN)

NTT

- **Flow-Level Programmability**
  - 経路制御
  - アクセス制御
  - ネットワーク管理
- **Packet-Level Programmability**
  - 従来プロトコル上パケット処理
    - キャッシュ
    - トランスコーディング
    - ネットワークコーディング等
  - 新プロトコルの解釈
    - IPvN (N>6)
    - CCN
    - Content Switching/Routing

我々の目指す
ＮＷ仮想化技術にて
全て可能

Deeply
Programmable
Network

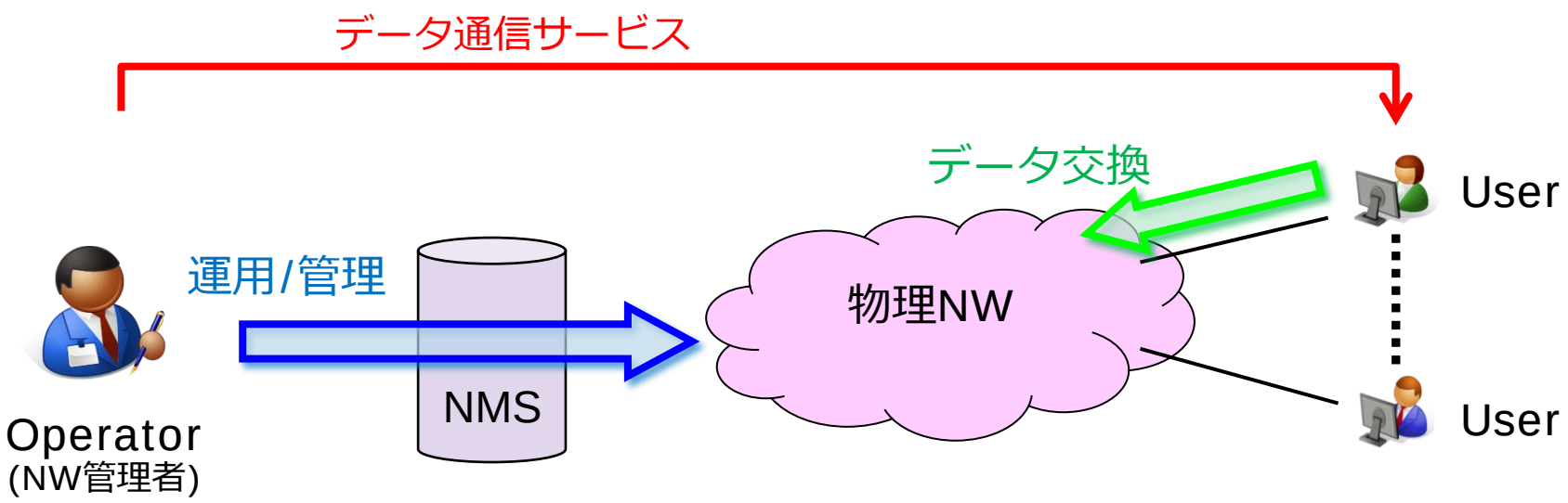
Deeper
Programmability

Outline

- ネットワーク仮想化基盤の概要
- **ネットワーク仮想化基盤の管理モデル**
- 仮想ネットワーク制御管理
- 仮想ネットワークとトランスポートネットワークの連携制御管理
- まとめ

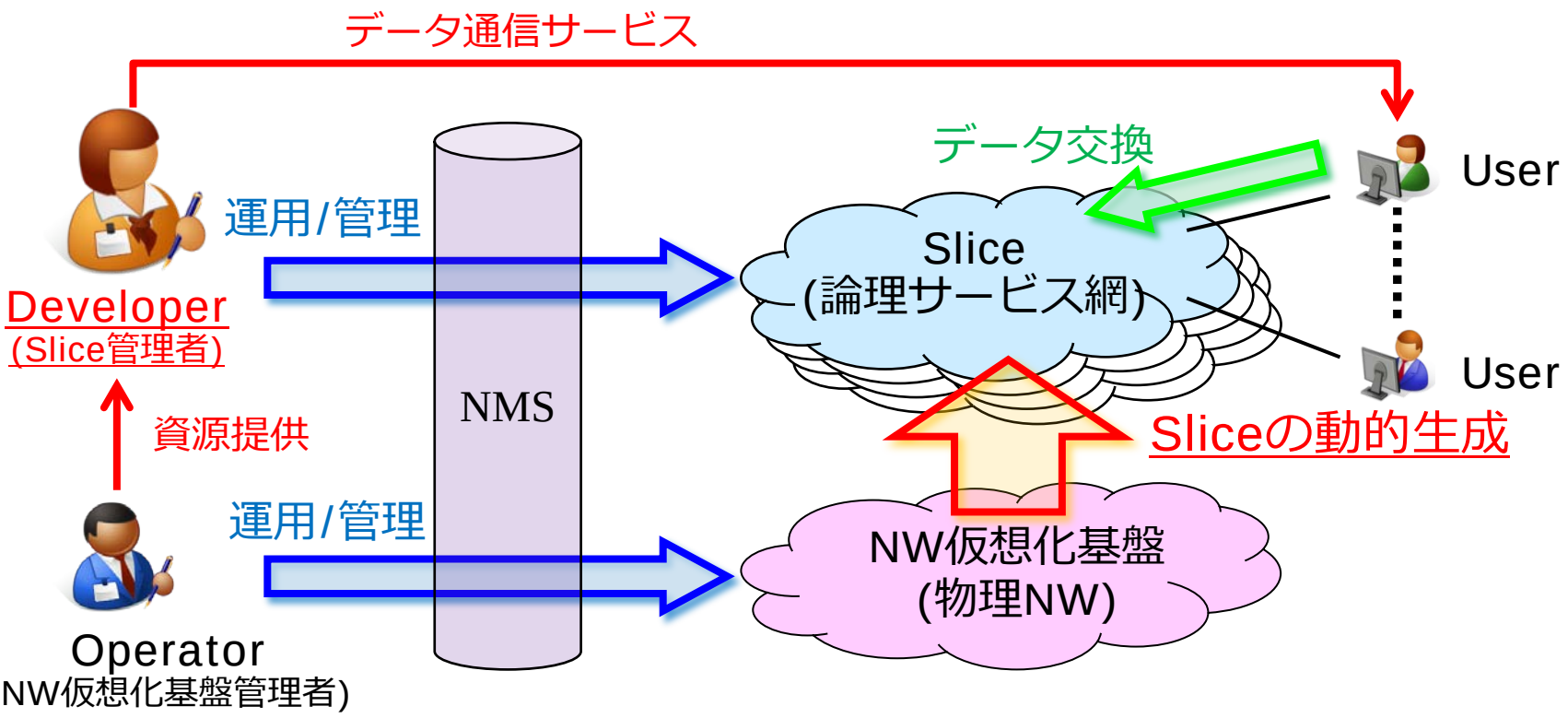
従来NWサービスにおける役柄

- 二つの役柄
  - Operator：物理NWの管理者(通信キャリア)
  - User：NWを用いてデータ交換を行う者



NW仮想化基盤における役柄

- 三つの役柄
  - Operator：NW仮想化基盤の管理者
  - Developer：Sliceの管理者, サービス提供者
  - User：Sliceを用いてデータ交換を行う者



NWアーキテクチャモデル

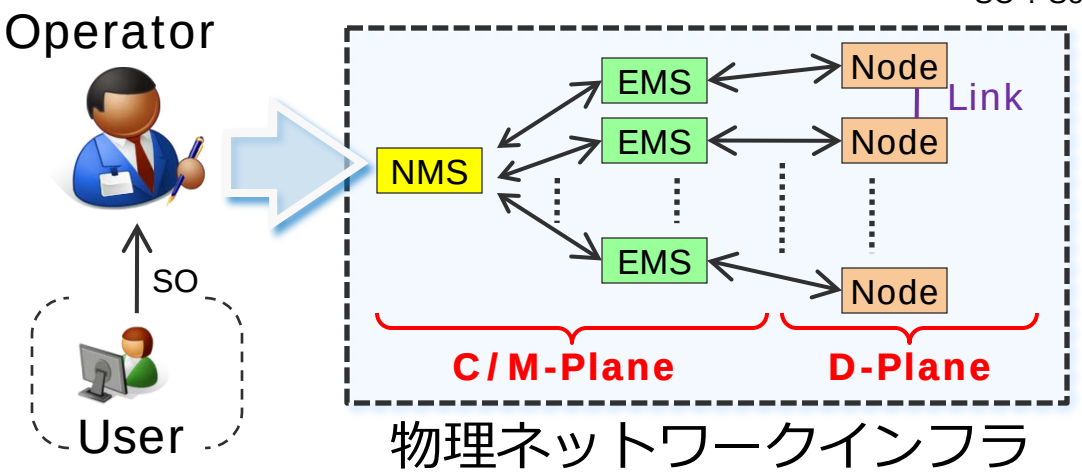
- 従来のネットワークアーキテクチャで用いられる3-Plane
  - D(data)-Plane, C(control)-Plane, M(management)-Plane

- 新たにDeveloperが存在
  - Developerが自身のスライスを運用するという形態

を考慮し、新たなPlaneの設定が必要

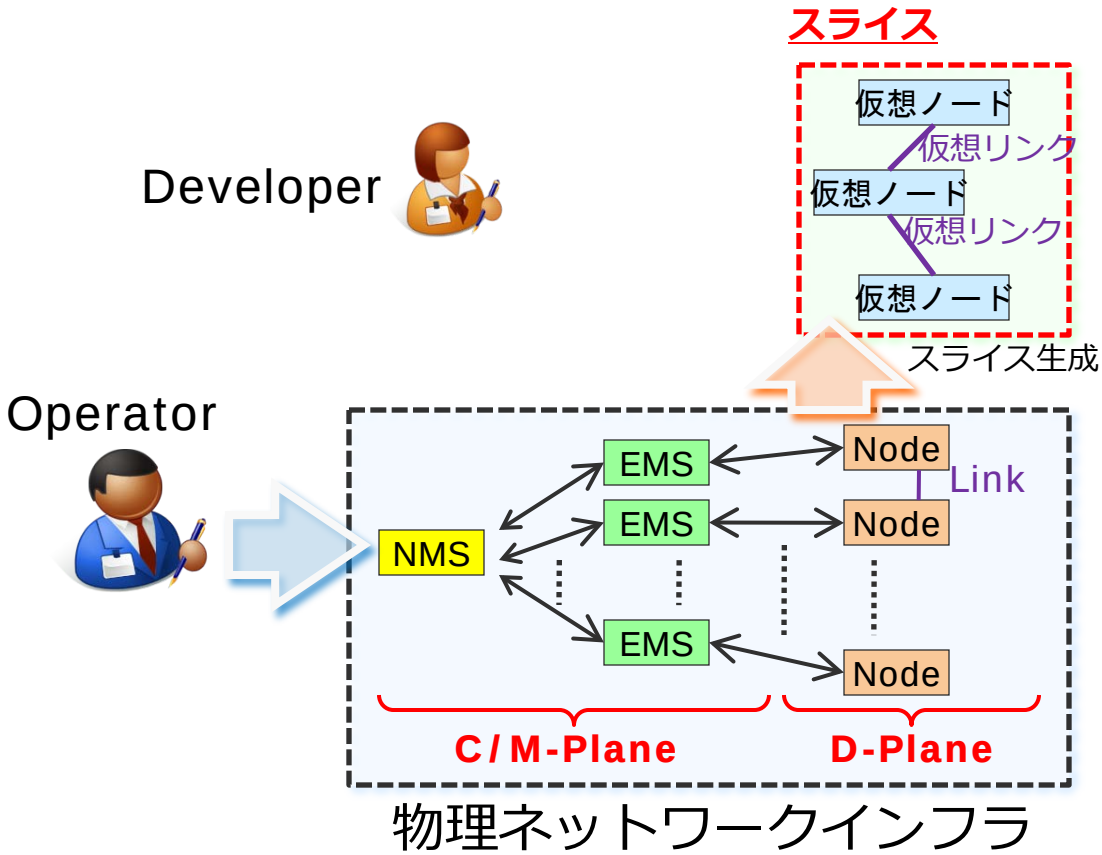
従来のネットワークアーキテクチャ

* NMS：Network Management System
* EMS：Element Management System
* SO：Service Order



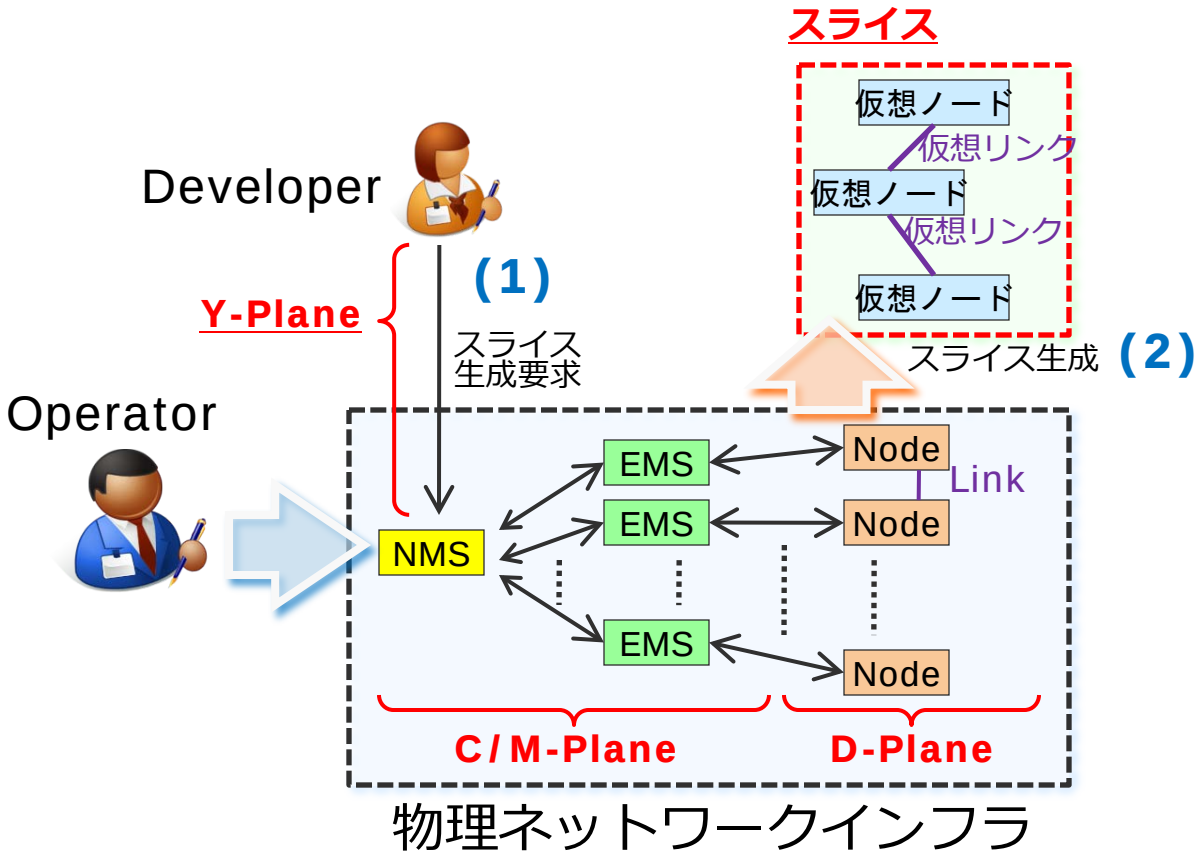
新たなPlane:”スライス”

- “スライス”：論理サービスネットワーク
  - Developerから見た、一つの独立したネットワーク
  - DeveloperのためのD-Plane



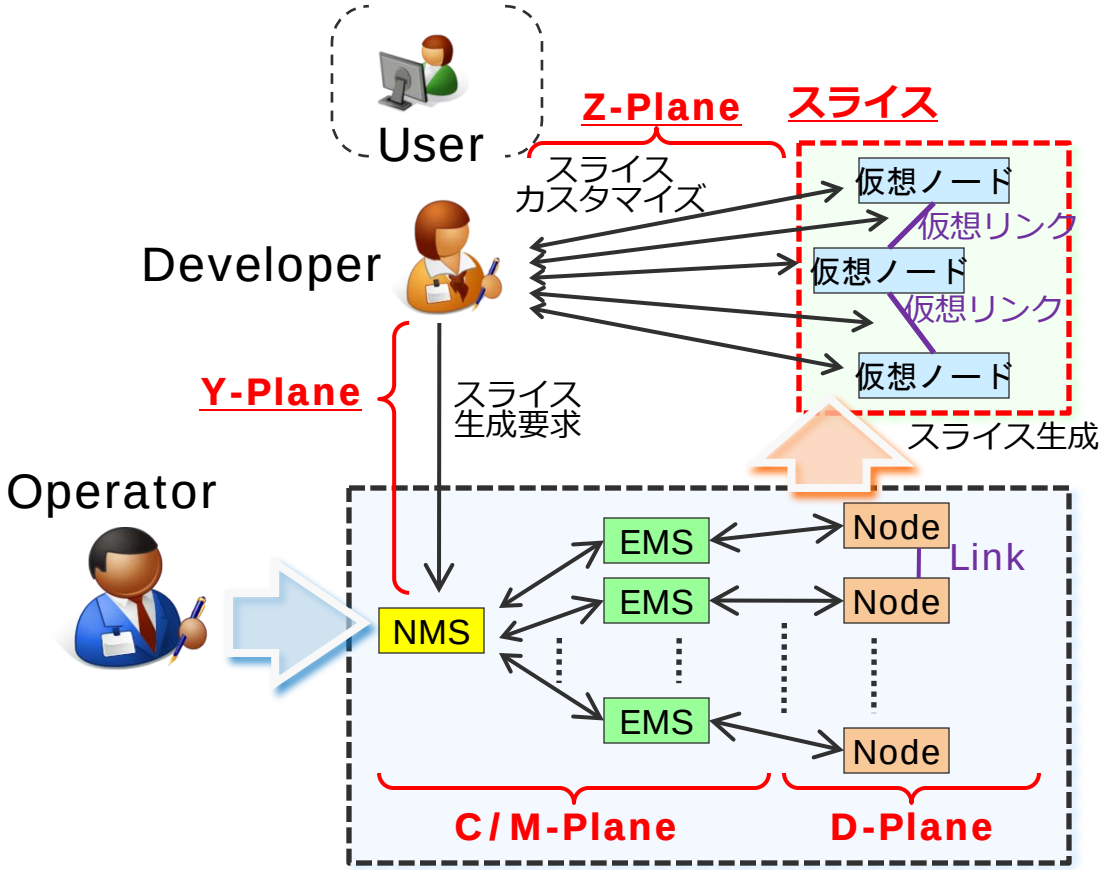
新たなPlane:”Y-Plane”

- “Y-Plane”：スライス生成を要求するインタフェース
- スライス利用手順
  - **(1)** DeveloperがNMSにスライス生成を要求
  - **(2)** NMSがC/M-Planeを通じてインフラを設定し、生成



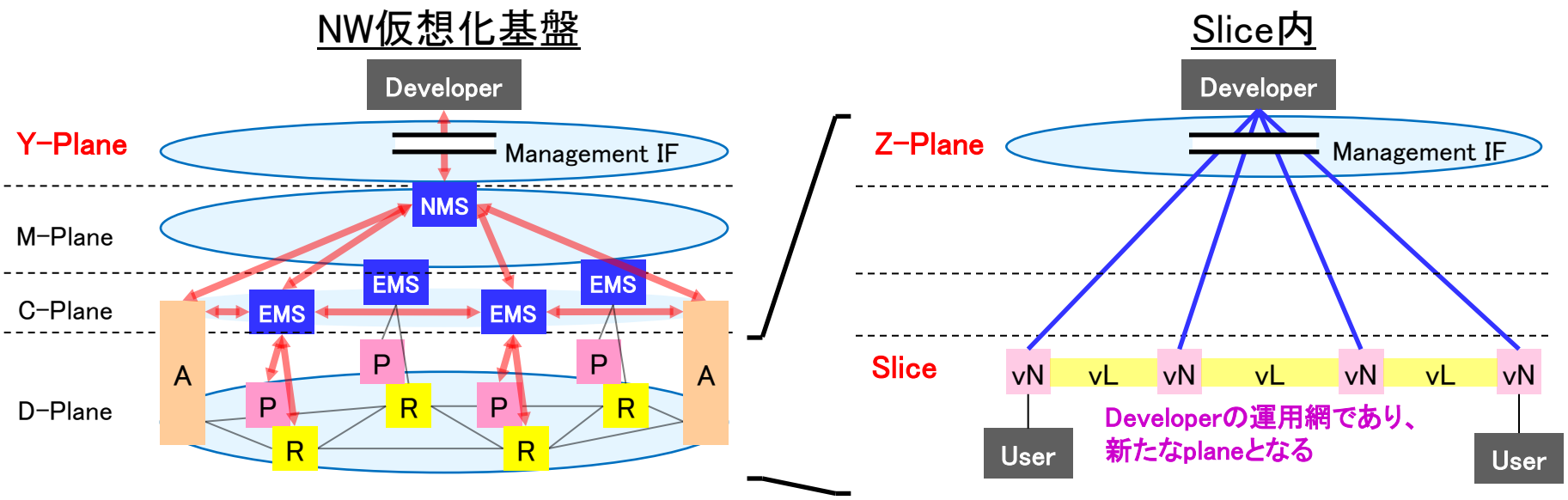
新たなPlane:”Z-Plane”

- “Z-Plane” : スライスをカスタマイズするインタフェース
  - スライス構成要素の特性などを変更するため
  - スライスに対するDeveloperにとってのC/M-Plane



6-Plane 管理モデル

- NW仮想化の管理モデルに基づき、6つのPlaneを定義
  - D/C/M-Plane：従来のD/C/M-Planeと同じ位置づけ
  - Y-Plane：DeveloperがNMSにSlice生成を要求する Plane
  - Z-Plane：Developerが仮想ノードの設定を行う Plane
  - Slice：Sliceのデータ送受信用 Plane



A: Access Gateway, P: Programmer, R: Redirector

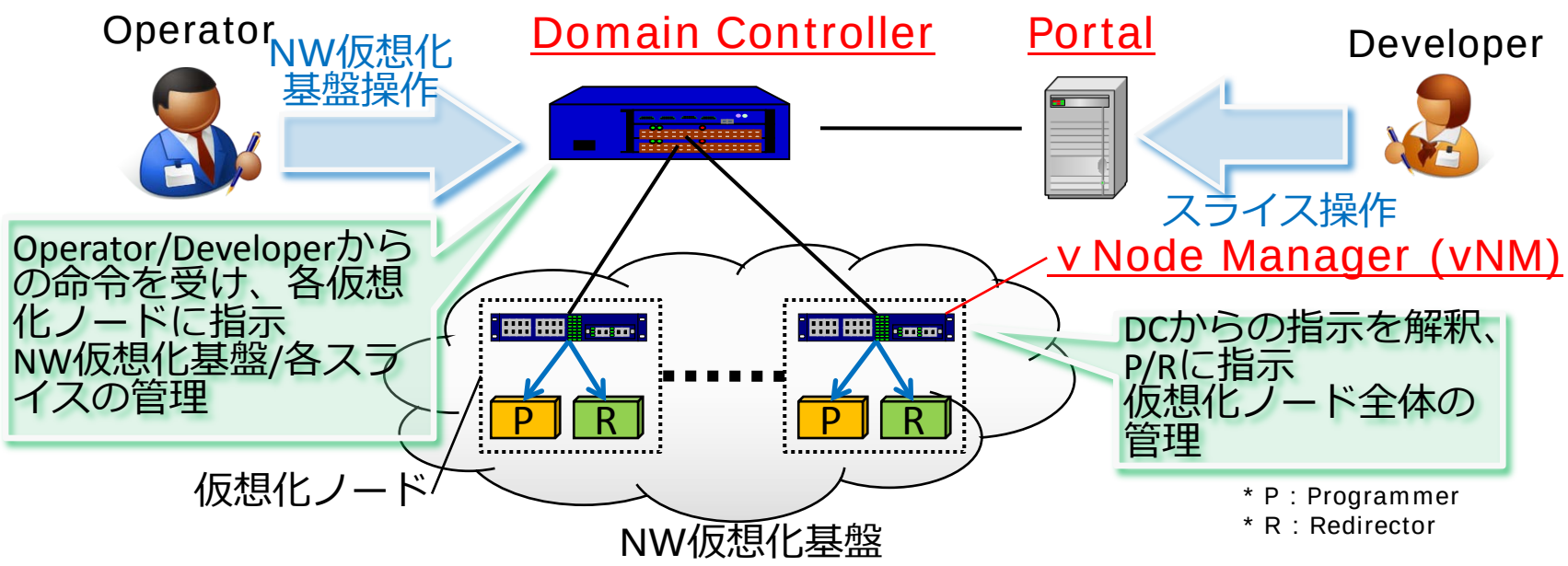
vN : virtual Node (仮想ノード), vL : virtual Link (仮想リンク)

Outline

- ネットワーク仮想化基盤の概要
- ネットワーク仮想化基盤の管理モデル
- **仮想ネットワーク制御管理**
- 仮想ネットワークとトランスポートネットワークの連携制御管理
- まとめ

仮想化NWマネジメントシステム

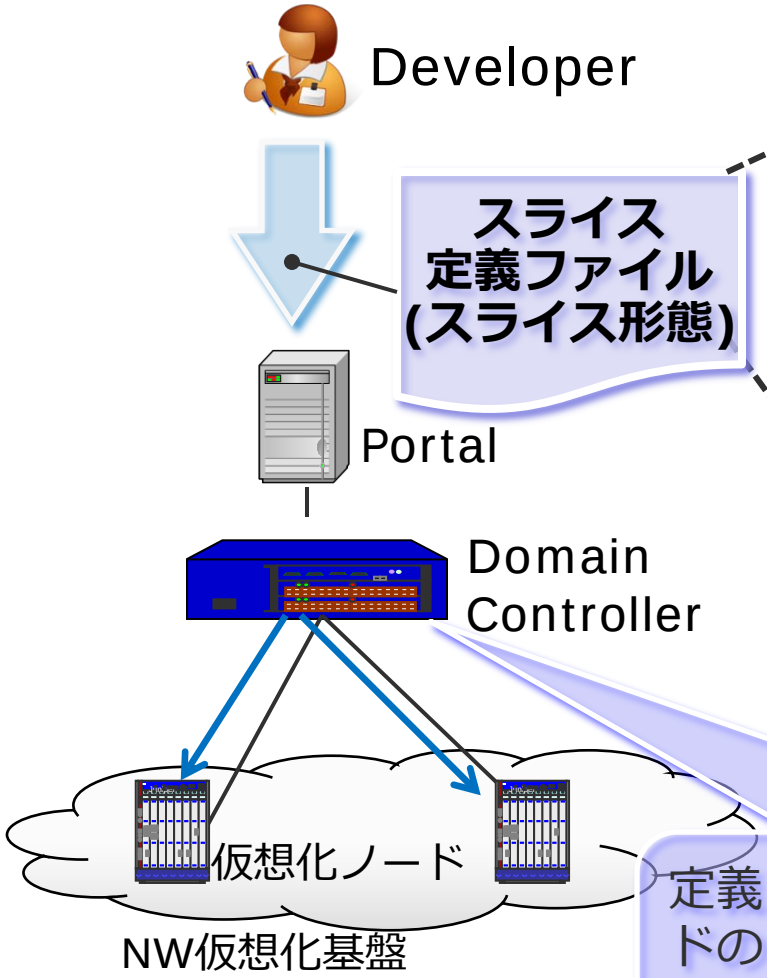
- Domain Controller (DC)
  - NW仮想化基盤/スライスの管理/制御を行う
- Portal
  - Developerへスライスにアクセスするインタフェースを提供
- vNode Manager (vNM)
  - DCからの指示を受け、仮想化ノードを管理/制御する



仮想ネットワーク（スライス）の構築

NTT

所望スライスの投入 ＝ スライスのテキスト定義




```
<?xml version="1.0" encoding="utf-8"?>
<slice-design>
  <slicespec name="NICT_Slice_001">
    <sliverdef>
      <!-- Link Slivers -->
      <linkSlivers>
        <!-- LinkSliver LS01 -->
        <linkSliver name="LS01" type="link" subtype="GRE">
          <vports>
            <vport name="e1" />
            <vport name="e2" />
          </vports>
        </linkSliver>
        <!-- LinkSliver LS02 -->
        <linkSliver name="LS02" type="link" subtype="GRE">
          <vports>
            <vport name="e1" />
            <vport name="e2" />
          </vports>
        </linkSliver>
        <!-- LinkSliver LS03 -->
        <linkSliver name="LS03" type="link" subtype="GRE">
          <vports>
            <vport name="e1" />
            <vport name="e2" />
          </vports>
        </linkSliver>
      </linkSlivers>
```

仮想ノード/仮想リンクの接続
物理上の位置、ゲートウェイの配置
利用物理資源量、などが含まれる

定義ファイルに従い、複数の仮想化ノードのリソースを割当て、仮想化ノードを制御し、スライスを構築

スライス定義ファイルの例

NTT

```
<?xml version="1.0" encoding="UTF-8"?>
<slice-design>
  <slicespec name="NTT_Slice_N">
    <sliverdef>
      <linkSlivers>
        <linkSliver name="LS00" type="link" subtype="GRE">
          <vports>
            <vport name="e1"/>
            <vport name="e2"/>
          </vports>
        </linkSliver>
        <linkSliver name="LS08" type="link" subtype="GRE">
          <vports>
            <vport name="e1"/>
            <vport name="e2"/>
          </vports>
        </linkSliver>
      </linkSlivers>
      <nodeSlivers>
        <nodeSliver name="NS00" type="prog">
          <vports>
            <vport name="vp5"/>
            <vport name="vp8"/>
          </vports>
          <hierarchy>
            <sliverdef>
              <nodeSlivers>
                <nodeSliver name="SP00">
                  <vports>
                    <vport name="vip5" />
                    <vport name="vip8" />
                  </vports>
                  <instance type="SlowPath_VM" subtype="KVM">
                    <resources>
                      <resource keyword="cpu" value="1" />
                      <resource keyword="arch" value="x86_64" />
                      <resource keyword="memory" value="2048" />
                    </resources>
                    <params>
                      <param keyword="bootImage"
value="http:/[redacted]10Server32.img" />
                    </params>
                  </instance>
                </nodeSliver>
              </nodeSlivers>
              <linkSlivers>
                <linkSliver name="ln05" type="link">
                  <vports>
                    <vport name="e1" />
                    <vport name="e2" />
                  </vports>
                </linkSliver>
                <linkSliver name="ln08" type="link">
                  <vports>
                    <vport name="e1" />
                    <vport name="e2" />
                  </vports>
                </linkSliver>
              </linkSlivers>
            </sliverdef>
            <structure>
              <bind name="wn51">
                <vport sliverID="__TOP__" portname="vp5" />
                <vport sliverID="ln05" portname="e1" />
              </bind>
              <bind name="wn52">
                <vport sliverID="SP00" portname="vip5" />
                <vport sliverID="ln05" portname="e2" />
              </bind>
              <bind name="wn81">
                <vport sliverID="__TOP__" portname="vp8" />
                <vport sliverID="ln08" portname="e1" />
              </bind>
              <bind name="wn82">
                <vport sliverID="SP00" portname="vip8" />
                <vport sliverID="ln08" portname="e2" />
              </bind>
            </structure>
            <params>
              <param keyword="authKey" value="ssh-dss
AAAAB3NzaC1kc3MAAACBAN0YZ7QybAoiruCPF8UOxaYiy9UfuUO2mZaGnU4WiOhjJ0mKN
I81RT[redacted]Anu9f
7buPc[redacted]1j8SqA
4uqHl[redacted]9lapOy
L9FPZ[redacted]JkooYn
JLUbS[redacted]8gdBhy
rRhQc[redacted]CrZ5xr
c2BNg[redacted]fddIUp
f8XW2[redacted]
            </params>
          </hierarchy>
        </nodeSliver>
      </nodeSliver>
      <nodeSliver name="AG00" type="agw">
        <vports>
          <vport name="vp1"/>
        </vports>
      </nodeSliver>
      <nodeSliver name="AG08" type="agw">
        <vports>
          <vport name="vp1"/>
        </vports>
      </nodeSliver>
    </nodeSlivers>
  </sliverdef>
  <structure>
    <bind name="w31">
      <vport slivername="LS00" portname="e1"/>
      <vport slivername="NS00" portname="vp5"/>
    </bind>
    <bind name="w32">
      <vport slivername="LS00" portname="e2"/>
      <vport slivername="AG00" portname="vp1"/>
    </bind>
    <bind name="w51">
      <vport slivername="LS08" portname="e1"/>
      <vport slivername="NS00" portname="vp8"/>
    </bind>
    <bind name="w52">
      <vport slivername="LS08" portname="e2"/>
      <vport slivername="AG08" portname="vp1"/>
    </bind>
  </structure>
  </slicespec>
  <mapping slice="NTT_Slice_N" vnetwork="JGNtestbed">
    <amap node="NS00" vnode="rp-nh0"/>
    <amap node="AG00" vnode="agw-f0"/>
    <amap node="AG08" vnode="agw-f8"/>
  </mapping>
</slice-design>
```

仮想リンク定義

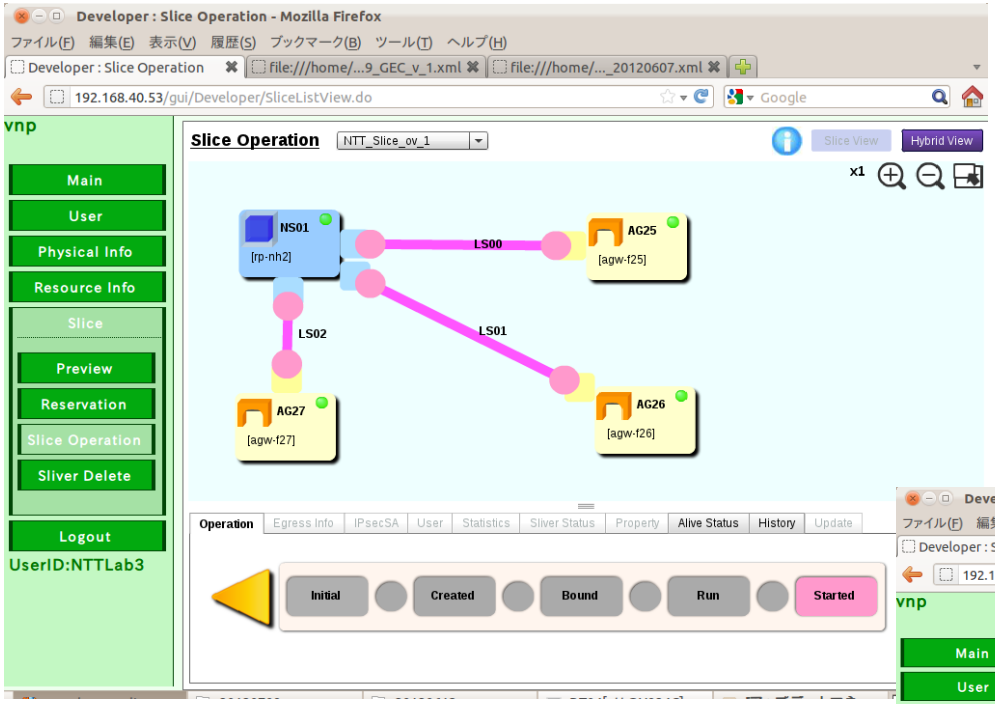
仮想ノード定義

仮想ゲートウェイ定義

仮想リンクと仮想ノードの接続定義

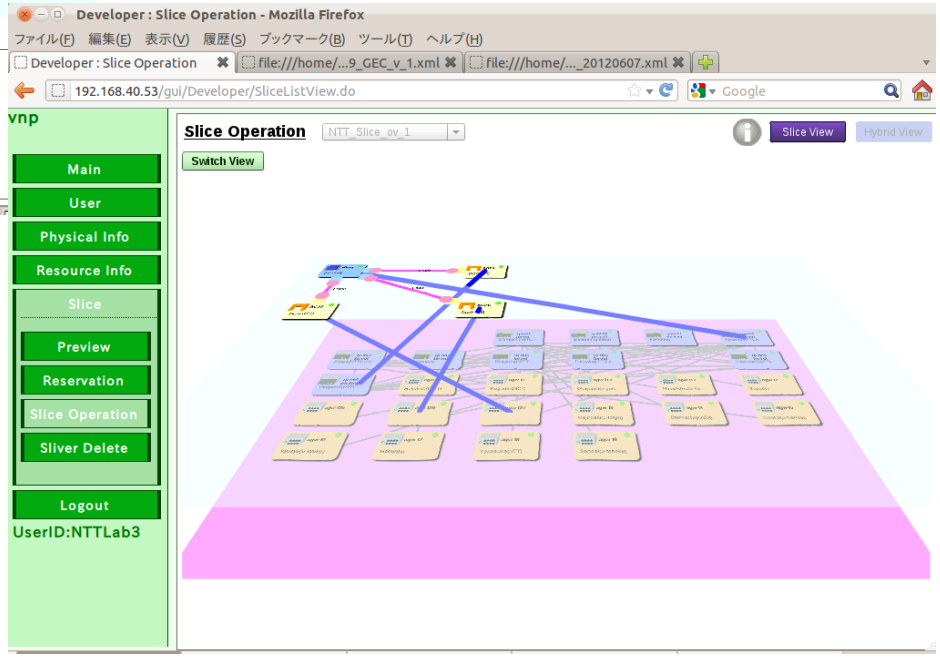
物理マッピング(明示指定)

Portal



スライス操作画面

スライスと物理ノードの対応



ドメインコントローラにおけるリソース管理

NTT

- 仮想化ノード毎のリソース量を管理
  - CPU, Memory, VM Instance, Card, NW帯域、キュー

Resource Information

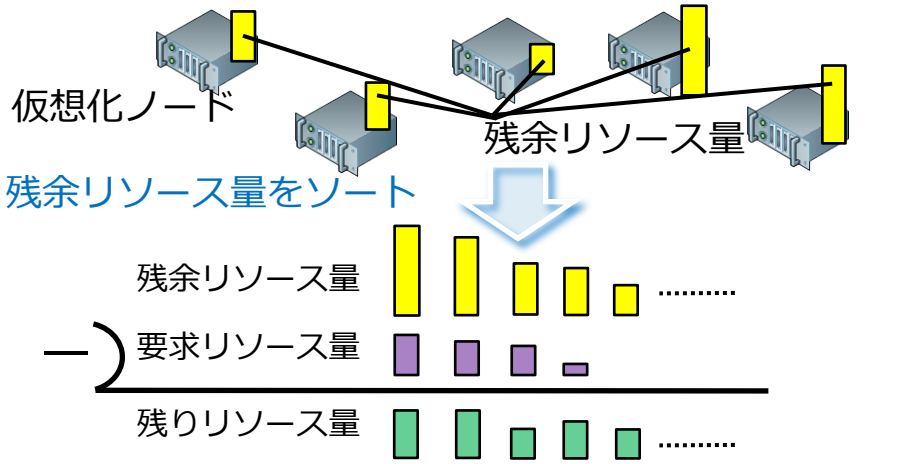
		Slow-Path VM			Best Effort VM	Fast-Path VM (Core)			Fast-Path VM (Card)	Network	
		Instance	CPU	Memory (MB)	Instance	Instance	vCPU	Memory (MB)	Instance	bandwidth (kbps)	queue
rp-nh1 (prog)	max	30	30	92160	60	40	40	6144	2	10170368	3
	used	2	3	3072	4	0	0	0	0	409600	0
	available	28	27	89088	56	40	40	6144	2	9760768	3
rp-nh10 (nc/sw)	max	0	0	0	0	0	0	0	0	10170368	3
	used	0	0	0	0	0	0	0	0	0	0
	available	0	0	0	0	0	0	0	0	10170368	3
rp-nh11 (nc/sw)	max	0	0	0	0	0	0	0	0	10170368	3
	used	0	0	0	0	0	0	0	0	0	0
	available	0	0	0	0	0	0	0	0	10170368	3
rp-nh2 (prog)	max	30	30	92160	60	40	40	6144	2	10170368	3
	used	3	4	6144	4	0	0	0	0	0	0
	available	27	26	86016	56	40	40	6144	2	10170368	3
rp-nh3 (prog)	max	30	30	92160	60	40	40	6144	2	10170368	3
	used	4	7	7168	4	0	0	0	0	409600	0
	available	26	23	84992	56	40	40	6144	2	9760768	3
rp-nh4 (prog)	max	30	30	92160	60	40	40	6144	2	10170368	3
	used	3	6	6144	4	0	0	0	0	0	0
	available	27	24	86016	56	40	40	6144	2	10170368	3
rp-nh5 (prog)	max	30	30	92160	60	40	40	6144	2	10170368	3
	used	3	6	6144	0	0	0	0	0	0	0
	available	27	24	86016	60	40	40	6144	2	10170368	3

ドメインコントローラにおけるリソース割当

- スライス定義にしたがって物理資源を割当（物理マッピングを明示）

- ドメインコントローラにて自動でリソースを割当（マッピングを省略）
  - 面的に広がったスライスを数多く収容するために、特定の仮想化ノードへの割当が集中しないようにリソースを割当

平準化割当て: 要求リソース量が大きな仮想ノードは、残余量の大きなノードに割当て
⇒ 特定ノードへの集中を防ぎ、均等な割当て



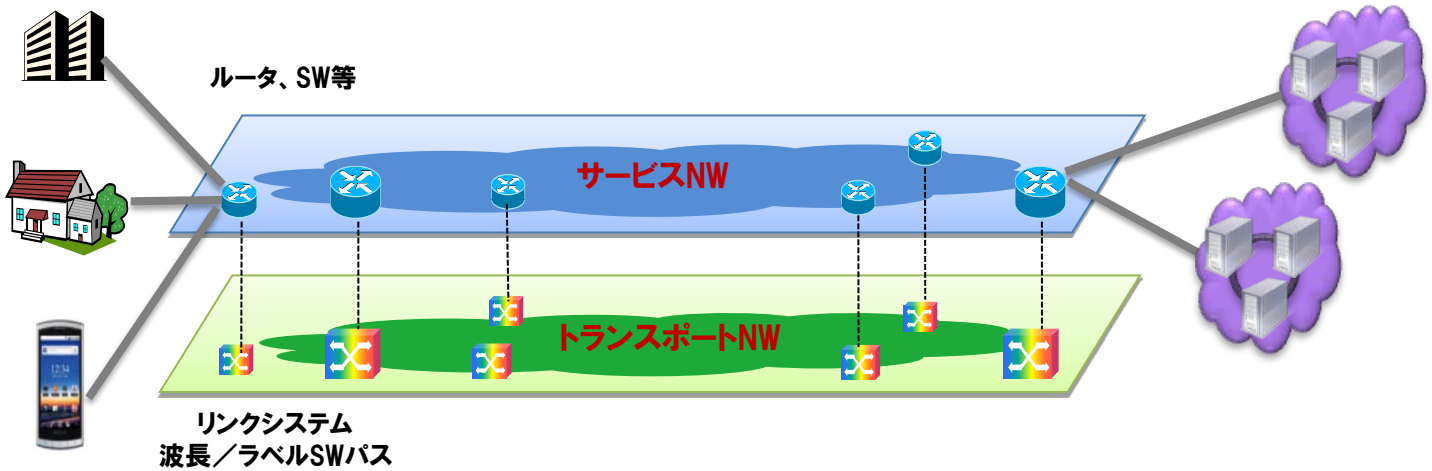
残りリソース量を均一化、多数のSliceを収容

Outline

- ネットワーク仮想化基盤の概要
- ネットワーク仮想化基盤の管理モデル
- 仮想ネットワーク制御管理
- **仮想ネットワークとトランスポートネットワークの連携制御管理**
- まとめ

トランスポートネットワーク

- 一般的なキャリア通信網の構成
  - ルータ等で構成されるサービスNW
  - 光ファイバ、波長、ラベルスイッチパス等で構成されるトランスポートNW
    - トランスポートNWを複数のサービスNWで共有利用
    - サービスNWとトランスポートNWを独立で運用

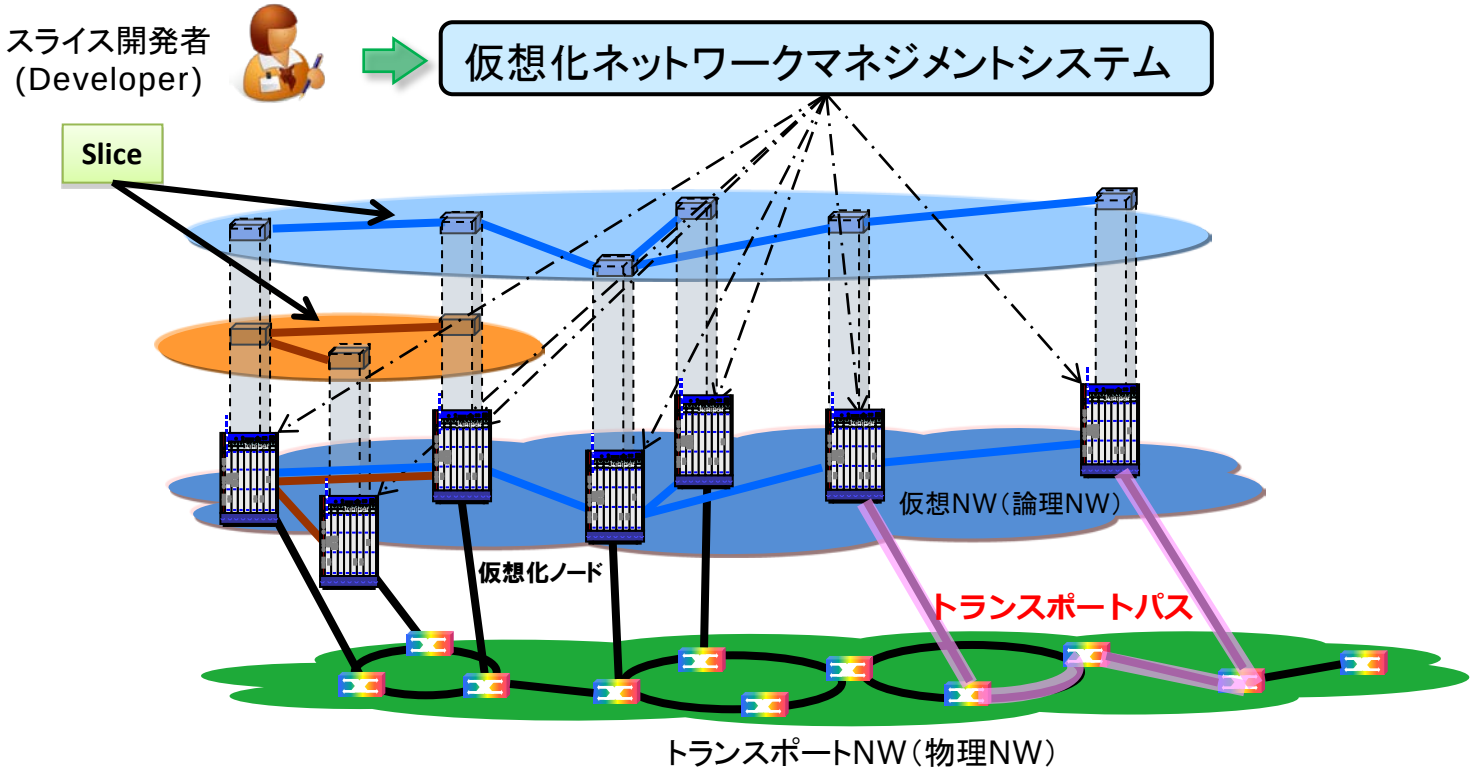


一般的なキャリアネットワークの構成

NW仮想化基盤におけるトランスポートNW

NTT

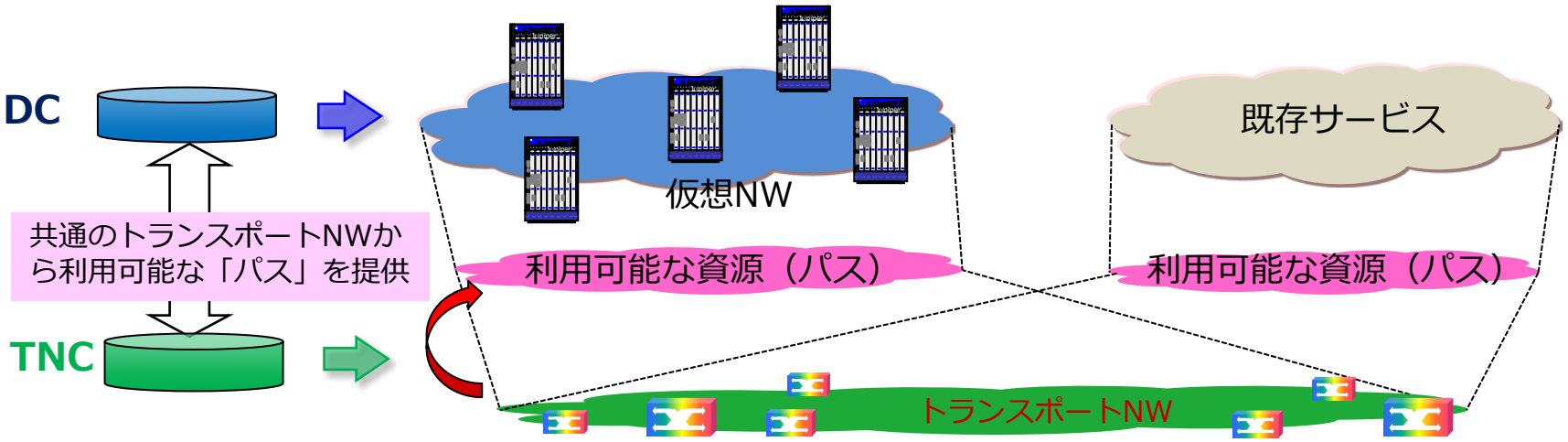
- ネットワーク仮想化基盤の構成
  - 仮想化ノードで構成されるNW：サービスNW
  - 仮想化ノード間はトランスポートNWで接続される
    - トランスポートネットワーク ：サービスの種別によらずパケット/フレームを転送
    - トランスポートネットワークは仮想化ノード間のTransport Pathを提供



トランスポートNWの管理

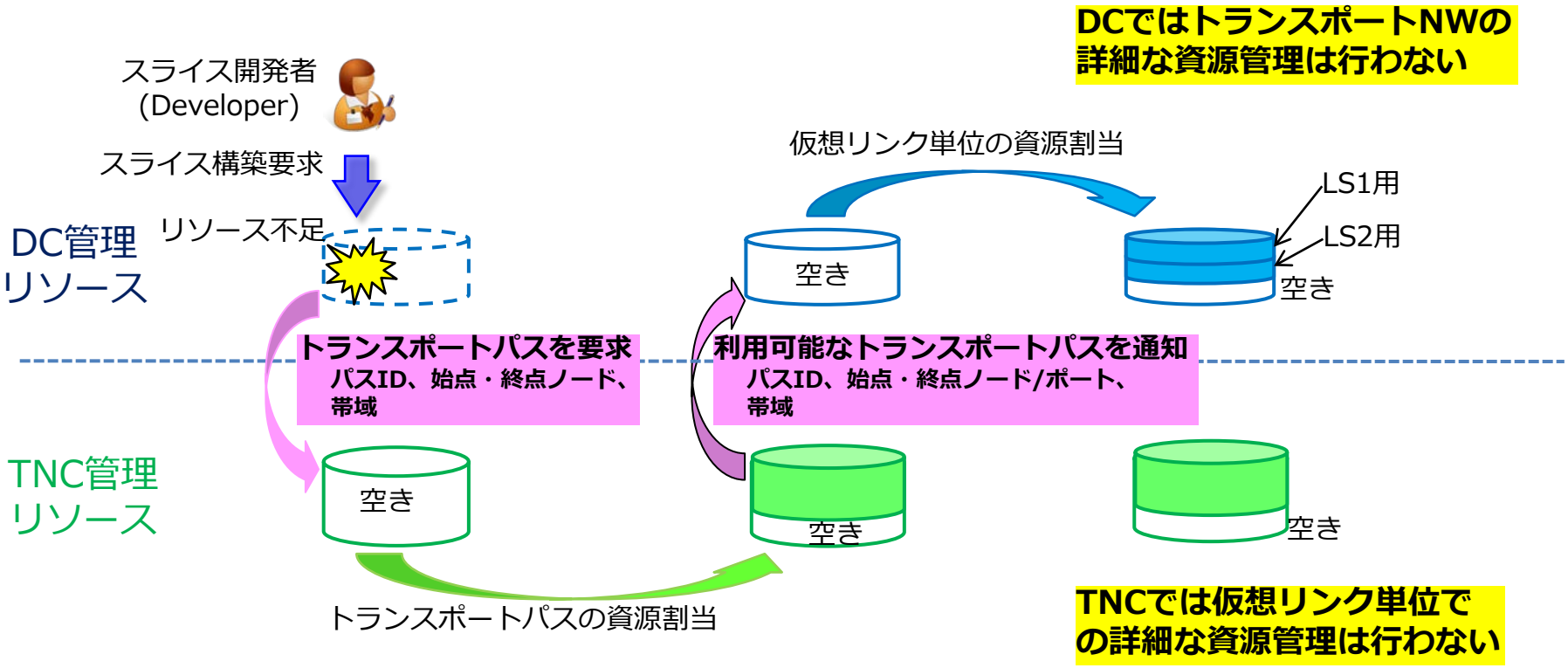
- トランスポートNW：既存サービスを含めて共有・相乗りすることでトランスポートNW資源の効率的な活用が可能
- 仮想NW管理＋トランスポートNW管理
  - スライスが利用出来るトランスポートNW資源の「パス」を作り出す

- トランスポートネットワークコントローラ(TNC)を仮想化ネットワーク管理システム（DC）とは独立に設置、TNCがトランスポートネットワークの管理/制御を実施
- 仮想化NW管理とトランスポートNW管理の明確な分離により複数のNWサービス間でトランスポートNWの資源の共有を実現
- DCとTNCを連携動作させることにより、スライス構築を契機に必要なトランスポートパスを生成



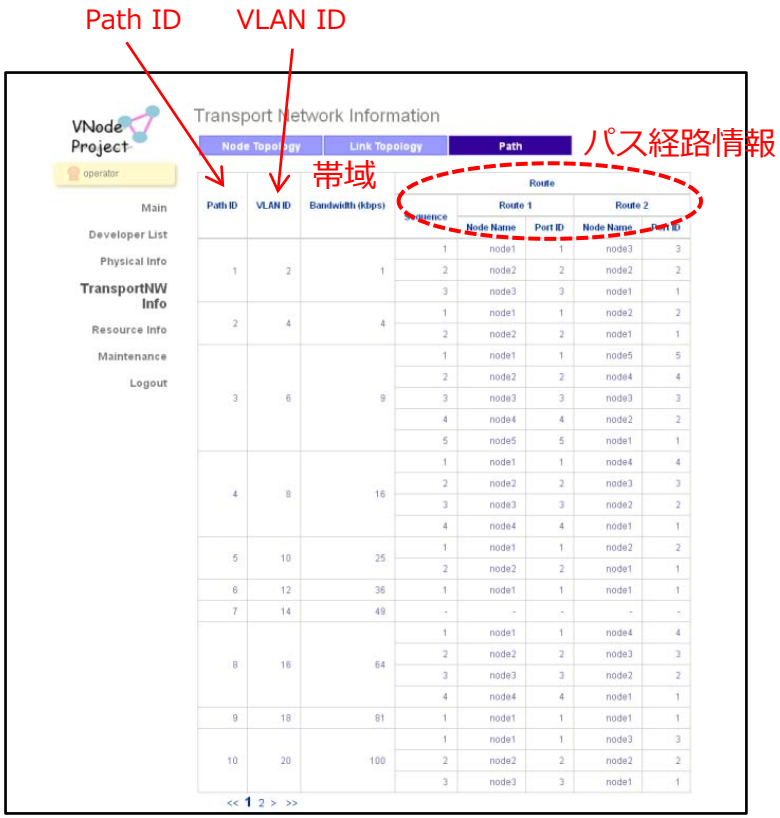
仮想NW管理とトランスポートNW管理の連携

- スライス開発者のスライス構築要求を契機に仮想リンク（LS）生成に必要なトランスポートパスの生成を動的に実行
  - DCとTNCの資源管理を明確に分離（互いの詳細な資源管理は行わない）
  - DCとTNCとの連携動作によるスライス構築要求に従ったトランスポートパスの生成手法

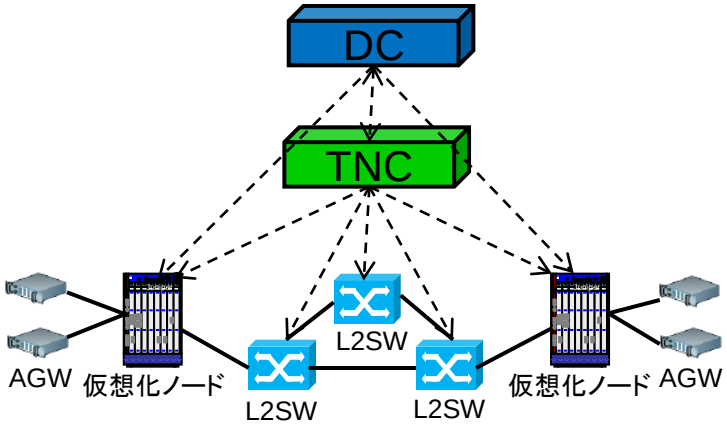


トランスポートNW管理の実装

- 機能拡張した仮想化ノード２台と市販L2SWを用いて実機による動作確認、機能検証を実施
  - VLANトランスポートパスの管理制御を確認



トランスポートネットワーク管理画面例



検証環境

まとめ

- 仮想化ネットワーク基盤の概要
  - 仮想化ノード
  - 仮想化NWマネジメントシステム
- 仮想化ネットワーク基盤の管理モデル
  - Operator, Developer, User
  - 6-Plane管理モデル
- 仮想ネットワーク制御管理
  - リソース管理と割当
- 仮想ネットワークとトランスポートネットワークの連携制御管理

  - 本研究は独立行政法人情報通信研究機構（NICT）の委託研究「新世代ネットワークを支えるネットワーク仮想化基盤技術の研究開発」として行われたものである