

通信事業者としてのSDN/OpenFlowに 対する期待と課題

2012年10月15日

NTTコミュニケーションズ株式会社

佃 昌宣

m.tsukuda@ntt.com

Agenda

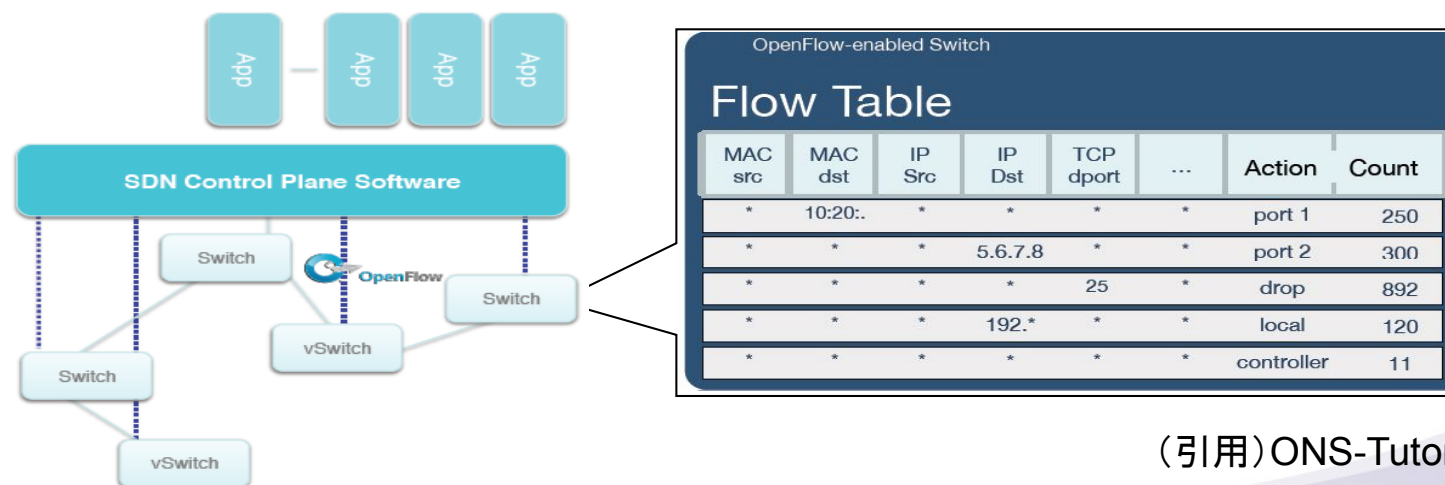
- SDN/OpenFlowの概要と期待すること
- 実際に既存のいろいろな製品を触ってみる
- WANへ適用する場合を考えてみる
- まとめ



SDN/OpenFlowの概要と期待すること

SDN/OpenFlowとは？

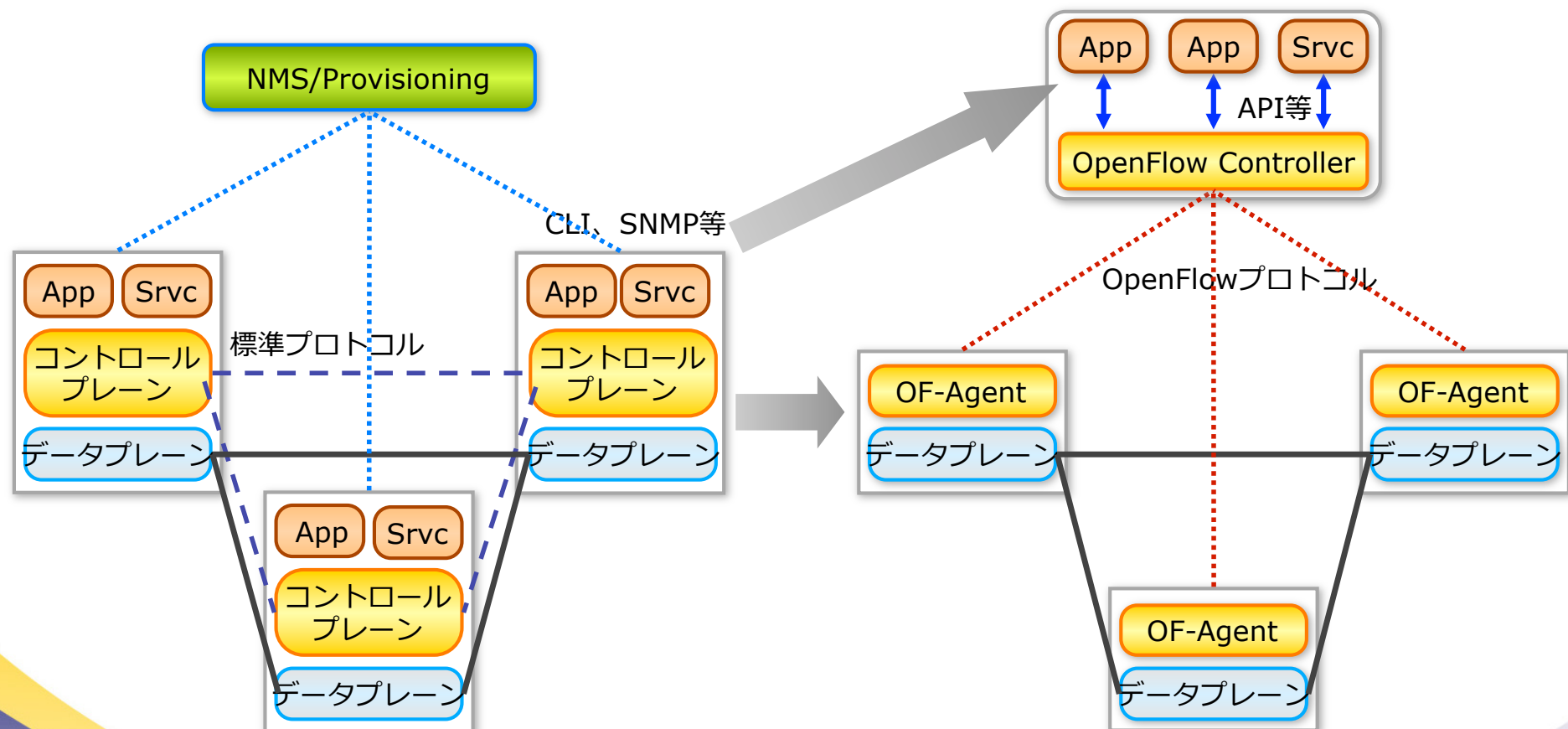
- SDN (Software Defined Network) とは？
 - 明確な定義はない？
 - ソフトウェアコンポーネントによってネットワークをダイナミックに制御する
 - <http://itpro.nikkeibp.co.jp/article/NEWS/20121010/428930/>
 - ネットワーク構成を動的に設定するために、ネットワーク全体をソフトウェアで制御（定義）する
 - http://cloud.watch.impress.co.jp/docs/interview/20120925_560748.html
 - 特徴：C/D分離、集中制御、直接的なプログラマビリティ
- OpenFlowとは？
 - OpenFlowはSDNを実現する技術の一つ
 - OpenFlowSW、コントローラー、OpenFlowプロトコルの3つで構成される
 - OpenFlowSWはFlowTableに基づいてパケットを転送していく



OpenFlowは何か新しいのか？

元々装置の内部にあった2つの機能を分離した

- アーキテクチャを変えただけ
- ネットワーク方式に対し、何か新しい技術をもたらしたわけではない



現在の一般的な構成

OpenFlowの構成

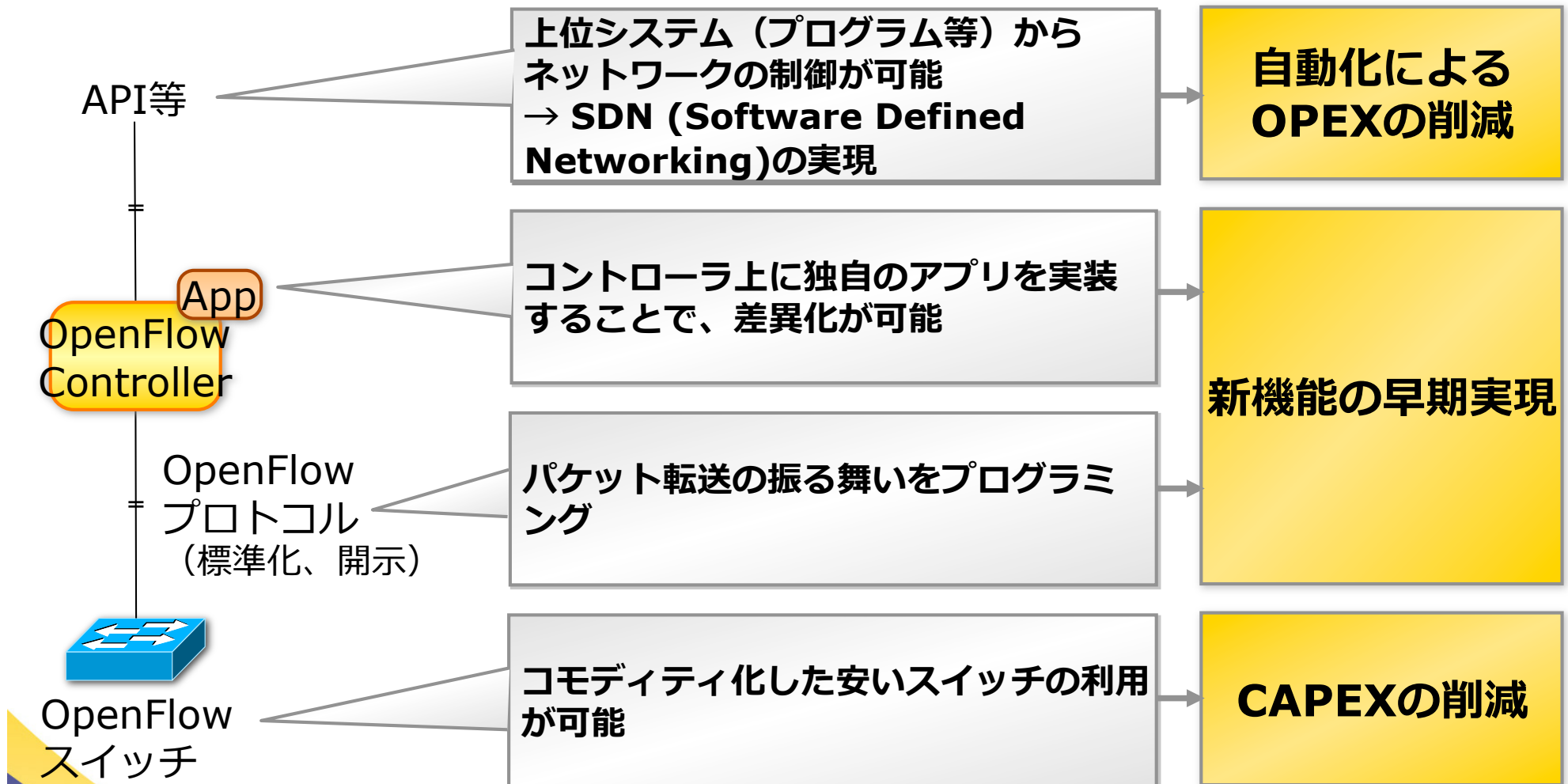
OpenFlowの標準化動向

	2006-2011.02	2011.03-
基礎的 研究活動	Stanford Clean Slate Program (旧ドコモUSA研、DT、Google、NEC、Cisco、Ericsson、Huawei、Xilinx)	Open Networking Research Center (ONRC) ・ Stanford大とUC Berkleyの共同リサーチチーム ・ ONRC Tech Team (企業)
研究と 実装活動	Stanford Clean Slate Laboratory (NEC、DT)	
デファクト標準と オープンソース配布	OpenFlow Switch Consortium (DT、Google、NEC、Cisco、HP、Juniper、Ericsson、Dell、Broadcom) ・ 2010.01 v1.0公開 ・ 2011.02 v1.1公開	OpenFlow Switch Consortium (www.openflowswitch.org) ・ オープンソースの配布と管理
標準化団体		Open Networking Foundation (ONF) (現在 81社 2012.10時点)

(*)NTT Comは設立当初よりメンバとして参画。2011年12月よりボードメンバー就任

2010年1月	OF v1.0	12Tupleのテーブル、テーブルミスマッチ時にPacket-In
2011年2月	OF v1.1	マルチテーブル、グループテーブル、MPLS
2011年12月	OF v1.2	マルチコントローラ、可変長マッチフィールド、IPv6
2012年4月	OF v1.3.0	帯域情報、SecureChの負荷分散、PBB、ミスマッチ時のPacket-In見直し
⋮	OF v1.3.1 (予定)	(バグフィックス)
2013年1月～	OF v1.4 (予定)	(v1.3と少し間をあけて実装を促す)

OpenFlowがなぜ注目されているのか？

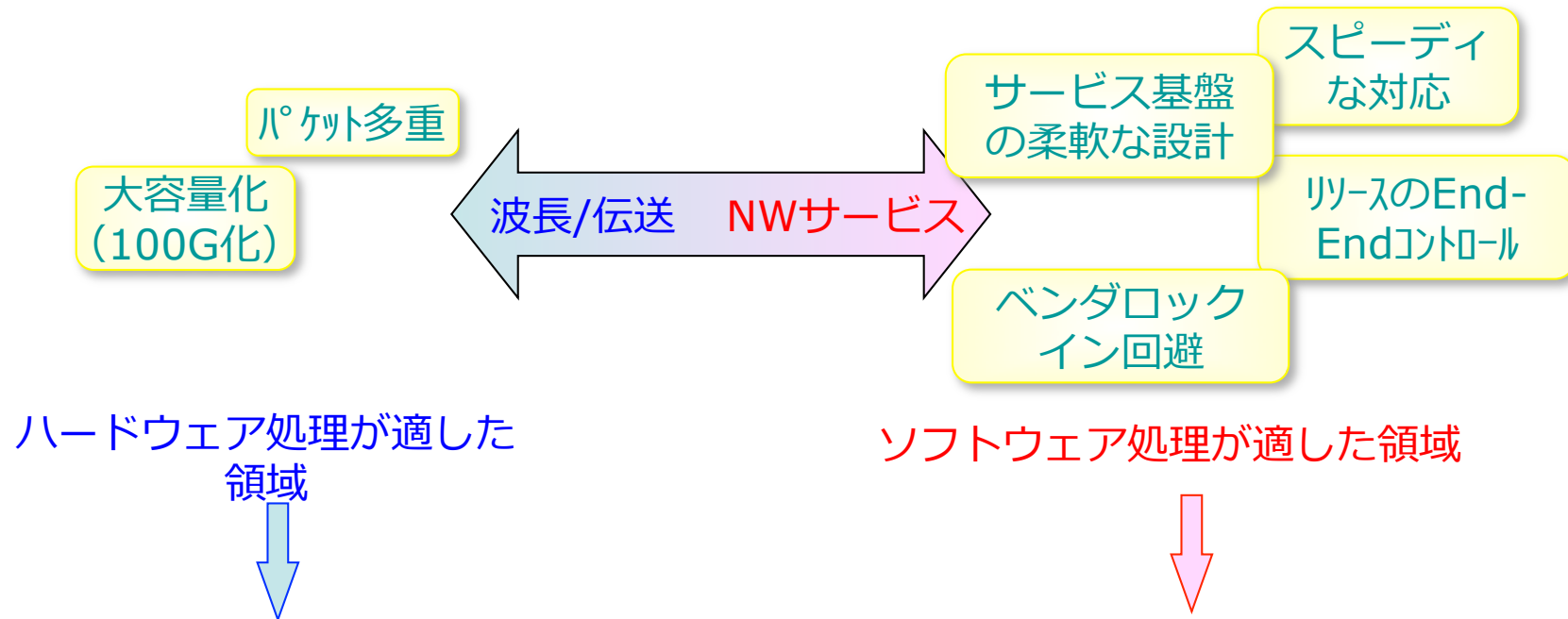


SDN/OpenFlowに期待すること

- Time-to-Marketの短縮
 - ベンダーの開発ロードマップやスケジュールに依存せず、自社に必要な機能を早期に導入することができる
- サービスの差異化
 - 市中製品の仕様に縛られず、上位システムや独自アプリなどを連携させることでサービスの差異化ができる
- CAPEX/OPEXの削減
 - コモディティハードウェア利用によるCAPEXの削減
 - SO業務や運用の自動化によるOPEXの削減

今後のネットワークの方向感

「スピード、柔軟性」と「高速、大容量」の両立



Next Gen PTN/Optical

NWサービス機能を支える波長／伝送は、ハードウェアでNWをコントロールする既存技術で対応する

SDN/OpenFlow

柔軟性でスピーディなNWサービス機能の実現に、ソフトウェアでNWをコントロールするSDNを導入する

- SDN/OpenFlowのコンセプトはいろいろ良いことがあるそうだけど、実際に使っていくためには？
- いろいろ試したり考えてみたりして、課題をあげてみる　これが解決すると　SDN/OpenFlowはより使えるかも！！！！



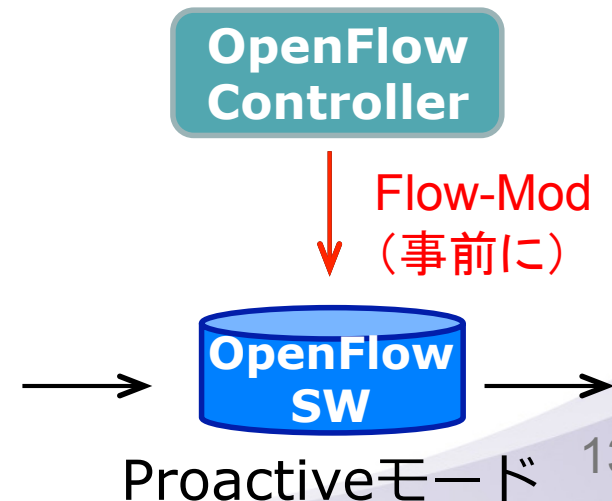
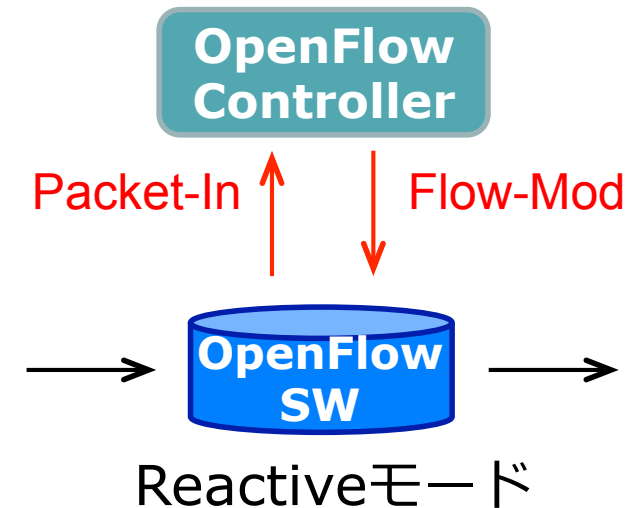
- まずはいろいろ試してみる
 - － 実際に既存のいろいろな製品を触ってみる
- 通信事業者なので、、、
 - － WANへ適用する場合を考えてみる

実際に既存のいろいろな製品を触ってみる

やってみたこと

- フローテーブル学習機能検証
- フローテーブル書き込み検証

- OpenFlow技術を用いてフローテーブルを学習させる方法
 - Reactiveモード
 - スイッチはUnknownパケット受信を契機にOpenFlow (Packet-In) を用いてControllerへ問合せ
 - Controllerから未学習パケットのフローを“Flow-Mod”で設定
 - Proactiveモード
 - フローエントリを事前に“Flow -Mod”を用いて設定

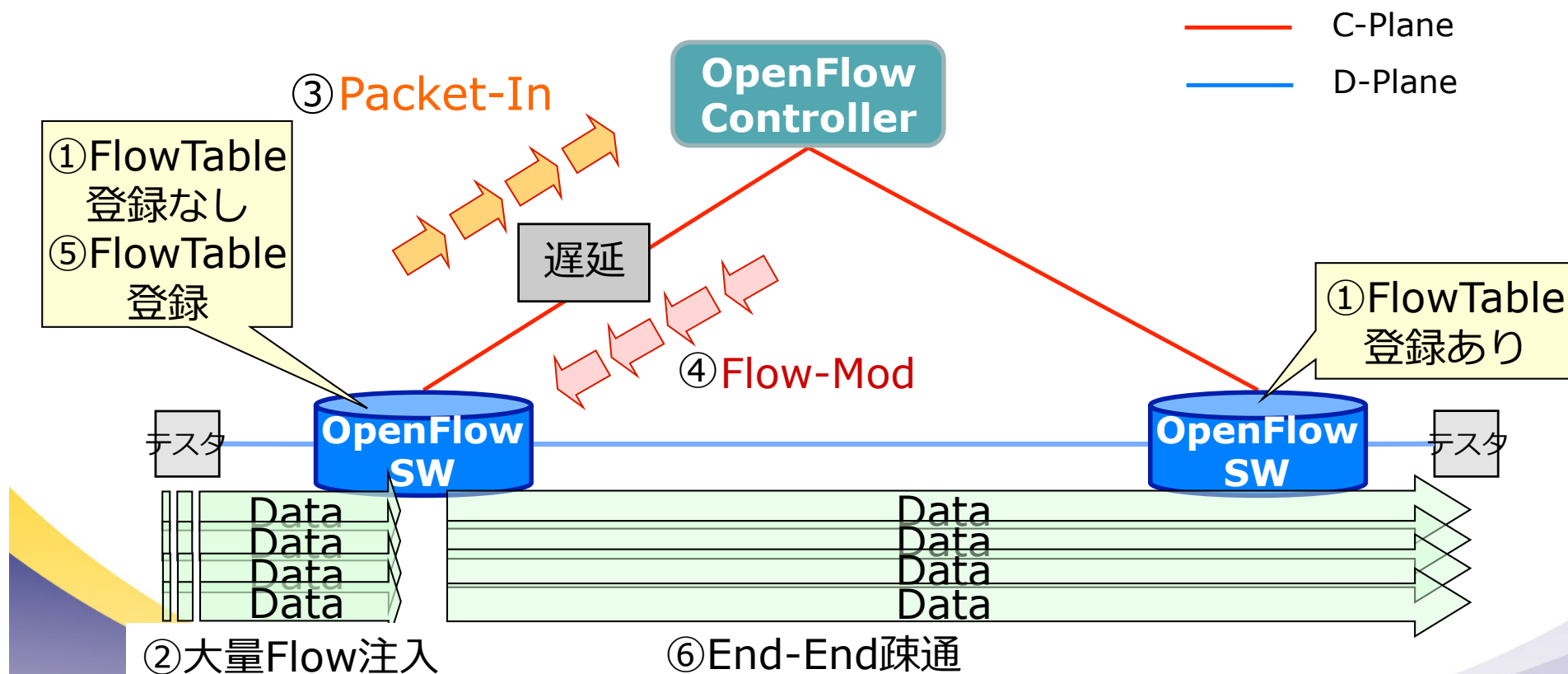


- 目的
 - OpenFlowの特徴であるC-Plane/D-Planeの分離に注目し、ReactiveモードにてC-PlaneにFlow数の増大や遅延があった場合の影響を検証



フローテーブル学習機能：検証内容

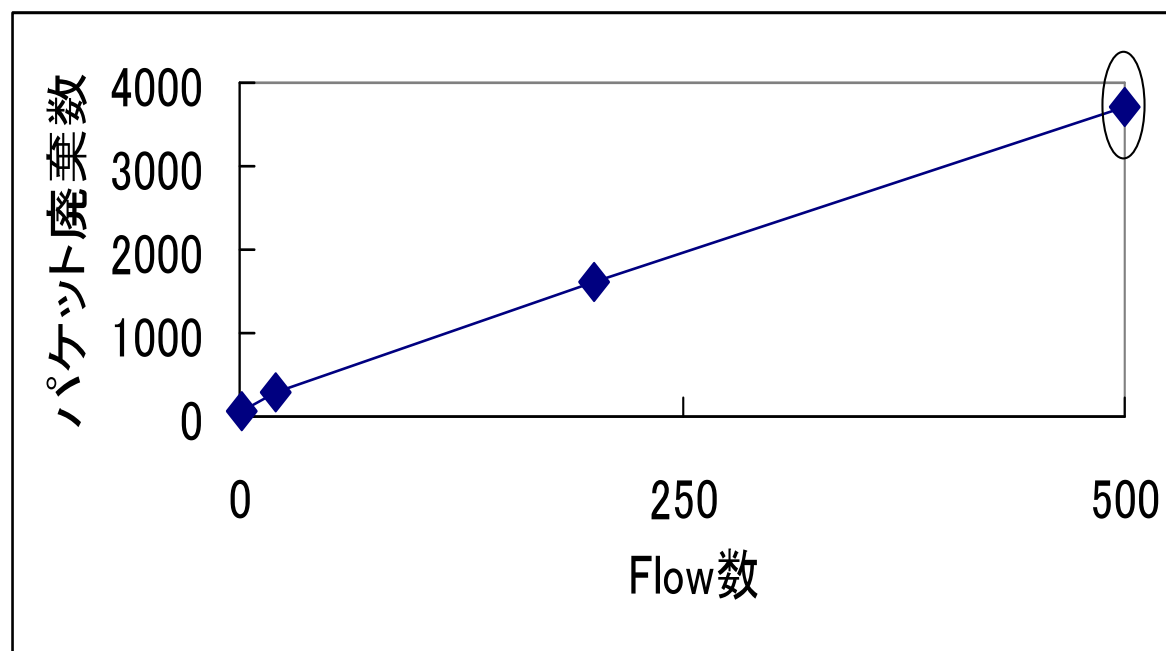
- C-Plane性能試験
 - 未知のFlowを大量注入した際のPacket-In、Flow設定の処理
 - ①～⑥のEnd-End疎通までのPacket廃棄量を測定
 - (試験a) Flow数を増大させた場合の影響(1～500Flow)
 - (試験b) 遅延を増大させた場合の影響 (0～200ms)



- (試験a) Flow数を増大させた場合の影響

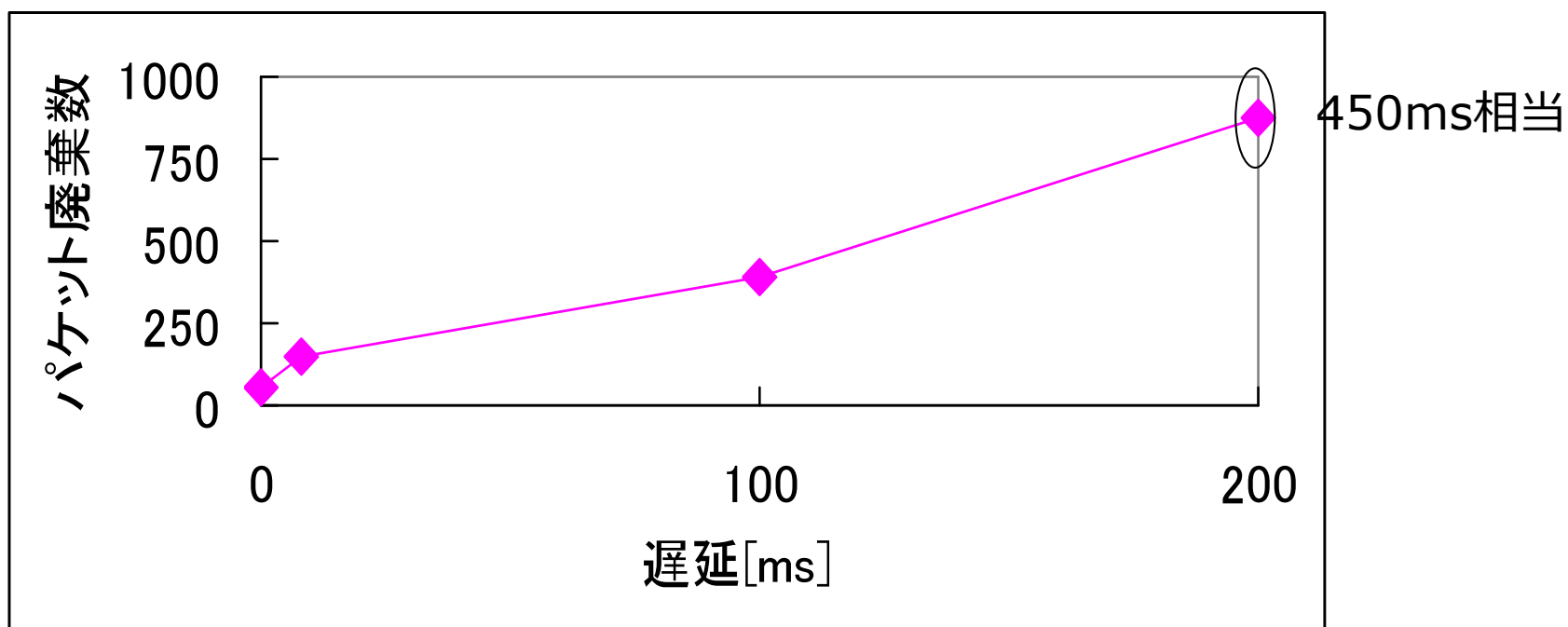
- 試験条件

- フロー数：1フロー、10フロー、200フロー、500フロー
- OpenFlow Channel 遅延：0ms



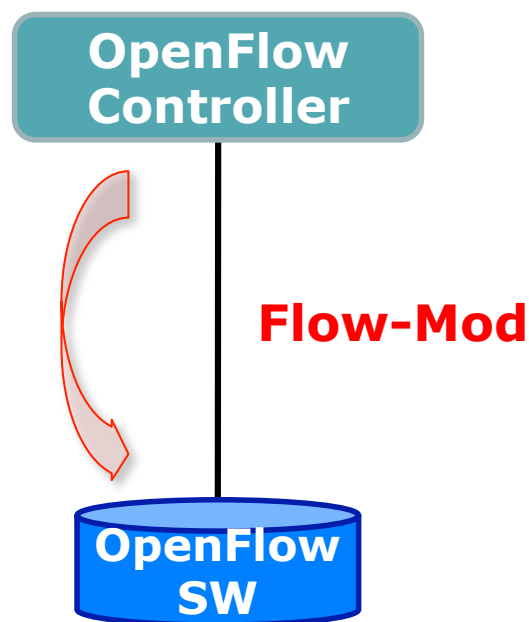
バースト的に新規フローが流れてきた場合、CPU処理の制約によりパケットが輻輳し、フロー数が多くなると最後のフローが登録されるまで時間がかかる

- (試験b) 遅延を増大させた場合の影響
 - － 試験条件
 - フロー数：1フロー
 - OpenFlow Channel 遅延：0.0ms、8.2ms、100ms、200ms



遅延の増加に伴いフローテーブルの学習に時間がかかる

- Proactiveモードによるフロー書き込み試験
 - バースト的にControllerからフローを書き込むことによる耐性試験
 - 経路切り替えが起きた際のパフォーマンス確認のため
 - 最大フローエントリー数の確認



フローテーブル書き込み検証：結果

ベンダ	製品名	測定値	備考
A社	A-1(10G)	768フロー	Chip-X
	A-2(GbE)	1637フロー	Chip-Y
B社	B-1	750フロー	Chip-X
C社	C-1	4096フロー	Chip-Z

- バースト的に書き込みを行ったが各スイッチ問題なく処理をすることができた
- フローテーブル数はネットワークプロセッサの影響が大きい
- 既存製品レベルだとフローテーブル数が少ない

実際に触ってみて分かった（改めて分かった）課題

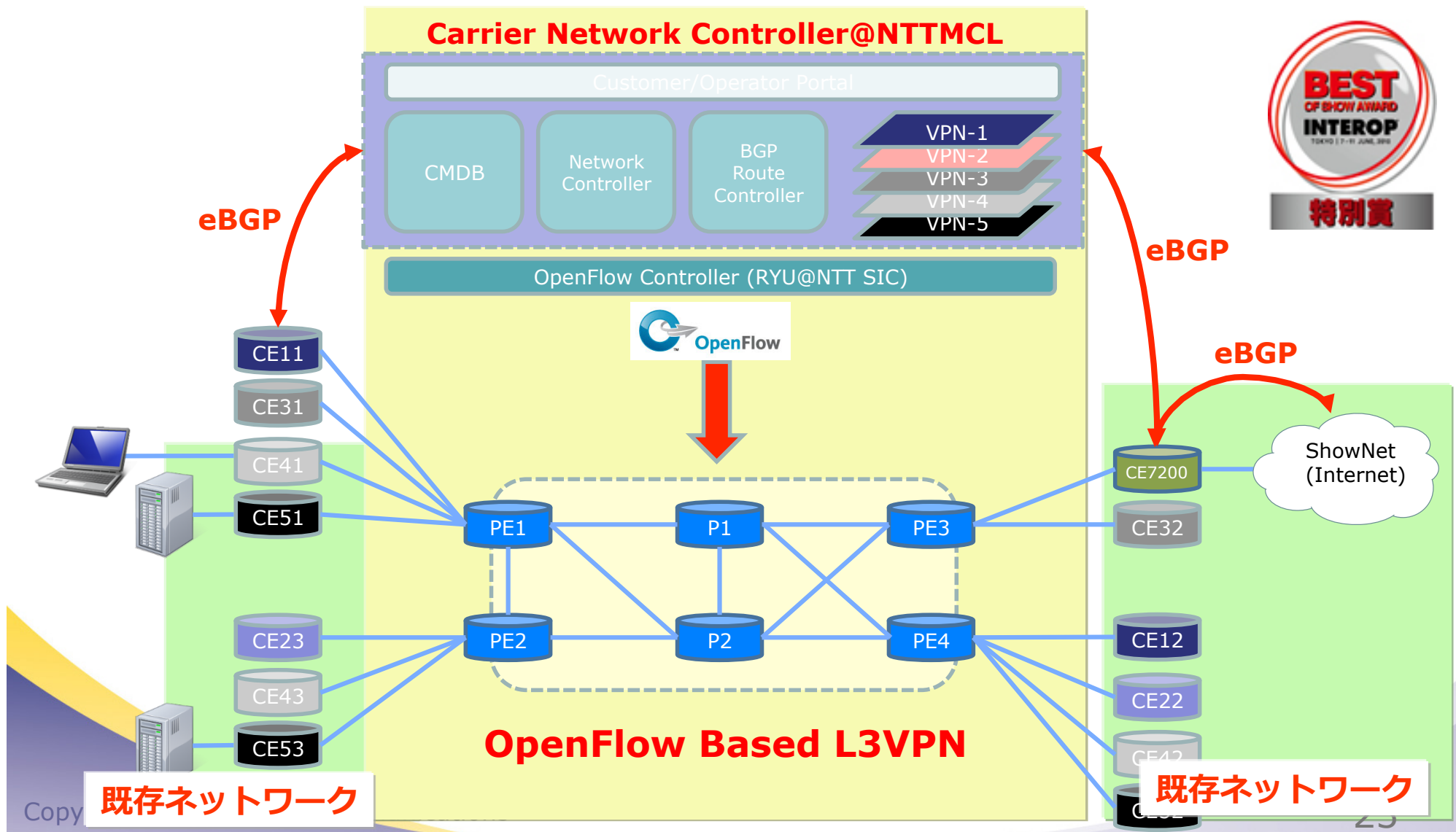
- Packet-In過多時の影響
 - 重要パケットにはQoSをかけてPacket-Inさせる
- C-Plane遅延による影響
 - コントローラ・スイッチの全体設計に考慮が必要
- Proactiveモード Reactiveモード
 - 上の結果を踏まえるとProactiveモードを中心として設定し、それ以外のパケットをReactiveモードで設定するのがよさそう
 - どちらの設定もできるような実装を希望
- フローテーブル数
 - フローテーブル数を多く持てるコモディティスイッチは作れるか？
 - これからに期待か
 - 既存Chipでも定義でテーブル領域を大きくすることは？
 - 検索するTupleを工夫することでフローテーブル数を稼げるか？
 - FPGAで安く作ることとはできるのか？

WANへ適用する場合を考えてみる

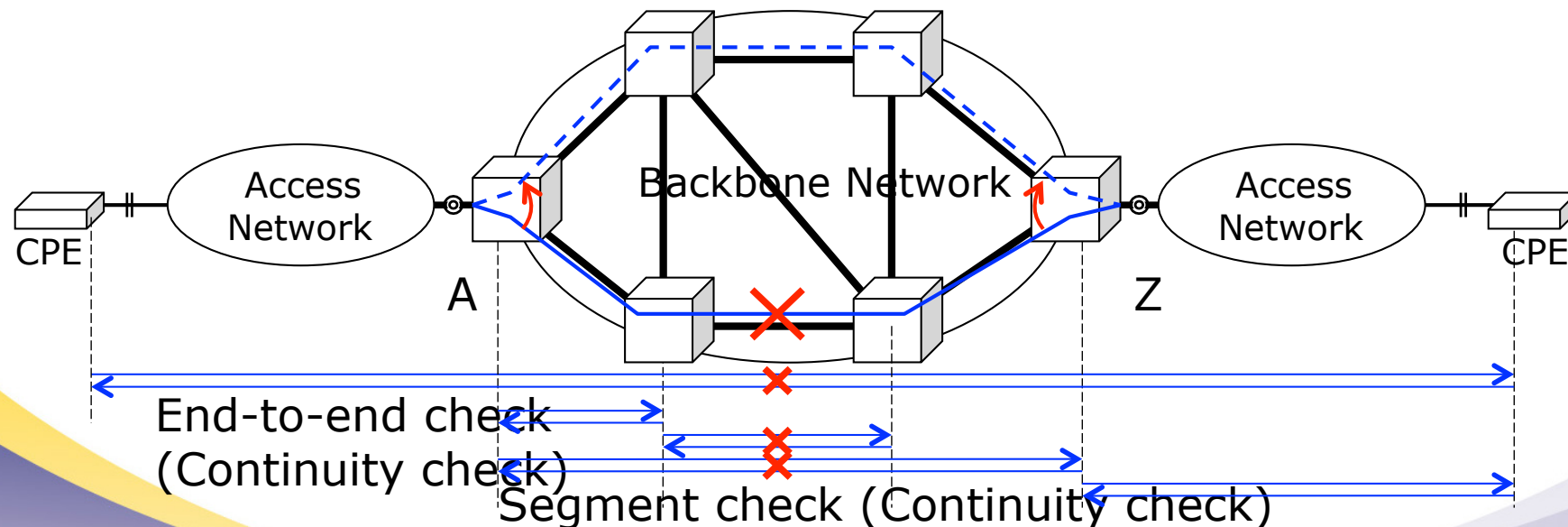
- 既存ネットワークをOpenFlowネットワーク経由で接続
 - 経路情報の伝播
 - 既存プロトコル（BGPなど）で実施 → スケーラビリティを考えるとこれがよさそう
 - Interop2012でのデモ
 - APIを介したシステム間で連携して実施することも可能？
- OpenFlowネットワークを既存ネットワーク経由で接続やOpenFlowネットワーク間接続（将来的に？）
 - これも実現方法の検討が必要

Interop2012でのデモ

BGPのプロトコル処理をコントローラ上で集中制御
外部ネットワークからはIP-VPNと等価に見せかける

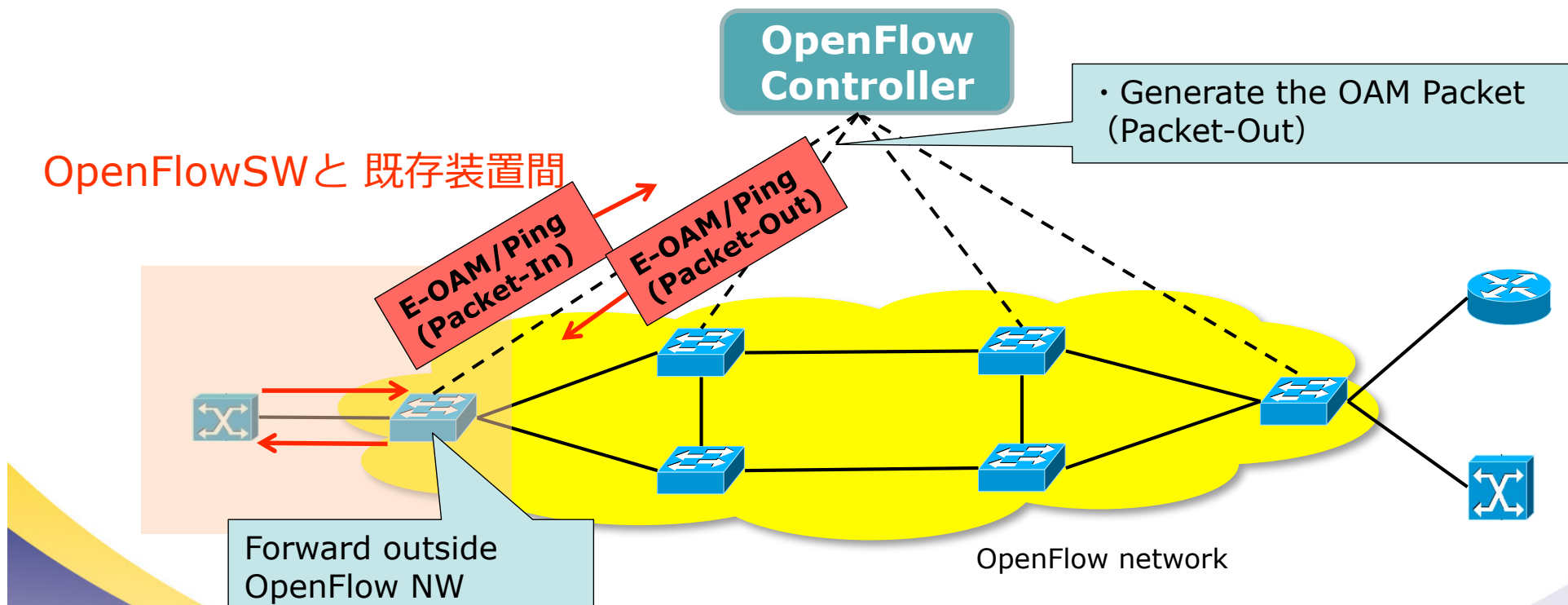


- 適用範囲を拡大する為には、運用に必要なネットワーク機能がまず先に必要
 - サービス／通信の正常性を確認する仕組み（開通時、運用時）
 - 「物理故障個所」と「サービス（論理NW）」の影響を特定
 - 異常検知、異常個所切り分け、冗長構成切り替えの仕組み、等
- ソフトウェアで実装することで、汎用スイッチで必要最小限のOAM機能を実装できないか？



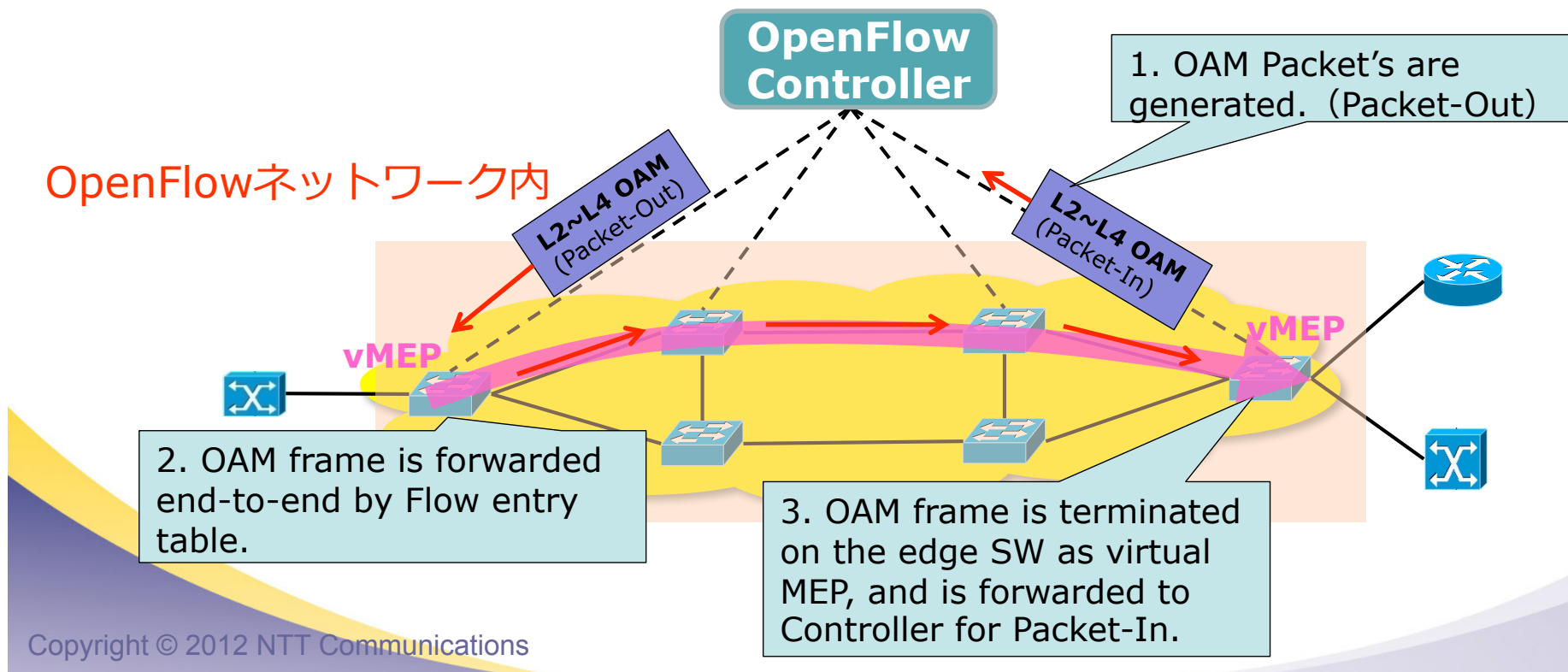
OAM監視機能:実現方式 1

- OpenFlowネットワークと既存装置間監視
 - 既存装置との接続性
 - L2 : Ethernet OAM、L3 : ICMP/BFD

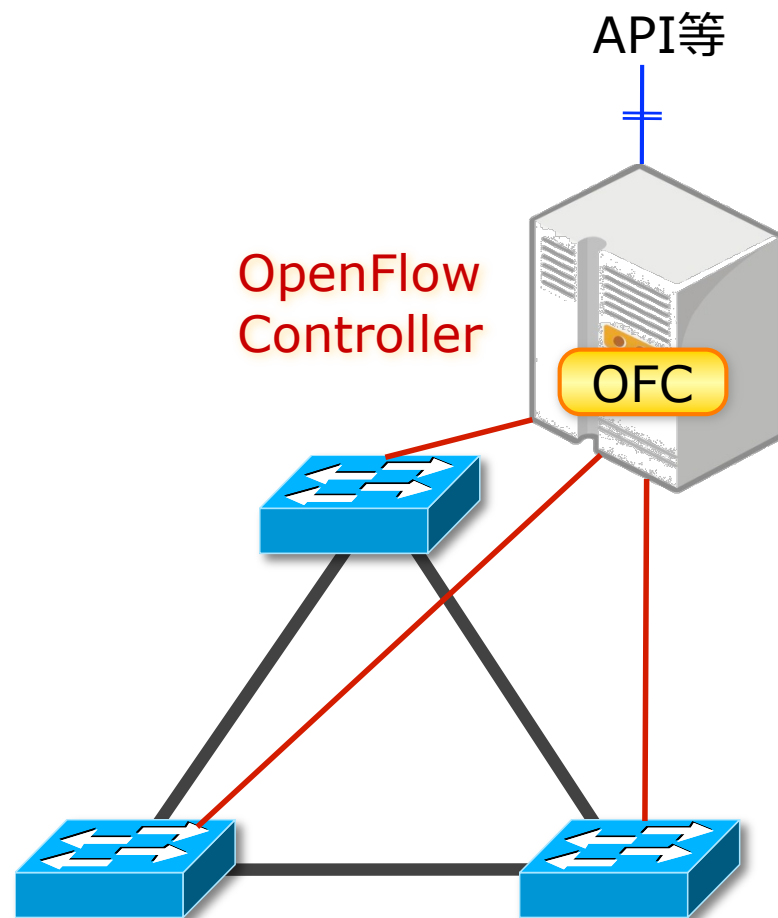


OAM監視機能:実現方式2

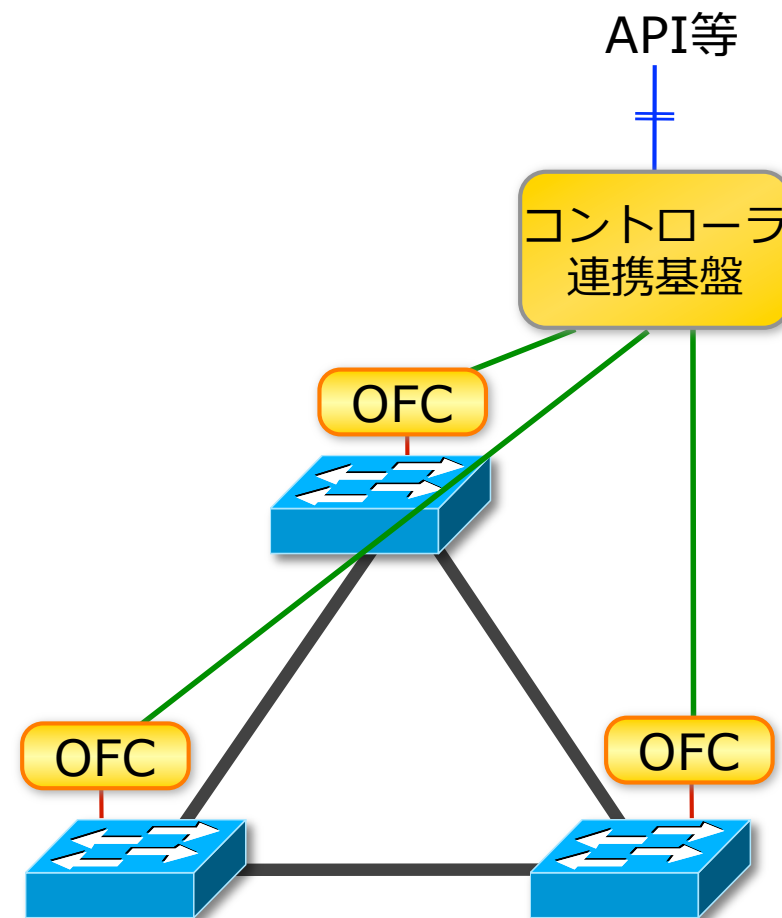
- OpenFlowネットワーク内の常時接続監視
 - フローエントリーの通りにパケットが疎通可能か確認できることが重要（開通、監視）
 - L2 : Ethernet OAM、L3 : ICMP/BFD
 - フローエントリーがMulti Tupleの場合の検討も必要



(参考) 分散型のOpenFlowコントローラ

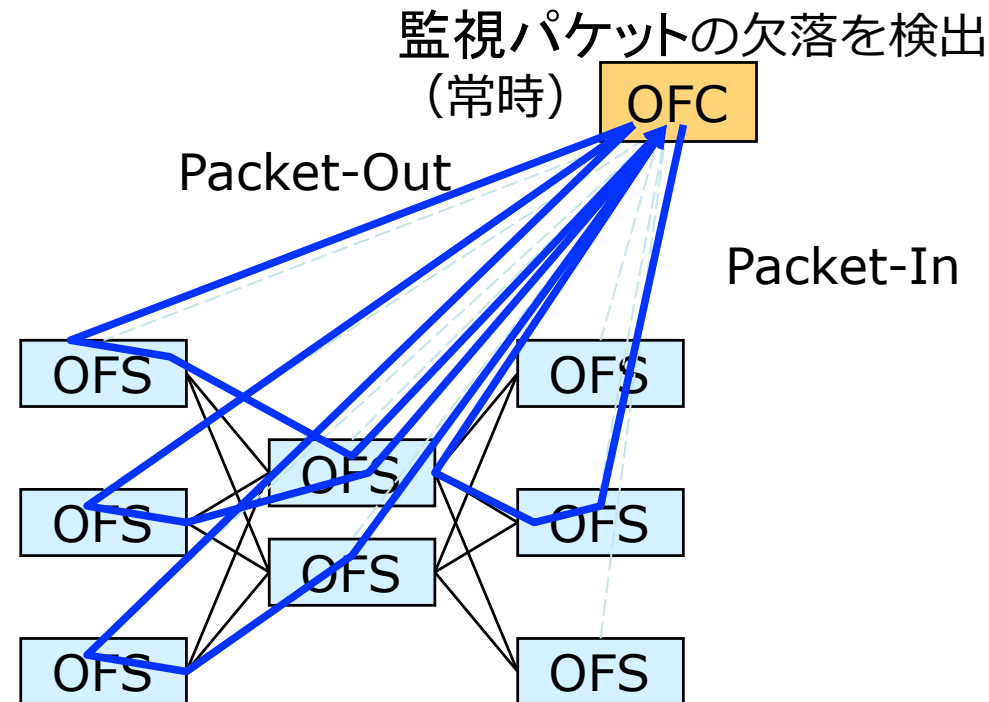


集中型のOpenFlowコントローラ
(physically centralized)



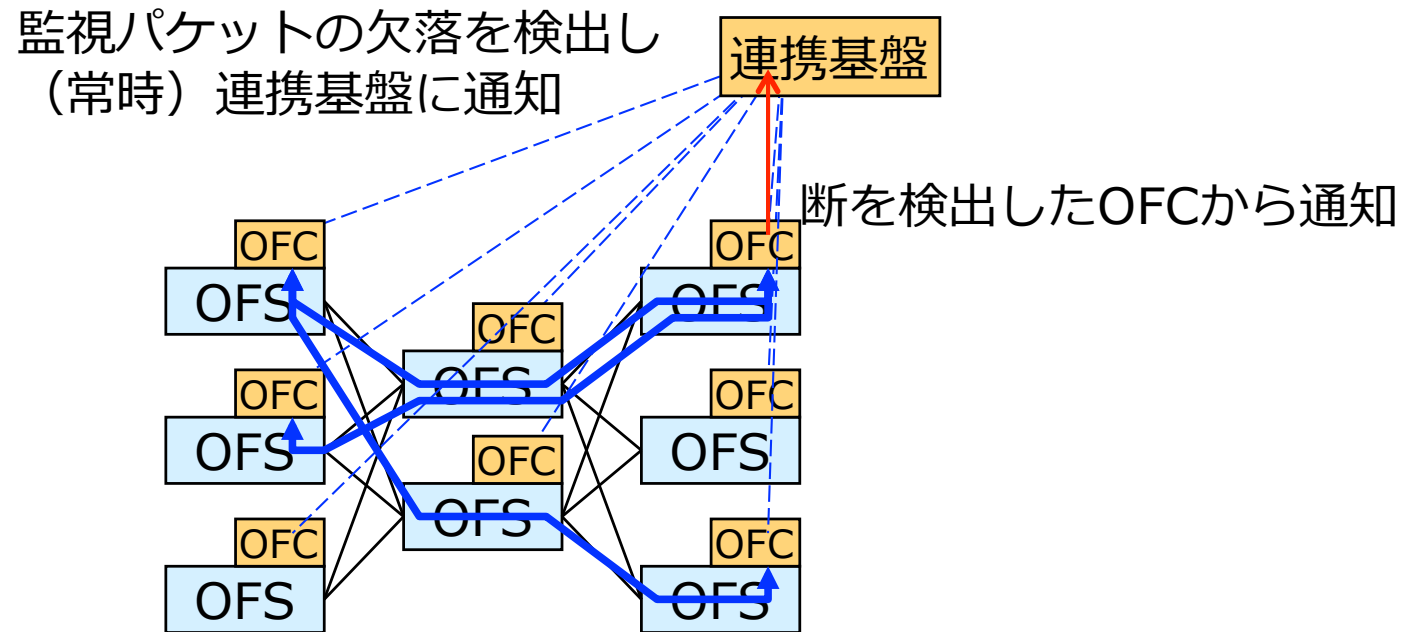
分散型のOpenFlowコントローラ
(logically centralized)

- 集中型 (physically centralized)



フロー数、スイッチ数の増加に伴い、コントローラの処理能力がボトルネックになる。OFC－OFS間の断でもネットワークが切れたことになる

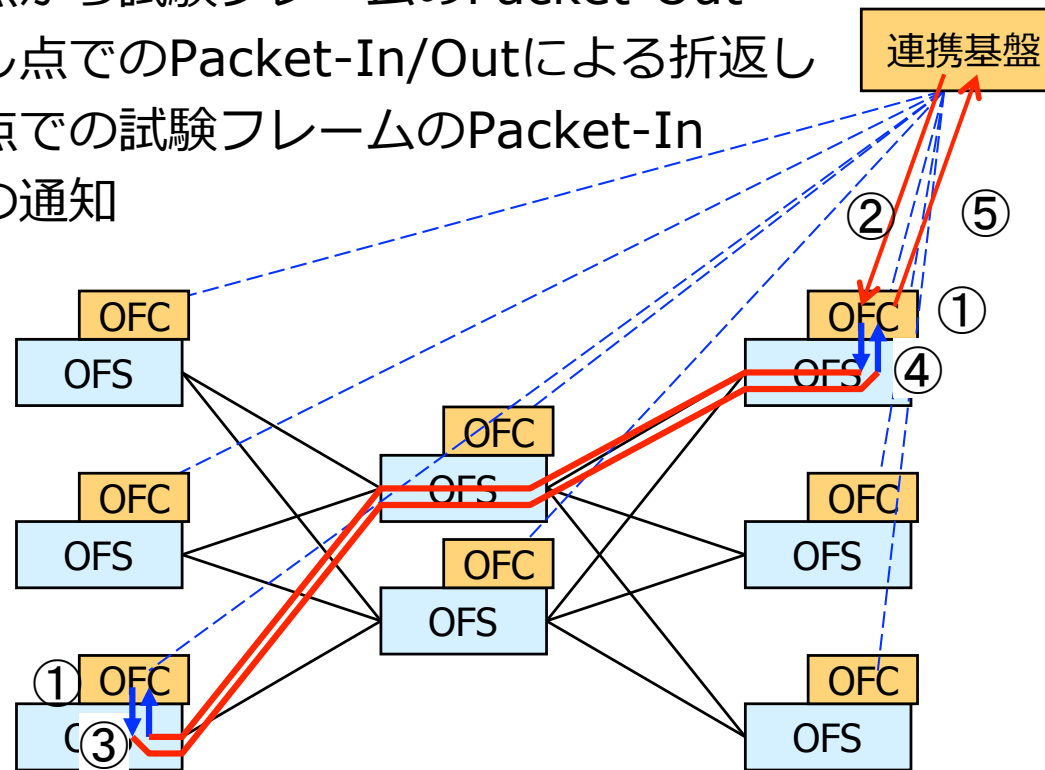
- 分散型 (logically centralized, physically distributed)



コントローラがボトルネックにならないように、分散配備したほうが良い機能は分散的に実装する

- ループ試験

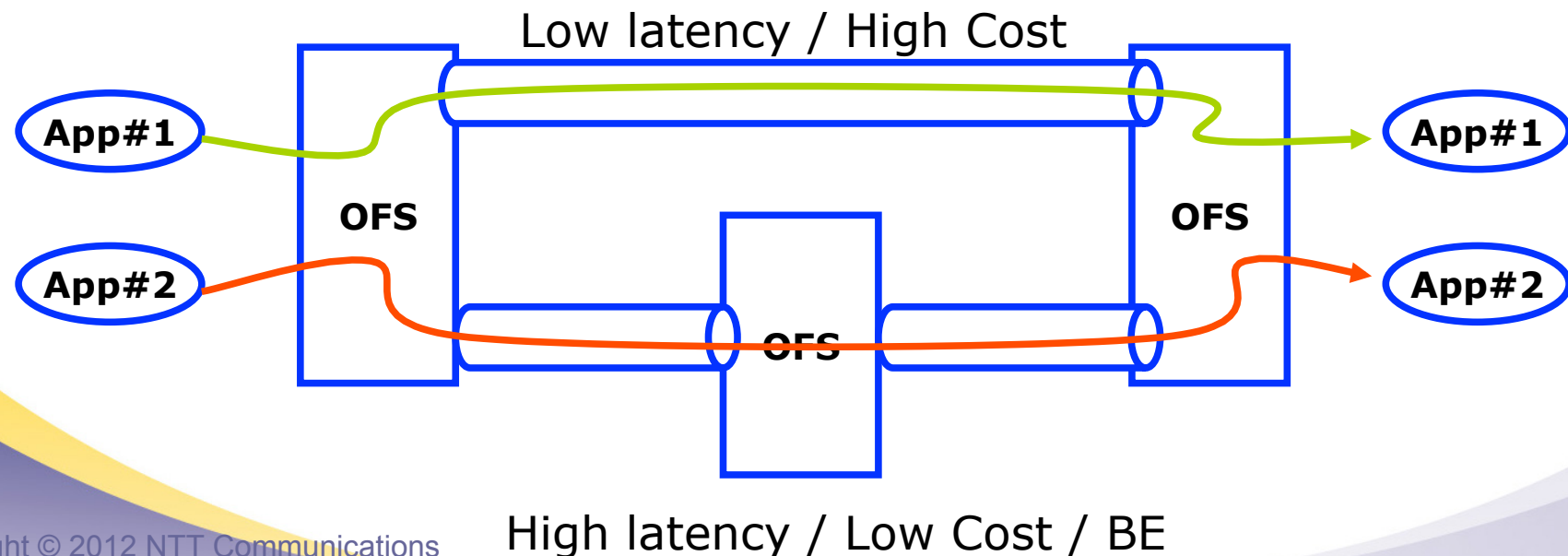
- ①試験点、折返し点でのPacket-In/Outのテーブル設定
- ②試験点から試験フレームのPacket-Out
- ③折返し点でのPacket-In/Outによる折返し
- ④試験点での試験フレームのPacket-In
- ⑤結果の通知



試験フレームのヘッダを試験対象フローと同一にすることで、同一系路上の導通確認が可能。試験フレームの生成、挿入、ドロップといった実装がソフトウェア化することでハードの実装を軽くできる

TEをどうするか？

- マルチパス負荷分散
 - 複数経路に効率的にトラフィックを流したい
- Applicationの要求品質に応じた経路選択
 - 柔軟なサービスメニューの提供
 - オンデマンド／カスタマ制御でQOSを変化
- 実現するためには
 - MPLSなどとHybrid → OpenFlowとの連動
 - OF1.3のPer Flow Meters機能を利用
 - TEインテグレーション



WANへ適用する場合の課題

- OpenFlowネットワークと既存ネットワークとの接続
 - 既存ネットワークをOpenFlowネットワーク経由で接続
 - 経路情報の伝播
 - OpenFlowネットワークを既存ネットワーク経由で接続やOpenFlowネットワーク間接続（将来的に？）
 - これも実現方法の検討が必要
- OAM機能
 - 実現方式
 - OpenFlowネットワーク – 既存ネットワーク
 - OpenFlowネットワーク内
 - アーキテクチャ
 - 集中型
 - 分散型
- TEをどうするか？
 - マルチパス負荷分散
 - Applicationの要求品質に応じた経路選択
 - 実現するためには

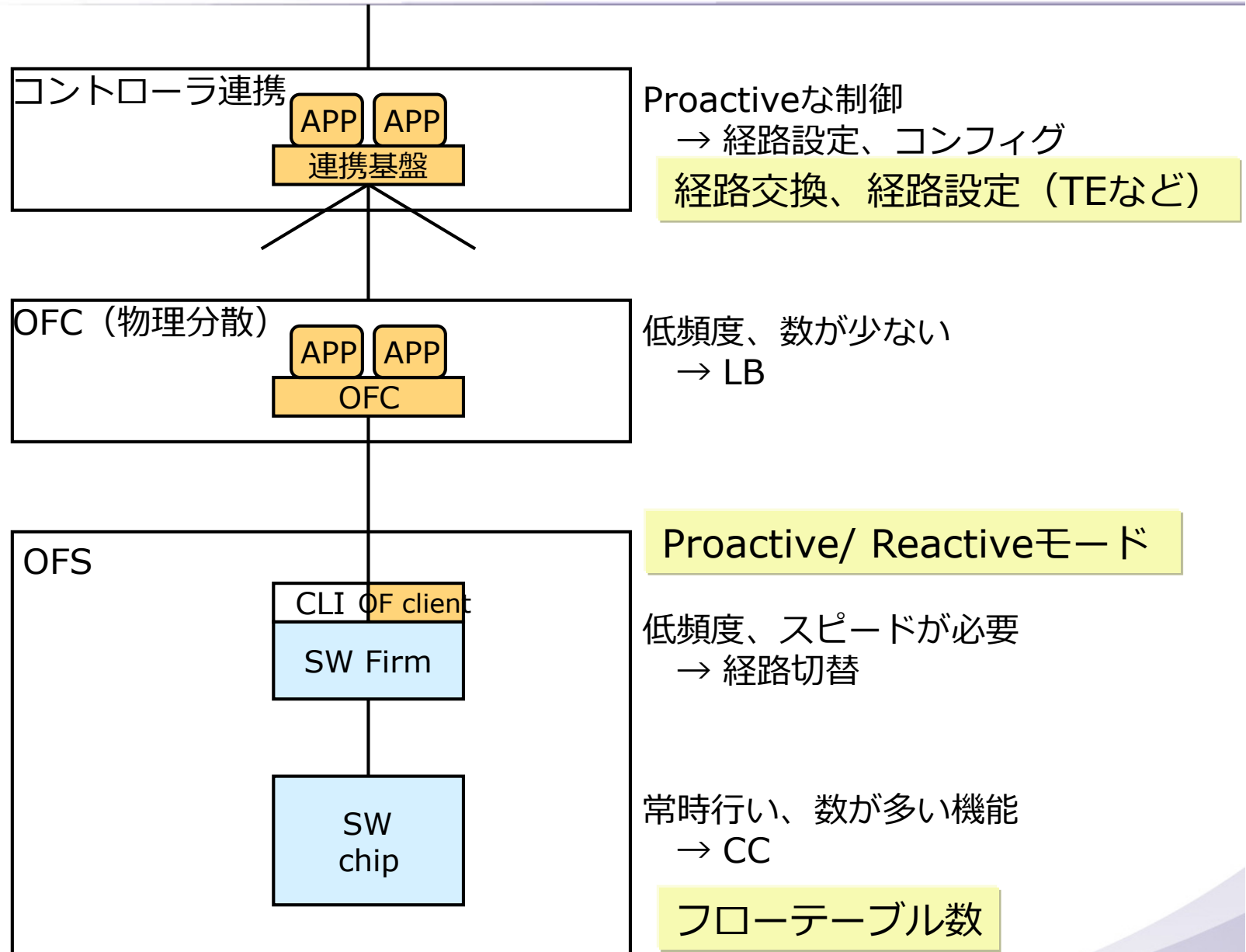
まとめ

- 実際に触ってみて分かった（改めて分かった）課題
 - Packet-In過多時の影響設定可能フロー数の制限
 - セキュアチャネル遅延による主信号への影響
 - Proactiveモード Reactiveモード
 - フローテーブル数
- WANへ適用する場合の課題
 - OpenFlowネットワークと既存ネットワークとの接続
 - OAM機能
 - TEをどうするか？

機能配備の考え方（案）

コントローラ設計

OAMなどの保守機能



- 固定観念にとらわれない
 - 誰もやったことがない
 - 何が正解か分からない
- 考え込むよりも試してみる
 - まずプロトタイプを実装して試してみる
 - 経験を積むと見えてくる地平線が変わってくる
- まずいところがあれば改良する
 - 場合によっては最初からやり直す
- オープンである事
 - スピード感のある技術開発

**皆さんと一緒にいろいろやっていくことで、SDN/
OpenFlowで出来ること、適用領域も
変わってくると思います！**

ご清聴ありがとうございました