FUJITSU
shaping tomorrow with you

Internet Week 2013

SDN再入門
＜WAN･キャリア編＞

富士通研究所
ネットワークシステム研究所
清水　翔

本日のアジェンダ

FUJITSU

- OpenFlowとは何か？
  - 基本的なコンセプトと機能
  - スイッチの制御方法
  - 規格の標準化と標準化団体
  - 規格の変遷
  - ユースケースおよび関連プロジェクト
- NFVとは何か？
  - 基本的なコンセプト
  - 登場の背景
  - アーキテクチャ概略
  - メリットと課題
  - ユースケース

自己紹介

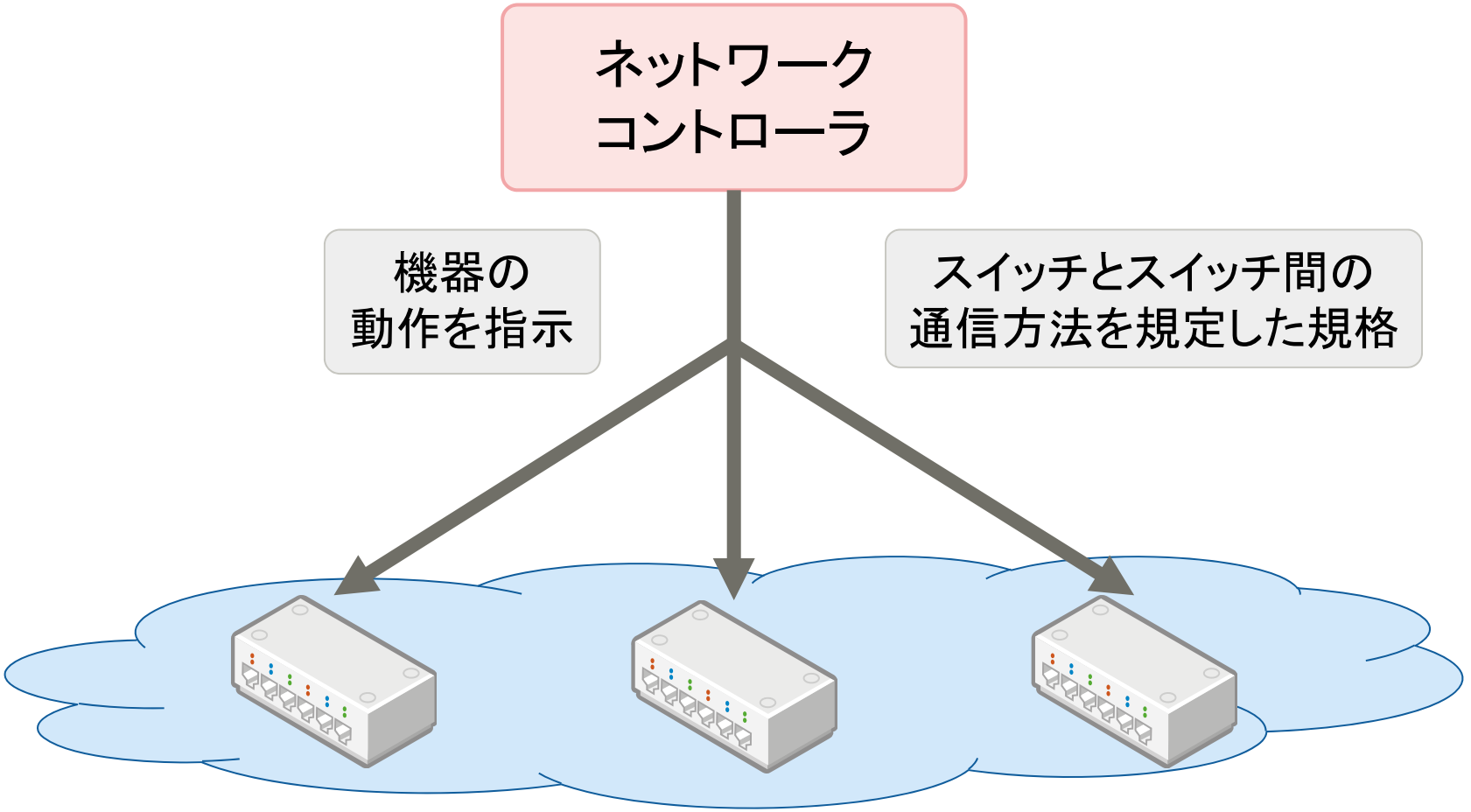
FUJITSU

- 名前
  - 清水　翔
- 経歴
  - 2010年3月　　慶應義塾大学大学院理工学研究科　博士課程修了
  - 2010年4月～　（株）富士通研究所　入社
- 研究テーマ
  - 大学：光ネットワークのトラフィックエンジニアリング、アーキテクチャなど
  - 会社：省電力ネットワーク（ルーティングアルゴリズム）→ SDN

OpenFlowとは？

FUJITSU

ネットワーク機器を制御するプロトコル



OpenFlowで可能な制御

FUJITSU

パケットの転送方法を指示できる

指示モデル

条件にマッチするパケットに対して
実行するアクションを指示する

例えば

条件	宛先MACアドレスがxであれば
アクション	ポート1に転送する

指定可能な条件 (OpenFlow 1.0)

FUJITSU

- ✓ 入力ポート番号

L2

- ✓ 送信元MACアドレス
- ✓ 宛先MACアドレス
- ✓ Ethernet Type
- ✓ VLAN ID
- ✓ VLAN Priority

L3

- ✓ 送信元IPアドレス
- ✓ 宛先IPアドレス
- ✓ プロトコル番号
- ✓ IP ToSビット値

L4

- ✓ 送信元ポート番号
- ✓ 宛先ポート番号

指定可能なアクション(OpenFlow 1.0)

FUJITSU

Forward	指定したポートにパケットを転送
Enqueue	キューに格納(QoS用途)
Drop	パケットを廃棄
Modify-Field	ヘッダの書き換え

- VLAN IDのPush/Pop/変更
- 送信元/宛先MACアドレスの変更
- 送信元/宛先IPアドレスの変更
- IP ToSビットの変更
- 送信元/宛先ポート番号の変更

レイヤを越えた制御

FUJITSU

スイッチ　　宛先MACアドレスをもとに転送

ルータ　　宛先IPアドレスをもとに転送

OpenFlow
- 送信元MACアドレスがxで
- 宛先IPアドレスがyで
- 宛先TCPポート80番のパケット

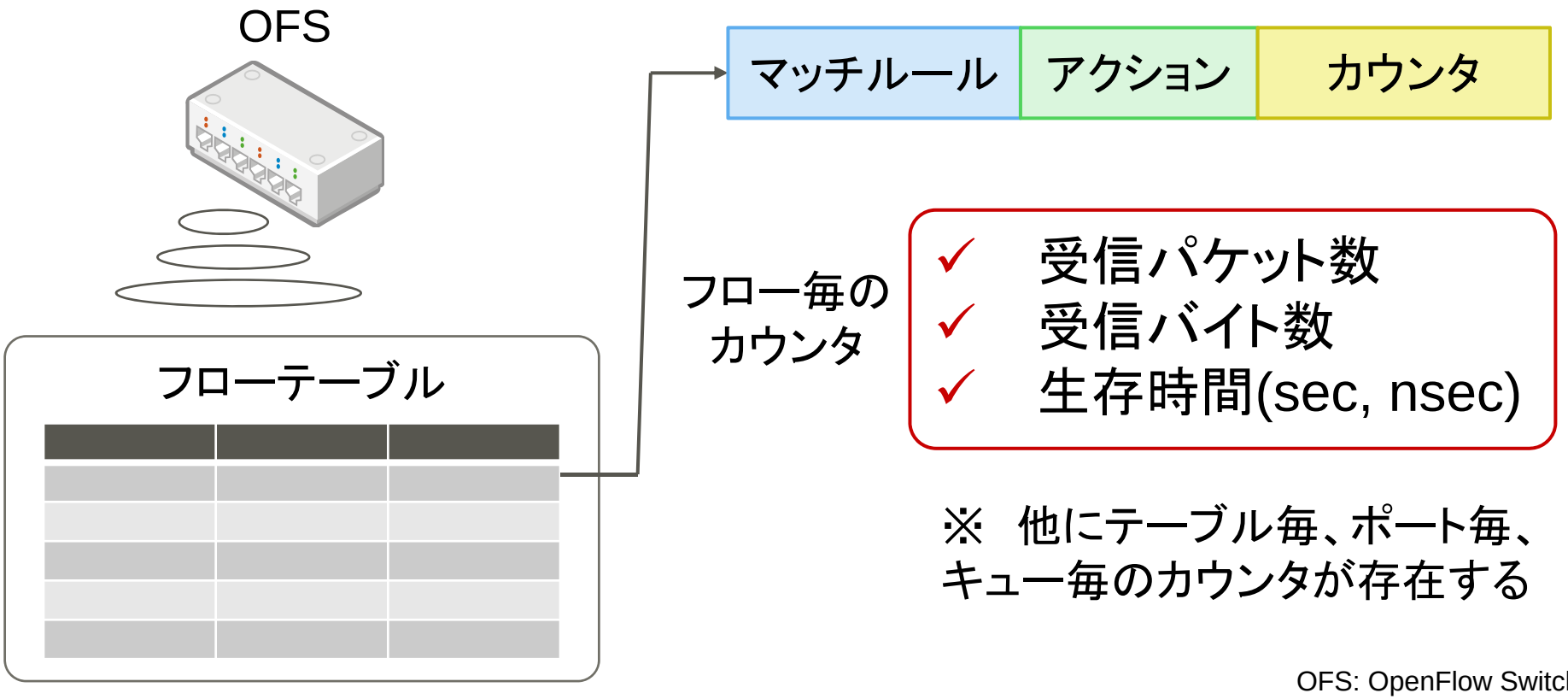
をポートzに転送

レイヤをまたがる転送ルールを決められる

フローエントリとその構造

FUJITSU

- フローエントリ
  - マッチルール、アクション、カウンタから構成される情報
- フローテーブル
  - フローエントリの集合体



OFS: OpenFlow Switch

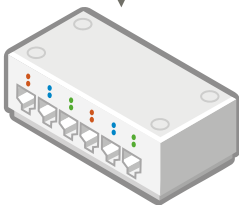
フローエントリの追加の仕組み

FUJITSU

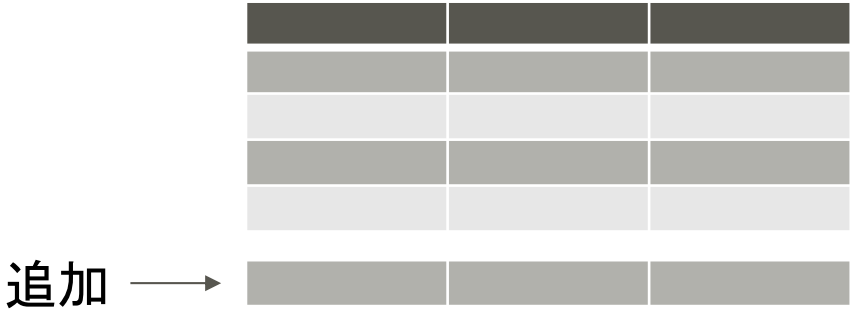
OpenFlow
コントローラ

FlowMod
メッセージ

OpenFlow
スイッチ



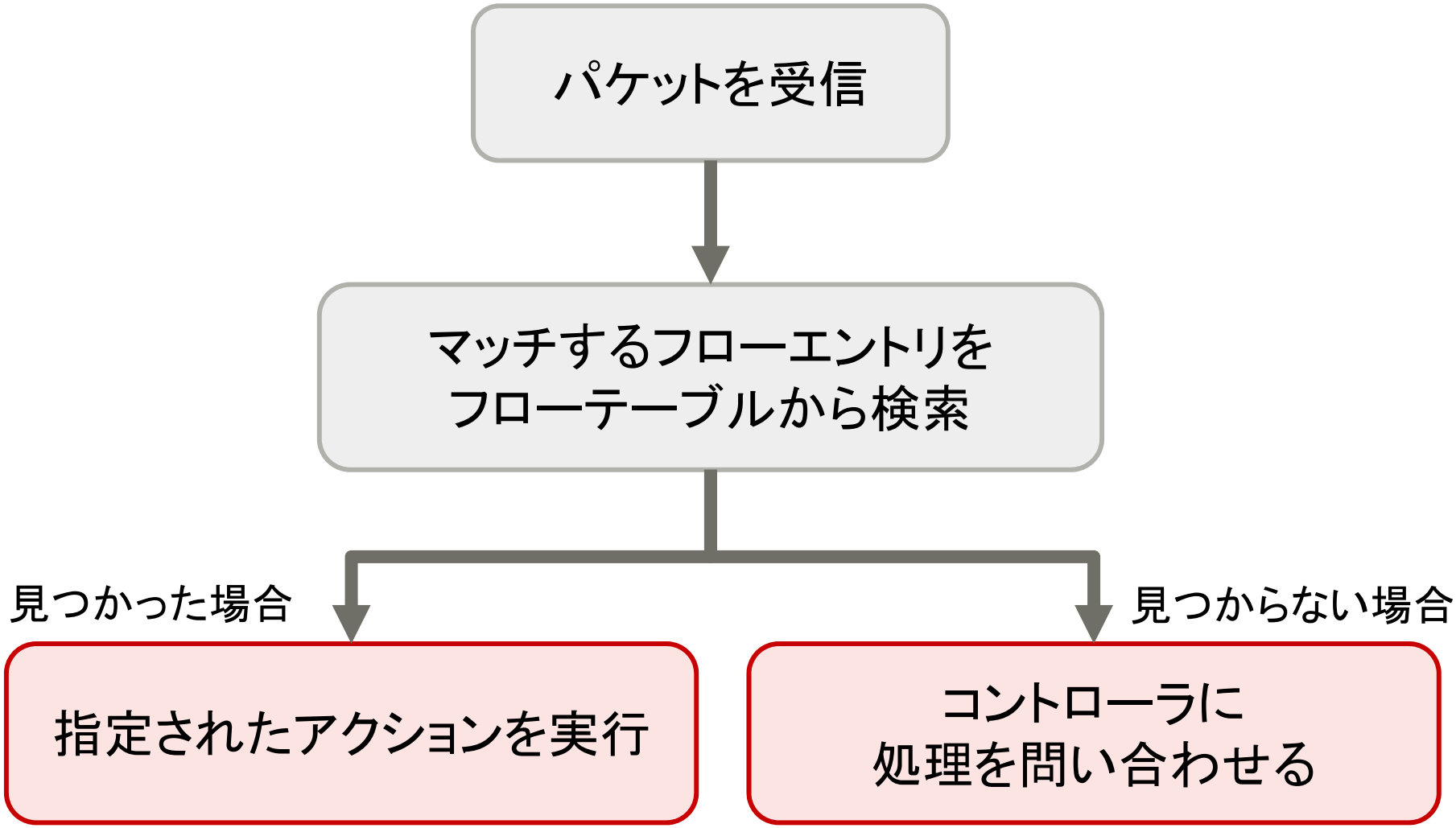
- マッチルール
- 優先度
- タイムアウト
  - Idle
  - Hard
- アクションリスト



パケット転送動作

FUJITSU

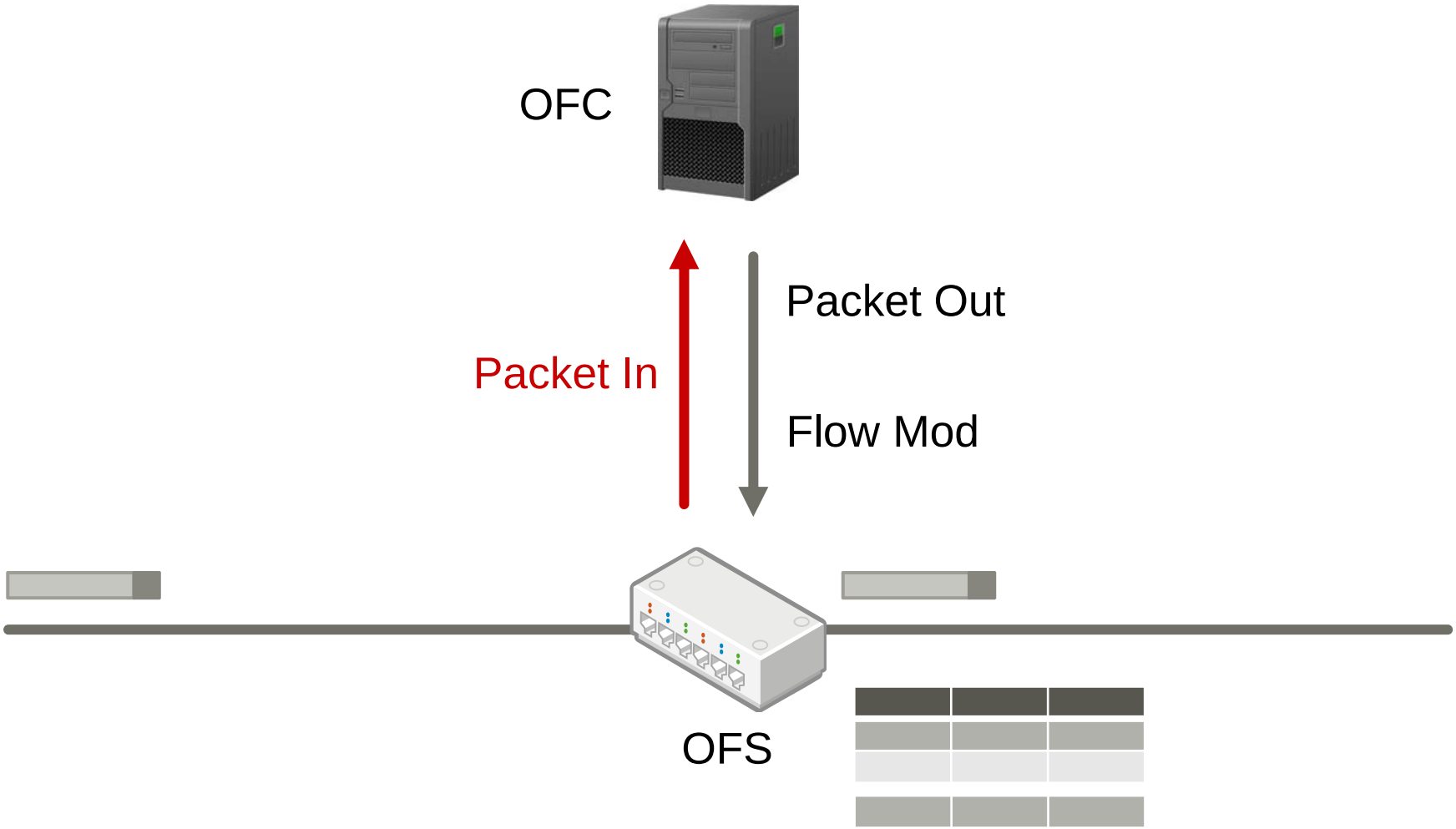
- OpenFlowスイッチは設定されたフローテーブルに従って動作する



マッチするエントリがない場合

FUJITSU

- スイッチがコントローラへPacket Inメッセージを発行

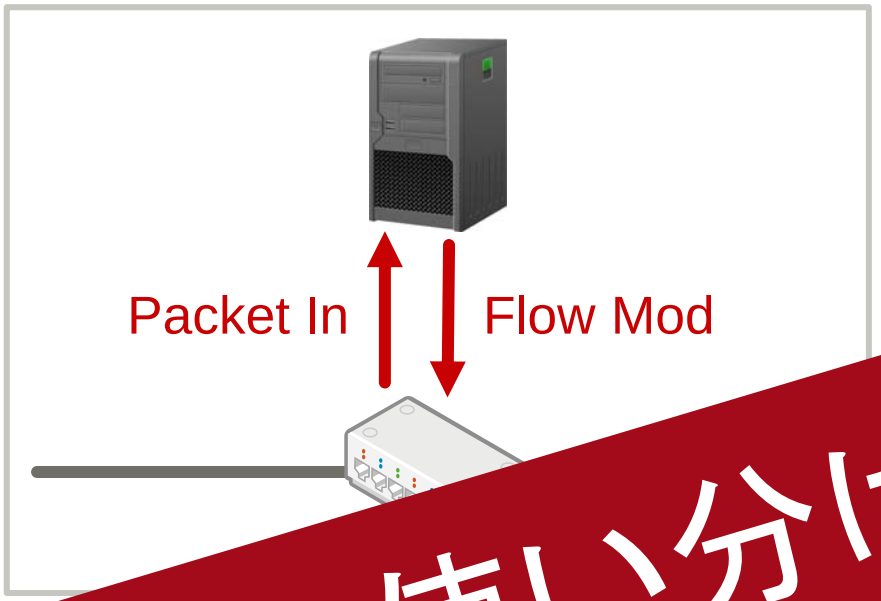


OFC: OpenFlow Controller　　OFS: OpenFlow Switch

Reactive方式とProactive方式

FUJITSU

Reactive方式



- にフローを設定

- ✓ 到着パケットに応じたダイナミックなフォワーディング制御が可能
- ✓ コントローラへの負荷が高い

Proactive方式

- コントローラから事前にスイッチに対してフローを設定

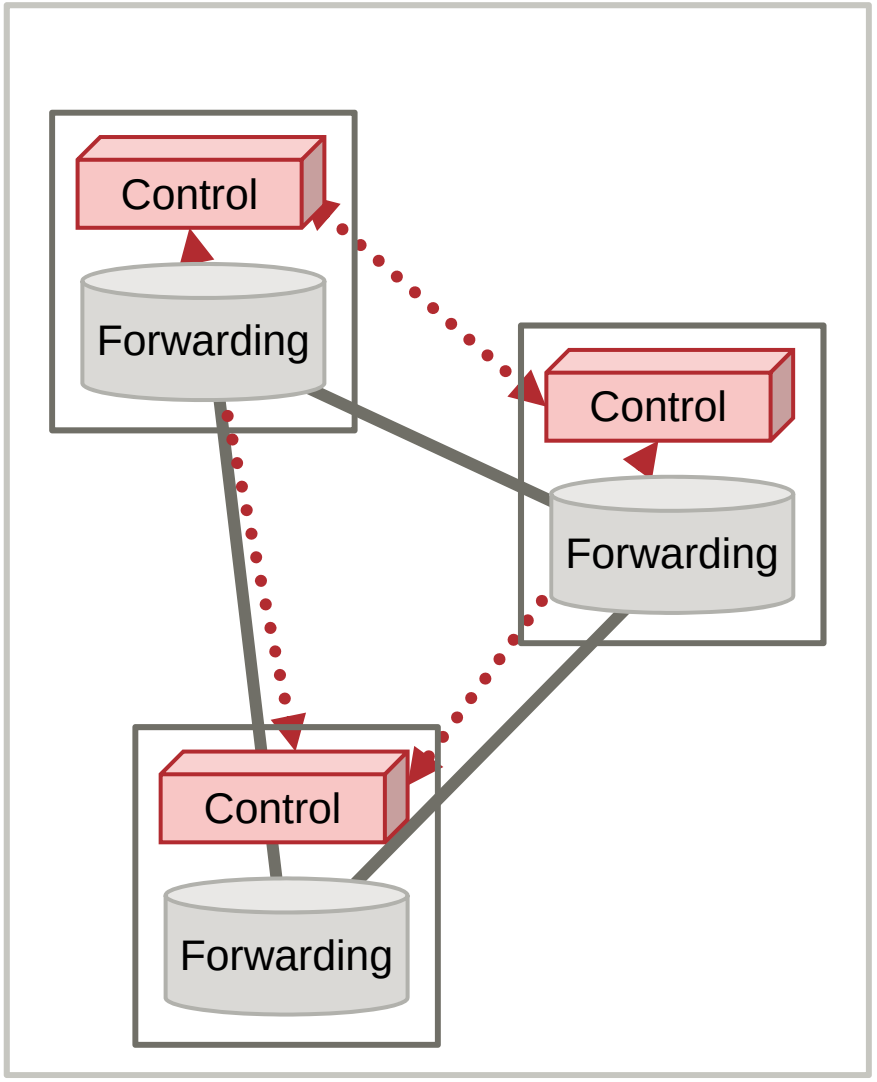
- ✓ 事前に決まったルールを用いて静的にフォワーディング
- ✓ コントローラへの負荷は低い

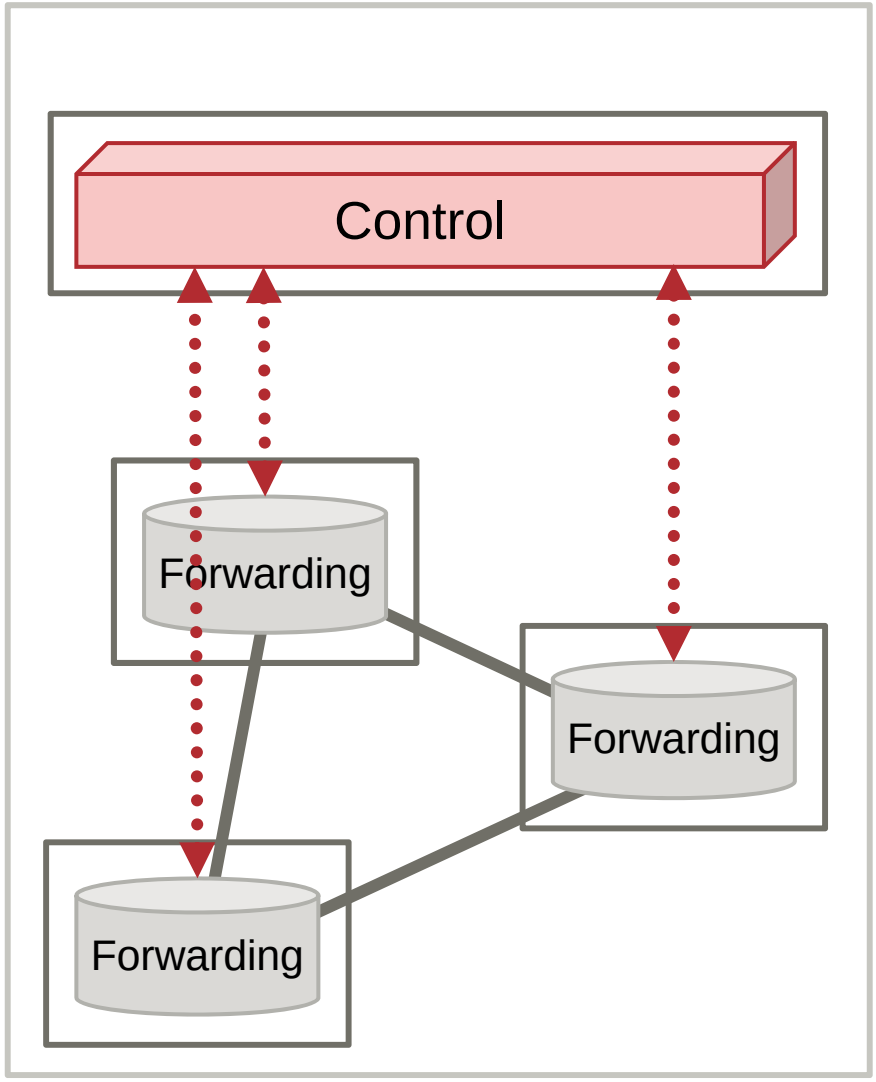
使い分けが重要

これまでとの比較 (1/2)

FUJITSU

■ 分散制御と集中制御

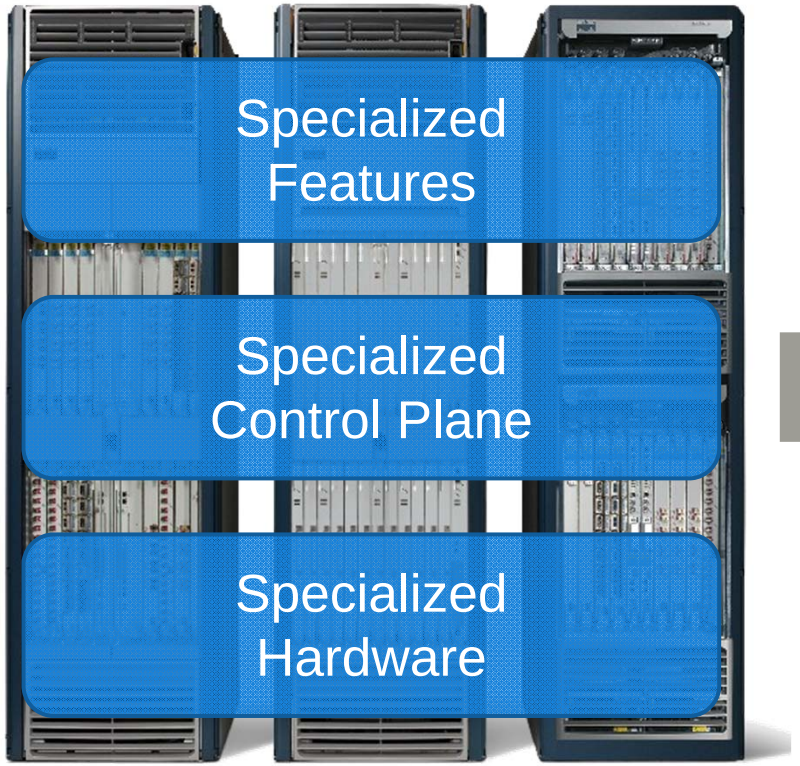


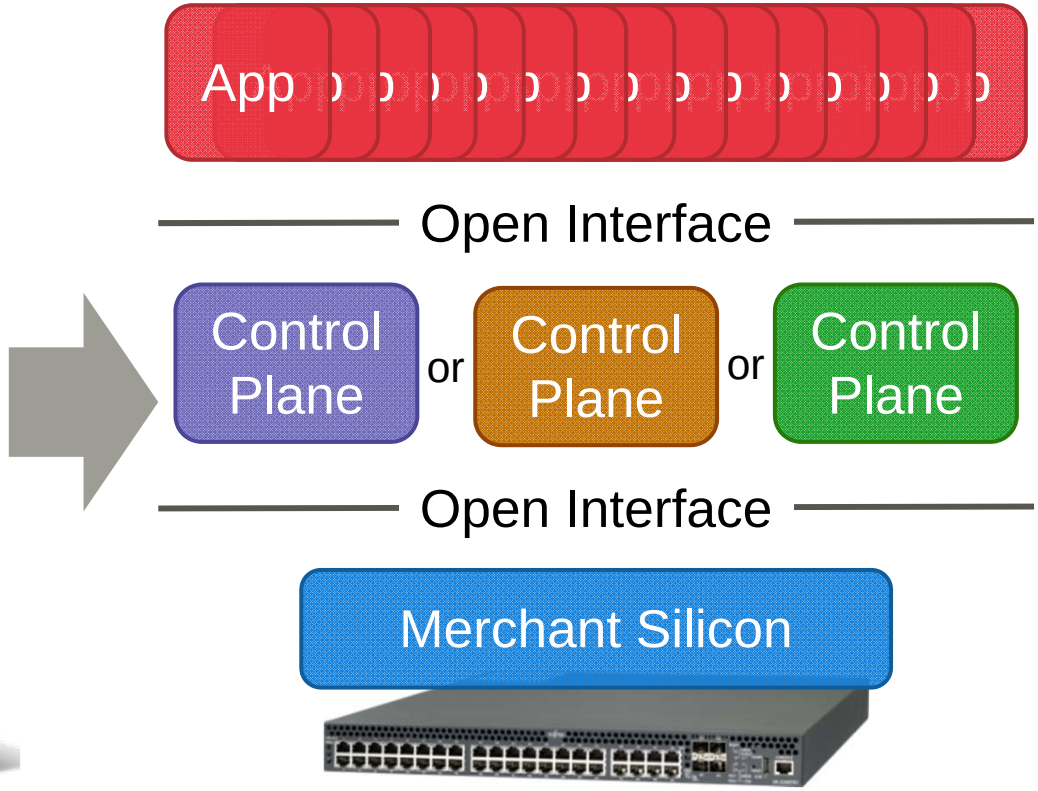


これまでとの比較 (2/2)

FUJITSU

■ 垂直統合と水平分業





✓ Closed
✓ Proprietary

✓ Open Standard
✓ Open Source

OpenFlowの標準化

FUJITSU

Ver. 1.1以前

OpenFlow Consortium

研究

STANFORD UNIVERSITY など

Ver. 1.2以降

Open Networking Foundation

商用

OPEN NETWORKING FOUNDATION

Deutsche Telekom　facebook　Goldman Sachs　Google

Microsoft　NTT Communications　verizon　YAHOO!

など
100社以上

Open Networking Foundation

FUJITSU

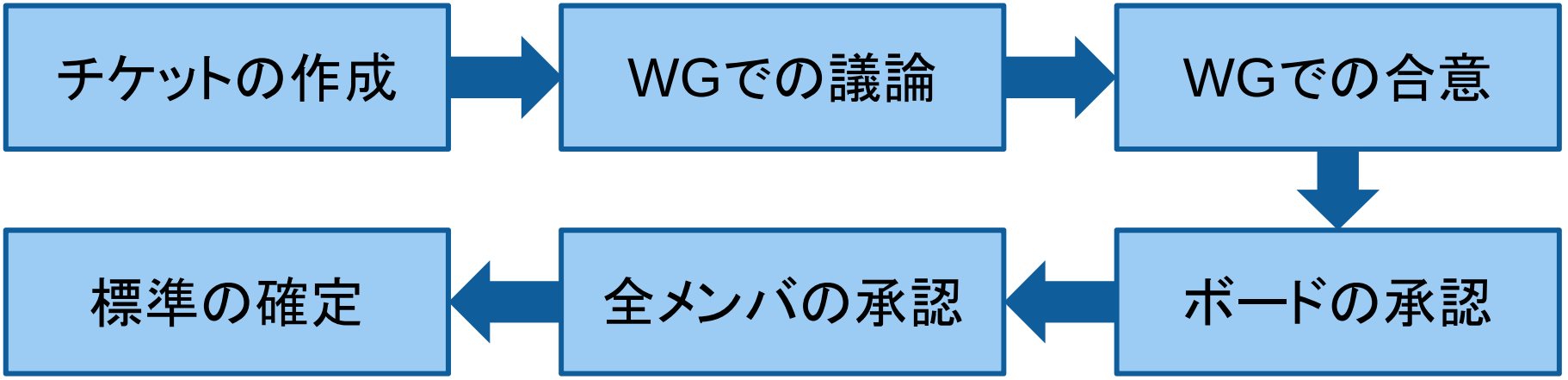
- 設立
  - 2011年3月21日
- 設立メンバ

- メンバ企業
  - 全リスト: http://sdndirectory.opennetworking.org/
  - 2013年11月11日時点で116社
  - キャリア、ベンダ、サービスプロバイダ、ユーザ、チップベンダなど多彩
  - メンバ企業間では自由にお互いの知的財産を使用することができる
- 活動
  - OpenFlowの規格標準化

ONFでの標準化プロセス

FUJITSU

- IETF, ITU-Uなどの標準化とは大分違う
  - 従来: ドラフト文書や寄書などの文書作成が重め
- アジャイル、オープンソース的な議論
  - プロジェクト管理ツールにチケットを切る
  - メーリングリストと電話会議で議論
- プロセス



WG: Working Group

OpenFlow仕様概要 (1/3)

FUJITSU

OpenFlow 1.0 2009年12月31日

- ✓ L2, L3 (IPv4), L4 (TCP/UDP)でのマッチング
- ✓ VLANのPush/Pop/変更
- ✓ シングルテーブル

OpenFlow 1.1 2011年2月28日

- ✓ Q in Q, MPLSラベルのマッチング
- ✓ MPLSラベルのPush/Pop/変更
- ✓ SCTPヘッダのマッチング、変更
- ✓ グループテーブル (Multicast/Flooding/LAG/Failover)
- ✓ マルチテーブル

OpenFlow仕様概要 (2/3)

FUJITSU

OpenFlow 1.2 2011年12月5日

- ✓ Extensible Match (OXM)への対応
- ✓ IPv6への対応（マッチング、変更）
- ✓ コントローラの冗長化機構 (Role Message)

OpenFlow 1.3 2012年4月13日

- ✓ 補助接続(Auxiliary Connection)のサポート
- ✓ IPv6拡張ヘッダへの対応
- ✓ Per flow meter（帯域制限などに用いる）
- ✓ PBB (Provider Backbone Bridge)への対応
- ✓ マルチパートメッセージ（統計情報取得方法の拡張）
- ✓ トンネルIDへの対応

OpenFlow仕様概要 (3/3)

FUJITSU

OpenFlow 1.4 2013年8月

- ✓ TLV形式による拡張可能なメッセージが増加
- ✓ 光ネットワーク用のポート属性の追加
- ✓ Packet Inの発生理由の詳細化
- ✓ フローモニタリング（複数コントローラの協調動作用）
- ✓ Roleの変更イベントの通知
- ✓ コントローラのデフォルトの待ち受けポートの変更
  - ✓ 6633から6653へ
- ✓ etc …

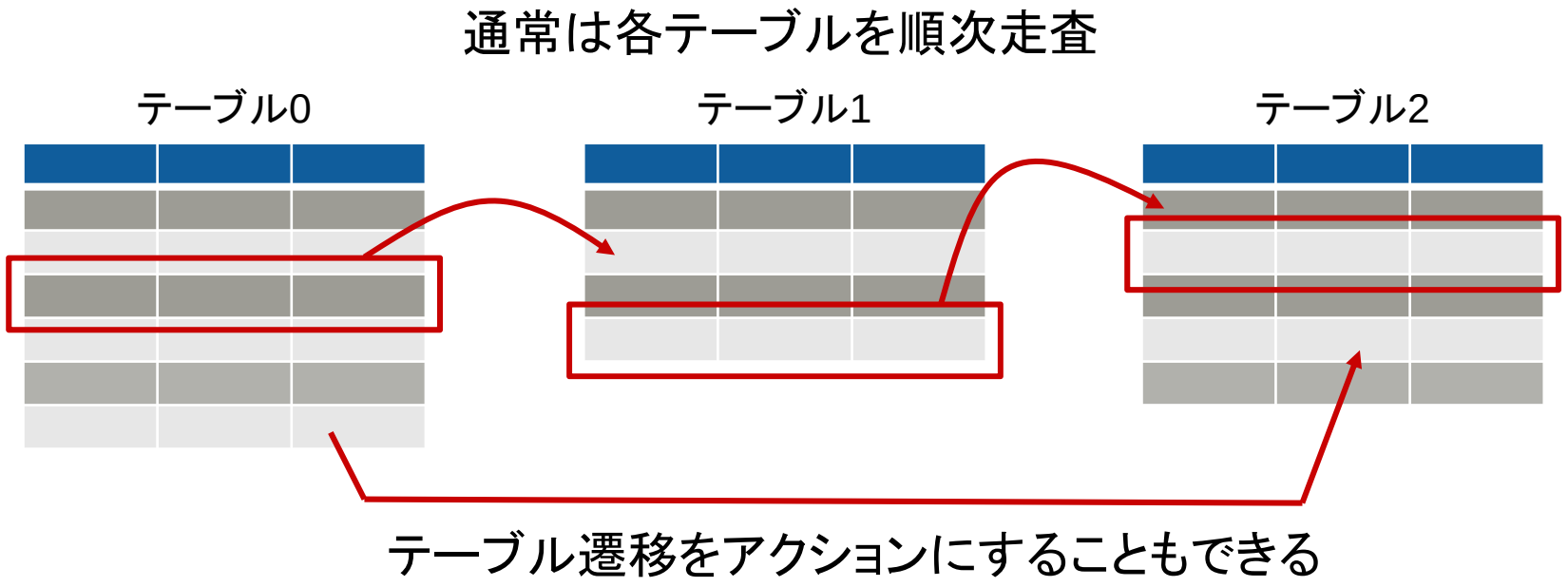
OpenFlow 1.5 これから標準化議論が開始される

標準化のスピードと規格の進化

FUJITSU

- IETFやITU-Tの標準化のスピードと比べて非常に早い
  - 約1年の間にVer. 1.1からVer. 1.3までバージョンアップ(3規格分)
- あまりに早い規格の変更によって実装が追いつかない事態
  - ハードウェア実装はもちろんのことソフトウェア実装も追従しきれない
  - リファレンス実装が存在せずに、規格のバージョンアップがどんどん進む状態
- 規格の飛躍
  - マルチテーブル: Ver. 1.1 → 実装の難しさ
  - Extensible Match (OXM): Ver. 1.2 → メッセージの大幅な変更
- Ver. 1.3で一旦規格のバージョンアップをフリーズ
  - 実装が進むのを待つ形に
- マイナーバージョンの登場
  - 仕様書の分かりにくい点やバグの修正
    - Ver. 1.0系列 (Ver. 1.0.x)
    - Ver. 1.3系列 (Ver. 1.3.x)

マルチテーブル (Ver. 1.1で導入)



- 複数のテーブルを用いた多段のマッチング処理が可能になった
  - テーブル0: ポートでのマッチング → VLANのPush
  - テーブル1: VLANでのマッチング → MPLSラベルのPush
  - テーブル2: MPLSラベルのマッチング → 出力ポートを決定
- ハードウェア実装の難易度が上がり、Ver. 1.1対応の実装がなかなか出てこない原因になった

OpenFlow 1.1までのMatch構造体

FUJITSU

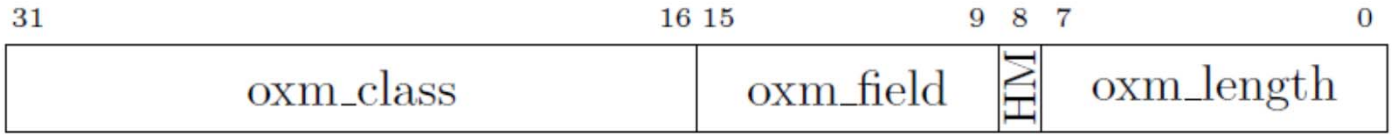
対応フィールドによる
互換性の問題

対応するフィールドが構造体にべた書き

OXM: OpenFlow Extensible Match

FUJITSU

- 対応するマッチフィールドに変更があってもメッセージ構造が変わらないような拡張可能なフォーマットについての議論が行われた
- いろいろな議論の末、TLV (Type-Length-Value)形式のフォーマットを採用
  - Open vSwitchに実装されていたNXM (Nicira Extensible Match)を事実上標準規格として追認



Name		Width	Usage
oxm_type	oxm_class	16	Match class: member class or reserved class
	oxm_field	7	Match field within the class
	oxm_hasmask	1	Set if OXM include a bitmask in payload
	oxm_length	8	Length of OXM payload

- ドラスティックなメッセージ構造の変化のため、Ver 1.2に対応する実装が出てこない要因となった（Ver. 1.3との間隔も短かった）

Source: OpenFlow Switch Specification (Version 1.4.0)

OpenFlowだけでは機能不足

FUJITSU

OpenFlow

スイッチ外部からフォワーディングの指示を
柔軟に行うための仕組み（＝プロトコル）

スイッチやポートの設定はスコープ外

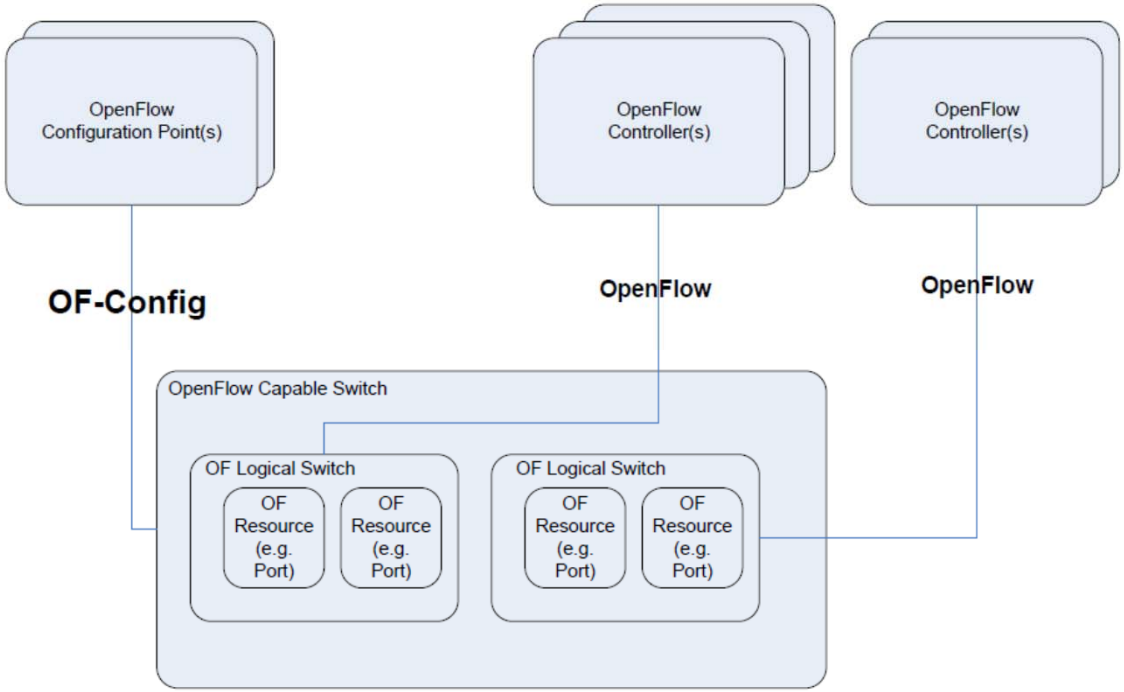
OF-Config

OF-Config

FUJITSU

スイッチの設定を行う仕組み

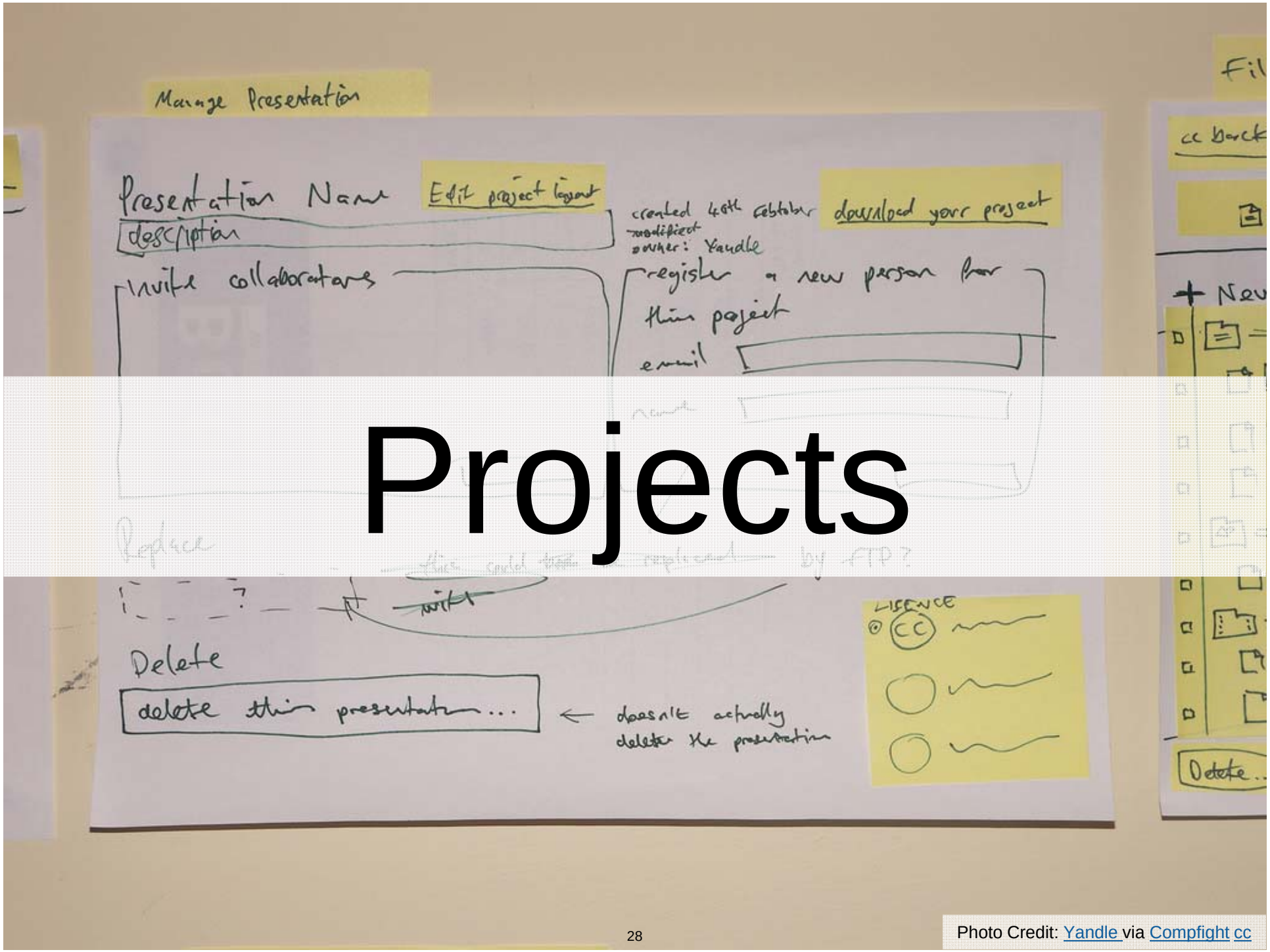
NETCONF + YANGでコンフィグを設定



- ✓ スイッチとコントローラの接続
- ✓ ポートの追加/削除
- ✓ Dapapath IDの設定
- ✓ ポートのUp/Down
- ✓ キューの設定
- ✓ etc …

Source: OpenFlow Management and Configuration Protocol (OF-Config 1.1.1)

ユースケース



Projects

OpenDaylight

FUJITSU

OpenDaylight

FUJITSU

PLATINUM MEMBERS

GOLD MEMBERS

SILVER MEMBERS

OpenDaylight

FUJITSU

- Linux Foundationがサポートするコントローラのプロジェクト
- License: Eclipse Public License Ver. 1.0 (EPL)
- 12月に最初のリリースを予定（コードネーム: Hydrogen）
- アーキテクチャ
  - Javaで記述
  - OSGiを用いたモジュール機構(Beaconと類似)
  - スイッチを制御するプロトコル(Southbound API)はプラガブル
    - オープンソース版ではOpenFlow (当初は1.0、IBMらの貢献によって1.3対応)
    - CiscoはONE Controller (XNC?)のコードの一部を提供していると思われる
- Cisco vs. Big Switch
  - Big SwitchはFloodlightという同じくJavaのOpenFlowコントローラを開発中
  - 紆余曲折の結果、コントローラのコア部分はCisco寄贈のコードを、アプリケーションはBig Switch寄贈のコードを採用することになった
  - Big Switchはプロジェクトの関与を大幅に縮小 (Platinum Memberを辞めた)

O3

FUJITSU

Photo Credit: flickrolf via Compfight cc

FUJITSU

NFV:
Network Functions Virtualisation

NFV = Network Functions Virtualisation

FUJITSU

- 広義のNFV
  - ファイアーウォール、ロードバランサ、VPN装置など従来専用のハードウェアアプライアンス製品が提供してきた機能を汎用サーバ上のソフトウェアとして実現すること
- 狭義のNFV
  - ETSI (European Telecommunication Standards Institution: 欧州電気通信標準化機構)のNFV ISG(Industry Specification Group)で議論が行われているネットワーク機能の仮想化技術の標準仕様
  - (ヨーロッパの)通信事業者が議論を主導
  - 2013年11月11日時点で5本の仕様書とホワイトペーパを公開
    http://portal.etsi.org/NFV/NFV_White_Paper.pdf
    http://www.etsi.org/nfv

通信事業者は多数を機器を利用

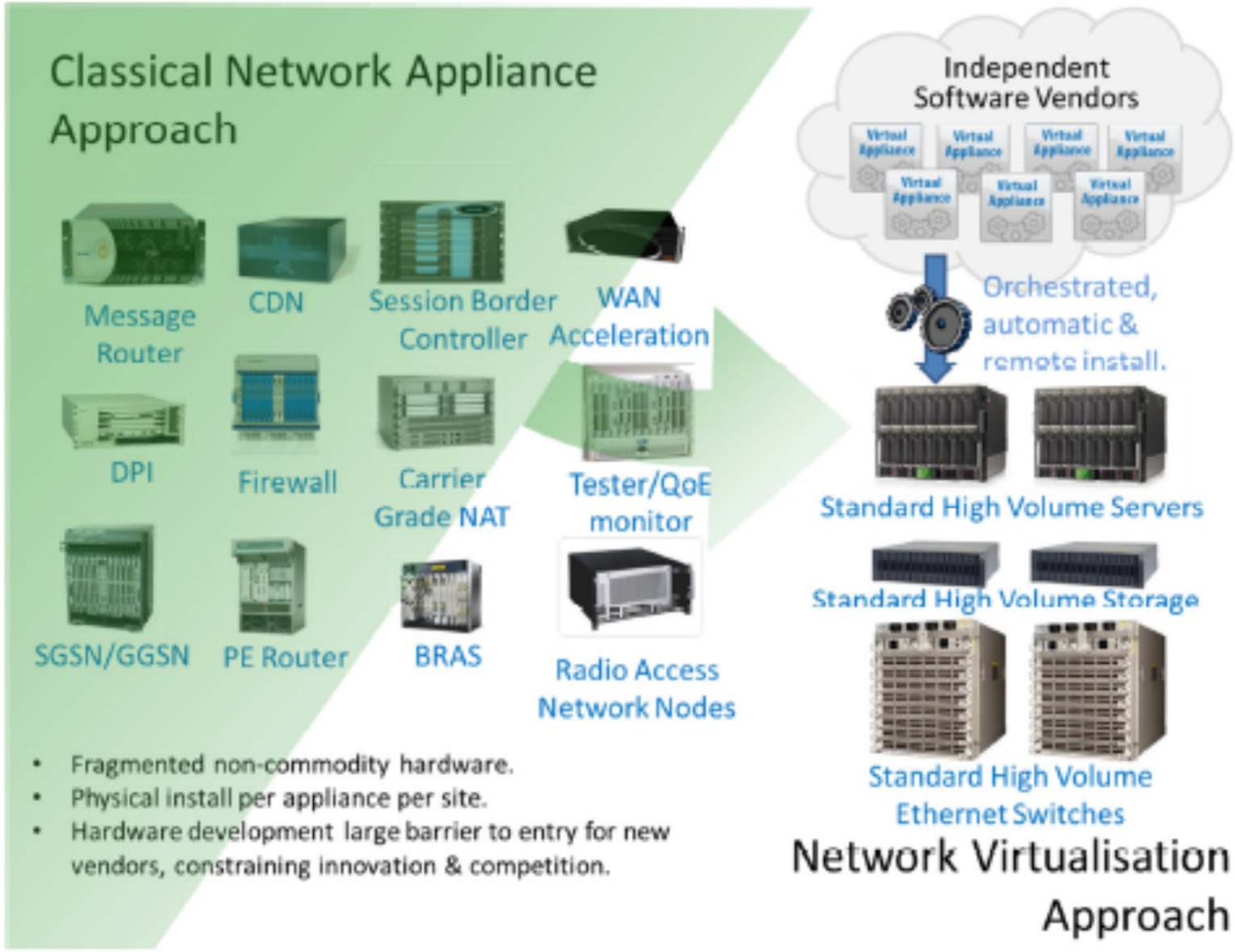


Figure 1: Vision for Network Functions Virtualisation

Source: http://portal.etsi.org/NFV/NFV_White_Paper.pdf

通信事業者の悩みとNFVのメリット

FUJITSU

- 機能毎に専用のハードウェアアプライアンスが必要
  - 取り扱いハードウェアの増加
    - いろいろなタイプの技術者が必要
    - 故障に備えて予備を確保する手間
- ベンダーロックイン
  - 機能によって強いベンダーが違う
  - 進化のスピードが遅く、コスト高になりがち

Airbus A320

- 汎用サーバ(e.g. IAサーバ)によって設備を統一ができる
  - 設備調達コストおよび保守コストの低減
- ベンダーロックインの回避
  - 汎用サーバの最新の技術進化のメリットを享受できる

例えば独自仕様プロセッサ（電電公社仕様？）

FUJITSU

アカウント作成　ログイン

ページ　ノート　閲覧　編集　履歴表示　検索

ウィキペディア
フリー百科事典

メインページ
コミュニティ・ポータル
最近の出来事
新しいページ
最近の更新
おまかせ表示

D60形デジタル交換機

D60形デジタル交換機（デーろくまるがたデジタルこうかんき）とは、電電公社と交換機メーカー4社（日本電気・富士通・日立製作所・沖電気）が共同開発した、中継系デジタル交換機である。

1982年（昭和57年）に東京の大手町局でサービスを開始。その後、市外系のディジタル化に向けて中継交換機として普及した。

その後、改D60形[illegible]で中継系および加入者階梯系交換機としていまだに広く使われている。

独自仕様プロセッサ?

ハードウェア [編集]

複数の32ビットプロセッサ、交換機独特の付帯装置、および様々なコミュニケーションするための端末から構成される。 なお、交換機自体は架（が）と呼ばれるラックに収められた各プロセッサや、付帯装置で構成されている。

プロセッサ [編集]

D60は電話交換機に特化した命令セットを持つ独自仕様プロセッサである。時代的にIBMの大型汎用機の影響を強く受けている。1台のD60形交換機は、少なくとも2つのプロセッサを持つ（後述）マルチプロセッサマシンである。通常の運用においては8つないしそれ以上のプロセッサを持つ事が多い。各プロセッサは役割により異なる名称が付けられるが、ハードウェア的にはほとんど同じものであり、走行するソフトウェアの違いにより、異なる機能を発現する。

近年のプロセッサ事情

FUJITSU

- プロセスの微細化が進み投資額が増大
  - 数の見込めるプロセッサでなければ最新プロセスを使えない
  - ネットワーク向け専用ASICなどは最新プロセスのメリットを享受できない？
- アーキテクチャ的にはIAとARMの2台巨頭という潮流
  - ソフトウェア開発を考えても独自プロセッサはハードルが高くなる

汎用プロセッサによるネットワーク処理

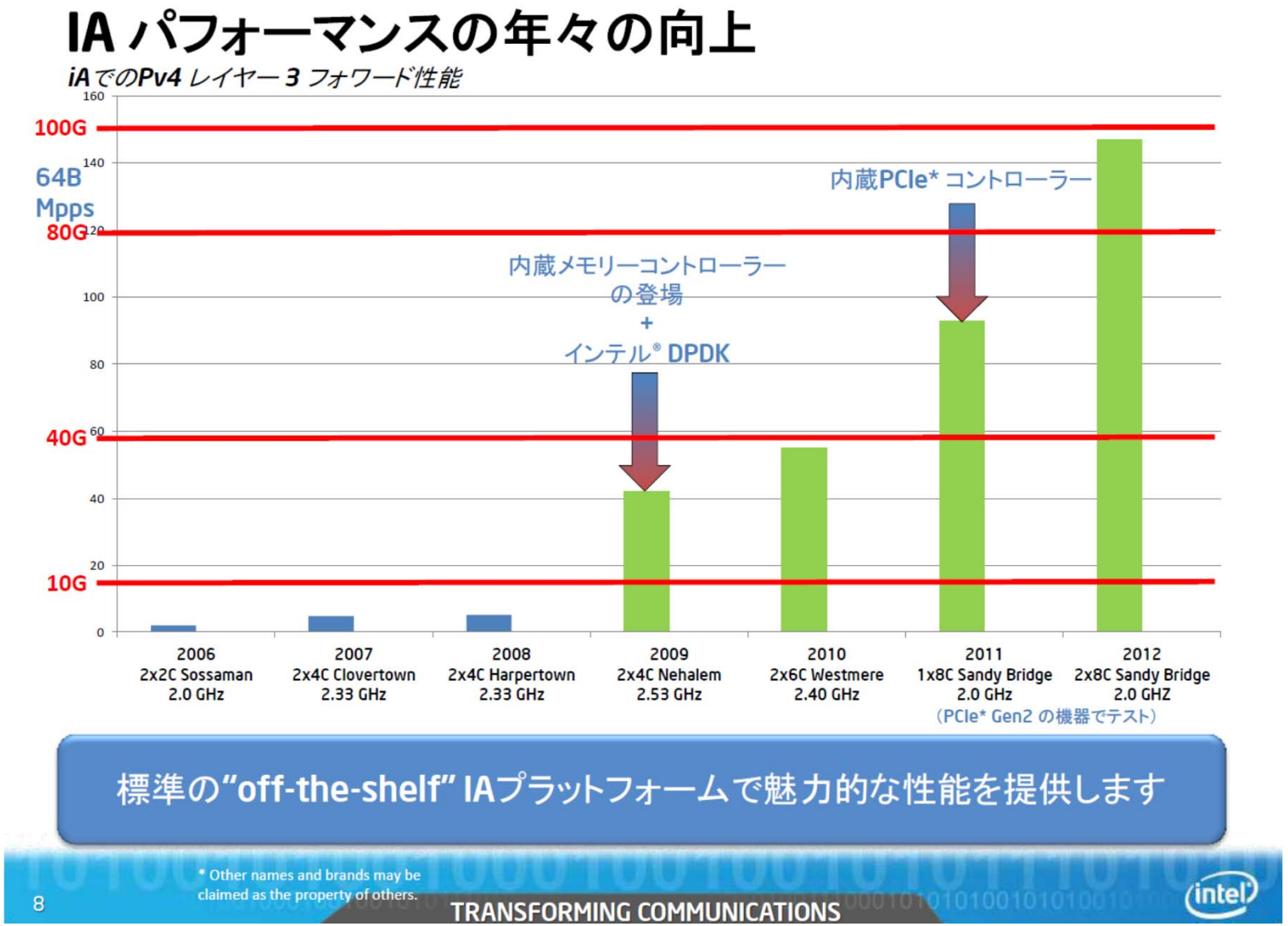
FUJITSU

- IAプロセッサの性能向上
  - マルチコア
  - 内蔵メモリコントローラ
  - 内蔵PCIeコントローラ
- パケット処理に最適化されたライブラリの登場

- Intel DPDK (Data Plane Development Kit)
  - BSDライセンス
  - メモリマネージャ、バッファマネージャ、フロー分類、ドライバなど

- 汎用プロセッサでも高速なパケット処理が出来る環境に

Intel DPDKのパフォーマンス

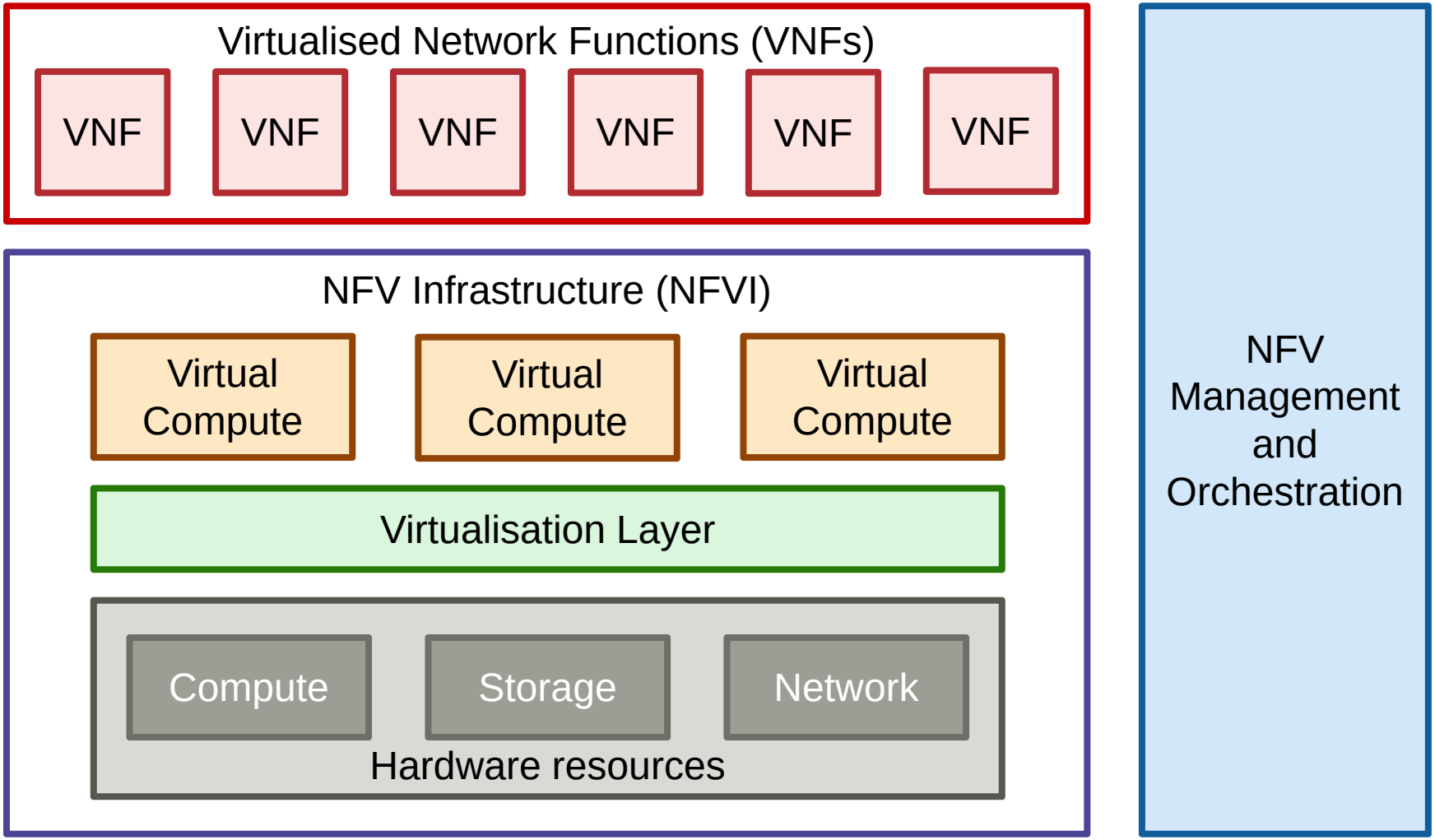


Source: https://www.nic.ad.jp/ja/materials/iw/2012/proceedings/d1/d1-Kohmura.pdf

ざっくり説明すると

FUJITSU

- VM上のソフトウェアで機能を実現すればメリット多いのでは？



Ref: Network Functions Virtualisation (NFV); Architectural Framework

メリットと課題

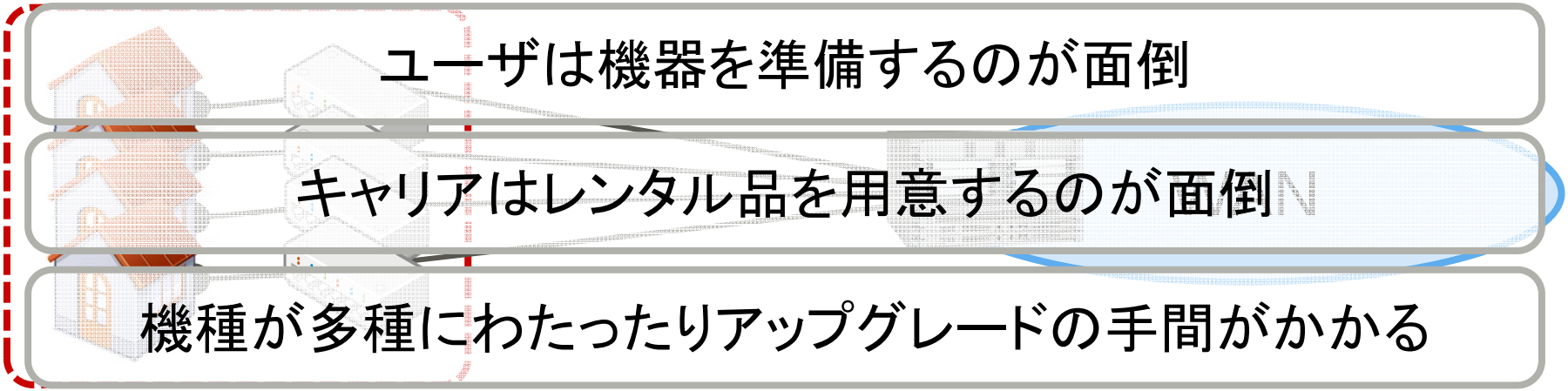
FUJITSU

- メリット
  - 設備コストの低減と省電力化
  - Time-to-Marketの短縮
  - ハードウェアの統一化と技術者の確保の容易化
  - オープンなエコシステムの構築によるイノベーションの創出
  - 耐障害性の向上
- 課題
  - 相互運用性（ハイパーバイザ、VM、データセンターの違いなど）
  - ソフトウェア処理による性能低下
  - 既存機器との共存や互換性、マイグレーション
  - 異なるベンダのVNF同士のインテグレーション

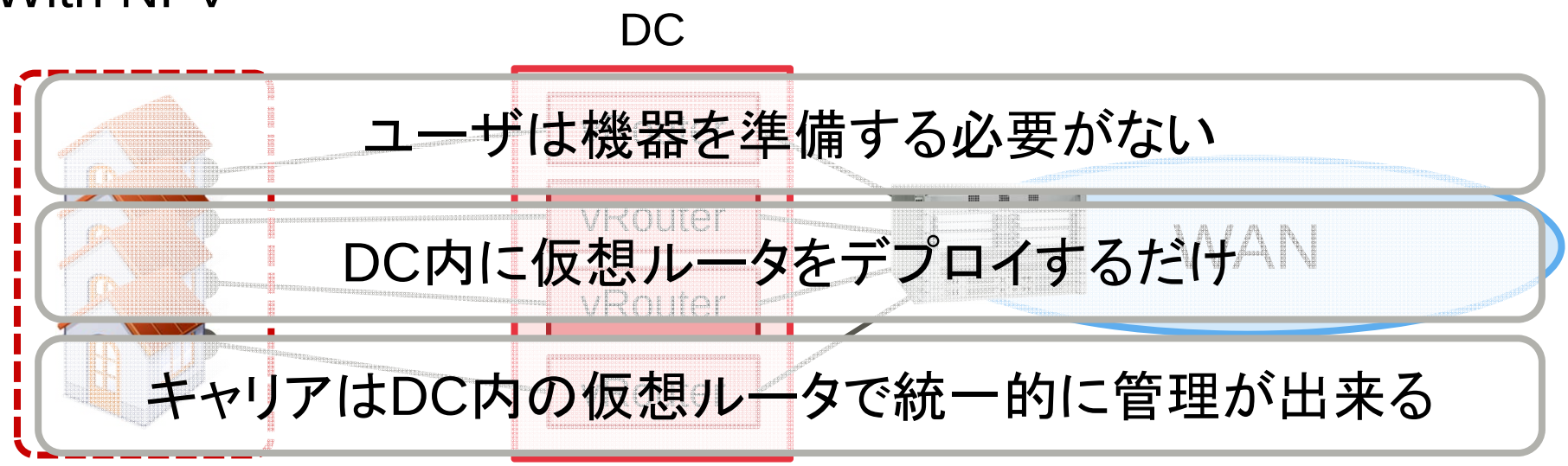
ユーザのルータをデータセンタ側へ

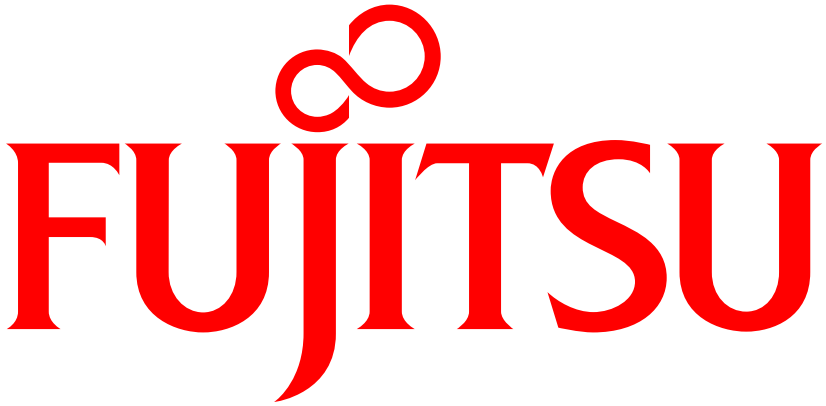
FUJITSU

Without NFV



With NFV





shaping tomorrow with you