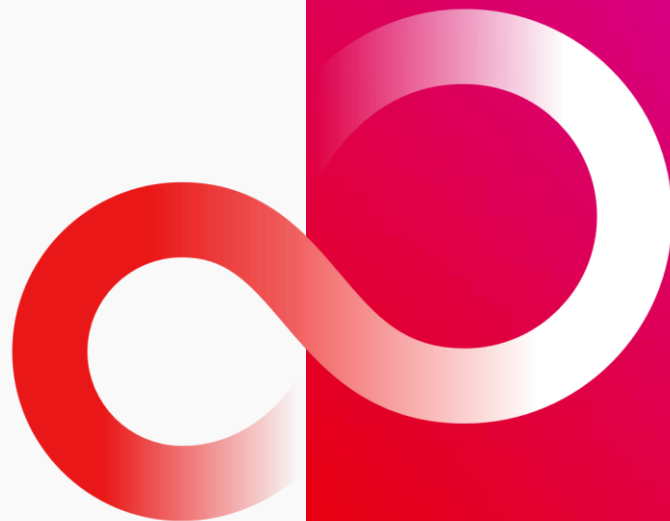


新たなICT基盤「DCI」における データ処理・通信の制御のための K8sコントローラ開発の取り組み

2024年10月11日

富士通株式会社

阿部 仁彦



- **名前**：阿部 仁彦（あべ きみひこ）
- **所属**：富士通株式会社 富士通研究所
先端技術開発本部 6Gプラットフォーム統括部 技術部
- **業務経歴**：
 - 製造業向けデータ処理基盤 & アプリの開発・運用（4年）
 - ローカル5Gを活用したデータ処理基盤の実証PJ（1年）
 - 次世代ICT基盤の企画・研究開発（3年）
- **現在の業務内容**：
 - 次世代ICT基盤におけるコントローラの研究開発
 - ネットワーク設計検証技術の研究開発



- はじめに - 2030年に向けた社会課題の想定
- DCIの研究開発の取り組み
- DCIの特徴
- DCIの機能構成
- K8sとコントローラについて
- K8sコントローラ開発の取り組み
- デモ：映像データに対する人物検知(GPU/FPGA利用)
- 実用化に向けた今後の展望

Beyond 5G時代に向けた3つの課題を想定

1.データ収集/
活用の拡大

動画の高精細化、データの3次元化、IoT化の進展

2.環境問題への対応

SDGs、カーボンニュートラルなど環境負荷低減

3.DX化進展/
ITサービス高度化

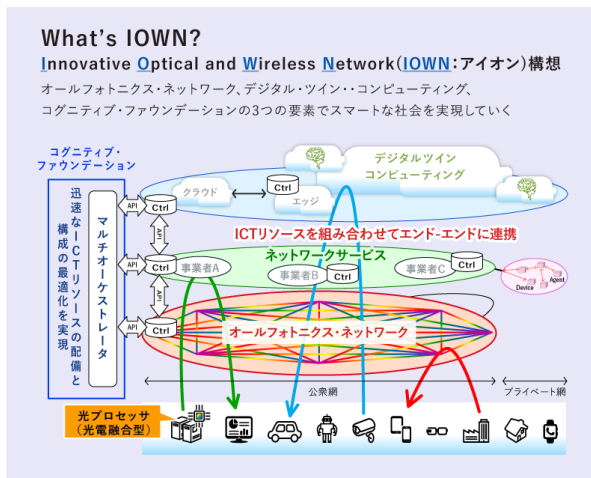
ロボット、自動運転、メタバースなどへの対応のための高度な性能要件



既存の汎用CPU中心のアーキテクチャの延長では、このような課題への対応が困難。
これらの変化に対応するためには新たなアーキテクチャと技術が必要

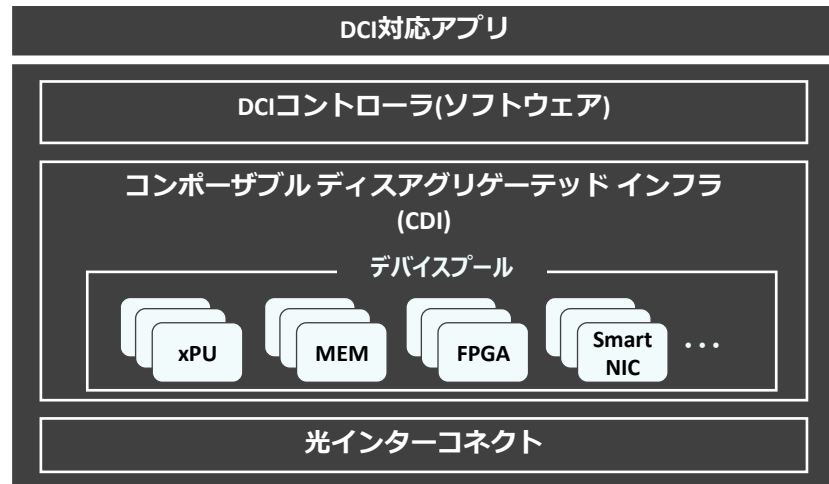
- 富士通は、2030年に向けた社会課題への対応のため、IOWN Global Forumの活動の中で、**NTT社と共同でDCI（Data Centric Infrastructure）の研究開発**を行っている
- **DCIはBeyond 5G時代に求められる高い電力性能比・低遅延を実現する次世代のICT基盤。**2030年に向けて段階的な実用化を目指している

IOWN構想とは？

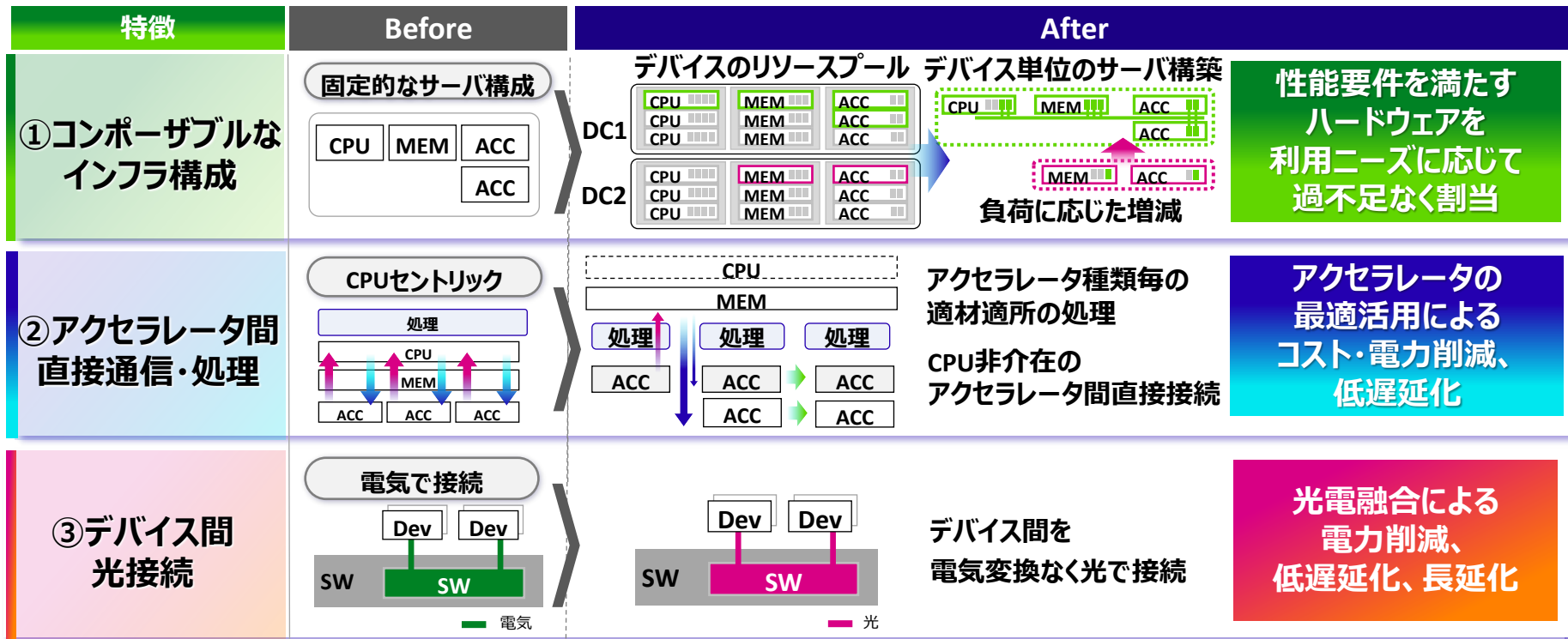


【出典】 <https://www.rd.ntt/iown/>

DCIの概要



以下の3つの特徴によって、高い電力性能比・低遅延を実現することを目指す



DCI対応アプリ

映像AIアプリ

...

... (etc)

対応アプリは順次拡張予定
※本日は映像AIアプリを対象にデモ実施

DCIコントローラ(ソフトウェア)

Kubernetes(略称K8s)

※これまでの試作開発で活用



DCI上でのデータ処理・通信を制御
K8sとそのコントローラを活用した試作開発を実施
➔ 本日はこちらの取り組み内容についてご紹介

コンポーザブル ディスアグリゲータッドインフラ(CDI)

CDI管理ソフト

PCIe Fabric Switch

デバイスプール



...

ディスアグリゲータッドコンピューティング技術により
コンポーザブルにサーバを構成

光インターコネクト

光エンジン

Switch ASIC


光電融合技術によりデバイス間を光で接続

DCI専用機能

汎用機能

K8sおよびコントローラの以下の特徴に着目し、DCIコントローラ開発に活用

K8sとは

- コンテナオーケストレーションの分野でデファクトスタンダードとなっているソフトウェア 

- 宣言型APIによってリソースを管理

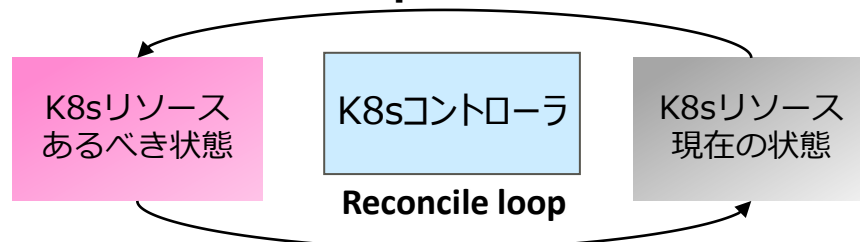


- データ処理/AI基盤としても活用



K8sのコントローラとは

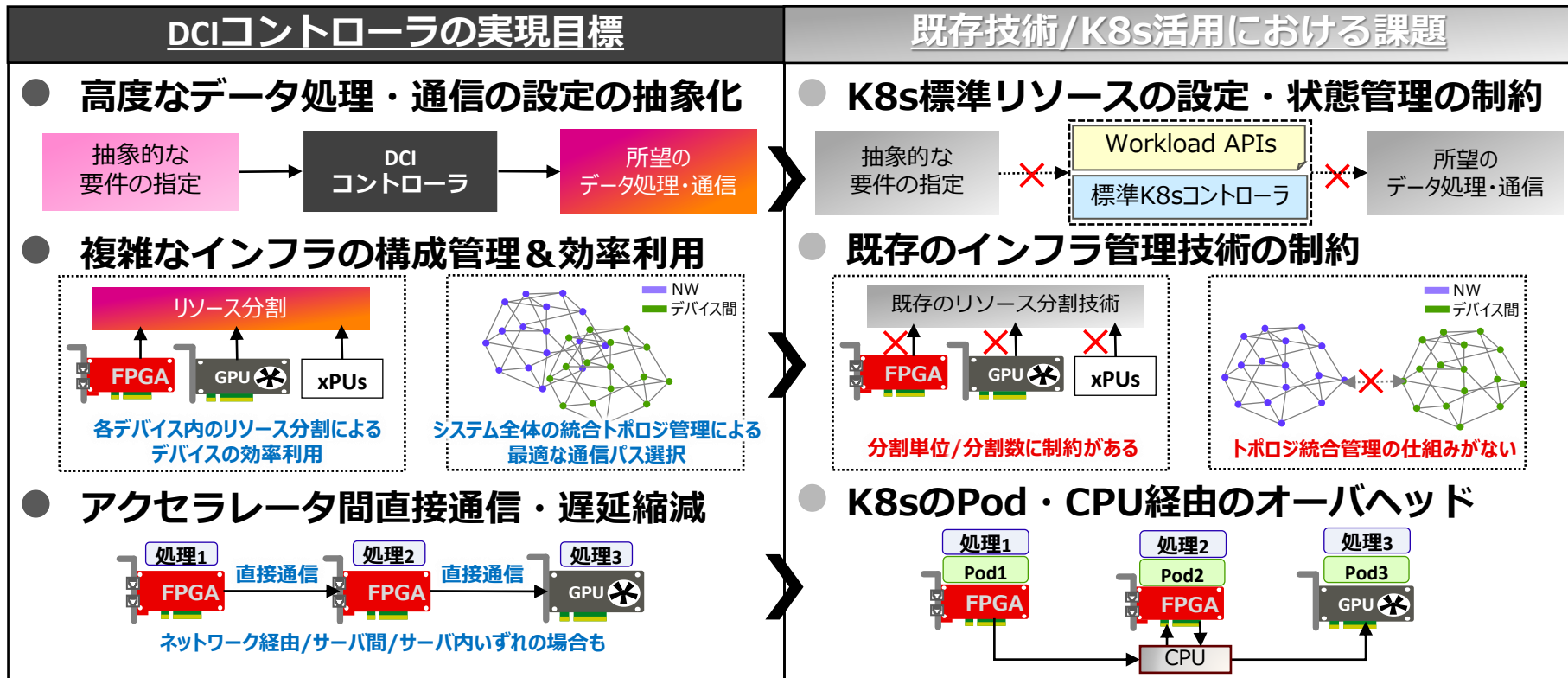
- K8sリソースの「あるべき状態」を、「Reconcile loop」によって管理



- 独自のカスタムリソース/カスタムコントローラを開発して、K8sのAPIを拡張することで、独自のアプリ・ハードウェアを管理可能



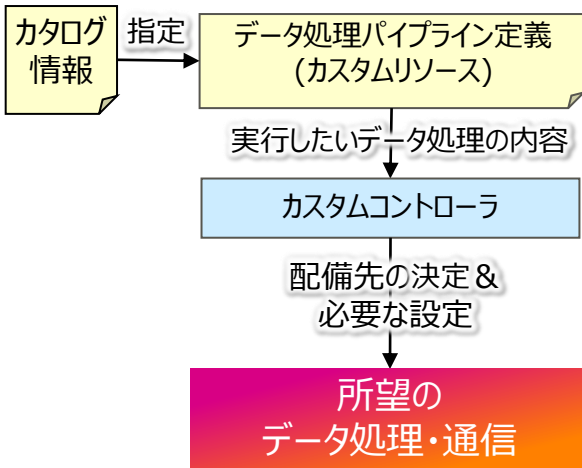
K8s活用を基軸とするDCIコントローラ開発における実現目標と課題



K8sコントローラ開発の取り組み： 取り組みの概要

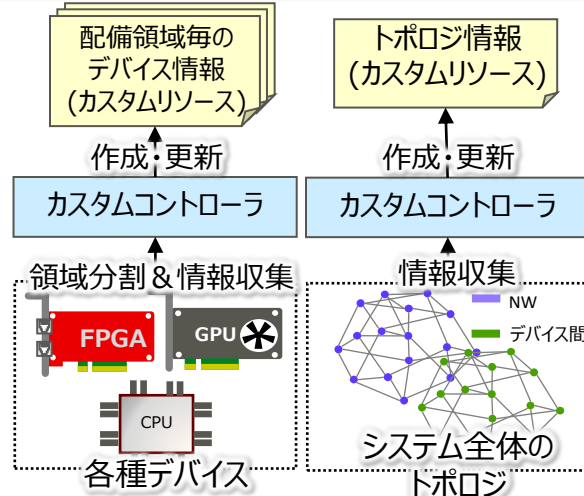
GPU/FPGAを利用する映像AIアプリのK8s上のワークロードを対象に、 高効率なデータ処理・通信の制御を実現するカスタムリソース/コントローラを開発

特徴①データ処理パイプラインの抽象化



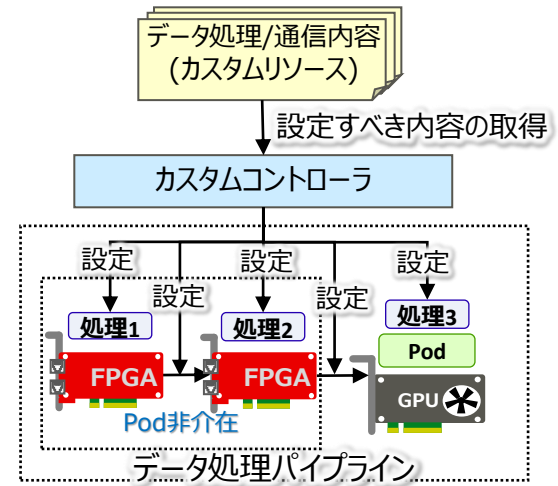
実行したいデータ処理のカタログ情報から具体的な配備先の決定や設定を実施

特徴②複雑なインフラの最適管理



独自の領域分割技術とデータモデルにて複雑なインフラ構成の把握・効率利用

特徴③アクセラレータの高度な制御



各データ処理・通信の内容に応じてアクセラレータに柔軟に設定を実施

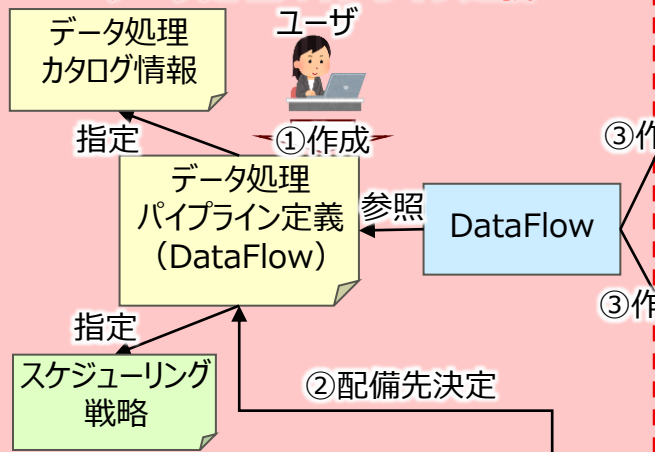
K8sコントローラ開発の取り組み： 主要機能および動作イメージ

1. データ処理パイプライン定義作成 & 配備先決定

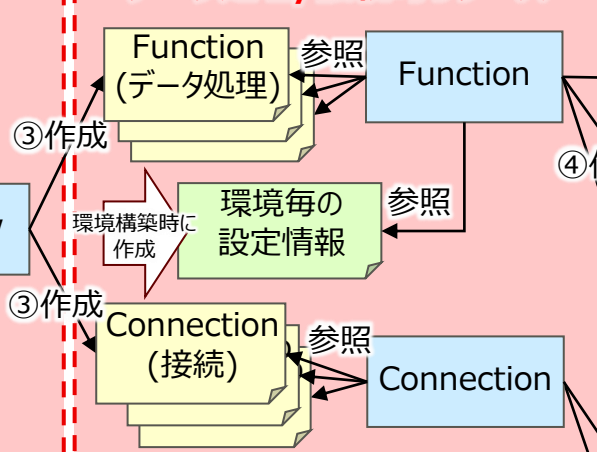
2. 各データ処理/接続毎のリソースの作成

3. 各デバイスに対する設定 & データ処理パイプラインの配備

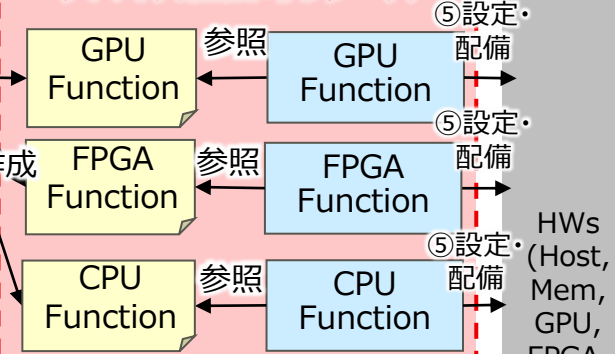
データ処理パイプライン定義



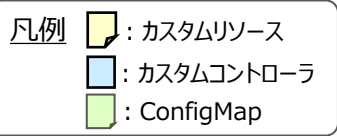
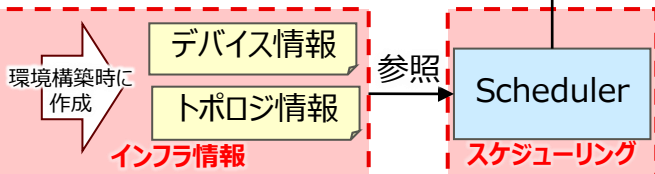
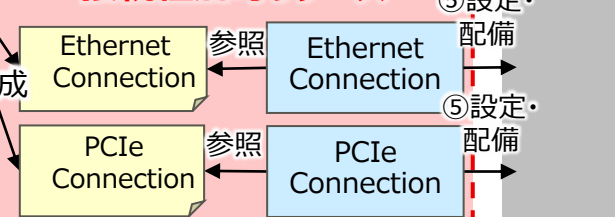
データ処理/接続毎リソース



デバイス種別毎リソース



接続種別毎リソース



配備結果をふまえて状態を更新

カスタムリソース/カスタムコントローラ詳細

①データ処理パイプライン定義 (DataFlow関連)

データ処理のカタログ情報とスケジューリング戦略の指定内容に基づいてSchedulerが配備先を決定

データ処理のカタログ情報

データ処理毎のカタログ情報

```
kind : FunctionType
metadata:
  name: "カタログ1"
spec:
  functionName: デコード
  functionInfoCMRef: スペック1
  ...
```

データ処理毎のスペック情報

```
kind : ConfigMap
metadata:
  name: "スペック1"
data
  ...
# 配備可能なデバイスの種類
regionType: FPGA
# In/Outの通信プロトコル
input : ethernet
output: pcie
```

データ処理パイプライン構成

```
kind : FunctionChain
metadata:
  name: "パイプライン1"
spec:
  # 実行するデータ処理群
  functions :
    デコード
    フィルタリサイズ
    推論
  # データ処理間の接続順序
  connections
  ...
  - from : デコード
  - to : フィルタリサイズ
  ...
```

```
kind: DataFlow
metadata:
  name: "flow1"
  ...
spec:
  # データ処理パイプライン名
  functionChainRef:
    name: "パイプライン1"
  ...
  # 実行時パラメータ
  ...
  # インフラ容量の消費量 (宣言値)
  requirements:
    all:
      capacity: 15
  #スケジューリング戦略
  userRequirement: "ユーザ設定1"
```

上記の設定を元に、
Schedulerが以下を決定

- 各データ処理の配備先デバイス
- データ処理間の接続方法

スケジューリング戦略

スケジューリング戦略の指定

```
Kind : ConfigMap
metadata:
  name: "ユーザ設定1"
data
  strategy : "戦略1"
```

具体的なスケジューリング設定

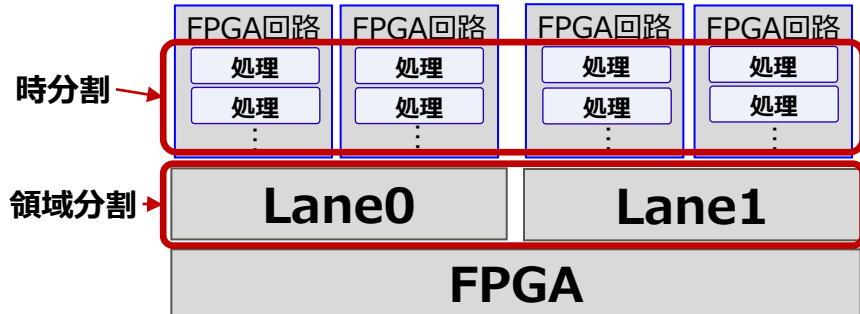
```
kind : ConfigMap
metadata:
  name: "戦略1"
data :
  ...
#フィルタリング/スコアリング設定
filterPipeline: |
  - フィルターA
  - フィルターB
  - スコアリングA
  - スコアリングB
  ...
```

カスタムリソース/カスタムコントローラ詳細： ②インフラ情報（独自の領域分割&データモデル）

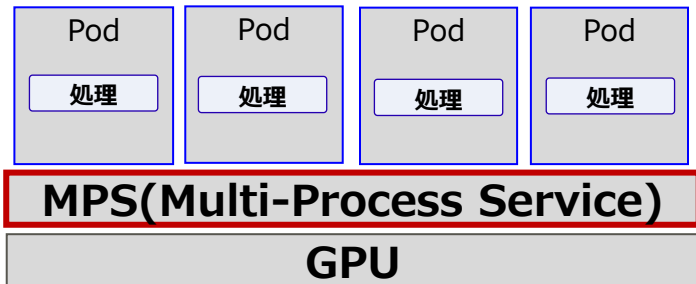
以下の各インフラ情報を管理し、データ処理パイプラインの配備先として利用

データ処理の配備領域毎のデバイス情報

FPGAの場合：領域分割&時分割Hybrid(独自技術)

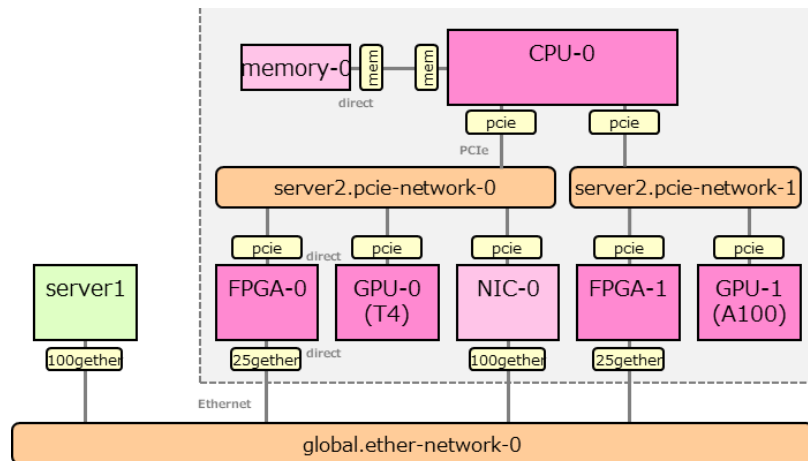


GPUの場合：MPSによるプロセス分割



NW・デバイス間接続の統合トポロジ情報

YANGベースのデータモデルを参考にした
独自データモデルを考案



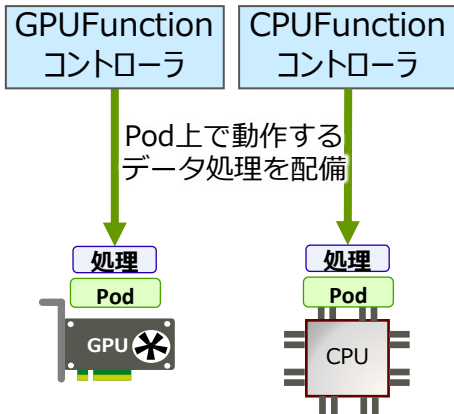
カスタムリソース/カスタムコントローラ詳細

③データ処理・通信のための各デバイスに対する設定

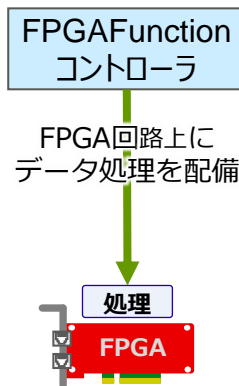
各コントローラが各デバイス種別/接続種別毎に必要な設定を柔軟に実施

デバイス種別毎のデータ処理の設定・配備

CPU/GPU利用



FPGA利用



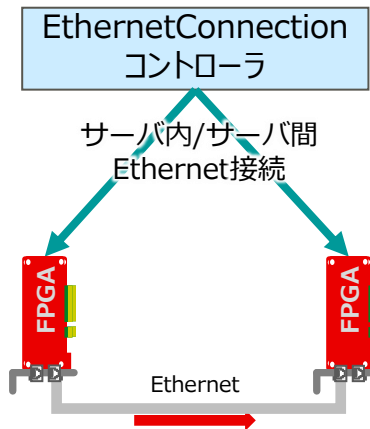
PCIe接続の場合：
共有メモリ利用設定

Ethernet接続の場合：
2nd NIC + SR-IOV利用設定

FPGAリソース割当
FPGA回路の書き込み

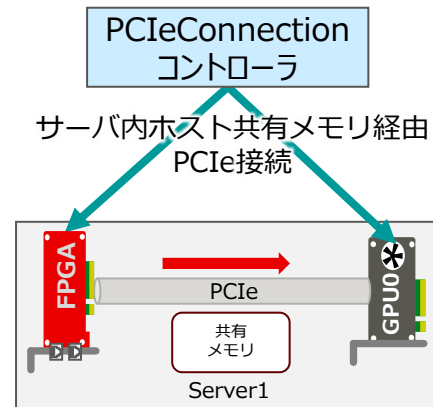
接続種別毎の接続確立のための設定

Ethernet接続



TCP接続の開放・確立

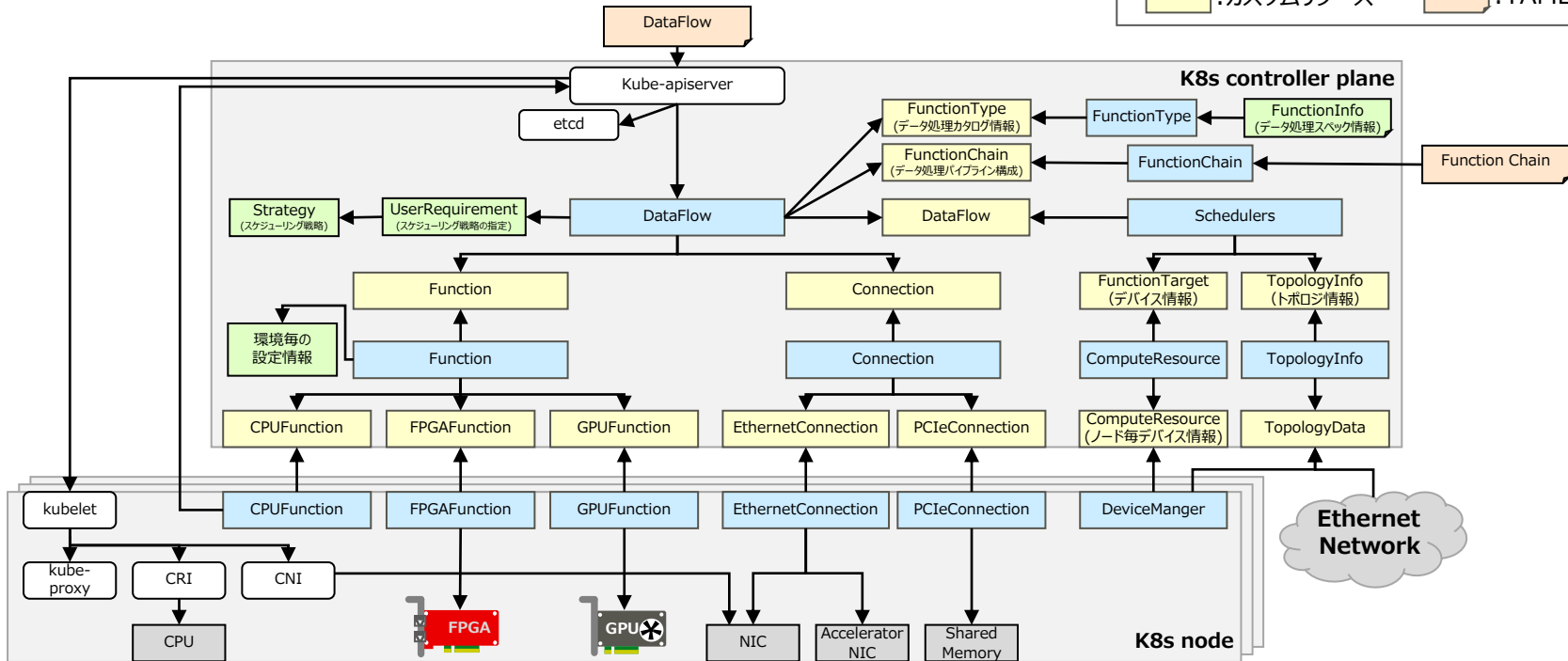
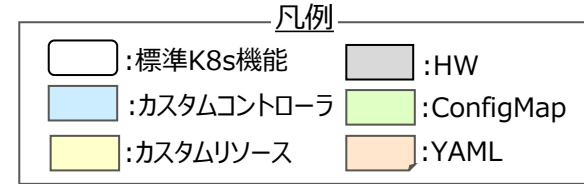
PCIe接続



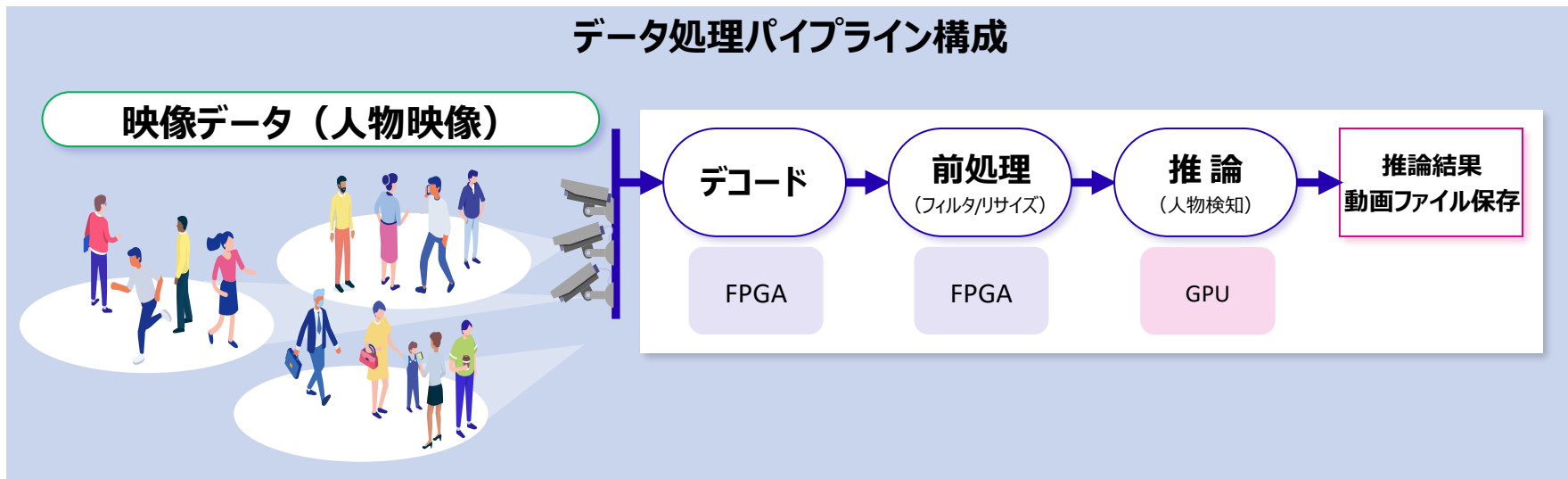
DMAチャネルの設定
共有メモリの設定

K8sコントローラ開発の取り組み： 開発物の全体像と環境構成

- サーバ構成：汎用またはAIワークロード向け高性能x86系サーバ
- ネットワーク構成：単一NWスイッチ配下のEthernet環境
- K8s環境構成：単一K8sクラスタのマルチノード環境



- 複数のアクセラレータの利用・負荷変動の大きさから、映像AIはDCIに適したユースケースの1つです
- **カスタムコントローラの制御により、映像データの人物検知が行われる様子**をデモとしてお見せします
(録画した映像を再生いたします)
- このデモにおける**データ処理パイプラインの構成**は以下となります



- これまでのK8sコントローラの開発から得られた知見も活用しながら、DCIおよびコントローラの実用化に向けた対応を進める
- 独自技術の研究開発を進めつつ、オープンコミュニティとの連携強化を図りたい
 - 本日まで説明したK8sコントローラ開発の内容は、今後オープンソースとして公開する予定
- DCIおよびコントローラの実用化に向けた課題は多岐に渡るが、その一部として、実用化に向けた環境構成をふまえた高効率なデータ処理・通信と、それに必要な運用管理の観点で、以下のような3つの課題が挙げられる
 - ① 広域・多拠点展開
 - ② データプレーン通信の高速化
 - ③ 計算資源・トラフィックの最適管理

実用化に向けた今後の展望： オープンコミュニティとの連携について

各分野のオープン技術との連携により、実用化に向けた課題解決に取り組みたい

課題

活用・連携の候補となるオープン技術の例

① 広域・多拠点展開

ハイブリッドクラウド環境への対応



ClusterAPI

クラスタ管理



Prometheus



Thanos

監視



KARMADA

K8sリソース伝搬

etc...

② データプレーン通信の高速化

RDMAによるアクセラレータ間直接通信を軸とする高速化



MULTUS

マルチホー ムPod



NVIDIA
GPU OPERATOR

アクセラレータ
運用管理



Spiderpool

アンダーレイ
NW管理

etc...

③ 計算資源・トラフィックの最適管理

処理内容および負荷に応じたリソース割当の最適化



kubernetes
cluster
autoscaler

各種スケーリング



kubernetes
Dynamic Resource
Allocation

リソース割当
最適化

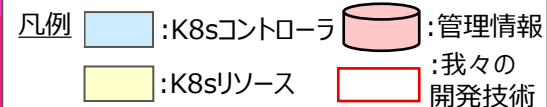


Open vSwitch

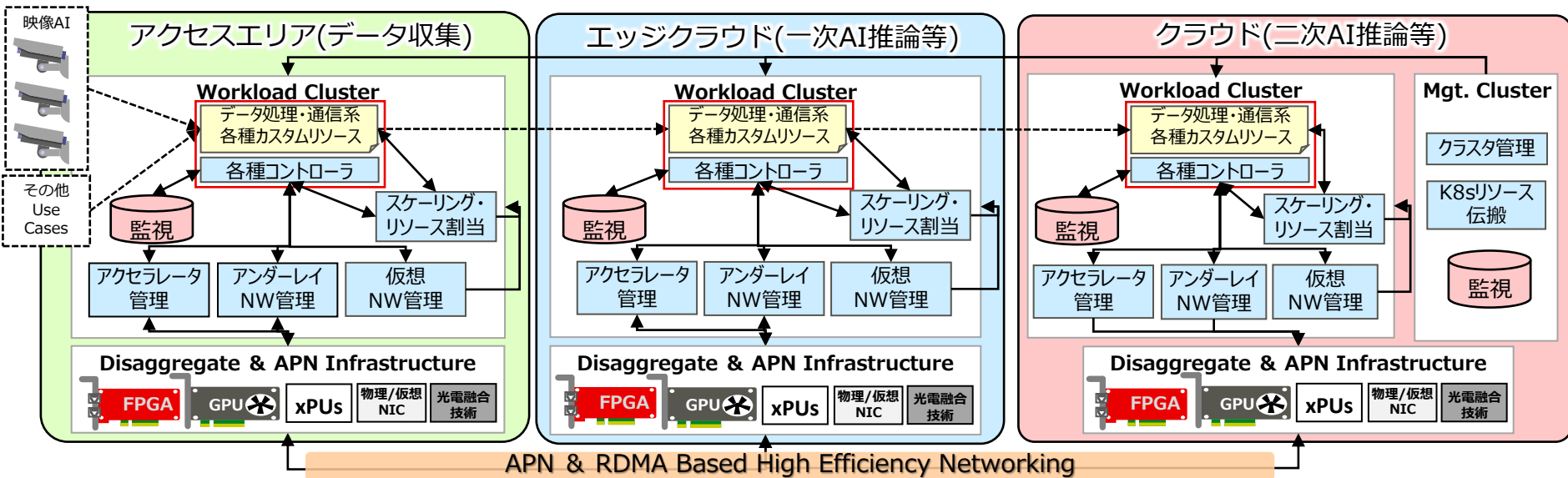
高度な仮想NW管理

etc...

実用化に向けた今後の展望： オープン技術と連携したDCIコントローラアーキ例



我々の開発技術と各分野のオープン技術との適材適所の役割分担・連携に基づいて、
広域・多拠点のNW環境下でもアクセラレータ間直接通信で高効率なデータ処理・通信を実現



① 広域・多拠点展開

- クラスタ管理
- 監視
- K8sリソース伝搬 等



② データプレーン通信高速化

- マルチホームPod
- アクセラレータ管理
- アンダーレイNW管理 等



③ 計算資源・トラフィック管理

- 各種スケージング
- リソース割当最適化
- 高度な仮想NW管理 等



本日は、富士通が研究開発を進めるDCIについて、これまでのK8sを活用したコントローラ開発の取り組みを共有させていただきました。

富士通は、DCIの実用化に向け、研究開発や、オープンコミュニティの活動への参画、社会実装に向けた実証などの取り組みを推進していきます。

ぜひ皆様と連携させていただき、取り組みを進めたいと考えておりますので、どうぞ宜しくお願い致します。



Thank you

