Cloud Native Telecom Operator Meetup

クラウドネイティブに向けた取り組み

KDDI株式会社 モバイル技術本部 次世代ネットワーク開発部 宮本 元

2019年11月29日
(全17枚)

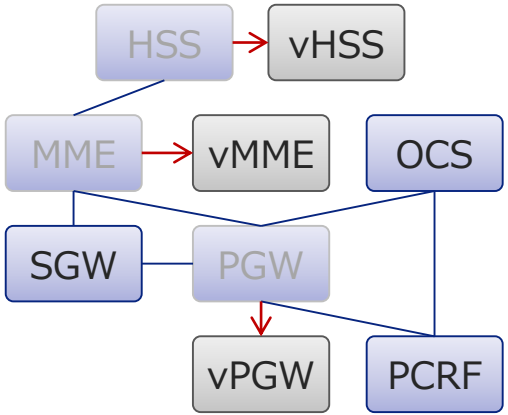
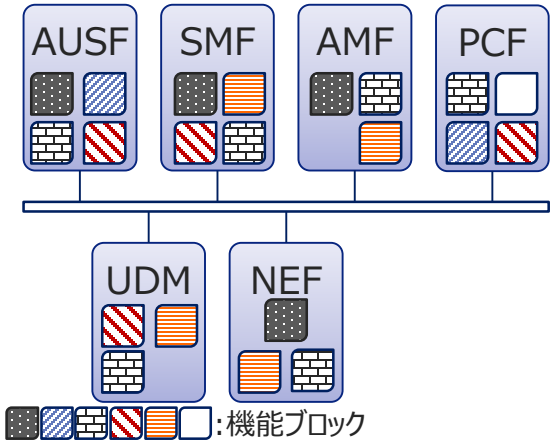
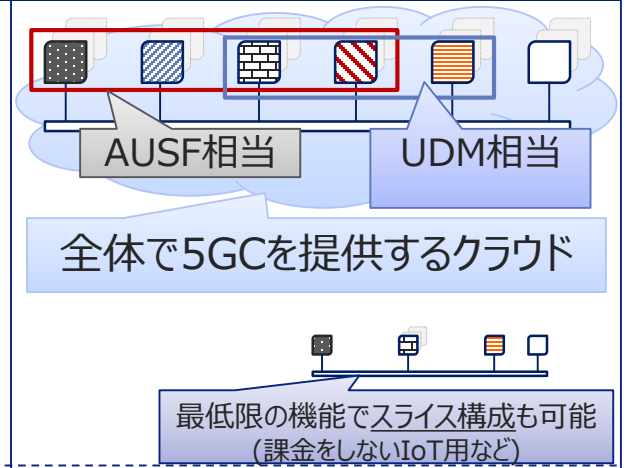
Tomorrow, Together
おもしろいほうの未来へ。

目次

テレコムネットワークとCloud Native

テレコムオペレータのCloud Native対応

■ モバイルネットワークでは5G導入の中でCloud Nativeなアプリケーションの適用が進むと見られる。

EPC/モバイルコア	5GC(SBA: Service Based Architecture)	Cloud Native
		
各ノード単位で仮想化を実施	バスを介してノードはステートレスに連携 ノード内の機能ブロックには同じ機能を有する者が存在する	機能ブロックに分解し、機能ブロックがバスを介して疎結合することで、ノードの機能を実現する。 これにより最低限の機能ブロックでコアを構成。 例）要件に応じたコアのスライスを複数提供

具体的な対応の深堀

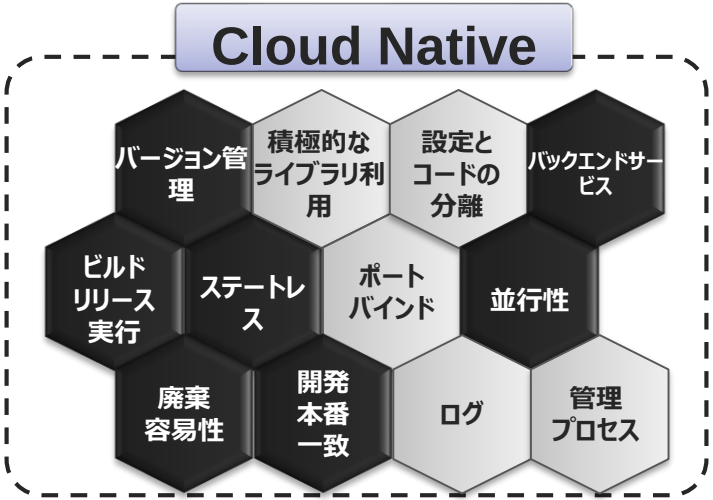
- Cloud Nativeをテレコムネットワークに適用するには具体的に何が必要？
  - Cloud Native＝Kubernetesを入れれば終わり？
  - アプリ構造が変わるだけならオペレータは何もしなくてよい？
  - Cloud Nativeのメリットはどこまで得られる？
- KDDIではこのアーキテクチャの変革を着実に迎えるため、Cloud Native時代に対応した開発手法の検証・技術開発を実施するべく検討を進めた。
  - 本発表では検討骨子と、活動事例を共有します。

KDDIが考えるCloud Native

Cloud Nativeの解釈

■ **“Cloud Native” という概念の構成する要素を「①設計要件」と「②技術要素」の二つの視点から整理。**

- 設計要件として Cloud Native の一般的な定義である “The Twelve-Factor App” から抽出
- 技術要素は、Microservice / Container / Continuous Delivery の三つに大別



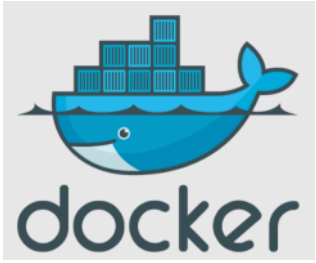
①Cloud Nativeを構成する「設計要件」

②Cloud Nativeを構成する「技術要素」

Container

■ コンテナの利用により、12の要件の実現が容易になるため現代的な実装ではコンテナが必須と見られる。

- **コンテナの特徴（Docker の場合）**
  - ① どこで実行してもアプリ実行環境をそろえられる
  - ② コンテナ起動はプロセス起動と行程が似ており高速に起動する
  - ③ 永続的なデータ保存を行わずアプリケーションをステートレスに保てる
  - ④ 設定を環境変数として取り扱える
  - ⑤ ログは標準出力に出力する

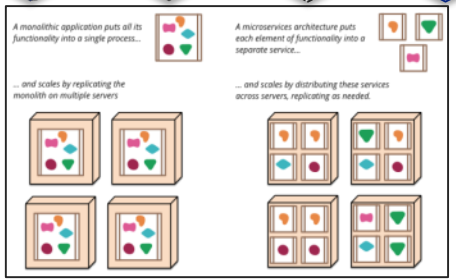


Microservice

■ Cloud Native の根幹をなすのが Microservice アーキテクチャ

- **Microservice の特徴**
  - ① サービスを通じたコンポーネント化
  - ② ビジネス遂行能力を軸にした組織編制
  - ③ プロジェクトではなくプロダクト
  - ④ 賢いエンドポイントと土管
  - ⑤ 分散統治
  - ⑥ 分散データ管理
  - ⑦ インフラ自動化
  - ⑧ 障害のための設計
  - ⑨ 進化的な設計

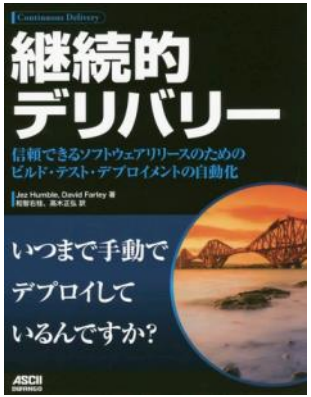
James Lewis/Martin Fowler
“Microservices” より転載
日本語訳:
http://kimitok.hateblo.jp/entry/2014/11/09/211820



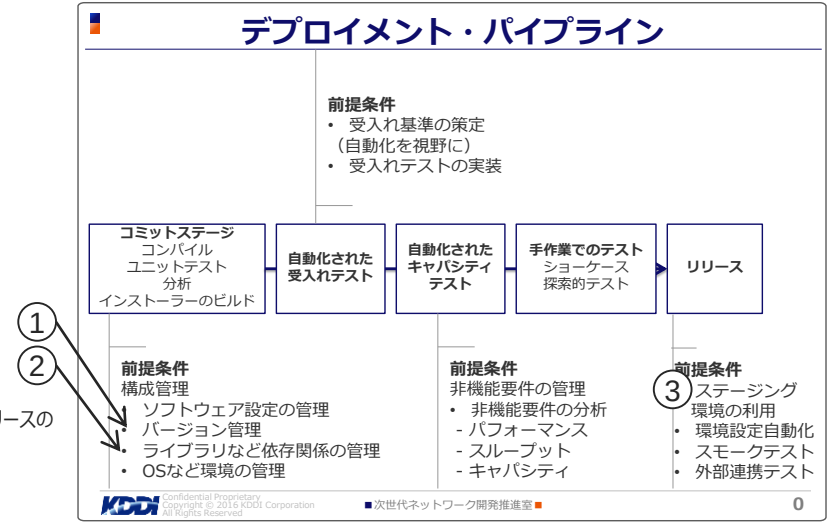
Continuous Delivery

■ アプリ実装後の検証・デプロイまでに必要な技術要件を定義

- リリースサイクルを細切れに行う（小さく開発し小さくリリース）
- 試験スコープや作業範囲（＝バックアウトの範囲）を狭める
- 構成管理や試験、リリースの自動化が前提にあり、
  人的リソースを増やさずリリースサイクルを短縮することを目指す



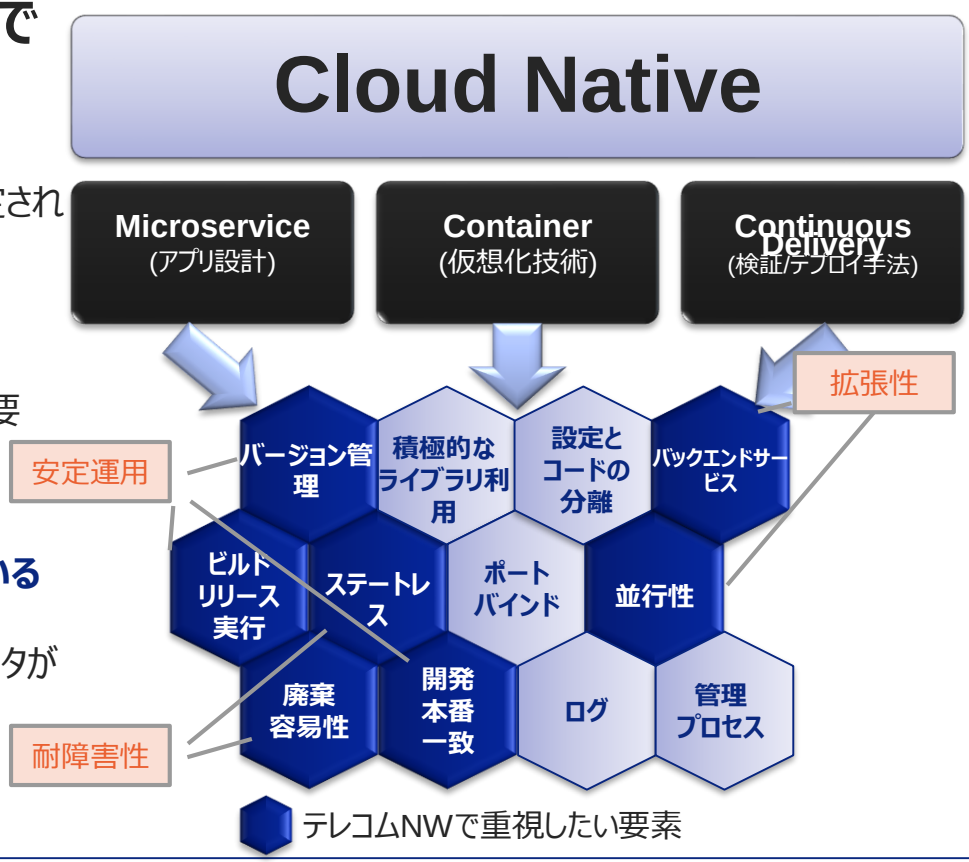
継続的デリバリー: 信頼できるソフトウェアリリースのためのビルド・テスト・デプロイメントの自動化
著: Jez Humble, David Farley



検討のまとめ

■ Cloud Nativeなアプリをテレコムで活用するには運用がキーワード

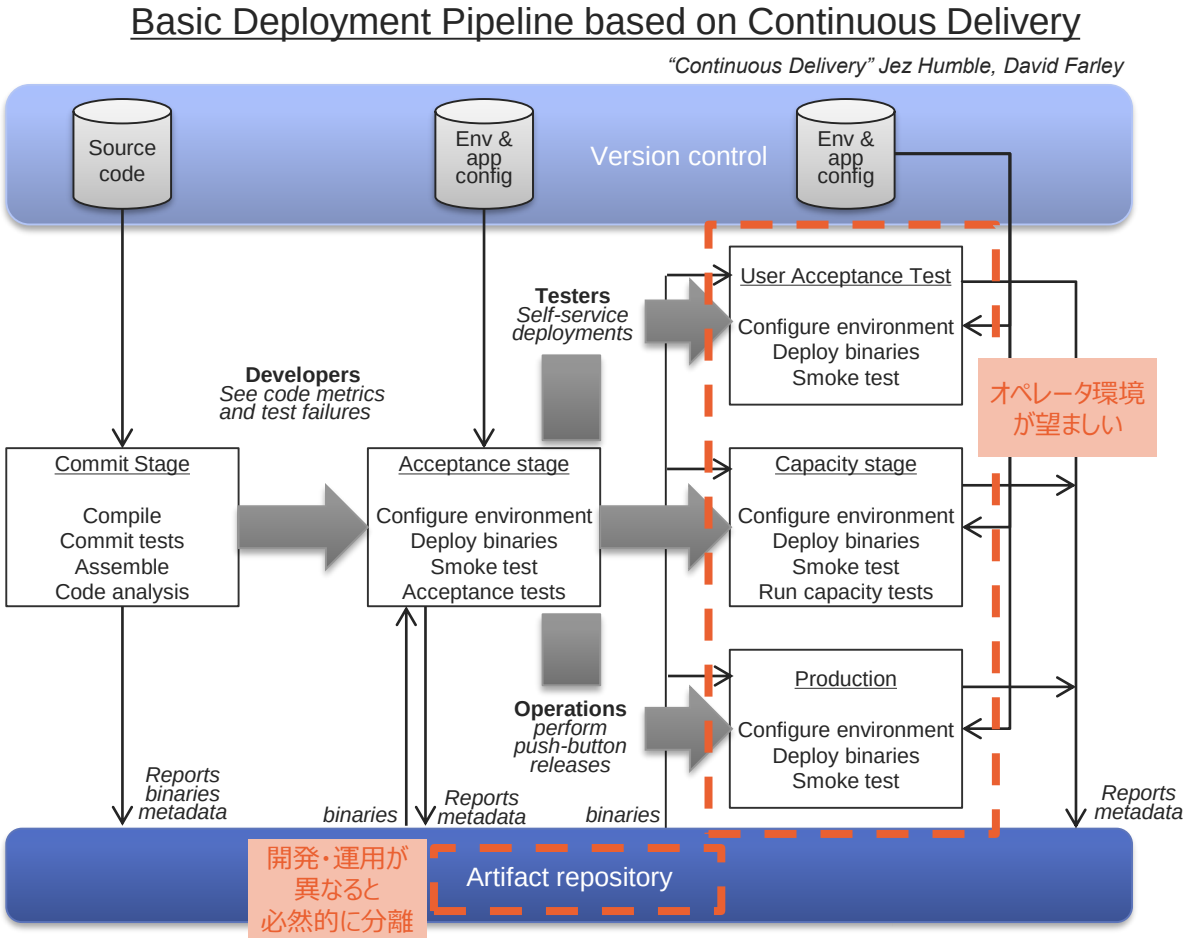
- **Microservice**
  - アプリ設計の領域であり、基本的にベンダ内で決定されるが、柔軟な運用を実現する設計が望ましい
  - ベンダ間の設計差分が運用に影響する恐れも
- **Container**
  - 新しく機能レイヤが増えるため運用面の対応が必要
  - **コンテナの特徴を活かす活用法は工夫が必要**
- **Continuous Delivery**
  - **アプリの開発者と運用者が、異なる立ち位置にいるテレコム業界では導入に障壁**
  - 商用適用を見据えると運用ノウハウを持つオペレータが主導した検討が必要



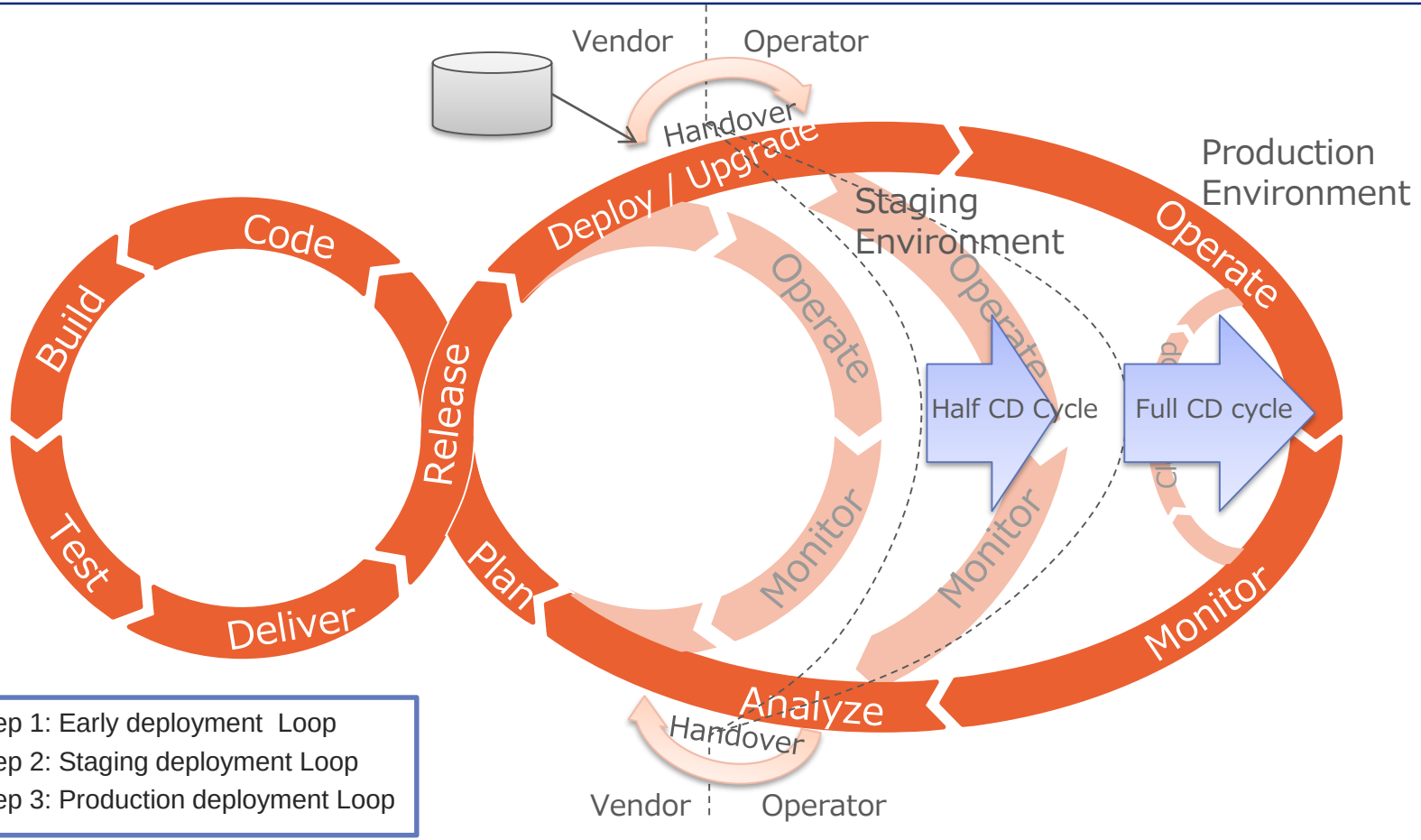
KDDIの取り組み

テレコムNWにおける Continuous Delivery 実現への課題

- **Continuous Deliveryでは、一般に開発・運用が一体であることが前提**
  - テレコムNWでは開発と運用が別体制。一気通貫の検証環境を構築するには、組織を超えた環境の構築が必要
  - Artifact Repositoryは知的財産の塊であり、別組織へ張り出しは慎重に行うべき
  - テレコムNWは複数のNFで構成されることから、複数のデプロイメントパイプラインからなる。複雑な環境を再現するIOT試験が大切



組織横断的なDevOpsループの構築

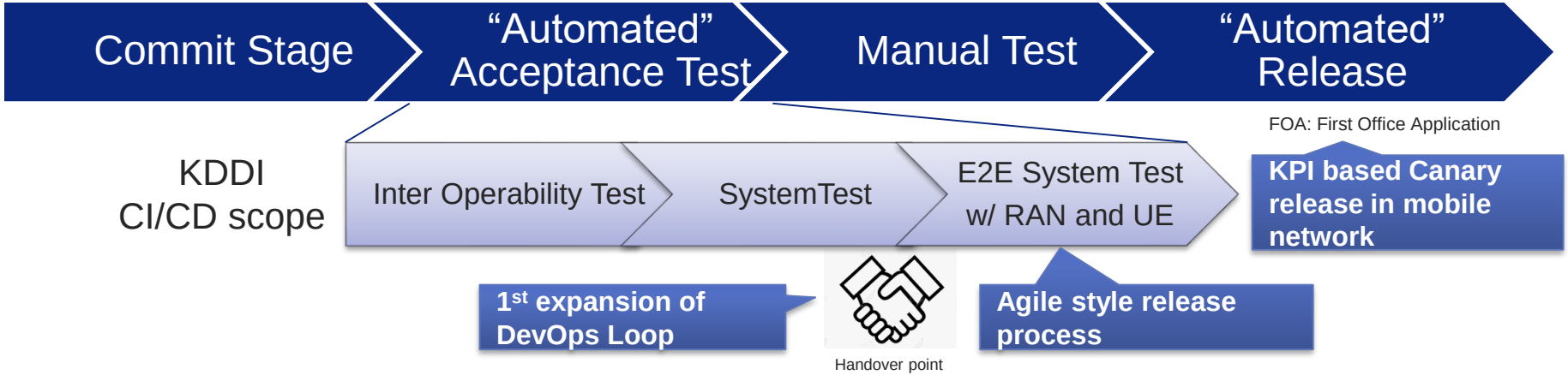


Step 1: Early deployment Loop
Step 2: Staging deployment Loop
Step 3: Production deployment Loop

Continuous Delivery 今後の展望

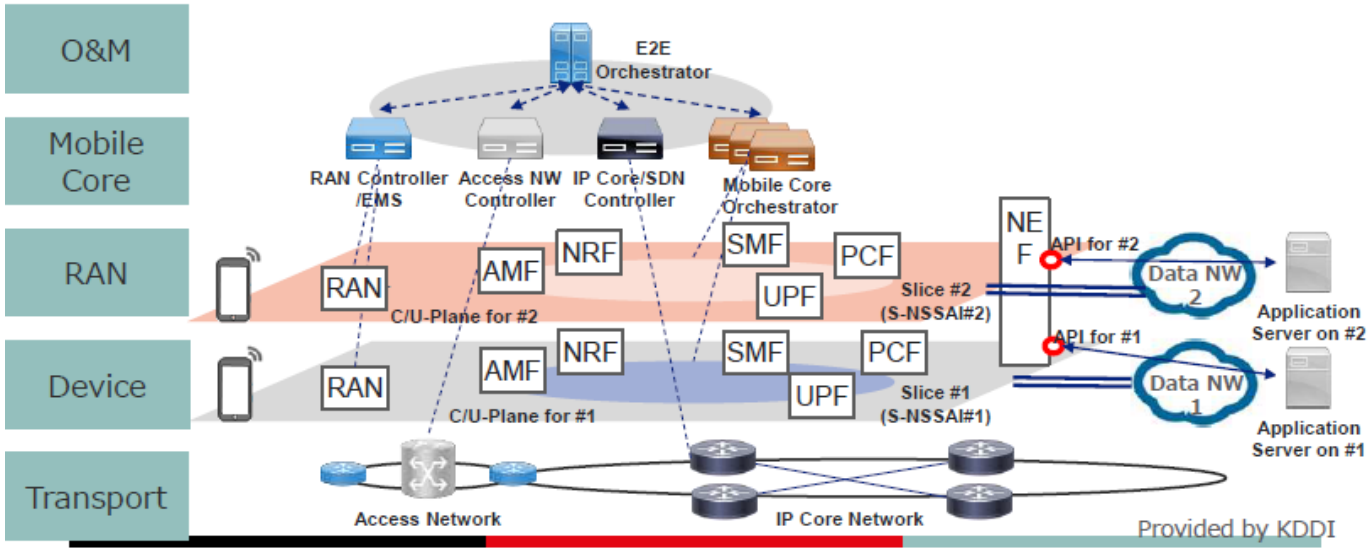
■ デプロイメントパイプラインの確立のため、引き続き検討を進める。

- **商用環境への適用**
  - カナリアリリースやロールバックなど、従来から行ってきた手順をCloud Native的なアレンジ
- **エラー検出時のフィードバックを活用し、Continuous Integrationを加速**
  - エラー解析チームのスケーラビリティ
  - 修正を反映するリリースサイクルの規定



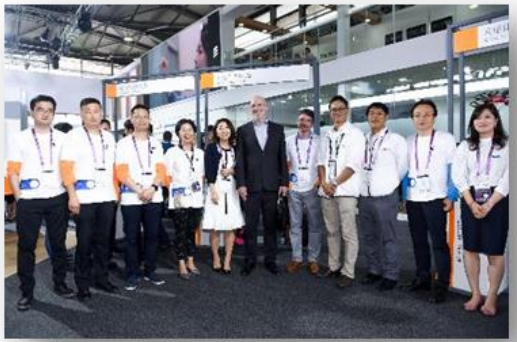
Container　特徴の活用

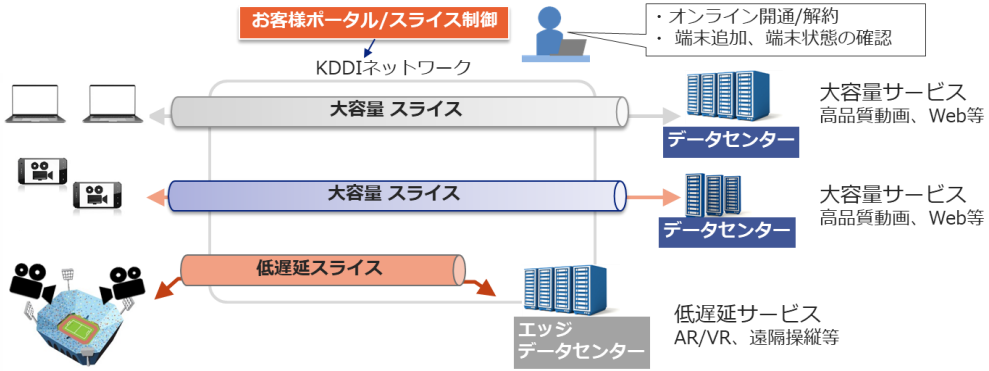
- コンテナを利用したMicroservice化されたNFを使うと「柔軟なネットワーク」を実現できる、と言われるがどのような世界観を実現できるのか？
  - たとえば、コアNWやRANで実現が期待されるNWスライスにおいて、大容量・低遅延などお客さまの用途に応じ、ネットワークを動的に変更するような運用を実現したい



コンテナ実装されたNFにおける実証開発

- ベンダ様と共同でコンテナ実装されたNFを連携（オーケストレーション）する技術検証を実施。実施内容をMWC上海2019でデモ展示、プレス発表
- 5GコアNWを利用したNW Sliceのオンデマンド構築、ゼロタッチ認証を訴求
  - NW Sliceのオンデマンド構築
    コンテナ技術を導入し、顧客操作に応じオンデマンドでNW Slicingを提供するデモ公開
  - 5GC NW-APIを用いたゼロタッチ認証
    5GCのNEF機能を用い、APIを外部公開する事でアプリとサービス提供者間の認証を自動化する仕組みをデモ公開

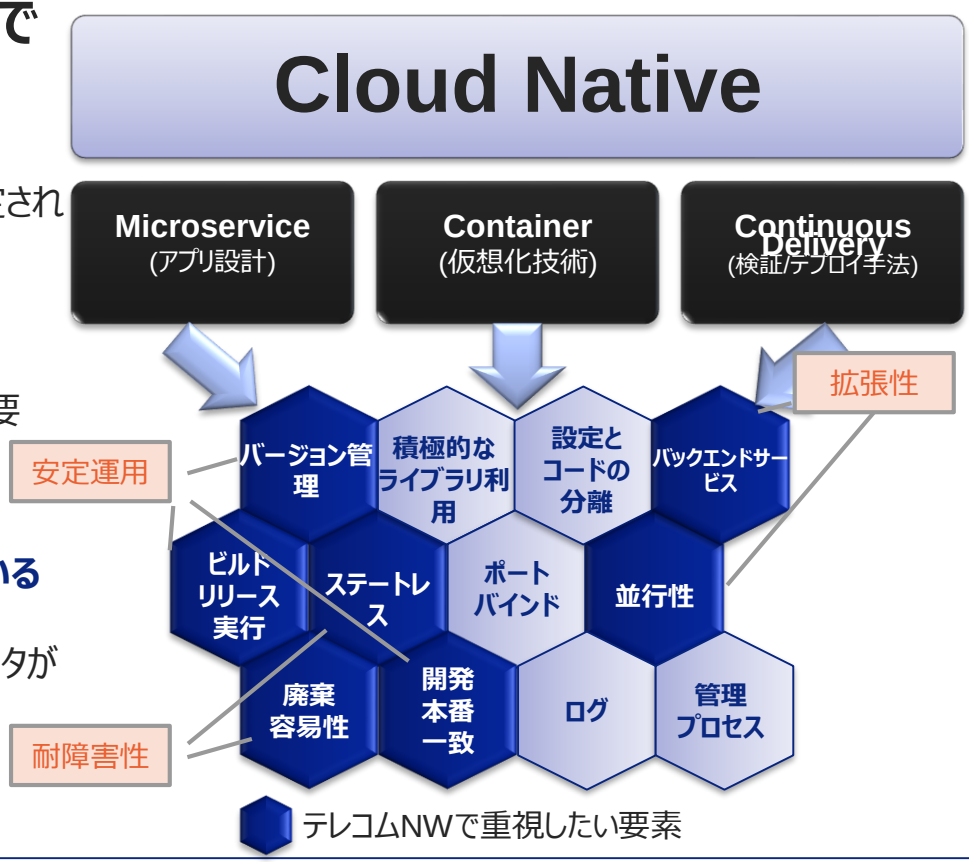




https://news.kddi.com/kddi/corporate/newsrelease/2019/06/24/3880.html

検討のまとめ（再掲）

- **Cloud Nativeなアプリをテレコムで活用するには運用がキーワード**
  - **Microservice**
    - アプリ設計の領域であり、基本的にベンダ内で決定されるが、柔軟な運用を実現する設計が望ましい
    - ベンダ間の設計差分が運用に影響する恐れも
  - **Container**
    - 新しく機能レイヤが増えるため運用面の対応が必要
    - **コンテナの特徴を活かす活用法は工夫が必要**
  - **Continuous Delivery**
    - **アプリの開発者と運用者が、異なる立ち位置にいるテレコム業界では導入に障壁**
    - 商用適用を見据えると運用ノウハウを持つオペレータが主導した検討が必要




Tomorrow, Together
KDDI
おもしろいほうの未来へ。
au