仮想化基盤ソフトウェア更改に向けた取り組みとその課題
～やっぱりOpenstackの複数世代Skipは大変だった～

NTTドコモ　　ドコモ・テクノロジ
磯田 卓万　　鎌田　亨

2024年1月17日

自己紹介

磯田 卓万

鎌田 亨

目次

1. ドコモのネットワーク仮想化基盤の概要 1/2

ドコモでは2016年以来、コアNWシステムを収容するETSI ISG NFVに準拠した仮想化基盤を構築・運用しています。

Data Centers **24**

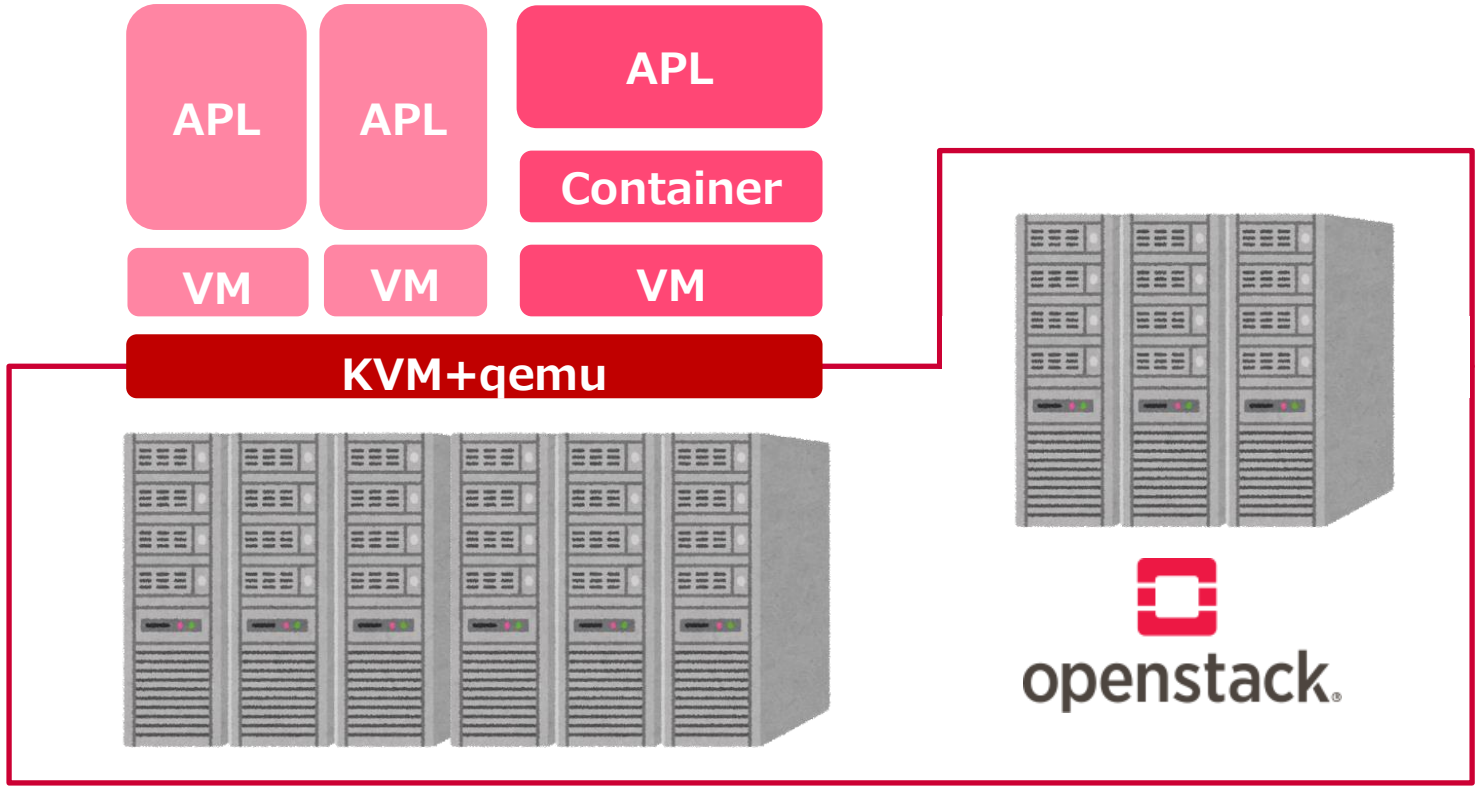
Virtual Network Function **40＋**

Virtual Machines **250,000＋**

Servers **9,000＋**

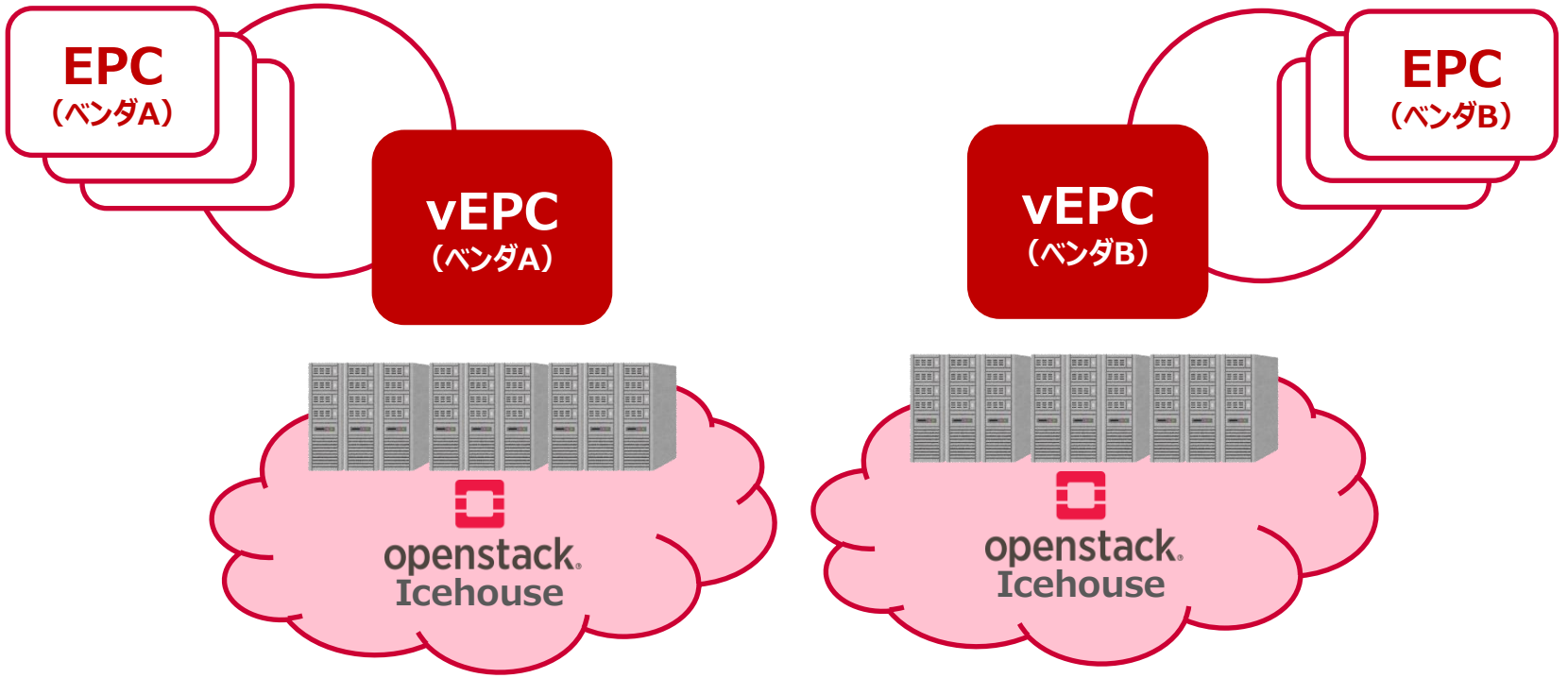
1. ドコモのネットワーク仮想化基盤の概要 2/2

OpenStack, KVMベースのIaaS基盤上で様々なアプリケーションを運用しており、一部のアプリケーションについてはContainer on VMの形式でデプロイしています。



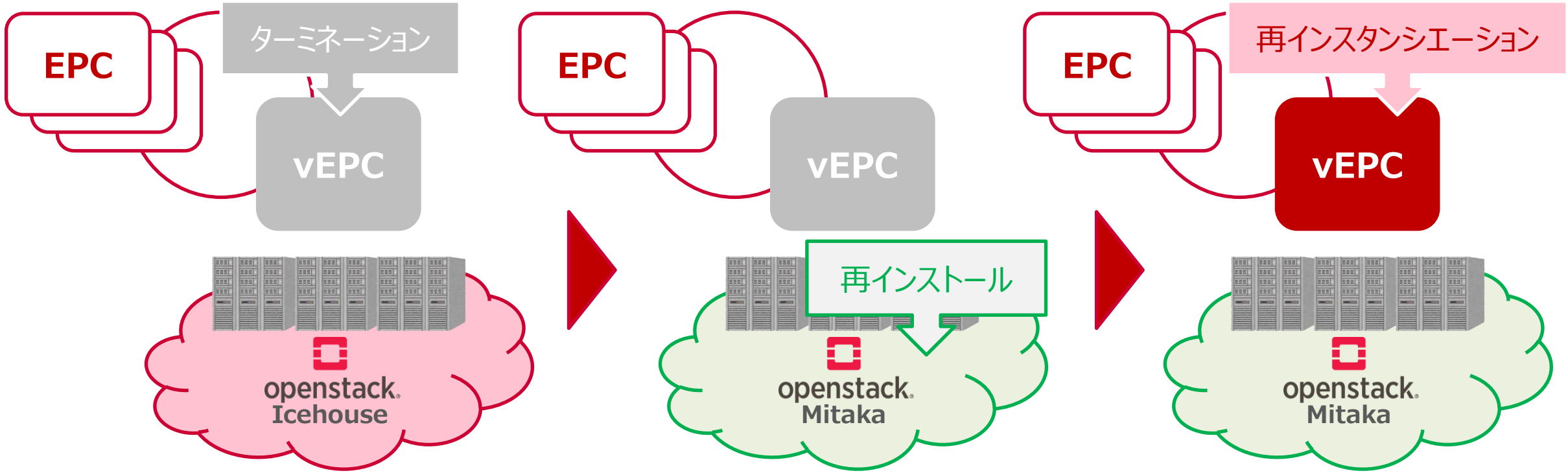
2016年に初期導入

- 2つのOpenStack RegionにvEPCをデプロイ。
- ベアメタルのEPCとプール構成を組むことで、問題発生時のサービス影響を極小化。



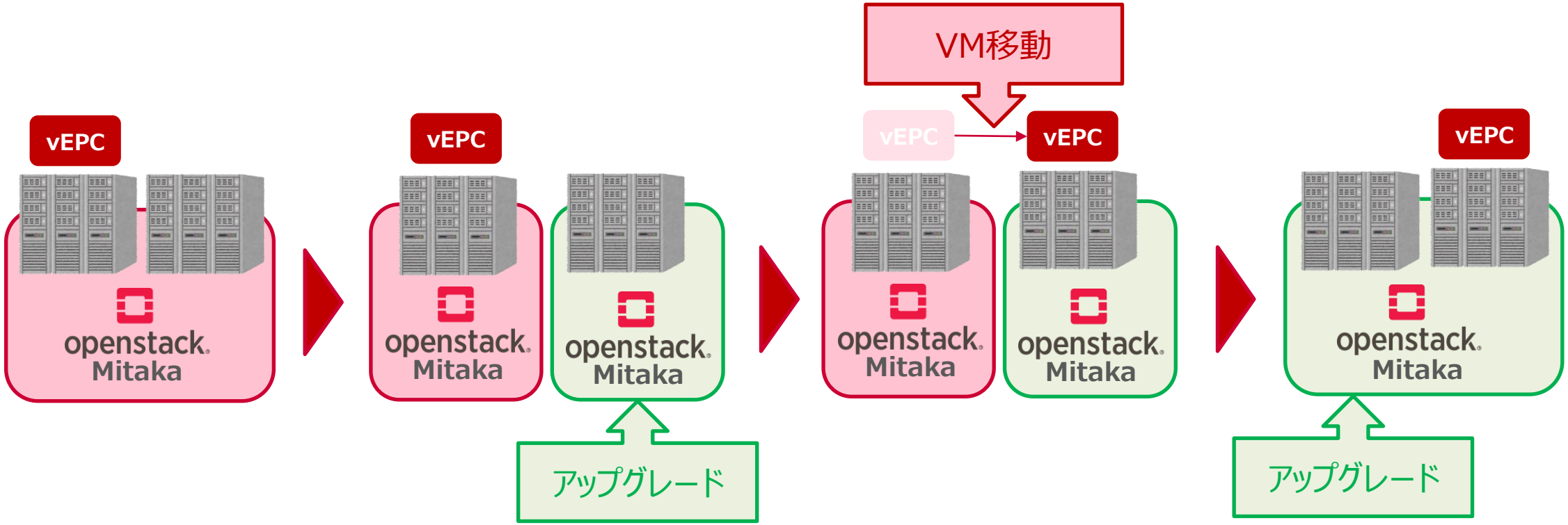
2017年に初のVersionUP

- Openstack Icehouse -> Mitaka にVersionUP
- 小規模なのでサービスを止めたうえで再インストールする手法を選択。



2019年にまたVersionUP

- 導入から3年経過し、運用規模が大きくなっていた。
  DC：9　OpenStack Region：33　サーバ台数：4000台　VNF種類：24種類
- サービスを止めずにVersionUPする手法に挑戦。

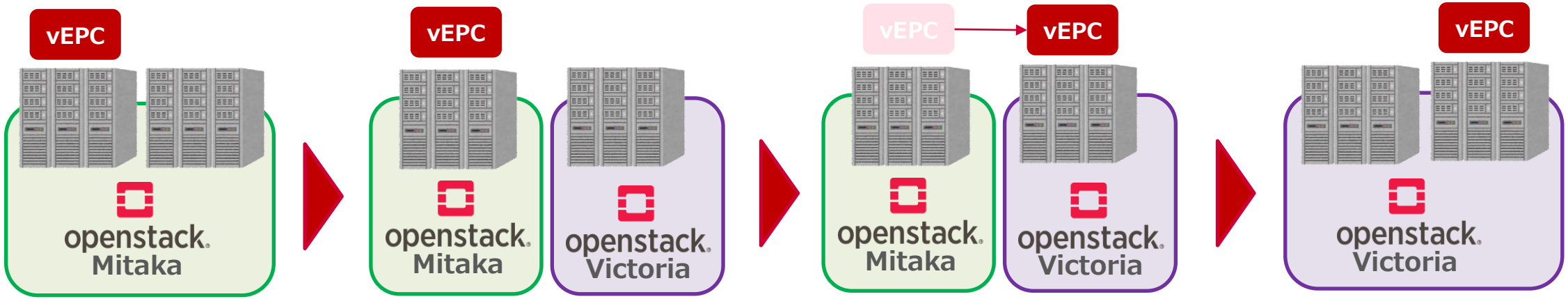


2023年にまたまたVersionUP

- 導入から6年経過し、運用規模が更に大きくなっていた。

  DC：24　OpenStack Region：76　サーバ台数：9000台　VNF種類：43種類

- 前回同様サービスを止めずにVersionUPする手法に挑戦・・・
  だがOpenStackをMitakaからVictoriaにジャンプアップしなければならない・・・



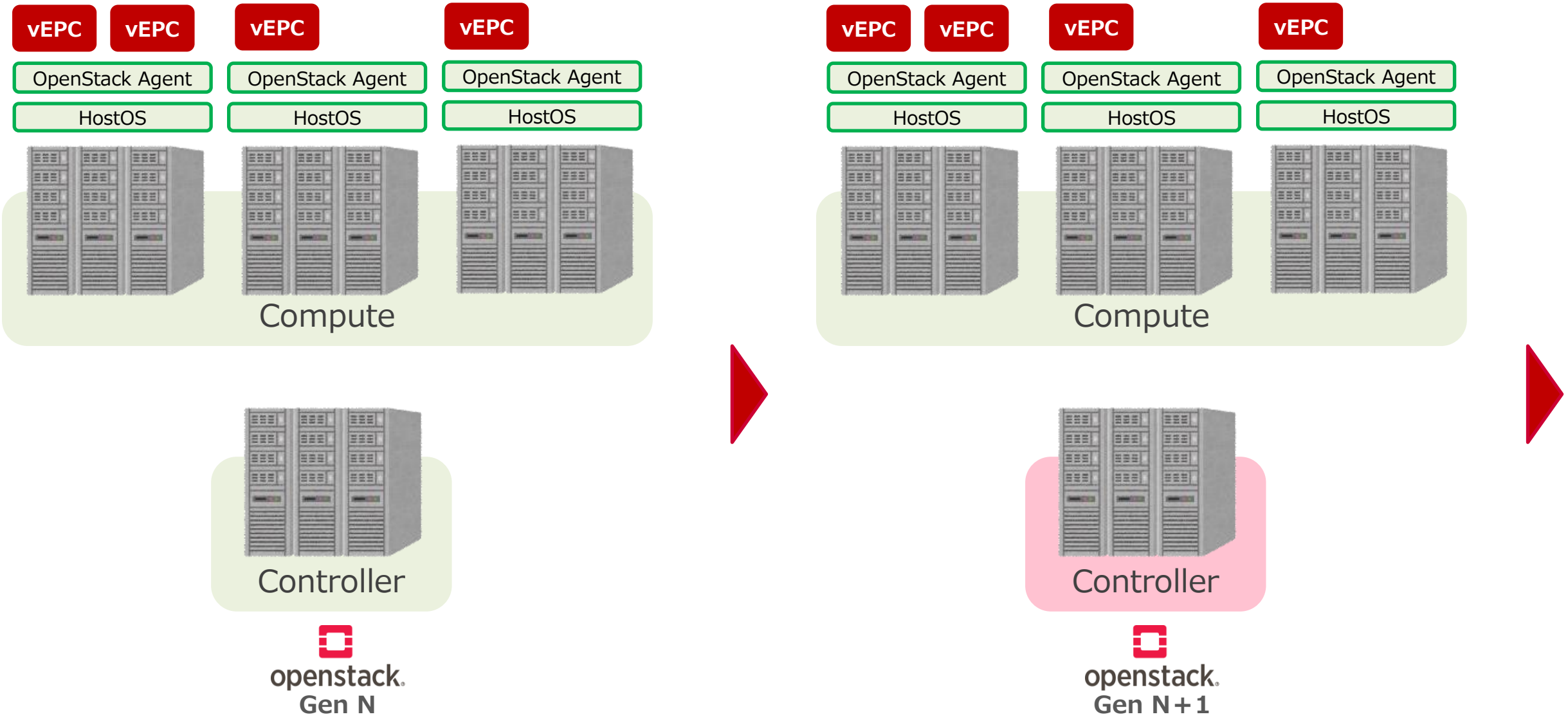
出来るのか？？

3. 仮想化基盤の更改方針

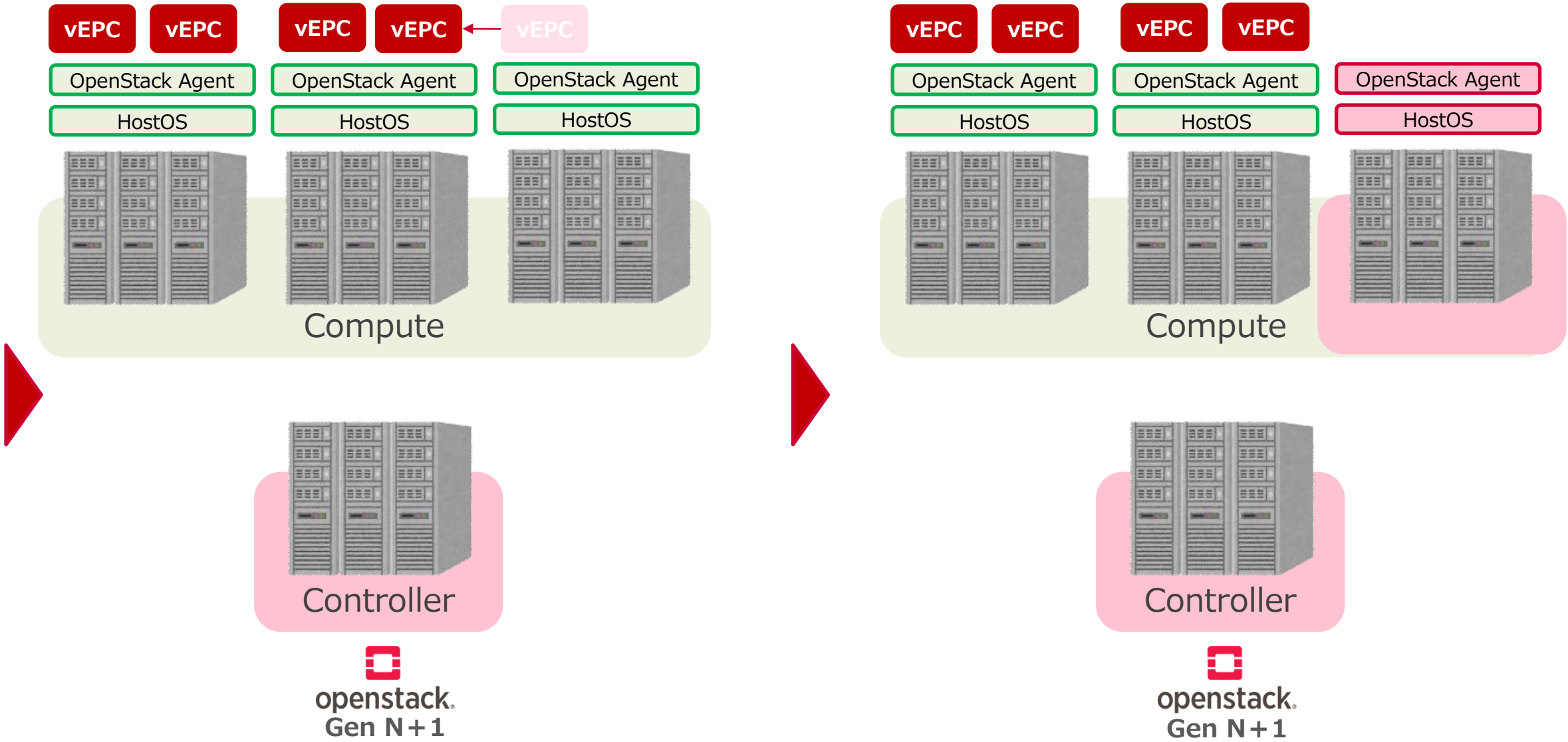
- モバイル通信を提供するアプリを収容している為、サービス無中断であることが絶対条件。
- サービス継続性はアプリの冗長性で担保すれば、基盤単位ではアプリを止められるが、それを加味した冗長度での設備構築が必要になる。

よし！アプリを止めずにローリングアップグレードだ！！

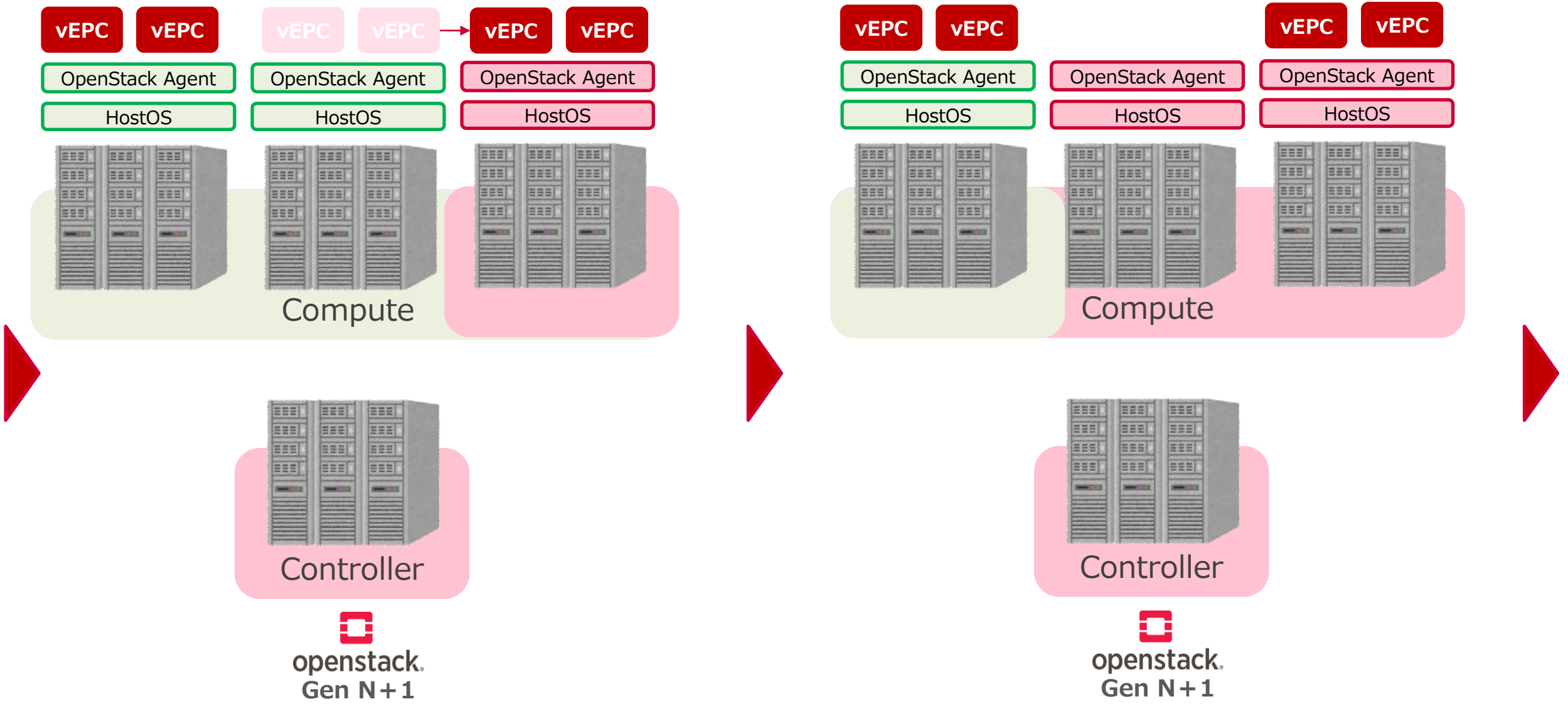
4. ローリングアップグレードの方式（同世代やN+1世代へ）1/4



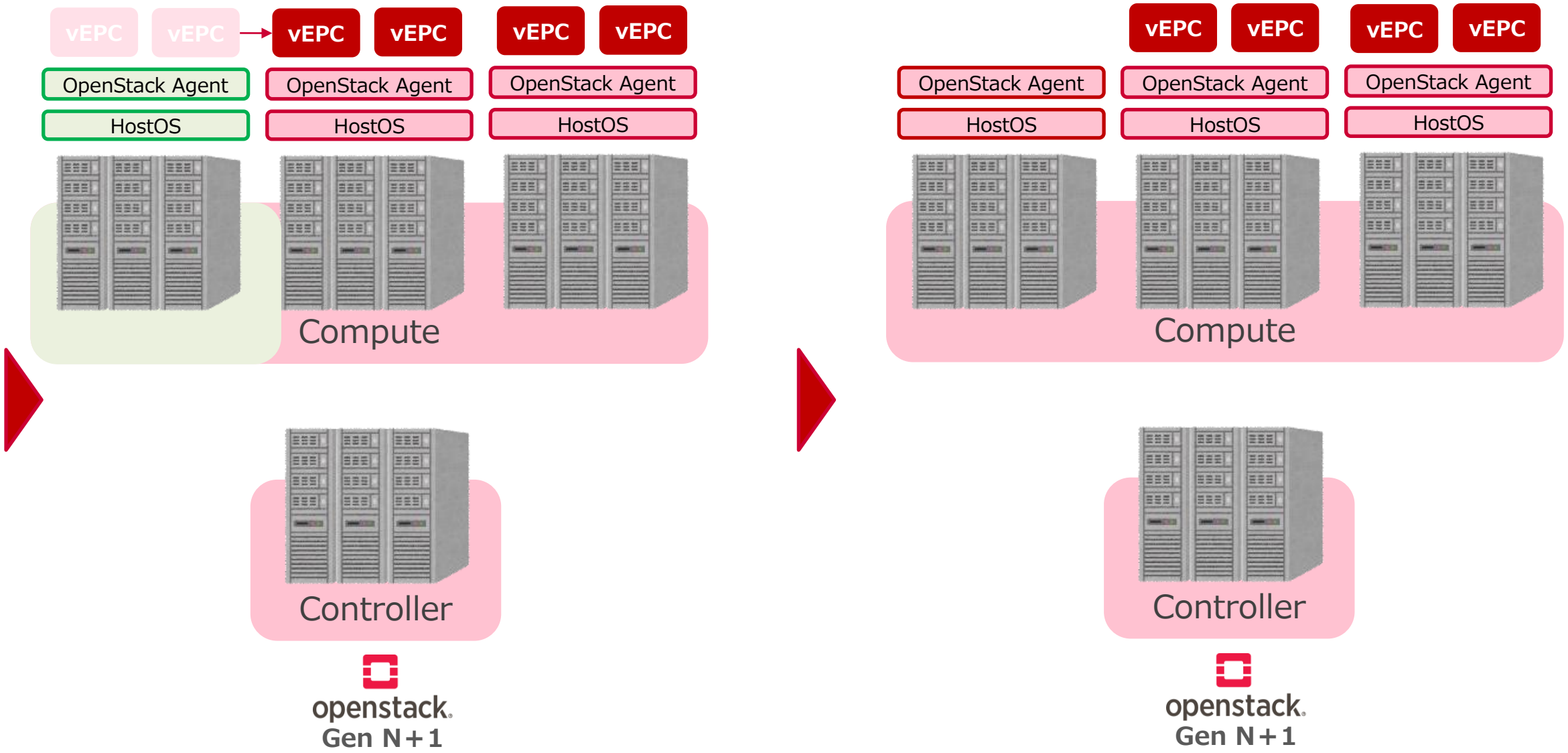
4. ローリングアップグレードの方式（同世代やN+1世代へ）2/4



4. ローリングアップグレードの方式（同世代やN+1世代へ）3/4



4. ローリングアップグレードの方式（同世代やN+1世代へ）4/4



余談

最近のOpenStackはN＋2世代までローリングアップグレード対応しているようですね

※OpenStack 2023.1 (Antelope) 以降

5. Openstack複数世代Skipのローリングアップグレードにおける課題

複数世代Skipしてしまうと・・・

- **根本的にOpenStackとしてローリングアップグレードが提供されていない**
  1. ローリングアップグレードを実現する独自の方式が必要

  2. 世代が開いており、新旧Controller～Compute間でおしゃべり出来ない

  3. 世代が開いており、データベース構造が変わっている

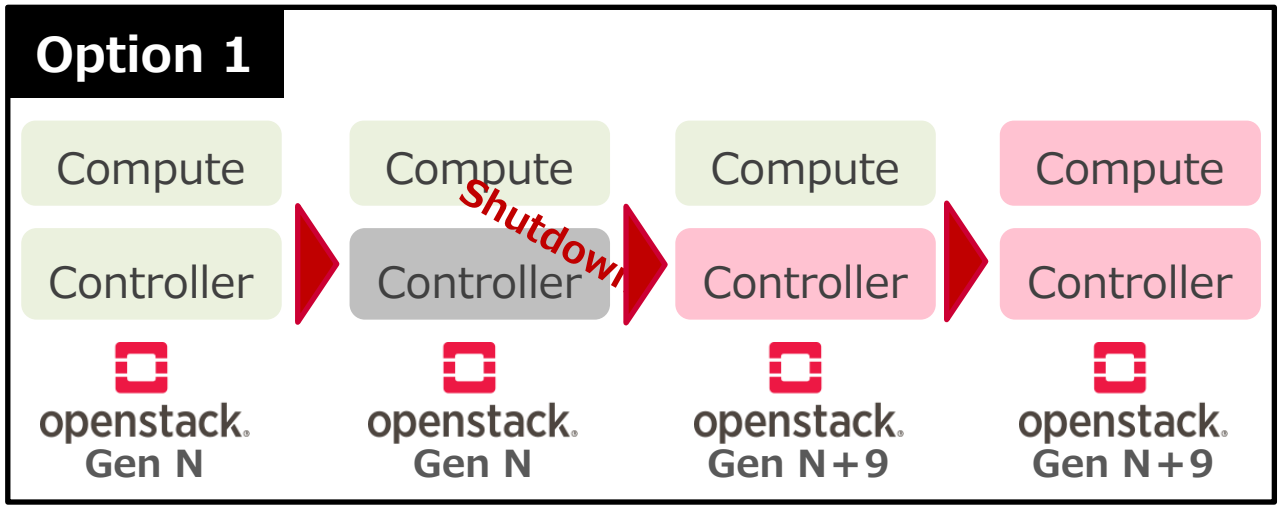
- **世代が新しくなりすぎるとAPI差分が大きい**
  4. 使っていたAPI VersionがDeleteされたり、オプションパラメータが増えたり減ったりしている・・

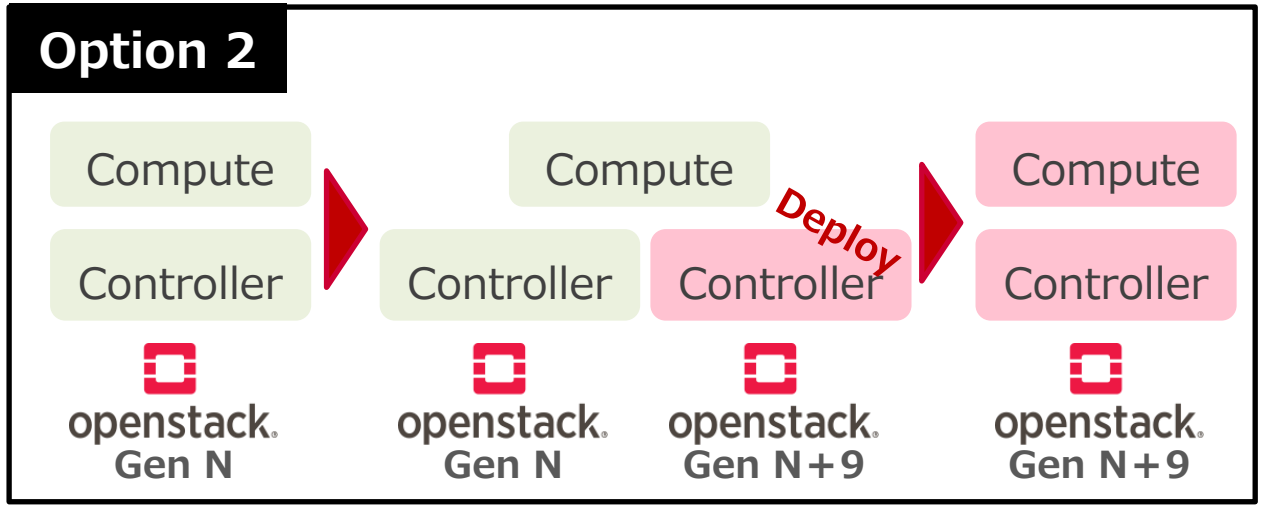
順に課題と対策を説明していきます

先に述べたような稼働中のControllerノード、Computeノードをそのままアップグレードする機能が提供されない為、独自にアップグレードを方式を考える必要があった。Controllerのアップグレードが出来ないので以下のどちらの道を選ぶかは大きな判断ポイントであり、ドコモではOption2を選択した。

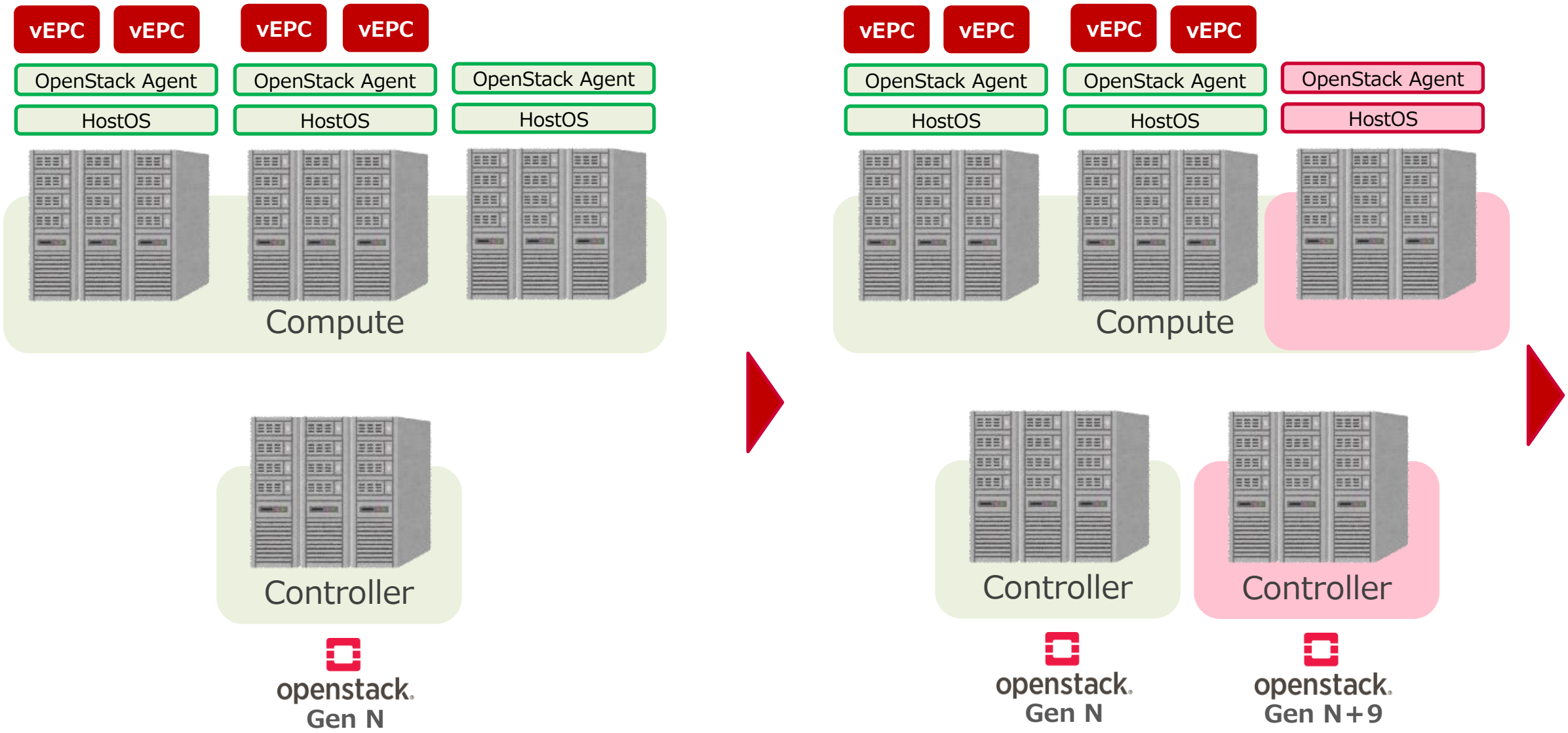
①旧世代Controllerを止めて新世代Controllerをインストールする

②新世代Controllerを追加で新設する

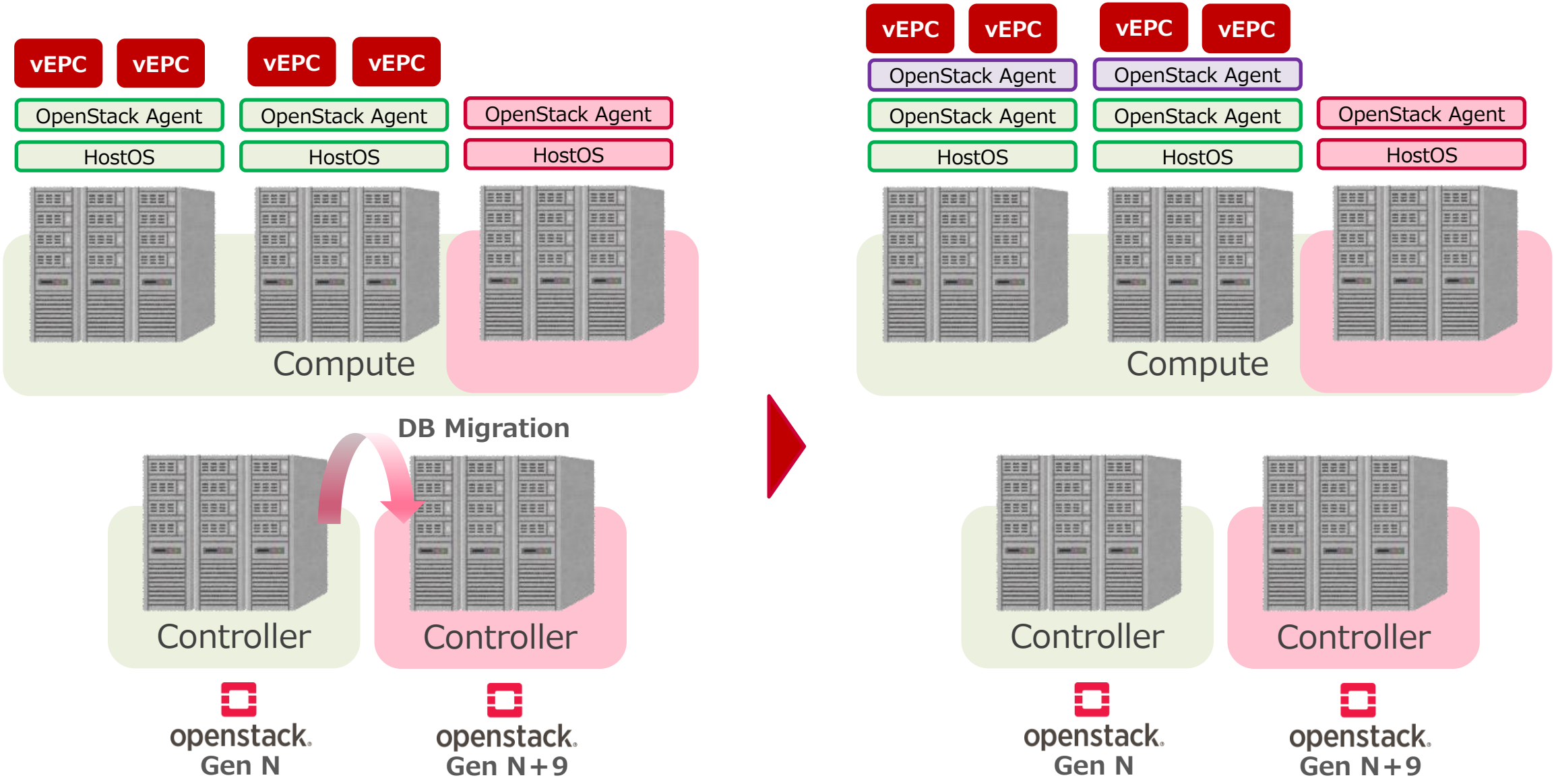




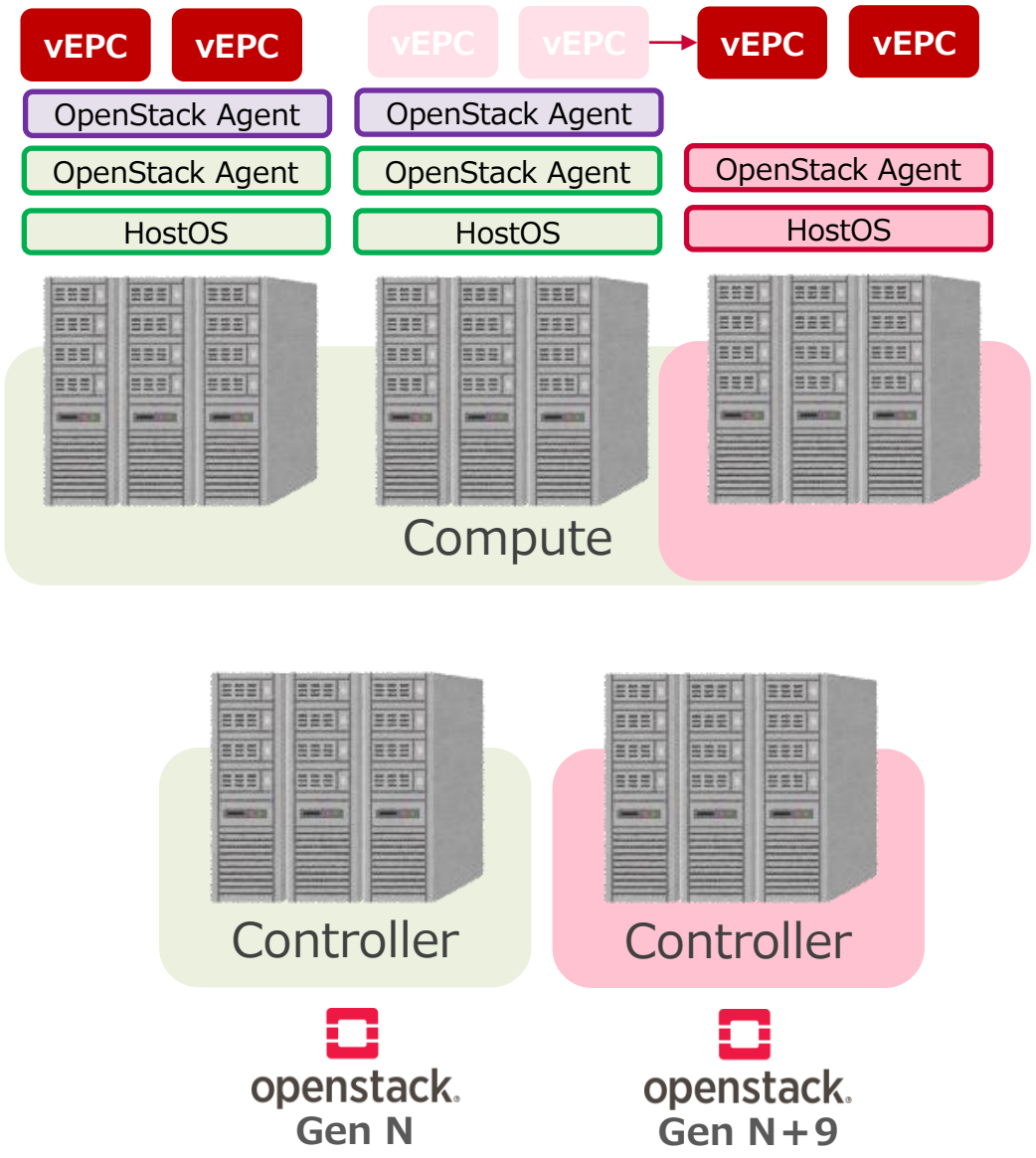
5-1. アップグレード方式

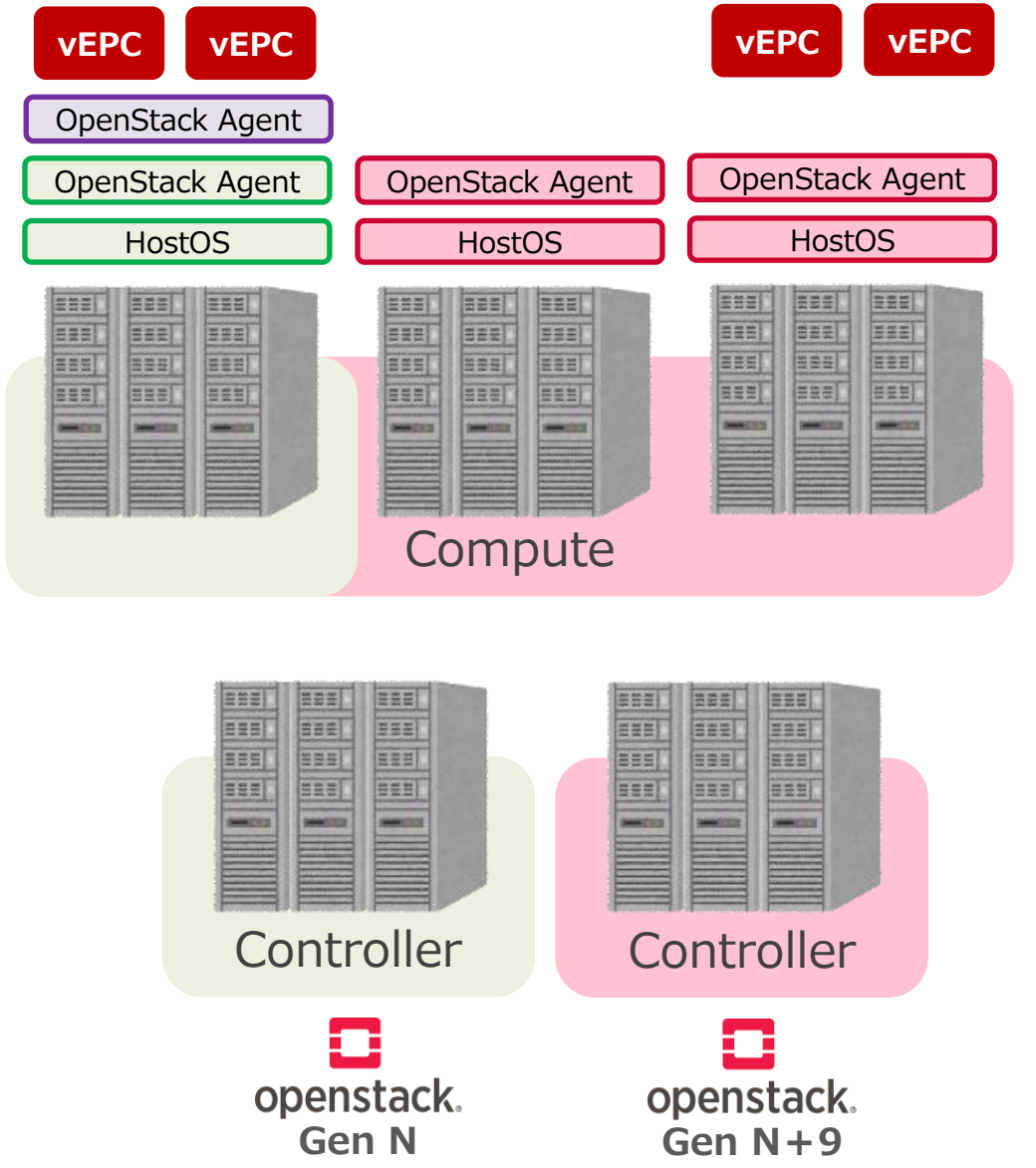


5-1. アップグレード方式

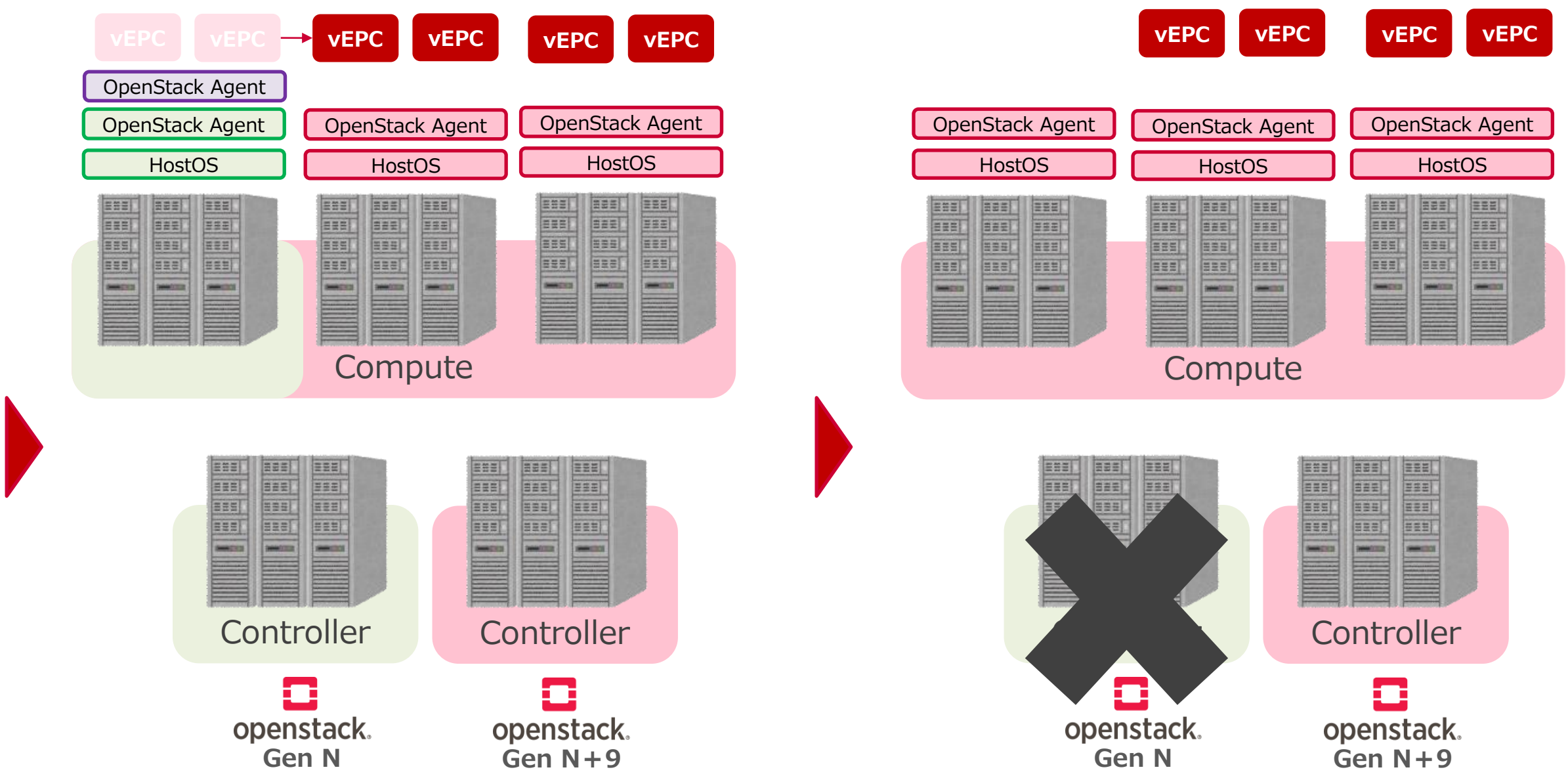


5-1. アップグレード方式





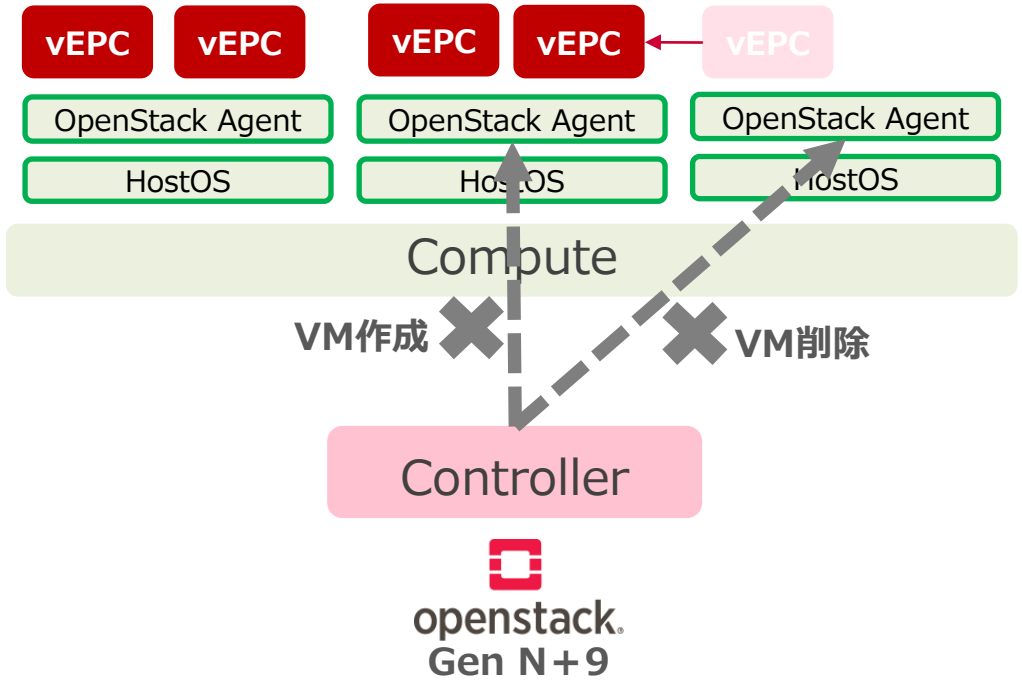
5-1. アップグレード方式

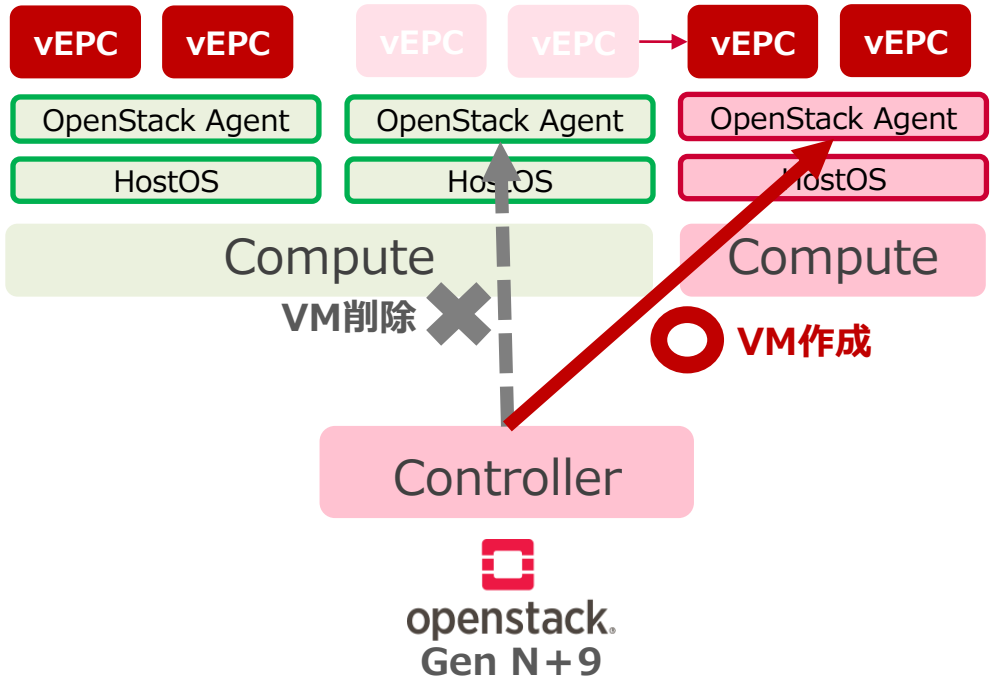


5-2. 互換Agent

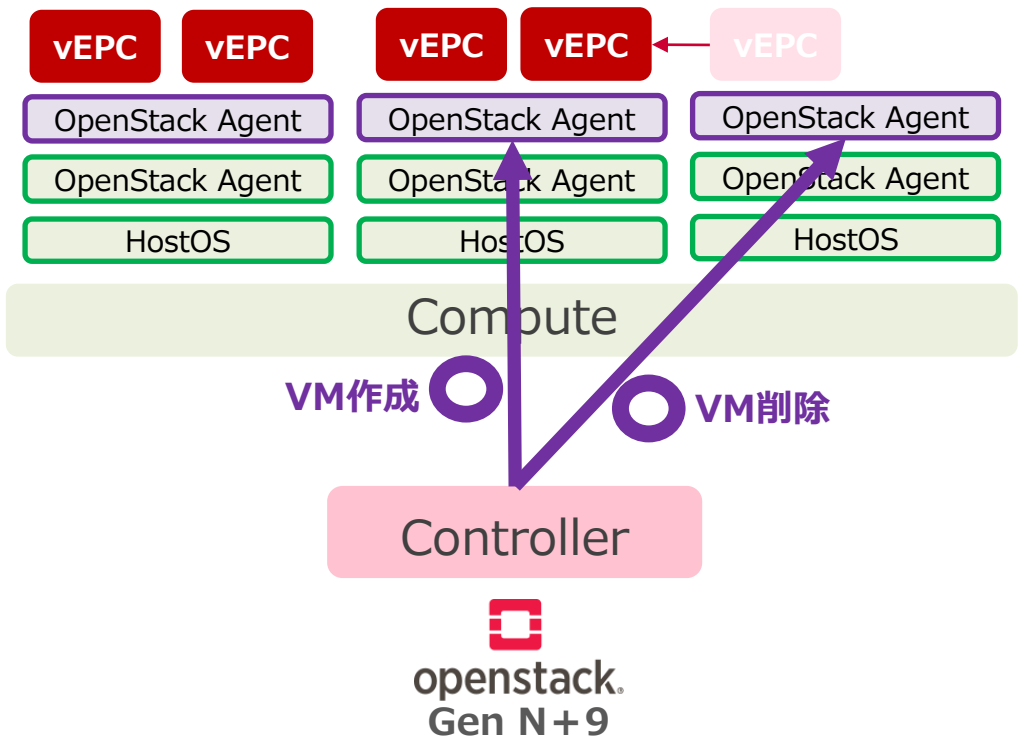
新旧世代間でControllerノードとComputeノードのサービスがサポートされておらずローリングアップグレードの肝となるVM移動が出来ない。

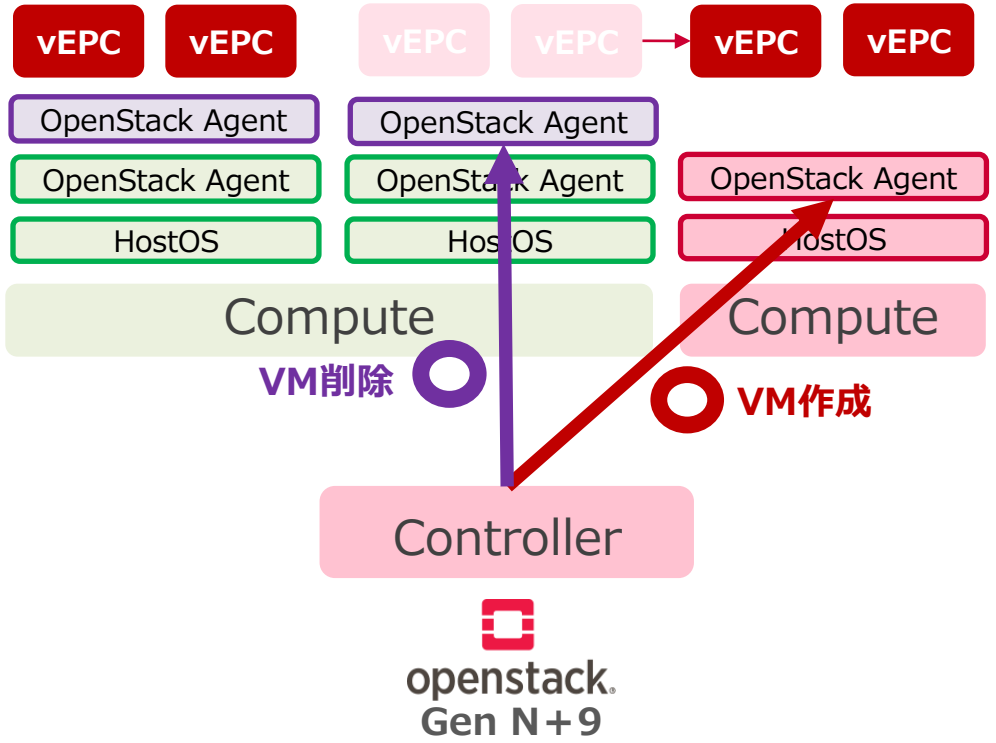
その為、VM移動を実現するための互換Agentを独自に開発することでこれの解決を目指した。





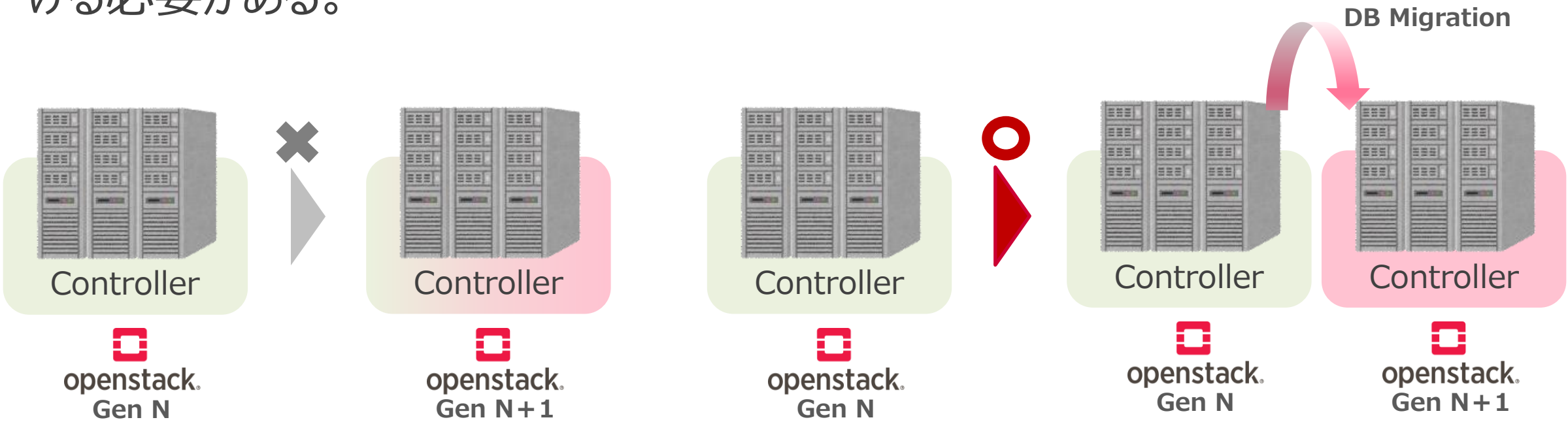
旧世代のCompute上に互換Agentをインストールし、新世代のControllerからの通信対地とすることでVM移動を実現。





同一Controllerノード上でのアップグレードがサポートされない為、Controllerノードを一時的に2世代デプロイする方式を選択し、そのうえで旧世代からデータベースを移行した。

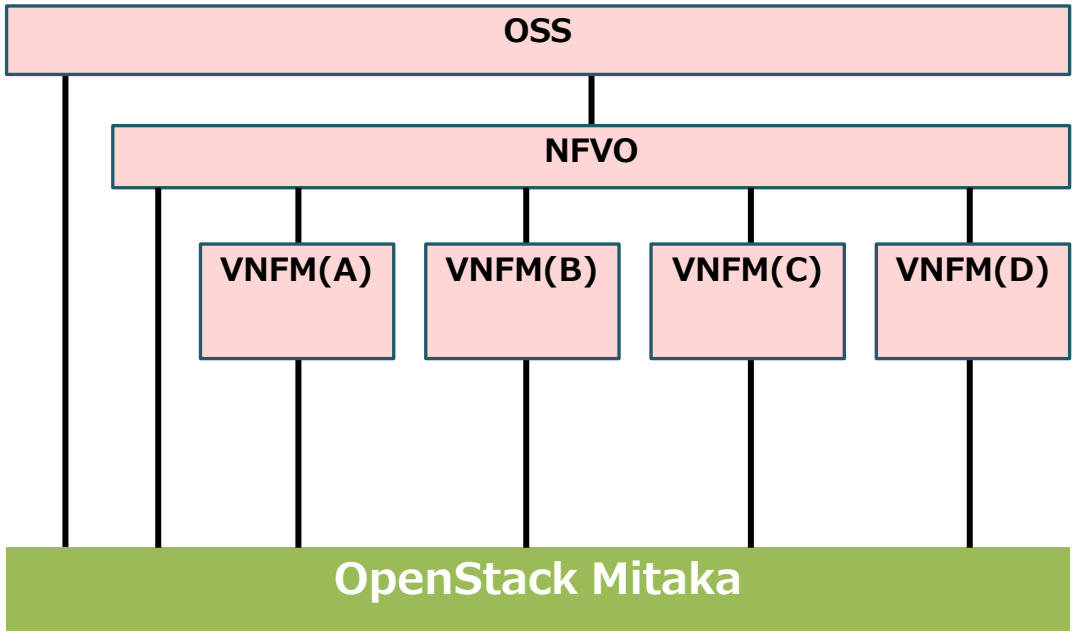
データベースの移行においては、世代間のスキーマ構成差分を踏まえた変換が必要な点や移行中は運用中のデータベースが更新されないように運用制限を掛ける必要がある。



次にAPI差分の課題と対策を説明していきます

5-4. API差分への対応

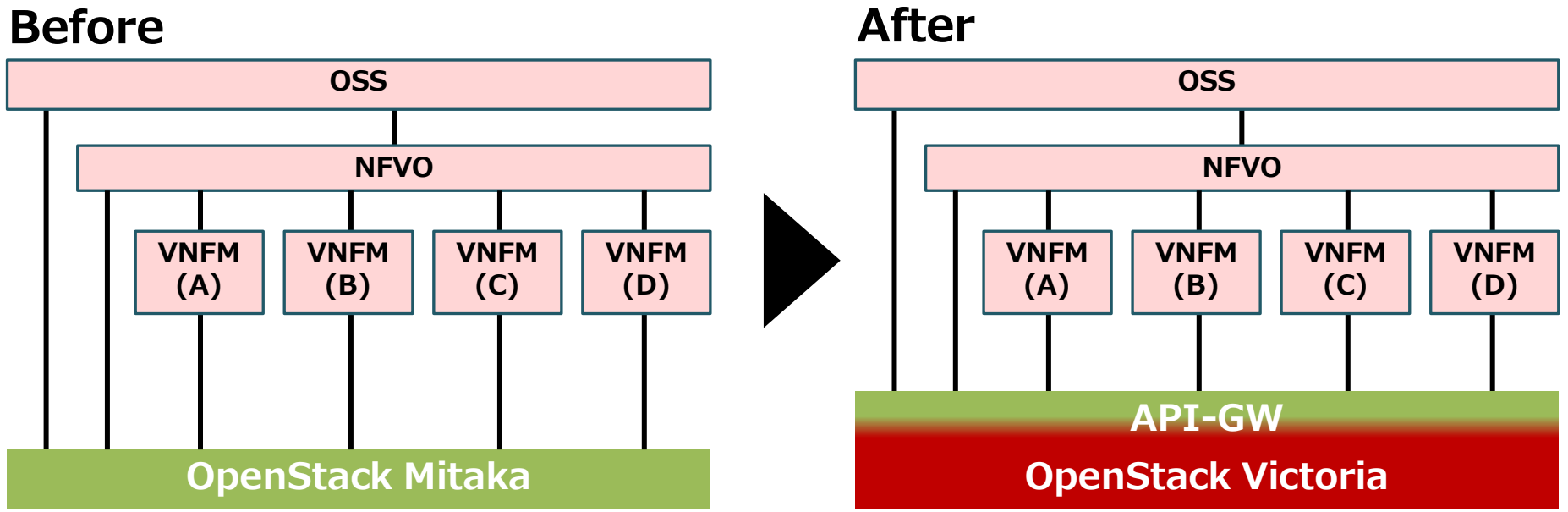
- 本課題は複数世代Skipならではの課題という訳ではありませんが、世代差が広がるほど遭遇する可能性が高くなると思われます。
- 一般的にAPIを発行するオペレーション、もしくはシステム側で新たなAPIに対応するのが基本ですが、ドコモの仮想化基盤は下図のようにAPIを発行するのは全てシステムで自動化されています。



5-4. API差分への対応

その為、新たなAPIやパラメータに対応する為には複数の上位システムで追従開発が必要になります。

これは開発期間やコストの観点で最適とは言えない状況であったため、ドコモでは下図の用にOpenStack側で世代間のAPI差分吸収するGWシステムを開発することで開発期間やコストの最適化に取り組みました。

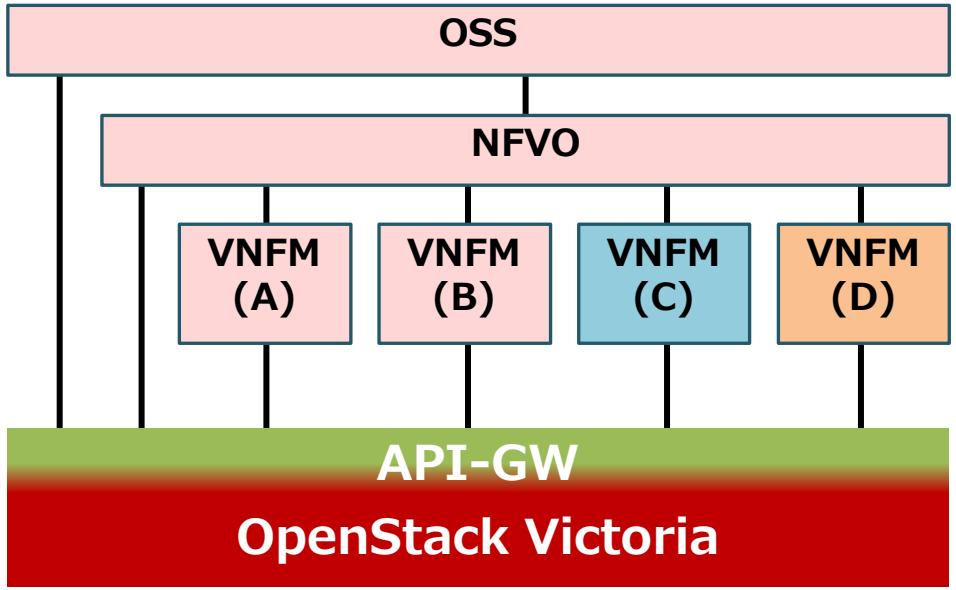


ところが・・・

APIを発行しているシステム(≒サプライヤ)自身が発行しているAPI、パラメータを厳密に把握していないという問題に直面した。

OpenStackからのResponseを参照する動作仕様も不明というケースもあり、机上確認、実機からダンプデータを取得する等で上流工程での確認には限りがあり、仕様・設計上の完全性は求めるべくもなかった・・

「7割程度の精度の設計」「実機対向試験は問題が出ることが前提」の方針で取り組んだ。



：IF仕様の開示可能かつ規定済

：**IF仕様の開示不可**

：収容VNF種が多く、**APIの網羅性懸念あり**

複数世代Skipしてしまうと・・・

➢ 根本的にOpenStackとしてローリングアップグレードが提供されていない

1. ローリングアップグレードを実現する独自の方式が必要　→**Controllerを増設する方式で解決**
2. 世代が開いており、新旧Controller～Compute間でおしゃべり出来ない　→**互換Agentを開発**
3. 世代が開いており、データベース構造が変わっている　→**運用制限を掛けながらデータ変換・移行**

➢ 世代が新しくなりすぎるとAPI差分が大きい

4. 使っていたAPI VersionがDeleteされたり、オプションパラメータが増えたり減ったりしている・・

   →**後方互換性を提供するGWを開発することで解決**

現在、商用で順次アップグレード中

ですが、今後の課題あり。

順に課題と対策(今後検討)を説明していきます

アップグレード手順概要と所要時間を比較すると、複数世代Skip更改では一部の工程の自動化を導入したが、**更改日数が2倍**。また、**保守作業不可日数も2倍**。

		保守作業不可		
#	**複数世代Skipの更改**	**日数**	**同世代/N+1世代の更改**	**日数**
1	新世代Controller新設	6		
2	データベース移行（1回目）	1		
3	自動VM制御停止	1	自動VM制御停止	1
4	上位システムとの通信を閉塞	4	上位システムとの通信を閉塞	2
5	互換Agentのインストール		Controllerのアップグレード	
6	新世代Controllerへサービス切り替え			
7	上位システムとの通信を開放	1	上位システムとの通信を開放	1
8	自動VM制御再開		自動VM制御再開	
9	Computeのローリングアップグレード	7	Storageのアップグレード	2
10	Storageのアップグレード	5	Computeのローリングアップグレード	5
	Total	**25**	Total	**12**
	保守作業不可	**5**	保守作業不可	**2**

以下の課題、抱えてます。

更改計画の課題

1. 基盤数が増加＆1基盤の更改期間が長期化&VNF側の制約の3点から商用更改がかなり長期に。。
2. 基盤更改中はVNF関連作業の不可期間があり、日/週単位のVNF側との工程調整が大変

更改作業時の課題

1. 更改作業中に発生した更改対象でなくとも物理故障による進行不可になる工程が複数ある。
2. 大量の監視アラーム発生により、作業アラームなのか、問題事象なのか？の切り分けが困難。
3. Controllerの切り替え時の保守作業不可期間が想定よりも長時間。
4. データベース移行は保守作業不可であるが、長時間＆ばらつきが大きい＆時間を推定する指標が無い。

他システムが関係する課題

1. 物理故障時、手動での復旧作業と自動化ツール側で再開に向けた準備作業が必要
2. 自動化ツールのエラー停止時のログを見ても、停止した原因がわかりづらい
3. 複数ロットを並行でローリングアップグレードする際に上位システム(VNFM等)の性能上限がネックになる

- 基盤数が増加、1基盤の更改期間が長期化

  →2年間の商用更改作業を計画し、実施中

- 商用更改期間とHW/SWのEoLを考慮すると、

  商用更改作業を実施しながら、次の更改開発！

  常に更改を考え続けるループにはまる..

6-3. 更改計画の課題 2/2

- 基盤更改中はVNF関連作業の不可期間があり、日/週単位のVNF側との工程調整が大変。（**※箇所**）

→調整の工夫として、

- #4 開始前で数日更改停止したり
- #9 アップグレードの途中で更改停止したり

結果として、基盤更改期間も延伸しうる

#	複数世代Skipの更改	日数
1	新世代Controller新設	6
2	データベース移行（1回目）※	1
3	自動VM制御停止	1
4	上位システムとの通信を閉塞 ※	4
5	互換Agentのインストール ※	
6	新世代Controllerへサービス切り替え ※	
7	上位システムとの通信を開放	1
8	自動VM制御再開	
9	Computeのローリングアップグレード ※	7
10	Storageのアップグレード	5
	Total	**25**

6-4.更改作業時の課題 1/4

- 更改作業中に発生した更改対象でなくとも物理故障による進行不可になる工程が複数ある。（※**箇所**）

→事前に故障サーバを0にしておきたいが、下記の理由により大変

  - 更改中は故障対応のパターンも多い
  - 復旧後の再開判断もパターンが多く複雑

#	複数世代Skipの更改	日数
1	新世代Controller新設　※	6
2	データベース移行（1回目）	1
3	自動VM制御停止	1
4	上位システムとの通信を閉塞	4
5	互換Agentのインストール　※	
6	新世代Controllerへサービス切り替え	
7	上位システムとの通信を開放	1
8	自動VM制御再開	
9	Computeのローリングアップグレード　※	7
10	Storageのアップグレード　※	5
	Total	**25**

6-4.更改作業時の課題 2/4

- 大量の監視アラーム発生により、作業アラームなのか、問題事象なのか？の切り分けが困難。

  - ComputeとStorageのアップグレード中は、アラーム通知漏れがないように、アップグレード前後両方で監視設定するため、2重発報されるケースがある。
  - ローリングアップグレード中はサーバの電源ON/OFFが頻発

#	複数世代Skipの更改	日数
1	新世代Controller新設	6
2	データベース移行（1回目）	1
3	自動VM制御停止	1
4	上位システムとの通信を閉塞	4
5	互換Agentのインストール	
6	新世代Controllerへサービス切り替え	
7	上位システムとの通信を開放	1
8	自動VM制御再開	
9	Computeのローリングアップグレード	7
10	Storageのアップグレード	5
	Total	**25**

- Controllerの切り替え時の保守作業不可期間が想定よりも長時間。
  - 上位システムとの通信解放前には、開発時の問題対処パッチ適用が必要だが、パッチ適用に長時間がかかり、保守作業不可日数も増加。

#	複数世代Skipの更改	日数
1	新世代Controller新設	6
2	データベース移行（1回目）	1
3	自動VM制御停止	1
4	上位システムとの通信を閉塞	4
5	互換Agentのインストール **＋パッチ適用（2日間）**	
6	新世代Controllerへサービス切り替え	
7	上位システムとの通信を開放	1
8	自動VM制御再開	
9	Computeのローリングアップグレード	7
10	Storageのアップグレード	5
	Total	**25**

- データベース移行は保守作業不可であるが、長時間＆ばらつきが大きい＆時間を推定する指標が無い。

開発時にデータベース以降の保守作業不可は認識。
→不可期間を分散するため、2回に分割。

ただ、1回目の時間がRegionによって異なり、作業計画が困難という課題あり。

商用の3Regionの結果：

1. 1回目：10時間44分、2回目：17分
2. 1回目：11時間32分、2回目：25分
3. 1回目：　4時間32分、2回目：28分

#	複数世代Skipの更改	日数
1	新世代Controller新設	6
2	データベース移行（1回目）	1
3	自動VM制御停止	1
4	上位システムとの通信を閉塞 **＋データベース移行（2回目）**	4
5	互換Agentのインストール	
6	新世代Controllerへサービス切り替え	
7	上位システムとの通信を開放	1
8	自動VM制御再開	
9	Computeのローリングアップグレード	7
10	Storageのアップグレード	5
	Total	**25**

6-5. 他システムが関係する課題 1/2

- 物理故障時、手動での復旧作業と自動化ツール側で再開に向けた準備作業が必要
- 自動化ツールからエラー停止時のログを見ても、停止した原因がわかりづらい

手動更改だったら → 一部不要

自動化ツールのエラー確認＋基盤のエラー確認

HW交換（現場対応）

HW復旧作業（手動）

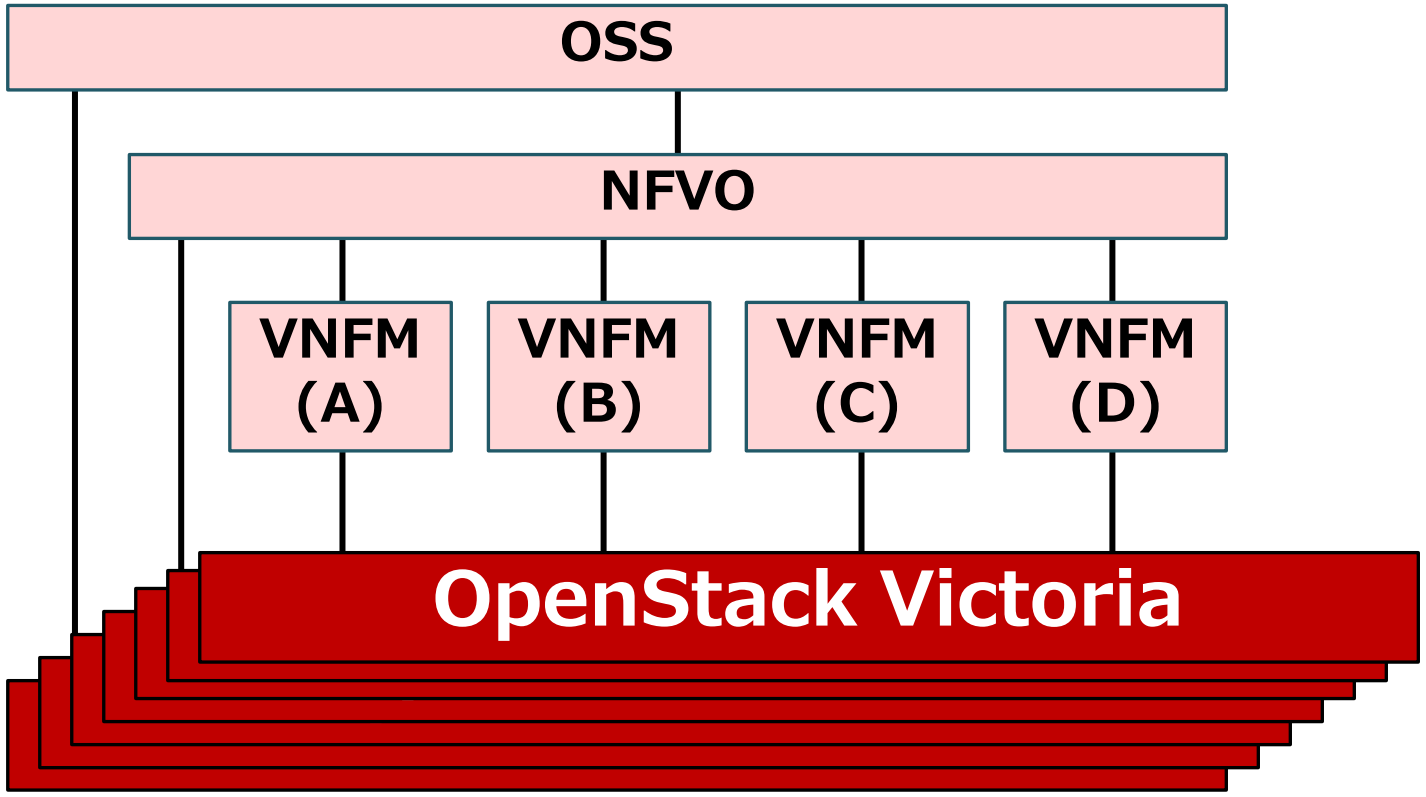
手動更改だったら → 不要

自動化ツールの設定作業（手動）

#	複数世代Skipの更改	日数
1	新世代Controller新設	6
2	データベース移行（1回目）	1
3	自動VM制御停止	1
4	上位システムとの通信を閉塞	4
5	互換Agentのインストール	
6	新世代Controllerへサービス切り替え	
7	上位システムとの通信を開放	1
8	自動VM制御再開	
9	Computeのローリングアップグレード	7
10	Storageのアップグレード 故障	5→2
	Total	**25→27**

6-5. 他システムが関係する課題 2/2

- 基盤数が増え、総VM数も多くなったため、複数ロットを並行でローリングアップグレードする際に上位システム(VNFM等)の性能上限がネックになる。



Openstack基盤は増加の一途であるが、VNFM等は1~数ロットのまま。

更改中は、通常時ではありえない数のVM Healing数が必要。

ここまでのまとめ

1. ドコモのネットワーク仮想化基盤の概要 → **全国24拠点、Openstackの基盤**
2. ドコモのネットワーク仮想化基盤の歴史 → **色々な更改をしてきました**
3. ドコモのネットワーク仮想化基盤の更改方針 → **サービス無中断は絶対！ローリングアップグレードへ**
4. 一般的なローリングアップグレード方式
5. Openstack複数世代Skipの課題と取り組み
    1. Openstack複数世代Skipのローリングアップグレード → **Contorollerを増設**
    2. 新旧Controller～Compute間でおしゃべり出来ない → **互換Agentを開発**
    3. データベース移行 → **運用制限を掛けながらデータ変換・移行**
    4. API差分 → **GW開発。机上検討での不完全性を許容**
6. 開発完了後の話 → **9つの課題あり**
7. まとめと議論ポイント

仮想化基盤のソフトウェア・ハードウェア更改でお困りではないですか？

- 商用では、仮想化基盤を長期に利用したいため、OSSやSWサポート期間内に仮想化基盤の商用更改を実施するのは現実的でない
- 短期間での商用更改を避けた場合、SWバージョンの複数世代Skipは不可避..

1. ローリングアップグレード方式で更改派
   → 複数世代のSWバージョンSkipされてますか？
2. Blue/Green方式で更改派
   → 更改に必要な予備設備をどのくらい確保されてますか？
3. 更改しない派！保守延長/自社保守してる！
   → 実現されてる方はいますか？

＃他に将来に向けた課題で気になる点あれば、議論したいです

更改期間や開発頻度はどの程度でしょうか？

→　商用更改＆更改開発しつづける状態なっていないでしょうか..？

準正常・異常系手順はどこまで開発時に確認し、どこから運用対処してますか？

→　ドコモでは各工程における復旧手順概要の検討まで。

更新作業のオペレーションの自動化をしていますか？

→　ドコモでは正常系の自動化までで、準正常系は対象外としてます。