ハイブリッドクラウドガイドライン

Ver. 1.0　2018年11月7日

ハイブリッドクラウド研究会

1. はじめに

1.1. 本ガイドラインの目的と利用方法

本ガイドラインは組織に対してハイブリッドクラウドを導入する際の指針を示すことを目的としています。特に日本企業がクラウドを導入していく際に現実的に参考になるガイドラインとなることを目的としています。

本ガイドラインでは「ハイブリッドクラウドの最適な活用方法はそれぞれの組織によって異なる」という考え方のもと、唯一無二のハイブリッドクラウド活用方法の解説という形ではなく、ハイブリッドクラウド活用の「スタイル」を定義しそのスタイルの中での活用ポイントや実際の利用例等を紹介しています。

自組織の現状のスタイルを選択、識別いただき、そのスタイルにおけるハイブリッドクラウドの活用例を自組織に適用していってもらいたいと思います。さらにはよりハイブリッドクラウドを活用できる別のスタイルへと自組織を導くガイドとしても利用いただければと思います。

1.2. 本ガイドラインの推奨される読み方

本ガイドラインは頭から通して読んでいただくことも可能ですが、すぐに活用いただくために下記のような読み方を推奨します。どの場合でもスタイルマップを別途すぐに見ることができる状態にして参照しながら読み進めていただくことをお勧めします。

自組織ですぐに活用できるユースケースを知りたい場合

1. スタイルマップを元に、現在の自組織が選択しているスタイルを確認する。
2. 該当のスタイルに紐付くユースケースを確認し、適用することを検討する。

自組織にてさらにクラウドを活用するためのロードマップを検討したい場合

1. スタイルマップを元に、現在の自組織が選択しているスタイルを確認する。
2. スタイルマップ上で次に自組織が選択したい、あるいは選択可能なスタイルを確認する。
3. 該当のスタイルに紐づくユースケースを確認し自組織への適用を検討する。

ハイブリッドクラウド自体の活用方法や考慮点を知りたい場合

1. スタイルマップを参照しながら、冒頭から読む。

2. 目次

1. はじめに
    1. 本ガイドラインの目的と利用方法
    2. 本ガイドラインの推奨される読み方
2. 目次
3. ハイブリッドクラウド概論
    1. ハイブリッドクラウドとは
    2. ハイブリッドデータセンターとハイブリッドクラウド
    3. クラウドネイティブシステム対応の重要性
    4. オンプレミス上でクラウドネイティブシステムを構築する際の基盤
    5. 一貫性のあるハイブリッドクラウド環境構築のアプローチ
    6. プライベートクラウドの構築および運用管理
    7. ハイブリッド環境の具体的な使い方と「スタイル」
4. スタイルマップ
5. ハイブリッドクラウドスタイル選択の考慮点
    1. スタイル選択ポイント
    2. 共通考慮ポイント
6. ハイブリッドクラウドスタイル
    1. オンプレミス環境のみ(物理および仮想)
    2. オンプレミス環境のみ(クラウドネイティブシステム)
    3. クラウドを補完的に利用
    4. データセンター拡張(データはオンプレミス)
    5. データセンター拡張(データもクラウド配置可能)
    6. ハイブリッドなクラウド基盤を利用したシステム
    7. クラウド環境のみ(レガシーシステム)
    8. クラウド環境のみ(クラウドネイティブシステム)
7. ハイブリッドクラウドユースケース
    1. レガシーシステムのクラウドネイティブ化
    2. 機密データをオンプレミスに配置しパブリッククラウドとシームレスに連携
    3. クラウドエッジ
    4. 高速な金融取引のためのハイブリッド構成
    5. クラウド災害対策
    6. Hybrid DevOps
    7. データ主権とデータグラビティ
    8. 分析のための階層化データ
    9. エッジAI
    10. 地域分散アプリケーション
    11. 医療機関Webサイト
    12. Oracleデータベースのマルチクラウド利用
    13. クラウドによるオンプレミスDBの分析と可視化
    14. EAI利用によるクラウドをまたいだデータの利活用
    15. ファイルサーバーのスモールスタートとハイブリッド化

16. ログ収集および分析基盤のクラウド上への構築
17. 自動処理のクラウド化
18. プライベートクラウド構築作業からの解放
8. 用語集
9. ハイブリッドクラウド研究会の活動への参加の誘い
10. あとがき的なもの

3. ハイブリッドクラウド概論

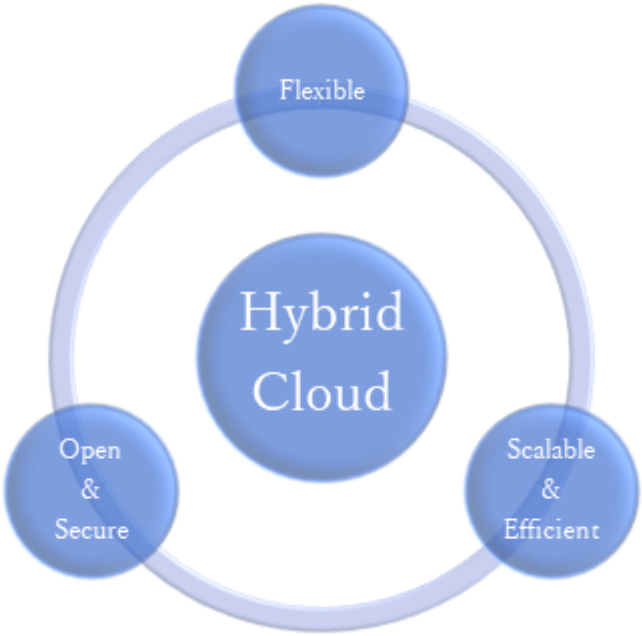
この章ではハイブリッドクラウドという言葉の本書での定義から始まり、ハイブリッドクラウドの特徴や長所、ハイブリッドクラウドとハイブリッドデータセンターという言葉の意味の違い、レガシーシステムとクラウドネイティブシステムの特徴やそれを動かす基盤等、ハイブリッドクラウドおよびハイブリッドクラウドに関連する事柄に関しての概要を記載します。

3.1. ハイブリッドクラウドとは

本書ではハイブリッドクラウドは、「オンプレミスとパブリッククラウドとを組み合わせることにより、データおよびアプリケーションの最適化を実現するためのテクノロジ」であると定義します。

データやアプリケーションをパブリッククラウドとプライベートクラウド間でシームレスに連携することで、自組織のビジネスに高い柔軟性や拡張性を提供して、インフラストラクチャ、セキュリティ、コンプライアンスの最適化を実現します。

本書では上記前提を念頭に、以下3つの要素をハイブリッドクラウドによって得られるメリットの柱として提示します。



3.1.1. 1. Flexible

オンプレミスとパブリッククラウドを組み合わせて、組織における全ての要求に対応可能な柔軟性を備えたITシステムを実現する

3.1.2. 2. Scalable & Efficient

オンプレミスとパブリッククラウドを組み合わせて拡張性、コスト効率の最大化を実現する

3.1.3. 3. Open & Secure

ベンダーに囚われないオープン性とパブリッククラウドによる強固なセキュリティの両立を実現する

下記は3つの柱に対する補足説明です。

3.1.4. 1. Flexible

組織の様々なシステム(レガシーシステム、クラウドネイティブシステム、膨大なリソースを必要とするシステム)を最適なロケーションに配置する事で、全てのビジネスニーズに対応する基盤を実現する。一例としてクラウドの持つ先進性や強力なパフォーマンスを必要としないLOB(基幹業務システム)などのレガシーなシステムは自社に配置し、ビジネススピードが求められるクラウドネイティブシステムはクラウドに配置するような使い分けが考えられます。

3.1.5. 2. Scalable & Efficient

オンプレミスとパブリッククラウドの組み合わせ方には様々な方法が存在する。例えば、いくつかの例として下記のようなケースが考えられる。

- 自社のもつ製品やサービスを展開する際、開発環境をオンプレミス、本番環境をクラウドに配置する事で複数のクラウドに跨った本番環境に対する迅速な展開を可能とする。
- オンプレミスの1環境をハブとして、単一のパブリッククラウドサービスだけでなく複数のクラウドサービスを結びつける。
- ビジネス要件、環境によっては本番環境をオンプレミス、開発環境をパブリッククラウドに配置する。

状況に応じて柔軟にオンプレミスとパブリッククラウドを組み合わせることで膨大なリソースを迅速に利用可能とする拡張性の確保とコスト効率の最大化が実現できる。複数のパブリッククラウドを同時に利用することも選択肢となります。

3.1.6. 3. Open & Secure

単一のクラウドサービスにロックイン(サイロ化)せず、複数のクラウドサービス利用をも可能とし、場合によってはパブリッククラウドサービスの切り替えやオンプレミスとパブリッククラウド間で自由にシステムを移動可能であるような柔軟性の高いシステムを実現する。同時に複数のクラウドサービスが提供するセキュリティ機能を活用、連携を実現する。パブリッククラウドによって提供されるセキュリティ対策サービスは多数の利用者や多数のサービス自体の運用から得られるビッグデータを元に機械学習等も用いて提供されており、オンプレミスにて自社組織で行うセキュリティ対策よりも高度で脅威に対しての対応速度が早いものとなる。

本書では、ハイブリッドクラウドの利用により享受できるメリットとしてこれらの3つの柱を原則としながら、ハイブリッドクラウドスタイル、ユースケースに関して述べていきます。

3.2. ハイブリッドデータセンターとハイブリッドクラウド

「ハイブリッドクラウド」という言葉は意味的に非常に広い言葉です。

前述のように本書では「ハイブリッドクラウド」という言葉を「オンプレミスとパブリッククラウドとを組み合わせることにより、データおよびアプリケーションの最適化を実現するためのテクノロジ」と定めています。ですが、ハイブリッドクラウド研究会内の議論においても単純にオンプレミスとパブリッククラウドをネットワーク的に接続し、仮想マシンだけを従来とまったく同じシステム構成で稼働させている状態の利用方法に対して「それをハイブリッドクラウドと呼んでいいのか?昔からある単なるホスティングと同じではないか?」という議論がありました。

一般にはこのような単純なホスティングと同等の方法で利用する場合も含めてハイブリッドクラウドと呼ばれている現状があります。ですが、本書では分かりやすさとハイブリッドクラウド活用スタイルの違いを表すために「ハイブリッドクラウド」という言葉を特徴に合わせて2つの言葉に分割し、もう一つ「ハイブリッドデータセンター」という単語を導入します。

「ハイブリッドデータセンター」はハイブリッドクラウドの一形態として、オンプレミスのデータセンターにパブリッククラウド連携の要素を追加したものです。主にWindows ServerやLinux等のオペレーティングシステムを稼働させながらパブリッククラウドのサービスを利用するスタイルを指します。発想としては従来からあるデータセンターや仮装基盤にパブリッククラウドの要素を付け加えるものであり、オペレーティングシステムを強く意識した言葉です。オンプレミスとパブリッククラウドをネットワークで接続し、ホスティングサービス的に利用するスタイルはハイブリッドクラウドの1形態ですが、その特徴を踏まえて本書ではハイブリッドデータセンターと呼びます。

「ハイブリッドデータセンター」と対比して「ハイブリッドクラウド」は発想としてパブリッククラウドがまず前提としてあり、そのパブリッククラウドの概念やコンセプト、システムの構築方法をオンプレミスにも拡張したものです。IaaSも主要なクラウドサービスの1つであるためハイブリッドクラウドにおいても仮想マシンおよびオペレーティングシステムを意識したシステム構築も可能ですが、ハイブリッドクラウドにおけるオンプレミスの基盤はパブリッククラウドを発端とするPaaSやサーバーレスアプリケーションを実現するためのサービスが提供される基盤であることが想定および期待されます。パブリッククラウドでのシステム開発に慣れた開発者がパブリッククラウドのサービスやAPI、セルフサービスの仕組み等を用いてパブリッククラウドと同じスピード感を持ってシステム構築が行えるクラウド基盤をオンプレミスにも持っている状態、これがハイブリッドクラウドです。

この「ハイブリッドデータセンター」と「ハイブリッドクラウド」は明確に線引きできるものではありません。そのあいだには連続的な関係がありますし、基盤のありかた、基盤上に構築されるプラットフォームのありかた、その上で構築されるシステムのありかたによって様々な形態がありえます。ですが、データセンターの観点から発想、着眼するポイントとパブリッククラウドの観点から発想、着眼するポイントには大きな差異があります。それを表現する単語の違いと考えていただければと思います。

なお、「ハイブリッドデータセンター」および「ハイブリッドクラウド」という言葉はMicrosoftもWindows Server 2019の文脈で利用しています。Windows Server 2019, System CenterはAzureと合わせて「ハイブリッドデータセンター」を実現するものであり、Azure StackはAzureと合わせて「ハイブリッドクラウド」を実現するものであるとされています。

クラウド、ハイブリッドクラウド、ハイブリッドデータセンター、ハイブリッドIT等、様々な単語がありそれぞれ話し手、受け手によって解釈が異なる状況があります。

本ハイブリッドクラウド研究会では曖昧なこれらの要素を「スタイル」という形でさらに分類し特徴づける事で相互理解と活用方法の明確化を目指します。

3.3. クラウドネイティブシステム対応の重要性

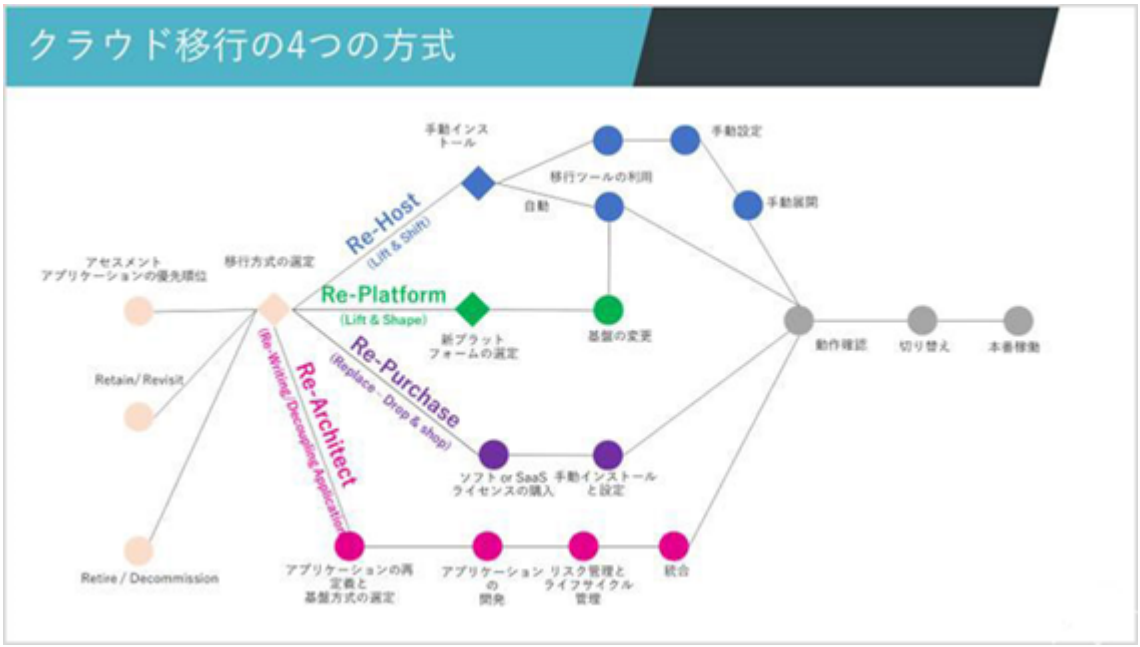
本ガイドラインでは「レガシーシステム」と「クラウドネイティブシステム」も大きな1つのスタイル選択ポイントとして扱っています。

本書では従来からある仮想マシン中心に構成されたシステムを「レガシーシステム」とよび、クラウドサービスを効果的に使うコンテナやサーバーレス中心のシステムを「クラウドネイティブシステム」と呼んでいます。

現在のシステムアーキテクチャ関連の最先端の話題はコンテナ、サーバーレス等クラウドネイティブシステムにカテゴリ分けされるシステムをどのように作成するか、という部分にかなりフォーカスがあたっている状況があります。

クラウドの活用の観点でも同一システムに対してRe-Host, Re-Platform, Re-Architectのステップを踏み、クラウド活用を促進させる考えもあります。

1. 一度クラウドに仮想マシンレベルで移動し稼働させる(Re-Host)
2. クラウドプラットフォームで提供されているサービスを活用しシステムを効率化する(Re-Platform)
3. アーキテクチャレベルで見直しモダンなアーキテクチャで再構築する(Re-Architect)



オリジナル引用元：[Cloud-Native or Lift-and-Shift? – AWS Enterprise Collection – Medium]

全てのシステムがクラウドネイティブシステムになることが最適解ではありません。システムによって最適なアーキテクチャ、開発方法、更新サイクルは異なります。従来のやり方で使い続けることが最適なシステムも確かに存在します。

一方で、クラウドプラットフォーム上で無駄を省き迅速にクラウドネイティブなシステムを構築、改善しビジネス状況の変化に合わせてシステムも継続的に変化し続けることこそが求められるものも多く存在していることもまた事実です。クラウドネイティブなシステムアーキテクチャを理解および把握できておらず選択する能力が無い中で「選択できない」ことと、きちんと把握し実装および運用することができるうえでシステムの特徴から「(選択するメリットがないため)選択しない」ことには大きな違いがあります。レガシーシステムおよびクラウドネイティブシステムのアーキテクチャの違いや特徴の違い、システムの向き不向きをきちんと抑え、しっかりと選択および場合によっては実装できることは大きな力となります。

3.4. オンプレミス上でクラウドネイティブシステムを構築する際の基盤

前述の通りクラウドネイティブシステムへ対応しそのスピードをビジネスに活かして行くことは現在の全ての組織において重要な事です。その上で、組織により、システムにより、その適切な配置場所は変わってきます。パブリッククラウド上にクラウドネイティブシステムを配置する、オンプレミス上にクラウドネイティブシステムを配置する、パブリッククラウドとオンプレミスにまたがってクラウドネイティブシステムを配置する。全てのパターンに容易に対応できるその実行基盤が求められます。

パブリッククラウド上でクラウドネイティブシステムを作成することはサービスが整備されている状況があり比較的容易ですが、オンプレミス上でクラウドネイティブシステムを作成することは

比較的困難です。多くの組織はオンプレミスに物理サーバーおよび仮想基盤は保持していても、「クラウド基盤」を保持していないことが多いためです。

クラウドネイティブシステムを配置するためのオンプレミスの「クラウド基盤」には、仮想マシンの実行基盤機能はもちろんのこと、コンテナの実行基盤やサーバーレスエンジンの実行基盤も必要です。Webサイトの実行サービスや自由にアドレス空間を設定できる仮想ネットワークサービス、ロードバランシングサービス等も必要です。セルフサービスで処理を行えるポータルや、プログラムから連携できるAPIも必要です。リソースの利用状況を計測し、利用状況の可視化と課金管理ができる機能も必要です。理想はパブリッククラウドと全く同じサービスが全く同じように利用することができ、全く同じように管理作業が行なえ、そもそも違いを意識する必要が無いことでしょう。

パブリッククラウドとオンプレミスのプライベートクラウドに一貫性があり、クラウドネイティブシステムの実装が行える環境。それはまさに理想的なハイブリッドクラウド環境と呼べるでしょう。

3.5. 一貫性のあるハイブリッドクラウド環境構築のアプローチ

一貫性のあるハイブリッドクラウド基盤を手に入れるアプローチとして現在いくつかのアプローチがあります。

1. オンプレミスとパブリッククラウドとで一貫性のあるハイブリッドクラウド環境を提供する製品を利用する
2. 一貫性を保つ層を定義しその環境を複数の環境上に構築する

1は現時点ではAzureおよびAzure Stackによるハイブリッドクラウドのみが例として挙げられます。パブリッククラウドサービスを提供するメガクラウド事業者(ここではMicrosoft)自体がサーバー製品を提供するハードウェアベンダー(Avanade, Cisco, DellEMC, Hewlett Packard Enterprise, Huawei, Lenovo, Terra)と統合システムを構築し、Azureと一貫性を持つAzure Stackをオンプレミス環境に提供しています。

VMwareとVMware on AWSによるハイブリッド環境という選択肢もありますが、こちらはどちらかというとハイブリッドクラウドよりもハイブリッドデータセンターに近い形態と言えるでしょう。

GoogleからもパブリッククラウドのGoogle Kubernetes Engine(GKE)とオンプレミスのGKE-OnPremによるハイブリッドクラウドを実現する製品が2018年7月に発表されました。こちらはまだ発表直後の状況のため今後の動きに注目したいところです。

2の例としてはCloud FoundryやOpenShift等が挙げられます。これらはオンプレミスの物理あるいは仮想基盤上にOSベースで構築することもできますし、AzureやAWS上でIaaSを利用して構築する事もできます。構築された環境上では場所を問わずに一貫性のあるプラットフォームを構築することができます。特にコンテナベースのクラウドネイティブシステムの構築が行えます。付け加えるとこれらはAzure Stack上に構築することも可能です。

どの層で一貫性を保った環境を構築、保持するがよいのかは組織で利用したいサービスの違いや基盤管理をどこまで行いたいか等で異なってくるでしょう。

3.6. プライベートクラウドの構築および運用管理

オンプレミスでクラウドネイティブシステムを稼働させるためのプライベートクラウド基盤としては極力運用コストがかからないものが求められます。クラウドネイティブシステム自体がよりビジネススピードを向上させることに重きをおいたアーキテクチャであるため、プライベートクラウド基盤自体の構築や運用管理に時間がかかってしまうことは本末転倒であるためです。そして同時に、パブリッククラウドとの一貫性を保ちつづけるために簡単に更新し続けられる基盤であることも求められます。

この点に関しては統合システムとして内部が隠蔽されており、継続的に一貫性維持のためにUpdateが提供され続けるAzure Stackのユニークさが際立ちます。パブリッククラウド上でクラウドネイティブシステムを構築する際にはその物理的な基盤を意識する必要が無いように、オンプレミス上でクラウドネイティブシステムを構築する際にも極力その物理的な構成やクラウドシステムを維持する仮想マシン等のコンポーネントを意識して管理する必要が無い方が望ましいと考える組織も多いでしょう。

逆に全てのコントロールを自組織で行う必要があると考える場合にはオープンなものを採用するということになるでしょう。

3.7. ハイブリッド環境の具体的な使い方と「スタイル」

ここまでで、ハイブリッドクラウド自体のもつメリットから始まり、ハイブリッドクラウドとハイブリッドデータセンターの違いや、クラウドネイティブシステムの重要性、それらを実行するための基盤の話を説明してきました。

オンプレミスという場所とパブリッククラウドという場所がある中でその利用方法は様々な形態があります。そして組織によって置かれている状況や求められるものが異なることから、システムアーキテクチャに求められるものも異なります。

どれが良い、悪いということではなく、ハイブリッドクラウド研究会では「それぞれの組織が置かれた状況により最適なクラウドの利用方法は異なる」という考えのもと、その利用「スタイル」とそれぞれの「スタイル」に基づいた利用ユースケースを収集しました。

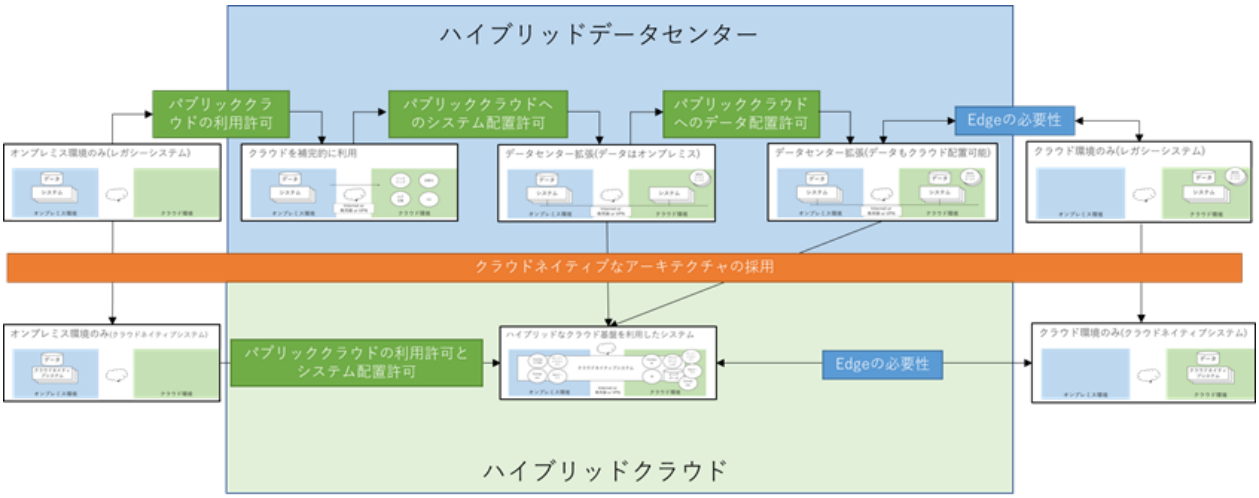
次の章からは具体的なハイブリッド環境の利用スタイル、スタイル間の関係性、スタイル選択時の考慮点、それぞれのスタイルに関連する利用ユースケース等について記載します。

4. スタイルマップ

本書ではまず「スタイルマップ」を提示します。スタイルマップは組織のクラウド利用のスタイルおよびスタイル間の関連を図として表現します。スタイルマップ内に書かれている要素の説明はこのあと本書のなかで行います。

現在自組織が選択しているスタイルからどのような点の考え方を変更すればどのようなスタイルに移行できるのかという関連性を認識しつつ、ハイブリッドクラウド活用のステージを引き上げる際の道標としていただければと思います。

ぜひこのスタイルマップをすぐに参照できる状態にした状態で本書を読み進めていただければと思います。



5. ハイブリッドクラウドスタイル選択の考慮点

ハイブリッドクラウドの構成方法、利用方法には様々な形態があります。

本ガイドラインではスタイルの特徴を決定づける「スタイル選択ポイント」とすべてのスタイルにて共通で考慮すべき「共通考慮ポイント」を定義しています。

「スタイル選択ポイント」および「共通考慮ポイント」はハイブリッドクラウド利用において常に重要なポイントであり、どのスタイルにおいても全て考慮されるべきです。本ガイドラインでは利用スタイルの分類のために「スタイル選択ポイント」と「共通考慮ポイント」と区分けをしているのみであり、その重要度や考慮すべき順番で区分けをしているわけでは無いことに注意してください。

スタイル選択ポイント

- システム配置場所
- データ配置場所(データ機密性)
- システムアーキテクチャ

共通考慮ポイント

- ネットワーク形態
- 法令遵守
- セキュリティ要件
- データ容量
- 遅延時間/応答時間
- ネットワークトラフィック量
- Updateおよびサポートライフサイクルへの追従ポリシー
- SLA / 可用性
- 価格/コスト
- 利用サービス
- システム利用者の場所
- 組織外部との連携(利用者/連携サービス)
- サービス、データ連携手段
- 開発手法/速度/リリース頻度/スキル/学習コスト
- 運用手法/コスト/スキル/学習コスト
- システムパフォーマンス
- パブリッククラウドへの接続性
- 可搬性
- ライセンスポリシー
- 認証システム
- システム稼働時間

まずこれらの要素の特徴を解説します。

5.1. スタイル選択ポイント

「スタイル」を決定づける要素とその選択肢に関して説明します。

5.1.1. システム配置場所

システムを配置する場所は具体的にどこに何を配置するかに直結し、システムアーキテクチャ上も非常に重要です。なお、本ガイドラインでは「システム」という表現の中にはデータは含めておらず特別扱いしている点に注意してください。データに関しては「データ配置場所(機密性)」という項目で独立させています。

オンプレミス

システム配置場所としてのオンプレミスは、システム設計者、管理者からすると過去の資産やノウハウが利用できるため選択しやすい選択肢です。オンプレミスであるため必ず物理的な機器の管理および運用を組織として実行する必要があります。ファームウェアバージョンまで含めてすべてをコントロールすることができますが、基盤としての安定した稼働や、不足リソースの充当や物理機器の故障時の対応等も自組織にて対応する必要があります。システムの動作基盤として仮想基盤を採用する場合には仮想基盤の構築、運用、管理、バージョンアップ等もすべて自組織で行う必要があります。

同時に物理的な機器を配置するためのスペース、空調設備、電源等ふくめ適切に管理される必要があります。自組織ですべて管理するケースから、データセンター事業者と契約し場所の提供を受けるケースもあります。

一般的にパブリッククラウドと比較した場合に自由度は非常に高くなる一方、管理工数が多くなることになります。

パブリッククラウド

システム配置場所としてのパブリッククラウドは、物理的な機器の管理をすべてパブリッククラウド事業者が行っており利用者はそれを意識する必要がない点がオンプレミスと決定的に異なります。ベアメタルで物理的なサーバー自体をサービス利用できるものもありますが、いずれの場合でも物理的な機器の管理責任はクラウド事業者であり、設置するためのスペース、空調設備、電源、利用する具体的なハードウェアの準備、故障時の対応、ファームウェア更新等含めてすべてクラウド事業者に任せることができます。仮想基盤の構築、運用、管理、バージョンアップ等もすべてクラウド事業者により行われます。

管理面で見ると非常に利用者が楽をできる構造ですが、逆にクラウド事業者によりすべてコントロールされており自由度はオンプレミスと比べて圧倒的に低くなります。

連携

システムは多数のコンポーネントからなり、コンポーネント同士が連携して動作します。システムの配置場所としてパブリッククラウドとオンプレミスを連携させる形態があります。これは単一のシステムを構成するコンポーネントがオンプレミスにもクラウドにも配置され連携して動作する形態もありますし、システムの配置をオンプレミスのみあるいはパブリッククラウドのみとし、災害やシステム障害等をトリガーとしてシステムの配置場所を切り替える形態もあります。

連携させる場合にはどちらにどのコンポーネントを配置するのか、何をトリガーに切り替えるのか、切り替えた後にもシステムが動作し続け、利用者からきちんとアクセス可能であるのか等を考慮する必要があります。

オンプレミスとパブリッククラウドの両方にコンポーネントを配置する中でコンポーネント間の通信に何を使うのか、コンポーネント間のネットワークをパブリックなものにするか、プライベートなものにするか等でバリエーションが分かれます。

オンプレミスとパブリッククラウドの両方を使うことで適材適所のコンポーネント配置が実現できることになります。場合によっては動作中にコンポーネントの配置場所が移動する形態も考えられるでしょう。

このように自由に配置するためにはオンプレミスとパブリッククラウドとで存在しているサービスや展開手法等に差異があることは好ましくありません。差異がある場合には学習コストが増加し、作業手順書もオンプレミスとパブリッククラウドで異なるものが必要になる等様々な箇所で負担が増えてしまいます。場所としてはどこであろうとも全く同じ手法で展開でき、全く同じ結果が得られることが望ましい事になります。このような一貫性をもった基盤であればシステムを適材適所で配置可能となります。

5.1.2. データ配置場所(データ機密性)

データ配置場所はスタイル選択に大きな影響を与える重要な要素です。どれだけ機密性の高いデータを扱うかという軸でデータの配置可能な場所が決まるケースが多くあります。

全データをオンプレミスに保持が必要

パブリッククラウドのコンピューティングリソースの利用は可能でも、データは必ずオンプレミス内に保持する必要があるケースです。業界によっては法令でデータ配置場所がオンプレミスに縛られているケースもありますし、組織としてルール決めがなされているケースもあります。

機密性の高いものはオンプレミスに保持が必要

データを機密性で区分し、機密性の高いものはオンプレミスに保持を必須とし、機密性が一定以上低いデータに関してはパブリッククラウドにも配置可能とするケースです。

機密性が高いデータはオンプレミスに配置しますが、機密性が高いデータであっても一部マスクする、オンプレミスでのみ可逆変換できるようにデータを保持した上でパブリッククラウドに配置可能とする等の手法を採用するケースもあります。

各種の暗号化技術を用いることでパブリッククラウドに配置可能なデータ区分を増やすルール決めを行っている組織もあります。

適材適所に配置

すべてのデータを場所にかかわらず適材適所で配置可能とするケースです。実際には自由に配置可能としながらも、災害時にのみクラウドを使う等、どちらかに重みをもたせるケースも見られます。

パブリッククラウドにデータを配置したほうがセキュアである？

多くの組織で「機密性の高いデータはパブリッククラウドには配置できない」というポリシーが持たれており、その実態を反映してデータ配置場所をこのように分類しています。この分類の暗黙の前提には「オンプレミスにデータを配置した方がパブリッククラウドにデータを配置するよりも安全である」という考え方があります。パブリッククラウドでは積極的にセキュリティに関する対策や認証取得情報を開示し安全性をアピールしていますが、仕組み自体がブラックボックス化となっている点がネックになっています。またデータの取り扱いに関して利用者側に自由度が無いということもありす。

ですがこの前提自体に疑問を提示する声もまた多くあります。「パブリッククラウドにデータを配置したほうがむしろセキュアである」という考え方です。

想定するデータの流出経路や原因によって様々な見解があると思いますが、メガパブリッククラウドのデータセンターのレベルで物理層の管理ができている組織は無いであろうということは事実としてあります。データセンター事業者自体にデータを盗まれるケースも一部想定されていると思いますが、そのケースはまだ確認されておらず、暗号化等を適切に用いることで技術的にも防ぐことが可能です。

実際にクラウド事業者はプラットフォームとしてセキュアであることの証明として各種の第三者認証を取得しています。認証の数という観点ではオンプレミスのシステムよりもパブリッククラウドのほうがはるかにセキュアであると認定されていると言えるでしょう。

一方で、パブリッククラウド上で誤った設定を行えば、簡単に重要なデータがインターネットに無防備にさらされ世界中からアクセス可能となってしまいます。そうなるように設定されているわけですから、これはパブリッククラウドプラットフォームがセキュアであるかどうかとは無関係です。この観点からオンプレミスでネットワーク全体の設計を安全になるようにしておけばパブリッククラウドよりも事故が起きずらいということは事実としてあります。ただ、このような形態を選択すると、パブリッククラウド利用のメリットを最大限に教授することができないことも事実です。

どこにリスクを想定するか、どのようなスタイルを組織として選択するかという点においてこのデータの配置場所が大きく影響していることは間違いない事実です。

5.1.3. システムアーキテクチャ

システムアーキテクチャは時代の流れとともにトレンドが移り変わるものですが、コンテナ技術の流行やクラウド及びクラウド上の「サービス」の登場により大きく変化してきています。

実際には一概にどちらであると全てのシステムを2つに完全に分類できるものではなく、様々な分類方法が考えられますが、ここではわかりやすさを優先し大きく「レガシーシステム」と「クラウドネイティブシステム」の2つに分類しています。

クラウドネイティブシステム

(オンプレミスかパブリッククラウドかを問わず)クラウド上でシステムを構築することを前提とし、クラウドが提供するサービスを活用したシステムをクラウドネイティブシステムと表現しています。

典型的に下記のような特徴があります。

- 仮想マシンへの依存度が低くコンテナ、サーバーレス、PaaSを有効活用した構成である。
- マイクロサービスアーキテクチャを採用している。
- DevOps、Infrastructure as Code, 継続的インテグレーション、継続的デリバリー等のプラクティスの実践を含めて展開速度が早く、継続的な変更が容易である。
- 運用の自動化、システムの信頼性向上のためのNoOps、Site Reliability Engineeringを採用している。

なお、クラウドネイティブなシステムはパブリッククラウド上でしか実現できないわけではなくオンプレミス上でクラウドネイティブなシステムを構成できる基盤を用いて実現することも可能です。

レガシーシステム

主に仮想マシン上にモノリシックなアプリケーションが展開されているシステムを「レガシーシステム」と表現しています。

典型的に下記のような特徴があります。

- ハードウェアとシステムが強く結びついている事が多い。
- 物理マシンあるいは仮想マシンによって構成されている。
- システムを構成するコンポーネントの単位が大きく、なにか変化させる場合に全体のテストが必要となることが多い。
- システムを頻繁に変更することはあまり想定されない。
- ウォーターフォールで開発されることが多い。
- オペレーターによる手作業での作業が多い。

「レガシー」という表現をしていますが、執筆時点では一般的にこの形態でのシステムが非常に多い現状があります。

5.2. 共通考慮ポイント

5.2.1. ネットワーク形態

ネットワーク形態もハイブリッドクラウド利用のスタイルを選択する上での重要な要素です。システムの配置場所、データの配置場所、さらに利用者が存在するネットワーク的な場所によって決定されるケースが多くあります。

ネットワーク形態には下記のように様々なバリエーションがあります。

- オンプレミスとパブリッククラウドの間にネットワーク的な接続も連携も無い形態
- オンプレミスとパブリッククラウド間をプライベートな接続(VPN, 専用線)で接続する形態
- オンプレミスとパブリッククラウド間をインターネットで連携させる形態

利用するクラウドのサービスによって必然的にネットワーク形態が決まってくるものも多くあります。

オンプレミスのデータセンターをクラウドにまで延長するようなコンセプトにおいてはインターネットを利用せず、プライベートな通信をクラウドとの間で可能とする形態がよく取られます。プライベートIPを用いて相互に通信できる状態となります。日本企業のパブリッククラウド利用の形態ではオンプレミスとパブリッククラウドとの間を専用線で接続するケースが世界の利用例と比較して非常に多い事が知られています。

一方インターネットを活用したネットワーク形態を取ることができれば全体的なコストを圧縮することや、システムを自由度高く適材適所に配置可能となる等のメリットもあります。

エンドユーザーがどこにいるのか、システムはどこに配置されているのか、認証システムは何を採用しているのか等の要素によっても必然的にネットワーク形態が限定される事もあります。

5.2.2. 法令遵守

特定の業種においては法律上の厳しい制約がありその制約の中でシステムを構築することが必須となります。特にデータの配置場所に制約があるケースが多くあります。

例えば改正個人情報保護法による制約などが挙げられます。

プラットフォームの選択やシステムアーキテクチャの構成はこれら制約の中で行う必要があります。

5.2.3. セキュリティ要件

セキュリティ要件は組織として採用しているものもあれば、業界として定義されているもの、法令として施行されているもの等複数の種類があります。セキュリティ要件は様々な要素に発展するため、ハイブリッドクラウドスタイル選択にも大きな影響を及ぼします。しっかりと把握したうえでの対応が求められます。

一方でオンプレミスにて適用しているセキュリティ要件を絶対的なものとし、それと全く同じものをクラウドにも適用しようとする組織が多く見られますが、それによってクラウドの可能性やメリットが大きく犠牲になる、オンプレミスと同じ使い方しかできない状態になってしまっている状況もあります。クラウド利用に際してセキュリティ要件も見直す必要も出てくるでしょう。

5.2.4. データ容量

データ容量はその生成場所、格納場所含めて慎重に検討する必要があります。

データ容量が少ない場合にはあまり制約はなく、自由にアーキテクチャを構成することが可能です。ですが、データ容量が大きい場合にはオンプレミスとパブリッククラウド間でデータを移動させるコストが大きくなるためデータを移動させないようにシステムを構築することが適切な場合が多くあります。特にパブリッククラウドはアップロードには課金されず、ダウンロードには課金される形態が多いためデータ移動時の課金額という観点にも注意が必要です。

またデータ容量がとても大きい場合にはデータを効率的に扱うための適切なアーキテクチャが必要となります。その基盤をオンプレミスで構築するのか、クラウドのサービスを選択するのかは重要な選択です。

データ容量が大きい場合にはその大量のデータを分析する等、データに紐づく後続処理が存在するケースも多くあります。その連携のしやすさという点も考慮が必要です。

5.2.5. 遅延時間/応答時間

遅延時間および応答時間はシステムのパフォーマンスや使い勝手、ひいてはシステム利用者の満足度に直結するため重要です。この項目はシステムの配置形態、データの配置形態、ネットワーク形態と密接に関連する部分です。システム自体の利用目的にも影響されます。

システムの利用者がオンプレミスに存在する場合にはシステムへのアクセスポイントをオンプレミスに配置したほうが一般的にシステムは高速に動作可能です。

クラウドにすべてのシステムを配置しつつ、その利用者がオンプレミスに存在する形態の場合にはネットワーク的な距離はオンプレミスとクラウドの間の経路分遠くなることになります。システムによっては専用線接続が事実上必須になるケースも考えられます。

一方システム利用者が主にインターネット上に存在する場合にはパブリッククラウドにシステムへのアクセスポイントを配置したほうが有利なケースも多いでしょう。さらにシステムの負荷が高まった場合のスケールアウトの容易さという点も観点として出てきます。この点もパブリッククラウド方が有利です。

いずれの形態にする際でもシステムがきちんと動作する上で必要な遅延時間、応答時間が達成できるアーキテクチャを選択しつつ、将来の利用量の変化にも対応可能にしておくことが必要です。

5.2.6. ネットワークトラフィック量

ネットワークトラフィック量はシステムおよびデータの配置場所とネットワーク形態に大きく関連する項目です。データ容量とも大きく関連します。

ネットワークトラフィック量が大きい場合にはそのトラフィックが発生する場所、経由する経路への考慮が必要です。特にオンプレミスとクラウドとの間に大量のネットワークトラフィックが発生するようなデザインはシステムパフォーマンスの観点からも課金の観点からも望ましくありません。

遅延時間/応答時間とも関連しますが、例えばオンプレミスで連続的に撮影しているカメラの映像をもとにリアルタイムに物体を認識してフィードバックを返すようなシステムを構築する際に、すべての生の動画データをクラウドに転送、分析した上で結果を返すような作りでは現実的に利用可能なシステムにはならないことが想定されます。このような場合には分析エンジン自体をオンプレミスに配置する等の考慮が必要となるでしょう。

5.2.7. Updateおよびサポートライフサイクルへの追従ポリシー

Updateおよびサポートライフサイクルへの追従ポリシーは現実的にシステムを配置可能な場所を限定します。常にシステムをUpdateし続け、サポートされる状態を維持しているのであればシステムを自由に配置可能ですが、システムを構築時のまま塩漬けとし、Updateを行わないような場合、パブリッククラウドには配置できないケースが出てきます。

仮想基盤部分の更新に関してもパブリッククラウドであればクラウド事業者によって強制的に行われるため、その影響を受けるケースも考えられます。オンプレミスであればタイミングはコントロールできますが、工数がかかります。このあたりをどのようにコントロールするのか、したいのか、によって選択可能なシステム配置場所が限定される事があります。

逆に、OSのメジャーバージョンアップを行っていないシステムに対して特定のパブリッククラウド上であればサポートが延長されるというケースが存在します。具体的にはWindows Server 2008のEOS後もMicrosoft AzureおよびAzure Stack上であればセキュリティパッチが3年間無償で継続提供されるという発表がなされています。これを目的としてシステムの配置先をAzure, Azure Stack上に移動するケースも考えられます。

5.2.8. SLA / 可用性

パブリッククラウド上のサービスには多くのケースでSLAの規定があります。あるいはSLAの規定がなくいつ何が起きても承諾するしかないケースもあります。そのSLAの範囲で行われる事が組織としてシステムとして受け入れ可能であればそのサービスは利用可能ですが、受け入れ可能でない場合には利用できないということになります。主にシステムやデータの配置場所やネットワーク接続形態の選択に影響します。SLAの中でも特に可用性に注目して利用の可否を判断するケースが多く見られます。

オンプレミスにおいて独自に構築、管理をすれば、一般的に目的のSLAおよび可用性を達成するように設計が可能です。その際に必要な工数、コストと、クラウドサービスを利用した場合の比較によって配置場所を決定するケースがあります。

5.2.9. 価格/コスト

価格およびコストはすべての項目に関連するものであり、トータルでのコストを最小化することをすべての判断基準においている組織も多くあります。

パブリッククラウドに従来のオンプレミス環境での運用基準をそのまま適用しようとするケースが多く見られますが、これではパブリッククラウドのメリットを享受することができず、逆に運用コストの増加に繋がります。パブリッククラウドの特性を把握したシステム設計や運用基準の取り決めが必要となります。

また、オンプレミスでは一般に機器を資産として購入するのに対してクラウドであれば資産を持たずに従量課金となるのが一般的であるため、組織の財務上の観点で有利になるシステム形態を選択するケースもあります。

5.2.10. 利用サービス

どのようなサービス、技術、プラットフォームを利用するかという大きなくくりとして、下記のような選択肢があります。

- IaaS(仮想マシン)
- CaaS(コンテナ)
- PaaS(プロットフォーム)
- FaaS(サーバーレス)

どれを選択するか、あるいは組み合わせるかによってシステム配置、データ配置、必要なネットワーク接続等も変わってきます。

また、どこまでを利用者が管理するのかという要素も変わってきますし、デザイン、運用、コスト、ライフサイクル、スピード等含めて劇的な違いを生みます。

現在の日本のエンタープライズ市場では仮想マシンベースでのシステム構築が大部分ですが、よりクラウドの利点を活かそうと考える場合には別の形態も利用していくことが鍵になります。

また、IaaSやCaaSのようにクラウドプラットフォーム毎に共通的な部分が多く選択の余地が広いものもあれば、クラウドプラットフォーム毎の独自色がつよく、「使いたいサービスがここにしか無い」ということでクラウドプラットフォーム、システム配置場所等が制限されるケースもあります。

5.2.11. システム利用者の場所

システムおよびデータの配置場所やネットワーク接続形態とも密接に絡んだ上で、遅延時間/応答時間を決定付ける要素でもあります。システム利用者がどこに存在し、どのような手段でシステムを利用するのかということを常に考慮する必要があります。

典型的な例としてはシステム利用者がオンプレミスに存在し、組織内のシステムを利用するケースと、スマートフォン等含めてインターネット上の任意の場所からシステムを利用するのとでは考慮すべきデータフローも、システムを配置すべき場所もすべて変化してきます。

5.2.12. 組織外部との連携(利用者/連携サービス)

システム利用者の場所とも密接に関係しますが、「外部の組織と組織間で連携する必要がある」、「外部のサービスと連携する必要がある」という場合には、それを前提としたシステム形態とする必要があります。

特に、連携先がレガシーでオンプレミスに閉じたシステムなのか、クラウド上に存在するモダンな連携手段を取ることが可能なAPI等も整備されているサービスなのかでは取るべき手段も変わってきますし、それに起因して選択すべきシステム形態、スタイルも変化してきます。

5.2.13. サービス、データ連携手段

これは単一のシステム内でコンポーネント間が連携する際の手段という側面と、システムとシステム外部との連携手段という側面の2つがあります。

サービス連携手段によって、完全にプライベートなネットワーク内での通信を想定しているものや、パブリックネットワーク上での通信を想定しているものなどの違いがあります。特にオンプレミスとパブリッククラウドとの間の連携を容易にするために通信ポートやファイアウォールの構成が少なくなることを狙ったものなども存在しており、利用するネットワーク形態とサービス連携手段には密接な関連性があります。

5.2.14. 開発手法/速度/リリース頻度/スキル/学習コスト

開発手法によって、速度、リリース頻度等が劇的に異なる現状があります。開発手法自体はどの場所、どのサービスを利用している場合でもどこにでも適用可能ではありますが「オンプレミスでのウォーターフォール型での開発」「クラウドでのアジャイルな開発」を比較した時にそれと相性のよいハイブリッドクラウドのスタイルは存在します。

これらはエンジニアのスキルセットや学習のためのコストも含め、組織文化の変革が必要となるため容易に選択や変更が可能なものではありませんが、ハイブリッドクラウドの利点を最大限まで享受するためにはどこかの時点で取り組むべき課題となるでしょう。

5.2.15. 運用手法/コスト/スキル/学習コスト

開発手法と同様に、運用の手法に関してもスタイル上大きな差異が発生するポイントです。オンプレミスとクラウドとで適用可能な製品、技術等に差異があり、具体的な方法が変化します。

ハイブリッドクラウドでは複数の場所、プラットフォームを運用管理することになりますが、そのなかでそれぞれの違いを許容するのか、同じ手法が通用するように利用するサービスや手法を限定するのか、差異を吸収するプラットフォームや製品を選択するのか等で大きな違いが生まれます。

5.2.16. システムパフォーマンス

システムを構築、利用する以上システムパフォーマンスは必ず考慮する必要があります。通常パブリッククラウドのほうが圧倒的にスケールしやすい環境が整っています。膨大なコンピューティングリソース、ストレージリソース、ネットワークリソースを利用することでシステムのパフォーマンスが向上し、そのパフォーマンスの向上に大きな意味があるケースではパブリッククラウドの利用のメリットが出るでしょう。また、急激なアクセス増加時のオートスケーリング等、システムのパフォーマンスを動的に変更できる点でも、パブリッククラウド利用のメリットがあります。

一方でオンプレミスでなければシステムのパフォーマンスが満たせないケースも存在します。典型的な例としてはネットワーク上の場所が重要であり、パブリッククラウドでは「遠すぎる」ようなケースです。このような場合にはオンプレミスにシステムを配置することが必須要件になることもあります。

5.2.17. パブリッククラウドへの接続性

ハイブリッドクラウドにおいてはパブリッククラウドのサービスをメリットある形で利用することが重要となります。その中でパブリッククラウドへの接続性が高い状態でシステムを可動し続けることが望ましいのは当然の話ですが、現実的にパブリッククラウドへの接続性が悪い場所、地域にてシステムを稼働させる必要がある場合もあります。このような場合には、よりオンプレミス側にシステムコンポーネントを重点的に配置し、パブリッククラウドへの接続ができない場合でも稼働するシステムにする必要があります。

5.2.18. 可搬性

システムを構成する仮想マシンやコンテナ等の構成要素をプラットフォームまたぎで移動させたいケースがあります。単純にシステムの配置場所を変更したいケースや、検証環境と本番環境を別のプラットフォームに配置したい場合もあります。パブリッククラウドへのマイグレーションを行いたい場合、またはその逆という場合もあります。このような場合、可搬性が高いプラットフォーム同士を選択することが重要な項目となります。

なお、システムを移動させるのではなく、最新のものを構築して古いものは破棄する等、そもそも可搬性を考慮しなくて良い手法を用いるという手段もありますし、PaaS、FaaSを利用するケースでは「可搬性」という概念そのものがあまり意味を持たない場合もあります。

5.2.19. ライセンスポリシー

システム内で利用しているプロダクトのライセンスポリシーの観点で、システムの配置場所や選択可能な構成パターンが制限されるパターンがあります。具体的にはパブリッククラウド上での利用が禁止されているライセンス形態もありますし、逆にパブリッククラウド上での利用が有利となるライセンスポリシーもあります。

技術的に実現可能であってもライセンスポリシー的に違反となってしまうような構成は現実的に選択できないため、ライセンスポリシー的に問題がない構成とする必要があります。

5.2.20. 認証システム

認証システムに何を利用するかもスタイル選択に大きな影響を与える要素です。

典型的な例としてWindows統合認証を用いたシステムがあります。Windows統合認証を利用するにはシステムコンポーネントに加えて利用クライアントもすべてプライベートネットワークで接続した上でActive Directoryドメインに参加させる必要があります。また、仮想マシンベースでIaaSを利用することになります。多くのPaaSはWindows統合認証をサポートしていません。

一方クラウド上のIDを利用する認証方式であればネットワーク構成やシステムコンポーネントの選択含めてかなり自由度がありますし、PaaSでのサポートも進んでいます。ですが逆にクラウド上のIDを利用するためにインターネット接続が必須となります。

さらに、クラウド上のIDであれば認証システム側で利用できるリッチな付加サービスが多数存在しそれを利用する事もできます。構築するシステムから認証やセキュリティに関連する要素を切り出す事も可能であり、全体のシステム構成への影響も大きなものになります。

5.2.21. システム稼働時間

システムの稼働時間もパブリッククラウドの利用によって大きく変化した概念でありスタイル選択にも重要なポイントです。

システムの稼働時間は従来からまちまちでしたが、オンプレミスに配置されるシステムはハードウェア自体のライフサイクルの概念とも紐付いて稼働期間として３年、５年という長期間であることも多くあります。

パブリッククラウド上に配置するシステムはその課金単位が秒単位にまで短くなっていることからごく短期間、極端な例では分単位、秒単位でしか稼働させないシステムも珍しくありません。サーバーレスのサービスを利用する場合にはシステムの稼働時間という概念すら適用が難しいものもあります。

もちろんパブリッククラウドでの短いシステム稼働時間の流れを受けて、オンプレミスでも同様のシステム構成を取る例も増えています。

一般的に言ってシステム稼働期間が短いものであればクラウド上に展開し、役割を終えればすぐに削除するようにすることでパブリッククラウドのコストメリットを享受することができます。

　また、システムが１日の中で稼働すべき時間も、システムを配置すべき場所に影響を与えます。24時間稼働し続ける必要がないシステムであればクラウド上に配置することでコストメリットを享受することができるケースが多くあります。

6. ハイブリッドクラウドスタイル

この章では典型的なハイブリッドクラウドの利用スタイルを「ハイブリッドクラウドスタイル」としてカタログ的に紹介をします。「スタイル選択ポイント」を軸にしながら重要なポイントに関しては「共通考慮ポイント」からも抜粋して記載を行います。

本ガイドラインで抽出しているスタイルは **「スタイル選択ポイント」に存在する要素とその選択肢の組み合わせをすべて網羅しているものではない** 事に注意してください。バリエーションとして理論的に存在しうるという可能性の話よりも、現実にそのシステム構成を選択した顧客とユースケースが存在している典型的な例であり、他の組織が自組織に適用を検討できる具体的なものであることを重視しています。

参照にあたっては「スタイルマップ」をあわせてご覧になりながら全体の中での位置づけを確認していただきながら読み進めていただくことを推奨します。

6.1. オンプレミス環境のみ(物理および仮想)

6.1.1. スタイル概要

スタイル選択ポイント	実装
システム配置場所	オンプレミスのみに配置
データ配置場所	オンプレミスのみに配置
システムアーキテクチャ	レガシーシステム

6.1.2. 特徴

このスタイルはハイブリッドクラウドではありませんが現実的に多いスタイルであり、特徴を抑えておく必要があるためスタイルの1つとして記載しています。

このスタイルはオンプレミス環境のみで仮想マシンを中心としたレガシーな手法でシステムを構築しているものを想定しています。典型的には仮想基盤を構築した上でその基盤上に仮想マシンを作成してシステムを構築します。

仮想基盤はSAN(Storage Area Network)を構築して専用ストレージを用いるものやHCI(Hyper Converged Infrastructure)を用いるものなど変遷がありますが、本書ではその違いを区別していま

せん。その点は「ハイブリッドクラウドの活用」という観点において本質的に重要な差異ではないと考えるためです。

システム構成としてはWindows系であれば下記のような構成が一般的です。

- 認証システムにActive Directoryを利用
- システムを構成するサーバー群にはActive Directoryに参加したWindows Serverを利用
- システムはWindows統合認証で構成
- クライアントはActive Directoryに参加したWindowsクライアント

全てプライベートネットワーク内に閉じているため、Active Directoryによるユーザーおよびグループの一元管理やグループポリシーによる種々の管理等が可能です。

「組織の内部」と「組織の外部」を明確に区切ることができるため、セキュリティ対策的にも「内部と外部の境界をネットワーク的に守る」という発想で組み立てられていることが多くあります。

本来は望ましくはありませんが、「組織内部からしかアクセスできないため、セキュリティパッチは適用しないが問題は無い」というロジックでUpdateを行わずに稼働され続けるケースが見られます。一度システムを構築すると極力変化させずに使い続けようとする傾向が強い組織でこのような運用形態が見られます。

6.1.3. メリット

- 過去のシステム構築、運用の実績をもとに同じ手法で新規システム構築が行える。
- 全てのコントロールが自組織にて行える。

6.1.4. デメリット

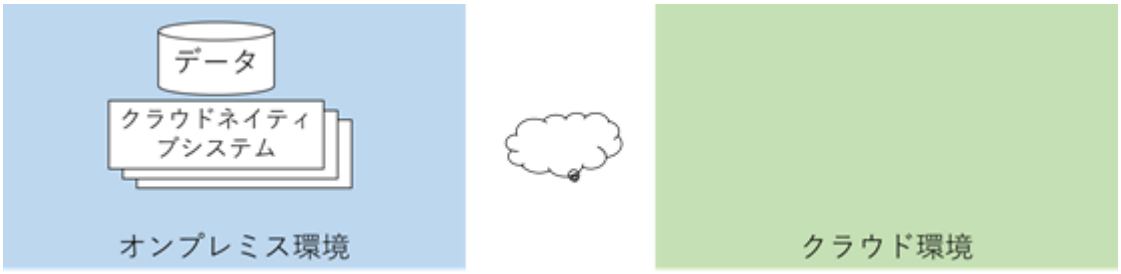
- パブリッククラウドを中心に起きている技術革新の恩恵を取り入れにくい。
- 物理的なものの管理から含め全てを自組織で行う必要があり展開スピードが相対的に遅い。

6.1.5. ユースケース

このスタイルに関してのユースケースは本ガイドラインの趣旨に合致しないため省略します。クラウドサービス登場以前は全てこの形態であったためユースケースは大量に存在することになります。

6.2. オンプレミス環境のみ(クラウドネイティブシステム)

6.2.1. スタイル概要



スタイル選択ポイント	実装
システム配置場所	オンプレミスのみに配置
データ配置場所	オンプレミスのみに配置
システムアーキテクチャ	クラウドネイティブシステム

6.2.2. 特徴

　このスタイルはハイブリッドクラウドではありませんが現実的に多いスタイルであり、特徴を抑えておく必要があるためスタイルの1つとして記載しています。オンプレミス環境のみでシステムを構築する点は「オンプレミス環境のみ(物理および仮想)」と同等ですが、このスタイルではクラウドネイティブシステムを構築します。物理サーバーや仮想サーバーを中心とした硬直的なシステム**ではない**システムの作りを指しています。

　クラウドネイティブシステムを構築する組織は相対的にパブリッククラウドの活用が進んでいる組織が多いですが、様々な理由でパブリッククラウドの利用ができないケースや、システムもデータも含めて全てをオンプレミスに配置することにメリットがあるケースではこのように場所としてはオンプレミスでありながら先進的な手法で先進的なシステムを構築し、生産性の向上を得ることができます。

　クラウドネイティブシステムを構築する上ではその基盤となるシステムもクラウドネイティブなシステムの提供に適したクラウドネイティブな基盤であることが重要となります。

6.2.3. メリット

- クラウドネイティブシステムを採用することでレガシーシステムと比較して大幅なビジネススピードの向上が可能。
- オンプレミスにシステム全体を配置することでインターネット(外部)のセキュリティ脅威との境界をこれまでと同じ考え方で構築可能。

6.2.4. デメリット

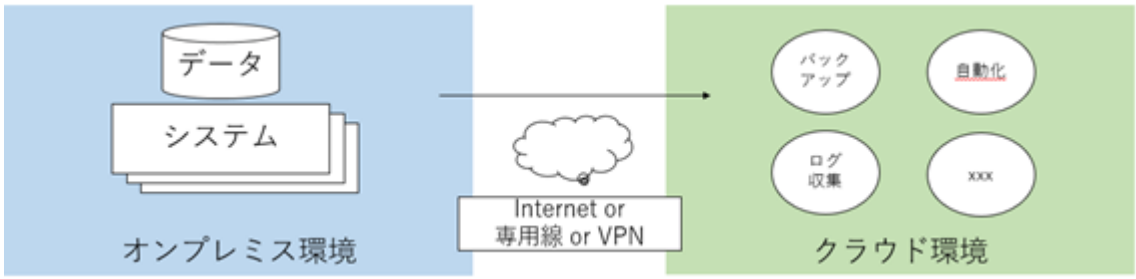
- パブリッククラウドを中心に起きている技術革新の恩恵を取り入れることが可能だが、パブリッククラウドのスピードには劣る。
- 物理的な機器の管理を自組織で行う必要があり、パブリッククラウドのスピード、スケールと比較すると劣る。

6.2.5. ユースケース

- レガシーシステムのクラウドネイティブ化

6.3. クラウドを補完的に利用

6.3.1. スタイル概要



スタイル選択ポイント	実装
システム配置場所	メインをオンプレミスに配置。補完的なクラウド上のサービスを利用
データ配置場所	メインをオンプレミスに配置。一部データをクラウド上に配置
システムアーキテクチャ	レガシーシステム

6.3.2. 特徴

　このスタイルはパブリッククラウド活用の1歩目としてよく採用されるスタイルです。

　オンプレミス側は既存と同じシステム構築手法を取りながら、バックアップやログ収集等既存システムを補完するサービスを採用し、システムを拡張します。

　パブリッククラウド側にはシステム本体があるわけではないため、パブリッククラウドどの接続はパブリックなインターネット回線を利用する事もできますし、パフォーマンス上あるいはセキュリティ上の懸念があれば専用線やVPNを利用して連携を行うことができます。

6.3.3. メリット

- メインのシステムは従来から利用している手法を用いることもできるためパブリッククラウドの利用を少ない変化で開始することができ、その恩恵を受けられる。

- クラウドのサービスを利用するものに関しては、それをオンプレミスで実現するためのリソース調達、基盤構築、運用、保守等に関して行う必要が無い。
- クラウドのサービスは従量課金でサービス利用できるためスモールスタートが可能。また、不要になった際の利用停止や、動的に拡張することも容易である。

6.3.4. デメリット

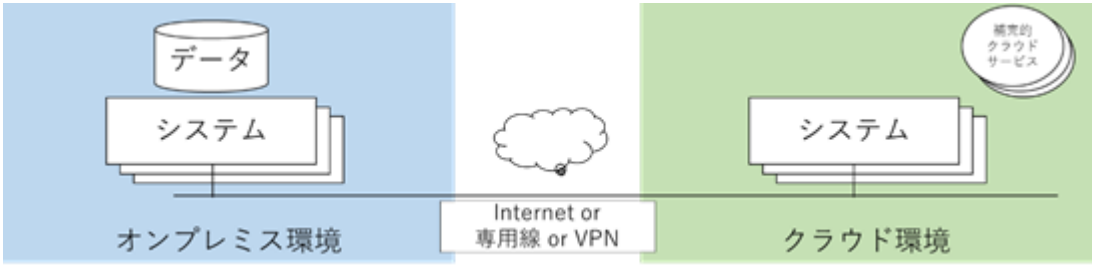
- パブリッククラウドという新しい場所の新しいサービスを利用開始するための組織的合意が必要。これがデメリットとならない組織も多くあるが、一部硬直した組織にとっては高いハードルと高いコストとなる。
- オンプレミスの環境に加えてパブリッククラウドの環境の管理が必要となる。過去の経験、ノウハウがそのまま適用可能なわけではないため、新しい概念の習得やポリシーの策定等が必要。
- オンプレミス環境とクラウド環境とで利用される技術、ユーザーインターフェース、考え方等が異なるため管理者の負担が大きい。
- 従量課金という形態での支払いとなるためお金の流れ、管理の仕方が必要となり、その流れを新たに構築する必要がある。

6.3.5. ユースケース

- クラウド災害対策
- ログ収集および分析基盤のクラウド上への構築
- 自動処理のクラウド化

6.4. データセンター拡張(データはオンプレミス)

6.4.1. スタイル概要



スタイル選択ポイント	実装
システム配置場所	オンプレミスとクラウドの両方に配置
データ配置場所	オンプレミスのみに配置
システムアーキテクチャ	レガシーシステム

6.4.2. 特徴

　このスタイルはパブリッククラウドの機能やサービスは利用したいが、主にセキュリティ上の懸念がある、法令上の縛りがある等の理由でパブリッククラウドにデータを配置できない場合に選択されることが多いスタイルです。

　データの配置場所がオンプレミスに固定されている関係上、オンプレミス環境とクラウド環境を繋ぐネットワークが重要であり専用線やVPNを使うパターンとInternetを利用するパターンとでは構成可能なシステムに大きな違いができます。

　クラウド環境と専用線やVPNで接続するケースでは接続速度がシステムにとって十分であれば完全にオンプレミス環境とシームレスに接続されるためクラウド環境であるということを特別に意識しないシステムの構成が可能です。クラウド環境とInternetで接続するケースに関してもその間を接続するための複数の技術が存在します。ユースケースの参照時にはオンプレミス環境とクラウド環境の接続手法にも注目して下さい。

6.4.3. メリット

- データをオンプレミス環境に保持する必要があるという要件を満たした上でパブリッククラウド環境を利用することができる。
- (オンプレミス環境とクラウド環境間を専用線あるいはVPNで接続した場合)オンプレミス環境の延長として従来から利用している手法を用いてハイブリッドな環境を利用可能となる。
- クラウドのサービスを利用するものに関しては、それをオンプレミスで実現するためのリソース調達、基盤構築、運用、保守等に関して行う必要が無い。
- クラウドのサービスは従量課金でサービス利用できるためスモールスタートが可能。サービス利用停止も容易。

6.4.4. デメリット

- データの配置場所がオンプレミスに限定されるため、システムによってはこのスタイルでの実装が現実的に困難となるケースがある。具体的にはパフォーマンス要件を満たせない事がある。
- 相対的にクラウドサービスを使うことのメリットが大きいビッグデータに代表されるようなクラウド環境上にデータを蓄積し分析するようなサービスの利用ができない。
- オンプレミス環境とクラウド環境とで利用される技術、ユーザーインターフェース、考え方等が異なるため管理者の負担が大きい。
- 単純に場所だけがクラウド環境になっているだけの使い方をする場合にはシステム上のメリットがあまり無い中で、クラウドのための新規の契約や支払い方法等が必要となり得られるメリットに対するオーバーヘッドが大きい。

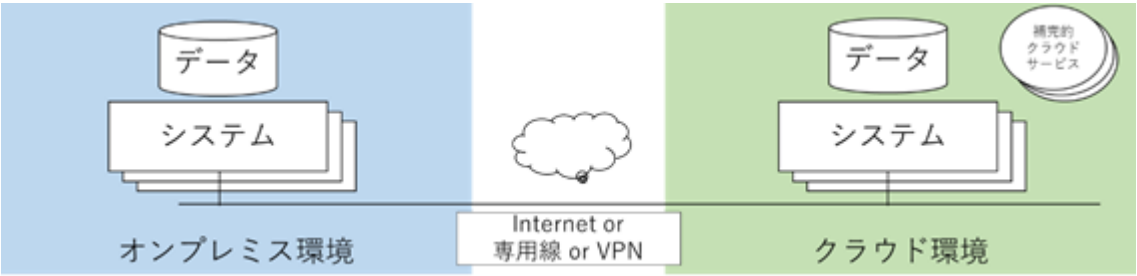
6.4.5. ユースケース

- 機密データをオンプレミスに配置しパブリッククラウドとシームレスに連携
- クラウド災害対策

- データ主権とデータグラビティ
- Oracleデータベースのマルチクラウド利用

6.5. データセンター拡張(データもクラウド配置可能)

6.5.1. スタイル概要



スタイル選択ポイント	実装
システム配置場所	オンプレミスとクラウドの両方に配置
データ配置場所	オンプレミスとクラウドの両方に配置
システムアーキテクチャ	レガシーシステム

6.5.2. 特徴

　このスタイルは自由にクラウド環境とオンプレミス環境を利用可能ななかでシステムを構成することができます。オンプレミスと専用線あるいはVPNで接続した構成であれば従来のオンプレミス環境で構築していたシステムと同じ考え方でシステムを任意の場所で構築する事もできますし、クラウドに存在する多種多様なサービスを自由にシステムに組み込むことも可能です。

　データをクラウドに配置することでむしろセキュリティ的には強化されるという考えを受け入れた組織がこのスタイルを選択することができます。

　クラウドサービスは常に進化を続けているため、システム構築の面でもどのサービスをどのように利用するかを考え続け、選択することになります。

6.5.3. メリット

- 多種多様なクラウドサービスを必要に応じてシステムで利用することが可能となる
- (オンプレミス環境とクラウド環境間を専用線あるいはVPNで接続した場合)オンプレミス環境の延長として従来から利用している手法を用いてハイブリッドな環境を利用可能となる。
- クラウドのサービスを利用するものに関しては、それをオンプレミスで実現するためのリソース調達、基盤構築、運用、保守等に関して行う必要が無い。
- クラウドのサービスは従量課金で利用できるためスモールスタートが可能。サービス利用停止も容易。

6.5.4. デメリット

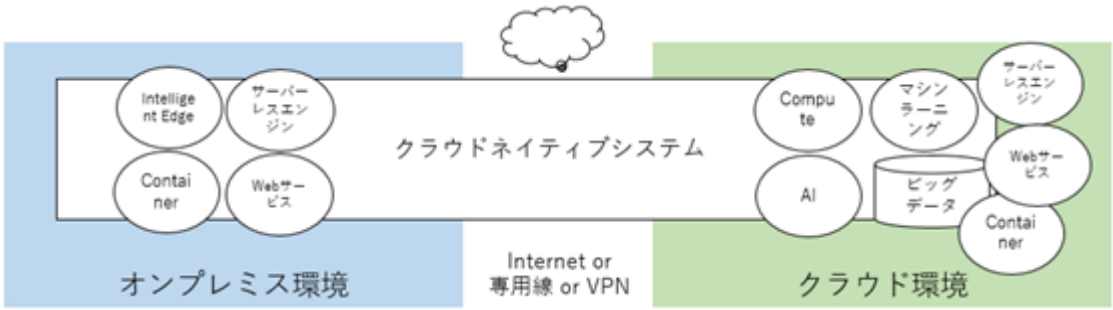
- オンプレミス環境とクラウド環境とで利用される技術、ユーザーインターフェース、考え方等が異なるため管理者の負担が大きい。
- 単純に場所だけがクラウド環境になっているだけの使い方をする場合にはシステム上のメリットがあまり無い中で、クラウドのための新規の契約や支払い方法等が必要となり得られるメリットに対するオーバーヘットが大きい。

6.5.5. ユースケース

- 高速な金融取引のためのハイブリッド構成
- クラウド災害対策
- Oracleデータベースのマルチクラウド利用
- クラウドによるオンプレミスDBの分析と可視化
- EAI利用によるクラウドをまたいだデータの利活用
- ファイルサーバーのスモールスタートとハイブリッド化

6.6. ハイブリッドなクラウド基盤を利用したシステム

6.6.1. スタイル概要



スタイル選択ポイント	実装
システム配置場所	オンプレミスとクラウドに適材適所で配置
データ配置場所	オンプレミスとクラウドに適材適所で配置
システムアーキテクチャ	クラウドネイティブシステム

6.6.2. 特徴

　オンプレミスにもクラウド基盤を持ち、パブリッククラウドと合わせてクラウドネイティブなシステムを構築するスタイルです。クラウドネイティブなシステムを構築することでよりクラウドのメリットを享受し、システムによる付加価値に注力します。

典型的にはIaaSよりもPaaSやコンテナを用いたマイクロサービス指向のアーキテクチャを採用し、Infrastructure as Codeの考え方や継続的インテグレーション、継続的デリバリー等のプラクティスを用いて頻繁に更新、改善するケースが多くあります。

6.6.3. メリット

- 多種多様なクラウドサービスを必要に応じてシステムで利用することが可能となる
- クラウドもオンプレミスもサービスを利用した分だけ支払う従量課金モデルで迅速にシステムを構築し、構成も変化させることができる。
- クラウドとオンプレミスとで同じ技術、コード、UI、考え方等を適用可能となる。
- 迅速にシステムを構築、改善することが可能となる。
- ハイブリッドなクラウド基盤の上で適材適所にコンポーネントを配置することが可能となり最適な形でのシステム構築が可能となる。

6.6.4. デメリット

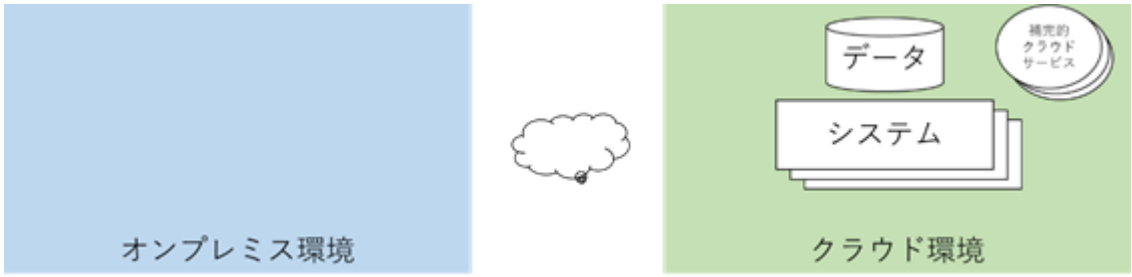
- レガシーなシステム構築とは異なる技術と考え方が必要となり、システム設計における根本的な意識変革が必要となる。
- 仮装基盤上のレガシーシステム構築時の手法は大部分が利用できないため、クラウドに即した手法の習得と実践が必要。
- クラウドとオンプレミス間の地理的/物理的なインターバルを意識しづらく、システム全体のパフォーマンス指標に対し、個々の処理のレイテンシーやスループットの問題への切り分けが困難となる。

6.6.5. ユースケース

- レガシーシステムのクラウドネイティブ化
- クラウドエッジ
- 高速な金融取引のためのハイブリッド構成
- Hybrid DevOps
- データ主権とデータグラビティ
- 分析のための階層化データ
- 地域分散アプリケーション
- 医療機関Webサイト
- Oracleデータベースのマルチクラウド利用
- クラウドによるオンプレミスDBの分析と可視化
- EAI利用によるクラウドをまたいだデータの利活用
- ファイルサーバーのスモールスタートとハイブリッド化

6.7. クラウド環境のみ(レガシーシステム)

6.7.1. スタイル概要



スタイル選択ポイント	実装
システム配置場所	クラウドのみ
データ配置場所	クラウドのみ
システムアーキテクチャ	レガシーシステム

6.7.2. 特徴

　システムを完全にクラウド環境上で構築、利用します。SaaSの利用と合わせて全てのシステムおよびデータをクラウド上に配置可能であり、システム利用者もインターネット経由での利用となります。

6.7.3. メリット

- オンプレミスに資産を持たず、全て従量課金でのシステム構築と利用が可能。
- クラウド上の多様なサービスを活用したシステム構築が可能。

6.7.4. デメリット

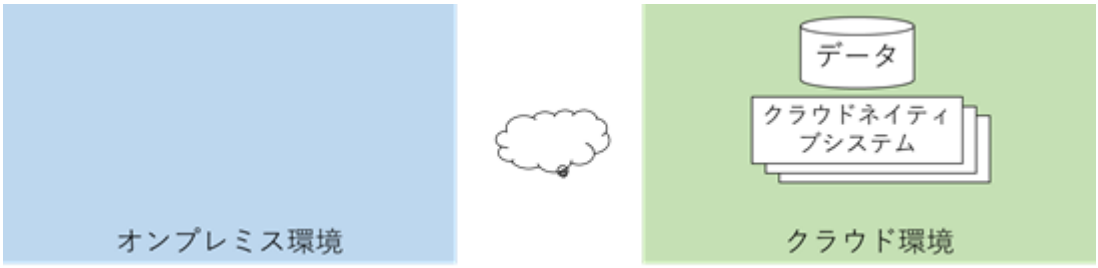
- システムによっては大容量のデータをクラウドと利用者との間でやり取りする必要が出るなど、パフォーマンス、コスト的に不利な構成となるケースがある。
- 法令等でオンプレミスでのデータ保持等が必須要件となっている場合には対応ができない。
- クラウドサービスの更新により、意図しない変更が意図しないタイミングで発生する可能性がある。
- サードパーティー製パッケージアプリケーションやSDK等、クラウド上での稼働を当該メーカーがサポートしないケースがある。

6.7.5. ユースケース

　このスタイルに関してのユースケースは本ガイドラインの趣旨に合致しないため省略します。

6.8. クラウド環境のみ(クラウドネイティブシステム)

6.8.1. スタイル概要



スタイル選択ポイント	実装
システム配置場所	クラウドのみ
データ配置場所	クラウドのみ
システムアーキテクチャ	クラウドネイティブシステム

6.8.2. 特徴

システムを完全にクラウド環境上で構築、利用します。SaaSの利用と合わせて全てのシステムおよびデータをクラウド上に配置可能であり、システム利用者もインターネット経由での利用となります。

典型的にはシステム利用者のIDもクラウド上のIDaaSを利用するフルクラウド構成となります。

アプリケーションやサービスの迅速な展開を可能とし、かつDevOpsを実現します。

6.8.3. メリット

- クラウドネイティブシステムを採用することでレガシーシステムと比較して大幅なビジネススピードの向上が可能。
- オンプレミスに資産を持たず、全て従量課金でのシステム構築と利用が可能。
- クラウド上の多様なサービスを活用したシステム構築が可能。

6.8.4. デメリット

- システムによっては大容量のデータのクラウドと利用者との間でやり取りする必要が出るなど、パフォーマンス、コスト的に不利な構成となるケースがある。
- 法令等でオンプレミスでのデータ保持等か必須要件となっている場合には対応ができない。
- クラウドサービスの更新により、意図しない変更が意図しないタイミングで発生する可能性がある。
- サードパーティー製パッケージアプリケーションやSDK等、クラウド上での稼働を当該メーカーがサポートしないケースがある。

6.8.5. ユースケース

- レガシーシステムのクラウドネイティブ化

7. ハイブリッドクラウドユースケース

この章では典型的なハイブリッドクラウド利用のユースケースを紹介します。同時にどの「スタイル」に適合するものかも明示します。

単一のユースケースが必ずしも単一のスタイルに適合するわけではないことに注意してください。 このユースケース集は多数のユースケースを集めて参考にすることができることを重視しており、その粒度はユースケースにより様々です。

なおユースケースは継続的に追加していくことを計画しています。

7.1. レガシーシステムのクラウドネイティブ化

7.1.1. 状況および課題

- 既存のレガシーシステムは手作業による操作が雑多で、属人性が高く、サービス要請ごとに大量の管理者が必要。人件費とシステムの運用費が高く、サービスインまでのリード時間が長い。
- 将来のITロードマップとして、IaaSだけでなく、PaaSも見据え、社内外のユーザーにより付加価値の高いサービスを提供するために、より俊敏性が高く、柔軟性の高いモダンなハイブリッドクラウド基盤が必要。

7.1.2. 解決方法

- IT運用におけるルーティンワークなどの手作業をInfrastructure as codeで自動化する
- Dockerや Kubernetesといったコンテナ技術、Azure Functionsなどサーバーレスコンピューティングの開発手法を導入する。
- クラウドのような瞬時性・セルフサービス・自動化などのメリットを享受できるハイブリッドクラウドのプラットフォーム（例：Azure Stack）を導入する。

7.1.3. 関連するスタイル

- オンプレミス環境のみ(クラウドネイティブシステム)
- ハイブリッドなクラウド基盤を利用したシステム
- クラウド環境のみ(クラウドネイティブシステム)

7.1.4. 解説

このユースケースは特にオンプレミスシステム重視ですが、新しい社会ニーズをいち早く満たすために、システムにクラウドのスピードを取り入れたい場合に適しています。

従来のウォーターフォール型のシステム開発はかなり時間がかかることから、アプリの設計運用にクラウドのアーキテクチャを導入し、迅速なサービス展開を図りたいという企業が増えてきています。例えば、アプリをコンテナ化すれば異なるプラットフォーム間での移行も簡単にできることで、クラウドロックインを回避できます。サーバーレスアーキテクチャーなら、トリガーとなる関数コードを変更するだけで、インフラを意識することなくイベントドリブン型のアプリ開発ができます。

オンプレ側にもクラウドネイティブな技術を取り入れ、より迅速かつ柔軟に変化する社会ニーズに対応できるシステム基盤が求められています。

7.1.5. 関連する技術

- Azure Stack
- ARM Template
- Azure Kubernetes Service
- Azure Functions

7.1.6. 参考情報

- 事例：Visionet社（インドネシアの大手ITサービスプロバイダー）が真のハイブリッドクラウド基盤としてAzure Stackを導入
- Azure Resource Manager テンプレートの構造と構文の詳細 | Microsoft Docs
- AKS - Microsoft Azure
- Azure Functions—サーバーレス アーキテクチャ | Microsoft Azure

7.2. 機密データをオンプレミスに配置しパブリッククラウドとシームレスに連携

7.2.1. 状況および課題

- 業界規制やコンプライアンス上、自社データや機密情報をオンプレに置きつつ、アプリなどのフロントエンドはパブリッククラウドを利用したい。必要に応じて、パブリック／オンプレ上のインスタンスをシームレスに拡張可能にしたい。
- オンプレミスとパブリッククラウドのアーキテクチャが違うと、運用の手間や、課金方式の違い、開発のコスト増などが懸念される。

7.2.2. 解決方法

- パブリッククラウドと一貫性のある基盤をオンプレミスに導入し、同じ運用・開発手法によって二重管理を防ぎ、学習コストも最低限に抑えられる。

7.2.3. 関連するスタイル

- データセンター拡張(データはオンプレミス)
- ハイブリッドなクラウド基盤を利用したシステム

7.2.4. 解説

　このユースケースは特に政府公共、金融機関などセンシティブな情報を扱う企業に適しています。

　昨今の環境変化において、パブリッククラウドの利用制限も緩和されつつあるが、パブリッククラウドに自社データを置くことができないエンタープライズ企業はまだ多くあります。また、パブリッククラウド特有のデータ転送料金やネットワーク課金があるため、大量のデータを抱える場合、データをパブリッククラウドに都度転送するとコストメリットが薄いというケースもあります。この際、データをオンプレのシステムに置き、クラウドでアジャイル開発したアプリとAPIで連携運用、継続的インテグレーション／継続的デリバリー（CI/CD）したりすることができます。

　また、パブリッククラウドの定期メンテナンス期間によるサービスダウンを回避する策として、パブリック上で立てたワークロードをシームレスにオンプレミスに移行できるので、ビジネス上の機会損失を最低限にすることができます。

7.2.5. 関連する技術

- Azure Stack

7.2.6. 参考情報

- 事例：iMOKO社（ニュージーランドのデジタルヘルスプロバイダー）がAzure Stackを自社デジタル医療システムの基盤としてを採用
- 事例：ANZ Internet Banking (オーストラリア・ニュージーランド銀行)がAzure Stackを社内の処理基盤として採用
- Azure Stack 向けの開発
- Azure と Azure Stack にアプリをデプロイする
- ハイブリッド クラウド アプリケーション

7.3. クラウドエッジ

7.3.1. 状況および課題

- クラウドのサービスを活用したアプリケーションを実現したい
- 全てクラウドで実装することも可能だが、クラウドまでの距離による遅延や、大容量のデータを全てクラウドまでアップロードする際にかかる時間のためアプリケーションのパフォーマンスが悪くなりすぎてしまう。

- 場合によってはインターネットへの接続回線の品質が悪い、速度が遅すぎるケースもある。場合によっては断続的に接続されない状況になるケースも考えられる。
- 従来の仮想マシンベースのレガシーシステムで構築することも不可能ではないが、パブリッククラウドに存在しているリッチなサービスや開発手法を活用したい。

7.3.2. 解決方法

- オンプレミス上にクラウドサービスのエッジとなるシステムを配置する。オンプレミス側で独立してシステムが稼働することが可能となるため遅延の問題は大幅に改善される。
- データ収集や処理はエッジシステムで行う。バックグラウンドで非同期的にクラウドサービスと連携し、追加の分析等を行う。これによりクラウドサービスのメリットも享受される。
- オンプレミス側でもクラウドのサービスや開発手法を活用するために、基盤にはAzure Stack等のクラウド基盤を導入する。

7.3.3. 関連するスタイル

- ハイブリッドなクラウド基盤を利用したシステム

7.3.4. 解説

　このユースケースは抽象度の高いユースケースですが、クラウド一辺倒では実現が難しいシステムの例と解決策を端的に表しているケースです。クラウド上のみにシステムを配置するだけでは生まれてしまう課題に対して、オンプレミス側に一部の役割をオフロードします。

　適材適所のシステム配置を迅速に行うことはオンプレミス側がレガシーなシステムでは難しいことが多々あります。特にパブリッククラウドをメインで利用し、各種サービスを活用している状況ではそれが顕著です。このためオンプレミス側にもクラウドと極力同じことができる基盤が求められることになります。

7.3.5. 関連する技術

- Azure Stack
- Azure IoT Edge

7.3.6. 参考情報

- Azure Stack とは | Microsoft Azure
- IoT Edge | Microsoft Azure

7.4. 高速な金融取引のためのハイブリッド構成

7.4.1. 状況および課題

- 金融における高頻度な取引を独自のアルゴリズムを用いてミリ秒単位で速度を競う必要がある。
- クラウド上にシステムを配置すると遅延時間の関係で競合優位性が全く出せない。
- 速度的優位性がある場所は大都会の中心部であり土地が極めて高価である。

7.4.2. 解決方法

- オンプレミスの速度的優位性がある場所に取引所利用のハードウェアおよびシステムを配置する。
- システム速度に影響しないコンポーネントはパブリッククラウドに配置することでオンプレミスの高価な場所の必要床面積を大幅に減らす。
- 分析および予測処理等はパブリッククラウドのAIサービスを利用する。

7.4.3. 関連するスタイル

- データセンター拡張(データはオンプレミス)
- データセンター拡張(データもクラウド配置可能)
- ハイブリッドなクラウド基盤を利用したシステム

7.4.4. 解説

　このユースケースは「クラウドエッジ」のより具体的な一例です。オンプレミスにコンポーネントを配置する絶対的な価値と、パブリッククラウドのサービスを利用する価値があるがゆえに、最適構成を模索すると結果的にハイブリッドクラウド構成となる例です。

　システムの具体的な構築方法には触れていませんが、目的を達成するために不必要な部分は極力排除したうえでクラウドネイティブなシステムとし、継続的にシステムを更新、改良し続けることで競合優位性を保ち、よりメリットが出せると考えられます。

7.4.5. 関連する技術

- Azure
- Azure Stack
- Express Route

7.4.6. 参考情報

- Microsoft Azure Cloud Computing Platform & Services
- Azure Stack とは | Microsoft Azure

- ExpressRoute - 仮想プライベート クラウド接続 | Microsoft Azure

7.5. クラウド災害対策

7.5.1. 状況および課題

- システム障害および災害対策のためにシステムの待機系を準備する必要があるが、待機系のための拠点を確保した上で、電源の確保、機器の購入、維持、管理等が必要なため有事の際にしかりようしない待機系のファシリティに対する高いコストが掛かっている
- 構築後の運用フェーズで継続的に管理を継続しながら「切り替えても問題ない」かつ「決められた RTO (目標復旧時間) および RPO(目標復旧時点)を担保している」状況を保持、確認し続けるための運用コストが高い。

7.5.2. 解決方法

- 災害対策のための待機系としてオンプレミスにファシリティを持たず、有事の際には災害対策先としてパブリッククラウドを利用する。

7.5.3. 関連するスタイル

- クラウドを補完的に利用
- データセンター拡張(データはオンプレミス)
- データセンター拡張(データもクラウド配置可能)

7.5.4. 解説

　このユースケースは劇的なコストメリットを出すことができるユースケースです。実際に災害対策用に待機系のファシリティを保持、およびきちんと切り替えられる状態を保つのは非常に困難です。実際問題として、(縮退運転ではなく)完全に切り替えを行うためには待機系のシステムにも本番系のシステムと全く同じスペック、容量が必要となりますが、災害等発生しなければ単純に使われないシステムとなってしまいます。災害対策先としてクラウドおよびクラウドサービスを利用することで、ファシリティの準備や運用も必要なく、必要な時にのみ待機系に切り替えることが可能です。

　災害対策システムの実装として、システムで構築されているサーバーや仮想マシンを、クラウド上のIaaS仮想マシンとして立ち上げておき、オンプレミスの本番システムとデータ同期する方法もありますが、Azure Site Recovery等の災害対策用のクラウドサービスを利用することで、IaaS仮想マシンを停止した状態でデータを同期、最新の状態に保持することができます。このことにより、待機系の維持コストを削減できる、というメリットを享受することができます。

　また、上記のクラウドサービスを利用することで、本番システムとは完全に切り離した状態で「テスト」を実施することが非常に容易になります。このテストのための独立した環境も必要な時に必要なだけクラウドの環境上で展開することが可能となります。実際にパブリッククラウドを利用する際の「予行練習」としてテストできる容易さは特筆すべきものがあります。

ただし、Active DirectoryなどID機能に依存するシステムの場合、切り替え先のシステムがきちんとIDシステムと通信できる必要があります。加えて、切り替え先のシステムにきちんとアクセスできるよう、DNSの切り替えも必要となるケースがあるため、実際には切り替える際の事前準備や考慮、工夫が必要となります。

メリットをより活かすためにはシステムの認証システムとしてクラウドのIDサービスを採用し、クライアントのアクセス場所もインターネット経由にする等の構成が望ましいケースが多くあります。

「クラウドを補完的に利用」「データセンター拡張(データはオンプレミス)」のスタイルでクラウドを災害対策として使用する場合には、システムをクラウド上に配置すること、データをクラウドに配置することに関して利用ポリシーの整合性を取る必要があります。災害発生時のみクラウドの利用を許可するというポリシーが選択される事が多くあります。

7.5.5. 関連する技術

- Azure Site Recovery
- Azure Backup
- Commvault
- Zerto

7.5.6. 参考情報

- Azure ディザスタ リカバリ サービス – Azure Site Recovery | Microsoft Azure
- クラウド バックアップ - サービスとしてのオンラインのクラウド バックアップ | Microsoft Azure
- ディザスタ リカバリ | Commvault
- Zerto仮想レプリケーション

7.6. Hybrid DevOps

このユースケースは下記で紹介されているハイブリッドアプリケーションパターンからの紹介です。

- MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

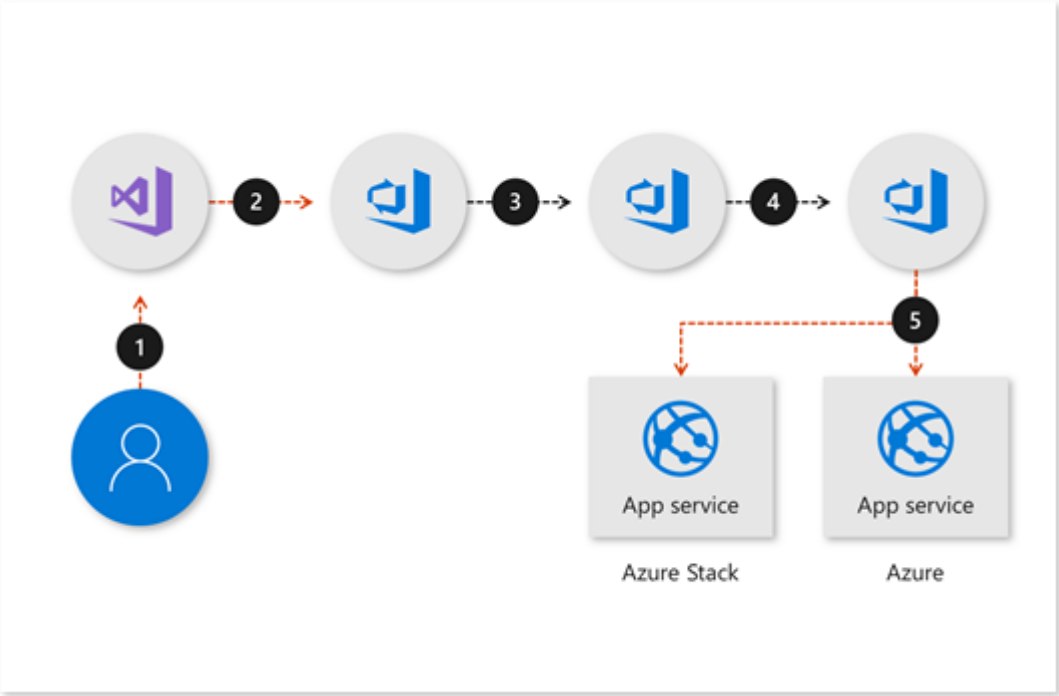
7.6.1. 状況および課題

- クラウドとオンプレミスに同一のクラウドネイティブなアプリケーションを継続的に展開したい。
- 継続的に開発と改善を行いたい。
- 開発者は同じツール、開発手法、コードを用いてクラウドの違いを意識する必要が無いようにしたい。

- 状況に応じてパブリッククラウドの利用もプライベートクラウドの利用も自由にコントロール可能な状況にしたい。

7.6.2. 解決方法

- パブリッククラウドにAzureを、オンプレミスにAzure Stackを利用しHybrid DevOps環境を構築する。



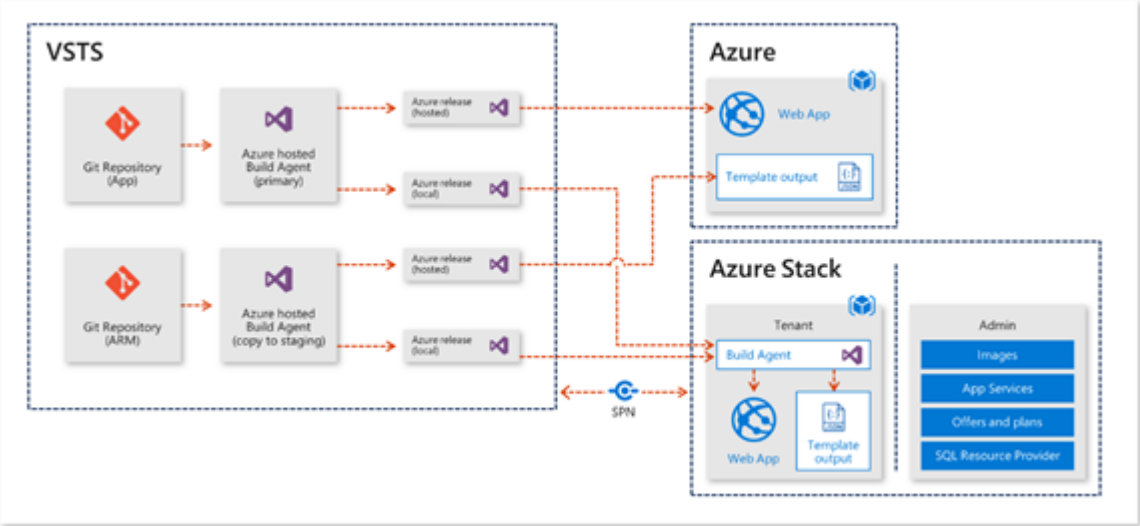
引用元：MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

1. アプリケーションコードおよびARMテンプレートを変更する
2. アプリケーションコードおよびARMテンプレートがAzure上のレポジトリにコミットされる
3. 自動的にアプリケーションが構築され、ユニットテストが実行される
4. アプリケーションが環境固有のパラメーターで自動展開される
5. アプリケーションがAzure上およびAzure Stack上のApp Serviceで実行される

7.6.3. 関連するスタイル

- ハイブリッドなクラウド基盤を利用したシステム

7.6.4. 解説



引用元：MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

- レポジトリおよびパイプラインにはAzure DevOps(Visual Studio Team Services)を利用します。
- それぞれの環境への認可のためにサービスプリンシパルを用います。
- Azure hostedのAzure DevOpsエージェントを用いることでビルドを高速にし、展開回数を多くすることが可能となります。
- Azureとエージェントとの間の接続性を担保する必要があります。オンプレミスのAzure Stackの環境に関してはインターネットからの接続を許可しないケースも多いと想定されますが、エージェントからインターネットへの一方向の接続のみで問題無ありません。
- 全ての利用サービスと利用OSイメージが両方の環境に存在している必要があります。Azure Stack側には必要に応じて追加のリソースプロバイダーをインストールし、サービスを利用可能とします。
- プロファイルを利用することで2つの環境への展開時の差異を最小化することができます。

7.6.5. 関連する技術

- Azure DevOps
- Visual Studio
- App Service
- Web Apps
- Azure Resource Manager

7.6.6. 参考情報

- MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack
- Azure DevOps Services | Microsoft Azure
- Visual Studio IDE、コード エディター、VSTS、App Center - Visual Studio

---

- Azure App Service - アプリのホスティング | Microsoft Azure
- Web App Service | Microsoft Azure
- Azure Resource Manager | Microsoft Azure

7.7. データ主権とデータグラビティ

　このユースケースは下記で紹介されているハイブリッドアプリケーションパターンからの紹介です。

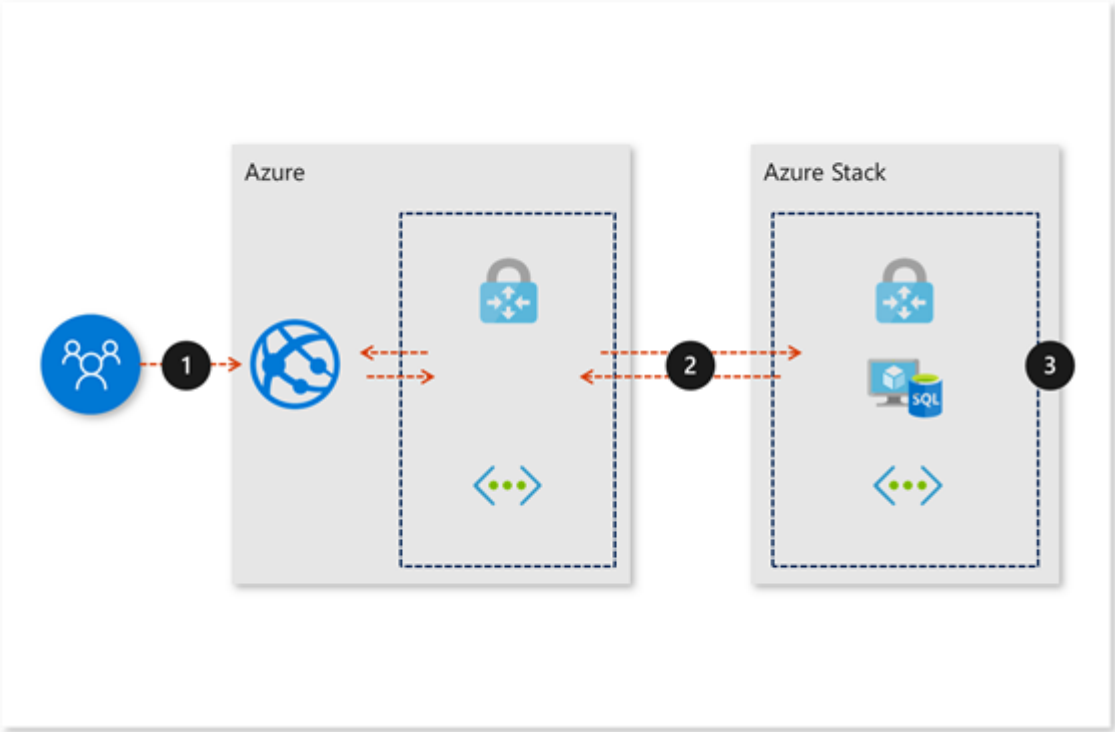
- MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

7.7.1. 状況および課題

- 法律あるいはプライバシーおよびコンプライアンスポリシーでデータの保管場所が規定されていて全てをパブリッククラウドに配置することができない。
- インターネットに向けて公開されたWebアプリケーションであり急激なユーザーアクセスの集中の可能性もあり、パブリッククラウドのスケールを使って処理したい。
- パブリッククラウドのサービス障害時にもサービスを継続したい。

7.7.2. 解決方法

- アプリケーションとサービスをデータが存在するべき場所にあわせて配置する。
- 堅牢なアプリケーションをクラウドの境界を超えて配置する。
- 法規制、プライバシーおよびコンプライアンス要件に合致させる。



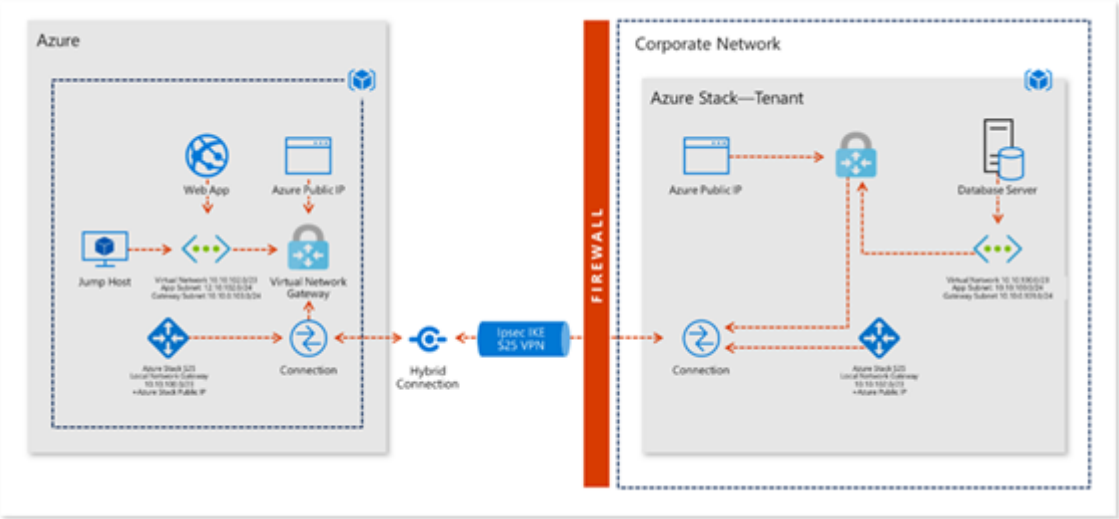
引用元：MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

1. ユーザーはデータをAzureに配置されたWebアプリケーションで入力する。
2. Webアプリケーションはデータを VPN経由で仮想ネットワークが接続されたAzure Stack上のデータベースにコミットする。
3. データはAzure Stack上に配置された仮想マシン上のSQLデータベースに保存される。

7.7.3. 関連するスタイル

- データセンター拡張(データはオンプレミス)
- ハイブリッドなクラウド基盤を利用したシステム

7.7.4. 解説



1. Azure上の仮想ネットワークとAzure Stack上の仮想ネットワーク内のサブネットに重複が無いことを確認します。アドレスの重複がある場合にはネットワークの接続ができません。
2. ARMテンプレートを利用してインフラを展開することで信頼性のある展開が可能になります。
3. オンプレミスのストレージが要件に合致していることを確認します。
4. Azure Stackがインターネットからルーティング可能である必要があります。
5. 強力な共有鍵を使用してVPNを構成します。あるいはExpress Routeを使用します。

7.7.5. 関連する技術

- Azure App Service
- Azure Web App
- VPN Gateway
- Express Route
- Azure Resource Manager
- ARM Template

7.7.6. 参考情報

- Azure App Service - アプリのホスティング | Microsoft Azure
- Web App Service | Microsoft Azure
- VPN Gateway | Microsoft Azure
- ExpressRoute - 仮想プライベート クラウド接続 | Microsoft Azure
- Azure Resource Manager | Microsoft Azure
- Azure Resource Manager テンプレートの構造と構文 | Microsoft Docs
- Azure クイック スタート テンプレート

7.8. 分析のための階層化データ

　このユースケースは下記で紹介されているハイブリッドアプリケーションパターンからの紹介です。

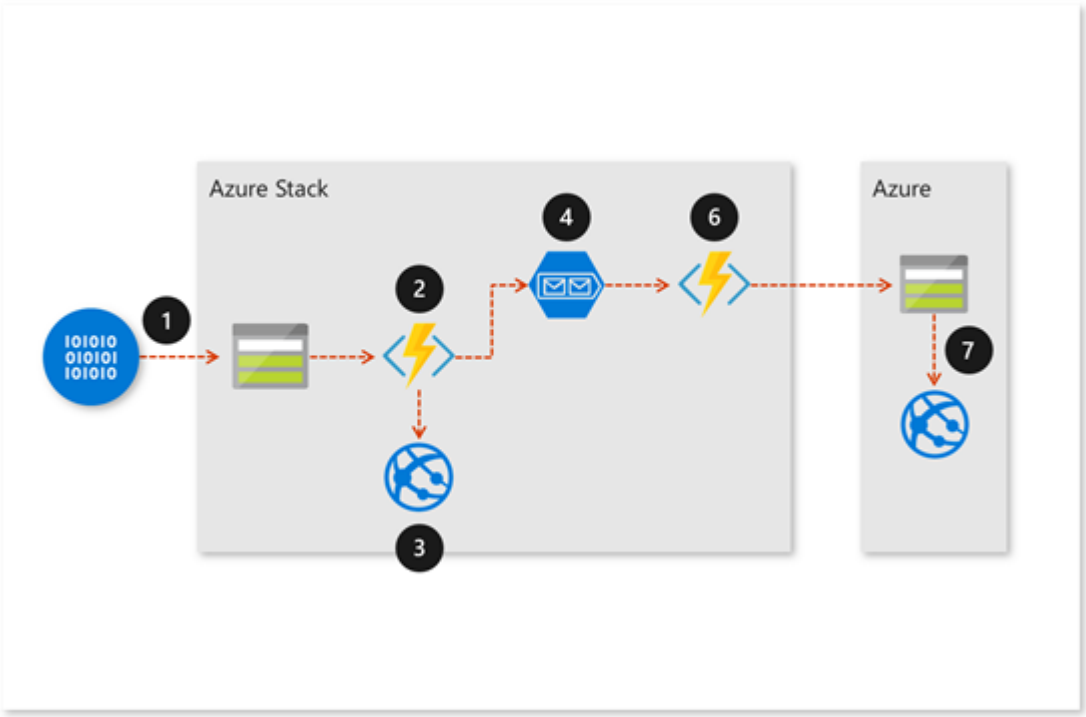
- MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

7.8.1. 状況および課題

- 機密性の高いデータを扱うアプリケーションを動作させたい。
- オンプレミス上では機密性の高いデータから得られる情報を元にアプリケーションを動作させたい。
- パブリッククラウド上では匿名化されたデータに基づく情報を元にアプリケーションを動作させたい。
- オンプレミスのデータとパブリッククラウドのデータとは極力リソースを消費せずに必要な分だけ最小限のコストで連動させるようにしたい。
- パブリッククラウドのアプリケーションも高速に動作させる必要がある。オンプレミスまでデータを参照させに行くようなデザインは許容されない。
- オンプレミスとパブリッククラウド間のネットワーク接続は限定的であり、不安定である。場合によってはしばらく接続できないこともありうる。

7.8.2. 解決方法

- データを最適化することで遅延問題への対策とします。
- クラウドベースの開発モデルをデータが存在する場所に持ち込みます。
- ビジネスのための予測的な洞察をシステム全体に提供します。場所により適切な匿名化を行います。

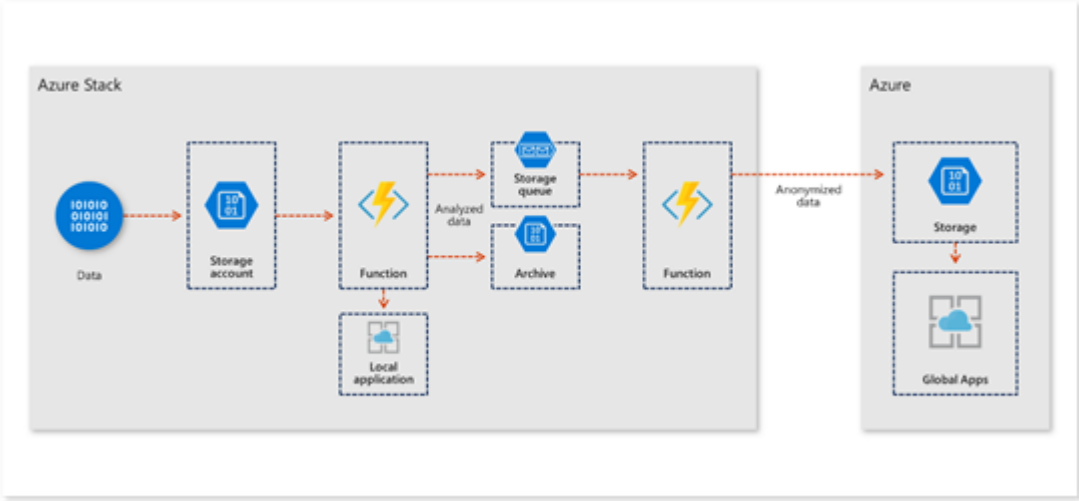


1. データがストレージアカウントに格納されます。
2. Azure Stack上のFunctionにてデータが分析されます。
3. プライベートのアプリケーションはローカルにあるデータを元に動作します。
4. 予測的な洞察は匿名化されキューに格納されます。
5. FunctionがデータをキューからAzureストレージに送信します。
6. Azure上の公開されたアプリケーションには匿名化されコンプライアンス的にも問題ない予測的な洞察が公開されます。

7.8.3. 関連するスタイル

- ハイブリッドなクラウド基盤を利用したシステム

7.8.4. 解説



1. アプリケーションコード作成時には断続的にしかインターネットに接続されない状況や完全に切り離された状況にも対応可能なように計画します。
2. コンプライアンスに合致するようにAzure上のデータは暗号化されて格納されます。
3. Azure FunctionsおよびApp Serviceを用いてアプリケーションは適切にスケールするように構成されます。

7.8.5. 関連する技術

- Azure Functions
- Azure App Service
- Azure Stack

7.8.6. 参考情報

- Azure Functions—サーバーレス アーキテクチャ | Microsoft Azure
- Azure App Service - アプリのホスティング | Microsoft Azure
- Azure Stack とは | Microsoft Azure
- Queue Storage | Microsoft Azure

7.9. エッジAI

　このユースケースは下記で紹介されているハイブリッドアプリケーションパターンからの紹介です。

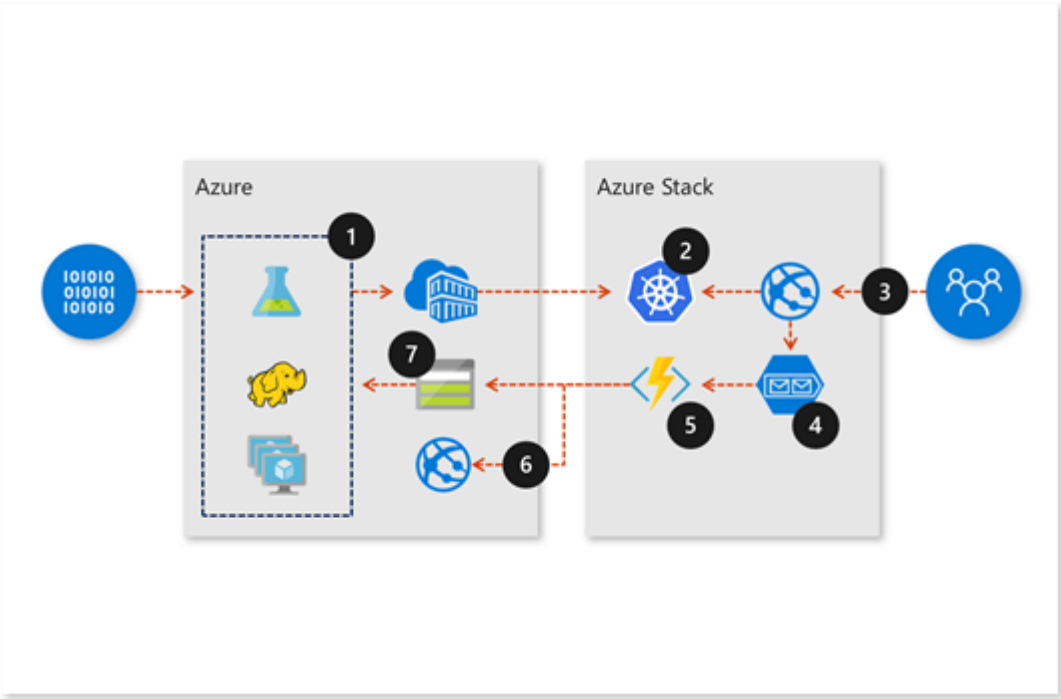
- MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

7.9.1. 状況および課題

- AIを用いた分析を行いたい。
- 実際にAIを利用したい場所はオンプレミス。
- システムとして極力早いレスポンスを得たい。

7.9.2. 解決方法

- AIとコンピューティングパワーをデータが生成される場所に配置します。
- モデルのトレーニングはパブリッククラウドのスケールを活用して行い、生成したモデルはエッジ(オンプレミス)に配置します。
- エッジにて処理を行うことですばやく結果を得ることを可能にする。

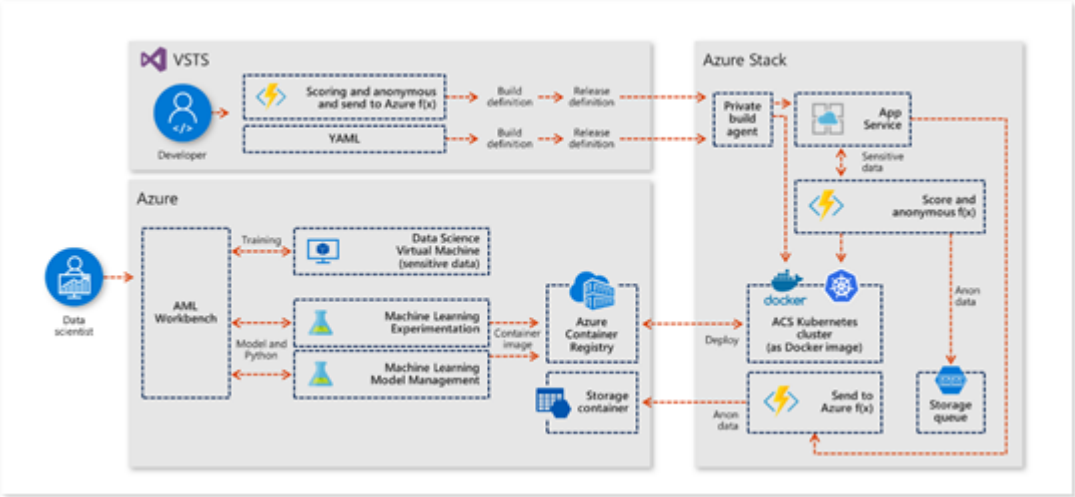


1. モデルはAzureマシンラーニングを用いてトレーニングされ、コンテナ化されます。
2. モデルはAzure Stack上のKubernetesクラスタに展開されます。
3. 入力はコンテナ上のモデルによってオンプレミスで独立してスコア付けされます。
4. スコアから得られた予測的な洞察はキューに格納されます。
5. Azure上に送信する際にデータは匿名化されコンプライアンス的に問題無い状態となり格納されます。
6. Azure上の公開されたアプリケーションには匿名化されコンプライアンス的にも問題ない予測的な洞察が公開されます。
7. エッジにてスコア付けされたデータはモデルの改善に利用されます。

7.9.3. 関連するスタイル

- ハイブリッドなクラウド基盤を利用したシステム

7.9.4. 解説



1. KubernetesがAzure Stack上に展開され、適切にネットワークが構成されている必要があります。
2. データのアップロードや新しいモデルの展開時に使用するインターネット接続が信頼できないものであるケースも考慮に入れて計画可能です。
3. Azure上のモデルトレーニングのためのクラスタは適切にスケールさせます。
4. スコアリングするアプリケーションが適切にDoS攻撃から保護されていることを確認します。

7.9.5. 関連する技術

- Azure Stack
- Azure Machine Learning
- Azure Container Service
- Azure Kubernetes Service
- Azure App Service
- Azure Functions
- Azure Container Registry
- Azure IoT Edge

7.9.6. 参考情報

- Azure Stack とは | Microsoft Azure
- Machine Learning | Microsoft Azure
- Container Service
- Azure Kubernetes Service (AKS) | Microsoft Azure
- Azure App Service - アプリのホスティング | Microsoft Azure
- Azure Functions—サーバーレス アーキテクチャ | Microsoft Azure
- Azure Container Registry – Docker Registry | Microsoft Azure
- IoT Edge | Microsoft Azure

7.10. 地域分散アプリケーション

　このユースケースは下記で紹介されているハイブリッドアプリケーションパターンからの紹介です。

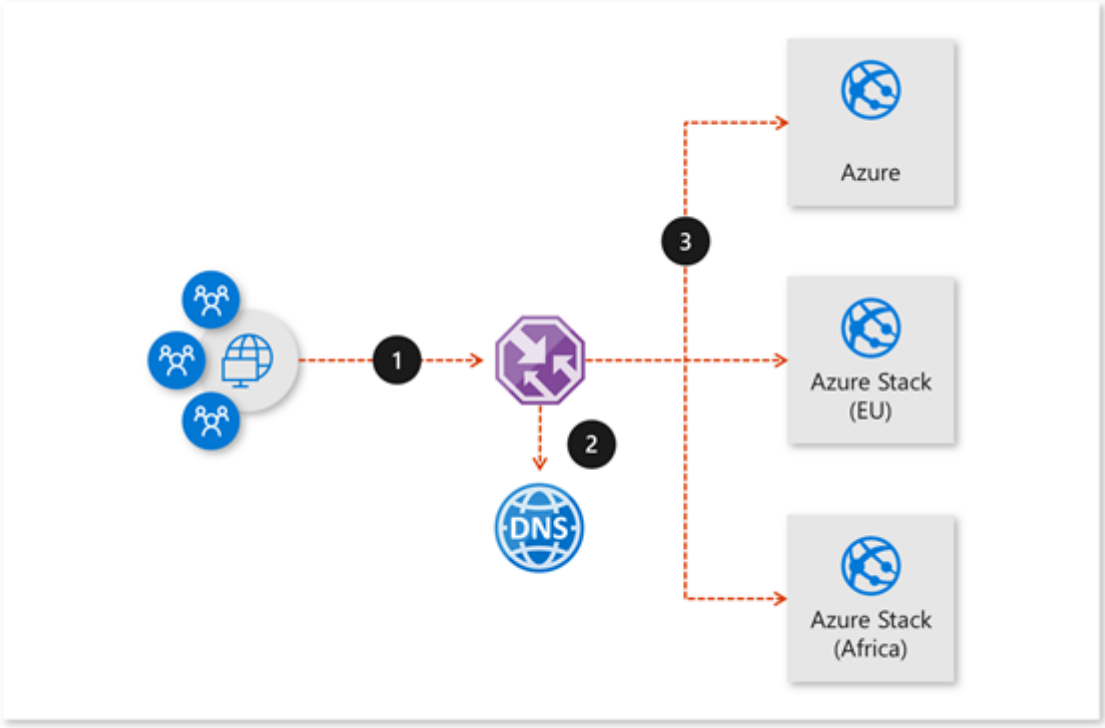
- MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

7.10.1. 状況および課題

- サーバーレス、マイクロサービス等を取り入れた最新のアプリケーションを世界展開したい。
- 地域ごとに異なるデータ保護要件がありパブリッククラウドを利用できる地域とできない地域がある。

7.10.2. 解決方法

- AzureとAzure Stackを用いてモダンなサーバーレスアプリケーションとマイクロサービスをグローバルに展開する。
- AzureとAzure Stackを用いることで同一のアプリケーションをコード変更無しで展開可能とする。
- 世界中のユーザーにコンプライアンス要件に合致したサービスを提供する。
- Azureサービスとのシームレスな接続を可能とし、世界中のユーザーが同一のURLでアプリケーションにアクセス可能とする。



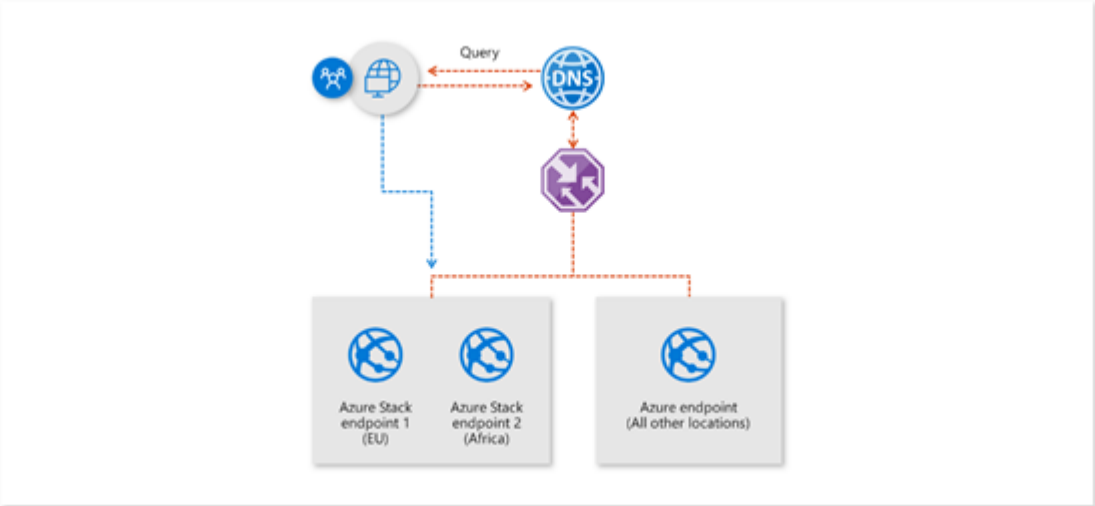
1. ユーザーがアプリケーションにアクセスします。

2. DNSがAzure Traffic Managerにクエリします。
3. Traffic Managerがユーザーに最も近いAzure上またはAzure Stack上のアプリケーションを返します。

7.10.3. 関連するスタイル

- ハイブリッドなクラウド基盤を利用したシステム

7.10.4. 解説



1. SSL証明書とドメイン名が適切に構成されている必要があります。
2. このソリューションはスケーラビリティの観点では記載していませんが、スケーラビリティの観点でのソリューションとすることも可能です。
3. Hybrid DevOpsで解説したハイブリッドパイプラインを用いて確実にアプリケーションを全てのクラウドに展開します。
4. アプリケーションコードと位置情報を用いて適切にユーザーを適切なアプリケーションにルーティングします。

7.10.5. 関連する技術

- Azure Stack
- Azure Front Door
- Azure Traffic Manager
- Azure App Service

7.10.6. 参考情報

- Azure Stack とは | Microsoft Azure
- Azure Front Door Service | Microsoft Azure
- Traffic Manager - クラウドベースの DNS 負荷分散 | Microsoft Azure

- Azure App Service - アプリのホスティング | Microsoft Azure

7.11. 医療機関Webサイト

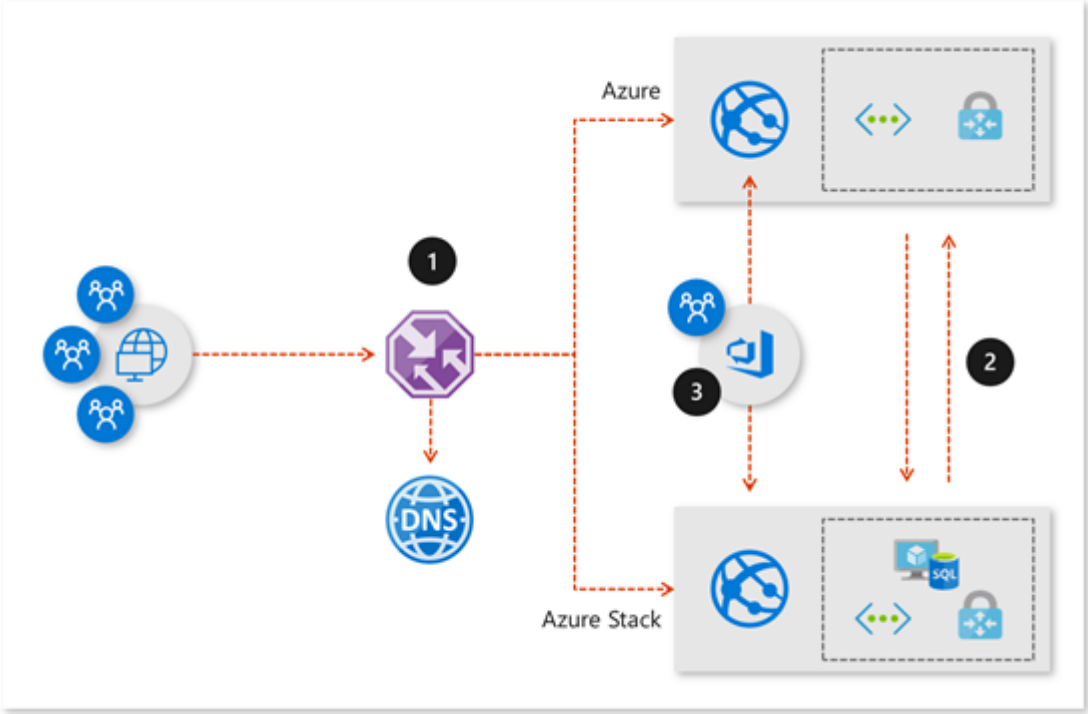
　このユースケースは下記で紹介されているハイブリッドアプリケーションパターンからの紹介です。

- MyIgnite - Understanding hybrid application patterns for Microsoft Azure Stack

7.11.1. 状況および課題

- 医療データは非常に機密性が高く、オンプレミスに配置する必要がある。
- Webサイトは非常に高い可用性を実現しつつ高負荷にも耐えられる必要がある。
- 開発者は上記要件を実現するためであっても極力無駄な工数は省き、一括でアプリケーションを展開可能としたい。

7.11.2. 解決方法



1. ユーザーは自動的に最もコンプライアンスに合致したクラウドに誘導される。
2. データはオンプレミスのAzure Stack上に保存される。
3. 開発者はハイブリッドなCI/CDパイプラインを利用し、アプリケーションを一貫性ある方法で展開する。

7.11.3. 関連するスタイル

- ハイブリッドなクラウド基盤を利用したシステム

7.11.4. 解説

医療機関のような機微なデータを取り扱うシステムにおいては、データはオンプレミスでかつ堅牢性が担保される必要があります。

利用者側からの観点からはネットワーク遅延を意識せずストレスフリーなユーザーエクスペリエンスを享受でき、開発者側からの観点からはAzureとAzure Stackの違いを意識する事なくWebアプリケーションの開発にフォーカスする事を可能にしました。

7.11.5. 関連する技術

- Azure Stack
- Azure Traffic Manager
- Azure DevOps
- Azure App Service

7.11.6. 参考情報

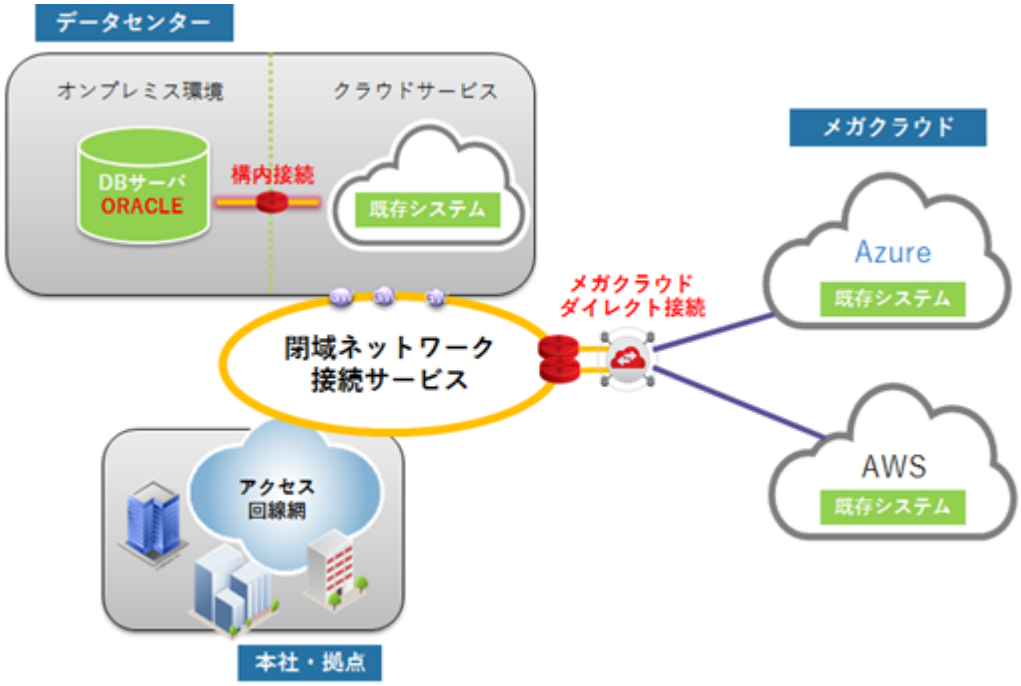
- Azure Stack とは | Microsoft Azure
- Traffic Manager - クラウドベースの DNS 負荷分散 | Microsoft Azure
- Azure DevOps Services | Microsoft Azure
- Azure App Service - アプリのホスティング | Microsoft Azure

7.12. Oracleデータベースのマルチクラウド利用

7.12.1. 状況および課題

- 既存システム（オンプレミス環境）のクラウドサービス利用を計画している。
- データベースはOracleを利用している。
- クラウドサービスの利用に当たってはOracleライセンス規約に抵触しないように注意する必要がある。
- 社内のセキュリティポリシーからデータベースは仕様・構成を把握できる環境に配置する必要がある。

7.12.2. 解決方法



- OracleサーバーはOracleライセンス規約及びセキュリティポリシーに準拠する必要があるため、オンプレミス環境に残す。
- アプリケーションサーバーなど関連システムはクラウドサービスもしくはメガクラウドに移行する。
- オンプレミス環境＋クラウドサービス環境＋メガクラウド間は閉域ネットワーク接続サービスを利用し、場所を意識することなくシームレスに通信できるようにする。
- DBサーバーとの通信量が多いサーバ（アプリケーションサーバーなど）は距離的にオンプレミス環境に近いクラウドサービスの利用が望ましい。（同一構内であることが望ましい。）

7.12.3. 関連するスタイル

- データセンター拡張(データはオンプレミス)
- データセンター拡張(データもクラウド配置可能)
- ハイブリッドなクラウド基盤を利用したシステム

7.12.4. 解説

クラウドサービスの利用を推進していく中で、Oracleライセンス規約の準拠と機密情報の信頼できる場所への保管は最後まで課題として残るケースが多いと言えます。またコストで担保できる要件でも無いことから現実的にはオンプレミス環境に残すという選択を取るケースが大半であると考えられます。

このケースではデータベースとアプリケーションサーバー間の通信が遅延時間や応答時間に影響を与えることになり、実際のシステム利用に耐えられるかがポイントになります。

オンプレミス環境とパブリッククラウド環境が同一構内にある場合はローカル接続と同等の性能を出すことが可能になると言えるが、WANを経由する場合には広帯域をサポートしている閉域ネットワーク接続サービスの利用が考えられるが、遅延時間や応答時間がシステム利用に十分に耐えられるかを見極める必要があります。

7.12.5. 関連する技術

- Azure Express Route

7.12.6. 参考情報

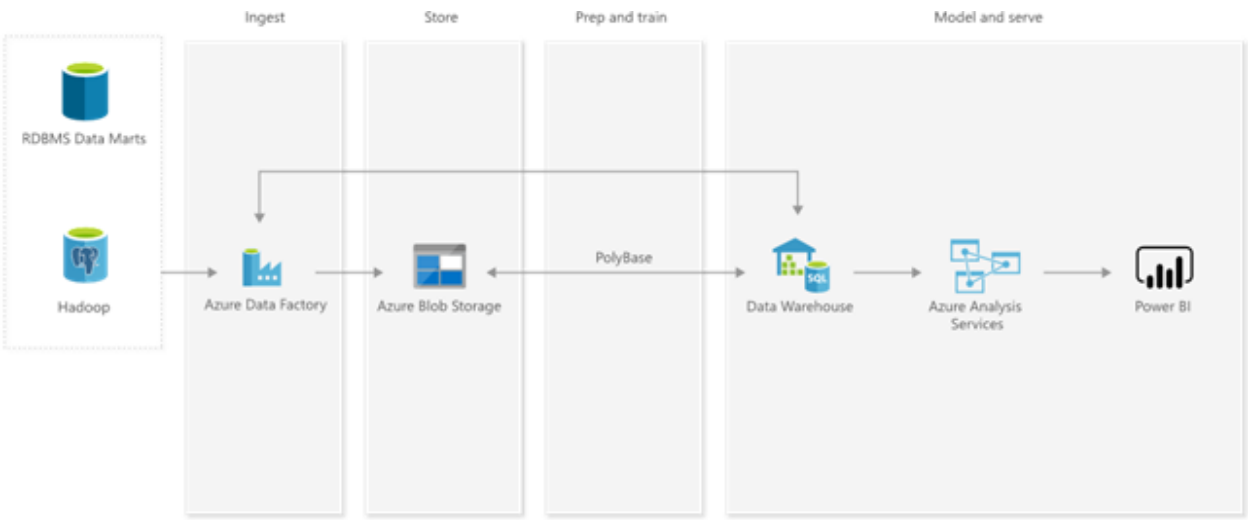
- [ExpressRoute - 仮想プライベート クラウド接続 | Microsoft Azure]

7.13. クラウドによるオンプレミスDBの分析と可視化

7.13.1. 状況および課題

- OracleサーバーはOracleライセンス規約及びセキュリティポリシーを準拠する必要があるため、オンプレミス環境に残す必要がある。
- 分析・可視化システムはクラウドサービスを有効につかって構築したい。

7.13.2. 解決方法



- OracleサーバーはOracleライセンス規約及びセキュリティポリシーを準拠する必要があるため、オンプレミス環境に残す。
- 分析・可視化システムはクラウドサービス上に新規に構築する。
- Azure Data Factoryを利用して既存データベースからのデータ収集、処理、変換を行い、データウェアハウスに分析用のデータを格納する。
- データウェアハウスとしてAzure SQL Data Warehouseを利用し、分析・可視化システムでデータを活用できるようにする。

7.13.3. 関連するスタイル

- データセンター拡張(データもクラウド配置可能)
- ハイブリッドなクラウド基盤を利用したシステム

7.13.4. 解説

Azure Data Factoryはデータ保存や処理、移動といったサービスを効率的かつスケーラブルで信頼性の高いデータ生成パイプラインとして構成、利用することができます。

主な機能は以下の通り。

- オンプレミスやクラウドサービスから取得したデータを結合、集計、変換するデータ主導のワークフロー（パイプライン）構築
- 様々なデータソース（半構造化、非構造化、構造化）の信頼できる情報への変換
- ビジネスインテリジェンス（BI）、分析ツール、その他のアプリケーションで容易に使用できるデータの生成
- JSONスクリプトによる複雑なデータ処理の設定
- Azureプレビューポータルが提供するUIによるパイプラインの監視と管理

Azure SQL Data Warehouseは、超並列処理（MPP）を利用して、ペタバイト単位のデータに対して複雑なクエリを短時間で実行することができるクラウドベースのエンタープライズデータ ウェアハウス（EDW）です。

データウェアハウスのような大規模なデータ基盤はオンプレミスで構築する選択肢もちろんありますが、パブリッククラウド上のサービスを利用することでスケールを担保し、スモールスタートを開始するということも有力な選択肢です。このユースケースのようにオンプレミスでこれまで利用していたデータベースに対してのデータウェアハウスと分析基盤をパブリッククラウド上に保持し、そのデータ連携もクラウドサービスを利用するというパターンはデータの扱いに関しての典型的なユースケースの1つです。

7.13.5. 関連する技術

- Azure Data Factory
- Azure SQL Data Warehouse
- Azure Analysis Service

7.13.6. 参考情報

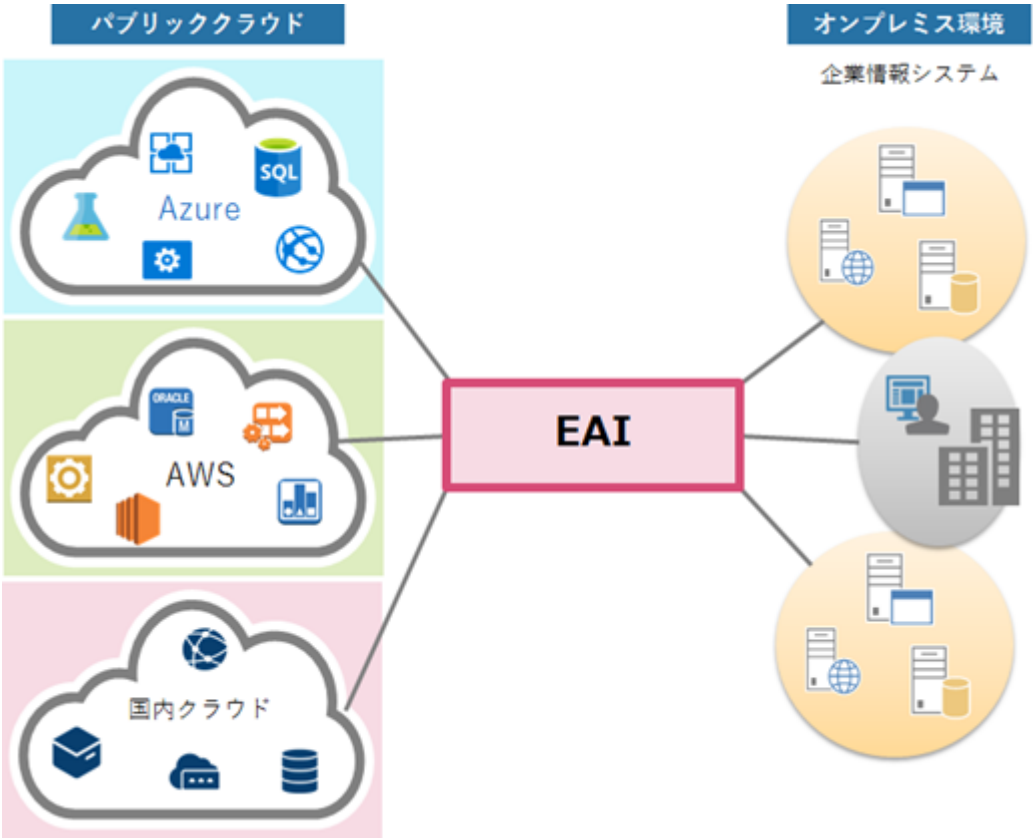
- Data Factory - データ統合サービス | Microsoft Azure
- SQL Data Warehouse – Azure SQL 用 ソリューション | Microsoft Azure

7.14. EAI利用によるクラウドをまたいだデータの利活用

7.14.1. 状況および課題

- オンプレミス環境にある既存データベースの利活用を図りたい。
- 既存システムはオンプレミス環境やクラウドサービス環境に分散している。
- クラウドサービスの利用に当たってはOracleライセンス規約に抵触しないように注意する必要がある。
- 社内のセキュリティポリシーからデータベースは仕様・構成を把握できる環境に配置する必要がある。

7.14.2. 解決方法



- OracleサーバーはOracleライセンス規約及びセキュリティポリシーを準拠する必要があるため、オンプレミス環境に残す。
- データ連携ツール(EAI)を利用し、オンプレミス環境、クラウドサービス環境に分散しているシステムと既存データベースを連携させる。

7.14.3. 関連するスタイル

- データセンター拡張(データもクラウド配置可能)
- ハイブリッドなクラウド基盤を利用したシステム

7.14.4. 解説

　業務システムのクラウド化が進み、クラウドサービスとオンプレミス環境間のデータ連携、あるいはクラウドサービス間でのデータ連携が必要不可欠になってきています。

　しかし、システム連携を行うにはプログラミング開発を伴うケースが多く、システムによってはプログラミング開発負荷が非常に高い、もしくは困難となるケースも多いと言えます。

　このような課題を解決する手段として、データ連携ツールの利用が考えられます。

　AWS（Amazon Web Service）やMicrosoft Azure、Google Appsなどのクラウド上に置いたシステムとオンプレミス環境にある既存システムとの連携やデータの活用を図ることが容易に可能となります。

7.14.5. 関連する技術

- EAI(Enterprise Application Integration)

7.14.6. 参考情報

- DataSpider Servista V4
- ASTERIA Warp

7.15. ファイルサーバーのスモールスタートとハイブリッド化

7.15.1. 状況および課題

- 業務システムで使用するファイル サーバーを新規に準備する必要がある。
- 予算の都合上、すぐにファイルサーバーを既存システムに追加することが難しく、どの程度容量を使用するかどうか判断できない。
- 今後、オンプレミス上にファイルサーバーを導入した際に、効率よくデータアクセスや容量を扱いたい。
- 機密情報を取り扱う予定があるため、適切なアクセス権を設定する必要がある。

7.15.2. 解決方法

- 使用した容量分だけ料金が発生するクラウドストレージサービスを利用する。
- データ連携ツールを用いて、オンプレミス上のファイルサーバーを連携、使用頻度の高いファイルはオンプレミスに、低いファイルはクラウドに、といったデータ階層化を行う。
- クラウドのディレクトリ機能を用いてアクセス権を設定し、データ連携ツールを用いている場合は、オンプレミスのディレクトリサーバーと連携しアクセス権を設定する。

7.15.3. 関連するスタイル

- データセンター拡張(データもクラウド配置可能)
- ハイブリッドなクラウド基盤を利用したシステム

7.15.4. 解説

業務を早急に行うためには、利用するための機器を準備する必要がありますが、容量などリソースのサイジングや、そのサイジング結果に合わせた機器の選定などで時間を要することが往々にしてあります。

まずはスモールスタートとして使ってみる、ということであればCommon Internet File System (CIFS)やServer Message Block (SMB)プロトコルを介してアクセス可能な、ファイル共有型のクラウドストレージサービスの利用が考えられます。

ファイル共有型のクラウドストレージサービスは、利用した分だけ料金が発生するため、あらかじめ容量のサイジングや、ハードウェアの購入は必要ありません。また、クラウドベンダーにもよりますが、アクセスする際のデータ暗号化やクラウドのディレクトリ機能を用いてアクセス権を設定することも可能なため、セキュリティ面での不安はないと言えます。

加えて、データの複製、スナップショットやバックアップといったマネージドサービスを提供しているクラウドベンダーも出てきており、従来のオンプレミスファイルサーバー運用と比較して、運用の手間やコストを低減することも可能となっています。

クラウドストレージサービスを利用したファイルサーバー利用の懸念点としては、基本的にインターネット経由でのアクセスとなるため、ファイル応答速度がオンプレミスよりも低下する点があります。回避策としては、専用線を用いることも可能ですが、オンプレミスファイルサーバーとデータ連携ツールを用いて、データ階層化を行うことでも可能と考えられます。データ階層化は、オンプレミスファイルサーバーのストレージ容量を拡張するだけでなく、使用頻度の高いファイルはオンプレミスに、低いファイルはクラウドに、と使い方が可能であるためです。

最後に、データ連携ツール (Azure File Sync)を用いてオンプレミスのファイルサーバーとクラウドストレージサービスが連携している場合、オンプレミスのディレクトリ機能を用いてアクセス権を設定することが可能となっています。そのため、利用者観点で見ると従来のオンプレミスファイル サーバーと同様に使用することが可能となっています。

7.15.5. 関連する技術

- Azure Files
- Azure File Sync
- AWS Storage Gateway

7.15.6. 参考情報

- File Storage | Microsoft Azure

- AWS Storage Gateway（オンプレミスをクラウドと接続するハイブリッドストレージ）｜AWS

7.16. ログ収集および分析基盤のクラウド上への構築

7.16.1. 状況および課題

- 全てのシステムにおいて適切に必要なログが収集されなくてはいけない。
- どれだけのログが発生し、収集、補完する必要があるのかを正確に見積もることは難しい。
- 必要な最大サイズのログを補完可能なログ保管システムをシステム構築当初から用意しておくのは無駄が大きい。
- ログ収集、分析システム基盤の維持のための仮想マシン、ストレージシステム等の維持管理に負荷がかかる。
- 収集したログが収集するだけとなってしまい、活用できていない。
- 収集したログの量が増えると必要な際に単純に検索をしようとするだけでも長い時間がかかってしまう。現実的にログの活用を進めることが難しい。

7.16.2. 解決方法

- ログ収集および分析基盤としてクラウド上のサービスを利用する
- ログはクラウド上で収集、分析を行う

7.16.3. 関連するスタイル

- クラウドを補完的に利用

7.16.4. 解説

　ログ収集システムは想定により容易に巨大なシステムとなり、それをオンプレミスに作成、維持することはその投資に対してリターンが少ないことが多い。法令や組織のルールで縛られ何年間もログを大量にためることを行うが、実際には「データを保持している」というアリバイ作りに終始し、現実的に利用できないシステムになってしまっているケースも多い。

　クラウド上のログ収集、分析サービスの利用はスモールスタートで始められ、従量課金のためコストも安く、データの可視化や他システムとの連携も容易である。

　オンプレミスであれば必要となってくるログ収集および分析のためのサーバーやストレージシステムも不要であり、管理も必要ない。サイジングすらも不必要となる。

　ログを貯めるだけではなく、蓄積したログからの可視化やAIによる修正アドバイスの提示等、クラウドサービスを利用することによるメリットは大きい。

7.16.5. 関連する技術

- Azure Monitor
- Azure Log Analytics

7.16.6. 参考情報

- Azure Monitor の概要 | Microsoft Docs
- Azure Monitor での Log Analytics データの分析 | Microsoft Docs

7.17. 自動処理のクラウド化

7.17.1. 状況および課題

- 日々自動的に定期実行するタスクや都度実行するタスク等がある。
- 個々人でそれぞれスクリプトを記述する等部分的な自動化がなされている領域もあるし、全くなされていないものもあり品質にバラつきがある。
- タスクを自動的に定期実行させるためだけに可動させ続けるサーバーがあり、OSのアップグレードや障害対応等の運用タスクが必要となっているが、可動していない時間が大部分である。

7.17.2. 解決方法

- クラウド上でロジック、スクリプト等を管理及び共有する。
- 定期的な実行、アドホックな実行含めてクラウドサービスにて実行させる

7.17.3. 関連するスタイル

- クラウドを補完的に利用

7.17.4. 解説

　自前で基盤を保持および維持管理せずにクラウド上にコードを保持させクラウド上で実行させることが可能です。オンプレミス上の特定のサーバー上で実行しなければいけないような処理に関しても、例えばAzure Automationであれば「ハイブリッドワーカー」という仕組みで実行させることが可能です。

　加えてクラウド上に配置することで特定のイベント発生時に自動で処理を開始させることや、クラウド上の他の自動処理をキックすることや他のサービスと連携させることも容易にできるようになります。大規模なリソースが必要なケースでもクラウド側で自動的にスケールさせてくれるサービスもある、災害対策にも強くなる等、使った分だけの従量課金でありコスト削減となる等、クラウド化による様々なメリットが得られます。

7.17.5. 関連する技術

- Azure Automation
- Azure Functions
- Azure AppService(Webジョブ)

7.17.6. 参考情報

- Azure Automation – クラウド オートメーション サービス | Microsoft Azure
- Azure Functions—サーバーレス アーキテクチャ | Microsoft Azure
- Azure App Service で Web ジョブを使用してバックグラウンドタスクを実行する | Microsoft Docs

7.18. プライベートクラウド構築作業からの解放

7.18.1. 状況および課題

- オンプレミスに仮想基盤あるいはプライベートクラウド基盤を構築している。
- 環境の構築、運用、保守、定期的なバージョンアップ、拡張作業等含めて多くの人数と多くの工数を使っている。
- 基盤への定期的なセキュリティ更新プログラムの適用だけでも膨大な工数が必要となってしまい、適用頻度が低くセキュリティ上問題がある。
- 最新の技術や最新OSがサポートさせる状況に基盤を維持し続けることができていない。

7.18.2. 解決方法

- 統合システムとしてAzure Stackを導入する。
- 構築作業は高度に標準化、自動化されており、OEMベンダーによって実施される。
- 基盤更新作業は管理ポータルからUpdateボタンをクリックするのみで完全に自動化されている。テナントのサービスへの影響も最小化されている。
- Update作業を継続的に実施することで、基盤自体も進化し続ける。

7.18.3. 関連するスタイル

- オンプレミス環境のみ(クラウドネイティブシステム)

7.18.4. 解説

　オンプレミスに仮想基盤を超えるプライベートクラウドを構築し、運用、更新し続けることは非常に大変な業務であり技術力も工数も必要です。パブリッククラウドを利用すれば何も考慮しなくてもよい部分も多く、パブリッククラウドの採用も1つの手ですが、本書でも解説している通り様々な理由でオンプレミス上にのこるシステムは多くあります。

オンプレミスのプライベートクラウドにもパブリッククラウドと同じ考え方を適用スルならば、重要なのは基盤自体の構築、運用ではない部分であり、任せられる部分は極力任せるという方針も選択肢となるでしょう。

Azure Stackであればプライベートの構築および更新の手順が完全に確立され、固定化されており、システムを「作る」ことから開放され、「使う」事に集中できます。

7.18.5. 関連する技術

- Azure Stack

7.18.6. 参考情報

- Azure Stack とは | Microsoft Azure

8. 用語集

用語	意味
Active Directory	Windowsサーバーに搭載されているディレクトリサービス。ネットワーク上に存在するサーバー、クライアント、プリンタなどのハードウェア資源や、それらを使用するユーザーの属性、アクセス権などの情報を一元管理することが可能。
AI(Artificial Intelligence)	人間の知的能力をコンピュータ上で実現する、様々な技術・ソフトウェア・コンピューターシステム
ARM(Azure Resouce Manager)	Azure上に存在するリソース全般を管理するための仕組み
AWS Storage Gateway	オンプレミスアプリケーションによる AWS クラウドストレージのシームレスな使用を可能にするハイブリッドストレージサービス
AWS (Amazon Web Service)	Amazon 社により提供されているクラウドコンピューティングサービス
Azure	Microsoft 社により提供されているクラウドコンピューティングサービス
Azure Analysis Service	Azure が提供するフルマネージドの分析ソリューションサービス
Azure App Service	Web アプリ、モバイル アプリ、API アプリの迅速な構築、展開、およびスケーリング可能なフルマネージドのアプリケーションプラットフォーム
Azure Automation	Azure 環境と非 Azure 環境を一貫性をもって管理するクラウド ベースのオートメーションサービスを提供。プロセスの自動化、更新の管理、および構成機能を提供。
Azure Container Registry	Azure Container Registry では、あらゆる種類のコンテナーイメージをビルド、格納、管理できます。
Azure Container Service	Azure Container Service では、運用環境で使用できる Kubernetes、DC/OS、または Docker Swarm クラスタを迅速に展開することができます。
Azure Data Factory	クラウド データ統合サービスである Data Factory を使用して、データの保管、移行、および処理のサービスを自動化データ パイプラインにまとめることができます
Azure DevOps	コード共有や作業の追跡など、ソフトウェア開発チームを支援するツールを提供します。
Azure Files	Azure 上で、Server Message Block (SMB) プロトコルを介してアクセスできる、フルマネージドのファイル共有サービスです
Azure File Sync	オンプレミスのファイル サーバーとの互換性を維持したまま Azure Files で組織のファイル共有を一元化できます。
Azure Front Door	グローバルな Web アプリケーションを高速に配信するためのセキュリティで保護されたスケーラブルなエントリ ポイントが提供されます。

用語	意味
Azure IoT Edge	IoT Hub 上に構築されるモノのインターネット (IoT) サービスです。 このサービスは、クラウド内ではなく、インテリジェントなエッジデバイスで分析可能となります
Azure Log Analytics	ハイブリッドIT環境をより詳細に把握でき、アドバンスドアナリティクスポータルを使って、ワンクリックでパフォーマンス問題を診断する
Azure Machine Learning	データを活用した予測分析ソリューションの構築、テスト、デプロイをドラッグ アンド ドロップで行うことができる、コラボレーションに対応したツールです。
Azure Monitor	最新の詳細なパフォーマンスと使用率のデータを取得したり、全てのAPI呼び出しを追跡するアクティビティログにアクセスしたり、Azure リソースの問題をデバッグするための診断ログを取得したりする
Azure Site Recovery	オンプレミスの物理サーバーおよびVMwareやHyper-V上の仮想マシンをAzureに移行する。または、カスタムレプリケーションタイミング、分離テスト、最終移行カットオーバーをサポートする
Azure SQL Data Warehouse	SQL Data Warehouse は、超並列処理 (MPP) を利用して、ペタバイト単位のデータに対して複雑なクエリを短時間で実行するクラウドベースのエンタープライズ データ ウェアハウス (EDW) です。
Azure Stack	オンプレミスで Azure サービスと機能を拡張します。Azure と Azure Stack で一貫性を維持したままハイブリッド クラウド アプリケーションを構築、デプロイ、および運用することができます。Azure Stack は、ソフトウェアと検証済みハードウェアの統合システムとして販売されます。
Azure Traffic Manager	Azure Traffic Manager は、お好みのルーティング方法に基づいて受信 DNS 要求を迅速かつ効率的に転送するために DNS レイヤーで動作します。
Azure Web Apps	Web アプリケーション、REST API、およびモバイル バックエンドをホストするためのサービスです。
CaaS(Container as a Service)	コンテナを用いた開発からデプロイ、運用までのアプリケーションライフサイクル全体を支援するサービスを指す
CD(Continuous Delivery)	コンテナを用いた開発からデプロイ、運用までのアプリケーションライフサイクル全体を支援するサービスを指す
CI(Continuous Integration)	ソフトウェア開発において、ビルドやテストを頻繁に繰り返し行なうことにより問題を早期に発見し、開発の効率化・省力化や納期の短縮を図る手法。
CIFS(Common Internet File System)	CIFSとは、Windowsのファイル共有サービスで利用されている通信プロトコルSMBを拡張し、Windows以外のOSやアプリケーションソフトでも利用できるよう仕様を公開したもの。
Cloud Foundry	オープンソースのPlatform as a Service(PaaS)ソフトウェアの名称
DevOps	ソフトウェアの開発担当と導入・運用担当が密接に協力する体制を構築し、ソフトウェアの導入や更新を迅速に進めること。「Development(開発)」と「Operations(運用)」の略語を組み合わせたもの
DNS	インターネットなどのTCP/IPネットワーク上でドメイン名やホスト名とIPアドレスの対応関係を管理するシステム

用語	意味
Docker	コンテナ型の仮想化ソフトウェアの一つ。Docker社(旧dotCloud社)が開発を創始したもので、オープンソースとして公開されている。
EAI（Enterprise Application Integration)	企業内の複数のシステムを連携させ、データやプロセスを統合させること、あるいはそのためのミドルウェアの総称である。
EDW(Enterprise Data Warehouse)	全社での情報活用を目的にデータを統合化／一元化させたデータウェアハウスです
EOS(End of Support)	サポート終了の意味で、製品についての問い合わせ受け付けや修理、交換などのサポートサービスの提供が打ち切られる期限を表す。
Express Route	お客様のオンプレミスのインフラストラクチャからインターネットを経由せずに Azure サービスにアクセスできるようにするものです。
FaaS(Function as a Service)	クラウド上で機能（関数）の実行環境を提供するサービスのこと
Google Apps	Google の提供するコミュニケーションツールとコラボレーションツールがセットになったグループウェア
Google Cloud Platform	Google 社により提供されているクラウドコンピューティングサービス
HCI(Hyper Converged Infrastructure)	SANやNASなどの外部共有ストレージを利用せずに、業界標準のx86サーバーに搭載されたローカルのSSD、HDDをソフトウェア機能によって統合したアプライアンス製品です。
IaaS(Infrastructure as a Service)	情報システムの稼動に必要な機材や回線などの基盤(インフラ)を、インターネット上のサービスとして遠隔から利用できるようにしたもの
IDaaS(Identity as a Service)	アイデンティティ（Identity）の管理をSaaSやIaaSなどと同じくクラウドにて管理するサービスです。
Infrastructure as Code	コンピューティング・インフラ（プロセス、ベアメタルサーバー、仮想サーバー、など）の構成を管理したり、機械処理可能な定義ファイルを設定したり、プロビジョニングを自動化する手法の総称
JSON(JavaScript Object Notation)	JavaScriptにおけるオブジェクトの表記法を応用したデータ形式。テキスト(文字)ベースの形式の一つで、多数の要素が複雑な構造で組み合わせられたデータを簡潔な表記で書き表すことができる。
Kubernetes	コンテナ化したアプリケーションのデプロイ、スケーリング、および管理を行うための、オープンソースのコンテナオーケストレーションシステム。
MPP(Massively parallel machine)	超並列処理。極めて多くのマイクロプロセッサやコンピュータを協調して動作させ、一つの高性能なシステムを構築する手法。
NoOps (No Operations)	「人間によるシステム運用作業の最小化」を目指す継続的な活動を指す
OEM(Original Equipment Manufacturer)	発注元企業の名義やブランド名で販売される製品を製造すること。

用語	意味
Openshift	DockerとKubernetesを基盤として採用しているオープンソースのPaaS (Platform as a Service)ソフトウェア。Red Hat社が主に開発を行い、商用サポートも提供している。
PaaS(Platform as a Service)	アプリケーションソフトが稼動するためのハードウェアやOSなどの基盤(プラットフォーム)一式を、インターネット上のサービスとして遠隔から利用できるようにしたもの。また、そのようなサービスや事業モデル。
RPO(Recovery Point Objective)	システム障害などでデータが損壊した際に、復旧するバックアップデータの古さの目標。
RTO(Recovery Time Objective)	情報システムが障害などで停止した際に、復旧するまでの目標時間。
SAN(Storage Area Network)	複数のコンピュータとストレージ(外部記憶装置)の間を結ぶ高速なネットワーク。
SDK(Software Development Kit)	あるシステムに対応したソフトウェアを開発するために必要なプログラムや文書などをひとまとめにしたパッケージのこと。
Site Reliability Engineering	サイトの信頼性向上のためにソースコードに手を加えてパフォーマンスを改善、スケーラビリティ向上を担う開発チームのことを指します。
SLA(Service Level Agreement)	稼働時間と接続に関する Microsoft の確約内容を示す契約です。 Azure サービスごとに固有の SLA があります。
SMB(Server Message Block)	ネットワーク(LAN)上の複数のWindowsコンピュータの間でファイル共有やプリンタ共有などを行うためのプロトコルおよび通信サービス。
Visual Studio	Microsoft社が開発・販売している、ソフトウェア開発のための統合開発環境製品。
VPN(Virtual Private Network)	公衆回線を経由して構築された仮想的な組織内ネットワーク。また、そのようなネットワークを構築できる通信サービス。

9. ハイブリッドクラウド研究会の活動への参加の誘い

　ハイブリッドクラウド研究会では本ドキュメントの継続的な更新も含めて、コミュニティとしての活動を継続的に行っていきますのでぜひ参加いただければと思います。

タイトル	URL	説明
Webページ	http://www.hccjp.org	ハイブリッドクラウド研究会の公式Webサイト
Facebookグループ	https://www.facebook.com/groups/hybridcloud/	ハイブリッドクラウド研究会のFacebookグループ。どなたで個人で自由に参加ください。
Twitterアカウント	https://twitter.com/hccjp1	ハイブリッドクラウド研究会の公式Twitterアカウントです。各種のお知らせをつぶやきますのでお気軽にフォロー下さい。
Twitterハッシュタグ	https://twitter.com/hashtag/hccjp?src=hash	ハイブリッドクラウド研究会のハッシュタグです。ハイブリッド研究会に関連する事柄はこのハッシュタグをつけていただければと思います。
Compass	https://hybridcloud.connpass.com	ハイブリッドクラウド研究会主催のイベントはこちらのConnpassで管理されます。各種イベントへの参加を希望される方はぜひこちらにメンバーとして登録いただければと思います。

10. あとがき的なもの

現在はクラウドを有効活用する企業が革新的なサービスを提供し、競合他社に対して優位性を持つのが当たり前の時代になりました。クラウドを有効活用した新しいビジネスモデルも多数生まれています。UberやNetflix、Airbnbなどの革新的なビジネスモデルと高度なクラウド利用の例は非常に有名になりました。世界の時価総額Top10にはApple、Alphabet、Microsoft、Amazon、Tencent Holding、Facebook、Alibaba Group Holdingと米国および中国のIT企業が7社入っていることも、時代を象徴しています。しかし、残念ながら世界の時価総額ランキングトップ企業のなかに日本企業はなかなか出てきません。今こそ日本企業も、よりITの力を取り入れて企業力を高める必要があるというのは、多くの人が同意するところでしょう。

そんななか、既に日本企業の70%近くが「ハイブリッドクラウドを導入済み」と回答しているというデータがあります。しかし「クラウドを使っていると言ってもメールサーバをSaaSに切り替えただけ」「仮想マシンをパブリッククラウド上でホスティングしているだけで、結局やっていることもスピード感もオンプレミスの仮想環境と全く同じ」「積極的にパブリッククラウドの活用を決めたけれど、大量に残る社内のITとビジネスの距離は広がるばかり」。――そんな状況でも「ハイブリッドクラウド導入済み」と呼んでしまっている実態があると感じています。

ハイブリッドクラウド研究会ではこのような状況を認識しながら、あらためて「クラウドとは何か」「ハイブリッドクラウドとは何か」「より連続性・一貫性のある基盤を活用することで、迅速にビジネス上の成果を挙げる方法とは?」といった問いに対し、企業の枠を越えてユースケースや方法論を議論・検証することで、日本へのハイブリッドクラウド導入を促進していきたいと考え活動しています。その活動の中で、日本の企業が自分の組織の置かれている状況をもとにしながらクラウド導入を進めていく上でのガイドラインを提示することができたら意味があるものになるだろうという意見が多く出て、本ガイドラインを作成することになりました。

ぜひ本ガイドラインを活用して自組織の現状の「スタイル」を認識し、その中でのハイブリッドクラウドの活用方法を参考にしていただければとおもいます。そしてよりビジネスに貢献できる「次のスタイル」も模索していっていただければと思います。

最後に、本ガイドラインはこれからも継続的に更新予定です。内容に関しての質問、改善点等のコメントは下記Facebookグループにお寄せいただければと思います。

- ハイブリッドクラウド研究会Facebookグループ