Microsoft Azure

Azure Light-up Well-Architected Framework

テクニカルパート 概要編

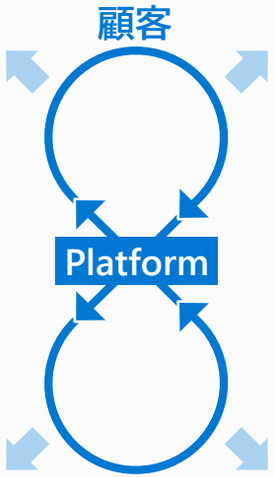
Time to Market が勝負を決める＝ Agility（迅速さ）× Scalability

競争のポイント

製品・サービス差別化・
顧客囲い込み・コスト競争

顧客基盤の拡大と関係強化・
顧客情報の獲得と活用につながる
俊敏な顧客への価値提供

ネットワーク外部性
Network Externality



より多くの
良質な価値が
ある Network・
Platform が魅力的

より多くの
良質な買い手を
擁する Network・
Platform が魅力的

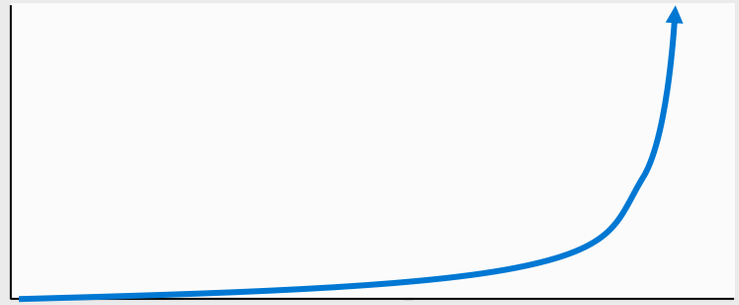
User-Base
顧客・
価値提供者の
増加

Data
活用できる
情報量・幅の
増加

Network・
Platform の
価値向上

幾何級数的成長
Exponential Growth

双方の参加が一定の段階を超えると、相乗効果で急速に魅力が高まり、参加者が激増→プラットフォームの魅力度・競争力が更に高まる



情報量の増加による生み出される価値の増分 ＞ 情報量の増加に伴って生じる情報処理コスト

現代のビジネス・組織が備えるべき 3 つの特性

Agility
Rapid, Frequent Time to Market

高品質・高信頼性のシステムの多頻度・高速投入による、事業機会獲得・競争優位構築

Efficient/Flexible
Scalability

小規模低コストで試行可能 & 急拡大可能であることによる、大胆かつ柔軟な戦略実行

Well-Architected Framework の５つの柱

オペレーショナル エクセレンス

DevOps によるシステム設計・開発・実装の一体化

- それを可能にするインフラ整備
- モニタリングとパフォーマンスの継続的管理

パフォーマンス 効率

効率的な方法でワークロードをスケーリング（スケールアウト）

- モジュール化・Microservice 化等スケーリングを考慮した設計
- ピーク負荷でのテスト
- 容量計画と監視・自動スケーリング

コスト最適化

需要の拡縮に比例する従量課金のワークロード コスト

- 最適な資源の最適量利用
- 継続的なコスト監視と最適化プロセスの組み込み

信頼性

信頼性＝回復性＆高可用性（システム・データ）

- 信頼性の要件定義と設計
- 監視と復旧の仕組み（地理的分散他）

セキュリティ

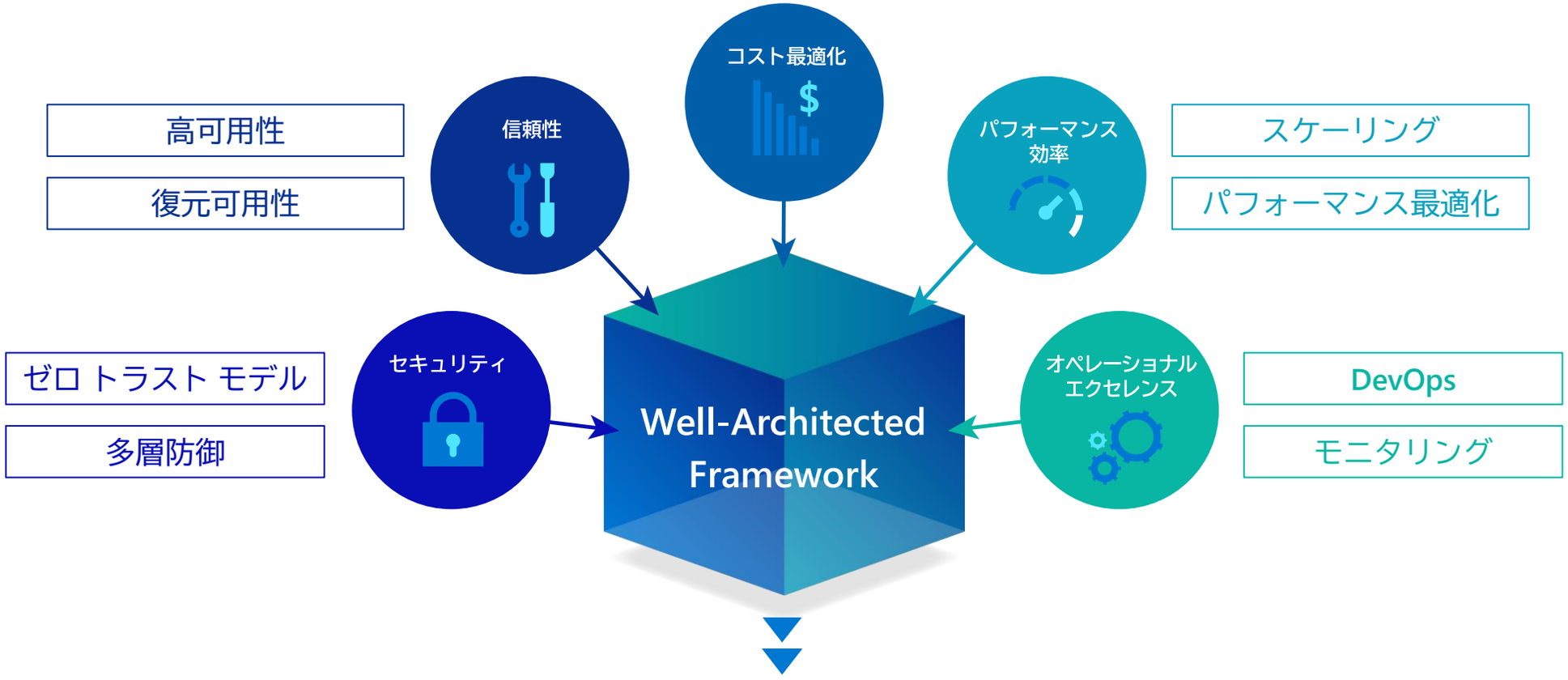
重要なデータやシステムへの攻撃・悪用に対し機密性、整合性、可用性を保証

- 協働責任モデルとクラウド プロバイダーの高度な専門性の活用
- ID・ゼロトラスト・多様認証等

High
Reliability/Security

顧客・社会からの高い信頼維持向上とリスク軽減・管理・事業継続性担保

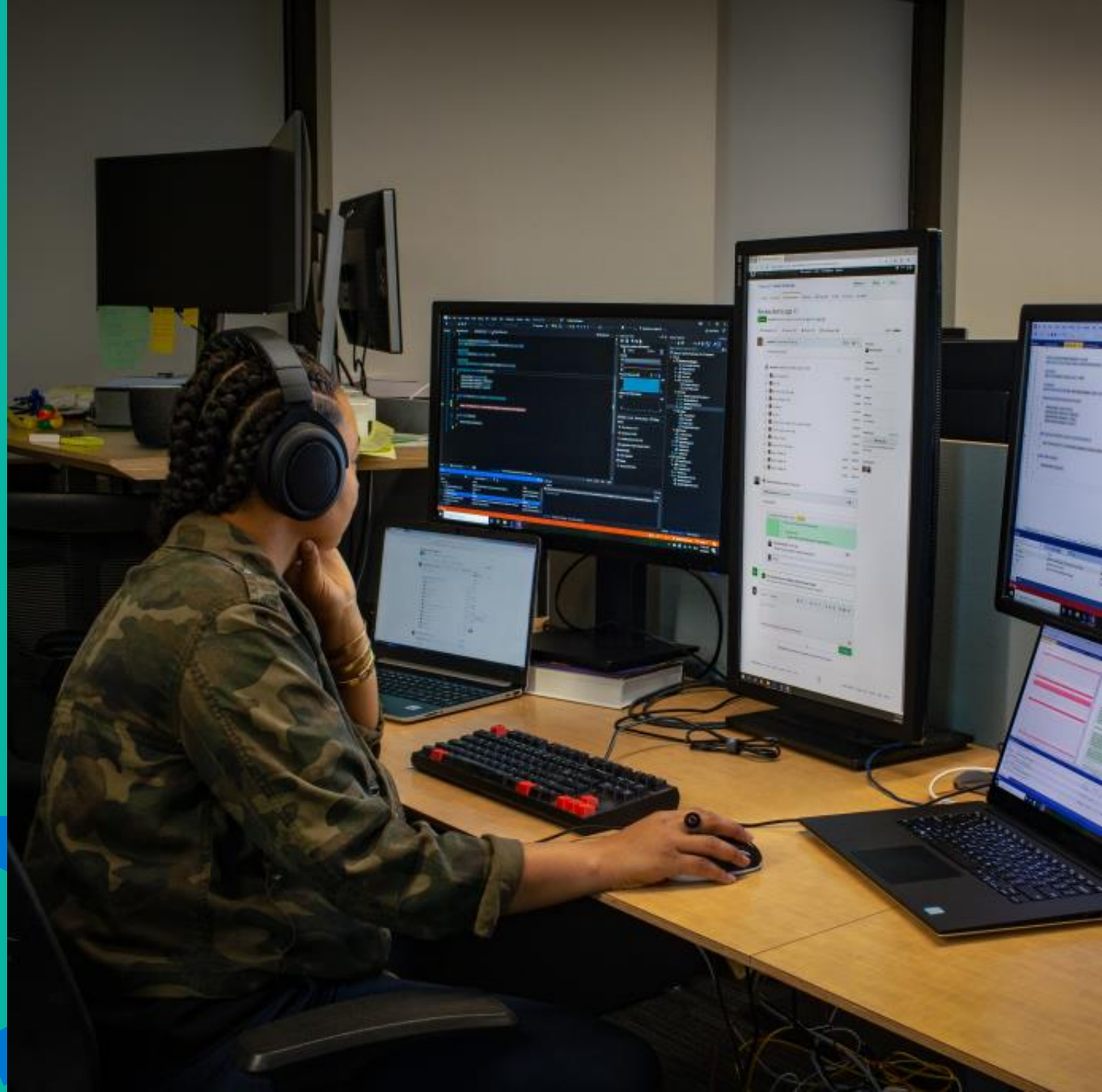
Well-Architected Framework を活用したプロダクト開発



- ☑ 市場への早期投入
- ☑ 変化に対応できる
- ☑ 低コストで始められる

オペレーショナル エクセレンス

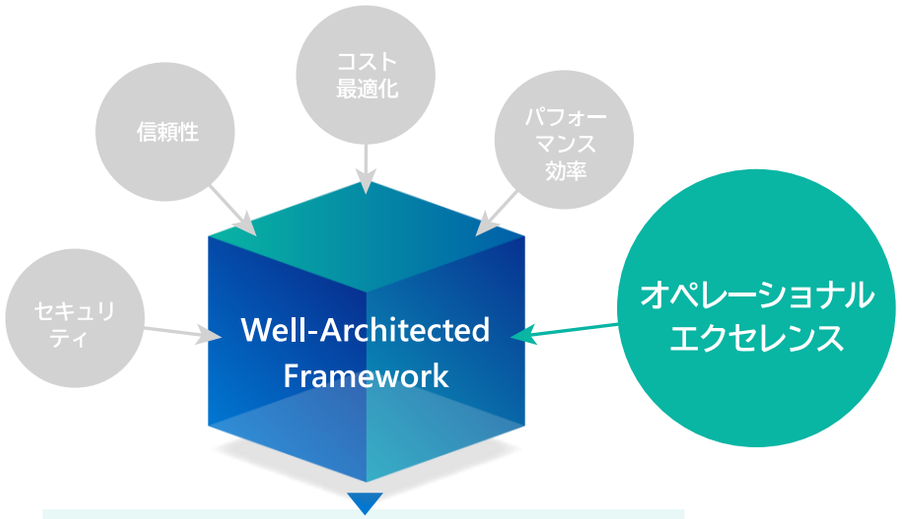
OPERATIONAL EXCELLENCE



オペレーショナル エクセレンスの観点

運用上の機能を強化することで、開発やリリースのサイクルのスピードアップと、アプリケーション ユーザーのエクスペリエンス向上を実現

- アプリケーションがどのように実行中であるかを完全に表示可能にして、ユーザーのために最適なエクスペリエンスを確保する
- 開発とリリースの実務をより機敏なものにする



- ☑ 開発のスピードアップ
- ☑ リリース サイクルの短縮
- ☑ トラブル シューティングの効率化

DevOps

- 継続的インテグレーション／デプロイ（CI/CD）
- インフラ構築の自動化（Infrastructure as Code）

モニタリング

- アプリケーション監視（ログ、例外）
- テレメトリ収集（パフォーマンス、監査）

DevOps と継続的インテグレーション

DevOps とは、エンド ユーザーへの価値の継続的デリバリーを可能にするために、人、プロセス、および製品を結合したもの

DevOps と継続的インテグレーションを考慮に入れた設計

- コードとしてのインフラストラクチャを使用したデプロイの自動化
- アプリケーション テストの自動化

関連する **Azure** サービス / テクノロジー

- Azure Pipelines, GitHub Actions
- Azure Resource Manager テンプレート

Azure Pipelines

監視と診断

パフォーマンスの問題や非効率なコストを特定し、イベントを関連付けて、
問題をトラブルシューティングする能力を高めることが可能

DevOps と継続的インテグレーションを考慮に入れた設計

- アーキテクチャ内で何が起きているかを監視するためのしくみ
  - アプリケーションログ、Web サーバーログ、プラットフォームに組み込まれている診断などのソースから生データを生成
- テレメトリ データを使用して、傾向をつかみ、運用チームに警告する

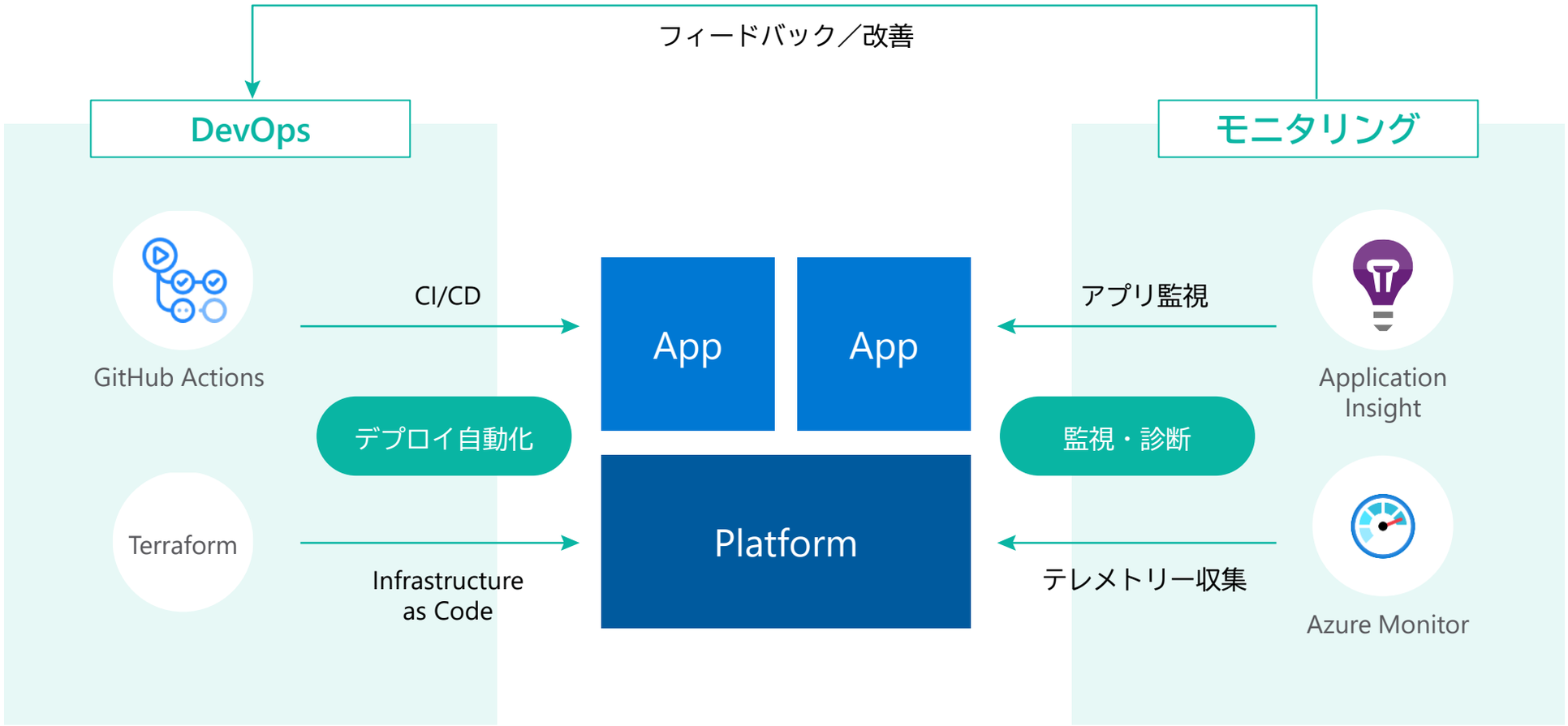
関連する **Azure** サービス / テクノロジー

- Azure Monitor
- Application Insights

Azure Monitor

Application Insights

「オペレーショナル エクセレンス」観点での Azure 活用



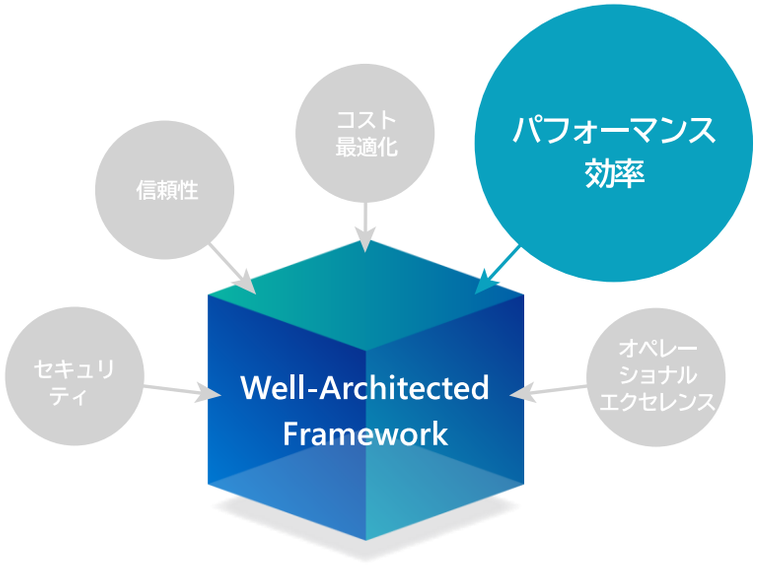
パフォーマンス効率

PERFORMANCE EFFICIENCY

パフォーマンス効率

パフォーマンス効率とは、アプリケーションが利用可能なリソースと、そのアプリケーションへの需要が一致すること

- リソースをスケーリングする
- 潜在的なボトルネックを識別して最適化する
- 最高のパフォーマンスが得られるようアプリケーション コードを最適化する



スケーリング

- スケールアウト
- 自動スケール

パフォーマンス最適化

- ネットワーク最適化
- ストレージ最適化
- コード最適化

スケーリング

「スケールアップ」「スケールアウト」の使い分け

- どちらも、リソースを削減し、コストの最適化を実現する
- 「スケールアウト」型が、よりクラウド パワーを活用できる

自動スケールの活用

- パフォーマンス要件に合わせてリソースを動的に割り当てる
- PaaS などマネージド サービスの方が導入しやすい

関連する **Azure** サービス / テクノロジー

- Azure Cosmos DB
- App Service

Azure Cosmos DB

App Service

パフォーマンス効率

パフォーマンスの最適化

ネットワーク

サービスとサービスの間にメッセージング レイヤーを追加する

- メッセージング レイヤーによりバッファーが形成される
- 処理が追い付かなくなった場合でも要求のフローが維持される

関連する **Azure** サービス / テクノロジー

- Event Hubs
- Azure Queue Storage
- Service Bus


Event Hubs


Storage Queue


Service Bus

ストレージ

パーティション分割戦略

- スケーラビリティが向上し、競合が少なくなる

キャッシュの活用

関連する **Azure** サービス / テクノロジー

- Azure Cosmos DB - [Partition]
- SQL Database – [Hyperscale]
- Front Door / CDN

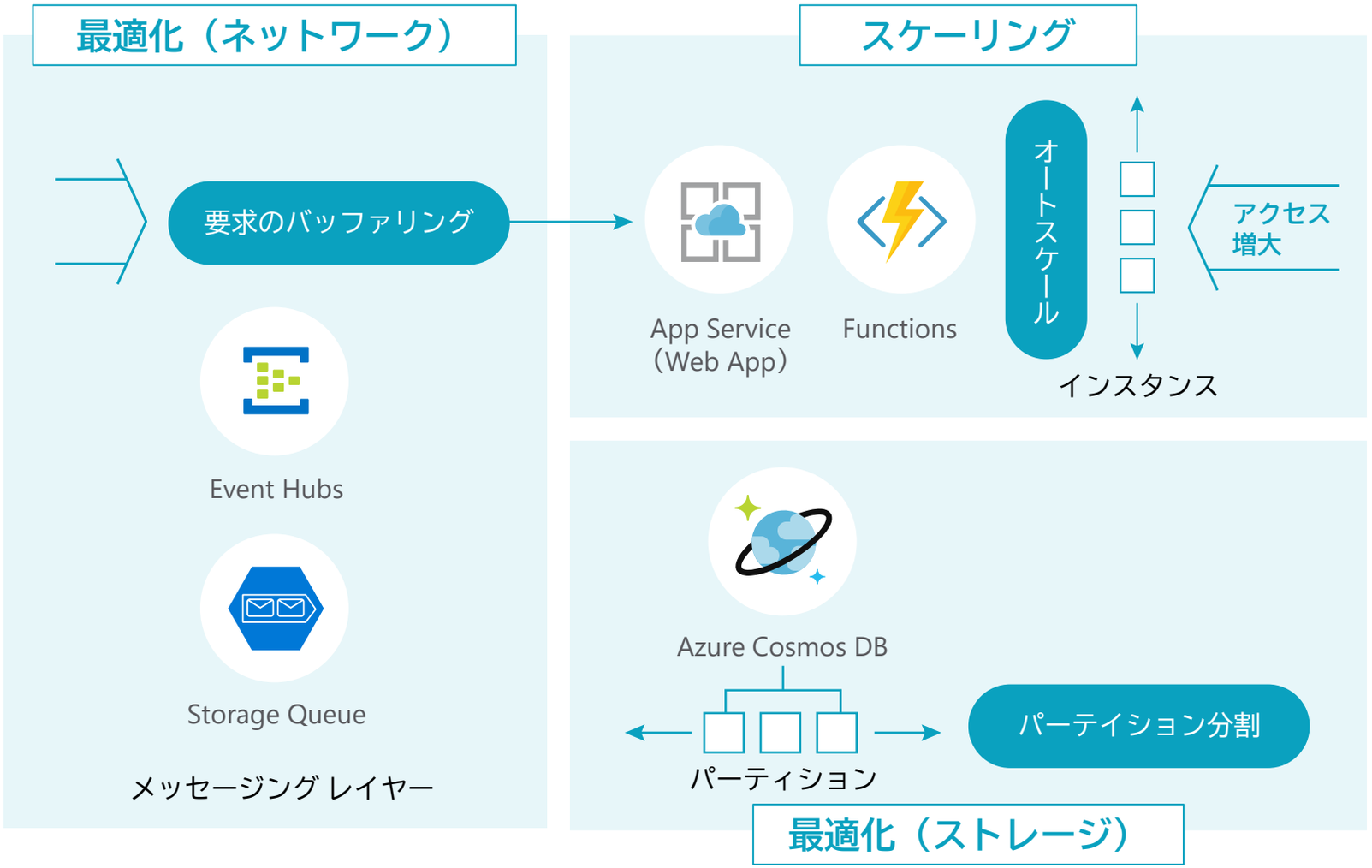

Azure Cosmos DB

Azure SQL Database


Azure Front Door Service


Content Delivery Network (CDN)

「パフォーマンス効率」観点での Azure 活用



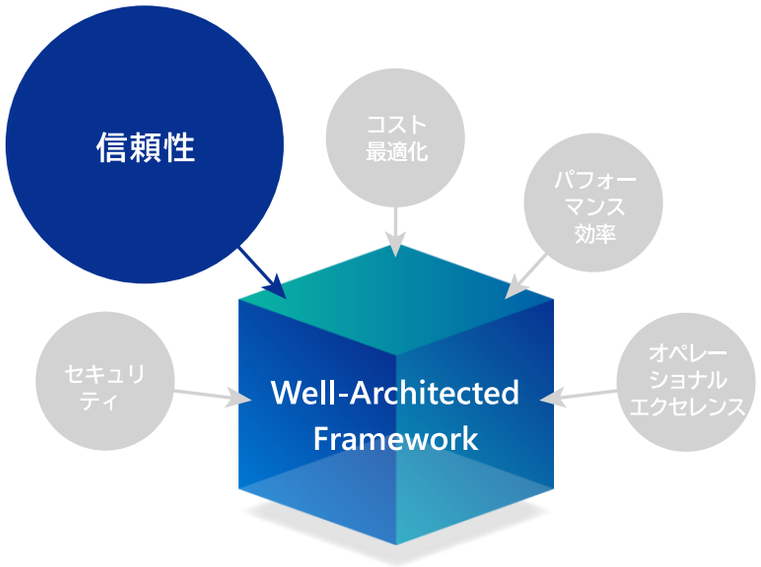
信頼性

RELIABILITY

信頼性

アーキテクチャの設計に高可用性と回復性を含めることで、
ダウンタイムやデータ損失の結果として生じる金銭的損失から企業を守る

- アップタイムを維持する
- データ損失や大規模災害からの復旧

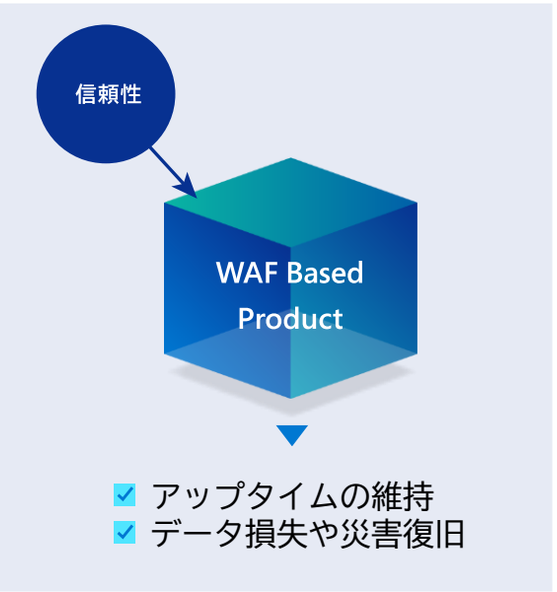


高可用性

- クラスタリング
- 負荷分散

復旧可用性

- RPO の短縮
- RTO の短縮



信頼性

高可用性アーキテクチャ

SLA を基準に適切なカバレッジがある箇所と改善が必要な箇所を特定する

クラスタリング

- 1 つのリソースで障害が発生した場合にサービスをフェールオーバーする

負荷分散

- 要求を分散させ、障害が発生したインスタンスにルーティングさせない

関連する **Azure** サービス / テクノロジー

- Azure Virtual Machine Scale Sets, Azure SQL Database –［自動フェールオーバー グループ］
- Front Door, Application Gateway

Azure SQL Database


Azure Front Door Service


Application Gateway

障害からの復旧が可能なアーキテクチャ

「RPO」「RTO」等の目標を達成するために、バックアップ、復元、レプリケーション、回復機能をアーキテクチャに取り入れる

RPO（目標復旧時点）

- データ損失が許容される最大継続時間

RTO（目標復旧時間）

- ダウンタイムが許容される最大継続時間

関連する **Azure** サービス / テクノロジー

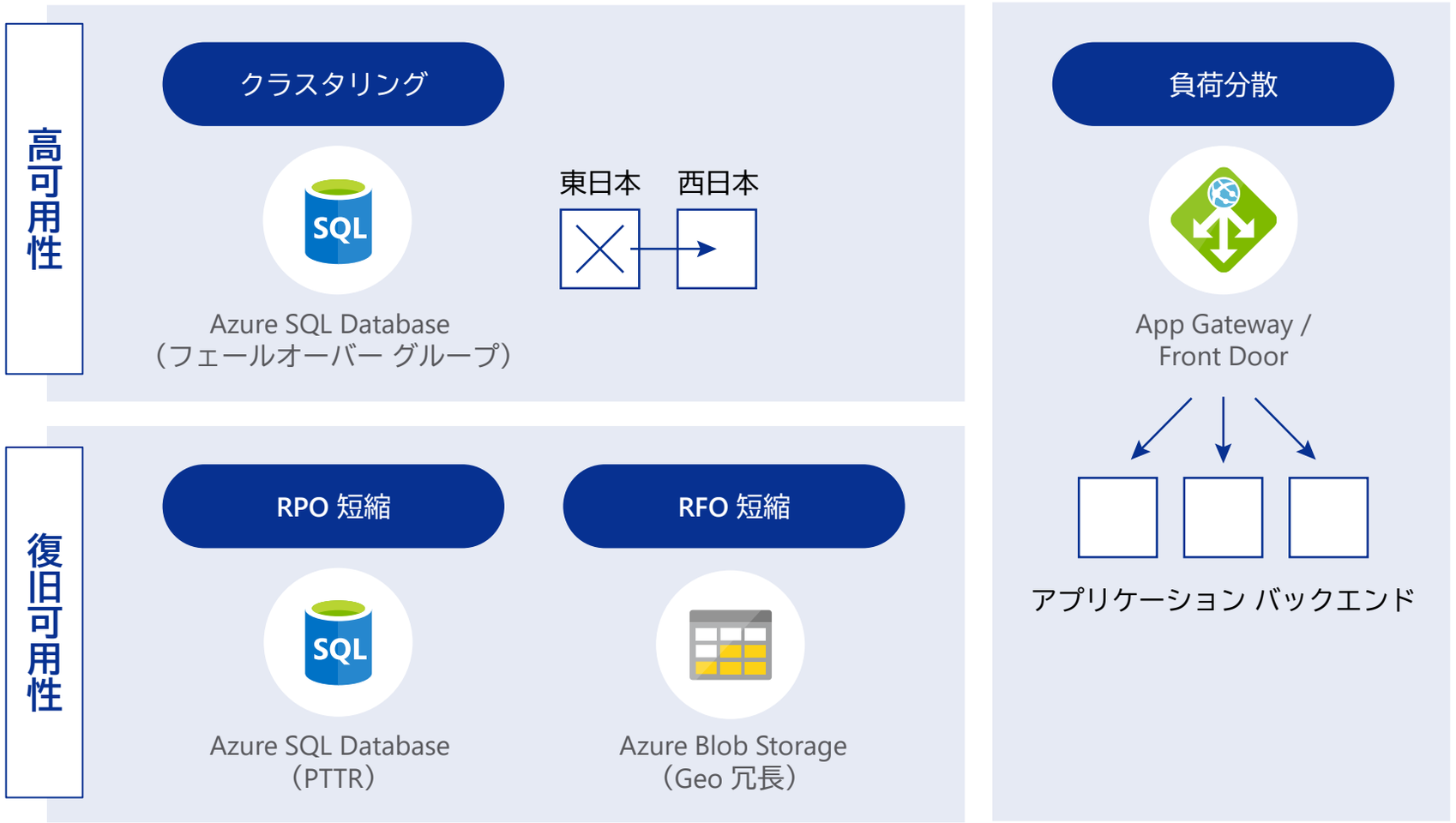
- Azure SQL Database – [ポイントインタイム リストア]
  Azure Storage – [geo 冗長ストレージ]

Azure SQL Database

Storage

信頼性

「信頼性」観点での Azure 活用



セキュリティ

SECURITY

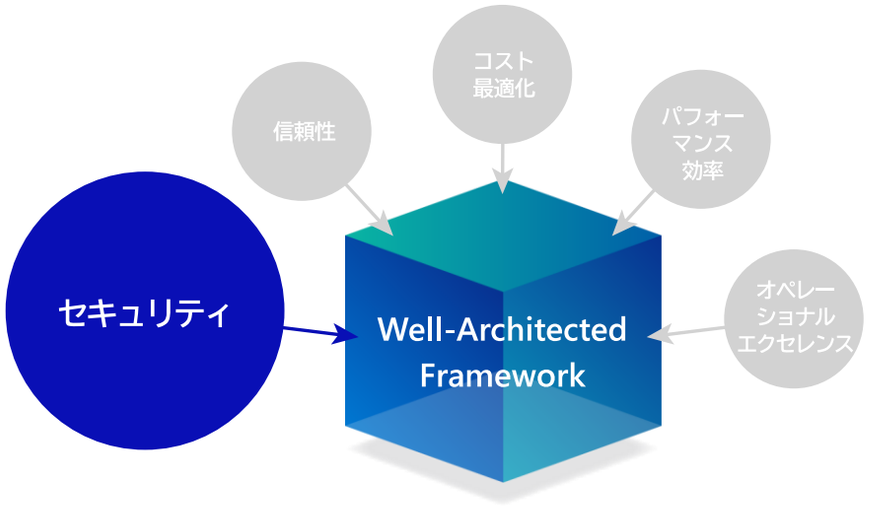
セキュリティ

セキュリティとは、組織が使用、保存、および送信するデータを保護すること

- 組織で保存または処理されるデータの保護
- データが存在するインフラストラクチャの保護

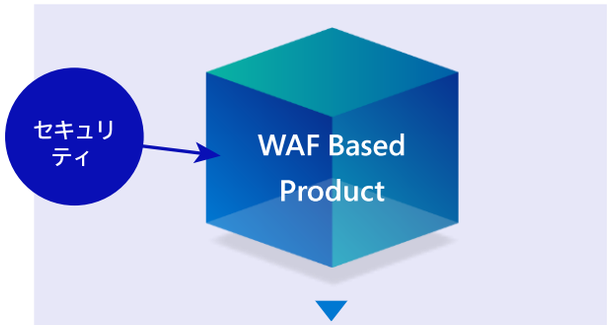
セキュリティのアプローチ

従来　境界防御だけを重視していた　▶　現在　侵害を前提としたゼロ トラスト モデル



多層防御

- データ
- アプリケーション
- コンピューティング
- ネットワーク
- 境界
- ID とアクセス管理
- 物理的セキュリティ



- ☑ 組織が保存するデータの保護
- ☑ データが存在するインフラの保護

ゼロ トラスト モデルを前提

多層防御

ゼロ トラスト モデルを前提とした階層型アプローチ
（ 各層で保護が提供され、1 つの層が破られても後続の層でも防御できる）

- **データ**: データベース、ストレージの保護・暗号化
- **アプリケーション**: 脆弱性対策、シークレットの安全な保管
- **コンピューティング**: エンドポイント保護、パッチの適用
- **ネットワーク**: 分断とアクセス制御、セキュリティで保護された接続
- **境界**: DDoS 保護、ファイアウォール
- **ID とアクセス**: アクセスを制御、シングルサインオンと多要素認証
- **物理的なセキュリティ**: ハードウェアへのアクセスを制御

ID とアクセス管理

クラウド中心のアーキテクチャでは、ゼロ トラスト モデルを前提に、
ID がセキュリティ保証の大部分の基礎を提供する

シングルサインオン（SSO）

- 単一の ID によりアプリケーションのセキュリティ モデルが簡素化される

多要素認証（MFA）

- 2 つ以上の要素を必須とする認証により、ID のセキュリティが強化される

関連する **Azure** サービス / テクノロジー

- Azure Active Directory
- Azure Active Directory B2C

Azure Active Directory

ネットワーク セキュリティ

ネットワーク内およびネットワーク外にあるリソースの通信を保護し、
サービスおよびシステム全体のネットワーク層での露出を制限する

インターネットからの保護

- 外部と接続しているリソースに対して、必要な場合にのみ通信を許可する

仮想ネットワークとネットワーク統合

- PaaS やオンプレミスを仮想ネットワークと統合しサービス間通信を制限

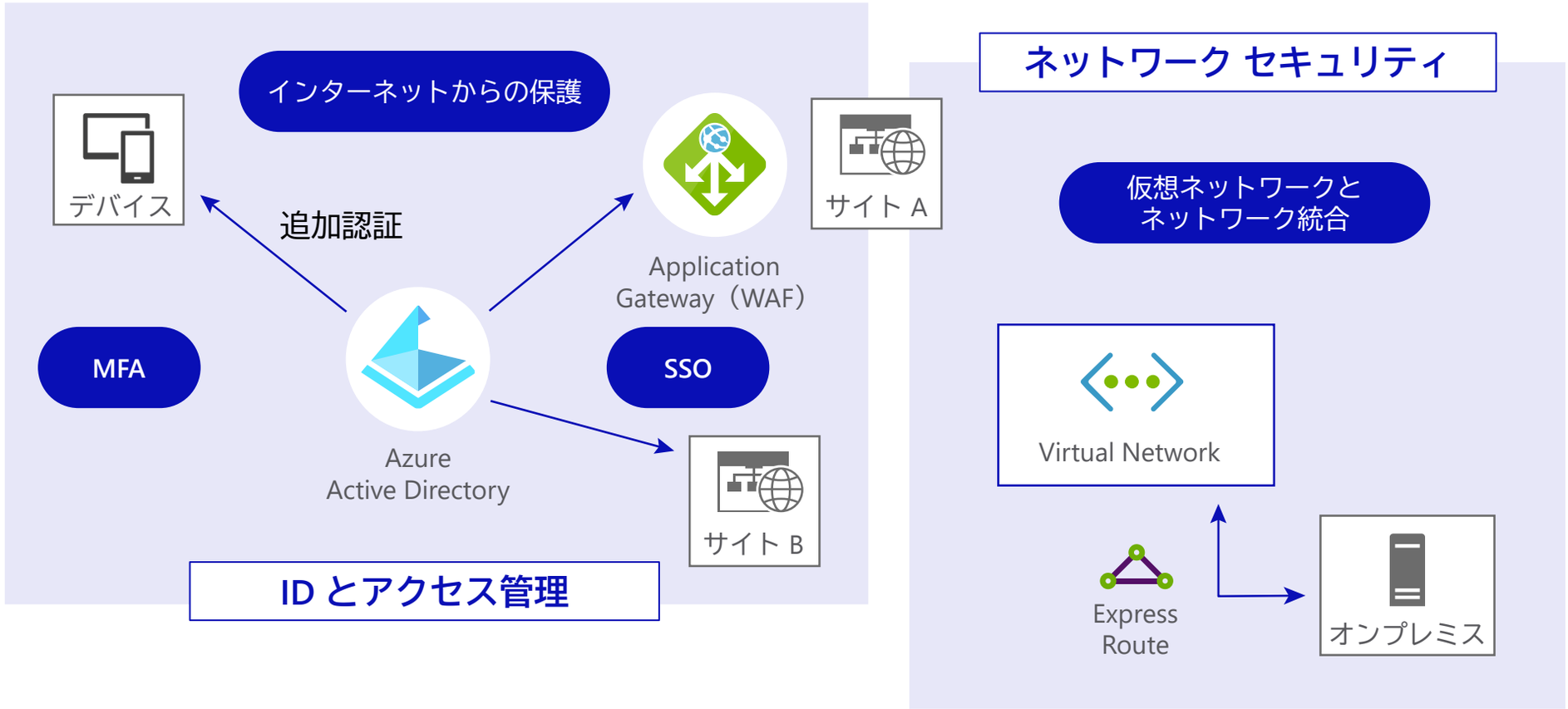
関連する **Azure** サービス / テクノロジー

- Virtual Network, Private Link,
- Azure Web アプリケーション ファイアウォール (WAF)

Virtual Network

Web App Firewall

「セキュリティ」観点での Azure 活用



コスト最適化

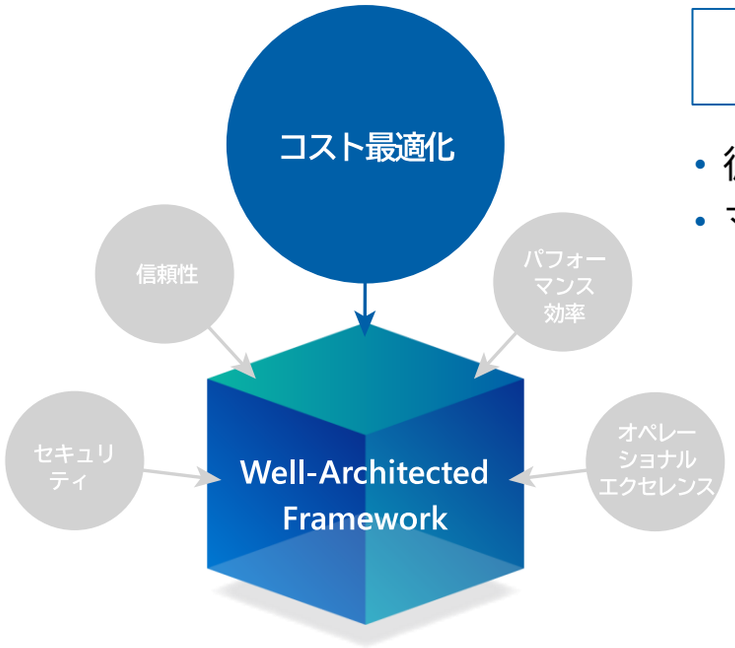
COST OPTIMIZATION

コスト最適化

組織が投じる金額が、効果が最大になるよう使用されていると保証すること

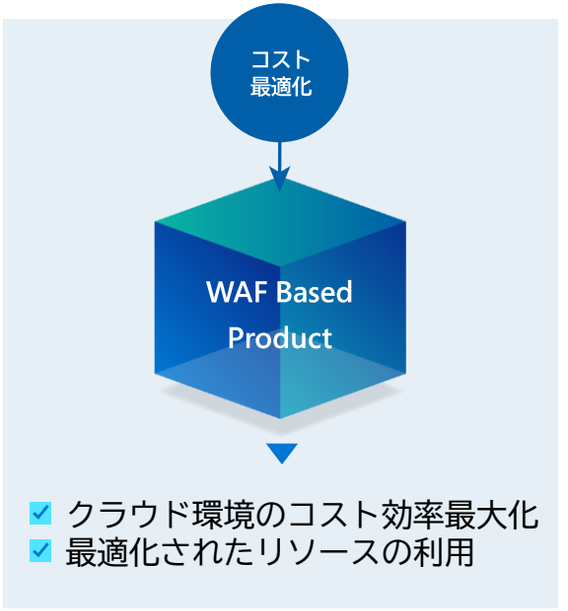
コスト最適化の原則

- アーキテクチャ設計の一部としてコスト最適化の観点を考慮する
- スケール、冗長性、コストに対する許容範囲（コスト制約）を特定する
- スケーラブルなコスト、従量課金制のメリットを生かせる構成
- 適切なリソース、適切なサイズでプロビジョニングする
- 継続的にコストの監視と最適化を行う



コスト最適化を考慮した設計

- 従量課金モデル
- マネージド サービス



コストの監視・分析

- クラウド支出の追跡
- コストのレビュー
- アラートへの対応

コスト最適化を考慮した設計

リソースの需要と技術要件は時間の経過と共に変化することを前提に設計する

従量課金モデルの利用

- トランザクション、CPU 時間、または実行時間に基づく課金
- アプリケーションが使用されていない時はリソースに料金がかからない

マネージド サービスの利用

- インフラの修正および管理が不要なため、運用コストが抑制できる

関連する **Azure** サービス / テクノロジー

- Azure Functions
- Azure SQL Database
- API Management

Functions

Azure SQL Database

API Management

コストの監視と分析

ビジネス要求、アプリケーションの変化にあわせてコストを最適化する

クラウド支出の追跡

- コストのかかる場所や使用率の低いリソースをデータで特定する

コストのレビューとアラートへの対応

- 支出に基づいてコストを可視化したりアラートする機能を利用する

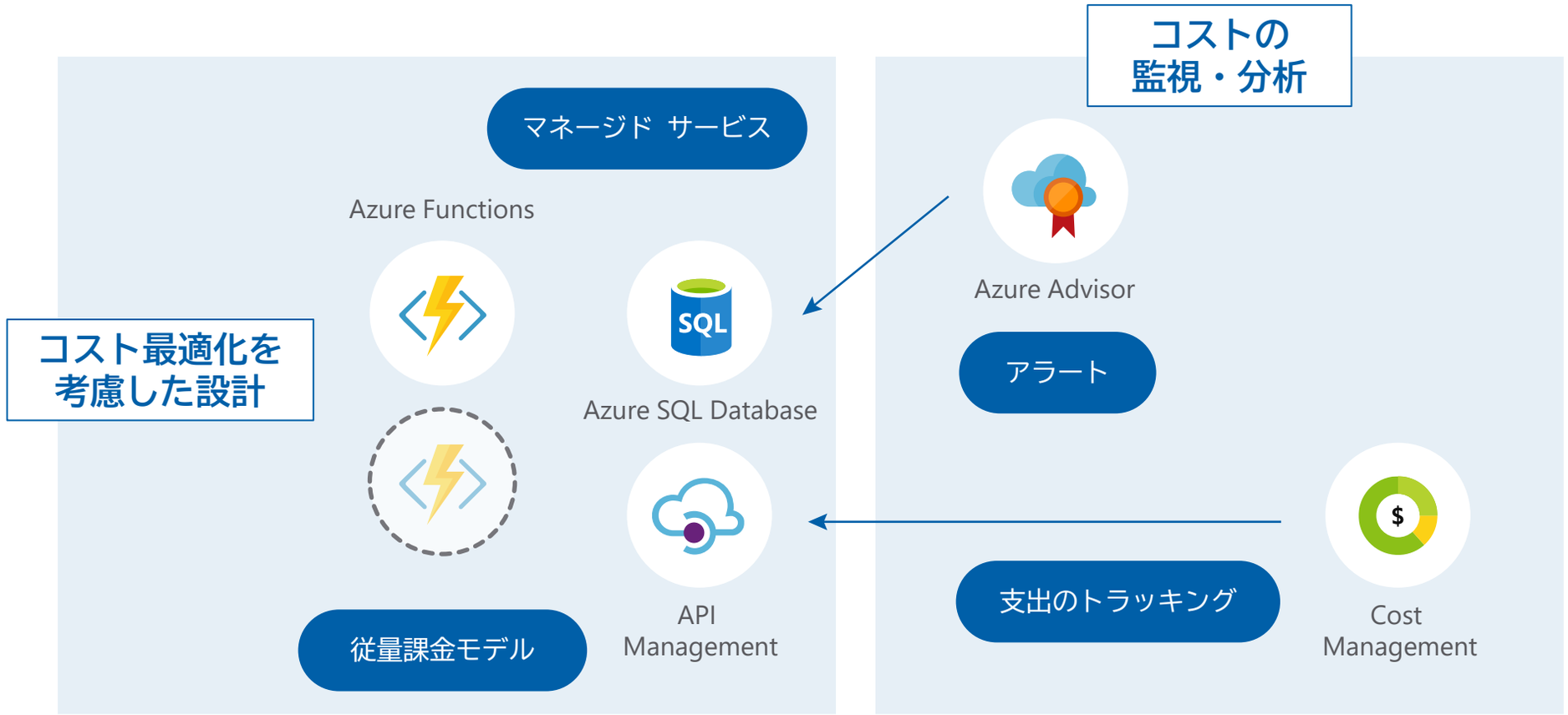
関連する **Azure** サービス / テクノロジー

- Azure Advisor
- Azure Cost Management

Azure Advisor

Cost Management

「コスト最適化」観点での Azure 活用



導入事例

富士フイルム 「IMAGE WORKS」

事例 URL: https://customers.microsoft.com/ja-jp/story/fujifilm-manufacturing-azure-ai-functions-jp-japan

従前の課題

- サービス利用クライアント(プロ野球球団)の増加により、データ処理量が増加していた
- 同期型／直列処理で実装していたため、処理速度が遅かった
- 大量のコンピューティングリソース利用によりコストが増大

- 画像解析処理パフォーマンスを向上させ、利用者のストレス軽減を達成し、サービス採用クライアントの増加にも耐えられるアーキテクチャにしたい

Well-Architected なアーキテクチャ変更により実現したこと (※)

コスト最適化

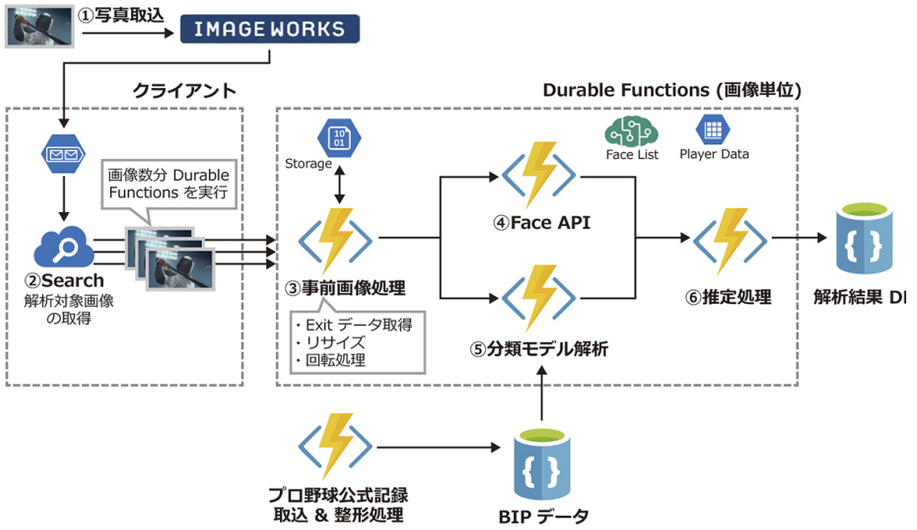
- AI エンジン部分に SaaS 型 API サービスを組み合わせることによる開発コスト削減
- オートスケール可能なマネージド サービス利用による運用コスト削減

パフォーマンス効率

- 水平スケール アーキテクチャによる処理効率アップ（1/20 に短縮）

信頼性

- グローバル分散の高可用性アーキテクチャ型データストアによるデータ保護



(※) 3 日間集中実装を行う ハッカソン プログラムよりアーキテクチャを一気に実装 https://www.itmedia.co.jp/business/special/mk200711/index.html

JFE エンジニアリング株式会社「Pla'cello」

事例 URL: https://customers.microsoft.com/ja-JP/story/817881-jfe-engineering-corporation-jp-japan

ハッカソン実施前の課題と達成すべき目標

- RDB (AWS RDS) にユーザーが直接アクセスすることによる性能／セキュリティの問題が発生していた
- 同期型／直列処理により ETL 処理に時間がかかる
- 手動運用による作業効率／拡張性の悪化

- 1 プラント当たり数千個のセンサー デバイスからのデータ分析の際のデータ量の制限や、ETL 処理に数時間かかる状況を打破したい

ハッカソン実施による主な達成事項

パフォーマンス効率

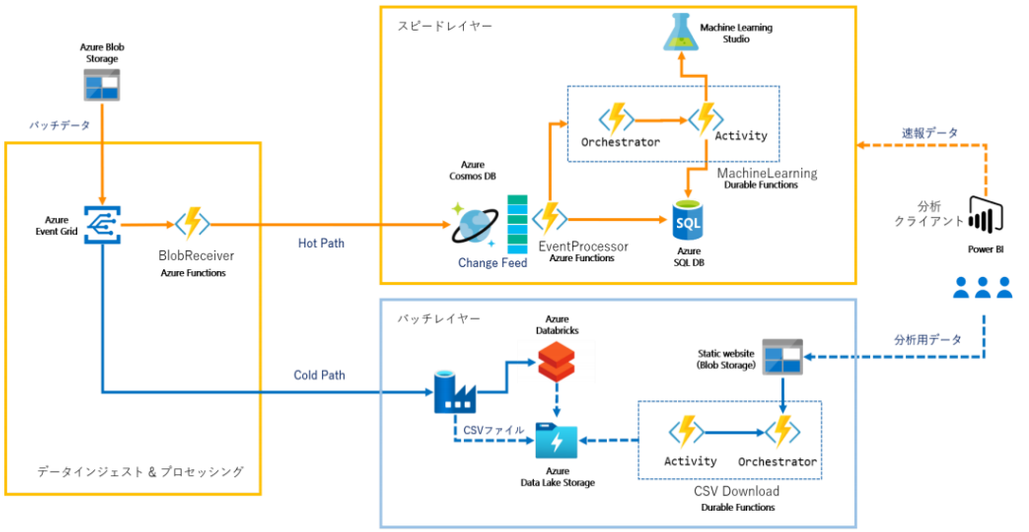
- ダウンロードの自由度・スピードが向上（約 10 倍以上）
- データ更新のリードタイムが短縮（2 時間 -> 10 分）

オペレーショナル エクセレンス

- Terraform を使った IaC により、プラント追加作業を短縮（3 日 -> 1 時間）

コスト最適化

- データ処理方法の見直しでストレージ コストが半分以下に



参考資料

タイトル	ダウンロードリンク
DX を実現するWell-Architected な開発環境・体制構築 (技術概要編)	https://aka.ms/WAF_TechSum_JA01
Well-Architected Framework 技術解説	https://aka.ms/WAF_TechOverview01
DX を実現するWell-Architected な開発環境・体制構築 (ビジネス編 – Word 詳細編)	https://aka.ms/WAF_BIZDetail_JADoc
DX を実現するWell-Architected な開発環境・体制構築 (ビジネス編 – Word 概要編)	https://aka.ms/WAF_BIZSum_JADoc
DX を実現するWell-Architected な開発環境・体制構築 (ビジネス編 – PPT 概要編)	https://aka.ms/WAF_BIZSum_JAPPT
Microsoft Azure Well-Architected Framework	https://docs.microsoft.com/ja-jp/azure/architecture/framework/
Microsoft Azure Well-Architected Framework の概要 （MS Learn）	https://docs.microsoft.com/ja-jp/learn/modules/azure-well-architected-introduction/
Azure 向けの Microsoft Cloud 導入フレームワーク (Cloud Adoption Framework)	https://docs.microsoft.com/ja-jp/azure/cloud-adoption-framework/
日本企業向けクラウド導入ガイドライン・ベストプラクティスガイド (2020 年 5 月版)	https://info.microsoft.com/JA-AzureApp-CNTNT-FY20-06Jun-08-CloudServiceInfrastructureUtilizationRegulations-SRGCM3554_01Registration-ForminBody.html
BizDevOps ラウンドテーブルホワイトペーパー	https://info.microsoft.com/JA-AzureApp-CNTNT-FY20-04Apr-29-LinkingbusinessandIT-SRGCM3421_01Registration-ForminBody.html

Microsoft Azure

Thank you.