Google Cloud

GKE をバックエンドとした時に Apigee はどう使えるのか

Table of Contents

01

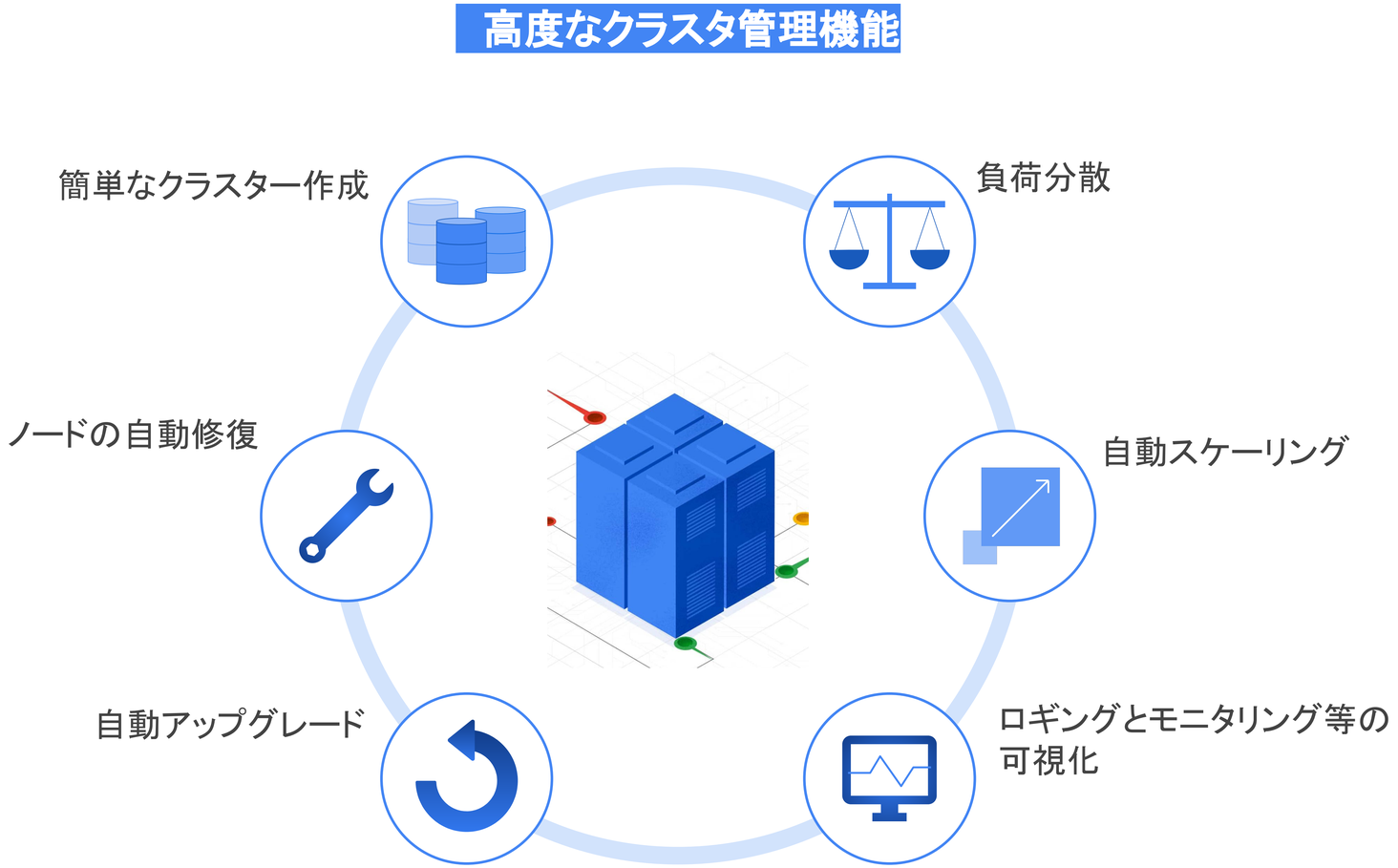
Google Kubernetes Engine 概要

Google Kubernetes Engine = Google のマネージド Kubernetes

Google Cloud により

完全に管理される**マネージド** Kubernetes

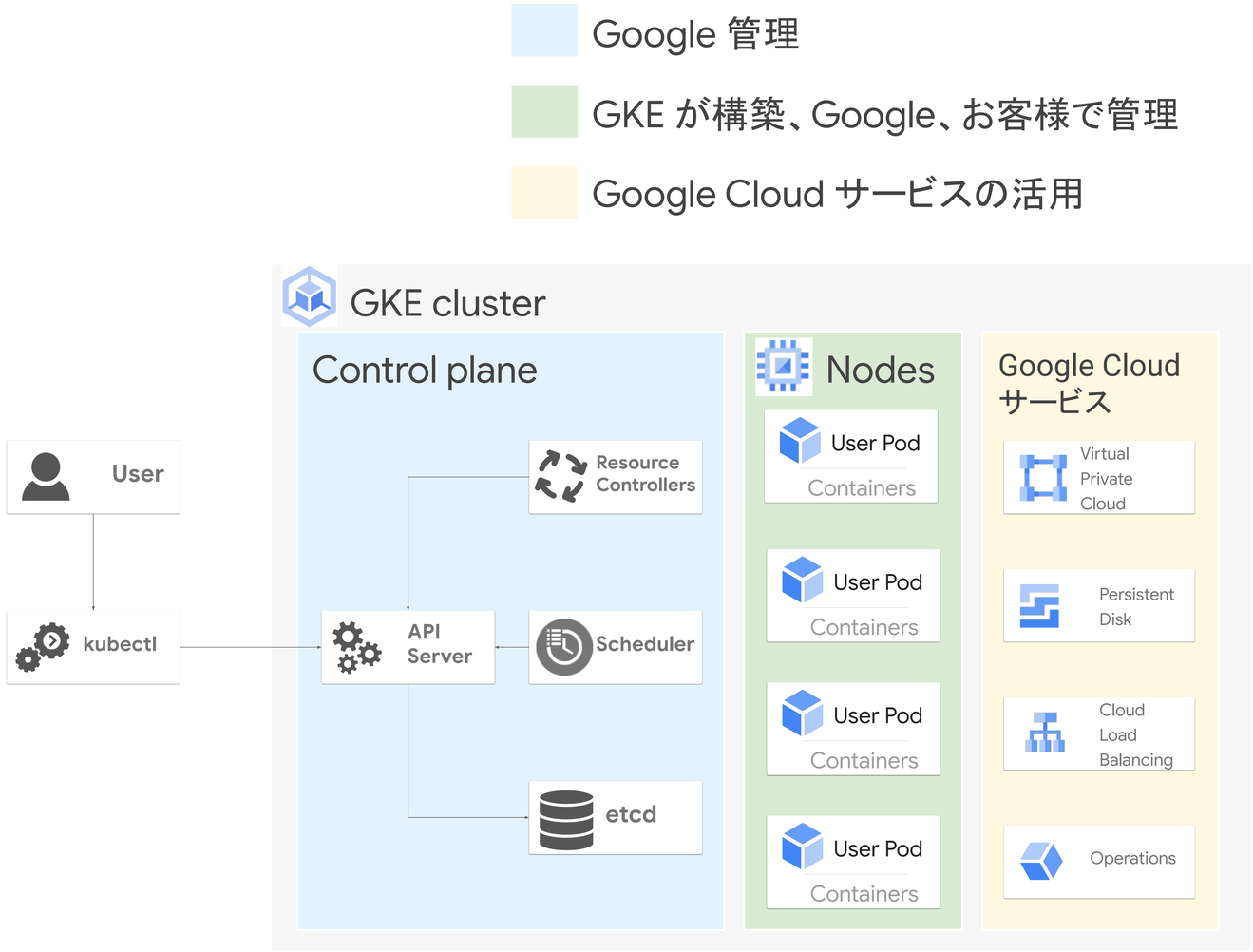
- 高度な**クラスタ管理**機能（**参照 ➔**）
- クラスタの可用性が確保され、**常に最新の状態**に保つ
- **コスト パフォーマンス**の高いサービス
- オンプレミス クラスタまたは他クラウド プロバイダとの**柔軟な相互運用**が可能
- **オープンソース**に準拠



Google Kubernetes Engine - Standard モード

Google のマネージド Kubernetes 環境

- **自動**でスケーリング、
  アップグレード、ノード修復
- Kubernetes 運用の**ベスト プラクティス**をマネージド サービスとして提供
- **セキュリティとコンプライアンス**
  - 業界をリードするセキュリティ機能群
  - HIPAA や PCI DSS など各種コンプライアンスに準拠



Google Kubernetes Engine - Autopilot モード

特長

- Control plane に加え

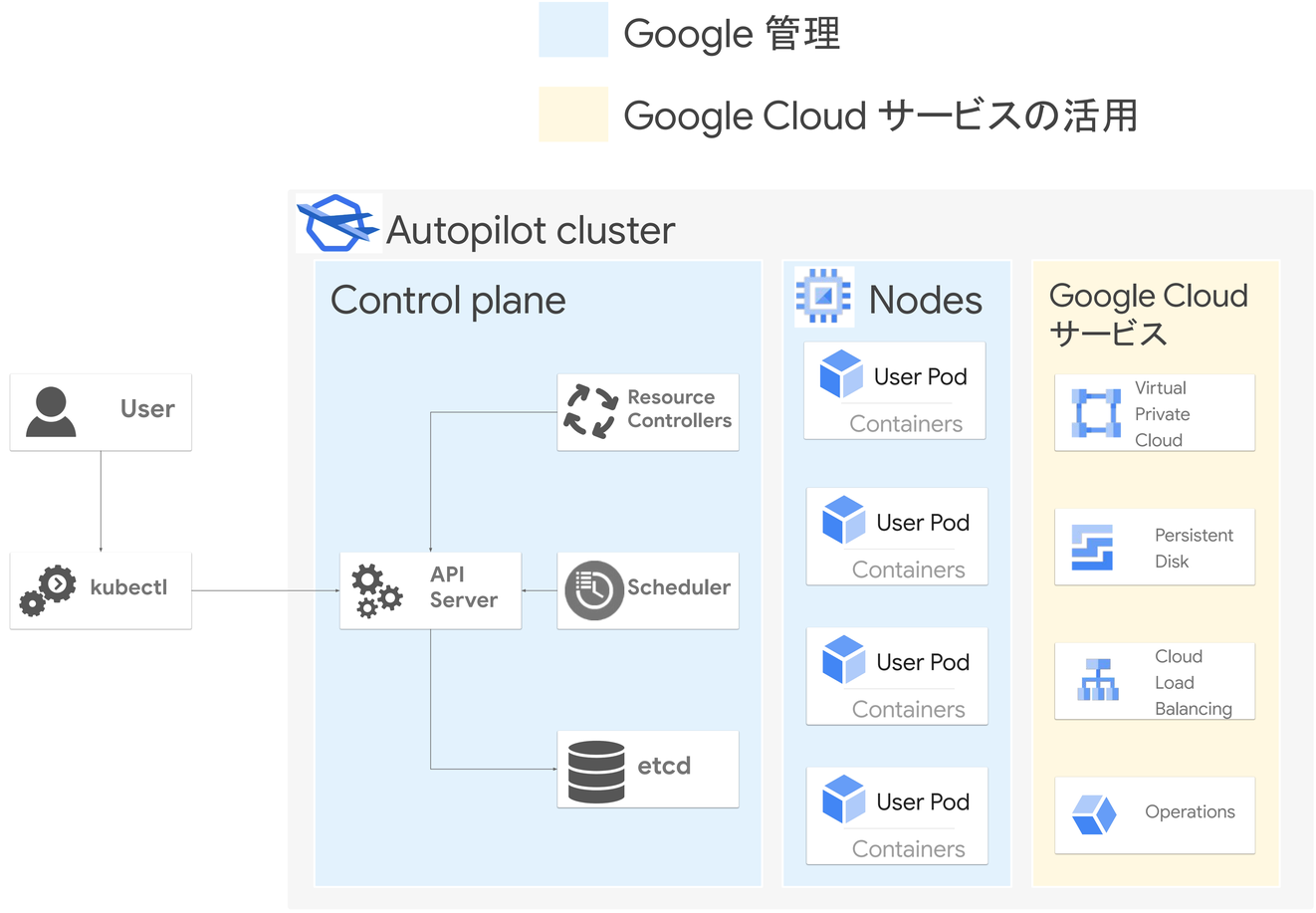
  Node も Google マネージド

- 本番ワークロードに適した

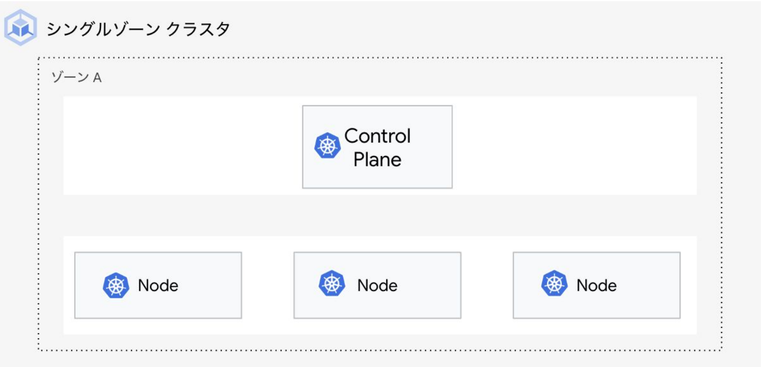
  ベスト プラクティスが適用済み

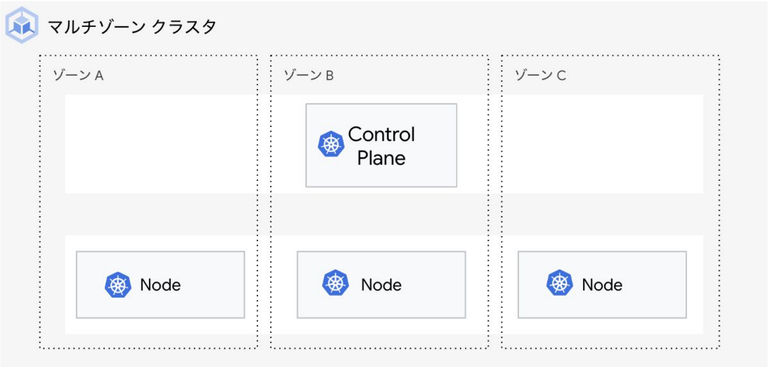
  - セキュリティ
  - ワークロード
  - その他設定

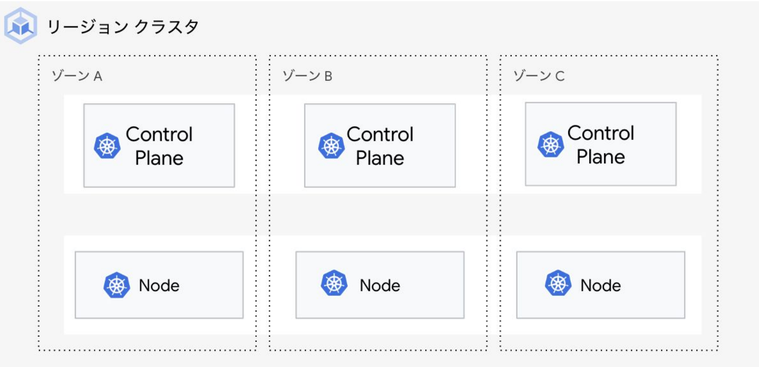
- Workload（Pod）ドリブン
  - Pod 単位での課金、Pod への SLA



リージョン内におけるクラスタの高可用性







クラスタ タイプ	**シングルゾーン**	**マルチゾーン**	**リージョン**
Control Plane	シングルゾーン	シングルゾーン	**マルチゾーン**
Nodes	シングルゾーン	マルチゾーン	**マルチゾーン**
SLA (Control Plane)	99.5%	99.5%	**99.95%**

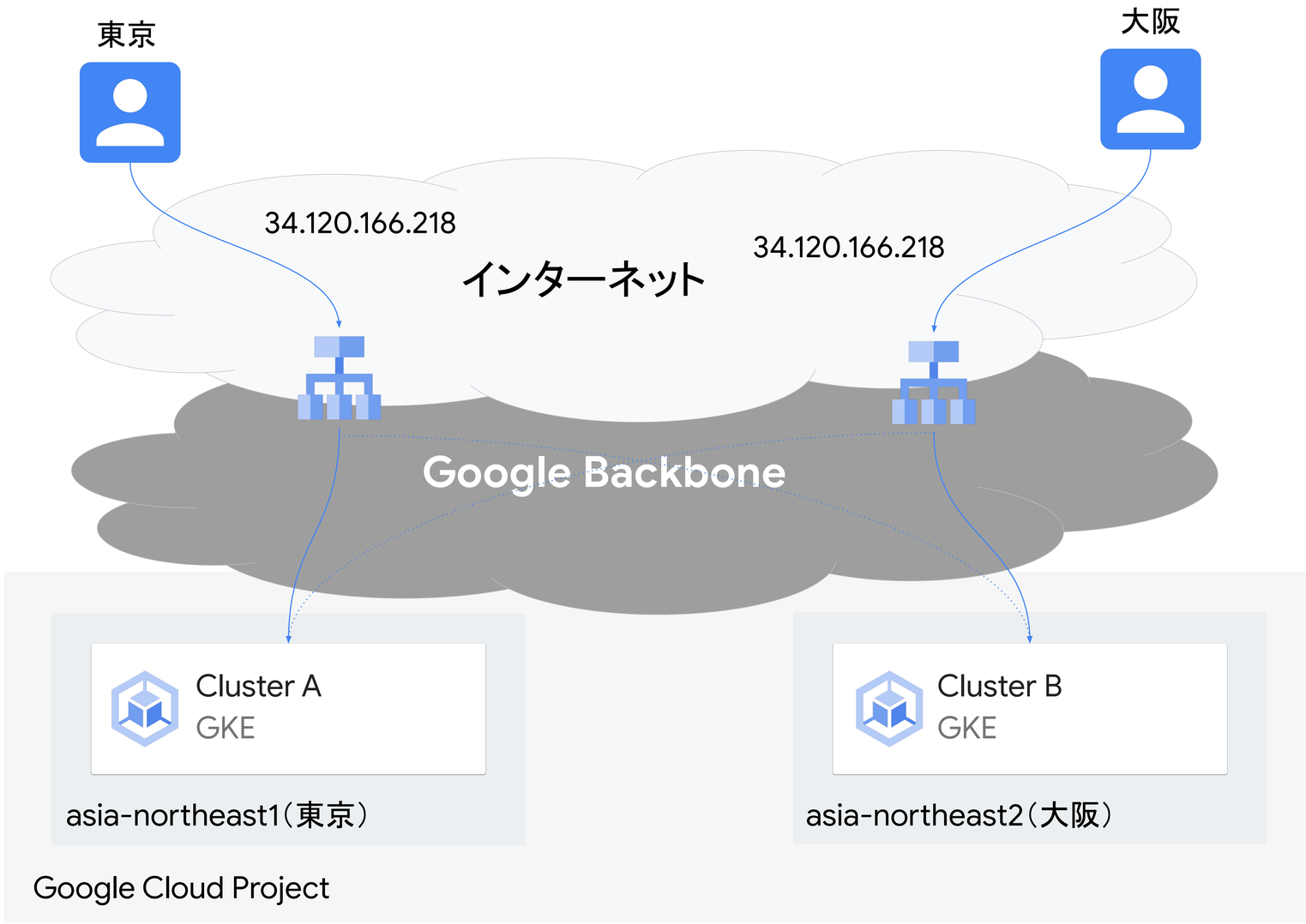
02

マルチクラスタ構成による可用性の向上

マルチクラスタ構成とは

複数の GKE クラスタ間で
負荷分散を行う仕組み

Active-Active 構成で
リージョン障害にも対応



Google Cloud のグローバル or リージョナル リソース

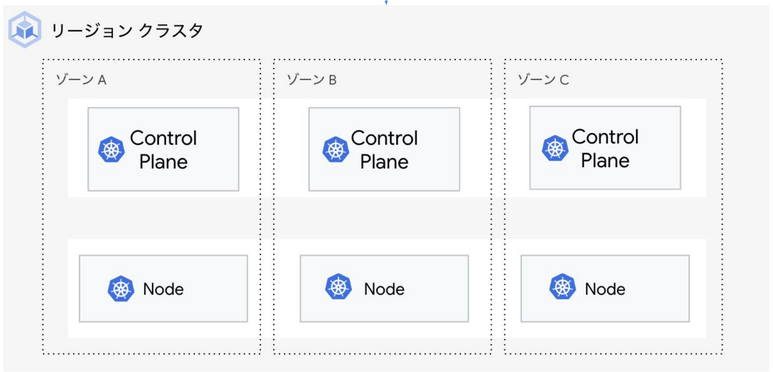
グローバル
リソース

グローバル外部ロードバランサ

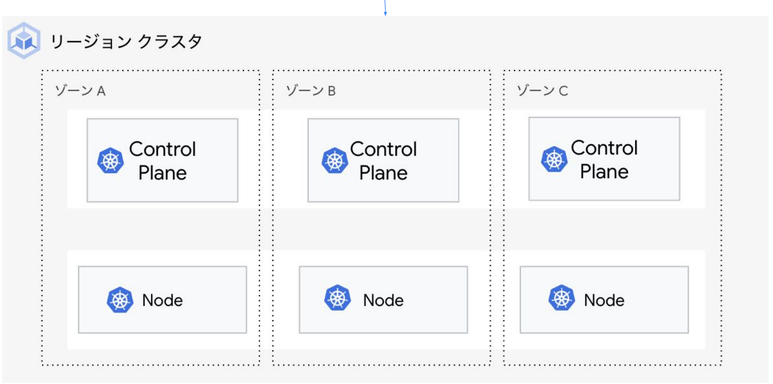
Cloud Load
Balancing

リージョナル
リソース

東京

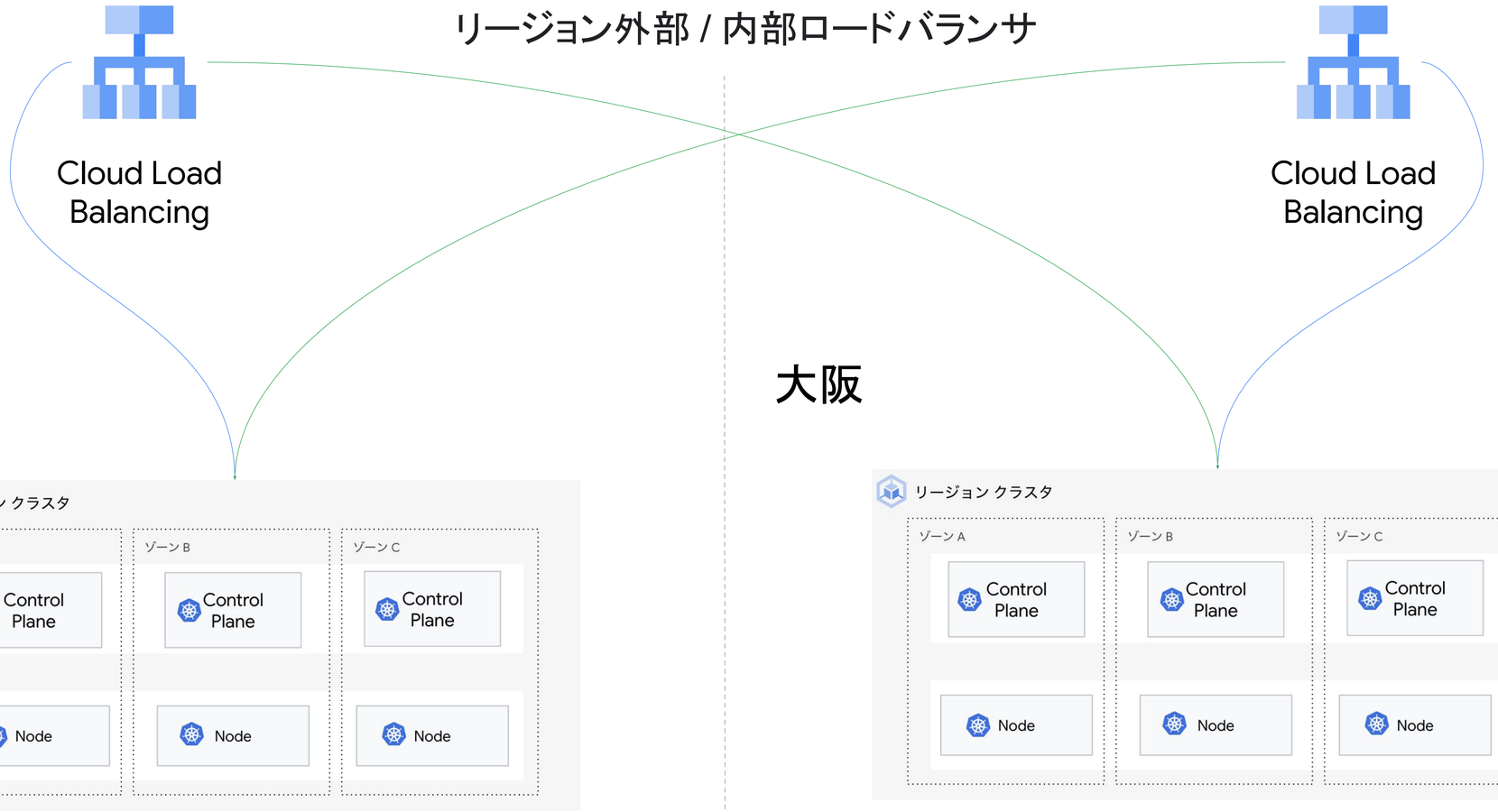


大阪



Google Cloud のグローバル or リージョナル リソース

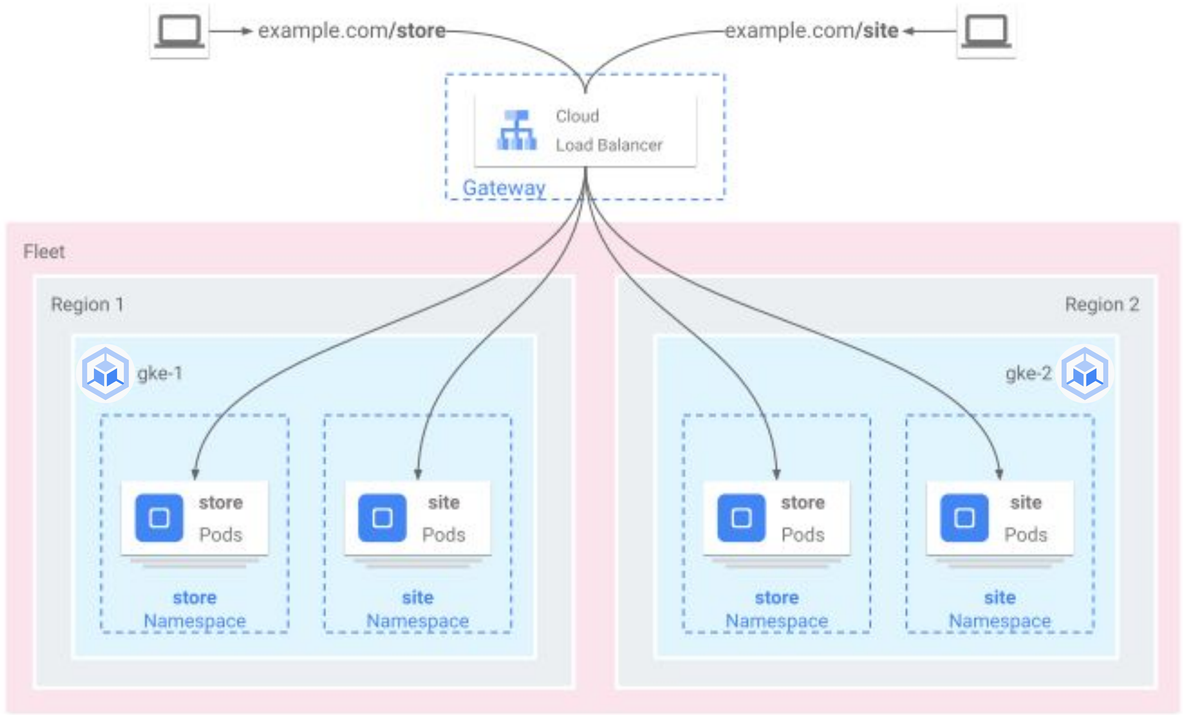
リージョナル
リソース



クラスタ間で外部 HTTPトラフィックの負荷分散をする方法

Multi-cluster Ingress（GA）または
Multi-cluster Gateway（Preview）を適用

- クラスタ間、マルチ リージョン間での HTTP/HTTPS ロードバランシングが可能
- バックエンドの状態（Health）をみつつ、最寄り（Latency）のクラスタに転送される



クラスタ間で Service Discovery をする方法

Multi-cluster Service や Anthos Service Mesh を適用

- Multi-cluster Service（Cloud DNS / Traffic Director を利用）
  - Multi-cluster Gateway 利用時は必須で利用
  - ServiceImport / ServiceExport リソースを構成
- Anthos Service Mesh（Istio ベースの Managed Service Mesh）
  - 特定サービスだけでなく、クラスタ内サービス全てをメッシュ化
  - MCS では実現できない、高度なトラフィック管理を実現可能（Weight Based Routing など）

03

Apigee を利用した場合の構成

どのようなユースケースで Apigee を使うか

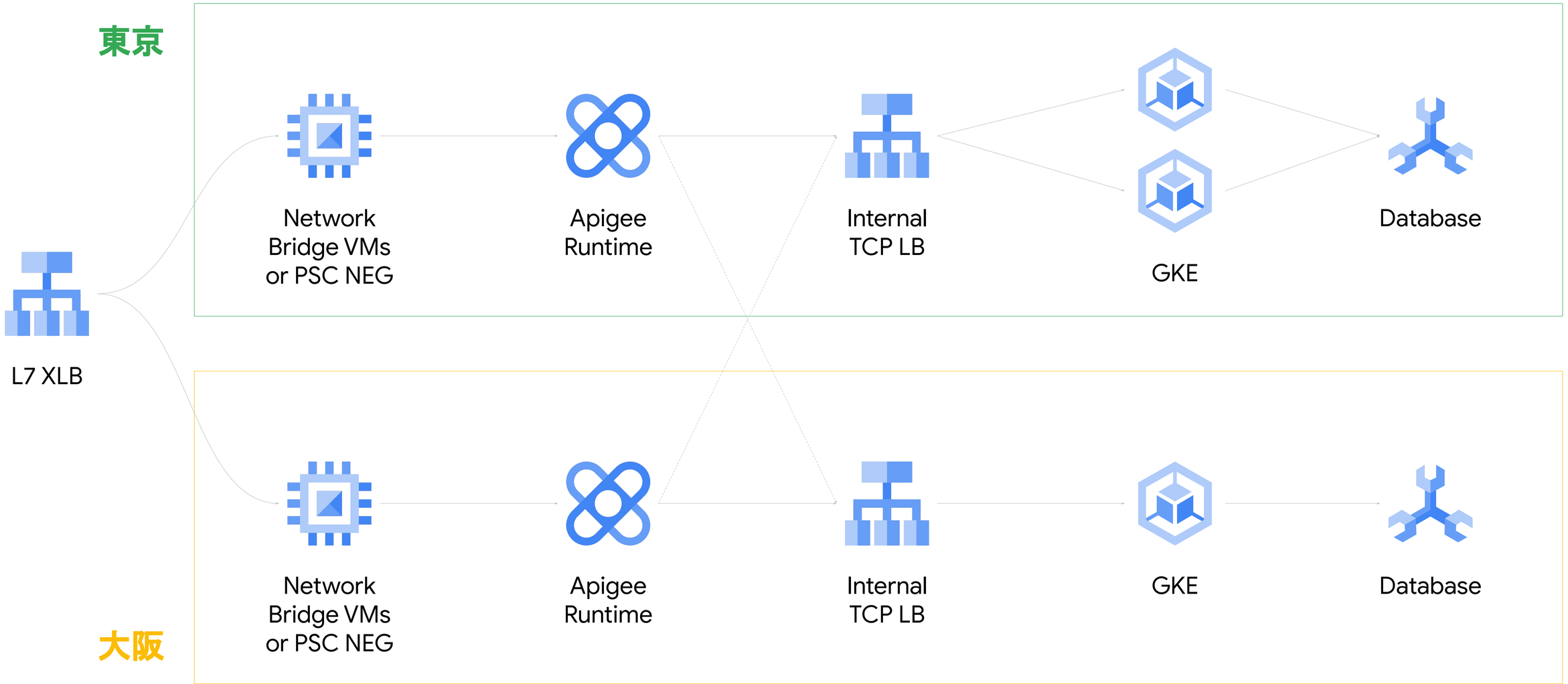
- リバース プロキシとして利用
  - 後段にあるバックエンド基盤の切り替え、リダイレクトなど
- クラスタの前段のレイヤで認証、認可を実施する
  - API キーや、OAuth 2.0 認証など、バックエンド転送前に実施するレイヤを挟む
- API を外部に公開して利用する
  - 複数のクライアントに提供するための制御項目（Rate-limiting など）を設定

etc...

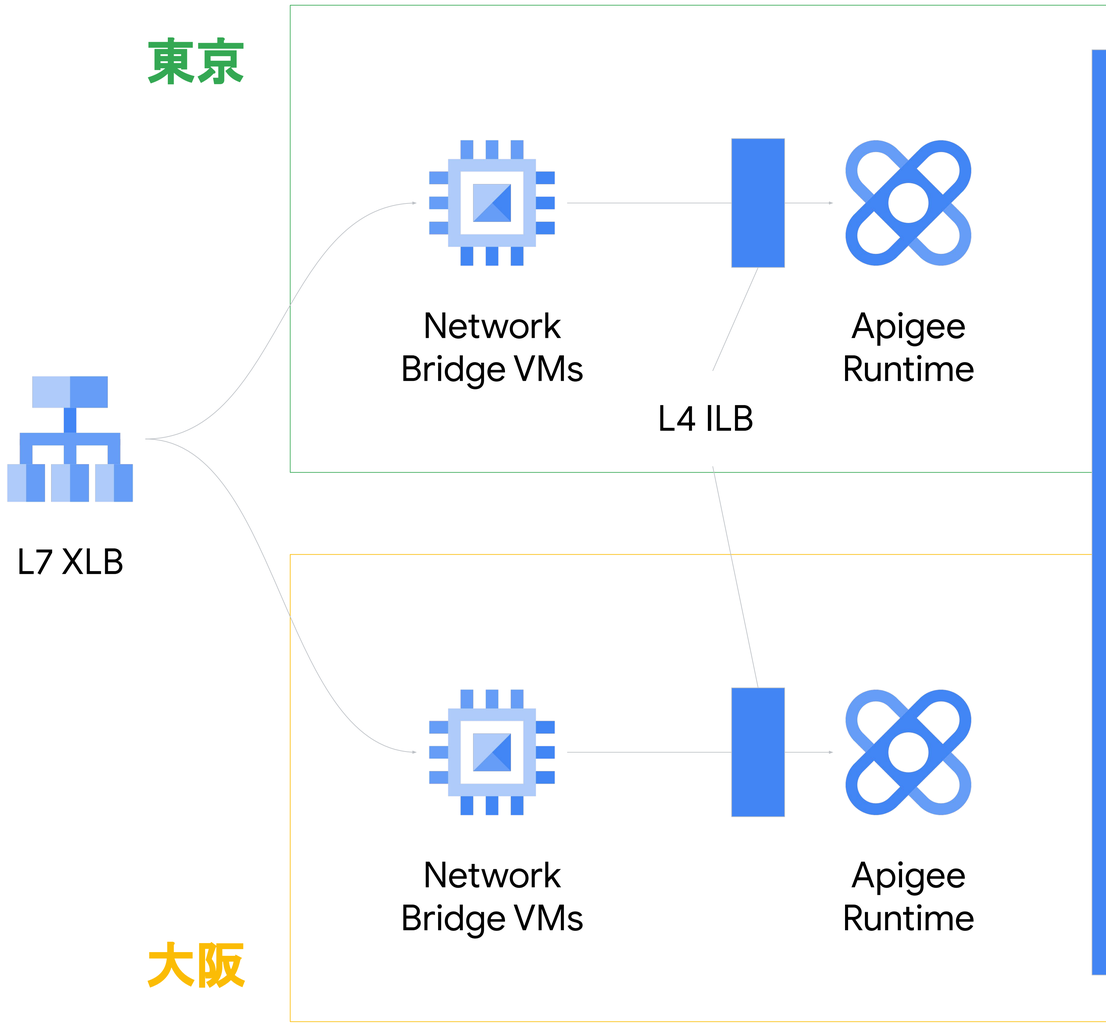
Apigee とバックエンド、共に高可用構成にしたい

- Apigee インスタンスを複数リージョンに構築
- GKE バックエンドを内部 LB を利用して 2 つそれぞれのリージョンで構築する
- エンドユーザーのトラフィックを、Apigee が受信してバックエンドへ流す
  - Apigee のプロキシ設定で、複数の内部 LB を設定することで高可用性を実現

アーキテクチャ例


東京
Network
Bridge VMs
or PSC NEG
Apigee
Runtime
Internal
TCP LB
GKE
Database
L7 XLB
大阪
Network
Bridge VMs
or PSC NEG
Apigee
Runtime
Internal
TCP LB
GKE
Database

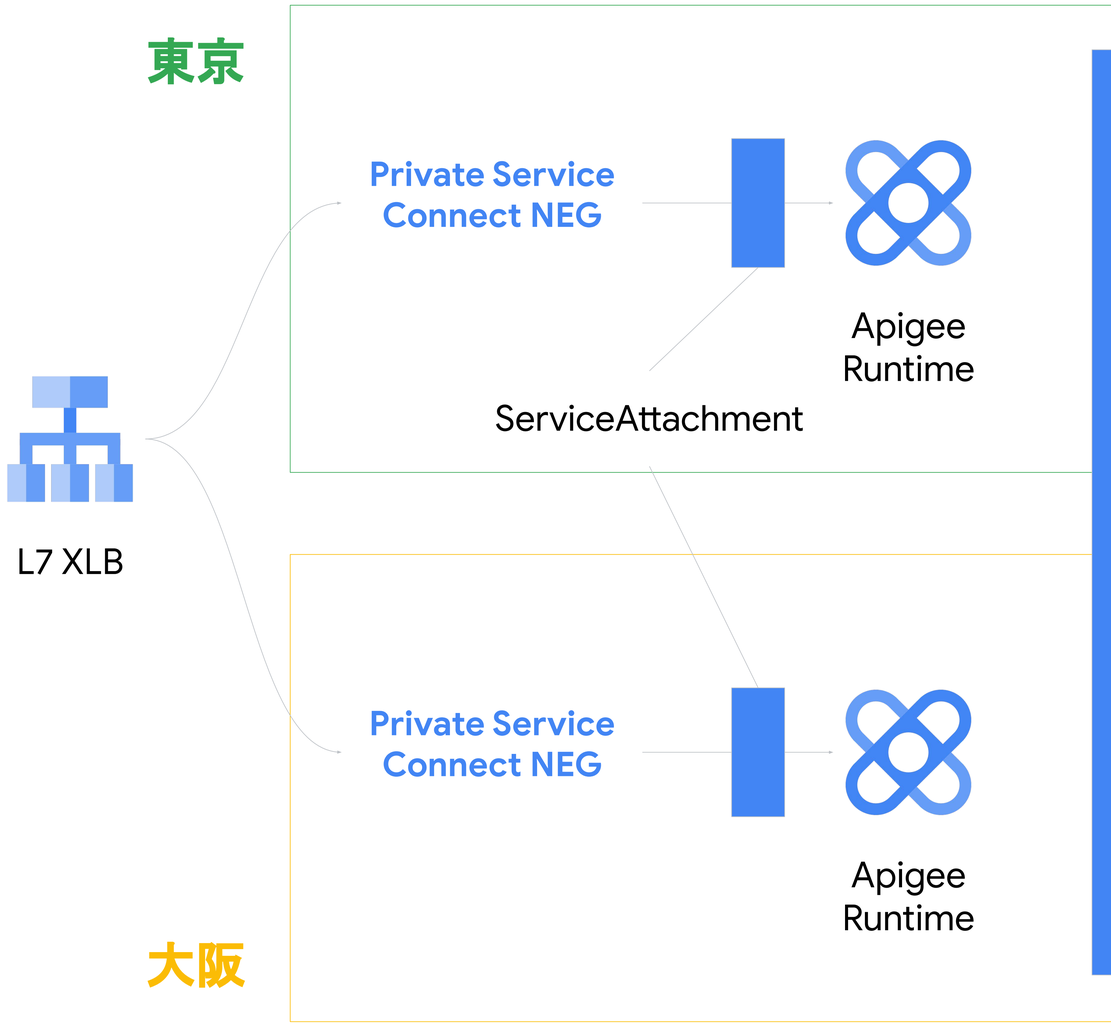
リクエストを処理する流れ via Network Bridge VMs



1. XLB が VM にリクエストを送信
2. VM は、VPC と Apigee VPC 間のブリッジとして機能する
3. L4 ILB 経由で Apigee がリクエストを受信すると、API プロキシ リクエストが処理される
4. Apigee によってリクエストがバックエンド サービスに送信され、レスポンスがクライアントに返される

リクエストを処理する流れ via Private Service Connect (PSC) NEG

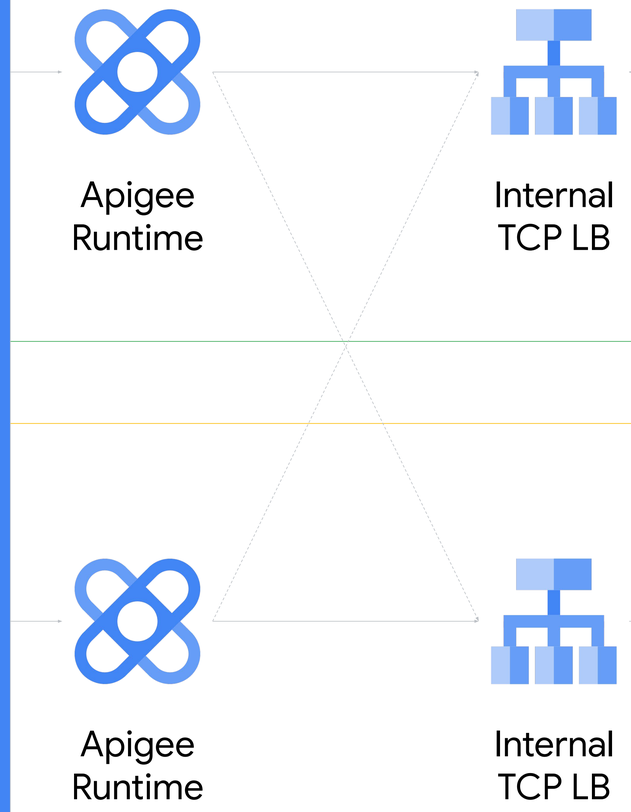
New



1. XLB が VM にリクエストを送信
2. Apigee インスタンスの ServiceAttachment を参照する Private Service Connect NEG を通る
3. Apigee にリクエストを送信すると、API プロキシ リクエストが処理される
4. Apigee によってリクエストがバックエンドサービスに送信され、レスポンスがクライアントに返される

バックエンド サーバー間の負荷分散

Apigee のバックエンド設定で、
複数のターゲットを指定可能



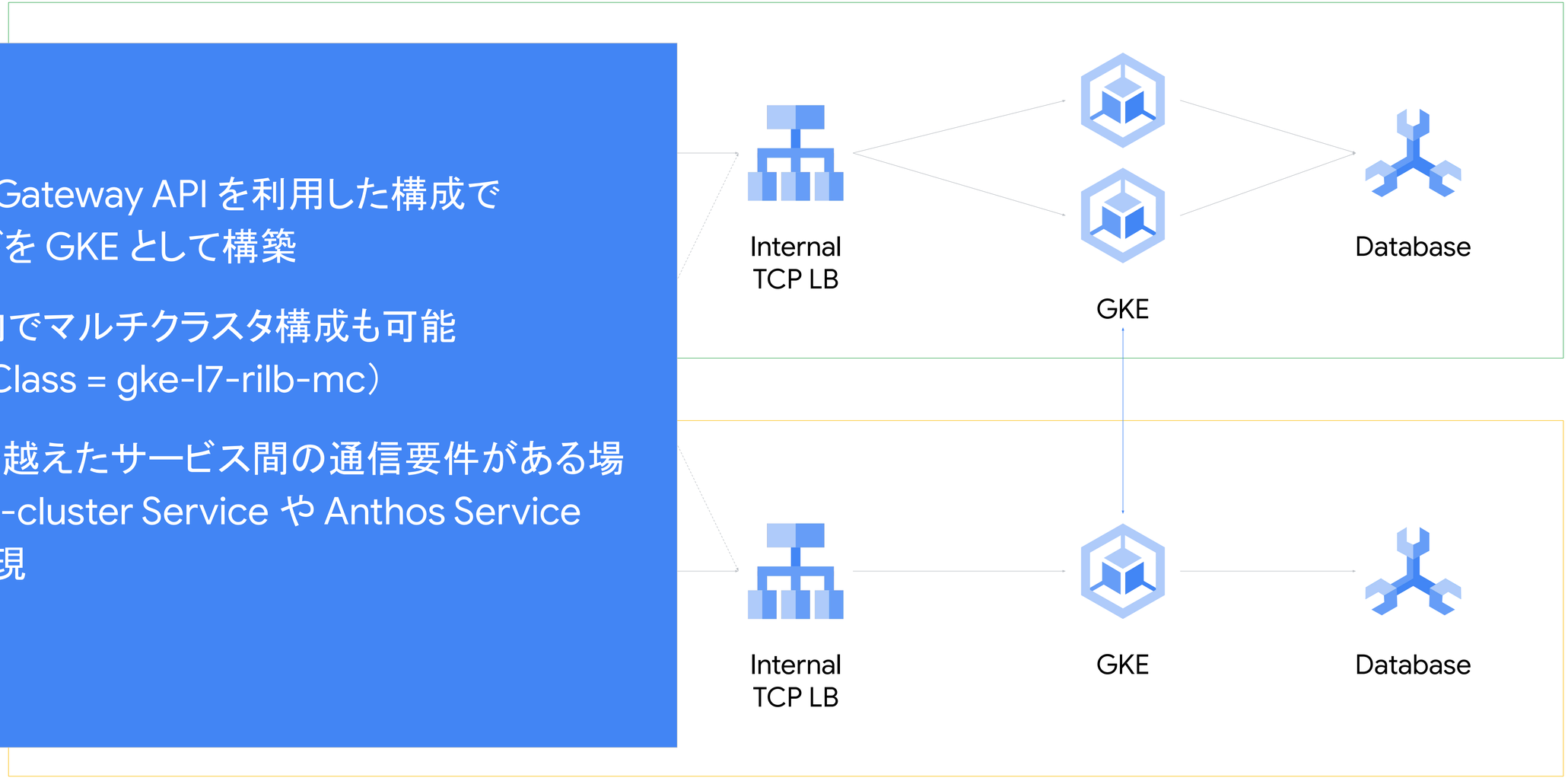

```
<TargetEndpoint name="default">
 <HTTPTargetConnection>
  <LoadBalancer>
   <Server name="target1" />
   <Server name="target2" />
  </LoadBalancer>
  <Path>/test</Path>
 </HTTPTargetConnection>
</TargetEndpoint>
```

バックエンド サーバーの構築を GKE で

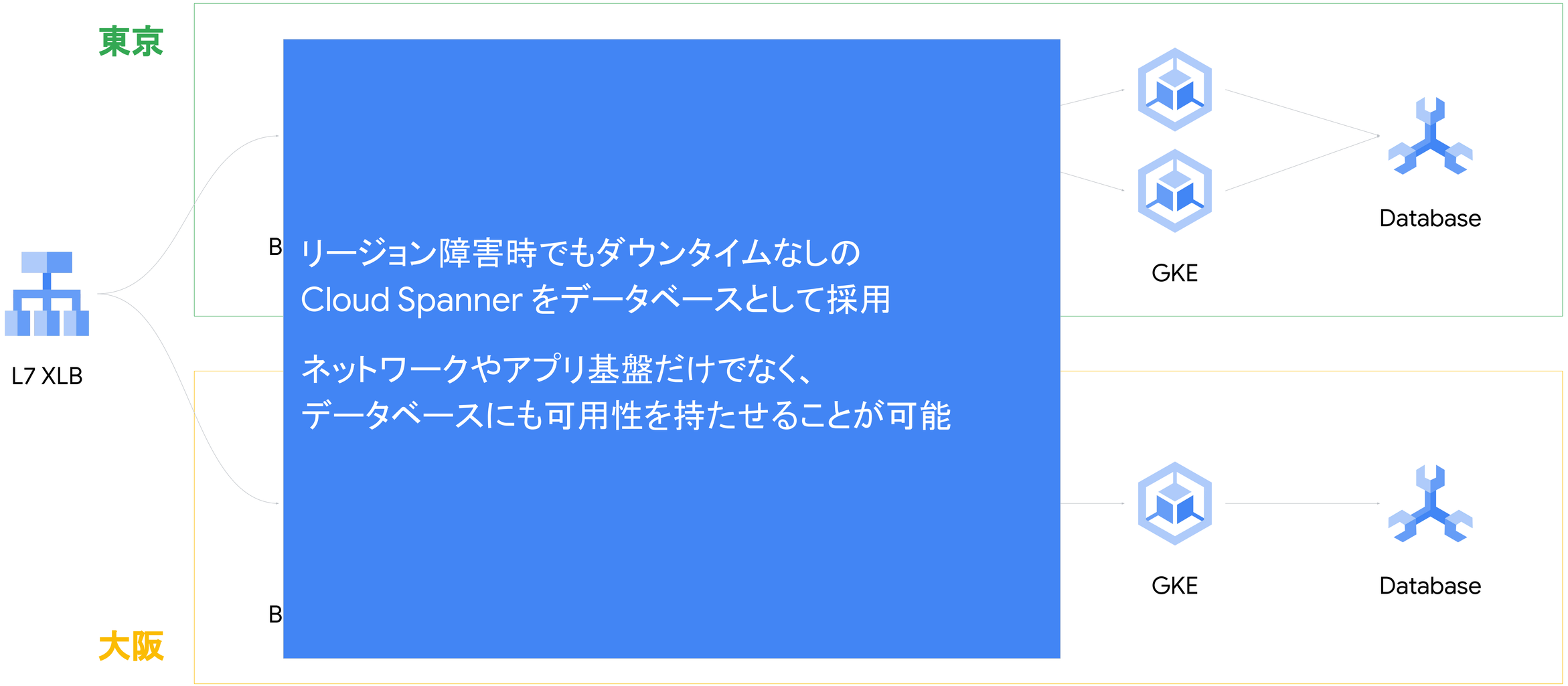
Ingress or Gateway API を利用した構成で
バックエンドを GKE として構築

リージョン内でマルチクラスタ構成も可能
（GatewayClass = gke-l7-rilb-mc）

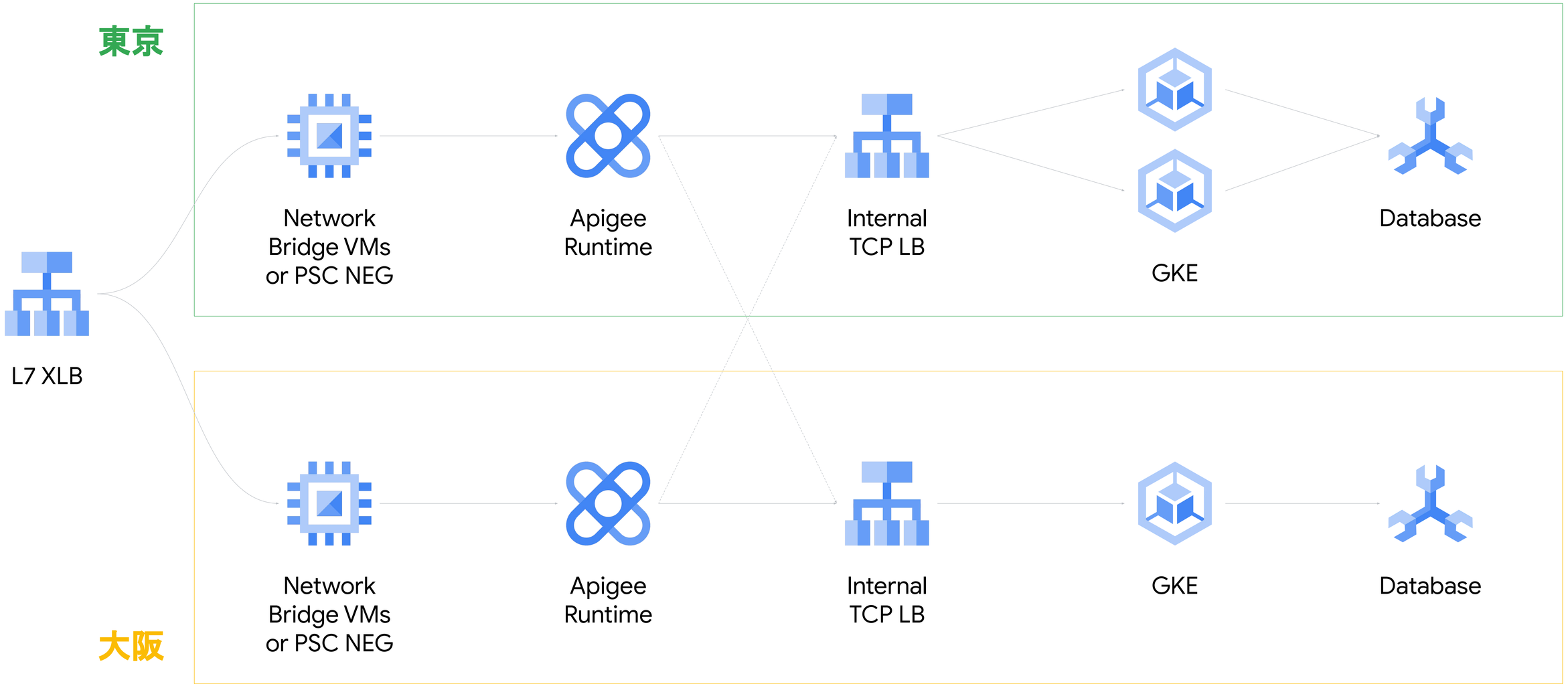
リージョンを越えたサービス間の通信要件がある場合は、Multi-cluster Service や Anthos Service Mesh で実現



データベースにも可用性を持たせる



【再掲】アーキテクチャ例


東京
Network
Bridge VMs
or PSC NEG
Apigee
Runtime
Internal
TCP LB
GKE
Database
L7 XLB
Network
Bridge VMs
or PSC NEG
Apigee
Runtime
Internal
TCP LB
GKE
Database
大阪

04

まとめ

まとめ

- GKE のマルチクラスタ構成
  - Multi-cluster Ingress、Multi-cluster Gateway で外部トラフィックを分散
  - Multi-cluster Service、Anthos Service Mesh でサービス間コミュニケーション
- Apigee を使う場合
  - API としての機構を GKE の外に出すことで、責任境界を分離できる
  - マルチリージョンで可用性を上げることも可能
  - GKE をバックエンド、Spanner をデータベースとして利用し、リージョン障害にも対応

Thank you.

Google Cloud