CUS-32

スケールする Oracle DB を求めて
～Oisix の Amazon RDS for Oracle 移行への挑戦～

原 智子
オイシックス・ラ・大地株式会社
システム本部 システム基盤部 SREセクション

子安 正史
オイシックス・ラ・大地株式会社
システム本部 システム基盤部 SREセクション

ご紹介の前に

本日ご紹介する内容は、弊社ECサイトのシステム基盤移行プロジェクトにおける、ほんの一部「DBの移行」の内容になります。

このプロジェクトには多くの社内エンジニアだけでなく、事業部の様々な方々の協力、意思決定者のサポート、アマゾン ウェブ サービス ジャパン合同会社様はじめ様々な協力会社様の助力があった上での事例になります。

この場を借りまして、各社様及び社内メンバーに、心から御礼を申し上げます。

本当にありがとうございました！！

Agenda

- Oisix ra daichiってどんな会社？
- 自己紹介
- ECシステムの抱える課題
- DB移行方式検討
- 課題に対するAWS移行後の効果
- これからのOisix ra daichi

- Oisix ra daichiってどんな会社？
- 自己紹介
- ECシステムの抱える課題
- DB移行方式検討
- 課題に対するAWS移行後の効果
- これからのOisix ra daichi


これからの食卓、
これからの畑
より多くの人が、よい食生活を楽しめるサービスを提供します
よい食を作る人が、報われ、誇りを持てる仕組みを構築します
食べる人と作る人とを繋ぐ方法をつねに進化させ、
持続可能な社会を実現します
食に関する社会課題を、ビジネスの手法で解決します
私たちは、食のこれからをつくり、ひろげていきます
Oisix ra daichi

主要事業

Oisix（オイシックス）

主要な顧客層 30代～40代

子育てと仕事の両立に
忙しい世代

らでぃっしゅぼーや

主要な顧客層 40代～50代

料理のスキルアップを
目指したい世代

大地を守る会

主要な顧客層 50代後半～

健康不安を抱える
シニア世代

Oisix ra daichi

Oisix

- 有機野菜やミールキットなど、安心安全でおいしい食材を宅配するネットスーパー
- ミールキットの先駆けであるKit Oisixは累計1億食を突破
- 大根の葉っぱを使ったチップスのようなアップサイクル商品も販売

Oisix ra daichi

- Oisix ra daichiってどんな会社？
- 自己紹介
- ECシステムの抱える課題
- DB移行方式検討
- 課題に対するAWS移行後の効果
- これからのOisix ra daichi

自己紹介

SQL> SELECT NAME, DEPARTMENT, ROLE, HISTORY, FAMILY, FAVORITE FROM ORD_SRE_DBA;

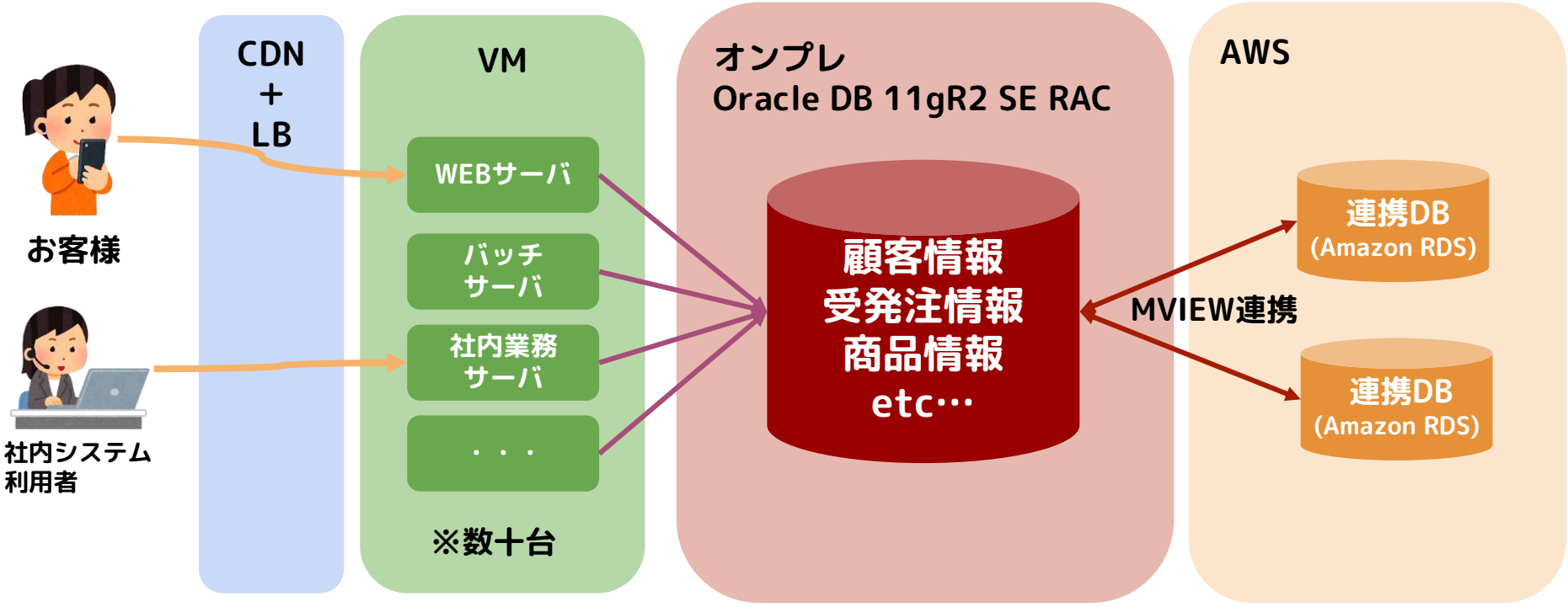
NAME：原 智子（HARA TOMOKO）
DEPARTMENT：システム本部 SREセクション
ROLE：SRE / DBA / PM
HISTORY：10年以上SIer経験を経て2020年JOIN
FAMILY：旦那さんと5歳の娘、水槽にいる🐠
FAVORITE：娘とカフェ巡り☕

Oisix ra daichi

- Oisix ra daichiってどんな会社？
- 自己紹介
- ECシステムの抱える課題
- DB移行方式検討
- 課題に対するAWS移行後の効果
- これからのOisix ra daichi

Oisix ECシステムの課題

過去Oisix ECシステム構成 概要図



Oisix ECシステムの課題

IaC

スケール
しない
システム

ロード
バラン
シング

デプロイ
統一化

環境の
分散

Oisix ECシステムの課題

IaC

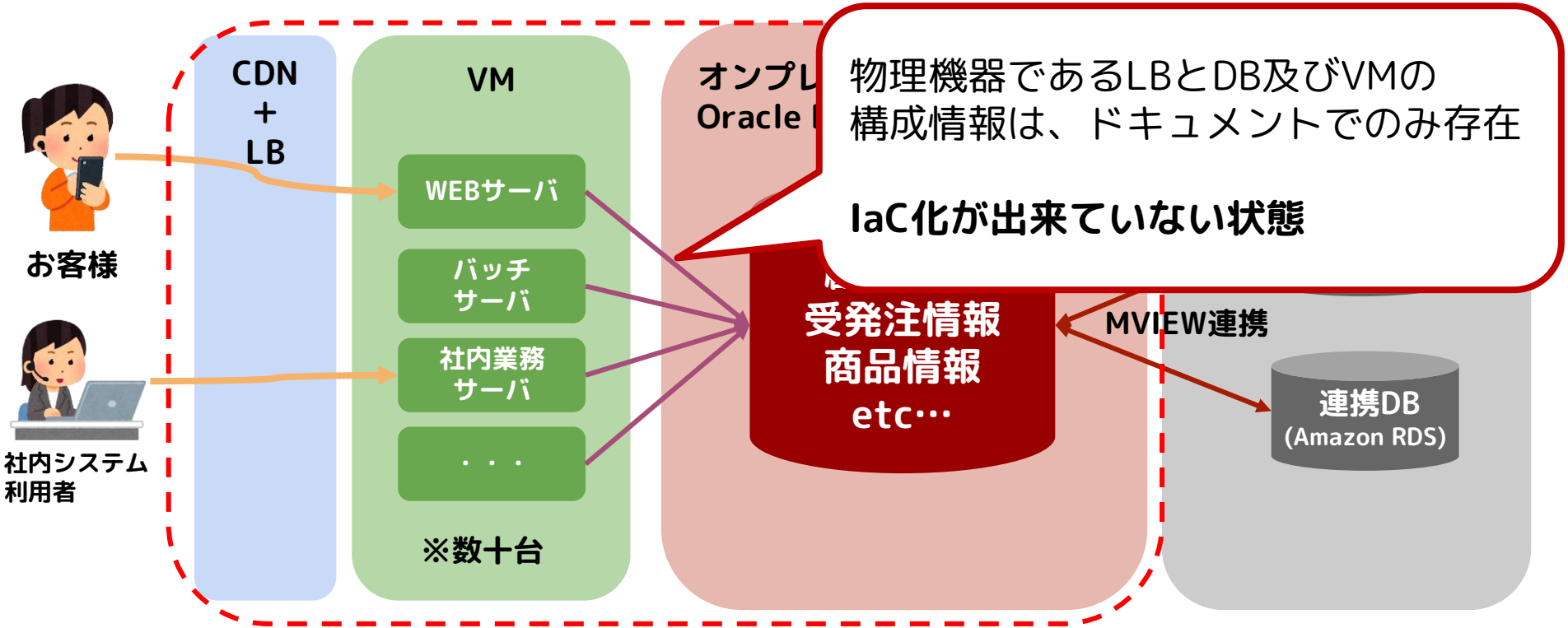
スケール
しない
システム

ロード
バラン
シング

デプロイ
統一化

環境の
分散

Oisix ECシステムの課題



Oisix ECシステムの課題

IaC

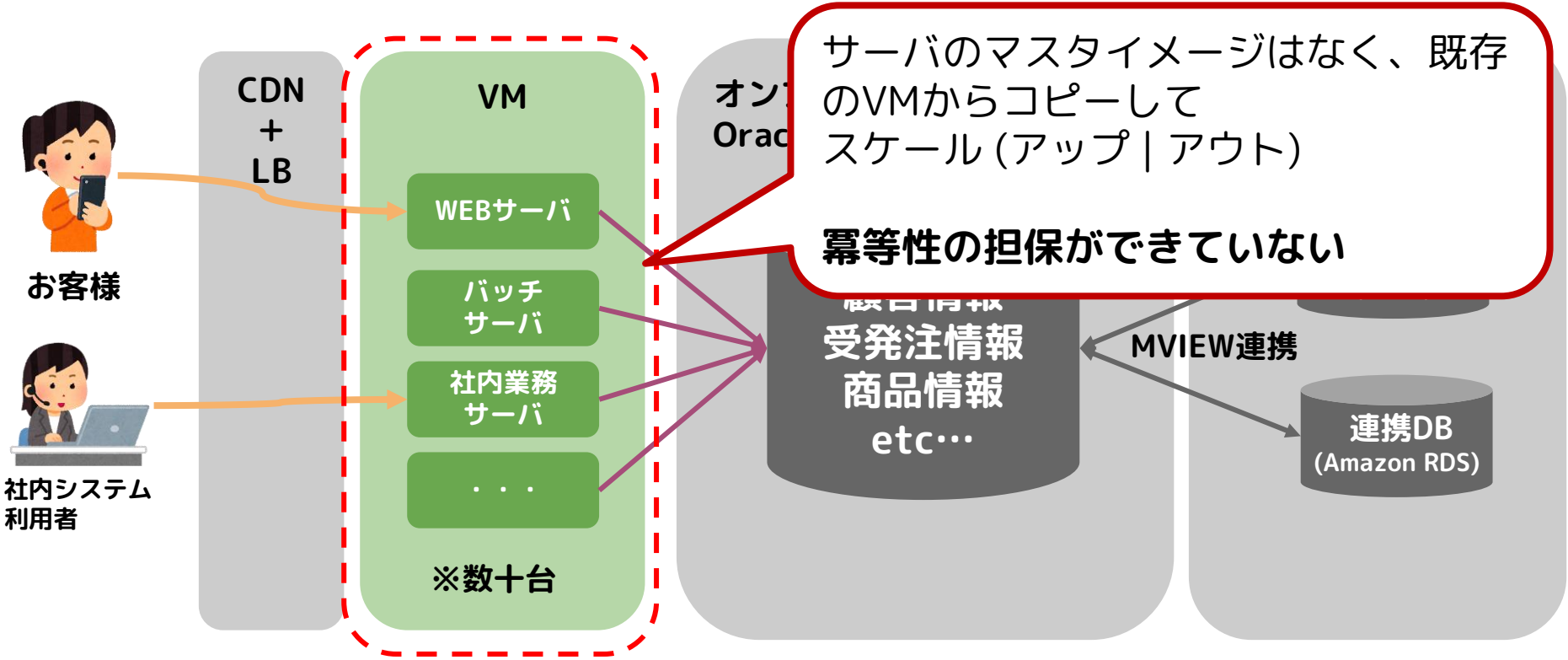
スケール
しない
システム

ロード
バラン
シング

デプロイ
統一化

環境の
分散

過去Oisix ECシステム構成 概要図



Oisix ECシステムの課題

IaC

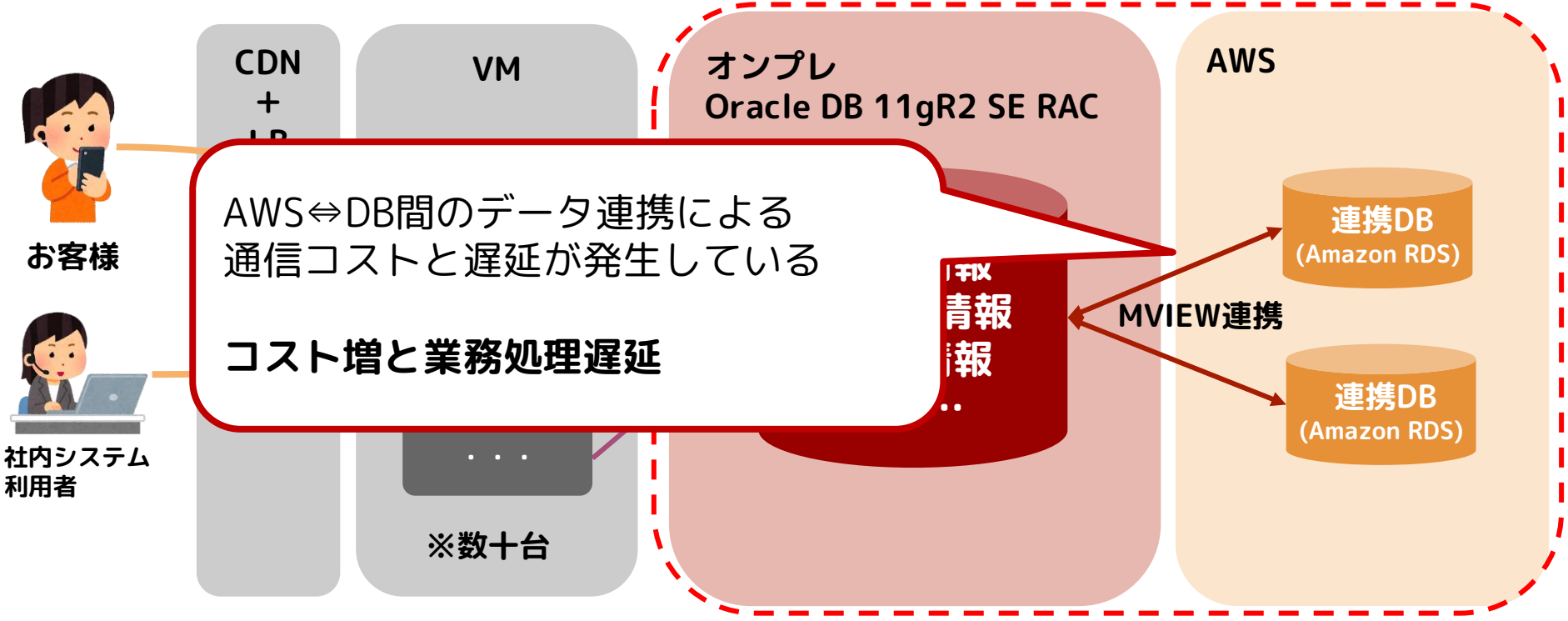
スケール
しない
システム

ロード
バラン
シング

デプロイ
統一化

環境の
分散

過去Oisix ECシステム構成 概要図



Oisix ECシステムの課題

IaC

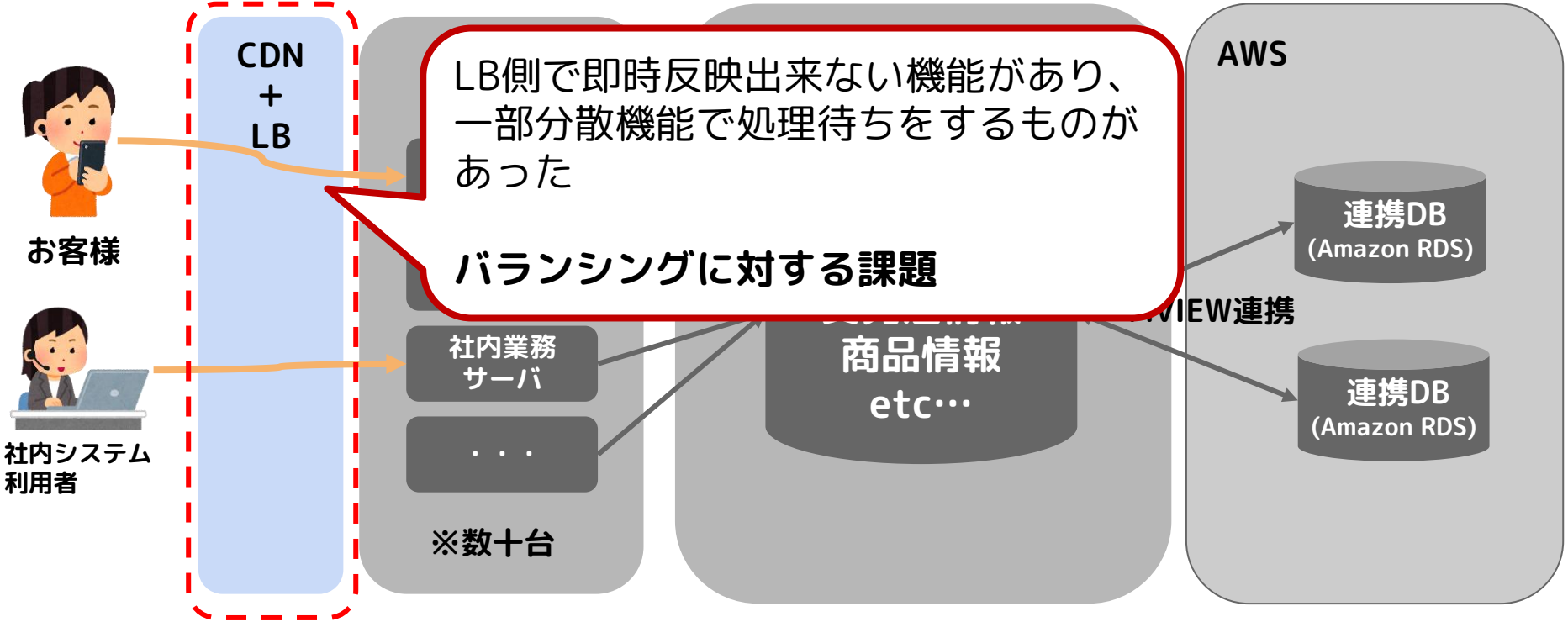
スケール
しない
システム

ロード
バラン
シング

デプロイ
統一化

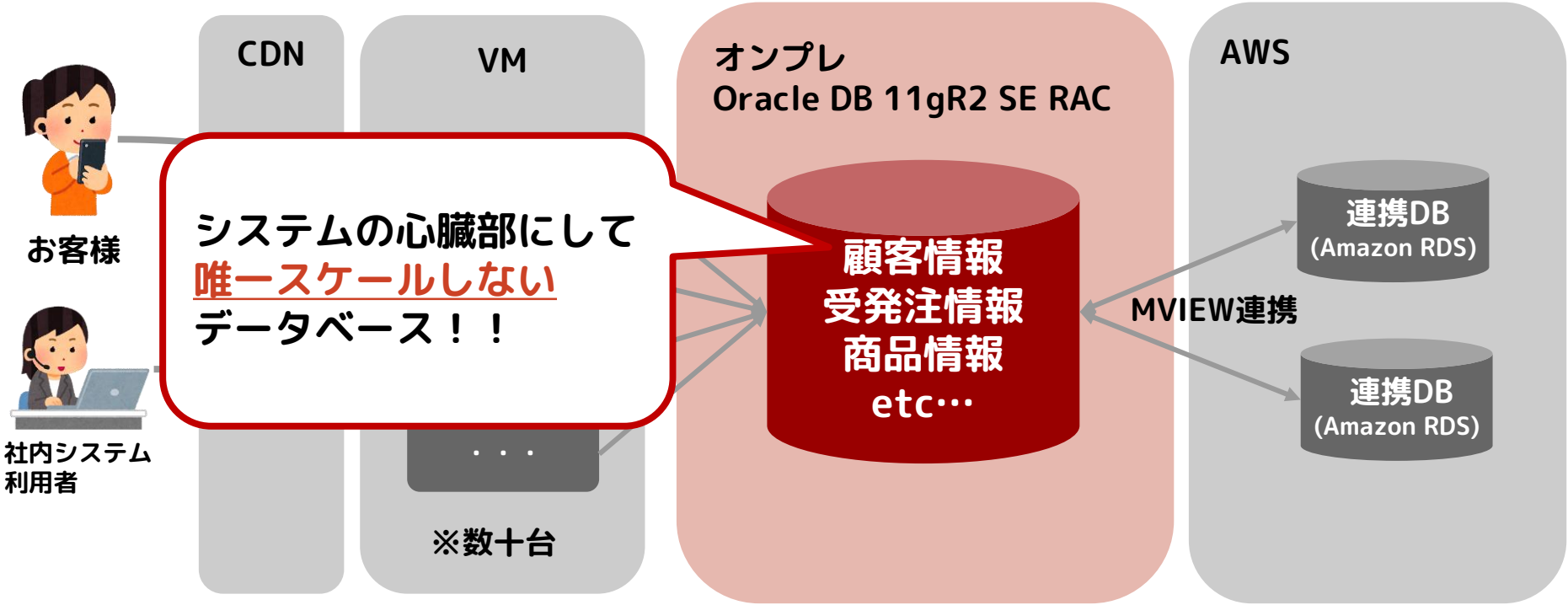
環境の
分散

Oisix ECシステムの課題

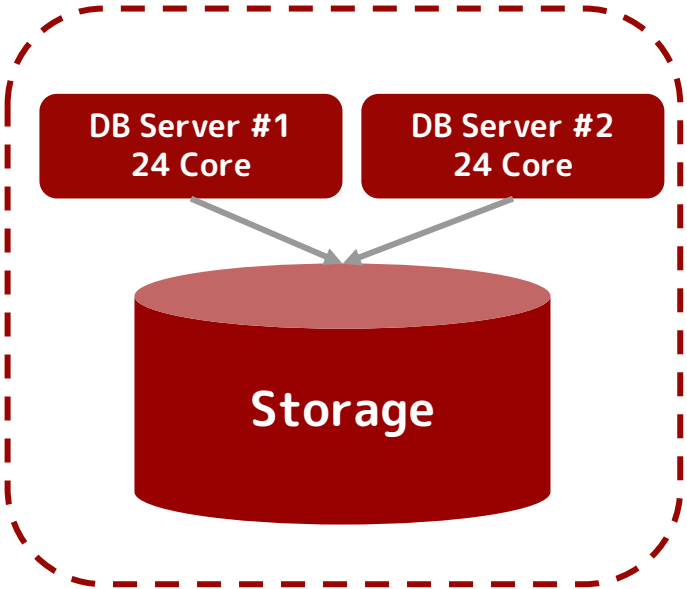


Oisix ECシステムの課題

Oisix ECシステムの課題



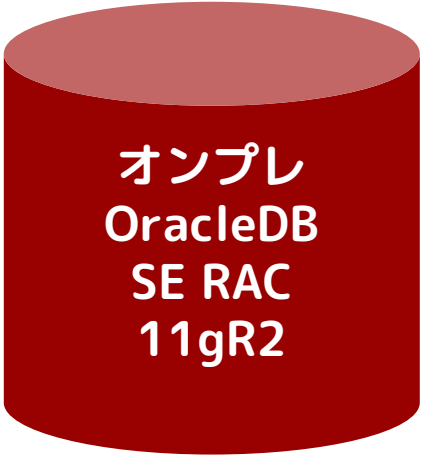
Oisix ECサイトの心臓部のOracle DB



Oracle DB 11gR2
2Node SE RAC

項目	内容
構成	Oracle DB 11gR2 SE RAC
サーバ	24Core *2Node
データ量	約4TB
備考	一部数十年分のデータを保持 テーブル数は数千を超える

DBが抱えていた課題



- スケールしないDB
  - オンプレミス構成で、特にコロナ禍による需要増からDISK使用量が逼迫
    - HWリソースの増強の為にサービス停止が必要、かつこれ以上のスケールは厳しい構成
- サポート期間とバージョンアップの課題
  - Oracle11gR2のサポート終了期限が迫っていた
  - 次のロングサポートバージョンである19cでは現在と同等のHW構成でSE RACを作ることが出来なくなっていた
    - SE RACは無くなりCPU数制限があった

次期DBに対する要件

- 需要増に対し柔軟にスケールするDB
  - HWリソース、特にDISKとメモリ容量に対し柔軟に増強出来るDB
- CPU24以上であること
  - 既存DBサーバのCPUが24であり、同等またはそれ以上のCPU数であること
    - このことからOracle EEにライセンスを変更することを決定

柔軟にスケールする
フルマネージドな Oracle DBが
私達は欲しい！！

クラウドサービスの選択

- 選択肢

※OCI = Oracle Cloud Infrastructure

候補の構成	選択の理由
AWS	当社において先行実績あり
Azure	既に一部サービスをAKSで動かしていた
Azure + OCI	AKS＋αの実績を元にしたアプリケーションとOracle DBの組み合わせ
OCI	Azure + OCIの別解として、OCIの統一

- OCIを早期に選択肢から外した
  - Azureとの相互接続が発表される直前ぐらいの時期で、Oracle DBの優れたコストメリットとAutonomous DBの存在は魅力だった
  - 一方で時期が早かったためにフィージビリティや事例に難点があり、移行を決断できる見通しを立てられなかった
  - 検討した時期によるところがあるため、現状とは異なる

クラウドサービスの決定

※全て選択した2020年当時、かつ当社実績が多分に含まれている

検討項目	AWS	Azure	備考
5年固定資産コスト	△	○	Azureが約15%低額
ランニングコスト	△	○	AWSが年間少々高額
実績	○	×	Amazon RDSの運用が当社先行実績あり
運用負荷の低さ	○	△	**AzureはマネージドOracle DBがなく、 H/W以外は現行と変わらない**

- コストはAzureが優れるが、当社でも世の中でも実績はAWSの方が多い
- 最終的に当初の目的に立ち返り、マネージドでスケールも可能なAWSを選択
- **コスト面は、事業成長に対するシステムの拡張性と柔軟性を得るための投資と捉え、
効率化とボリュームディスカウントやリザーブドインスタンスの利用で低減していく**

次期DBに対する期待

HW構成及びDBパラメータの
IaC化

非RAC構成の代わりとなる
冗長性の担保

既存AWS上のDBとの連携強化

次期DBに対する期待

HW構成及びDBパラメータのIaC化	Terraform管理の実現化
非RAC構成の代わりとなる冗長性の担保	multiAZ構成、且つADG構成で冗長性を保つ
既存AWS上のDBとの連携強化	AWSに移行することで既存DBとの同期速度を向上

備考：他DBMSエンジンへの移行検討

- 他DBMSエンジンへ移行しなかった理由
  - アプリケーションの一部機能がPL/SQLを利用しており、これを他機能に代替することは容易ではなかった
  - 他連携DBもOracle DBであり、DB LINKやMVIEW連携を他機能に変更した場合のリスクが完全に拭えなかった

- Oisix ra daichiってどんな会社？
- 自己紹介
- ECシステムの抱える課題とAWSを選択した理由
- DB移行方式検討
- 課題に対するAWS移行後の効果
- これからのOisix ra daichi

DB移行方式検討において
重要な検討ポイント

ほぼ無停止移行？
or
停止して移行？

RDS for Oracleにおける移行方式検討

移行方式	使用有無	メリット	デメリット	サービス停止有無
RMANによるデータ移行	✗ AWSサポート外	確実に静止点の完全なDB移行が可能	RDS for Oracleではサポート外	最も停止時間が長い
Data Pumpによるデータ移行	○	指定したデータや構成情報等だけを柔軟に移行可能	静止点のデータしか移行出来ない	一定時間停止
DMS/GoldenGateによるデータ移行	○ DMSが便利	静止点のデータや更新データをリアルタイムに移行出来る	静止点データの整合性確認が時間がかかる リアルタイム同期は失敗する可能性がある	停止時間が最も少ない

※Transportable Tableによる移行は未検証のため評価しない

Oisix ra daichi

RDS for Oracleにおける移行方式検討

移行方式	使用有無	メリット	デメリット	サービス停止有無
RMANによるデータ移行	✗ AWSサポート外	確実に静止点の完全なDB移行が可能	RDS for Oracleではサポート外	最も停止時間が長い
Data Pumpによるデータ移行	○	指定したデータや構成情報等だけを柔軟に移行可能	静止点のデータしか移行出来ない	一定時間停止
DMS/GoldenGateによるデータ移行	○ DMSが便利	静止点のデータや更新データをリアルタイムに移行出来る	静止点データの整合性確認が時間がかかる リアルタイム同期は失敗する可能性がある	停止時間が最も少ない

※Transportable Tableによる移行は未検証のため評価しない

サービス停止時間
は
事業売上減と比例する

事業売上減を回避するため
無停止同期を採用するか？

いいえ

私達はその選択をしませんでした

Data PumpとAWS DMSの技術的な比較

移行方式	メリット	デメリット
Data Pumpによるデータ移行	・静止点データをハッシュ値レベルで同一に移行するため確認が早く確実 ・DBエンジンのダンプツールであり推奨且つバージョン間の同期も安全	・静止点データしか移行出来ない ・Exp/ImpやDUMPファイルを転送する等オペレーションの手間がある
AWS DMSフルロードによるデータ同期	・静止点データを移行可能 ・AWSコンソールから簡単に設定し実行可能	・静止点データしか移行出来ない ・内部的にはSELECT/INSERTのためデータ整合性確認はハッシュ値等では行えず、時間がかかる
AWS DMS CDCによるデータ同期	・指定した静止点以降に発行されたSQLをほぼリアルタイムに移行先DBに伝播することが出来る	・複雑なSQLの場合、伝播が失敗し同期が停止する可能性があるため、発行される全てのSQLの伝播検証が必要 ・データ整合性の確認に時間がかかる

Data PumpとAWS DMSの技術的な比較

移行方式	メリット	デメリット
Data Pumpによる データ移行	・静止点データをハッシュ値レベルで同一に移行するため確認が早く確実 ・DBエンジンのダンプツールであり推奨且つバージョン間の同期も安全	・静止点データしか移行出来ない ・Exp/ImpやDUMPファイルを転送する等オペレーションの手間がある
AWS DMS フルロードによる データ同期	・静止点データを移行可能 ・AWSコンソールから簡単に設定し実行可能	・静止点データしか移行出来ない ・内部的にはSELECT/INSERTのためデータ整合性確認はハッシュ値等では行えず、時間がかかる
AWS DMS CDCによる データ同期	・指定した静止点以降に発行されたSQLをほぼリアルタイムに移行先DBに伝播することが出来る	・複雑なSQLの場合、伝播が失敗し同期が停止する可能性があるため、発行される全てのSQLの伝播検証が必要 ・データ整合性の確認に時間がかかる

Data Pumpを採用した3つの理由

データ
整合性

同一
エンジン

ロール
バックの
安全性

Data Pumpを採用した3つの理由

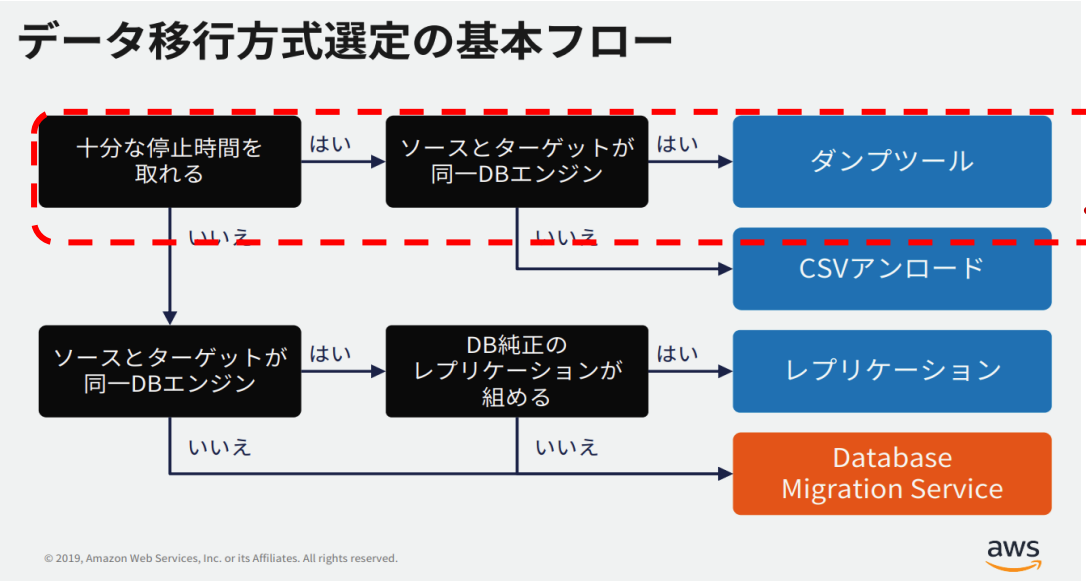
- 移行前後のデータ整合性
  - 全テーブルデータに対し、ORA_HASH等を利用してハッシュ値をCSVに出力する独自ツールを採用
  - Export/Import直後にハッシュ値を出力して差分を確認
    - 確実に同一データであることと、データ欠損が無いことを確認

Data Pumpを採用した3つの理由

- 同一エンジン間の移行であること
  - DBは機能及びデータ保持方法が複雑であり、データ移行における網羅性においては開発元が提供している機能を採用したほうが良いと判断
  - 同一エンジン間のデータ移行ではダンプツールの採用がAWS社資料でも推奨されている

Data Pumpを採用した3つの理由

- 同一エンジン間移行ではダンプツールが推奨



この資料がわかりやすい！

出典元：
https://pages.awscloud.com/rs/112-TZM-766/images/04_20191008_AWS-Database-Migration_screen.pdf

Data Pumpを採用した3つの理由

ロールバックの安全性

- 「確実に」ロールバックが出来る方式
  - 今回の移行ではSE→EEというエディション間の移行だけでなく、11gR2→19cというバージョン間の移行があった
  - 万が一、移行後のDBで問題が発生した際、19c→11gR2へのバージョン戻しを確実かつ迅速に行う必要があった
    - そのため、ロールバック時においても「データの整合性の確認」が同一手順で行える方式を採用した

Data PumpとAWS DMSの技術的な比較

移行方式	メリット	デメリット
Data Pumpによる データ移行	・静止点データをハッシュ値レベルで 同一に移行するため確認が早く確実 ・DBエンジンのダンプツールであり 推奨且つバージョン間の同期も安全	・静止点データしか移行出来ない ・Exp/ImpやDUMPファイルを転送する等 オペレーションの手間がある
AWS DMS フルロードによる データ同期	・静止点データを移行可能 ・AWSコンソールから簡単に設定し実行 可能	・静止点データしか移行出来ない ・内部的にはSELECT/INSERTのため データ整合性確認はハッシュ値等では 行えず、時間がかかる
AWS DMS CDCに よる データ同期	・指定した静止点以降に発行されたSQLを ほぼリアルタイムに移行先DBに伝播 することが出来る	・複雑なSQLの場合、伝播が失敗し同期が 停止する可能性があるため、発行され る全てのSQLの伝播検証が必要 ・データ整合性の確認に時間がかかる

Data Pump案のデメリットに対する対策

- オペレーションの自動化
  - ExportからDUMPファイル転送、Importまでのオペレーションを全てJenkinsにて自動化
  - サイズの大きいテーブルはテーブル名を入力
    小さなテーブルは纏めて移行するように
    予め複数のジョブを作成
  - 作業当日は並列実行にすることで、データ移行の所要時間を短くし、サイト停止時間を短縮化
  - これらの調整と検証ため、データ移行検証を4度実施

決して安易に
サービスを停止をした訳では
ありません

データ不整合 データロスト
これらによる
お客様の信頼を失う方が
私達の最大のリスク

私達はシステムを通じて
サービスを提供し
お客様と信頼関係を築くこと

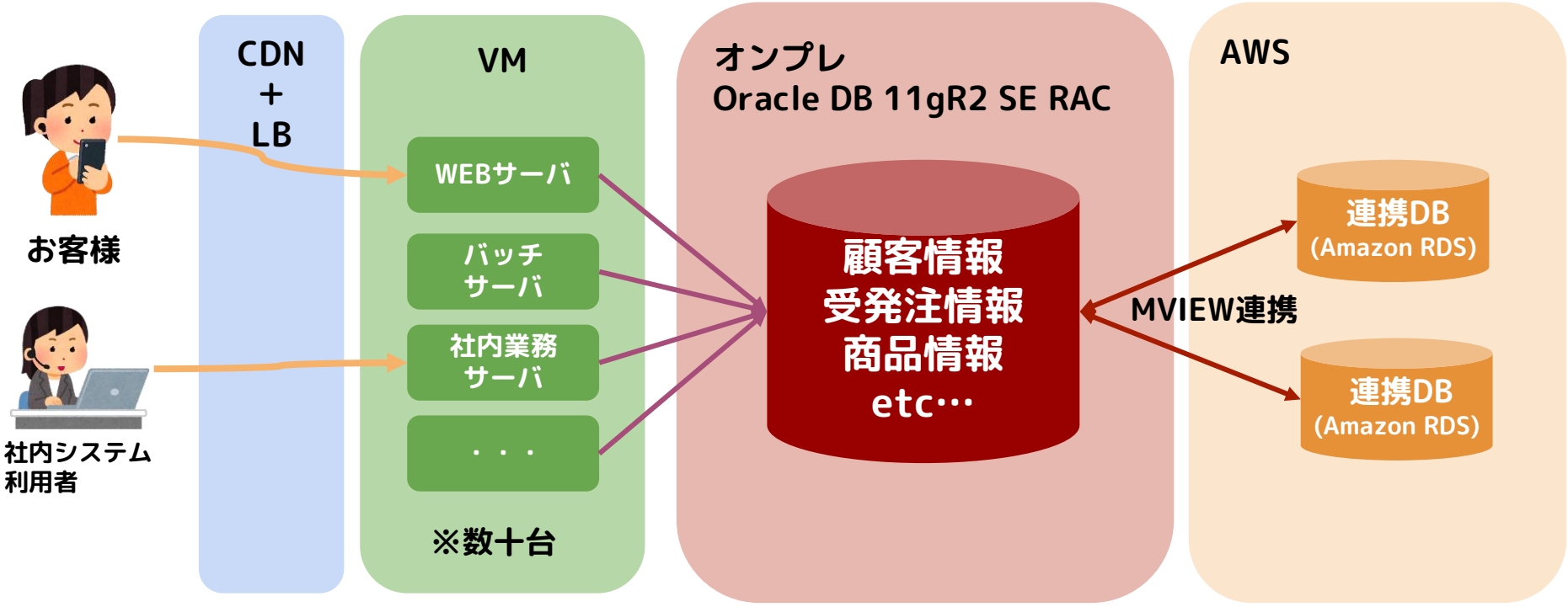
- Oisix ra daichiってどんな会社？
- 自己紹介
- ECシステムの抱える課題
- DB移行方式
- 課題に対するAWSに移行後の効果
- これからのOisix ra daichi

自己紹介

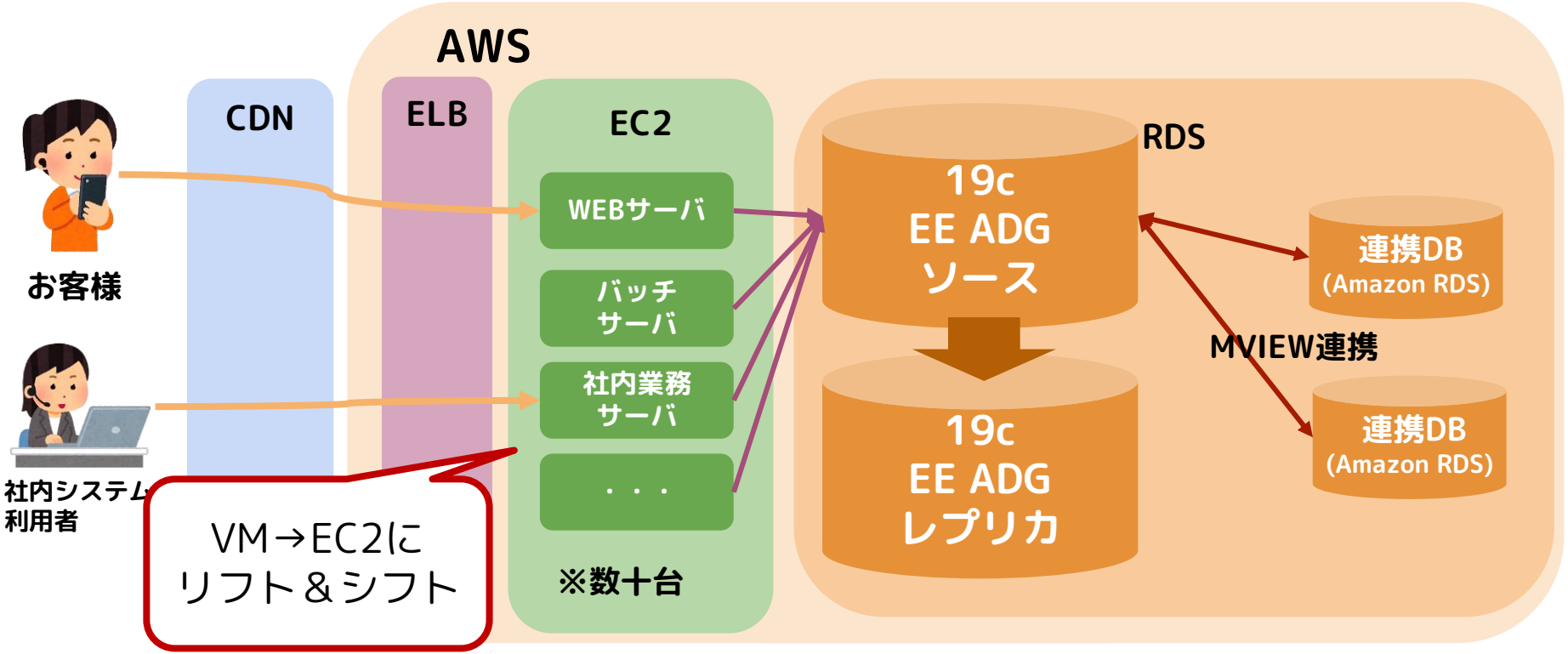
SQL> SELECT NAME, DEPARTMENT, ROLE, HISTORY, FAMILY, FAVORITE FROM ORD_SRE_DBA;

NAME：子安 正史（KOYASU MASASHI）
DEPARTMENT：システム本部 SREセクション
ROLE：SRE / DBA
HISTORY：DBAとして6年、SREとして2017年JOIN
FAMILY：最近1歳になった🐱
FAVORITE：最近購入した車でDrive

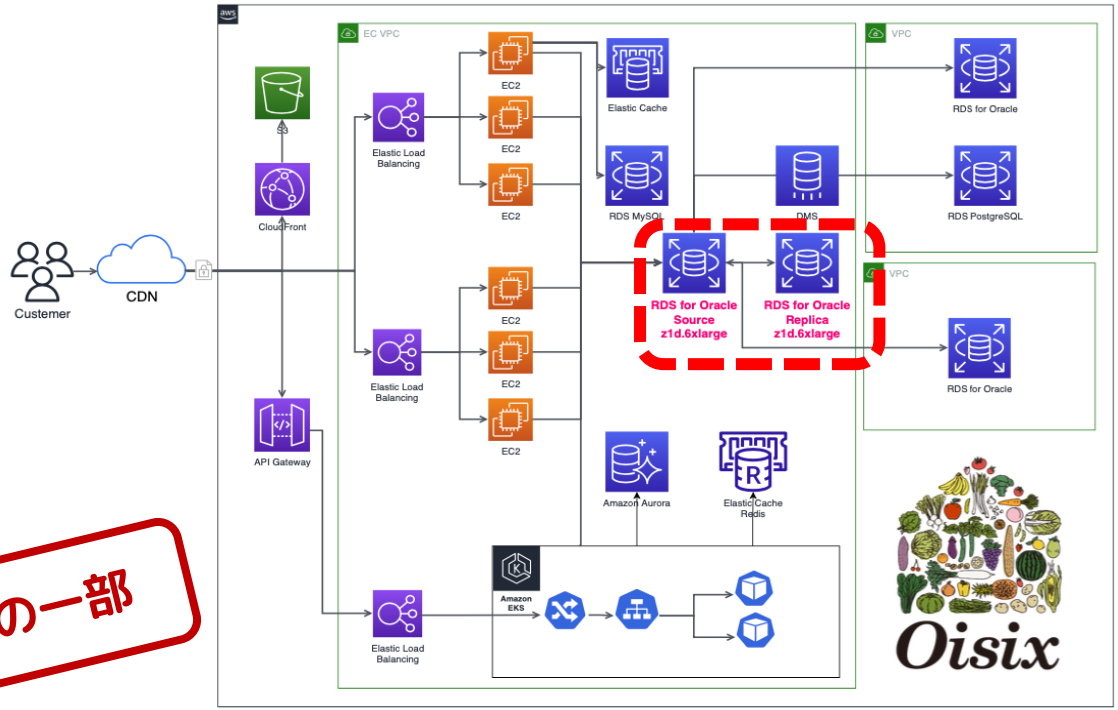
過去Oisix ECシステム構成 概要図



現在のOisix ECシステム構成 概要図



現在のOisix ECシステム構成 概要図



ほんの一部

Oisix ECシステムの課題

Oisix ECシステムの課題

IaC

スケール
しない
システム

ロード
バラン
シング

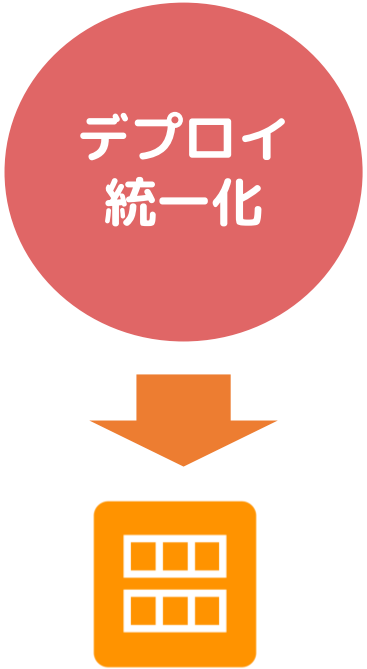
デプロイ
統一化

環境の
分散

IaCに対する課題解決

- TerraformによるIaCを実現
  - 現在の環境はTerraformによりIaC化を実現
  - これにより構成管理がコード化され、
    ドキュメントによる構成管理から脱却
  - リソース増減についてもTerraformのデプロイ
    で実施することで、容易に実現可能に

デプロイ統一化に対する課題解決



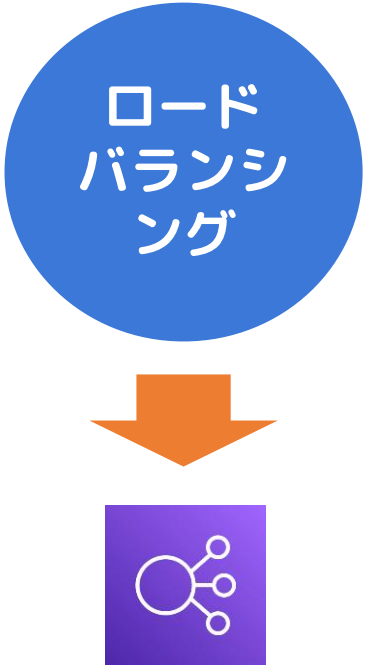
Amazon マシンイメージ (AMI)

- AMIによるOSイメージのテンプレート化
  - マスタとなるAMIを作成し、
    OSレイヤの設定について統一を実現
  - AMIに対する設定についても、
    Ansibleでコード化し、マスタとなるAMIの
    設定変更も柔軟に対応可能に

環境の分散に対する課題解決

- AWSに環境集約し、基盤環境間の通信コストや通信遅延を解決
  - 点在していた環境をAWSに集約
    これにより基盤間の通信コストを大幅に削減
  - 基盤間のNWレイテンシーで遅延していた
    データ同期等の遅延も、VPCピアリングにて
    解決

ロードバランシングに対する課題解決



Elastic Load Balancing

- Elastic Load BalancingによりBlue/Green環境への柔軟なトラフィックコントロールを実現
  - Blue/Green環境へのデプロイ時、既存LBではトラフィック分散のコントロールが即時ではなかったが、 Elastic Load Balancingにより即時分散を実現化
  - 分散コントロールについてもAnsible又はAWS CLIにてコマンド化し、リリース時に自動で分散コントロールを実施

スケールしないシステムに対する課題解決

スケール
しない
システム

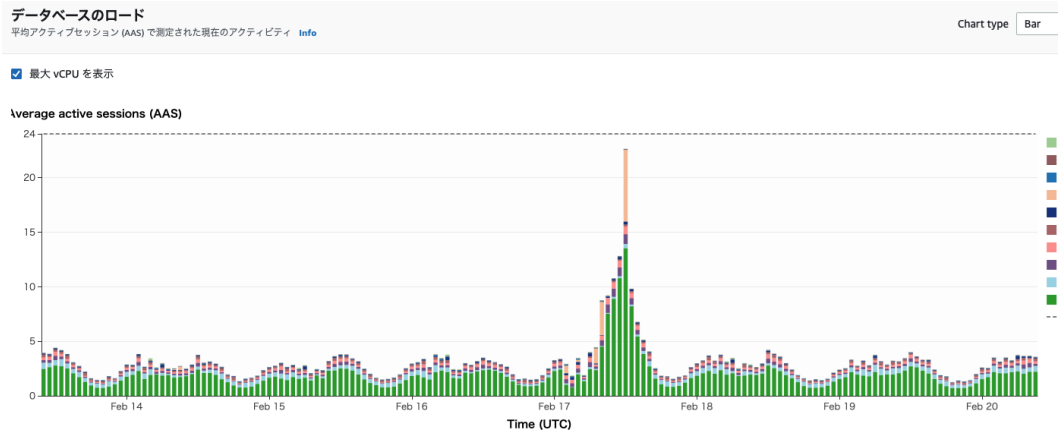
Amazon RDS

- 柔軟にスケールアウトするシステムに
  - Amazon RDS for Oracleになることで、リソースを柔軟に追加することが可能に
  - 将来のトラフィック増にはADGのレプリカを増やしSELECT処理を退避させることなどによりスケールアウトが可能

※EnterpriseEditionはBYOLモデルしかないため、インスタンスやCPU増時はOracle社より追加ライセンスの購入が必要になります。

Amazon RDS for Oracleにしたことによる相乗効果

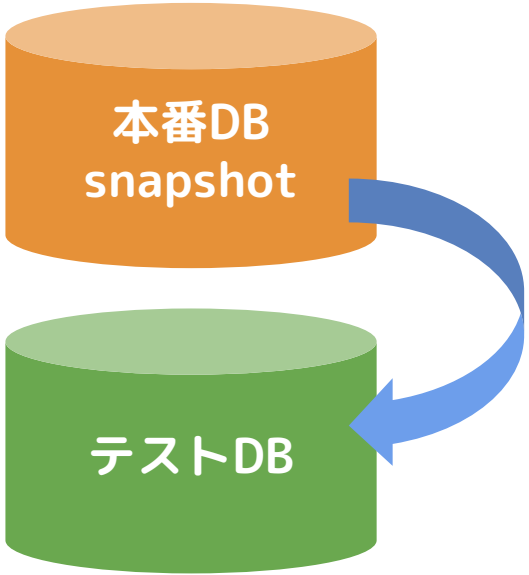
Performance Insightによる可視化



- Performance InsightによるDBの内部情報の可視化
  - 今までDBAでしか確認出来なかったDBの待機イベントや長時間SQLをDBA以外のメンバ、特に開発者が確認出来るように
  - これによりDB高負荷時の問題や課題を開発者と共有することが実現

Oisix ra daichi

より本番に近い開発DBの作成



- 本番DBとほぼ同様のテストDBを
  - 過去のテストDB（開発やSQL試験用のDB）は本番DBと比較して非RAC構成やデータ量の乖離等の問題があった
  - 現在はAWS CLIで毎週末に本番DBのsnapshotから、個人情報等をマスキングして複製
    - 既存のテストDBの課題であった本番と同じ構成、データ量を実現
  - 本番DBのワークロードを保持していることから、実行計画レベルのチューニングもテストDBで一部実現可能に

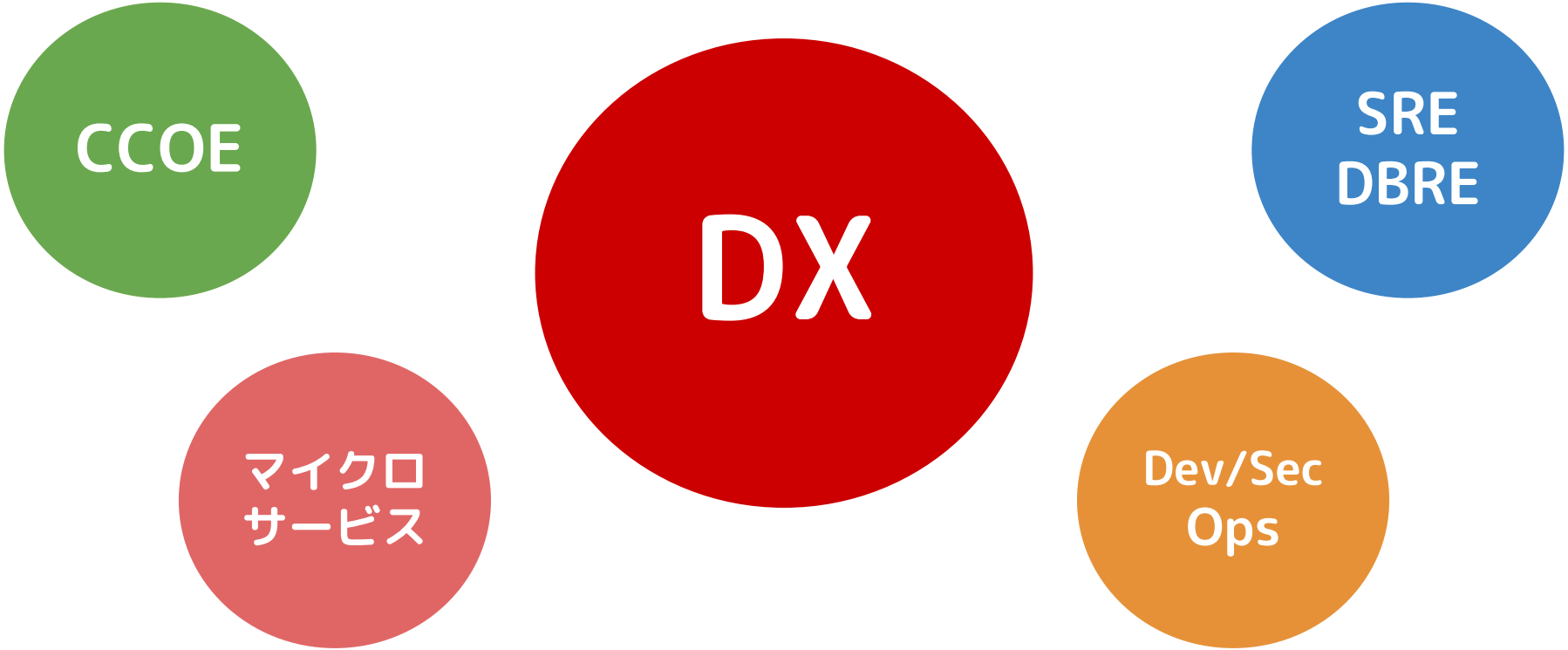
- Oisix ra daichiってどんな会社？
- 自己紹介
- ECシステムの抱える課題
- DB移行方式
- 課題に対するAWSに移行後の効果
- これからのOisix ra daichi

過去と今、そしてこれから

項目	過去	今	これから
環境	オンプレ＞クラウド	ほぼクラウド	オンプレ環境からの 完全脱却
インフラ管理	設計書管理ベース	クラウド環境はIaC	設定管理は完全IaC化
アーキテクチャ	柔軟にスケールしない インフラ	クラウド環境は 柔軟にスケール	より柔軟なスケールを DBもCloudNative化へ
運用	大半が手動運用	一部オートスケール	オートスケール オートヒーリング

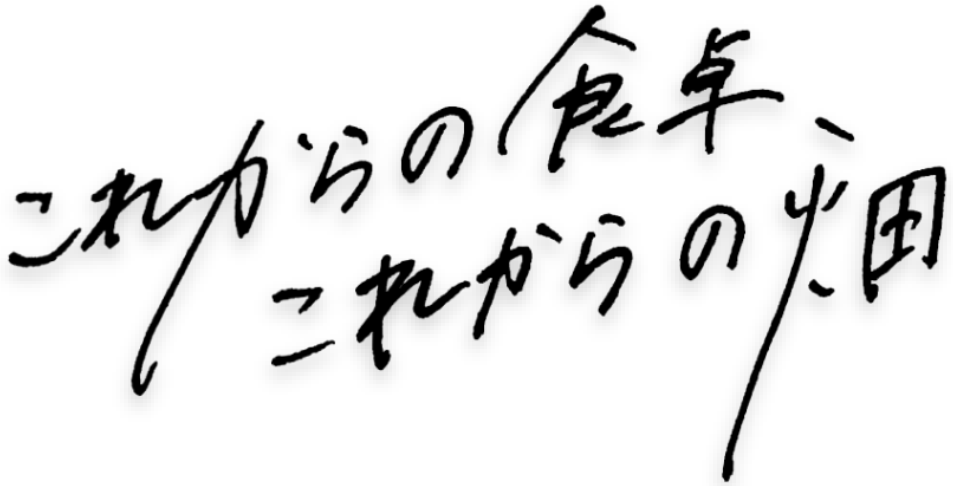
スケールする
ヒーリングする
それがGOALではない

クラウドへ移行して成し遂げていくこと



お客様、生産者の方へ
最適な提案
安心安全なサービスを提供する

これからのシステムへの
第一歩を



ご清聴ありがとうございました！

Thank you!

原 智子

オイシックス・ラ・大地株式会社
システム本部 システム基盤部 SREセクション

子安 正史

オイシックス・ラ・大地株式会社
システム本部 システム基盤部 SREセクション