技術解説

次世代品質データ管理システムの開発

Development of a next-generation quality data management system

古　川　　輝
FURUKAWA Akira

要　　旨

近年，世界的にDX（Digital Transformation：デジタルトランスフォーメーション）が注目されており，製造業，金融，IT（Information Technology：情報技術）などのあらゆる業界でDXの取り組みが実施されている．

しかし，日本国内においてはDXによる十分な成果が出ている企業は少なく，米国企業と比べても遅れている状況である．その要因の一つとしてレガシーシステム（古い技術や仕組みで構築されたシステム）の存在が挙げられる．レガシーシステムは度重なる機能追加によるシステムの複雑化や長期稼働に伴うデータの肥大化などで保守性が低下しており，システムの運用コストを増加させている．また，DX推進のための改良や他システムとの連携も困難であるため，レガシーシステムからの脱却が求められている．

一方，カヤバでも多くのレガシーシステムを抱えており，これらのシステムの運用管理に多くの工数を費やしている．また，個別最適で構築されているため，DX推進のための他システムと連携したデータ活用が困難となっている．

そこで，これらの課題を解決するためにレガシーシステムの一つである品質データ管理システムを刷新し，次世代のシステムとして次世代品質データ管理システムを開発した．当社では次世代システムを，「障害・災害対策を各段に向上させ，複数システムを連携して多角的な分析が実施でき，費用面と運用管理工数の両方において画期的な低コストで実現できるシステム」と考えており，本システムもこの考えに基づいて開発している．

本報では開発したシステムの各要素機能とレガシーシステムからの移行方法，今後システムの運用を維持するために構築した開発・運用体制の仕組みや特徴について解説する．

Abstract

In recent years, Digital Transformation（DX）has been attracting attention worldwide. All sorts of industries, including manufacturing, finance, and information technology, are working on DX.

However, few companies in Japan have achieved adequate results with DX, and they lag behind their U.S. counterparts. One reason for this is the existence of legacy systems. Legacy systems are becoming less maintainable due to increased system complexity and data bloat, which increase system operating costs. In addition, it is difficult to improve and integrate these systems with other systems to drive DX, so there is a need to break away from legacy systems.

KYB also has many legacy systems and spends many man-hours on their operation and management. It is also difficult to utilize data linked with other systems to promote DX.

To solve these problems, we developed a next-generation quality data management system by renewing a legacy quality data management system. We believe this next-generation system will be a revolutionary low-cost system in terms of both cost and man-hours required for operation and management, and have developed the system based on this concept.

This paper describes the functions of the developed system, the migration method from the legacy system, and the development and operation system for continuous operation of the system.

1 緒言

近年，世界的にDXが注目されており，製造業，金融，ITなどのあらゆる業界でDXの取り込みが実施されている．日本国内においても多くの企業でDXの取り組みを開始しており，2022年度には69.3%まで増加している[1)]．

しかし，日本国内においてはデジタル化による成果は出ている一方で，顧客価値の創出やビジネスモデルの変革などで成果を出している企業は少なく，米国企業と比べて人材や技術の両面でDXの推進が遅れている状況となっている．DXの推進が遅れる要因はいくつか考えられるが，代表的な要因として以下の3点が挙げられる．

①組織体制の問題
②人材不足
③レガシーシステムの存在

①については経営者が経営戦略としてDXを取り組むための指針が提示できておらず，組織全体として活動する体制になっていないことである．

②についてはDXを推進するためにはクラウドやAI[注1)]などのデジタル技術を活用する必要があるが，これらの技術を活用するためのデジタル人材が不足していることである．

③については古い技術や仕組みで構築されたレガシーシステムが多く存在しており，これらのシステムではシステムの肥大化・複雑化，ブラックボックス化によってDX推進のための改善が困難になっていることである．また，レガシーシステムを運用するためにITエンジニアが多く工数を費やしているため，改善のために時間を費やせない問題も発生している．

よって，これらのDXを推進する上で，まずはこれらの要因を解決することが必要である．

一方，当社でもDX推進のための活動を実施している．2019年にDXを推進するための部署であるDX推進部を設立し[注2)]，各種取り組みを実施してきた．まずはデータ活用を加速するため，データを利活用する基盤として，AWS（Amazon Web Services）クラウド上にカヤバ-IoTプラットフォームを自社で構築した[2)]．そして，この基盤上に生産領域におけるDXとして設備予知保全システムの開発[3)]，製品開発領域におけるDXとしてSA（ショックアブソーバ）要素開発へのAI技術活用[4)]などに取り組んでいる．さらに，デジタル人財育成にも取り組んでおり[5)]，会社全体でDXを推進するための体制も整えている．

しかし，レガシーシステムの脱却に向けた取り組みについてはこれまで実施できておらず，これらのシステムの運用管理に多くの工数を費やしている[注3)]．また，これらのシステムは個別最適で構築されているため，DX推進のための他システムと連携したデータ活用や新たな機能開発が困難な状況となっている．

そこで，これらの課題を解決するためレガシーシステムの一つである品質データ管理システムを刷新し，次世代のシステムとして次世代品質データ管理システムを開発した．当社の考える次世代システムとは，「障害・災害対策を各段に向上させ，複数システムを連携して多角的な分析が実施でき，費用面と運用管理工数の両方において画期的な低コストで実現できるシステム」である．そして以下の特徴を持つシステムであると考えており，本システムにおいてもこの考えに基づいて開発を実施した．

①クラウドを活用した障害・災害に強いシステム開発と運用管理工数の低減
②ローコード開発ツール活用による開発者の負担軽減と開発後のメンテナンス属人性の排除
③システム異常（エラー処理や不正操作）検知による早期不具合の発見と解決
④データ集約，統合による異なるシステム間でのデータ連携の容易化
⑤IaC[注4)]を用いたシステム構成管理とCI/CD[注5)]を活用したテストと実装の自動化による，更新時の障害の未然防止，管理者・開発者の属人性の排除

注1）Artification Intelligenceの略で人工知能のこと
注2）2023年4月にはDX推進部とIT企画部が統合されてデジタル変革推進本部となっている
注3）当社では従来から生産領域におけるシステムを自社開発していたが，それらのシステム運用をDX推進部で実施していた
注4）Infrastracture as Codeの略でシステムインフラの構築を，コードを用いて行うこと
注5）Continuous Integration/Continuous Deliveryの略でソフトウェア開発において，ビルドやテスト，リリース，デプロイなどの工程を専用のツールなどを用いて自動化し，開発の効率化や省力化，本番環境への迅速な反映を図る手法のこと

2 従来システムの概要と課題

従来の品質データ管理システムについて紹介する．本システムは設備や試験機で収集した製品の加工条件や試験計測値などの品質情報を収集・蓄積するシステムである．計測値の推移や傾向を確認することで日々の品質管理や不具合発生時のトレサビリティ

などに活用している．本システムはパワーステアリングやCVT[注6)]を生産している国内外の6拠点に対して導入しており，最初に導入した拠点では20年以上経過している．

注6）Continuously Variable Transmissionの略で無段変速機のこと

2.1　システム構成

従来のシステム構成図を図1に示す．各設備で収集した品質データを記録したファイルをオンプレミスサーバに集約し，サーバ上でデータを一部整形した後にリレーショナルデータベース（以下RDB）に保存して一元管理している．そして，自社で開発した専用ソフトを使ってRDBにアクセスすることで実績データの閲覧やシステムに必要なマスタデータの登録を実施している．RDBはOracle社のDBを活用しており，実績データの閲覧やマスタデータを登録するソフトはMicrosoft社のVisual Basic .NETで自社開発している．

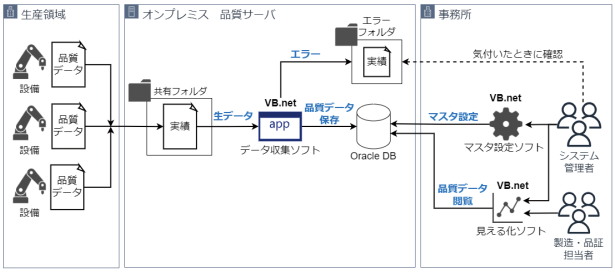

図1　従来システムの構成図

2.2　従来システムの課題

従来システムの課題を以下に示す．

①システムの運用管理工数やコストの増加
- サーバの定期的な更新や保守，老朽化によるトラブル対応などで工数とコストが増加している
- RDBのライセンス費用と保守費用が増加している
- 当時開発者の不在とドキュメントの不備により，機能追加や不具合修正時に多くの工数を費やしている

②障害・災害対策が不十分である
- システムの冗長化は実施していないためサーバやRDBの障害時にシステムダウンしてしまう
- 地理的に離れた場所でのバックアップが実施できていないため，自然災害によりシステムを復旧できない可能性がある

③データ活用の推進が困難
- 個別最適で構築しているため他のシステムとの連携が困難である
- 20年以上前に設計したシステムであるため技術が古く，非効率となっている処理がある

3　要件

従来システムの課題を解消し，システムとして運用していくための要件を以下に示す．

①カヤバ-IoTプラットフォーム上にシステムを構築し，それにあわせてクラウドはAWSを活用すること．

②システムを単にクラウドに移行するだけでなく，より最適な仕組みとなるように再構築すること．

③従来の主要機能は継承し，使い勝手の悪い機能を改善すること．

④水平展開や機能追加・保守が容易であること．

⑤RDBは商用DBからOSS[注7)]のDBに移行すること．

②について補足する．AWSではクラウド移行するにあたり7つのR[6)]と呼ばれる移行戦略を提示している．その移行戦略について表1に示す．移行難易度の高い順に並んでいるが，本開発で狙っているのは一番難易度の高いRefactorである．開発に時間を要するが，最もクラウド移行による効果が期待できる．

表1　クラウド移行における7つの移行戦略

移行戦略	概要
Refactor	クラウド中心の機能を最大限に活用するようにアプリケーションを再設計する
Replatform	特定のコンポーネントに対してクラウドベースのサービスに移行する
Repurchase	SaaSやパッケージを購入して乗り換える
Rehost	クラウドインフラストラクチャーに既存オンプレミスの構成をそのまま移行する
Relocate	VMWare Cloud on AWSに既存オンプレミスの構成をそのまま移行する
Retain	クラウド移行しないで現状を維持する
Retire	廃止する

注7）Open Source Softwareの略でソースコードの改変や配布が自由に認められている無償ソフトウェアのこと

4　新システムの開発

4.1　システム構成

新たに開発したシステム構成の概略図を図2に示す．

生産領域の処理には変更を加えず，それ以降の赤枠で囲った領域に対して再構築を実施している．

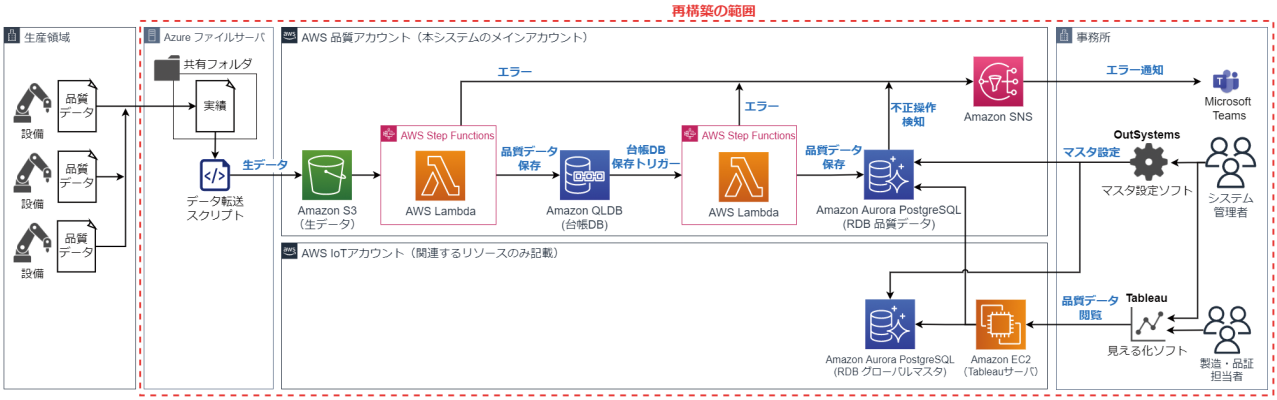

図 2　新システムの構成図（概略図）

各設備で収集した品質データファイルを集約するサーバにはMicrosoft Azureの仮想サーバを利用した．今回クラウド上に構築したリソースで唯一のIaaS[注8)]であるが，設備側の処理を変更しないでデータ収集するためには従来システムと同様にWindowsサーバが必要であったためである．また，当社ではこれまでいくつかのWindowsサーバをAzure上に構築しており，統括的に管理するために本システムもAzure上にサーバを構築している．本サーバは主に品質データファイルの一時保存が目的であり，その後AWSにそのままファイルを転送する．

収集した品質データを保存して一元管理する処理についてはAWS上に構築した．図中では詳細は割愛しているが，ほとんどの処理にAWSマネージドサービスを活用しているため，管理者の負担を軽減することができ，障害や災害に対して強い構成となっている．

参考までにRDB管理における従来と新システムとの比較を図 3 に示す．新システムではサーバやネットワーク，DBミドルウェアの管理などが不要となり，大幅に管理者の負担を軽減することができる．

図 3　RDBの構築・運用管理における責任範囲の比較

拠点ユーザが利用する，収集した品質データを閲覧する見える化ソフトとシステムで必要なマスタデータを設定するソフトをローコード開発ツール[注9)]であるTableau[注10)]とOutSystems[注11)]を使って再開発した．いずれもドラッグアンドドロップの操作主体で少ないソースコードで容易にソフト開発が可能であり，従来よりも大幅に開発工数を低減することができた．開発したソフトの詳細については4.5節で解説する．

本開発での具体的な実施事項は大きく以下の 6 つに分類され，詳細は次節より解説する．

①データ移行
②データ保存・管理
③エラー・不正操作検知
④見える化ソフトの開発
⑤マスタ設定ソフトの開発
⑥データ連携性の向上

注 8）Infrastructure as s Serviceの略でインフラストラクチャーを提供しているクラウドサービスのこと

注 9）最小限のソースコードでソフトウェア開発を行うためのツール

注10）Salesforce社が提供するビジネスインテリジェンス（BI）ツールで，多彩なグラフやチャートを用いたビジュアル分析が可能

注11）OutSystems社が提供しているローコード開発プラットフォームで，Webアプリやモバイルアプリなどの様々なアプリケーション開発が可能

4.2　データ移行

新システムを稼働するにあたり，これまで従来システムで蓄積していたデータを新システムに移行する必要がある．本開発では従来の構成でそのままデータ移行するのではなく，以下の変更を加えてデータ移行を実施した．

①データベースタイプをOracleからOSSのPostgreSQLに変更

②スキーマとテーブルの構成や名称を変更

①への変更理由を以下に示す.

・OSSのDBを選定することによりライセンス費用と保守費用を削減

・Oracleと互換性の高いデータ型や関数を多くもっているため移行が比較的容易

・性能面で高いパフォーマンスをもっている

②への変更理由を以下に示す.

・従来のDBがデータの肥大化によってパフォーマンス低下していたため構成を見直した

・今後の他システムとの連携を考慮してデータの持ち方を統一させた

データ移行の流れを図4に示す.

まずはオンプレミスのOracle のデータをAmazon RDS for Oracleに移行し，一部データ加工を加えてからAWS Database Migration Service（以下DMS）を使って全データをAmazon Aurora PostgreSQL（以下Aurora PostgreSQL）に移行している．そしてAurora PostgreSQL上でもデータ加工・変換を実施することで最終的に目的とする構成でデータ移行が完了する.

本手法では一度にまとめてデータ移行を実施するため手順としてはシンプルであるが，データ移行中は移行データ以外のデータ更新を避けるためにシステムの一部機能を停止させる必要がある．システムの停止時間を短くするための手法もあるが，その分移行作業が複雑化して作業が難しくなってしまう．本システムでは数日のシステム停止は許容できるので手順がシンプルな本手法を採用した.

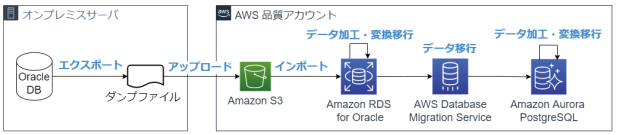

図4　データ移行の流れ

4.3　データ保存・管理

設備で収集した品質データは製品の品質や性能を証明するために重要な情報であるため厳重に管理して必要に応じて即座に検索できる必要がある．本開発では最終的なデータ保管先としてAmazon Quantum Ledger Database（以下QLDB）とAurora PostgreSQLの2つのDBを利用する構成としている．QLDBはフルマネージド型の台帳DBであり，一度保存したデータは管理者であっても物理的に変更ができない特徴をもっている．また，Aurora PostgreSQLはクラウド向けにAWSが構築した高性能マネージド型のRDBであり，通常のRDBよりも高いパフォーマンスと可用性を実現している.

QLDBに品質データを保存することで物理的にデータ変更ができなくなるため，確実に改ざんできないデータ管理の仕組みが実現できている．しかし，QLDBはデータの保管・管理が主機能であり，従来のような複数の情報を紐づけるような複雑なデータ検索をすることは困難である．そこで品質データをRDBのAurora PostgreSQLに対しても保存するようにし，ユーザからはAurora PostgreSQLに対してアクセスしてデータを検索できるようにした．また，システムを稼働するために必要なマスタデータについても状況に応じて変更する必要があるため，あわせてAurora PostgreSQLで管理するようにしている.

このようにタイプの異なる2つのDBを活用することによって改ざん防止とデータ検索性を両立させたデータ管理を実現している.

4.4　処理エラー・不正操作通知

品質データを各DBに保存する際にはデータの書式や情報が正しいかのチェックを実施しており，最終的に問題なかった場合にデータ保存処理に進むようにしている．チェック処理については従来システムでも実施しており，エラーになった際は対象ファイルをエラー用フォルダに移動するようにしていたが，エラーが発生したことを通知していなかったため対応が遅れてしまうことがあった.

そこで本システムではエラー発生時にMicrosoft Teams（以下Teams）に対して通知する処理を開発した．図5がTeamsに通知されるメッセージのサンプル画面であるが[注12)]，メッセージの内容からどの設備のどのファイルでエラーが発生しているかを把握することが可能である．また，メッセージ上のURLをクリックすることで後述するマスタ設定ソフトが表示され，ソフト上からエラーとなったファイルをダウンロードして対処することが可能である．このようなエラー通知と対処の仕組みにより，従来の課題の解決を図った.

図5　書式エラー時のTeams通知メッセージ事例

また，通知機能に関してはAurora PostgreSQLの不正操作に関しても実装している．4.3節で示したようにQLDBに保存されているデータについては管

理者であっても改ざんできないため特に対策する必要はないが，ユーザが参照しているのはRDBのAurora PostgreSQLであるためそのDBのデータが改ざんされてしまった場合は問題である．一方，Aurora PostgreSQLでは厳密なアクセス管理をしており，一般ユーザではデータを変更できないようにしているため通常の運用でデータが改ざんされることはない．しかし，万が一管理者のアクセス権限が流出し，その権限を使ってアクセスされた場合は不正操作されてしまう可能性がある．

そこでAurora PostgreSQLに保存されている品質データに対して変更や削除が実行されたり，通常とは異なるユーザや場所からデータ追加や不正に繋がる可能性がある操作が実行された場合にTeamsに対して通知するようにした．Teamsに通知されるメッセージのサンプル画面を図６に示す[注13]．メッセージからいつ，どこで，誰が，何に対して，どのような操作をしたが把握できるためその情報から不正操作なのかを判断し，その後の処置を行うことが可能である．また，必要に応じてQLDBに保存されているデータと比較することでデータ改ざんの有無やデータの真値の把握が可能となっている．

注12）メッセージ内容はサンプルであり，実際とは異なる

注13）メッセージ内容はサンプルであり，実際とは異なる

図６　DB不正操作時のTeams通知メッセージ事例

4.5　見える化ソフトの開発

収集した品質データを見える化するために，Tableauを使って見える化ソフトを開発した．開発した一部画面について紹介する．

4.5.1　測定データ閲覧画面

収集した品質データの時系列の推移をグラフと表で表示する画面である（図７）[注14]．本画面で品質データの傾向を確認することで，異常データや長期的に見て異常に近づいているデータの把握など，日々の傾向管理に活用することができる．

注14）一部の情報は非公開とし，意図的に消去またはぼかしを入れてある

4.5.2　ヒストグラム閲覧画面

収集した品質データをヒストグラムで表示する画面である（図８）[注15]．本画面で品質データのばらつきを確認することで異常データの把握など，日々の傾向管理に活用することができる．

図７　計測データ閲覧画面

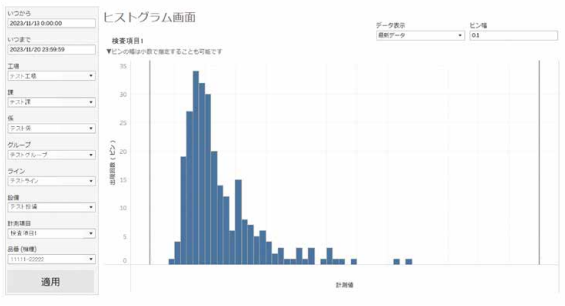

図８　ヒストグラム閲覧画面

注15）一部の情報は非公開とし，意図的に消去またはぼかしを入れてある

4.5.3　トレサビリティ検索画面

製品のシリアルNoに関連するアッセンブリ部品や子部品の情報を一覧表示する画面である（図９）[注16]．問題発生時に該当するシリアルNoから関連する部品を特定でき，影響範囲を特定することが可能である．

図９　トレサビリティ検索画面

注16）表示データはサンプルであり，実際とは異なる

4.6　マスタ設定ソフトの開発

システムを稼働させるために必要なマスタデータの登録や発生したデータ書式エラーの修正を支援するために，OutSystemsを使ってマスタ設定ソフト

を開発した．開発した画面の一部紹介する．

4.6.1　品質検査項目マスタ登録画面

収集する具体的な品質データの検査項目を登録する画面である（図10）[注17]．

図10　品質検査項目マスタ登録画面

新規データ収集開始時や途中で検査項目を変更する場合に本画面上から登録を実施する．

注17）表示データはサンプルであり，実際とは異なる

4.6.2　書式エラーファイル処理画面

4.4節で解説した，データ保存前の書式チェック処理でエラーになったファイルを確認して必要に応じて再アップロードや削除を実施する画面である（図11）[注18]．画面上にはエラーとなったファイルが一覧表示されており，そこから任意のファイルを選択してダウンロードする．そして，そのファイルを確認し，エラーとなっている箇所を修正して再アップロードすることが可能である．マスタ設定ソフトでは画面単位でアクセス権限を設定しており，本画面は特定の管理者のみアクセスすることが可能である．また，操作履歴も収集しているため，万が一不適切な操作があった場合は後から追跡することも可能である．

注18）表示データはサンプルであり，実際とは異なる

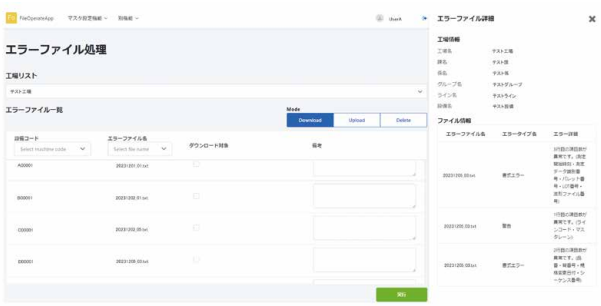

図11　書式エラーファイル処理画面

4.7　データ連携性の向上

従来のシステムでは個々のシステムでマスタ情報を管理していたため，仮に同じラインの情報を収集していたとしてもシステムによってラインの管理単位が異なりシステム間でのデータ連携が困難となっていた．そこで本開発では複数のシステムで使用されるような工場，ライン，設備などのマスタ情報を管理するグローバルマスタテーブルを作成して一元管理する仕組みを構築した（図12）．そうすることで異なるシステムでも共通のラインや設備単位で情報を取得でき，システム間のデータ連携性の向上を図ることできた．

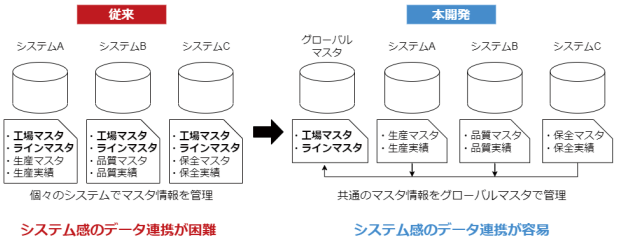

図12　データ連携性の向上

本書執筆の時点では他システムとの連携は実施できていないが，今後は本システムで収集している品質情報だけでなく，製造情報や設備保全情報などと連携することでより多角的なデータ分析が可能となり，品質や生産性向上に寄与することが期待できる．

5　継続的な開発・運用体制に向けた取り組み

5.1　マルチアカウントによるクラウド運用

当社ではこれまで複数のシステムをAWS上に構築して本格的にサービス提供を開始している．利用環境やアクセス権限が異なるシステムを間違いなく運用するために，用途ごとにAWSアカウントを分離し，マルチアカウントによる運用を進めている．

マルチアカウントによる管理・運用のイメージを図13に示す．AWS Organizationsを活用してアカウントを統合的に管理しており，特定アカウントによる最低限必要な設定（セキュリティや監査のためのログ収集など）の漏れを防止している．また，AWS IAM Identity Centerを活用したシングルサインオンも実施しており，各アカウントに対して適切な権限でアクセスできるようにしている．

本システムは品質に関するデータを保持しており，より厳密なアクセス管理が必要になってくるため個別にアカウントを作成している．また開発環境と本番環境でアカウント分けており，各アカウントに対してアクセス権限を分けて設定している．一例を挙

げると開発者は開発環境に対しては読み書きが可能であるが本番環境に対しては書き込みできないようにしている．このようにアカウントを分離して適切に権限管理することで開発者が本番稼働中のシステムを誤って停止してしまうようなトラブルを未然に防止している．

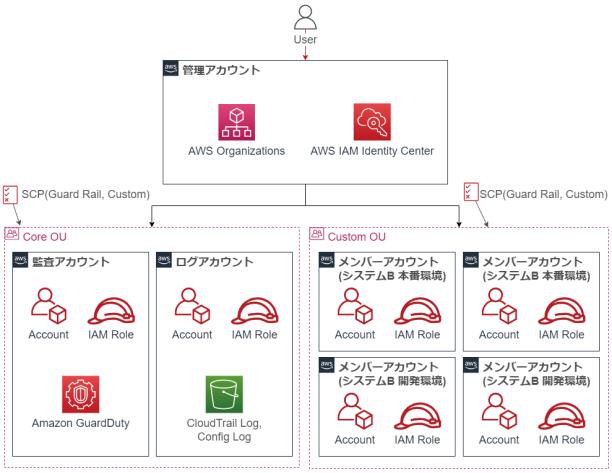

図13　マルチアカウントによる管理・運用のイメージ

5.2　IaCを活用したシステム開発

本システムでは，その後の展開性や保守性を考慮して，IaCを活用したシステム開発を実施している．IaCを活用することでシステム構築における再利用の効率化，手動実行による人的ミスの低減，バージョン管理の容易化，CI/CDの仕組みが可能になりテストから実装までの自動化，などを図ることができる（バージョン管理と実装までの自動化については次節で解説する）．

本開発では，AWS Step FunctionsやAWS Lambdaなどを使って構築したワークフローやアプリケーション（4.4節で解説したデータ保存処理などが該当する）はServerless Frameworkを活用し，DBやネットワーク，セキュリティ関連のインフラはHashiCorp社のTerraformを活用して構築している．

5.3　CI/CDを活用したテストと実装の自動化

IaCを活用することによって容易にバージョン管理を行うことができる．本開発ではソースコードのバージョン管理ツールであるGitLabを使って5.2節で解説したアプリケーションやインフラのバージョン管理を実施している．また，GitLabのCI/CD機能を使ってGitリポジトリにpushされたソースコードに対するテストとAWSクラウドへの適用をあわせて実施している．図14がソースコードを作成してからAWSクラウドに適用されるまでの流れを示している[注19]．

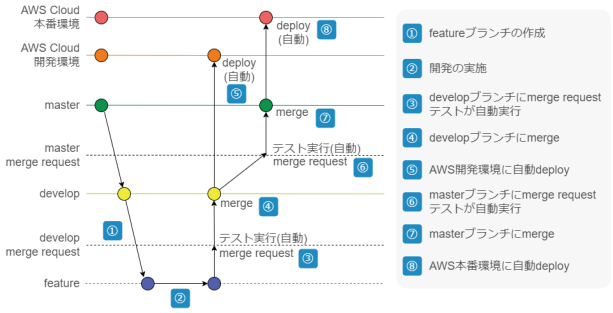

図14　GitLabを使った開発から適用までの流れ

まず，ソースコードを作成する際は，GitLab上に現行版を取り出したfeatureブランチを作成して，そのブランチ内で作業を実施する．作業が完了したらソースコードをfeatureブランチにpushし，その後developブランチに対してマージリクエストを作成する．このときGitLab内でソースコードに対するテストを自動実行して，コード上に問題がないかを自動検証する．その後，管理者による手動チェックを実施して問題なければマージを実行する．そして，マージと同時にAWSクラウドに対する実装タスクが自動実行されて，最終的にAWSクラウドの開発環境に対してソースコードで記述したアプリケーションやインフラが適用されるようになっている．さらにdevelopブランチからmasterブランチに対してマージする際にも同様の流れで自動テストと実装のタスクが実行されて，最終的にはAWSクラウドの本番環境に対してアプリケーションやインフラが適用される．以上のような一連の流れで開発からAWSクラウドへの適用までを実行することで，作業間違いの防止とAWSクラウドへの適用工数の低減を図ることができている．

注19）当社ではGit Flow[注20]をベースにGit管理をしているが，releaseブランチは使用しておらず，developブランチからmasterブランチにマージする運用としている．

注20）Vincent Driessenにより提唱された，効果的なバージョン管理と開発のためのワークフローのひとつ

6　今後の展望

本システムは2023年12月時点で１拠点に対して導入し，本格的な運用を開始している．まだ運用期間が短く大きな成果は得られていないが，オンプレミスサーバの廃止による管理工数の低減や，追加機能による日々の作業工数の低減など，少しずつ成果が見え始めている．今後は残りの５つの拠点に対して水平展開を実施して品質データ管理システムにおけるレガシーシステムからの脱却を完了させる．

また，本システム以外にも当社では多くのレガシーシステムを抱えている．今後は本開発で得られた知見を活用して，これらのシステムに関しても順次クラウドへの移行や再構築を推進していき，全てのシステムにおいてレガシーシステムからの脱却を完了させる予定である．

さらに，レガシーシステムから脱却させるだけではDXを推進することはできない．今後は並行して他システムとのデータ連携やAIによるデータ分析基盤の構築を図ることでデータ活用を活性化させ，DXの推進によって生産性向上や品質向上に貢献していく．

7　結言

本開発によって，障害・災害対策を各段に向上させ，複数システムを連携して多角的な分析が実施でき，費用面と運用管理工数の両方において画期的な低コストで実現できる次世代のシステムを構築することができた．結果として，従来のレガシーシステムで抱えていた多くの課題も解消し，今後のDXを推進するための足がかりをつくることもできている．しかし，レガシーシステムから脱却することはDXを推進するための前準備に過ぎないので，今後も継続してデータ活用するための基盤構築や機能開発を進めていくことでDXの推進を目指していく．

最後に，本開発にあたり多大なるご支援，ご協力を頂いた関係部署の方々に，この場をお借りして熱く御礼を申し上げる．

参　考　文　献

1）IPA：DX白書2023，p. 9，（2023年 2 月）．
2）内藤：KYBの生産領域におけるAI×IoTの取り組み，KYB技報第60号，（2020年 4 月）．
3）古川・井指：設備予知保全システムの開発，KYB技報第63号，（2021年10月）
4）大内田・宮内・提箸：SA要素開発へのAI技術活用，カヤバ技報第66号，（2023年 4 月）
5）宮内・井指・瀧野：デジタル人材育成の取組み，カヤバ技報第66号，（2023年 4 月）
6）Amazon Web Services, Inc：クラウド移行とは何ですか?, https://aws.amazon.com/jp/what-is/cloud-migration/,（参照2023年12月12日）

著　者

古川　　輝

2005年入社．デジタル変革推進本部システム開発室．
クラウド活用したIoTプラットフォームの構築と生産システムの開発に従事