

HITACHI

金融業界での超短期システム開発を実現した成功ポイント

日本における新たな市場の 短期創生への挑戦

日本国内の
IPX総研

市場の短期創生に
挑戦したのか？

株式会社 日立製作所

適切な開発プロセスとアプリケーションアーキテクチャーで 「カーボン・クレジット市場システム」の開発プロジェクトを推進

株式会社日立製作所（以下、「日立」といいます）と日本取引所グループ（以下、「JPX」といいます）は、経済産業省が2022年9月22日に実証を開始したカーボン・クレジット市場の試行取引の取引システムとして、JPX傘下の株式会社JPX総研（以下、「JPX総研」といいます）と共同で「カーボン・クレジット市場システム」を開発しました。「カーボン・クレジット市場システム」開発プロジェクトは、JPX総研と日立にとって大きな挑戦でした。実質的な開発期間が約3か月半という超短納期のスケジュールの中で、スピードと生産性を最大化するシステム開発が求められたのです。



この課題を解決するために、日立は3つの施策によって、アプローチしました。

1 点目は、システム全体を独立性の高い機能単位に分割しそれぞれを開発した後に、順次統合していく「インクリメンタル開発」を実施しました。2 点目は、インクリメンタル開発を円滑に進めるためにシステムを小さなサービスとして構成する「マイクロサービス」というアーキテクチャーを採用しました。3 点目として、限られた開発期間に対応できるよう充実した開発体制も整えました。これらのことが、プロジェクトの成功の技術的要因であり、本書により成功ポイントを解説いたします。

Contents

- 1 新たな実証事業への挑戦の背景
- 2 開発に求められた3つの技術要件
 - 成功ポイント1：適切なアーキテクチャーと開発プロセスの構想力
 - 成功ポイント2：顧客要望に対応したクラウド技術の活用力
 - 成功ポイント3：短期開発を実現するアプリケーション開発力
- 3 早期稼働を実現するプロジェクト体制と日立のソリューション
- 4 まとめ

1 新たな実証事業への挑戦の背景

「カーボン・クレジット市場システム」実証事業では、開発開始が2022年5月、実証開始が同年9月と決まっていたため、早期稼働に向けて超短納期の達成が必要となりました。また、実証事業という位置付けから、事業費を抑えつつ、ハードウェアなどの固定資産を抱え込まない柔軟な推進も必要であったため、パブリッククラウドでのSaaS/PaaSなどを積極的に活用したシステム構築が求められました。

その一方で、実証事業の中にあるプロジェクト由来のクレジットは、実際の取引結果に基づいた資金決済やJ-クレジットの登録簿の振り替えが発生するため、実証といえども、本格稼働に近い品質を確保しなければなりません。

こうした背景から、「カーボン・クレジット市場システム」では、実証段階では機能・非機能要件ともに必要最小限で開始するものの、本格稼働に備えて機能追加・変更に対応できる開発の進め方が不可欠でした。まとめると、以下の課題を解決する必要がありました。

- ・ 超短納期の達成
- ・ スモールスタートと拡張性
- ・ 機能追加・修正に備えた柔軟性の確保

これらすべてを対応するためには、

- ・ リアリスティックな要件の見極めと調整力
- ・ 多岐にわたる関係者の円滑で密なコミュニケーション
- ・ 適切な設計・開発技術の適用

が必要でした。

これらを満たせる「カーボン・クレジット市場システム」の開発プランをJPX総研と日立で取りまとめ、共同開発が始まりました。

本書では、「適切な設計・開発技術の適用」に絞って、解説します。

2 開発に求められた3つの技術要件

「カーボン・クレジット市場システム」の開発を推進していくうえで、次の3つの技術要件が求められました。

①アプリケーション可搬性の確保

短納期で実証事業の開始を実現するためには、すでに構築・設定が終わっていたJPM総研のパブリッククラウド基盤の活用が前提になっていました。しかし、実証事業を経て創生されていく市場やビジネスのためには、その後の本格的なシステム稼働の際にクラウド基盤の変更も想定しなければなりません。このことを踏まえ、「カーボン・クレジット市場システム」の側で基盤の変更に備えた柔軟性を確保する必要がありました。

②開発容易性の確保

実証事業を進める中での機能改修や、商品性の異なるカーボン・クレジット商品の追加、また実証結果を反映した本格システム開発など、後継プロジェクトでのサービス追加や機能追加の容易性を考慮した設計が必要でした。

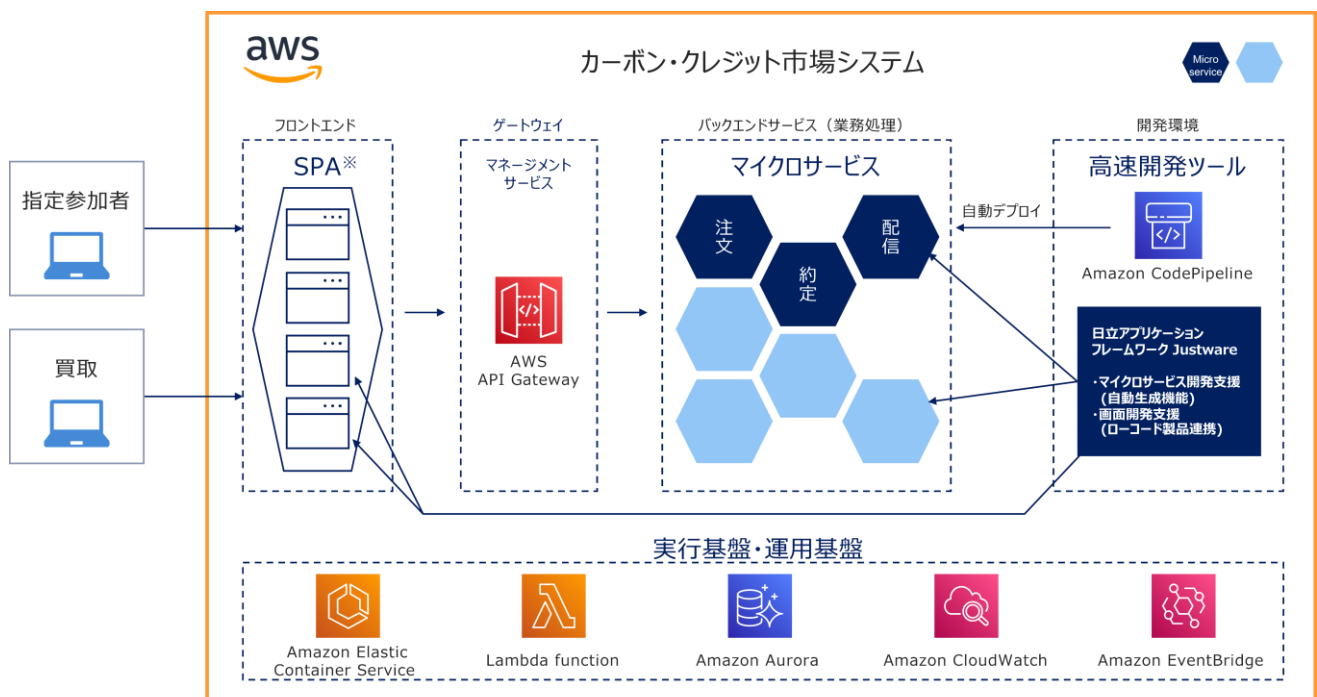
③高速リリースサイクルの対応

まったく新しい市場の短期創生という性質上、仕様を机上で完全に固めてから開発するという進め方は選べません。売買という根幹となる業務構造は変わらないものの、管理する項目や業務フロー、市場が開かれてる時間、およびユーザーエクスペリエンスなど、市場を運営しながら対応していく必要性が予見されました。

こうした3つの要件をクリアするために、日立はクラウドとアプリケーション技術のプロフェッショナルを開発チームに組み入れ、適切な開発プロセスとアーキテクチャーの適用を推進し、以下のように対応しました。

- コンテナ技術を活用した可搬性の確保
- 機能を分解して同時並行で進めるマイクロサービスアーキテクチャーを採用
- 並行開発に適した開発プロセスとしてインクリメンタル方式を採用するとともに、高い生産性を実現するため適切な高速開発ツールを適用

「カーボン・クレジット市場システム」の全体構成は、次のようになります。



※SPA : Single Page Applicationの略。単一のページで動作し、ユーザー体験を向上させる技術。

これらの取り組みによって、短納期でのカットオーバー(Day1)、実証事業中の変更要請に対応した推進を、無事に実現しました。

この開発プロジェクトにおける、日立の技術的な成功ポイントを振り返ります。

成功ポイント 1

適切なアーキテクチャーと開発プロセスの構想力

短納期と変更に強い開発を実現するために、適切な開発プロセスとアーキテクチャーを選定する必要がありました。短納期に対応するためには開発対象を分解して並行開発することで期間短縮する方法が効果的ですが、単に機能単位に並行で開発するだけでは超短納期を実現する施策として不十分だと判断しました。

そこで、従来よりもシームレスに並行開発を進めるため、マイクロサービスアーキテクチャーを軸にした開発プロセスにはインクリメンタル方式を採用しました。

開発プロセスとして採用したインクリメンタル方式は、決めたところから開発に着手し、独立して完成させていく方式です。短納期開発が求められる場合、一般的にはアジャイルプロセスの適用が想起されますが、今回の超短期開発ではアジャイルプロセスが得意とする試行錯誤を繰り返しては、納期に間に合わないことが想定されました。また、ウォーターフォールのように全体を詳細まで決めきってからの開発着手では、納期達成は難しいことが想定されました。インクリメンタル方式であれば、有識者が一筆書きで構成した全体の骨組みを個々の業務（機能）ブロックに分けることで、試行錯誤するべきブロックと一気につくり込んでいくブロックで、独立した開発を推進できます。

本システムで採用したマイクロサービスアーキテクチャーは、凝集性を高め、疎結合化し、システムをブロック単位で独立させて扱いやすくします。コスト効率が最大化するのはモノリシックなアーキテクチャーなのですが、ウォーターフォールで開発されるのが一般的です。可能な限り共通化するよう突き詰めて進めるやり方なので、設計工程に十分な時間が必要になるうえ、共通化する部位に対して修正が発生した場合、その部位に依存しているほかの部位への影響の有無を調査するなどにも必要になります。

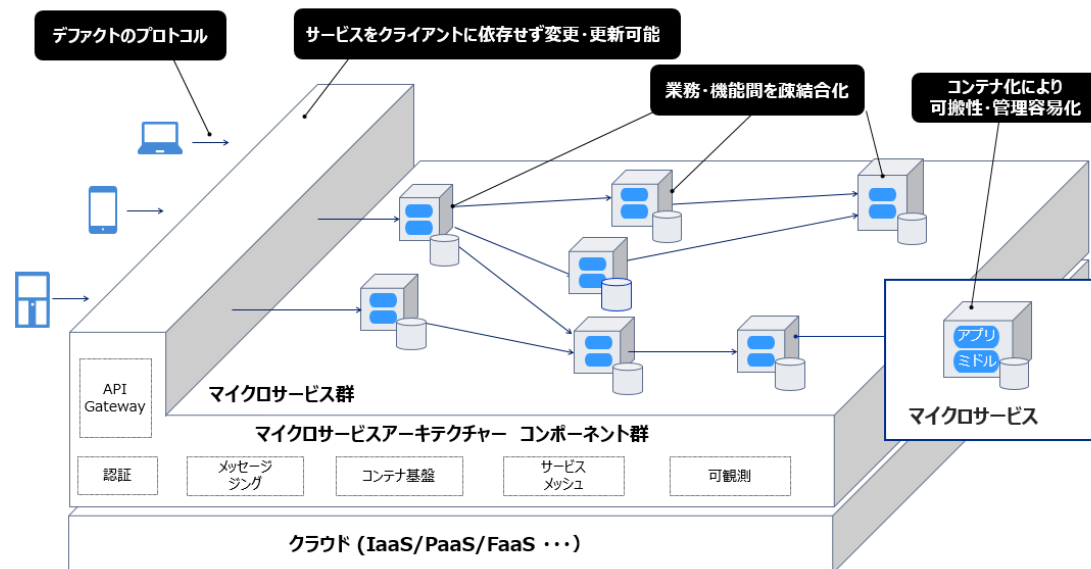
「カーボン・クレジット市場システム」開発プロジェクトでは、機能ブロック単位で詳細をすり合わせていると時間が不足し、共通化をしながら進めるとスピードが落ちる可能性もありました。そこで、可能な限り機能間での影響を小さくできるマイクロサービスアーキテクチャーを採用したのです。

マイクロサービスアーキテクチャーによる開発のメリット

マイクロサービスでは、APIによってバックエンド（業務機能）とフロントエンド（画面）を分離することで、両者の独立した開発とデプロイを実現できます。さらには、API管理層で、サービス構成をクライアントから隠蔽し、疎結合化します。そして、サービスを分割することで、業務機能間を疎結合化し、各業務機能で使う個々のサービスで独立した改修・再利用・スケールアウトを可能にします。加えて、マイクロサービス単位でコンテナ技術を活用し、インフラからの疎結合化を図り、可搬性やデプロイ（SI作業）の自動化を促進できます。

マイクロサービスアーキテクチャーは、現時点でもっとも疎結合化したシステムの分割を可能にし、デプロイ頻度（機能追加／修正の頻度）の高まりに対応して、改修のしやすさを実現します。

※API：Application Programming Interfaceの略。接続先のOSを呼び出すことや、あるソフトウェアの機能を呼び出す仕組み。



成功ポイント2

顧客要望に対応したクラウド技術の活用力

「カーボン・クレジット市場システム」の開発のような短納期とスモールスタートを実現するためには、パブリッククラウドの活用が非常に有効な施策となります。そこで、JPX総研でアマゾン ウェブ サービス(AWS)上に構築済みのクラウド基盤を活用する方針を立案しました。これにより、システム基盤の構築期間が大幅に短縮できる見通しが立ちました。

一方で、お客さまが構築済みのクラウド基盤を活用するためには、単なるクラウドベンダーのサービスの知識だけでなく、お客さまごとの設計ポリシーやセキュリティなど、運用上の制限事項を前提条件にした使いこなしが必要となります。

システムインテグレーターとしての日立に求められたのは、お客さまが構築済みのクラウド基盤の設計を理解し、その制限の中での個々のシステム設計を積み上げる技術力です。これまで多くの実績を有する日立は、本プロジェクトでもクラウド技術の活用力を発揮することで、システム構築をスムーズに進めました。加えて多岐にわたるコンテナ基盤（Amazon Elastic Container Service ほか）、サーバーレス機能（AWS Lambda）、インフラ構成管理機能(Terraform)、クラウドにビルドインされたデプロイ自動化機能(AWS CodePipeline)などを高レベルで使いこなし、素早い開発を実践しました。

成功ポイント3

短期開発を実現するアプリケーション開発力

開発期間を短縮するためには、アプリケーション開発作業の生産性を向上させる適切な開発ツール選びも重要でした。開発ツールを選定するにあたっては、以下の観点で検討しました。

1. アプリケーションの可搬性（さまざまな環境で実行できる）

コーディング作業がないノーコード開発ツールの採用も視野に入れましたが、実行基盤がノーコード製品やサービスにロックインされる傾向が高いため、これを回避。生産性向上を図るために採用したのは、ローコードのツールの活用でした。

2. フロントエンドとバックエンドでの適切な開発方式

フロントエンド（UI）とバックエンド（業務処理）を同時並行で開発でき、かつそれぞれのサービス提供に適した開発ツールとして、バックエンド開発には、マイクロサービスではデファクトスタンダードであるOSS（Open Source Software）のSpring Framework*をベースにローコードを実現している日立アプリケーションフレームワーク「Justware」（以下、「Justware」といいます）を適用しました。そしてフロントエンド開発には、GUIで見たままに画面がデザインでき、画面仕様をユーザーと詰めながら開発できるローコードツール「Nexacro Beyond」を採用し、素早い仕様変更の取り込みを実現しました。

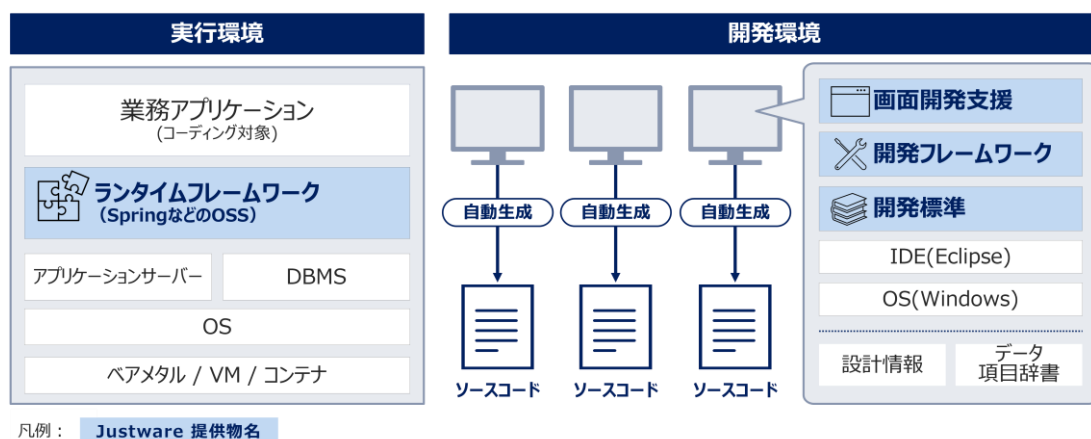
これらの採用により、高い生産性を実現できました。

*Spring Framework

Javaプラットフォームを対象とするアプリケーションフレームワークで、Javaベースのエンタープライズアプリケーション向けの包括的なプログラミングおよび構成モデルを提供します。

Justwareの特長

- ① アプリケーションの実行環境が標準的なアプリケーションサーバーで機能するOSSの「Spring」ベースは、メガクラウドベンダー / オンプレミスを選びません。
- ② エンタープライズアプリケーションで必要な非機能要件まで意識した自動生成が可能で、かつ生成したソースコードはJava技術者による可読性が高く、ロックイン要素も排除されています。
- ③ バックエンドはデファクトスタンダードなREST APIベースで機能を公開可能で、フロントエンドの実装技術やツールを自由に選ぶことができます。



3 早期稼働を実現するプロジェクト体制と日立のソリューション

日立は、お客さまと深く連携して個々のプロジェクトを推進するフロント系チームと、製品や技術を軸に専門的なサービスを提供するテクニカル系チームが連携してこのプロジェクトを推進しました。取引業務の業務知識を理解したフロント系チームメンバーがお客さまと協力し円滑な推進を実現したことに加え、システム開発の技術面としては、以下のソリューション、サービス、製品を活用して早期稼働の一助としました。

1. アプリケーションアーキテクチャーの設計段階から支援する日立の「マイクロサービステクニカルソリューション」

超短納期でも提供するサービスの完成度が求められた「カーボン・クレジット市場システム」開発プロジェクトは、JPX総研にとっても日立にとっても、新たな挑戦でした。そこで、「マイクロサービステクニカルソリューション」を提供しているメンバーがプロジェクトに参画し、綿密な検討を実施しました。

特に、マイクロサービスの分割と開発方式に関しては、必要最低限の仕様に絞りながらも、そのメリットを享受できるように、プロジェクトのディスカッションをガイドしながら設計を推進。その成果によって、新規技術の採用リスクを低減しつつ、目標を達成できました。



2. お客さまのクラウドニーズに合わせた推進を実現する日立のクラウドエンジニアリング

AWSやMicrosoft Azure、Google Cloud などのハイパースケーラーは、独自仕様のクラウドサービス・機能を多数展開しています。この、独自に提供しているサービス全般をキャッチアップできているエンジニアは、そう多くはいません。

そこで、日立ではAPN AWS Top Engineers選出者をはじめとする国内トップクラスのエンジニア後方支援としてプロジェクトに参画させる準備をしています。パブリッククラウド専門のエンジニアリングが、個々のお客さまの思想・戦略を理解した顧客アカウント別のフロントサービスのメンバーと連携する開発体制の提供で、よりご要望に応じた柔軟な対応を実現しています。日立は、さまざまなハイパースケーラーのクラウドサービスやその使いこなしに関して、お客さまの戦略と個別技術の両面に対応していきます。



3. 変化・変更に強く開発ニーズに適合した開発を実現するフレームワーク「Justware」

高い生産性を実現し、早期稼働を可能にしたフレームワーク「Justware」には、さまざまな施策が凝縮されています。例えば、必要なOSSを動作確認済みの状態で同梱して提供しているので、すぐに開発を始められます。また、開発者が迅速に開発に着手できるよう、標準的に必要と想定される設計書フォーマットや基準類もパッケージングしています。標準的なJava技術者であれば、翌々日から使いこなして、ツールのメリットを享受できるように構成しています。さらに、短納期の場合に、非常に難易度が高くなる品質確保の面でもメリットを発揮します。ログの出力やエラーハンドリングなど、品質の中核となる非機能要件を満たすための機能を実現するソースコードが自動で出力されるので、機能要件の実現に開発者がフォーカスできます。

— 関連情報

<https://www.hitachi.co.jp/products/it/appsvdiv/service/microservices-tech/>

<https://www.hitachi.co.jp/products/it/appsvdiv/service/justware/>

<https://www.hitachi.co.jp/products/it/harmonious/cloud/articles/microservices/index.html>

4 まとめ

「カーボン・クレジット市場システム」のように、新しい市場やビジネスの創生という取り組みに挑む場合、どのようなプラットフォームやアーキテクチャーを採用するのかはもちろん、信頼できる開発チームと密にコミュニケーションをとりながら「ワンチーム」としてプロジェクトを推進できるかが、成否を分けるといっても過言ではありません。

また今回の開発でAWSを最大限に活用したように、お客さまが構築済みのクラウドインフラの思想を素早く理解して使いこなすためには、システムを構築するインテグレーターの役割が重要になります。

日立は今後も、さまざまな挑戦で新たな市場やビジネス創生に取り組まれているお客さまを、課題・要件を見極めるためのコンサルティングから、目的に合致した効率的な開発、運用まで、業務、技術に精通したプロフェッショナルで構成したチームが、お客さまとともに社会インフラを支えるシステム開発を実現してまいります。

プロジェクトの成功ポイントと日立のソリューション



— 他社商標 —

- ・Amazon Web Services、AWS、Powered by AWS ロゴ、その他 AWS商標は、Amazon.com, Inc. またはその関連会社の商標です。
- ・Microsoft Azureは、マイクロソフト 企業グループの商標です。
- ・Oracle®、Java及びMySQLは、Oracle、その子会社及び関連会社の米国及びその他の国における登録商標です。
- ・その他、記載の製品名、会社名は、各社の商標または登録商標です。

— 本書の記載情報 —

※本書は、2023年4月現在の製品・サービス・ソリューションの内容に基づいて記載したものです。
予告なしに変更され、検索日と情報が異なる可能性もありますので、あらかじめご了承ください。
また、本書に記載した対応状況は、ISO規格や各種ガイドラインなどの改定により、予告なく変更になることがあります。

お問い合わせは下記へ

■マイクロサービステクニカルソリューション 情報サイト

<https://www.hitachi.co.jp/products/it/appsdiv/service/microservices-tech/>