Customer CASE STUDY

databricks

多様なデータソースを分析・可視化し、次世代交通システムMaasの本格的な普及に取り組む。

MaaS Tech Japan

Maasの社会実装を通して、交通分野の課題を解決。

株式会社MaaS Tech Japan（マーステックジャパン）は、2018年に創業された、MaaS（モビリティ・アズ・ア・サービス）を軸に事業展開をしている企業である。当該分野において、日本マイクロソフト、東京メトロ、東京海上日動火災保険、LINEなど、様々な企業や組織とパートナーシップを組みながらソリューション開発を進めており、日本経済新聞などの多くの主要メディアを通じて、MaaSの社会実装をリードするスタートアップとして、注目を集めている。

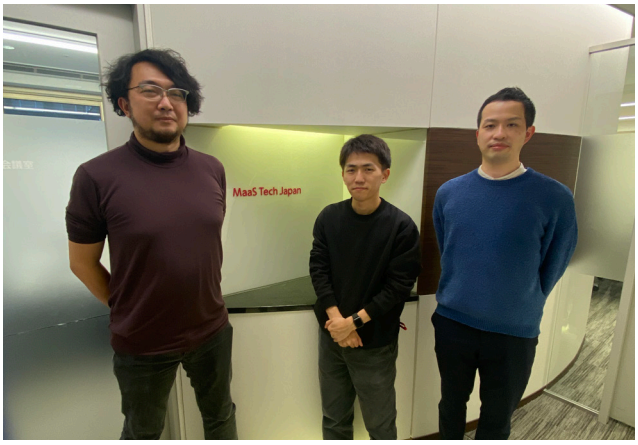
MaaSが注目される背景は、交通分野に様々な課題が山積していることにある。「都市エリア」で人口密集への対処やデジタルによる都市競争力向上、「地方エリア」では、人口減少に伴う交通インフラの維持や運転手不足、「観光地エリア」では、回遊・消費の創出による観光経済の維持・拡大がある。また、昨今の新型コロナウィルスの蔓延による、移動経済と安心安全の両立や環境への影響なども大きな課題となっている。

これまでの、都市・交通・道路計画は、日本では1960年代から踏襲された手法で行われていたが、昨今のスマートフォンの普及により、データ収集において、パラダイムシフトが起こりつつある。これにより、収集可能なデータの質・量・鮮度が大幅に改善し、より多くの、よりリアルタイムな、よりパーソナルなデータの分析が可能となる。

あらゆる交通情報をリアルタイムでユーザーに。

MaaSデータ統合基盤「TralSARE」では、交通事業者や自治体、外部データ(携帯キャリアやモバイルアプリケーション事業者などからのデータ)を収集し、分析・可視化をした上でクライアントへ提言やインサイトを提供している。将来的には、BtoBtoCという形で、シミュレーションなどを用いた情報(迂回ルートの提案など)を、リアルタイムでユーザーに提供することを計画している。

「TralSARE」の分析対象となるデータや接続システムの増加に際し、パフォーマンス、スケーラビリティ、データソースの多様性に関する課題に直面していた。従前のアーキテクチャー(図1)では、処理自体の信頼性担保や必要なパフォーマンスを実現することが難しく、



（左）取締役CTO
渡邊　徹志 氏

（中央）技術開発
中村　優 氏

（右）技術開発　シニアマネージャー
荒居　孝紀 氏

データ加工や更新処理もオペレーションも個別システムを用いた煩雑で工数を割く必要があった。分析対象のデータ種別は、鉄道、バス、航空、船舶などの交通データに加え、地図、イベント、地物情報などの非交通データが存在する。フォーマットに関しては、GTFS-RT、GTFS-JP、ODPTやそれらに加えて事業者個別データがあり、形式に関しても、JSON、Protocol Buffers、CSV、PDF、XLSとさまざまである。プラットフォームのロードマップを考えると、よりスケーラブルなアーキテクチャに変更する必要性も感じ、Databricksのレイクハウスプラットフォームを採用した。

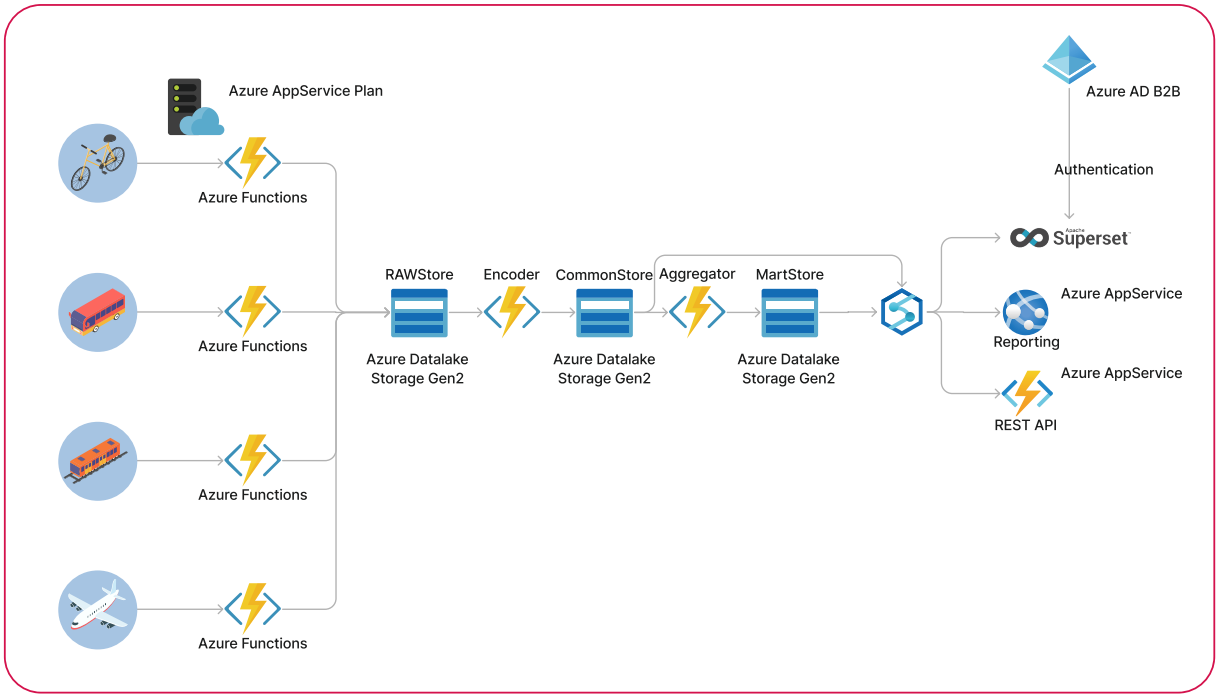
採用の決め手は、オープンなデータプラットフォーム。

Databricksは、様々なデータ種別や形式に対応することができ、パフォーマンスも期待どおり得られている。また、Databricksを採用した大きな理由の一つとして、Databricksのレイクハウスプラットフォームが、オープンであることだ。交通という公共データを扱っているビジネスの特性上、データプラットフォームのアーキテクチャーはオープンであるべきと考えており、他ベンダーが提供するベンダーロックインの環境は避けたいという思いが強かった。

Databricks導入後の具体的なアーキテクチャー(図2)としては、様々なデータソースをAzure Data Lake Storageに保存し、Databricksのクラスター内で立ち上げたdbtでデータ変換処理を

実行し、その結果をオープン形式であるDeltalakeのテーブルとして保存する。さらにDatabricks Unity Catalogのアクセス権限管理の下で、Databricks SQLでSQLクエリを実行し、ユーザーへアウトプットを可視化する。データ収集、ETL/ELT、データ分析、アクセス権限管理、可視化という一連の流れが、シームレスかつシングルプラットフォームで実現する開発体験にも非常に満足している。今後、より多く第三者機関とのデータの共有することを想定しており、Databricks Delta Sharingによる、セキュアかつ柔軟なデータ共有の仕組みにも期待をしている。

(図1)Databricks 導入前のアーキテクチャー



(図2)Databricks 導入後のアーキテクチャー