モビリティサービスを支えるプラットフォーム技術*

Survey: Connected Platform Technologies for Mobility Services

小泉 清一 Seiichi KOIZUMI
栗原 浩 Hiroshi KURIHARA

Mobility services, such as ride hailing, car sharing, are provides new mobility values. In this paper, we survey technical features of mobility service platform which enhances mobility service development. We describe functionalities of mobility service platform, technical trends and future works. We also introduce Denso's mobility service platform architecture.

Key words :
Mobility Service Platform, Connected Platform, Vehicle IoT, Digital Twin, Device Shadow, Data Lake, Edge Computing

1. はじめに

近年，Uber のようなライドシェアリング，タイムズカープラスのようなカーシェアリング，テレマティクス保険，車両の遠隔診断，駐車場シェアリング，といった，ICT を活用して車両等の交通手段による新しい移動価値を提供するモビリティサービスの広まりが目覚ましい．このようなモビリティサービスは，ICT を活用して車両等の交通手段による移動機能を有機的に結び付け，さらに道路の交通情報，鉄道やバスの運行情報，タクシーの位置情報，などの移動・交通に関する大規模なデータを活用して，新しい移動価値をタイムリーに提供することを実現している．

そのようなモビリティサービスを実現する仕組みとして，Fig. 1 に示すようなモビリティサービスプラットフォームが提唱されている．モノを活用したサービスの開発を促進するプラットフォームとしては，IoT プラットフォームがあるが，モビリティサービスプラットフォームは，その IoT 基盤が活用する機能として，車両等の移動するエッジコンピュータ，通信キャリアやサービス提供者が設置するフォグコンピュータ，V2x 通信，車両・公共交通機関などの移動・配送手段と，交通関連の外部情報源が含まれている点が異なっている．

本稿では，第 2 章にてモビリティサービスプラットフォームを構成する技術要素を解説し，3 章にてモビリティサービスプラットフォームを強化する技術動向について紹介する．第 4 章では，デンソーの取り組みについて紹介し，第 5 章では今後拡張すべき技術領域について考察する．

＊（公社）自動車技術会の了承を得て「自動車技術 2019 vol 73 P82 ～ P87」から一部加筆修正して転載

2. モビリティサービスプラットフォームとは

デバイスとデータを活用して新たなサービスを提供するプラットフォームとしては，例えば IEEE P2413 [1]，ITU-T Y.2060 [2] 等で定義された IoT プラットフォーム [3] がある．このプラットフォームは，センサー・デバイスからデータをスムースに収集し，データに基づいて状況を把握し，その状況に合わせてアプリケーションを動かすことで，サービス開発の促進を実現している．このとき，移動特性，ヘテロデータ特性，電源特性といった車両固有の特性を備えているため，それらの特性を考慮して車両等を活用したサービスを実装するためのモビリティサービスプラットフォームとして，Vehicular cloud computing [4] や，Cloud computing in the automotive domain [5] が提唱されている．これらの活動を俯瞰し，本稿ではモビリティサービスプラットフォームのアーキテクチャを Fig. 1 のように定義している．

モビリティサービスプラットフォームは，大別して 8 種のレイヤー・機能ブロックから構成されている．各節では，それぞれについて解説する．

2.1 Edge Gateway layer

このレイヤーは，各種車両にビルトイン・または後付けされた Edge Gateway によって構成され，車両のセンサー・デバイス・アクチュエーターのデータを収集・配信，データの中間処理を行うゲートウェイ機能と，車両とクラウド・他の車両をつなぐ通信機能を有する．

IoT プラットフォームのゲートウェイと異なり，電波状況やバッテリー残量の変化といった車両特有の事象 [6] が発生するため，これらを考慮したデータ処理を行う必要がある．

2.2 Connectivity and Fog Layer

このレイヤーは，携帯網，Wi-Fi, LPWA, DSRC といった各種通信手段を活用し，車車間・路車間・車／クラウド間等 (V2X) の接続機能を提供する．また，Software-defined Networking (SDN)/Network Function Virtualization (NFV) といった，ネットワークの経路・機能をソフトウェア制御する技術により，データの種類ごとの帯域制御，ネットワーク攻撃等を防ぐファイアウォール，ワームやトロイの木馬を防ぐ Intrusion Detection System/Intrusion Prevention System (IDS/

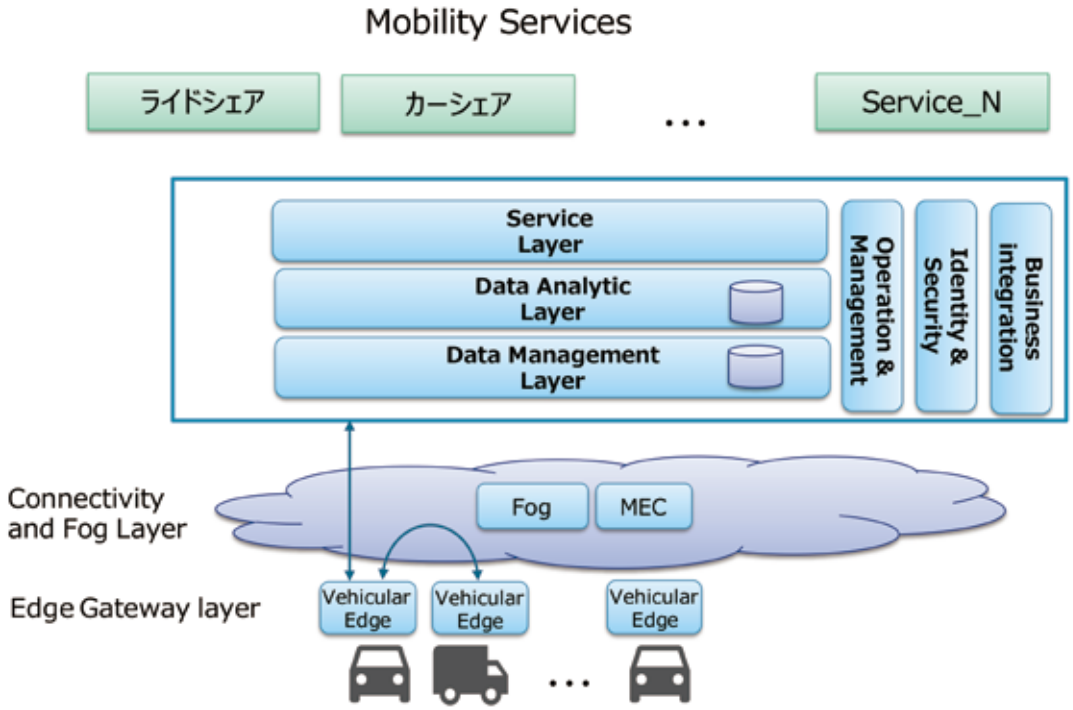

Fig. 1 Mobility Service Framework

IPS)，といったネットワーク機能[7]を活用し，重要なデータの保護や，優先制御などを行うことができる．

また，ネットワーク経路上に設置されたフォグコンピュータを用いてデータの中間処理を行うことで，データ収集量の削減や，データ配信の効率化を行うことが可能である．そのようなフォグコンピュータとしては，通信キャリアが提供する Mobile Edge Computing (MEC) や，ロードサイド・駐車場等に設置されたカスタマーフォグがある[8]．

2.3 Data Management Layer

このレイヤーでは，車両から収集した情報を蓄積し，多様なモビリティサービスにて利用可能なデータストアを提供する．車両の位置情報や経路情報，ドライブレコーダーの映像，地図情報，渋滞・工事・事故などの交通情報，といった多様な情報をデータレイクと称されるデータストアに蓄積し，検索可能とする．

IoT プラットフォームとは異なり，道路標識情報などの静的データ，車両の速度・向きなどの動的データ，渋滞予測などの予測データ，道路地図などの時空間データ，といった異種のデータが混在している．

データを蓄積する際には，どの車両のどのデバイスから発生したデータであるかの把握と，その車両やデバイスの状態（故障や電源 OFF など）も把握する必要があるため，車両やデバイスの構成管理機能も本レイヤーに含まれる．

2.4 Data Analytics Layer

このレイヤーでは，モビリティサービスにて必要な情報を Data Management Layer から取り出すためのデータ処理を行う．例えば，あるドライバーに関する統計情報を算出したり，ある車両に関する走行経路マップを生成したりといった処理を実行し，モビリティサービス開発を支援する．

前節で言及した通り，異種データに跨った解析が必要となる点が，IoT プラットフォームとは異なっている．本レイヤーにてデータ処理を行うことで，データレイク内のデータをサービス毎に適したデータ（e.g. 車両でクラスタリングした統計データ，ドライバー事の集計データ，など）とし，サービス開発を支援する．

2.5 Service Layer

このレイヤーでは，モビリティサービスの実装を支援し，動作環境を提供する．サービス指向アーキテクチャ（SOA）の機能単位での疎結合・再利用という概念を取り入れつつ，サービスの機能を実装しやすくする仕組み（e.g. Ruby on Rails, Python Django, など）や，機能を連結するコレオグラフィ・オーケストレータを活用して，サービスを実装・提供する．

2.6 Operation and Management

この機能ブロックでは，プラットフォーム，及びアプリケーションの開発と運用を一体化したサイクル（DevOps）を実現するために，監視・復旧機能，インシデント・障害管理，といった機能を提供する．モビリティサービスでは，車両からソフトウェアアプリケーションまでの End to End でのサービス品質管理が必要で，本機能ブロックはその品質維持の要となっている．

2.7 Identity and Security

この機能ブロックでは，プラットフォーム，及びアプリケーションのユーザ認証・認可・権限管理を一元的に行い，全レイヤーに渡ったセキュリティ対策機能を提供する．

2.8 Business Integration

この機能ブロックでは，既存の ICT システムとの連携を行う．例えば，課金システムとの連携，CRM システムとの顧客情報連携，ERP システムとの発注情報連携，といった機能を提供する．

3. モビリティサービスプラットフォームを強化する技術動向

モビリティサービスをより開発しやすく，またはよりユーザニーズに合ったものにするために，先行するインターネットサービスの実装技術のモビリティサービスプラットフォームへの取り込みが進みつつある．特に，クラウドネイティブと称されるクラウドの技術を活用してサービス開発をしやすくするソフトウェア

設計・開発・運用アプローチが注目されている．

また，モビリティサービスプラットフォーム関連の研究動向としては，モビリティ特有の問題である車両等からのデータ量の増加やデータの異種性の広がり，といった点に対処する研究開発が進められている．

3.1 クラウドネイティブ

クラウドネイティブとは，クラウドの利点を活用してサービスを開発するソフトウェア設計・開発・運用アプローチを指す．クラウドサービスは，ソフトウェア開発手法・開発モデルを活用することで多大な設備投資や人件費の必要性を排除し，Time to market 重視でサービスを投入することで，サービス価値の向上を実現している．主に 4 つの特徴を備えている [9]．

- マイクロサービス：ソフトウェアの開発モデルの 1 つで，独立して配置可能な最小限の機能単位の組み合わせでアプリケーションを構築する．例えば，EC サイト（電子商店）の機能を注文・決済・メール通知という機能単位で分解してマイクロサービスとし，マイクロサービスの間は REST API や RPC という疎結合インタフェースで連携することで，アプリケーションの機能を提供する．
  このように機能単位で分割することにより，例えば決済機能のみを QR コード決済対応に拡張する，といった柔軟性を獲得できるため，ユーザニーズの変化等に対応しやすいという利点がある．その反面として，細かいソフトウェアを大量に管理することになるため，連携の整合性や，ネットワークの制御といった複雑性が増加するため，ソフトウェア開発は難化する．
- コンテナ型仮想化：仮想マシン（VM）は，CPU/RAM/HDD といったハードウェアを仮想化するために，どうしてもコンピュータの処理負荷（オーバーヘッド）が大きくなる．それに対し，コンピュータ上の OS を共有し，その上にコンテナと呼ばれる「隔離されたアプリケーション実行環境」を用意したものがコンテナ型仮想化である．オーバーヘッドが小さいため，VM より多くのアプリケーションを稼働できるという利点があるが，OS の種類が限定されるという制約もある．
  マイクロサービスに合った，粒度の小さなソフトウェアを動かすのに適している．
- DevOps：開発（Development）と運用（Operations）を組み合わせた名称で，開発と運用の垣根を無くし，互いに協力しあうソフトウェア開発手法．ソフトウェアの開発を行う「クリエイト」，機能・非機能要件を試験する「検証」，配置するために必要なデータや設定ファイルをまとめる「パッケージ」，本番稼働させる「リリース」，安定して稼働するための設定と管理を行う「コンフィグレーション」，性能などの品質監視を行う「モニター」，ユーザからのフィードバック等を元に要件を練り直す「計画」，という 7 つのフェーズにて構成されている．ビジネスの価値をエンドユーザに届ける，という共通の目標に向かって開発と運用が協力することで，ビジネスの価値をより高めることが期待されている．
- CI/CD：Continuous Integration/Continuous Delivery（継続的インテグレーション／継続的デリバリー）の略称で，ソフトウェアの変更を常にテストして自動で本番環境にリリースする，ソフトウェア開発プロセスの自動化手法．CI では，開発者が変更後のコード変更をリポジトリにマージすると，その都度自動化されたビルドとテストを実行する．CD では，テスト環境またはステージング環境にデプロイして，システムテストなどを行い，本番環境へのリリースを行う．この CI/CD により，開発者の手動作業削減や，バグ修正の容易化が可能となる．

3.1.1 クラウドネイティブを取り入れたモビリティサービス

BMW は，DriveNow, ParkNow, ChargeNow といったモビリティサービスを提供しており，RedHat Summit 2017 では，クラウドネイティブを取り入れたモビリティサービス開発を発表している [10]．2017 年時点で，300 種類のマイクロサービス開発，1100 件の CI/CD ジョブが進行中とされ，前節の 4 つの特徴を活用した開発が進められている．

また，Ford Motor Company でも同様に 4 つの特徴を生かした開発を進めており，特に “Cloud Native Reference Application” というリファレンスアーキテクチャを強みとしている [11]．このリファレンスでは，Service Registry, Circuit Breaker といったよく使われる機能のベストプラクティスを共有することで，モビリティサービスの質の向上を目指している．

3.2 データ分散型アーキテクチャ

データの生成量は，計算能力やストレージ容量の向上を上回り増加しているが，データの総量に対し，目的に適したデータの量は必ずしも比例しない．さらに，新たなデバイス・センサーの登場により，データの種類も増加し続けている．そこで，データの質と量という特性を考慮したデータ処理の仕組みとして，データ分散型アーキテクチャが注目されている．

アプローチの 1 つとして，データ価値密度を定量化し，その値に応じて処理する場所を決める手法 Krill [12] が提案されている．Krill では，ある時刻に測定されたデータ価値を，そのデータの消費者から見た価値の総和とし，時間が経過するほどデータ価値の分散が大きくなるとする．後の時刻のデータ価値はその時刻の確率分布によって決定される．その予測価値によって，価値の高いデータのキャッシュへの格納，低いデータの削除，といった最適化を行う．著者らは，この手法を用いて，発生するデータの大部分をネットワークのエッジ部分で格納・処理するエッジ・ヘビー・データアーキテクチャを提唱している．

モビリティサービスにおいても，センサー・カメラの高精度化・高解像度化やセンサー数の増加に伴い，車両ごとのデータ量が継続的に増加することが見込まれ，その全てを集約することがコスト的または転送量的に不可能となる状況が想定される．その際，本手法のようなデータ分散型アーキテクチャを適用し，データの質と量に応じた処理が必要となると考えられる．

3.3 異種データ統合

2.3 節で述べた通り，モビリティサービスでは静的データ，動的データ，予測データ，時空間データ，といった異種のデータが混在しており，別々のデータベースや検索言語が必要なため，各種データを組み合わせた解析が困難となっている．これらのデータモデルの違いを吸収し，SQL ライクな統一検索言語でデータ操作が可能な情報統合手法 [13] が提案されている．例えば，動的な情報を道路地図上に重畳させたダイナミックマップを作成する場合は，下記の 4 種の情報を取り扱う必要がある．

動的情報：車両 (位置，速度，向き，..) ，歩行者（位置，速度，向き，..)，信号（状態，動作モード，..）..

静的情報：車両（サイズ，スペック，..)，歩行者（身体的情報，運転行動，..）..

予測情報：T 秒先の予測，時間帯毎の予測，走行プラン ..

道路地図：リンク，レーン，道路形状 ..

これらの情報は，時空間データ，ストリームデータ，グラフデータ，構造化データ，といった異種のデータモデルから構成されており，別々のデータ操作が必要となってしまう．

情報統合手法では，異種データをカバーする共通データモデルと，共通の操作体系（検索言語）を提供することにより，データの統合利用を実現する．

モビリティサービスの開発において，多様なデータから新たなサービスを作り出す必要性は高いため，本手法のようにデータを容易に扱う異種データ統合の技術動向に着目する必要がある．

4. デンソーのモビリティサービスプラットフォーム

当グループでは，3 章の技術動向を踏まえたモビリティサービスプラットフォームを開発・運用している．本節では，そのプラットフォームの概要と技術的特徴，及びモビリティサービス品質を支える SRE 活動について述べる．

4.1 アーキテクチャ概要とその特徴

Fig. 2 に示すように，デンソーではマルチモーダル等のモビリティサービス，モビリティサービスの安定稼働および開発促進を実現するモビリティサービスプ

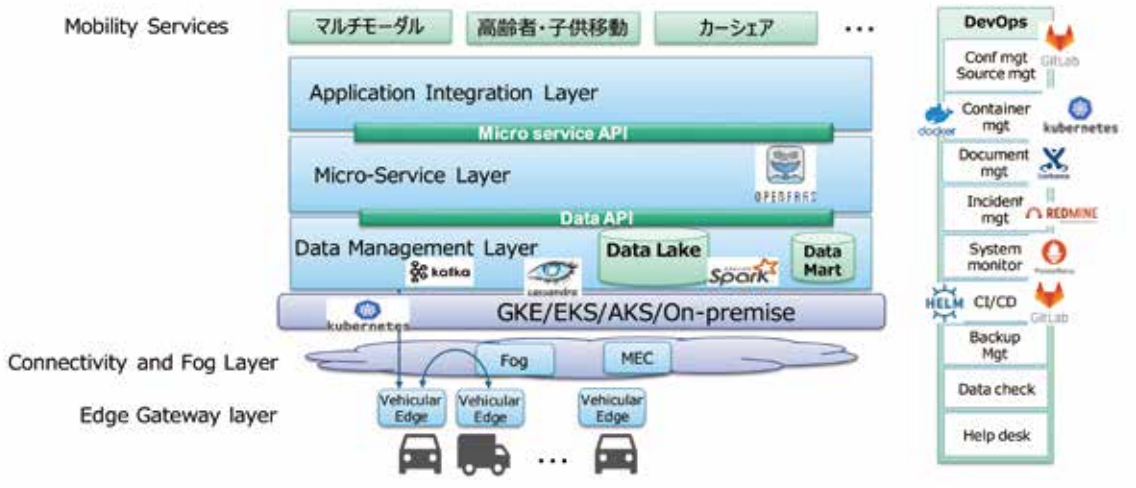

Fig. 2 Denso Mobility Service Platform (prototype)

ラットフォーム，車両をコネクテッドにする Vehicular Edge を提供している．特に，モビリティサービスプラットフォームでは，モビリティサービスの開発しやすさとスケールしやすさを重視し，クラウドネイティブ技術とパブリッククラウドのマネージドサービスを積極的に取り入れている．

4.2 モビリティサービス品質を支える SRE 活動

SRE は Site Reliability Engineering の略称である．SRE とはソフトウェアのコーディングやエンジニアリングによって，システム全体の可用性・性能・拡張性・セキュリティなどの品質を高める活動全般を指す．明確な定義はないが，その活動内容はデジタルビジネスを運営する企業が必然的に取り組んできた活動である．Google 社がこの活動を SRE という言葉で定義したことで注目が高まっている．

我々デジタルイノベーション室の SRE 課でも，MaaS やコネクテッドプロダクトの「品質要件／ SLI・SLO・SLA の定義とそれを実現するためのアーキテクチャ設計，実装，評価，そして本番稼働後の監視と改善の仕組みづくり」において，アジャイル開発チームを支援している．

デジタルイノベーション室が支援するプロダクトは様々で，プロダクトの用途やそのビジネス計画によって品質要件／ SLI・SLO・SLA や投資金額・期間・体制といった制約も異なり，したがってその実装も異なる．例えば，あるプロダクトに求める可用性が 99.0% なのか 99.999% なのか，システム障害が発生した際の復旧目標時間が 1 時間なのか 1 日なのか，対象車両台数が 1,000 台なのか 1,000,000 台なのか，求める時間効率性が 5MB/min なのか 5GB/min なのか．これらによって冗長構成や処理方式など，プロダクトのアーキテクチャが大きく変わる可能性がある．さらに，プロダクトが利用できない状態に陥った時の利用者への影響が大きいか小さいか，本番稼働後に短期間で車両台数が大幅に増減する可能性があるかないか．これらによってもプロダクトがこれらの品質要件／ SLI・SLO・SLA を継続的に満たしていることの監視方法，異常を検知した場合の対応プロセスや体制，仕組みが変わる可能性がある．

そのような多様性に対して効率的に対応するために，SRE 課では，プロジェクトで培ったスキルとアセットを活用・改善しながら活動している．アセットとしては下記のようなものがある．JIS X 25010 で定義されているシステム及びソフトウェアの品質特性（Table 1）や，我々が実際に定義した品質要件／ SLI・SLO とその監視観点の例（Table 2）と合わせて参照いただきたい．

1. 24 時間 365 日稼働する車両と繋がる高可用性サーバーシステム基盤アーキテクチャ：
   自家用車も商用車も 24 時間 365 日いつでも利用される可能性があるため，自動車とサーバーアプリケーションとの間をつなぐモバイル網の電波が繋がりづらい環境があったり，サーバーアプリケーションが利用するクラウドサービスの稼働率が

100% ではないという制約がある一方で，モビリティサービスに対しては 24 時間 365 日連続稼働が求められるケースもある．これを実現するために，SRE 課では下記のような設計要素を含むアーキテクチャをベースにして，個別のソリューションアーキテクチャを策定することに努めている．

（包含する設計要素）単一障害点の排除，プロセスダウン時の自動再起動，バッチアプリケーションエラー終了時の自動再実行，要求増減に応じたサーバー台数の自動加減，品質要件／ SLI・SLO・SLA に基づいたシステム監視とアラート

2. 数十万台の車両からあがってくるデータを分単位で収集し，時間単位で集計処理するデータ連携・集計ソリューションアーキテクチャ：
自動車は日本国内での所有台数だけでも 8,000 万を超える数になり，モビリティサービスの潜在対象台数はかなりの数となる．しかも，モビリティサービスで扱うデータはセンサーデータから動画まで多様かつ大量対象となる．これをすべて処理するとなるとかなり高いコストを払う必要がありビジネスが成立しづらい．このため SRE 課では，下記のような設計要素を含むアーキテクチャをベースにして，効率的にデータを処理する個別のソリューションアーキテクチャを策定することに努めている．
（包含する設計要素）車載器で処理できるものは結果をクラウドに送信，クラウド上で処理する必要がある処理要求は負荷分割してキューイングして並列処理
3. ミスなく短時間でサーバーシステム基盤を構築するためのインフラ構築ツール：
自動車の利用期間が数十年という長期にわたることに加えて，その間に数台規模から数百万台規模までの車両台数の増減やモビリティサービスの継続的な追加・改善が求められる．それらに対応するために，開発，テスト（性能，障害，脆弱性），ステージング，本番といったサーバー環境の構築・削除を効率的に運用する必要がある．そのため SRE 課では下記のようなツールを作成し，活用している．
（包含するツール）サーバーシステム基盤設計の標準テンプレート，構築操作を自動化するベンダー非依存形式で実装したコード，構築した環境の妥当性チェックツール

特に SRE 課が重視しているのが「リーズナブルな品質要件／ SLI・SLO・SLA の定義」である．ビジネス計画上で数年後に数百万台を目指すことがうたわれている場合でも立ち上げ時点では対象台数が 10 台であったり，プロダクト利用地域に地下やトンネルの中などモバイル網の電波が届きづらい場所が含まれている場合がある．そのようなときに必要以上に品質要件を求めたり，制約を無視して設計されたプロダクトはコスト高になりビジネスが成立しづらい．したがって，我々は，プロジェクト開始前に必ず品質要件／ SLI・SLO・SLA についてステークホルダーとリーズナブルな値で合意してから活動を進めるように努めている．

Google 社の SRE ディレクターのトッド氏は，SRE チームのエンジニアとして最も大切な「素養」「視点」について以下のように発言している [17]．読者の皆様にも是非参考にしていただきたい．

全てに対して「No」としか言わないような，純粋に消極的なエンジニアは役に立ちません．反対に，常に全てのデータセンターが機能し，ネットワークに帯域があり，ディスクスペースに余裕があるという前提で行動するような，楽観的なエンジニアも好ましくありません．

私たちが必要としているのは，障害は必ず起こるという前提の下に，正確に障害の種類を予測することができ，それぞれの障害がシステムにどのように影響するかを理解し，そうした状況に対して建設的に対処できる人間です．

Table 1 System and software quality characteristics (source: JIS X 25010)

ソフトウェアの品質特性	副特性
機能適合性	機能完全性、機能正確性、機能適切性
性能効率性	時間効率性、資源効率性、容量満足性
互換性	共存性、相互運用性
使用性	適切度認識性、習得性、運用操作性、ユーザーエラー防止性、ユーザーインターフェース快美性、アクセシビリティ
信頼性	成熟性、可用性、障害許容性、回復性
セキュリティ	機密性、インテグリティ、否認防止性、責任追跡性、真正性
保守性	モジュール性、再利用性、解析性、修正性、試験性
移植性	適応性、設置性、置換性

Table 2 Example of system monitoring requirements based on SLO and SLO

SLI・SLO			監視範囲	監視要件			補足
				予防のための分析	早期発見のためのアラート	早期解決のための情報提供	
信頼性	可用性（業務継続性）	24時間無停止（計画停止あり）	1.死活監視 応答があることを監視 2.エラー監視 動作異常がないことを監視	・エラー発生数の増減傾向分析	・Web応答異常アラート ・Webアプリのエラーログ検知アラート ・バッチアプリのエラーログ検知アラート ・プロセスダウン/稼働エラー	・エラー発生箇所 ・エラーログ ・エラーの影響範囲 ・過去の問題解決方法事例	・時に初期流動においてはアプリエラー発生数の増減傾向を注視する
	回復性（目標復旧時間）	6時間以内					
	可用性（稼働率）	99.9%					
性能効率性	時間効率性（オンラインレスポンス）	ピーク時は90%適守、通常時は95%適守	3.リソース監視 リソース不足（応答遅延の兆候）が発生していないことを監視 4.パフォーマンス監視 応答遅延がないことを監視	・同時アクセス数の増減傾向分析 ・応答時間の長短顕化傾向分析 ・システムリソース使用量の増減分析	・同時アクセス数の閾値超えアラート ・Web応答時間の閾値超えアラート	・閾値超え発生箇所 ・同時アクセス数 ・Web応答時間 ・システム内の処理時間の内訳 ・システムリソース使用量 ・システムベンチマークデータ	・対象業務の稼働が現時点で不明で、各社ごとに異なる可能性が高いしたがって、特に各社の利用開始時期においてはデータ件数の推移に注意を払って、十分な対策を事前に講じる必要がある ・性能が十分に出ていて、かつリソース使用量が低い場合には該当構成を縮小することによってインフラ費用の低減に努める
	時間効率性（バッチレスポンス）	○○データは車載器にて発生後5分以内で画面で確認できる状態にすること。 運転評価は翌日朝9時に確認できる状態にすること		・データ件数の増減傾向分析 ・応答時間の長短顕化傾向分析 ・システムリソース使用量の増減分析	・データ件数の閾値超えアラート ・処理時間遅延の閾値超えアラート ・ジョブ滞留数の閾値超えアラート	・データ件数 ・処理時間 ・システム内の処理時間の内訳 ・システムリソース使用量 ・システムベンチマークデータ	

5. 考察

モビリティサービスを生み出すプラットフォームについて，2章でプラットフォームの基本機能，3章ではプラットフォームを強化する技術動向，4章ではデンソーの取り組みについて解説した．

このような単体のモビリティサービスを生み出す仕組み以外に，モビリティサービス同士の組み合わせ（マッシュアップ）や，交通情報などのオープンデータとの組み合わせにより新たなサービス (Mobility as a Service: MaaS) を作り出すという動きがある．特に欧州が先行しており，MaaS Alliance [14]，MaaS4EU [15] という団体が設立されている．このようなMaaSにより，場面に応じてユーザが最も望む交通手段をより手軽に使えるようになることが期待されている．

インターネットサービスでは，Google Mapsが2005年に登場した際，サービス単体としての機能拡張を続けつつ，Google Maps APIを活用した他のサービスやオープンデータとのマッシュアップにより付加価値を向上させ，地図サービスという市場を開拓した．例えば，レストラン口コミサービスとのマッシュアップによるアクセス経路表示や，災害時の避難所オープンデータとのマッシュアップによる避難経路提示というように，多岐に渡って使用されている．

そのアナロジーで考えると，モビリティサービスにおいても，Fig. 3に示すようにモビリティサービスプラットフォームと他のモビリティサービス・交通データが今後統合されていくことが考えられる．プラットフォームによるモビリティサービス単体の機能拡張と，マッシュアップによるエコシステムの拡大が相乗効果を生み，さらなるユーザの利便性の向上と，モビリティサービス市場拡大が期待できると思われる．

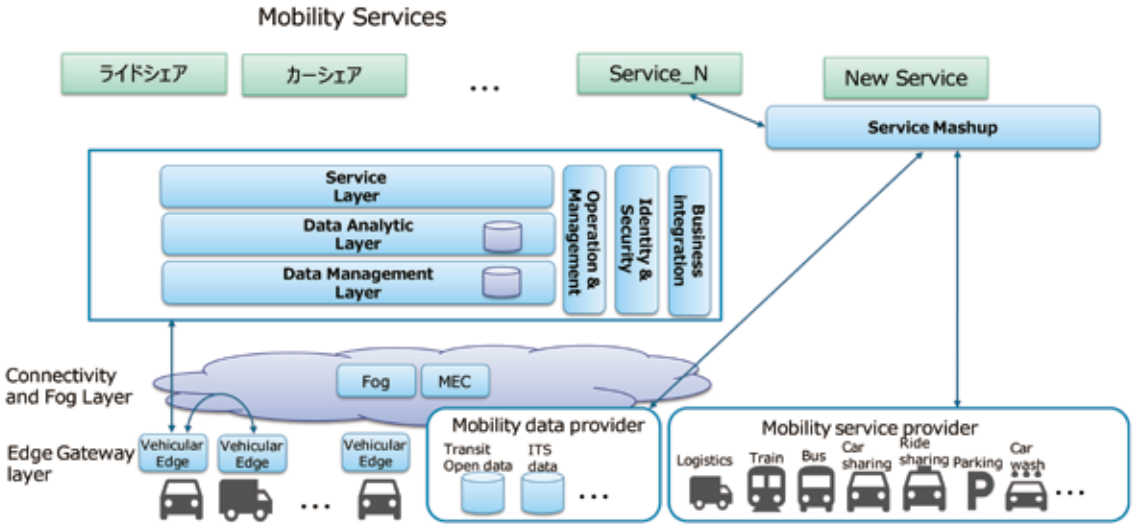

Fig. 3　Mobility Service Framework + Service Mashup

また，本稿では車両内のエッジコンピュータを含むクラウド側の領域を対象としているが，エッジコンピュータより下の車両側をソフトウェア化 (Software Defined Connected Car: SDCC)[16] するという動きがある．

一般的な Software Defined のアプローチでは，対象のハードウェアを仮想化し，ソフトウェアで仮想ハードウェアを定義し，制御する．SDCC のアプローチでは，下記のような機能単位で Electronic Control Units (ECU) を仮想化し，機能の柔軟な追加・変更や，サードパーティによる新たな機能提供を目指している．

・Instrument Cluster：インストルメントパネルに表示する機能
・IVI systems：マルチメディア，ナビゲーション，空調，リアビューカメラといった機能
・Telematics：センサー情報，CAN 情報等を収集する機能
・Safety critical functions：Parking/lane assistant, digital mirror といった ADAS 機能

まずは走る・曲がる・止まるに影響しない範囲でのソフトウェア化が進む見込みであり，デファクト化までは時間がかかると思われるが，その動向には留意すべきと思われる．

6. おわりに

本稿では，モビリティサービスの開発を促進するプラットフォームの技術動向について解説した．モビリティサービスプラットフォームを構成する技術要素と，モビリティサービスプラットフォームを強化する技術動向について言及し，デンソーの取り組みについて紹介した．また，今後拡張すべき技術領域について考察した．

モビリティサービス市場には，異業種からの参入が相次いでおり，今後も技術開発の加速が見込まれる．特に，クラウドネイティブ関連の動きが早いため，動向を注視する必要があると考えられる．

参考文献

1) IEEE P2413 “Standard for an Architectural Framework for the Internet of Things“
2) ITU-T Y.2060 “Overview of the Internet of things”, 2012
3) A. Al-Fuqaha,” Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications”, IEEE Communications Surveys & Tutorials, Volume: 17, Issue: 4
4) M. Whaiduzzaman, et. al., “A survey on vehicular cloud computing”, Journal of Network and Computer Applications, 2014
5) W. He, et. al., “Developing vehicular data cloud services in the IoT environment”, IEEE Transactions on Industrial Informatics, Volume10, Issue2, 2014
6) N.Lu, et. al., “Connected Vehicles: Solutions and Challenges”, IEEE Internet of Things Journal, Volume: 1 , Issue: 4 , 2014

7) T-System “SD-WAN white paper”
8) R. Roman, et. al., “Mobile Edge Computing, Fog et al.: A Survey and Analysis of Security Threats and Challenges”, Future Generation Computer Systems, Volume 78, Part 2, 2018
9) A. Wu“Taking the Cloud-Native Approach with Microservices: White Paper“
10) BMW “Cloud Native@BMW Group, Technology for the agile transition”, RedHat Summit 2017
11) Ford Motor Company, “Ford Motor Company’s Cloud Native Reference Application”, SpringOne Platform 2017
12) D. Okanohara, S. Hido, N. Kubota, Y. Unno, and H. Maruyama, “Krill: an architecture for edge heavy data,” in Third Workshop on Architectures and Systems for Big Data, 2013.
13) 渡辺陽介，”交通マネジメントに向けたダイナミックマップアーキテクチャの研究”，システム／制御／情報：システム制御情報学会誌，Vol.60，No. 11, 2017
14) MaaS Alliance, “White Paper – MaaS Alliance”, 2017
15) MaaS4EU, “http://www.maas4eu.eu/”
16) The Linux Foundation, Automotive Grade Linux (AGL) Virtualization Expert Group (EG-VIRT) “The AGL Software Defined Connected Car Architecture”, 2018

引用文献

17) 内野宏信，“米 Google SRE ディレクターに聞く，運用管理の意義，価値，役割”，＠ IT，2017

著者

小泉 清一
こいずみ せいいち

MaaS 開発部
車載及びクラウドのコネクティッド基盤先行開発を担当

栗原 浩
くりはら ひろし

MaaS 開発部
アジャイル開発チームを横断した非機能要件対応・運用管理を担当