Vol.2017-ITS-68 No.1
2017/2/28

完全自動リアルタイムフルデマンド交通システムSAVS向けプラットフォームの設計と実装

平田 圭二[1,a)]　鈴木 恵二[1,b)]　野田 五十樹[2,c)]　落合 純一[2,d)]　金森 亮[3,e)]　松舘 渉[4,f)]
中島 秀之[5,1,g)]　佐野 渉二[6,h)]　白石 陽[1,i)]　松原 仁[1,j)]

概要：現在我々は，完全自動リアルタイムフルデマンド交通システム SAVS のために移動サービスのクラウド化を実現するプラットフォームの設計と実装を行っている．本稿では，SAVS の目的を述べて，Uber との相違点を検討する．そして，当該プラットフォームの機能，仕様，実装などについて現状報告を行う．

Hirata Keiji[1,a)]　Suzuki Keiji[1,b)]　Noda Itsuki[2,c)]　Ochiai Junichi[2,d)]　Kanamori Ryo[3,e)]
Matsudate Wataru[4,f)]　Nakashima Hideyuki[5,1,g)]　Sano Shoji[6,h)]　Shiraishi Yoh[1,i)]
Matsubara Hitoshi[1,j)]

1. はじめに

現在我々は中規模都市を対象とした新たな公共交通システム Smart Access Vehicle (SAV) System を開発している [2]．SAV System (SAVS) はデマンド応答型公共交通 (Demand Responsive Transport，オンデマンドバスとも呼ばれる) の一種であり，固定経路を持たずユーザの呼び出しに応じて即時に配車される．この時，先に乗車している/しようとしている乗客が乗合いを許容しその乗客の目的地到着希望時刻が守れるのであれば，乗合いが発生する．つまり，従来のタクシーとバス両方のサービスを兼ね備えたような交通システムである．

SAVS は，ICT を活用することで初めて可能となる移動サービスのクラウド化 [3] の提供を目指している．タクシーやバスはサービスとハードウェアが分かちがたく結びついており，タクシー車輌でなければタクシーという移動サービスは受けられないし，バス車輌でなければバスという移動サービスは受けられない．しかし，ユーザにとって，移動サービスに関して様々な条件を提示できる方が望ましい．例えば，今すぐここで乗りたい，何時より後に迎えに来て欲しい，乗合いをしても構わない，何時までに目的地に到着したい，できるだけ早く目的地に到着したい，安い料金が望ましい，途中下車したいなどである．ここで ICT は，ユーザの希望するサービスに最適な車輌の手配を可能にする．実際に配車される車輌はタクシーかもしれないし，バスかもしれない．このように，サービスとハードウェアを分離し，需要としての移動サービスと車輌の最適な組み合わせをユーザに提供することを移動サービスのクラウド化と呼ぶ．

サービスとハードウェアの分離と組み合わせに関して，アナロジとして音楽とその記録媒体を考えると理解しやすいだろう．アナログレコード時代は，音楽とそれを記録するアナログレコードは分かちがたく結びついており，音楽を購入するとは即ちアナログレコードを購入することであった．CD 時代は，デジタル情報のコピーが可能となり，音楽という情報と記録媒体の間に組合せの自由度が生まれた．そして現在のようなストリーミング時代は，ネットワークという伝達媒体が記録媒体に取って代わった．ま

1 公立はこだて未来大学：Future University Hakodate
2 産業技術総合研究所：National Institute of Aadvanced Industrial Science and Technology
3 名古屋大学：Nagoya University
4 株式会社アットウェア：Atware
5 東京大学：University of Tokyo
6 金沢工業大学：Kanazawa Institute of Technology
a) hirata@fun.ac.jp
b) kjsuzuki@fun.ac.jp
c) i.noda@aist.go.jp
d) j.ochiai@aist.go.jp
e) kanamori.ryo@nagoya-u.jp
f) wmatsu@atware.co.jp
g) h.nakashima@fun.ac.jp
h) sano@neptune.kanazawa-it.ac.jp
i) siraisi@fun.ac.jp
j) matsubar@fun.ac.jp

た別の見方として，記録という機能がアナログレコード，CD，ネットワークと仮想化されてきたと考えることもできる．つまり，移動サービスのクラウド化は，移動手段の仮想化とも考えられる．

移動サービスのクラウド化に類似する概念として Mobility as a Service (MaaS)[1] がある．ユーザからの移動サービスの要求に対して，移動オペレータというエージェント（もちろん ICT である）が，ユーザの希望や状況を総合的に考慮して，徒歩，自転車，タクシー，オンデマンドバス，通常のバス，鉄道などの候補から最適な方法や組合せを選択する．ユーザの条件や希望には，例えば，雨の日は濡れないルートを選択したい，乗り物に乗るなら揺れの少ない席が嬉しい，荷物が少なくて身軽な時は距離が少し長くても自転車で構わないなどがある．結果として，社会全体で交通システムの効率化が図られるという．商用サービスであるナビタイムの拡張版と考えることもできよう．このように MaaS では，SAVS のようにタクシーとバスの中間というきめ細かい最適サービスの提供までは考えていない．しかし，SAVS もその仮想化の対象を徒歩，自転車，鉄道などまで拡大し，さらにユーザのアウェアネスを考慮することで，サービス水準を向上させることが考えられる．

移動サービスのクラウド化の特長は，上で述べた要求されたサービスレベルへの柔軟な対処と交通システム全体での動的な最適化に加えて，サービス共創の促進，人流と物流の統合などがある．まず，SAVS が提供する移動サービスは他のサービスとの組合せが容易である．例えば，飲食店が送迎サービスまで含めた宴会プランを提示したり [*1]，SAV によって医療ツーリズムとショッピングを組合せることもできる．さらに，集配される個々の品物が SAV を呼び出すと考えれば，人流と物流の自然な統合が可能となる．このように，SAV を媒介とすることでサービス共創の活性化が期待される [*2]．

今後の大規模な実証実験に向けて，より快適なユーザ体験を提供し，より効率的かつ柔軟な全体最適化を達成し，かつどのような都市にでも容易に展開し運用継続する必要がある．そこで我々は，SAVS は Web API 群を持つプラットフォーム・アーキテクチャを採用すべきと判断し，新 SAVS の実装を進めている [5]．本稿では，その SAV 向けプラットフォームの設計と実装について述べる．

2. Smart Access Vehicle システム

2.1 これまでの研究開発と実証実験

SAVS の本格的な研究開発は 2013 年より始まり，これまで実証実験を 4 回実施した（表 1）．第 1 回目実験の目的はフィージビリティ・スタディであり，半自動での運用であった．第 2 回目において世界初の完全自動リアルタイム配車に成功した．第 3 回目は，公立はこだて未来大学にて開催された人工知能学会全国大会参加者を対象とした実験であった（利用登録者約 500 名）．第 4 回目は，乗合い発生頻度を高く設定し，システム全体の安定運用と配車アルゴリズムの動作観察，ユーザ体験に関する情報収集を行った．これら実験の結果を踏まえて改善点を洗い出し，さらにサービス共創を促進する仕組みとして，現在 Web API 群を持つプラットフォーム構築を進めている．

2.2 SAV システムの動作と特徴

ユーザが SAV を呼び出す時の動作を図 1 に示す．まず

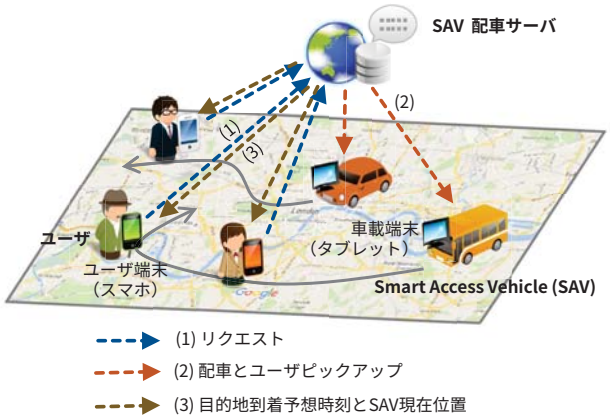

図 1　呼び出し・配車の基本動作

ステップ (1) にて，ユーザはスマホ上のアプリを通じて，SAV 配車サーバにリクエストを上げる．この時，移動サービスのクラウド化による全体最適化を実現するため，ユーザは事前に移動サービスに関する全ての条件をサーバに通知しなければならない（3.4 節）．ユーザのリクエストに対し，配車サーバが最適な SAV を選択し，ステップ (2) にて当該 SAV にユーザをピックアップする指令を送る．SAV 運転手は，SAV に積んだ車載端末（タブレット）上のアプリでその指令を受信し，ユーザを迎えに移動する．同時に，配車サーバはユーザに対し，目的地到着の予想時刻，迎えに行く SAV の現在位置を通知する．

SAVS には次の 3 つの特徴がある．1 つめは，ユーザからのリクエスト応答処理に，予約ベースでなく即時実行ベースを採用したことである．これより，SAVS は，例えばバスのような固定路線，固定ダイヤの運行形態をエミュレートできる．つまり，多数のユーザが毎回同時刻に同じ場所間を移動するデマンドを発出し，それを SAVS が即時に処理すると考えれば，バスと等価な運行形態が実現される．同様に，従来タクシーをエミュレートすることも，デマンドバスをエミュレートすることもできる．

[*1] 自家用車通勤が多数を占めるような地方都市では，宴会の日は出勤時及び退社時の移動も SAV を利用することで，勤務先と飲食店の間の効率的な移動も可能となる．
[*2] SAVS で配車される車輌として Uber や Lyft が最適な場合もあろう．

表 1　全 4 回の SAV 実証実験

回	場所	年月	期間・時間帯	SAV 台数	リクエスト受付数	サービス完了数
1	函館	2013/10	7 日間・7:30〜19:00	5	941	781
2	函館	2014/4	1 日間・11:00〜18:00	16	107	58
3	函館	2015/5	4 日間・12:00〜19:30	20†	772	483
4	東京	2016/12	4 日間・10:00〜18:00	8	888	717

† 4 日間の内，1 日だけ 30 台投入した

2 つめは，過疎地域ではなく，少なくとも中規模都市より大きいエリアとユーザ数を想定していることである．シミュレーション結果より，ある規模を超えると現状のタクシーやバスによる乗客輸送輸送システムの効率を上回る領域があることが明らかになっている [4]．ユーザは移動サービスに関する条件を全て事前に SAVS に通知し，計算機がそれを考慮して完全自動で最適な配車を決定するので，原理的に SAVS を展開するサイズが大きくなればなるほど全体最適化の余地が増える．

3 つめは，乗客輸送を制御するパラメータ，例えば移動経路，出発時刻，乗合の可否などを動的かつ連続値として設定できることである．これより，ユーザの要望，その場の状況や制約に応じて，柔軟かつきめ細かいにサービスレベルを実現することができる．一般の交通システムに同じく SAVS でもコストと利便性の間にはトレードオフがあるが，SAV 車輌ごとに乗客ごとに輸送に関するパラメータを動的に変更してサービスレベルを調整して，そのトレードオフに対処する．また，そのパラメータ変更を受けて全体最適化の制御を変更することになる．

2.3　SAVS の Uber に対する優位性

我々は頻繁に SAVS と Uber の違いや SAVS の Uber に対する優位性に関する質問を受ける．ここで一度，SAVS の優位性を整理しておく．

(1) SAVS は移動サービスを全体最適化する: SAVS ユーザが発出する移動リクエストには受けたい移動サービスに関する全ての情報と条件が含まれている（乗降車位置，時間制約，予約，料金，乗合いの可否など）．SAVS は全ユーザの移動リクエストと全 SAV の状況を考慮して，動的に最適な SAV を選び出し経路とサービスレベルを決定する．ユーザはサービスカテゴリを選択する必要がなく，ユーザに提供されるサービスレベルは全て SAVS が決定する．したがって例えば，イベントの開催時に多数ユーザが同時に SAV を呼べば，SAVS は大型バスを手配する．時間に余裕のある乗客に対してより多くの乗合いを割り当てる（ただしその分，運賃は安くなる）．また，各移動リクエストの乗降車位置とその時点での効率的な SAV の待機状況を考慮した上で，どの SAV をどのユーザに配車するかを決定する．

一方，Uber は，限定された移動リクエスト情報に基づいて 1 台の車輌をできるだけ効率的に配車しようとする．例えば，Uber は（ユーザに入力させるが）降車場所情報を利用せずユーザの近くにいる車輌を配車する（UberPOOL を除く）[*3]．また，車輌は基本的に一般乗用車のみが使われる．

(2) Uber は従来のタクシー呼び出しのオンライン化である: そのオンライン化により若干のサービスレベル向上はあるものの [*4]，ワークフロー自体は同じである．ユーザの立場から見れば，SAVS も Uber もそのワークフローに大差は無いように見える．それは，SAVS も Uber もユーザは乗降車場所をシステムに申告し，運転手はシステムからユーザを乗降車させる場所を指示されるからである．

しかし，ユーザは Uber を利用する前に，Uber の乗合いサービス (UberPOOL，エリア限定) なのか，その他（タクシーやハイヤー相当のサービス，例えば UberX や UberBLACK）なのかを選択しなければならない．これも従来のワークフローと同じである．一方 SAVS では，ユーザは受容するサービスカテゴリを指定せずに移動リクエストを上げるだけである．ユーザが受容するサービスカテゴリは SAVS が決定する．さらにシステムの立場から見れば，上述したように，配車を決定する際に降車場所の情報を使うか否かという大きな差がある．

(3) SAVS は B2B2C である: もし移動サービスが，(a) 純粋に移動するというサービス，(b) 移動すること自体に対する付加価値，(c) 移動した後で受容するサービスの 3 つから構成されているとしたら，(a) はどのような事業体が提供しても差異がないインフラの位置づけになる．それならば，できるだけ効率良くかつ利用しやすい形で提供されることが望ましい [*5]．我々は，SAVS はそのようなインフラとしての移動サービスを提供する仕組みとして最適だと考えている．したがって，SAVS の提供する移動サービスの取引形態は B2B2C であり，SAVS は一番左の B に位置する．つまり，SAVS は中間の企業向けにクラウド化された移動サービスを提供するプラットフォーマであり，中間の企業 B がサービスデザイナとして移動サービスを他のサービスと組み合わせてカスタマ C に提供する．

*3　日本での UberX は目的地の入力をスキップすることができる．
*4　海外で Uber の評価が高い理由の一つに接客やサービスの良さが挙げられるが，日本のタクシー会社なら容易に達成可能なレベルだと思われる．
*5　例えば，オペレーティングシステム，ネットワーク，電気，水道，道路などを想像していただくと分かりやすいだろう．

これより，Uber on SAVS というソリューションが可能となる．つまり，Uber は，純粋に移動するというサービスを SAVS から購入し，それに例えば運転手とユーザの相互評価という付加価値を付けて，カスタマに提供することで，Uber のサービスを継続できる．あるいはレストランと組合せて UberEATS を実現することもできる．現在でも Uber 自体は営業車輌を保有していないが，SAVS から B2B で移動サービスをバルクで購入した方が低コストとなろう．

3. SAVS 向けプラットフォーム

3.1 機能拡張性

ユーザが発出する移動リクエストには，移動サービスに関する全ての情報と条件が含まれると述べたが，最初から全ての情報と条件を列挙し尽くすことは不可能である．それは，実験や運用を重ねていく内に追加されたり置換されていくものである．例えば，ユーザの乗合い可否を動的に変更したい，運転手の休憩時刻を予約入力したいなどである [*6]．

さらに，現在の配車アルゴリズムは即時ベースであるが，ユーザからの数時間先や数日先の予約を受け付けて欲しいという声は大きい．しかし，予約の受付は台数制限を含む効率向上とトレードオフの関係にある．さらに，きめ細かいサービスレベルを実現するために，乗降車時刻のハード締切とソフト締切の指定などが可能となることが望ましい．

実証実験では，ユーザがどの程度の金額を妥当な運賃と考えているかの調査を行うため，降車後ユーザにアンケートのメールを送信した．しかし，そのアンケートのメールを読むには，移動リクエストを発出するスマホアプリとは別のメーラアプリを起動する必要があり，アンケート回収率は高くなかった．ここでもしユーザのスマホにプッシュ通知する一般的な機能（例えば Android の notification）を実装しておけば，アンケートの実施だけでなく，広告やトラブル時の緊急連絡などにも容易に対応できるだろう．

実用化に向けて，必要経費の算出と適正な料金設定に関する検討は必須である．また我々は，インフラとしての移動サービスは安心して利用できることも要件の一つと考えるので，運賃事前確定が望ましい．そのためには，乗合いが確率的に発生する前提のもとで移動時間，移動距離等の正確な事前予測が必要である．この予測技術は常に改良されていくであろうし，さらに，移動時間，移動距離等からユーザにとって妥当な料金を計算する関係式も常に改良されていくであろう．

したがって，SAVS には高い機能拡張性が要求される．

3.2 ユーザ体験

ユーザが移動リクエストを発出する際，移動サービスに関する全ての情報と条件に加えて，ユーザや運転手の選好に関する要望も尽きることはないだろう．増える一方のそれら情報，条件，要望を効率良く SAVS に入力するために，ユーザ端末の UI，運転手端末の UI に様々な改良を加える必要がある．

ユーザが移動リクエストを発出した後，SAV 配車結果が戻ってくるまでの時間は短ければ短いほど好ましい．表 1 の各実験で用いたシステムでは，短い時で数秒，長い時では 30 秒から 1 分間かかることもあった．あるユーザはこの待ち時間に不満を覚えたと答えている．

実証実験にて実際に SAVS を運用してみると，トラブルは乗車時に最も多発した．それは，所定の時刻に乗車地点にてユーザや SAV 車輌がお互いを発見できず探し回るようなケースである．このようなトラブルは 1 回でも発生すると SAVS に対する印象を大きく損なう [*7]．

したがって，SAVS には良好なユーザ体験が要求される．

3.3 システム構成

以上より，SAVS には全体最適化による高効率の他，高い機能拡張性，良好なユーザ体験，実時間性が要求されるので，現在開発中の SAVS は，Web API 群を持つプラットフォーム・アーキテクチャを採用した．そのシステム構成を図 2 に示す．まず，モジュラな構成とし，配車サーバ，配車エンジン，料金計算サーバ間の通信を全て標準的な REST API で実装する．配車サーバ内に全乗客と全 SAV の状態を管理するデータベースを置き，サーバ間通信でアクセスする．モジュラな構成としたことで，プログラムの見通し向上，開発スピード，モジュールテストの効率化が向上した．特に，配車サーバの中で動作する配車計算ロジックは，今後改良と拡張が頻繁に繰り返されるところなので，独立性高く開発できるメリットは大きい．

システムは大よそ次のように動作する．乗客端末のアプリから配車リクエストが届くたびにデマンドキューにそのリクエストが登録され，配車計算制御を経て，その時点での全ユーザ，全 SAV 配車状況などの情報とともに配車エンジンに配車リクエストとして送信される．配車エンジンに状態を持たせないのは，簡潔なデータ管理と開発効率のためである．配車エンジンが新しい SAV 割り当ての計算結果を出すと，配車計算 API 経由で配車サーバ内のデータ

[*6] SAVS と Uber の設計思想の違いは，運転手の休憩の取り方にも現れている．Uber の運転手が休憩を取るには，単にシステムからの配車依頼通知に応答しなければいいだけである．その時システムは乗車地点の近くにいるまた別の運転手のアプリに配車依頼通知を送信する．Uber は乗車地点の情報のみで車輌を決定するので，該当する候補は比較的多い．一方 SAVS の場合，システムからの配車依頼通知には必ず応答し指示に従うことになっている．これは乗車地点だけでなく降車地点も含めて最適な車輌を決定しているので，その代替候補が比較的少ないためである．そのため，SAVS の運転手アプリには明示的な即時休憩リクエストボタンが設置されている．

[*7] 表 1 から，第 4 回実験での完了しなかったサービスの割合は約 20%である．

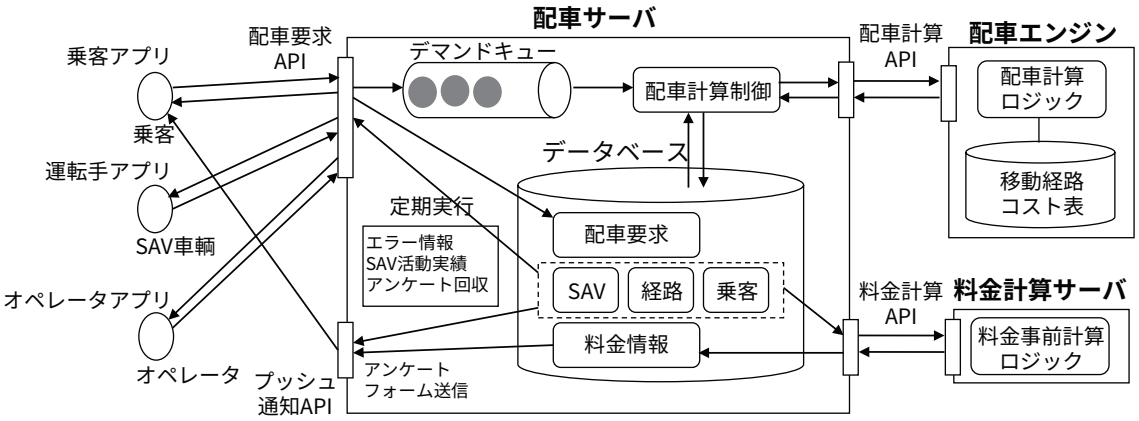

図 2　SAVS 向けプラットフォームのシステム構成

ベースに登録される．そして，その新しい SAV 割り当ては，配車要求 API を通じて乗客と SAV 車輌に通知される．

デマンドキューは，配車エンジンが配車計算中に受信した（複数の）移動リクエストを保存しておき，配車計算が終了したら，すぐにその保存してある（複数の）移動リクエストを配車エンジンに送る．このデマンドキュー導入によって実時間性が向上した．また，旧版の SAVS で採用されていた，データベースのポーリングと，配車サーバと配車エンジンからのデータベース直接アクセスが解消されたことで，データベース中の各エンティティのロックや更新処理が不要になり，実時間性向上に寄与した．料金サーバは，乗合いを考慮した料金計算のモデルを構築し，ユーザに提示する予測料金を計算する．降車後確定した実際の料金を記録し，それ以降の予測精度向上に役立てる．ユーザにプッシュ通知を行う仕組みを利用して，アンケートフォームを送信する．

3.4　Web API の設計とメッセージ例

まず，配車要求 API の内，乗客アプリ関連の主なもの 4 つ，運転手アプリ関連の主なもの 3 つ，そしてプッシュ通知 API の概略を表 2 にまとめる．SAVS において，ID を振られる主要エンティティは，乗客 (userId)，SAV 車輌 (savId)，デマンド (demandId) であり，主要な構造体は，SAV 車輌の乗降地点のリスト (viaPoints)，配車情報（乗客，SAV 車輌，デマンド，乗降地点のリストから成る）である（図 3，図 4）．これらは，表中の API メッセージのパラメータとして現れる．

乗客からの配車リクエスト（デマンド）に対して SAV を配車する処理を説明するために，現在実装中の配車エンジン API のリクエストメッセージ例（図 3）とレスポンスメッセージ例（図 4）を示す．図中の “//” はコメントを表す．図 3 に，配車サーバが乗客からリクエストを受けた時に，配車エンジンに向けて送信する配車要求のメッセージ例を示す．配車エンジンは状態をもたないので，図中コメント (1) にあるように，現在の全 SAV の情報がメッセージ

表 2　配車要求 API とプッシュ通知 API の概要

	機能	パラメータ
乗客アプリ向け API	ユーザ・プロファイルを与えると，配車サーバに登録され乗客 ID が振られる	乗客 ID
	乗客に配車された SAV 車輌の現時点での位置，待合せ場所到着までの時間，目的地までの見積移動時間を返す	SAV ID と SAV 車輌情報
	配車された SAV をキャンセルする	SAV ID
運転手アプリ向け API	SAV ID を与えて，その SAV 車輌の配車情報を取得する	配車情報
	SAV ID を与えて，その SAV 車輌に乗降する客の情報を取得する	乗降客情報
	配車受付可か不可（休憩モード）の情報を取得する	稼働・休憩情報
プッシュ通知 API	乗客アプリへのプッシュ通知を行う．乗客からの意見，要望，アンケート結果の回収を行う	メッセージ

として送信される．コメント (3) のパートには，(1) の中で参照されるすでに SAV 車輌の割当てが完了した配車リクエストの全てが来る．したがって，サービス展開規模が実際の中規模都市以上になる場合は，リクエストメッセージのサイズを何らかの方法で圧縮する必要があろう．

次に，配車エンジンが配車サーバに戻す配車計算結果を図 4 に示す．新しい配車リクエストを処理した結果，すでに割当てられていた経路上の乗降場所への到着時刻の変更（遅延）が生じるケースがある．図中コメント (4) のパートには，そのような配車計算によって乗降時刻の変更が生じたた全 SAV のリストが来る（変更が生じなかった SAV の情報は含まれない）．コメント (5) のパートには，新しく割り当てられたデマンドと，その影響を受けて乗降時間が遅延したデマンドがリストされる．実際に遅延した時刻は，estimatedPickUp と estimatedDropOff の estimation に入る．

```
{
  "requestId": "98765",      // リクエスト ID
  "requestTimeStamp": "2016-12-27T11:50:30.45+09:00",
// (1)  現在の全 SAV の情報
  "availableVehicles": [
    {
      "savId": "1",
      "capacity": 6,         // 乗車定員
      "guests": 3,           // リクエスト乗客数
      "viaPoints": [         // 乗降場所リスト
        { "demandId": 1, "mode": "PICKUP" },
        { "demandId": 1, "mode": "DROPOFF" },
        { "demandId": 2, "mode": "PICKUP" },
        . . .
      ],
      "currentLocation": {  // 現在位置
        "latitude":45.500,  // 経度
        "longitude":140.450 // 緯度
      },
    }
    . . .
  ],
// (2) 新規配車リクエストのリスト
  "newDemands": [
    {
      "demandId": "10",
      "userId": "10",
      "accommodatePersonNumber": 1,
      "pickUp": {            // ピックアップ情報
        "position":{
          "latitude": 45.000,
          "longitude": 140.500
        },
                             // この時刻より後に来て欲しい
        "expectedTime": "2016-12-27T13:30:00.00+09:00"
      },
      "dropOff": {           // 降車（目的地）情報
        "position":{
          "latitude": 46.000,
          "longitude": 141.500
        },
                             // この時刻より前に着いて欲しい
        "expectedTime": "2016-12-27T13:50:00.00+09:00"
      },
      "shareable": true      // 乗合い OK
    },
    . . .
  ],
// (3) availableVehicle の全 viaPoints の全 demand
  "assigningDemands":[
    {
      "demandId": "1",
      "userId": "1",
      "accommodatePersonNumber": 1,
      "pickUp": {
        "position":          // ピックアップ位置
        "nearestLink":       // 地図上の位置
        "expectedTime":      // この時刻より後に来て欲しい
      },
      "dropOff": {
        "position":          // 降車位置
        "nearestLink":       // 地図上の位置
        "expectedTime":      // この時刻より前に着いて欲しい
      },
      "shareable": true,
      "savId": "1",
    },
    . . .
  ]
}
```

図 3 配車計算 API のリクエストメッセージ例（主要部分のみ抜粋）

4. おわりに

これまで SAVS の研究開発を進めてきて，見通しのよい開発，拡張性の高さ，サービスの仮想化を両立させようとした結果，マイクロカーネルと仮想マシンの考え方を汲むプラットフォーム・アーキテクチャに到り，Amazon Web Service (AWS) 上に実装中である．SAV 運転手に指示を出し乗客に様々なレベルのサービスを提供するのは，human-based computation の一種である．我々は今後も計算機科学の知見を活かして，より機能的かつ効率的な公共交通システムの構築を目指したい．配車エンジン中の配車計算ロジックと料金事前計算ロジックについては稿を改めようと思う．

```
{
  "requestId": "98765",
  "requestTimeStamp": "2016-12-27T11:50:30.45+09:00",
// (4) 配車計算によって変更を受けた全 SAV のリスト
  "assignedVehicles": [
    {
      "savId": "1",              // SAV ID
      "viaPoints": [             // 乗降場所リスト
        { "demandId": "1", "mode": "PICKUP",
          "estimatedPeriod":{
            "estimation":"2016-12-27T12:06:00.00+09:00"}
        },
        . . .
      }
    ]
  }
],
// (5) 新しく割り当てられたデマンドと変更を受けたデマンド
  "newAssignedDemands": [
    {
      "demandId": "10",
      "savId": "1",
      "estimatedRequiredMeter": 5000,   // 予想移動距離
      "estimatedRequiredMinutes": 30,   // 予想移動時間
      "estimatedPickUp": {              // ピックアップ時刻
        "from":          // 乗客指示の時刻枠の先頭
        "to":            // 乗客指示の時刻枠の終端
        "estimation":    // 実際に配車エンジンが予測した時刻
      },
      "estimatedDropOff": {             // 降車時刻
        "from": "2016-12-27T13:40:00.00+09:00"
        "to": "2016-12-27T13:45:00.00+09:00"
        "estimation": "2016-12-27T13:45:00.00+09:00"
      },
      "nearestLink": {
        "linkId": "192",
        "distance": 0.7
      }
    }
    . . .
  ],
  "assigningFailureDemands":[
    {
      "demandId": "11",
      "reason": "out of area",
      "code": 12345
    }
  ]
}
```

図 4 配車計算 API のレスポンスメッセージ例（主要部分のみ抜粋）

謝辞 本研究は，総務省戦略的情報通信研究開発推進事業 (SCOPE) 地域 ICT 振興型研究開発 162301003 の助成を受けています．SAVS の実装を担当しているアットウェア社の佐竹雅央氏と三嶋拓氏に感謝します．

参考文献

[1] Sonja Heikkilä, *Mobility as a Service – A Proposal for Action for the Public Administration: Case Helsinki*, Master's Thesis of Aalto University, Civil and Environmental Engineering (2014).

[2] 中島秀之, 小柴等, 佐野渉二, 白石陽, 平田圭二, 野田五十樹, 松原仁, Smart Access Vehicle System : フルデマンド型公共交通配車システムの実装と評価, 情報処理学会論文誌 Vol.57, No.4, pp.1290–1302 (Apr. 2016).

[3] 中島秀之, 野田五十樹, 松原仁, 平田圭二, 田柳恵美子, 白石陽, 佐野渉二, 小柴等, 金森亮, バスとタクシーを融合した新しい公共交通サービスの概念とシステムの実装, 土木学会論文集 *D3 (*土木計画学*)*, Vol.71, No.5, p.I_875-I_888 (Dec. 2015).

[4] 野田五十樹, 篠田孝祐, 太田正幸, 中島秀之, シミュレーションによるデマンドバス利便性の評価, 情報処理学会論文誌 Vol.49, No.1, pp.242–252 (2008).

[5] 佐野渉二, 落合純一, 平田圭二, 鈴木恵二, 野田五十樹, 中島秀之, デマンド応答型公共交通を用いたサービス連携プラットフォーム構築に向けて, (社) 情報処理学会 高度交通システムとスマートコミュニティ研究会 研究報告, 2016-ITS-66 (15) (Sep. 2016).