技術ノート

HTML5による車載HMI開発

Development of HTML5-based In-vehicle HMI

沢田　輝　*Teru SAWADA*
林　国勝　*Kokusyou RIN*
平嶋善隆　*Yoshitaka HIRASHIMA*

1 はじめに

近年、スマートフォンに代表されるモバイル機器の進化はめざましく、新機能や新サービスに対応した端末が次々と市場に投入されている。一方、車載機器は車両買い替えサイクルの長期化に伴い、搭載機能の陳腐化が顕在化しており、モバイル機器の進化との乖離が問題視されている。このため、自動車業界では車載機器に通信機能を搭載し、最新の情報やサービスの提供を通じて、車載機能の陳腐化防止を図ってきた。

昨今では、モバイル機器向けの技術やサービスを車載へ適用する動きが活発化しており、モバイルアプリケーションの開発技術を用いた車載機器開発が各社で進められている。

本稿では、汎用性の高さとオープンスタンダードな仕様により、モバイル業界のみならず自動車業界からも高い注目を浴びているHTML5（Hyper Text Markup Language Ver.5）を用いた車載アプリケーションの開発事例を紹介し、その有用性と課題について述べる。

2 テレマティクスシステムの進化と課題

1990年代後半から国内・海外で始まった自動車向け通信連携サービス（テレマティクスサービス）は、車載機に通信モジュールを搭載し、最新情報の提供や車両の盗難通知などの様々なサービスを提供している（**図1**：専用システム）。近年では、スマートフォンとの連携によるモバイルアプリの活用（**図1**：モバイル連携システム）や、モバイル機器で採用されている汎用OSの車載化（**図1**：汎用システム）による車載機器への直接アプリ配信など、様々な取り組みが進められている。

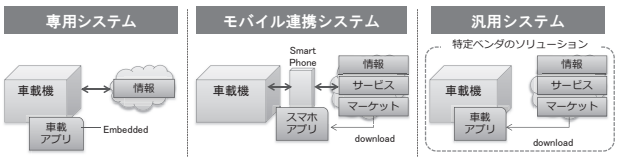

図1　テレマティクスシステムの構成
Fig.1 Configuration of telematics system

一方、現状のシステムには以下に示す課題がある。

- ●専用システム：個々のシステム仕様に依存した個別のサービス開発が必要
- ●モバイル連携システム：サービスを追加・改修するたびに膨大なモバイル機器との適合評価が必要
- ●汎用システム：OSやマーケットを提供する特定ベンダのビジネスモデルに支配される

HTML5は、W3C（World Wide Web Consortium）が標準化を推進するグローバルでオープンな標準規格であり、現状のテレマティクスシステムが持つ様々な課題を解決する技術として、自動車業界からも注目されている。

3 APIと車載アプリケーション開発事例

HTML5を用いた車載アプリケーションの開発に当たり、以下の３つのAPI（Application Programming Interface）を適用した。

- ●WebGL：車速などの環境に合わせて動的に変化する画面デザインを実現するための技術
- ●WebSocket：車載端末とセンターを連携させたアプリケーションを実現するための技術
- ●WebRTC：カメラやマイクなどの周辺機器を活用したアプリケーションを実現するための技術

3.1 WebGLについて

3.1.1 技術概要

WebGLは、コンピュータグラフィックスを扱うための規格であるOpenGL2.0 以上が利用できるプラットフォーム上で、Webブラウザから３次元グラフィックスの描画を可能にする標準仕様である。更に、このAPIを簡単に扱えるようにしたライブラリ（Three.js）も公開されており、Javascriptで立体的なアプリケーションが比較的容易に開発できる環境が用意されている。

3.1.2 試作アプリケーション

WebGLによる複雑で動的なデザインを実現した場合のドライバーの注意力散漫（Driver Distraction：以下DD）を評価するため、DDを考慮したUI（User Interface）を試作した。DDに影響を及ぼす外部因子としては、車速が最も重要なパラメータとなっており、現状の車載向けUI

では、車速検知によって「走行中」「停車中」の2モード切り替えを行う方式が一般的である。今回は、車速と連動して動的に変化するUIデザインを、WebGLを用いて実装した（**図2**）。

低速走行時　　高速走行時

図2　車速連動メニュー表示
Fig.2 Menu display linked to vehicle speed

3.1.3 有用性

車載向けの意匠部品（描画パーツ）は、専用の描画ツールで視覚効果を付与することが多いが、WebGLを利用することにより、比較的容易に描画処理が実現できる。そのため、従来の組込み系の開発と比べ、デザイン開発の効率化に寄与することができる。今回試作した車速と連動するUIデザインについても、Three.jsとJavascriptを用いるだけでメニュー表示の動的な視覚効果を実現することができた。

3.1.4 課題

車載機に標準的に搭載されるグラフィック・プロセッサ（GPU）の能力では、WebGLの持つ、光沢、視点効果などを適用すると、重なったアイコンのちらつき、フリック時の動作遅延などの事象が発生する。特に描画更新レートの高いアニメーションのレンダリングを行う場合（60fps以上）、GPUの負荷は非常に高くなってしまう。そのため、利用するハードウェアの性能を意識したチューニングが重要である。

実際の車載機への適用時は、描画更新レートを固定してGPUの負荷低減を図り、充分なレスポンスを確保することが必要になる。ナビゲーションの地図描画を行うような場合は、GPSの更新タイミングで描画を行えば良いため、レンダリング周期を大幅に低減することができる。

3.2 WebSocket

3.2.1 技術概要

WebSocketは、従来のWeb通信技術であるXMLHttpRequestやCometによる通信に比べ、サーバ負荷が少なく、同時に大量セッションを確立できることと、サーバ側から車載端末側へのPUSH通知を実現可能にする技術である。

3.2.2 試作アプリケーション

車載向けUIのデザインをHTML 5形式でセンターに配置することにより、車載機とセンターとの間で画面を共有することができる。さらに、PUSH通信に対応したWebSocketを使うことで、センター側から車載機の画面をリモート操作する（オペレータサービス）ことができる。今回は、コンテンツ配信のためのWebサーバ、相互通信連携のためのWebSocketサーバを用いて、リモートオペレータサービスを実現した。Webサーバ上（**図3**）に共有する画像ファイルや画面構成・遷移データを設置し、車載機と外部端末からそれぞれに応じたHTMLコンテンツへアクセスすることで、コンテンツの共有を行っている。また、操作時のタッチ座標、地図の中心座標情報、画面番号、ユーザID等をWebSocketサーバ経由で送受信し、相互の状態遷移を同期させている。

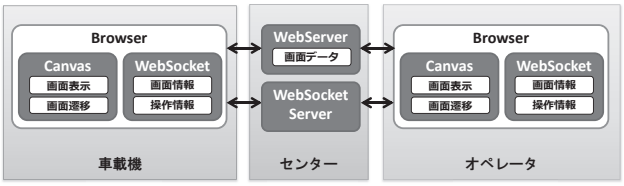

図3　WebSocketによる画面共有
Fig.3 Screen sharing using WebSocket

3.2.3 有用性

現状、車載端末の画面をオペレータが遠隔操作するようなシステムは、車載機側とオペレータ側が高度に連携する仕組みの実装が必要で、高級車向けの有償サービスとして定着している。しかし、WebSocketの機能を用いることによって、車載機側もオペレータ側も基本的にはHTML5に対応したブラウザを用意するだけで実現可能となるため、エントリクラスの車両向けにも容易に提供可能である。

3.2.4 課題

音声や映像といった比較的データ容量の大きなコンテンツは、データ送受信のためにサーバがデータリレーを行うため、通信データ量が増大し、通信速度低下とデータの遅延が起き、リアルタイム性が失われる。このため、サーバを経由しないP2P（Peer to Peer）通信が可能なWebRTC（後述）を適用することが望ましい。このように、取り扱うデータの容量に合わせた最適なAPIの選択が重要である。

3.3 WebRTC

3.3.1 技術概要

WebRTCはブラウザ上でリアルタイムコミュニケーションを可能にするAPIである。その仕様には、P2P通信を実現するPeerConnection API、P2Pで任意のデータの送受信を行えるDataChannel API、データストリームの取得

とカメラ、マイクにアクセスできるMediaStream APIが含まれる。

3.3.2 試作アプリケーション

WebRTCのうち、現在最も実装が進んでいるgetUserMedia（MediaStream APIでカメラ、マイクにアクセスできる機能）を利用した、車載デバイスとの連携アプリケーションを試作した。ブラウザからローカル接続されているカメラ映像を取得し、車載搭載カメラの映像をHTML5アプリケーションでブラウザ上に表示する。このカメラ映像をCanvasを使って特定のレイヤへ描画することで、これまで組み込み実装が一般的だったカメラ画像処理をHTML5を用いてブラウザ上で実装することが可能になる。また、取得したカメラ映像と自車位置（HTML5のAPIの一つであるGeoLocationにより位置情報を取得）を関連付けることでARナビゲーションを容易に実現することが可能である。上記技術を活用したARナビアプリケーション（**図4**）と、描画レイヤの構成（**図5**）を下記に示す。

図4　WebRTC, CanvasによるARナビの画面表示例
Fig.4 Example of screen display of AR navigation system using WebRTC and Canvas

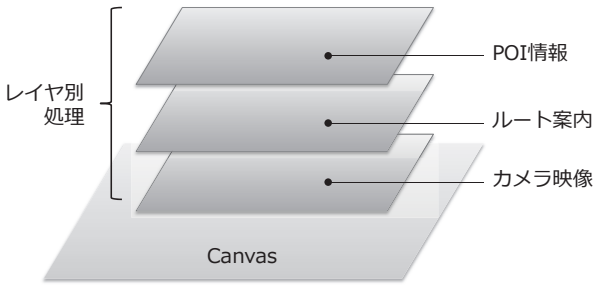

図5　ARナビの構成
Fig.5 Configuration of AR navigation system

3.3.3 有用性

従来、カメラ画像の取り込みや画像処理を車載機で行うためには、専用のハードウェアと組み込みの画像処理アプリケーションを実装する必要があった。

一方、WebRTCを用いることでブラウザレベルでカメラ画像の処理や様々な描画パーツとの重畳表示が可能になり、ARナビゲーションやバックガイドモニタなどのアプリケーションが比較的容易に実現できる。また、車載機とカメラを接続するインタフェースさえ実装しておけば、様々なバリエーションのカメラ活用サービスをHTML5のアプリケーションとして配信することができるため、車載カメラの付加価値向上に対しても有用である。

3.3.4 課題

getUserMediaで取得したカメラ映像に対してナビ情報などの重畳処理を行っているが、その処理量が増加すると、映像のフレームレートを確保できなくなるため、Javascriptの最適化や、アプリケーションに最適なフレームレートに調整する必要がある。

また、現状のブラウザ標準ではgetUserMediaでアクセスできるカメラは２台までであり、リアカメラ以外にも、フロントカメラ、サイドカメラなどが搭載されている場合には対応できない。他にもVGAを越える高解像度カメラへの対応などの課題が残存している。

4　おわりに

HTML5の持つ様々な技術のうち、HTML5対応車載機の実現に向けて特に重要と考えられる、WebGL、WebSocket、WebRTCについて、その有用性と課題について述べた。

HTML5に関する仕様は、本稿執筆時点ではまだ最終勧告に至っていない。しかし、W3Cを中心とした業界全体で標準化と実装が確実に進んでおり、車載向けHTML5の仕様検討も始まっている。

本稿でも述べたように、車載機器には固有の開発要件があり、この要件への対応力が車載機器メーカ各社の腕の見せ所でもある。当社では、今後も引き続きHTML5による車載アプリケーションの試作開発を行い、ノウハウの蓄積を図っていきたい。

筆者紹介

沢田　輝
（さわだ　てる）
SS技術本部サービス技術室
室長

林　国勝
（りん　こくしょう）
SS技術本部サービス技術室

平嶋　善隆
（ひらしま　よしたか）
SS技術本部サービス技術室