

コンシューマ・システム論文

# 軽量スクリプト言語を用いた 自動車ソフトウェア遠隔更新制御方式の検討

寺岡 秀敏<sup>1,a)</sup> 山崎 裕紀<sup>2</sup> 櫻井 康平<sup>2</sup> 船迫 陽介<sup>3</sup> 尾崎 友哉<sup>1</sup>

受付日 2018年2月28日, 採録日 2018年8月3日

**概要:** 近年, 自動車分野においても機能追加やセキュリティリスクへの対策などのため, 電子制御ユニット (ECU) に搭載されたソフトウェアを販売後に更新する機会が増大している. そのような中で, デジタルテレビや携帯電話などのコンシューマエレクトロニクス分野で実用化されている無線による遠隔更新技術 (Over the air firmware update: OTA) の自動車システムへの適用が検討されている. OTA による遠隔更新では, 更新処理の自動化が必要でありそのための更新制御技術が重要となる. 本稿では, 自動車システム向けの OTA 更新制御についての要件を整理し, 整理した要件を満たす機能配置を提案する. また, 自動車システム特有の多様性に対応した OTA システムの構成と更新制御方式を提案する. 提案方式を車載ゲートウェイ用マイコン上に実装し, 提案方式が搭載可能であることを検証する.

キーワード: 車載 ECU, OTA, 更新制御, 軽量スクリプト

## A Study of OTA Update Control Method for Vehicle System Using Lightweight Script Language

HIDETOSHI TERAOKA<sup>1,a)</sup> HIROKI YAMAZAKI<sup>2</sup> KOHEI SAKURAI<sup>2</sup>  
YOUSUKE FUNASEKO<sup>3</sup> TOMOCHIKA OZAKI<sup>1</sup>

Received: February 28, 2018, Accepted: August 3, 2018

**Abstract:** Recently, the increase of amount of program code on Electronic Control Units (ECUs) in vehicles causes the increase of firmware update after sales which stems from bugs in the program code. In this situation, automakers are studying to introduce over the air firmware update (OTA) technology which is used in digital TV and mobile phone. In this paper, we clarify functional requirements for OTA update control for a vehicle system and propose an update control method for OTA. We implement the method on microcontroller for in-vehicle gateway and show that the proposed method is applicable to automotive field.

**Keywords:** In-vehicle ECU, OTA, update control, lightweight script

### 1. はじめに

#### 1.1 背景

Advanced Driver Assistance System (ADAS) や自動運

転の導入により, 自動車の価値を差別化する要素として, ソフトウェアの価値が高まっている. そのような中, テスラモーターズ社は, 2015 年 8 月には販売済みの車両に対して, ソフトウェア更新によって有料で自動運転機能を追加するという, 新たなビジネスモデルを試行している [1].

現在, 自動車が販売された後に車載 ECU のプログラム更新を行う場合, 顧客 (自動車所有者) がディーラに車両を持ち込み, ディーラの整備士が専用ツールを使ってマニュアル作業で 1 台 1 台 ECU 上のプログラムを書き換えることが一般的である. しかし, この方法は, 車両の持ち込みなど時間がかかることでユーザの利便性が悪い. この問題

<sup>1</sup> 株式会社日立製作所研究開発グループ  
Hitachi, Ltd. Research & Development Group, Yokohama, Kanagawa 244-0817, Japan

<sup>2</sup> 日立オートモティブシステムズ株式会社  
Hitachi Automotive Systems, Ltd., Hitachinaka, Ibaraki 312-8503, Japan

<sup>3</sup> 株式会社日立産業制御ソリューションズ  
Hitachi Industry & Control Solutions, Ltd., Yokohama, Kanagawa 244-0817, Japan

a) hidetoshi.teraoka.rf@hitachi.com

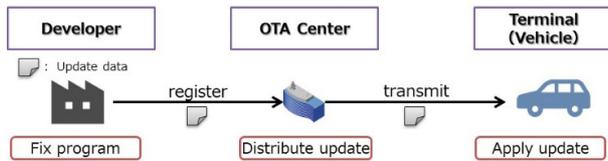


図 1 OTA システム構成  
Fig. 1 System configuration of an OTA system.

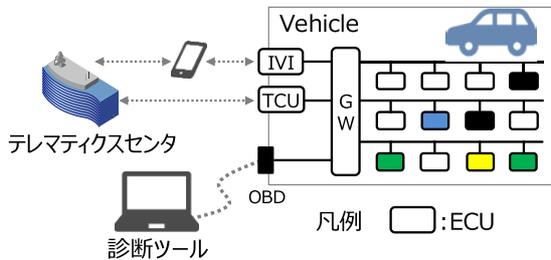


図 2 自動車システムの構成  
Fig. 2 Configuration of vehicle system.

を解決するため、車載 ECU に対して遠隔ソフトウェア更新を適用し、プログラム更新時のユーザ利便性を向上させることが検討され始めている [2].

携帯電話網などの無線接続を利用してソフトウェア/ファームウェアを端末に配信し、遠隔で更新する技術は OTA (Over the Air software/firmware update: 無線によるソフトウェアの更新) と略称される。デジタルテレビや携帯電話分野では OTA は一般化され、多量の端末のソフトウェアが効率的に更新されている。OTA システムを自動車に適用する場合、新プログラムを OTA センタに登録し、OTA センタは車両に修正した新プログラムを配信する。車両では、新プログラムを受信し、これを更新対象 ECU に適用する (図 1)。

近年、車載機 (カーナビ) においても、遠隔での地図やソフトウェアの更新が導入されている。前述の、テスラモーターズ社の取り組みも、OTA を車両に適用することで実現されている。

### 1.2 自動車システムの構成

図 2 に OTA によるソフト更新の対象となる自動車システムの構成を示す。図中の ECU の色が異なるのは、ハードウェアリソースなどの特徴の異なる ECU であることを示す。

これまで OTA が適用されてきたコンシューマエレクトロニクス製品と異なり、自動車システムは 1 つ 1 つが携帯端末に相当するような多数のデバイス (ECU) で構成され、これらが連携して機能を実現する複雑なシステムである。自動車 1 台あたりの ECU の搭載数は 2025 年には平均 30.4 個になると予想されている [3].

従来の ECU に加えて、カーナビやオーディオなどの機能を備える In-vehicle infotainment (IVI) や車両外部との

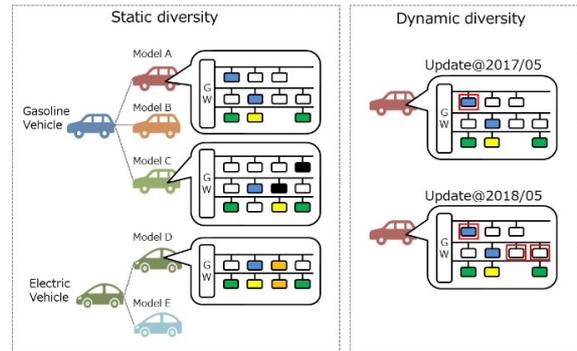


図 3 自動車システムの多様性  
Fig. 3 Overview of diversity of vehicle configuration.

通信機能を備える Telematics Control Unit (TCU) が搭載され、テレマティクスセンタと通信することで様々なサービスが提供されている。

また、車両には整備士が診断ツールを接続して故障診断などを行うための On-board diagnostic (OBD) コネクタが備えられ、現在、ECU のソフトウェア更新は本コネクタ経由で診断ツールから実行される。

近年、これら車両外部とのインタフェースからのサイバー攻撃の事例 [4], [5] が報告されており、その対策として IVI, TCU や外部接続機器および OBD のような外部インタフェースとの間で Firewall として機能するセキュリティゲートウェイの搭載が進みつつある。

### 1.3 自動車システムのソフトウェア更新における特徴

ソフトウェア更新の対象となる ECU の組合せは、電気自動車やガソリン自動車などの車種によっても異なる。このため、システム構成として 1 デバイスの場合が多いコンシューマエレクトロニクス分野の製品と比較すると、ソフトウェア更新対象は非常に多様である。

図 3 に示すとおり、自動車システムのソフトウェア更新は静的・動的の両面で多様である。すなわち、ガソリン車、電気自動車といった車種により ECU の構成が異なる。さらに、同じガソリン車でもモデルにより ECU 構成が異なる (静的多様性)。たとえば、車種による違いにともない、ガソリン車の更新はイグニッション OFF で、電気自動車の更新は充電中に、など更新条件が異なる可能性がある。また、同じ車種でも構成要素となる ECU ごとに、それぞれ更新条件や更新手順が異なる可能性がある。他にも、同じ車両でも、1 回目の更新と 2 回目の更新では更新対象となる ECU が異なる可能性もある (動的多様性)。たとえば、1 回目の更新では単一の ECU を更新するが、2 回目の更新では複数の ECU を更新する必要がある、などのバリエーションが考えられる。

### 1.4 課題と目標

前述のとおり、自動車のソフトウェアを更新する場合、

従来はディーラーで整備士が診断ツールを使ってマニュアルで実施していた。整備士の作業としては、たとえば開始前の車両の状態チェック、更新対象 ECU と更新データの取得、実行中の車両状態の監視、完了後の結果確認などがある。自動車システムに OTA を適用する場合、これらの更新手続きをシステムが自動で実行する必要がある。以下、このような更新手続きの実行を更新制御と呼ぶ。

本研究では、自動車システムに OTA を適用するための更新制御についての機能要件を整理し、これらの機能要件を満たしつつ、さらに自動車システム特有の多様性への対応が容易な OTA システムの機能配置と制御方式を提案する。

## 2. 関連研究

OTA によるソフトウェアの更新制御に関して、de Boerらは ECU ごとの更新手順の相違への対応という課題に対し、Head Unit に OSGi フレームワーク [6] を搭載して ECU ごとのシーケンス制御ソフトをバンドル (Java のアプリケーションプログラム) として配信する方式を提案している [7]。しかしながら、近年重要となっているサイバーセキュリティ対応の自動車システム構成を想定した機能配置や提案方式の運用容易性については考慮されていない。

また、Ryu らは、テレマティクスユニットに OMA DM クライアントを搭載し、複数の ECU で構成される自動車の構成管理や、ソフトウェアのアップデートを行うアーキテクチャを提案している [8]。しかしながら、ソフトウェア更新に関して、具体的な振舞いや OMA DM の規定範囲外である更新のためのパッケージの内容については考慮されていない。

他にも、Zhang らは、携帯電話を経由して ECU ソフトウェアの更新パッケージをダウンロードし、ECU を更新する Onboard platform を用いた OTA ソフト更新システムを提案している [9]。しかしながら、車載システムにおける更新制御機能の要件についての整理やそれに基づく構成の提案はなされていない。

本研究では、前述の研究をふまえ、自動車システムへの OTA 適用を実現する更新制御方式についての機能要件を整理し、これを実現するシステム構成を提案する。特に、自動車システム特有といえる前述の多様性に着目し、ECU ごとの更新手順の相違に加えて、車種などの多様性への対応も可能な制御方式を提案する。

## 3. 自動車システム向け OTA 更新制御機能

図 4 に、本研究で提案する自動車向け OTA ソフトウェア更新システムの全体像を示す。本研究では、更新制御機能は車載ゲートウェイに配置し、そこで OTA センタから取得するパッケージに含まれる更新スクリプトに基づいて制御を実行する構成を提案する。更新スクリプト

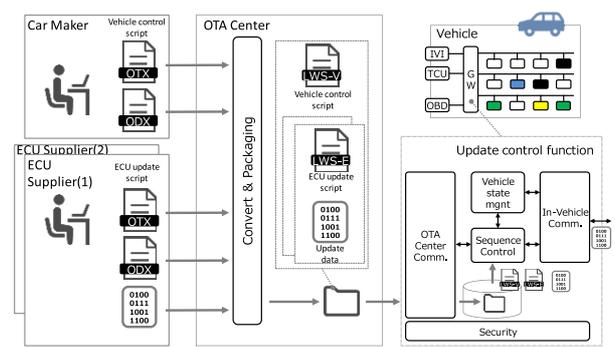


図 4 自動車向け OTA ソフトウェア更新システム全体像  
Fig. 4 Overview of OTA software update system for automotive.

表 1 更新制御機能の要件

Table 1 Requirements of OTA update control function.

Req#	内容
1	更新シーケンス制御が可能であること。
1-1	シーケンス制御は多様なシステムに対応可能な柔軟性を持つこと。
2	車両状態の把握が可能であること。
3	外部からの更新データの取得や外部との構成情報の同期可能であること
4	セキュリティ機能を備えること

は、ISO 標準である Open Test sequence eXchange (OTX) (ISO13209) [10], [11] および Open Diagnostic data eXchange (ODX) (ISO22901) [12] で記述されたシーケンスおよびパラメータをゲートウェイ上で実行可能な軽量な形式に変換されたものを用いる。ここで、OTX/ODX ファイルは自動車メーカーや ECU の部品メーカーにより作成され、OTA センタで軽量スクリプトに変換されて新プログラムそのものや差分データとともにパッケージに同梱される構成を想定する。スクリプトの作成が自動車メーカーと部品メーカーで行われるのは、自動車全体や複数の ECU にまたがる部分は自動車メーカー、ECU 固有の部分は部品メーカーが担当することを想定するためである。

本章では最初に更新制御機能の要件を整理し、次に、要件に基づく機能配置についての検討結果を述べる。その後、自動車システム特有の課題といえる多様性に対応するための更新シーケンス制御の柔軟性確保の方法についての検討結果を述べる。

### 3.1 OTA 更新制御機能の要件

表 1 に更新制御機能への要件を示す。

#### (1) 更新シーケンス制御

更新制御機能は、車両状態などに基づいてソフトウェア更新のプロセスの開始・中断や、Controller area network (CAN) などの車内ネットワークを介した ECU への更新開始指示、更新データ転送および ROM 書き換えなどの制御を行う必要がある。1.3 節に示したような多様性を持つ

自動車システムに対して OTA を適用する場合、更新制御機能がガソリン車や電気自動車などの車種に起因する手順の相違や、ECU の違いに起因する手順の相違に柔軟に対応可能であること (Req#1-1) が特に重要となる。本課題については、3.3 節で詳細要件の整理を行う。

(2) 車両状態の把握

更新制御機能は、前述のシーケンス制御を行うため、また、車両状態を OTA センタと同期するため、車両状態を把握できる必要がある。ここでの車両状態には、イグニッション OFF/ON、電源状態、走行状態、ユーザとのインタラクション結果、有線/OTA でのソフトウェア更新状態などが含まれる。

(3) 更新データの取得および構成情報の同期

更新制御機能は、更新対象 ECU に適用する新プログラムや付加情報など更新用のデータ (以下、パッケージと称する) を外部から取得する必要がある。また、必要に応じて取得したデータを蓄積する必要がある。他にも、更新の要否を判断するため、ECU のソフトウェアバージョン情報などの構成情報を外部とやりとりする必要がある。

(4) セキュリティ機能

外部からの攻撃などにより更新制御機能ののっとりや不正動作が引き起こされると、不正なプログラムの書き込みにより ECU の不正動作や機能停止が引き起こされる可能性がある。これらは自動車システムの安全性に関わるため、更新制御機能は、OTA センタとの通信などのセキュリティを確保し、取得した更新データの正当性を保証する必要がある。

3.2 OTA 更新制御機能の配置

本研究では、前述の要件を満たす更新制御機能の配置先としてゲートウェイを提案する。これは、表 3 の機能配置に関する比較結果に基づく。比較の観点と比較観点とした理由は以下のとおりである。

(A) ECU へのアクセス性

車両状態の把握および更新シーケンス制御のため、車両内の ECU へのアクセス性が容易であることが必要となるため。本評価では、更新制御機能から ECU までの経路となるコンポーネント数に基づき比較する。

(B) セキュリティ

更新制御にあたっては、セキュリティ確保が必要であるため。本評価では、JASO TP-15002 [13] に規定された脅威分析を行い、Common Vulnerability Scoring System (CVSS) [14] に基づくリスク評価手法である CVSS based Risk Scoring System (CRSS) 方式を用いて抽出したリスクの基準値 (CVSS 基本値) の最悪値に基づいた比較を行う。ここで、基本評価値のパラメータは、JASO TP-15002 に記載の値を用いる。また、保護資産を「OTA 更新制御機能」および「パッケージ」の 2 つとする。

表 2 車載デバイスの想定リソース

Table 2 Assumed resources of in-vehicle devices.

車載デバイス	RAM	ROM	CPU
IVI	数 GB	数十 GB	数 GHz
TCU	数百 MB	数百 MB	~GHz
Gateway	数百 KB	数 MB	数百 MHz
ECU*	数十 KB ~数 GB	数百 KB ~数 MB	数十 MHz ~GHz

\*ECU は用途により多様な構成が考えられる。

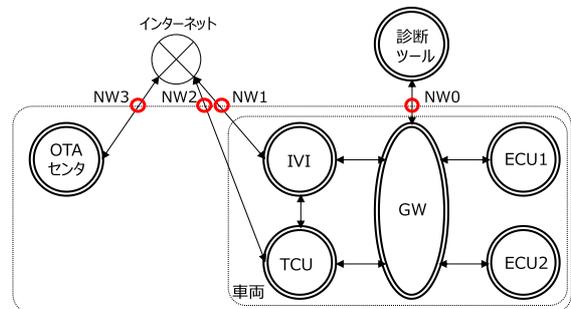


図 5 OTA システムモデル

Fig. 5 A model of OTA system.

表 3 更新制御機能の配置先比較

Table 3 Comparison of location of OTA update control function.

配置先	(A)	(B)	(C)	(D)
OTA センタ	3	8.5	0	大
IVI	2	8.5	1	中
TCU	2	8.5	1	中
Gateway	1	7.3	2	小
ECU	0	6.4	3	小

(C) OTA センタへのアクセス性

OTA センタからのパッケージの取得、車両状態の OTA センタとの同期のためには OTA センタへのアクセス性が容易であることが必要となるため。本評価では、更新制御機能から OTA センタまでの経路となるコンポーネント数に基づき比較する。

経路となるコンポーネント数のカウントおよびセキュリティリスクの評価に用いたモデルを図 5 に示す。

(D) ハードウェアリソース (CPU, メモリ)

更新制御はソフトウェアで実現することを想定した場合、更新制御機能が搭載可能なリソースを持つことが必要である。一例として、関連研究 [7] の提案手法を用いて更新制御機能を実現する場合、数 MB の RAM, ROM が必要になると考えられる。本稿で想定する車載デバイスのリソースを表 2 に示す。

更新制御機能を OTA センタに配置する場合、ECU までの経路上のコンポーネント数は 3 と最も多く、ECU へのアクセス性が悪いといえる。特に、OTA センタと車両の間の通信の安定性が保証されていない場合、更新制御中に ECU へのアクセスが遮断されて更新が継続できなくなるリスクがある。OTA センタに配置した場合の CVSS 基本

値の最悪値は8.5であり、インターネット (NW3) から更新制御機能に不正な動作を引き起こす攻撃のリスクが最も高い値となった。一方で、パッケージはOTAセンタで生成・管理されるため、経路上のコンポーネント数は0と最も少なく、データ取得は容易である。また、組み込み端末であるそのほかの配置先と比較した場合、十分なリソースがあるといえる。

更新制御機能をIVIに配置する場合、ECUまでの経路上のコンポーネント数は2であり、ECUへのアクセスはOTAセンタと比較して良好といえる。本配置におけるCVSS基本値の最悪値は8.5であり、インターネット (NW1) から更新制御機能に不正な動作を引き起こす攻撃のリスクが最も高い値となった。本モデルでは、IVIは携帯電話を用いたテザリングなどで外部ネットワークに接続されており、OTAセンタにアクセスするための経路上のコンポーネント数は1であり、データ取得は容易であるといえる。また、地図やユーザインタフェースを備えるIVIはそのほかの配置先と比較した場合、OTAセンタほどではないがリソースには余裕があるといえる。

更新制御機能をTCUに配置する場合、ECUやOTAセンタへのアクセス、セキュリティリスクはIVIと同等である。また、近年のLTEなど高速回線に接続する機能を持つ場合、そのほかの配置先と比較した場合、OTAセンタやIVIほどではないが、リソースは多いといえる。

更新制御機能をゲートウェイに配置する場合、ECUまでの経路上のコンポーネント数は1であり、ECUへのアクセスは非常に良好といえる。本配置におけるCVSS基本値の最悪値は7.3であり、IVIなどを經由して遠隔 (NW1など) から更新制御機能に不正な動作を引き起こす攻撃のリスクが最も高い値となった。すなわち、IVIなどのように直接インターネットに接続し、遠隔での攻撃を直接受ける可能性が少ないため本配置のほうがセキュリティ上優位であると考えられる。また、元々の外部からのセキュリティ保護のためのコンポーネントとしての位置づけから、ハードウェアセキュリティモジュールなどを備え、攻撃への対策が行いやすいといえる。一方で、外部ネットワークとの接続はTCUやIVIを介して行う必要があるため、OTAセンタまでの経路上のコンポーネント数は2となり、データ取得は少し困難となる。また、現在の車載ゲートウェイ用マイコンのリソースはTCUほど潤沢ではない場合が多い。

更新制御機能をECUに配置する場合、経路上のコンポーネント数は0である。ゲートウェイで保護された領域に配置することになるため、セキュリティ保護は良好であり、CVSS基本値の最悪値は6.4である。外部ネットワークへの接続はゲートウェイおよびTCUまたはIVIを經由する必要があり、経路上のコンポーネント数は3となるため、OTAセンタへの接続性は悪い。また、大半のECUでは、本来の制御機能以外を動作させるためのリソースの余裕

がない場合が多いと考えられる。さらに、更新対象となるECUそれぞれに更新制御機能を搭載することは機能重複が多くなりコストアップにつながる。

以上の比較結果から、配置先として優位であるのはIVI、TCU、ゲートウェイであると考えられる。本研究では、低価格帯の車両などIVIが搭載されない車両がありうること、ソフトウェア更新にあたってはセキュリティが重要であり、そのためのセキュリティを確保しやすい点を考慮し、ゲートウェイを更新制御機能の配置先として提案する。ゲートウェイにはリソースが少ないという課題があるため、少ないリソースで実現可能な更新制御方式を検討する。

### 3.3 OTA更新制御の柔軟性確保

本節では、3.1節の(1)において述べた更新シーケンス制御機能の柔軟性に関する要件 (Req#1-1) が、さらにどのような要件から構成されるかを詳細化し、これらを満足する更新制御方式の検討を行う。

#### (1-1-1) 処理実行有無の制御

各シーケンスにおいて、ECUの異常状態を示すDiagnostic trouble code (DTC)の確認をする/しない、ダウンロードしたソフトウェアの適用可否についてユーザの許諾確認をする/しないなどの処理実行有無の制御が必要となる。

#### (1-1-2) 状態判定閾値の変更

ソフト更新可否判断のための電池残量などの判定条件の変更などが可能である必要がある。

#### (1-1-3) 処理順序の入れ替え

シーケンス内の処理順序の変更が可能である必要がある。

#### (1-1-4) 新規処理の追加

状態確認対象ECUの追加やその手順の追加が可能である必要がある。

(例：ガソリン車：イグニッション状態の確認。電気自動車：イグニッション状態の確認+充電状態の確認)

#### (1-1-5) 省リソース

車載ゲートウェイに搭載するため、省リソースで動作する必要がある。

#### (1-1-6) 運用容易性

更新制御機能において様々なシーケンスの実行を実現することは、当該シーケンスの生成・管理が必要になり運用が複雑になる可能性があるため、運用が容易である必要がある。

多様性に対応するために、それぞれの車種・モデルごとに更新制御ソフトウェアを開発する方法が考えられるが、それには開発コストの上昇という問題がある。

そこで、このような問題に対応するため、前述の要件を満たす方法として、以下の2通りの制御方式を比較した。

#### (a) パラメータによる制御方式

OTAセンタからダウンロードするパッケージに、更新

表 4 OTA 更新のシーケンス制御における要件

Table 4 Requirements of OTA update sequence control.

Req#	内容	(a)	(b)
1-1-1	処理実行有無の制御	○	○
1-1-2	状態判定閾値の変更	○	○
1-1-3	処理順序の入れ替え	×	○
1-1-4	新規処理の追加	×	○
1-1-5	省リソース	○	×
1-1-6	運用容易性	○	×

制御に関するパラメータを設定し、更新制御ソフトウェアはパラメータに基づいて制御を行う。

(b) スクリプトによる制御方式

OTA センタからダウンロードするパッケージに、シーケンスを記述したスクリプトを同梱し、更新制御ソフトウェアは当該スクリプトを実行して制御を行う。

表 4 に、両方式の比較結果を示す。

パラメータ方式は、リソースの消費が少ないが、順番の入れ替えや新規処理の追加など要件を満たせない場合がある。また、同一車両において更新キャンペーンごとに異なる可能性のあるパターンも網羅してパラメータ設計を行うことは困難である。一方で、限られた範囲での柔軟性となるため、パラメータシートを作成して管理することができるなど、運用は容易であるといえる。

一方、スクリプトを用いる方式は、要件#1-1-1~1-1-4を満たすが、小リソースな機器上での実行は困難な可能性がある。また、柔軟性が高く処理内容を自由に定義できるため、作成や管理への負担が大きくなる可能性があり、運用が困難であると考えられる。

本研究では柔軟性に対する要件の満足性が高いと考えられるスクリプト方式について、運用が容易かつ小リソースな機器上で実行可能となる更新制御方式を検討する。

3.4 ISO 標準技術によるスクリプト記述

整備士が、自動車システムの診断や、ECU のソフトウェアを更新する際に用いる専用ツールやツールと ECU の通信インタフェース向けの規格として UDS (Unified Diagnostic Service) (ISO14229) [15], ODX, OTX が標準化されている。

UDS は診断装置と ECU 間の通信規格を定めている。具体的には、CAN など接続した ECU との間で送受信するデータを定義するものであり、診断処理 (サービスと呼ばれる) ごとにバイナリフォーマットが規定されている。

ODX では前述の UDS で規定された診断サービスで利用するデータの定義を行う記述方式が規定されている。具体的には、ECU に書き込むデータについて、書き込み先のアドレスやサイズ、データ形式などが記述される。他にも、ECU から情報を取得する処理やリセットを行う処理など ECU のソフトウェアを更新する処理で扱うデータの内容

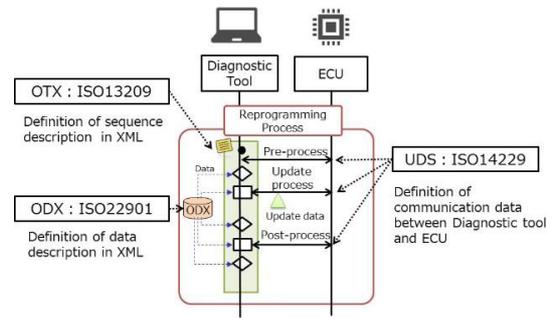


図 6 ISO 標準の概要

Fig. 6 Overview of ISO standards.

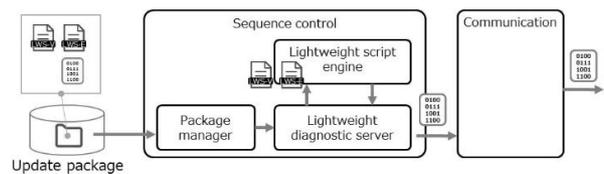


図 7 更新制御ソフトウェアアーキテクチャ概略

Fig. 7 Overview of update control software architecture.

が含まれる。これらは XML で記述される。

OTX は、診断サービスの処理シーケンスを記述するための標準技術である。具体的には、前述の UDS で規定され、ODX でデータが記述された診断サービスを、どの順番で実行するか、ECU と通信した結果によって条件分岐する、というような処理の流れを XML で記述する。

図 6 に、これら ISO 標準技術の概要を示す。

このような、標準化された記法で手順を記述することにより自動車メーカーや自動車メーカーと部品メーカー間で更新制御手続きの共有が容易になる。また、これらをオーサリングする市販ツール [16] があるため、記述にあたって市販ツールを利用することで、市販ツールを利用して従来運用されている手順と同様の運用を行うことが可能となる。

一方で、これら標準は XML での記述を前提としているため、リソースの少ない車載ゲートウェイ上で解析・実行することは困難である。そこで、本研究では ISO 標準で作成されたスクリプトを、OTA センタで軽量スクリプトに変換して車載ゲートウェイ上で実行する方式を提案する。

3.5 軽量スクリプト言語を用いた車載ゲートウェイアーキテクチャ

図 7 に、車載ゲートウェイ上の更新制御ソフトウェアのアーキテクチャ概略を示す。

Package manager は、更新パッケージを解析し、スクリプトや ECU への送信データを分離する。

Lightweight diagnostic server は、更新パッケージに含まれるスクリプトを読み出し、Lightweight script engine を呼び出す。また、軽量スクリプトに ECU への命令を抽象化した API を提供し、軽量スクリプトからの呼び出しに

応じて UDS コマンドの生成や受信した応答の解釈を行い、結果をスクリプトに伝える。さらに、更新パッケージに含まれる ECU への送信データを取得し、軽量スクリプトからの指示に従って ECU に送信する。

Lightweight script engine は軽量スクリプトの実行環境であり、OTA センタから受信したスクリプトを実行する。

このように UDS コマンドの構築・解析といったバイナリデータ処理など、負荷が高い処理は C 言語で実装した Lightweight diagnostic server に配置し、比較的処理負荷が小さいシーケンス制御などをスクリプト側の処理とすることで、柔軟性と処理負荷の低減を両立できる。

### 4. 評価

更新制御機能における要件の実現性を検証するため、提案方式の評価を行った。表 5 に機能要件とそれに対する評価内容を示す。本研究では、センタとの接続機能については、評価の対象外とし、ECU とのインタフェースを持つ機能を中心に、更新制御機能による車載システム制御の実現性を評価した。スクリプトとしては、組み込みスクリプト言語の中でも軽量の Lua [17] を利用した。また、セキュリティアルゴリズムとしては、パッケージの情報秘匿に用いる共通鍵暗号、認証および改ざん検知に用いる公開鍵暗号、ハッシュアルゴリズムとしてそれぞれ AES (CTR モード、鍵長 128 bit), RSA (鍵長 3,072 bit), SHA256 を評価した。評価にあたっては、OpenSSL [18] をベースに実装した。

#### 4.1 評価環境

図 8 に評価環境の外観を示す。提案方式の車載ゲートウェイ搭載可否を評価するため、車載ゲートウェイ向けのマイコンに前述のアーキテクチャのソフトウェアを実装した。車載ゲートウェイ向けマイコンとしてルネサスエレクトロニクス社製の RH850/F1L [19] を用い、これをテセラ・テクノロジー社製のマイコン評価ボード FL-850/F1L-176-S [20] に搭載して評価環境を構築した。また、更新対象となる ECU は同じ車評価環境を利用し、車両状態を模擬するアプリケーションは PC 上のアプリとして実装して評価した。

表 5 更新制御機能の要件と評価内容

Table 5 Requirements of OTA update control function and evaluation.

Req#	要件	評価内容
1	更新シーケンス制御	(a)スクリプトでの更新シーケンス実行に必要なメモリ使用量 (b)スクリプトでの ECU 更新時間 (c)スクリプト変換処理の実現性
2	車両状態の把握	(a)スクリプトでの車両状態チェックに必要なメモリ使用量
3	更新データ取得と構成同期	本研究の評価対象外とする。
4	セキュリティ	(a)パッケージを保護するセキュリティアルゴリズムに必要なメモリ使用量

評価用ゲートウェイと PC は、Vector 社製の VN1610 [21] を用いて CAN で接続した。CAN の通信帯域は 500 Kbps を利用した。表 6 に評価環境の詳細を示す。

評価用車載ゲートウェイ、更新対象 ECU および疑似車両アプリの機能配置を図 9 に示す。車載ゲートウェイにはスクリプトで規定された手順に従い更新シーケンスの制御を行う更新シーケンス制御部、ISO15765-2 に対応した CAN 通信を行う CAN ミドルおよび CAN ドライバを搭載する。更新制御部は、提案のとおり Package manager, Lightweight diagnostic server および Lightweight script engine で構成する。本研究では、パッケージはダウンロードされた状態を想定してあらかじめ ROM に配置し、これを読み出す。Lightweight diagnostic server を AUTOSAR OS 上の 1 タ

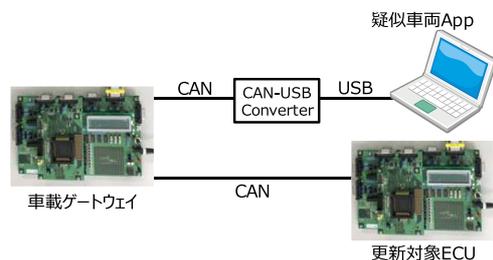


図 8 評価環境外観

Fig. 8 Appearance of evaluation platform.

表 6 評価環境

Table 6 Specification of evaluation platform.

項目	値と説明	
Gateway and Target ECU	CPU	80MHz Renesas RH850/F1L
	ROM	2MB
Pseudo vehicle application	CPU	Intel(R) Core(TM) i5-6500 CPU @ 3.2GHz
	RAM	4.00GB
	OS	Windows 7
CAN	Bandwidth	500Kbps HI-Speed CAN 帯域 100%利用
	Protocol	ISO14229(UDS), ISO15765-2[22]
	Parameter	Separation time=0, Block size=1
Lightweight script engine	Lua5.3 (virtual machine with standard libraries)	
Update data	走行系 ECU 制御ソフトウェアの差分データ (45,178byte)	
Security algorithm	秘匿	共通鍵暗号 AES128
	認証および改ざん検知	公開鍵暗号 RSA3072
		ハッシュ SHA256

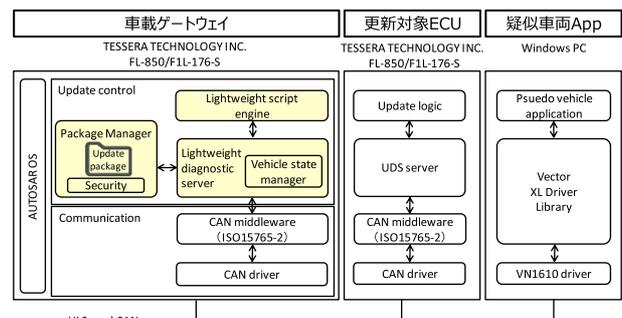


図 9 評価環境機能配置

Fig. 9 Functional layout of evaluation platform.

スクとして実装し、これがライブラリとしての Lightweight script engine と Package Manager を呼び出す構成とした。Lightweight script engine には、script から呼び出される API として、バージョン情報取得や ECU のモード変更、更新データ転送、ソフトウェア更新の結果確認要求などを実装する。ここで更新データを転送する API は、UDS に規定されたデータ転送のサービス RequestDownload, TransferData, RequestTransferExit の順次発行するように実装し、script に記述が必要となるステップを削減するようにした。更新制御時の評価にあたっては、Lightweight diagnostic server は外部からのトリガによって Package manager を呼び出し、実行するスクリプトを抽出する。次に、Lightweight script engine を呼び出して抽出したスクリプトを実行し、車両状態を把握するシーケンスや ECU 更新のシーケンスを実行する。本研究では、車両状態の把握のための Vehicle state manager を Lightweight diagnostic server 内の 1 機能として実装し、取得した車両情報の管理を行う。また、セキュリティアルゴリズムは Package manager の一部として実装した。ただし、パッケージの検証と復号は ECU 更新とは別にダウンロード時などに行われることを想定し、ECU 更新時の処理には含めない構成とした。

更新対象 ECU には、同様の通信ミドルウェアおよび車載ゲートウェイからの指示に従って更新を実行する機能を搭載する。更新対象 ECU 内の UDS server は車載ゲートウェイから発行されるコマンドの解析を行い、Update logic は差分の復元や Flash ROM の消去・書き込みなどの制御を行う。

疑似車両アプリは、Windows PC 上のネイティブアプリケーションとして実装した。評価用車載ゲートウェイとの接続に必要な CAN 通信機能は、Vector 社の提供する XL Driver Library [23] を利用した。

車両状態把握機能については、図 10 に示すシーケンスを評価した。車両システムの車両状態の把握は、周期メッセージなどで CAN バス上を流れている情報を取得する方法と ECU に問い合わせる取得する方法の 2 通りがある。本研究では、前者の情報として車速、シフトポジション、

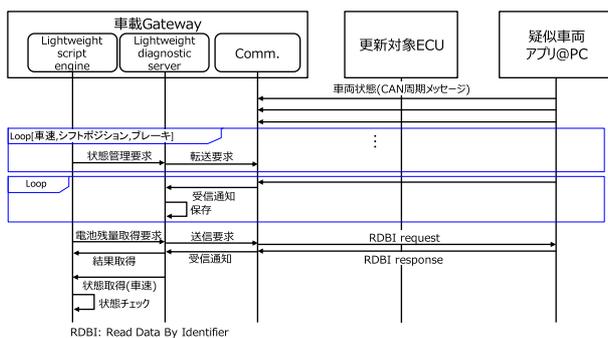


図 10 車両状態管理機能評価用更新シーケンス

Fig. 10 An evaluation sequence for vehicle state check.

ブレーキ状態、後者の情報としてバッテリーの残量を想定し、これら 4 つの情報を更新開始可否を判定する場合想定した評価とした。この更新開始条件も多様であるため、状態確認のシーケンスもスクリプトで実現することが好ましい。また、開始条件の判定にあたっては、車両内を流れるすべての情報を管理する必要はないため、本研究では、取得する情報をスクリプトから設定する構成を評価した。

更新シーケンス制御機能の評価には図 11 に示す簡易的な更新シーケンスを用いた。本評価の簡易シーケンスでは、ECU のソフトウェアバージョンの取得と確認、ECU に対するプログラム書き換えモードへの変更、更新プログラムの転送と ROM 書き込み、書き込み後の新プログラムの検証を行う。更新前のバージョンチェックは、部品交換や整備工場での有線ソフトウェア更新などにより ECU のソフトウェアバージョンが OTA センタと同期されているものと異なる場合の不具合を防止するために実施する。たとえば、差分更新を行う場合には、配信された差分データの基バージョンと ECU のバージョンが異なる場合は正常なデータの復元ができない。基本シーケンス以外にも、実運用で想定されるシーケンスとして、故障状態の取得、更新前の ECU のデータ書き込みを有効にするためのセキュリティロックの解除、パラメータの再設定など ECU 個別の処理が実行される。また、更新対象の ECU が他の ECU と依存関係を持つ場合には、その ECU のバージョンチェックも実行する必要がある。車両に搭載される ECU のバージョンチェックは、更新直前のみでなく、OTA センタとの構成情報の同期のためにシステム起動時に毎回実施するのが望ましい。

以上の評価環境により評価する内容とその方法を表 7 にまとめる。メモリ使用量および ECU ソフトウェア更新時間は、提案内容の車載ゲートウェイへの搭載可否の検証のために評価する。また、スクリプト変換処理は、配信にあたっての運用負荷低減のために、OTA センタにおいて機械的に実行されることを想定する OTX から Lua スクリプト

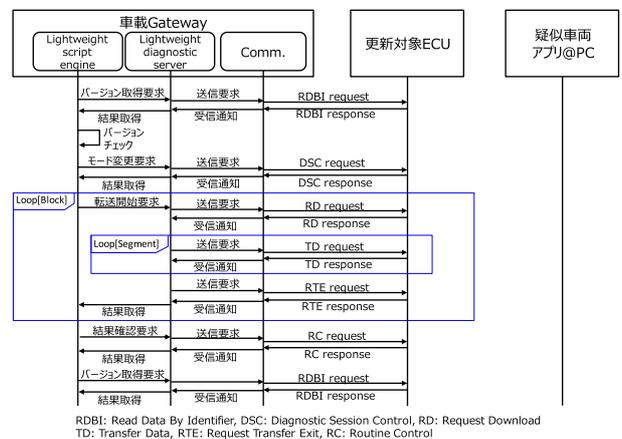


図 11 ECU 更新制御評価用更新シーケンス

Fig. 11 An evaluation sequence for an ECU update control.

表 7 評価項目と方法

Table 7 Evaluation items and methods.

評価内容		評価方法
メモリ 使用量	ECU更新制御	Req#1(a) 前述の評価環境で Lua によるメモリ確保要求を積算
	状態管理	Req#2(a) メモリマップより算出
	セキュリティ	Req#4(a) 前述の評価環境で実測
ECUソフトウェア更新時間	Req#1(b)	前述の評価環境で実測
スクリプト変換処理	Req#1(c)	OTX から Lua スクリプトへの変換における課題

表 8 提案方式によるメモリ使用量 [KB]

Table 8 Memory consumption of proposed method [KB].

Item	提案方式(Lua VM)		OSGi with JavaME	
	状態取得	ECU更新	OSGi	Java VM
RAM	15.6	12.3	6,000	128
プログラム (ROM)	76.0		100	1,000

表 9 セキュリティアルゴリズムのメモリ使用量 [KB]

Table 9 Memory consumption of security algorithms [KB].

Item	秘匿機能	認証/改ざん検知機能	
	AES128	RSA3072	SHA256
RAM	0.4	11.6	0.2
プログラム(ROM)	13.0	129.8	4.6

表 10 ECU ソフトウェア更新時間と内訳

Table 10 A breakdown of ECU software update time.

Item	Time[ms]	Ratio[%]
更新時間	14,392	100.0
通信&ECU 処理	14,074	97.8
更新制御機能処理	318	2.2

トへの変換処理の自動化可否の検証のために評価する。

## 4.2 評価

### (1) メモリ使用量

評価環境上で前述のシーケンスを実行した際の車載ゲートウェイにおける Lightweight script engine の RAM および ROM の使用量を表 8 に示す。また、比較対象として、関連研究 [7] で用いられた OSGi フレームワークを用いた場合の想定メモリ使用量を併記する。OSGi フレームワークの使用量は (株) 日立ソリューションズ社製品の公表値 [24] を用いた。また、Java VM の使用量は Oracle 社の公表値 [25] を用いた。

表 9 に前述のセキュリティアルゴリズムを車載ゲートウェイ用マイコンに搭載する場合のメモリ使用量を示す。

本評価により、セキュリティアルゴリズムを含む更新制御機能で使用するメモリが RAM は 16 KB 程度、ROM が 224 KB 程度とメモリ使用量の観点からは提案方式が車載ゲートウェイ用マイコンに搭載可能であることが確認できた。

### (2) 更新制御機能処理時間

評価環境上で前述のシーケンスを実行した際の ECU ソフトウェア更新時間とその内訳を表 10 に示す。

表 11 Lua に変換困難な OTX ステートメント

Table 11 OTX statements which is hard to convert to Lua.

#	ステートメント	内容
i	Parallel	並列処理
ii	Exception	例外処理
iii	break	loop 指定してのループ停止

本評価における更新制御機能の処理時間は全体の 2.2% であり、スクリプトを用いることにより処理時間に与える影響は軽微であることが確認できた。

### (3) スクリプト変換処理

OTX によるシーケンス記述を Lua スクリプトに変換するにあたっては、表 11 に示すステートメントについて対応が困難であることが判明した。これらの課題は前述のシーケンス、すなわち、ECU を一つずつ順次更新するケースを実現するにあたっては問題とならない。

#### (i) Parallel

OTX では Java で規定されているような thread による並列処理の記述が可能である。一方、Lua の並列処理はコルーチンで実装されており、スレッドの切替えを明示的に記述する必要がある。並列処理を行うには、OTX にスレッド切替えを行うためのステートメントを追加する、あるいは Lua 側で疑似的にマルチスレッドを実現するような記述への変換を行う必要がある。

#### (ii) Exception

OTX では Java で規定されているような try-catch による例外処理が記述できるが、Lua には相当する要素が存在しない。そのため、OTX スクリプト作成時に Exception を用いない制限を行うか、Lua 側に例外処理機構を実装する必要がある。

#### (iii) Break

OTX の break ステートメントでは多重ループの場合停止するループの指定が可能であるが、Lua ではそのような機能はない。そのため、OTX スクリプト作成時にループ指定の break を用いない制限を行うか、Lua 側にループを指定して処理を停止できる機構を実装する必要がある。

## 5. 考察

本稿では、自動車システムのソフトウェアを遠隔で更新するための更新制御機能について、機能要件を整理し、自動車システムの構成と特徴に基づいて機能配置と制御方式を提案した。また、車載ゲートウェイ向けマイコンへの実装評価によって、OTA センタとの接続機能を除く提案手法が実現可能であること示した。ただし、以下に述べる課題については、引き続き検討が必要である。

更新シーケンス制御機能については、単一 ECU の更新についての評価を行った。本評価の結果は、依存関係のない複数の ECU を逐次更新する場合には同様に適用可能である。一方で、実運用において、依存関係のある複数の ECU

を同時に更新する場合の検討がさらに必要である。このような場合には、1つのパッケージにこれらの更新データとスクリプトを同梱する。更新用のパッケージを格納する領域の制限などから、たとえば数十個の ECU を同時に更新することは運用上困難であり、同時に更新が行われる ECU の数はせいぜい数個である。しかし、これらは、依存関係を持つため、依存関係を考慮した更新順序の制御や、更新失敗時の制御が必要となる。たとえば、1つの ECU の更新に失敗した場合は、書き換えが完了した他方の ECU を元に戻すような制御が考えられる。また、複数の ECU の更新を短時間でを行うためには、更新の並列化が有効である一方で、並列化を行う場合は RAM 使用量が増加する可能性がある。この場合は、コルーチンによる並列化を実装することで、RAM 使用量を大きく増加させることなく、複数 ECU の同時更新の時間短縮が可能であると考えられるが、前述のとおりスクリプト変換処理に課題がある。

セキュリティ機能については、暗号化やデジタル署名によるパッケージの保護を想定し、これらに対応するためのセキュリティアルゴリズムの車載ゲートウェイマイコンへの搭載可否をメモリ使用量の観点で評価した。一方で、セキュリティの実運用のためには、アルゴリズムの搭載性に加え鍵情報などの管理が重要である。ゲートウェイは自動車システムのセキュリティ保護のために搭載されるため、ハードウェアセキュリティモジュールなどが搭載される。そのため、前述のセキュリティ保護に必要な鍵情報などをセキュアに管理することが可能であり、これらの保護機能を搭載することは容易であると考えられる。また、TLS などによる OTA センタとの通信路の保護も必要となるが、OTA センタとの通信路の保護については、後述の OTA センタとの通信機能とあわせて検討が必要である。

表 1 にあげた要件を満足するためには、さらに更新データの取得および構成情報の同期機能、すなわち OTA センタとの通信機能の検討が必要となる。本機能については、車種などに依存する部分はほとんどないと考えられるため、スクリプトなどでの対応による柔軟性の確保は必要ない。関連文献のように OMA DM のクライアント機能を搭載することで実現が可能となる。ただし、ゲートウェイはリソースが少ないため、最小限の機能を搭載するなどの検討が必要と考える。

## 6. おわりに

本研究では、自動車システムにおける OTA によるソフトウェアの遠隔更新向けの更新制御方式を提案した。提案にあたり、更新制御機能への要件を整理し、当該要件に基づいて更新制御機能の配置先を提案した。さらに、従来 OTA が適用されてきたコンシューマシステムと異なる自動車システムの多様性という課題に対して、スクリプトを用いて更新シーケンスの制御を行う方式を提案し、これを

実装してメモリ消費量、更新時間の観点から、提案方式が車載ゲートウェイ向けのマイコンに搭載可能であることを確認した。また、スクリプト変換の制限を明確にした。

今後は、依存関係を持つ複数 ECU の同時更新を行う場合の制御および OTA センタとの通信機能の方式検討を行うとともに、更新の高信頼化など自動車システムへの OTA 適用課題についての検討を行う。

## 参考文献

- [1] TESLA: MODEL S SOFTWARE RELEASE NOTES v7.1, available from ([https://www.teslamotors.com/sites/default/files/pdfs/release\\_notes/tesla\\_model\\_s\\_software\\_7.1.pdf](https://www.teslamotors.com/sites/default/files/pdfs/release_notes/tesla_model_s_software_7.1.pdf)) (accessed 2016-04-16).
- [2] 寺岡秀敏, 中原章晴, 黒澤憲一: 車載 ECU 向け差分更新方式, 情報処理学会論文誌コンシューマ・デバイス & システム (CDS), pp.41-50 (2017).
- [3] 富士カメラ総研ニュースリリース, 入手先 ([http://www.group.fuji-keizai.co.jp/press/pdf/170517\\_17042.pdf](http://www.group.fuji-keizai.co.jp/press/pdf/170517_17042.pdf)) (参照 2017-09-04).
- [4] Miller, C. and Valasek, C.: Remote exploitation of an unaltered passenger vehicle, Black Hat USA (2015).
- [5] Foster, I.D., Prudhomme, A., Koscher, K. and Savage, S.: Fast and Vulnerable: A Story of Telematic Failures, WOOT (2015).
- [6] OSGi Alliance, available from (<https://www.osgi.org/>) (accessed 2017-09-04).
- [7] de Boer, G., Engel, P. and Praefcke, W.: Generic remote software update for vehicle ecus using a telematics device as a gateway, *Advanced Microsystems for Automotive Applications 2005*, Springer Berlin Heidelberg, pp.371-380 (2005).
- [8] Ryu, H.K., Cho, S.R., Piao, S. and Kim, S.H.: The design of remote vehicle management system based on OMA DM protocol and AUTOSAR S/W architecture, *International Conference on Advanced Language Processing and Web Information Technology, 2008, ALPIT'08*, pp.393-397, IEEE (2008).
- [9] Zhang, J., Liao, Z. and Zhu, L.: Research on Design and Implementation of Automotive ECUs Software Remote Update, *Applied Mechanics and Materials*, Vol.740, pp.847-851, Trans Tech Publications (2015).
- [10] ISO13209-2 Road vehicles - Open Test sequence eXchange format (OTX) - Part2: Core data model specification and requirements.
- [11] ISO13209-3 Road vehicles - Open Test sequence eXchange format (OTX) - Part3: Standard extensions and requirements.
- [12] ISO22901-1 Road vehicles - Open Diagnostic Data, Exchange (ODX) - Part1: Data model specification.
- [13] JASO TP-15002(JP) 自動車の情報セキュリティ分析ガイド.
- [14] IPA, available from (<https://www.ipa.go.jp/security/vuln/CVSS.html>) (accessed 2018-02-28).
- [15] ISO14229-1 Road vehicles - Unified diagnostic services (UDS) - Part1: Specification and requirements.
- [16] Softing OTX.studio, available from (<https://automotive.softing.com/en/products/otx-studio.html>) (accessed 2017-02-26).
- [17] Lua.org, available from (<https://www.lua.org/>) (accessed 2017-02-26).
- [18] OpenSSL, available from (<https://www.openssl.org/>)

(accessed 2018-06-28).

- [19] RH850/F1x, available from (<http://japan.renesas.com/products/mpumcu/rh850/rh850f1x>).
- [20] マイコン評価 KIT FL-850/F1L-176-S, 入手先 (<http://www.tessera.co.jp/fl/f1l-176.html>) (参照 2016-09-19).
- [21] VECTOR: VN1600, available from ([https://jp.vector.com/vj-vn1600\\_jp.html](https://jp.vector.com/vj-vn1600_jp.html)) (accessed 2018-06-28).
- [22] ISO 15765-2:2011 Road vehicles — Diagnostic communication over Controller Area Network (DoCAN) — Part 2: Transport protocol and network layer services.
- [23] VECTOR : XL ドライバーライブラリー, 入手先 ([https://jp.vector.com/vj\\_xl\\_driver\\_library\\_jp.html](https://jp.vector.com/vj_xl_driver_library_jp.html)) (参照 2018-06-28).
- [24] 日立ソリューションズ: ニュースリリース, 入手先 (<http://www.hitachi-solutions.co.jp/company/press/news/soft/archive2008/news496.html>) (参照 2018-02-26).
- [25] ORACLE: Oracle Java ME Embedded FAQ, available from (<http://www.oracle.com/technetwork/java/embedded/documentation/me-e-otn-faq-1852008.pdf>) (accessed 2018-06-28).

1. CAN は, Bosch 社の登録商標です.
2. Java は, Oracle Corporation およびその子会社, 関連会社の米国およびその他の国における登録商標です.
3. OSGi は, 米国 OSGi Alliance の登録商標です.



船迫 陽介

(株) 日立産業制御ソリューションズ.  
2006 年室蘭工業大学工学部卒業. 携帯電話, 組み込みシステム, 車載システムに関連する研究開発に従事.



尾崎 友哉 (正会員)

1990 年名古屋大学大学院工学研究科修士課程修了. 同年 (株) 日立製作所入社. 組み込みシステム, ユーザインタフェース, EMS (Energy Management System) に関する研究開発に従事.



寺岡 秀敏

2002 年京都大学大学院工学研究科修士課程修了. (株) 日立製作所. 組み込みシステム, ネットワーク, 車載システムに関連する研究に従事.



山崎 裕紀

2007 年東京工業大学大学院情報理工学研究科修士課程修了. 日立オートモティブシステムズ (株). セキュリティ, 組み込みシステム, 車載システムに関連する研究開発に従事.



櫻井 康平

日立オートモティブシステムズ (株). 現在, 車載情報安全システム製品の量産開発業務に従事. 博士 (情報科学). 自動車技術会, 日本物理学会各会員.