oneM2M 規格に基づいた車両情報収集サービスの試作

笹原将章 [1]　大西亮吉 [1]　佐藤一馬 [1]

概要：車両に搭載された多様なセンサのセンシング情報がセンタサーバに集められ，故障診断や仕様改善，経路案内等に役立てられている．しかし，車両情報を収集するシステムが普及するに従い，通信網やセンタサーバの負荷が増大してしまう．さらに，外部のデバイスとの接続性やソフトウェアの安定性，メンテナンスの継続性，将来拡張に関しても課題が残る．そこで我々は，通信量削減が期待される出版購読モデルを用いた車両情報収集サービスを，標準化された oneM2M 規格に基づいて検討・試作した．今回の試作システムにおいて，車両情報収集の一連の動作（車両のサーバへの接続，サーバから車両へのデータの要求，要求に基づいた車両でのデータ処理，車両からサーバへの結果データの伝送）が 3 秒以内で完結することを確認した．

Vehicle Data Collection Lab Prototype based on oneM2M Standards

MASAAKI SASAHARA[1]　RYOKICHI ONISHI[1]　KAZUMA SATO[1]

1. 背景

車両には多様なセンサが搭載され，部品（タイヤの空気圧やエンジン，電池，電源など）や運行（測位，進行方向など）に関する情報を生成する．情報はセルラー等により固定網側のセンタサーバに集められ，故障診断や仕様改善，経路案内等に役立てられてきた．更には，高精度な地図の構築や車両以外の情報，例えば搭乗者の健康状態や道路インフラの劣化具合，農作物の生育状況，大気汚染の程度の把握なども期待される．

1.1 実現に向けた課題

やがて二つの課題に直面すると考える．一つは通信網やセンタサーバの負荷増大である．情報は増加する一方であり，アクセス網やバックホールの通信負荷やセンタサーバにおける情報処理負荷も増大する．もう一つは，外部のデバイスとの接続である．独自の通信仕様は一般に相互接続性が損なわれる．またソフトウェアの安定性やメンテナンスの継続性，将来拡張に関しても課題が残る．

1.2 課題解決の方策

負荷増大の課題に対しては，センタサーバ側の情報処理の一部を車載機側に移すことによる解決を提案する．サーバ負荷軽減に加えて，結果のみを伝達するため通信量削減も期待される．プル型通信の一種である出版購読モデルで実現する．相互接続性の課題に対しては，oneM2M[1]規格による解決を提案する．IoT/M2M は数多の規格が存在するが，oneM2M 規格に着目したのは次の 2 つの理由からである．i) 固定網との接続性．oneM2M は各国の通信標準化団体によって推進されており，3GPP Release 13 との相互接続仕様も含む．ii) デバイスとの接続性．デバイス側の有力な M2M 規格として AllJoyn や OIC が存在する．両者の間での接続は検討されていないが，固定網アクセスの必要性から oneM2M との接続仕様はそれぞれ検討されている．

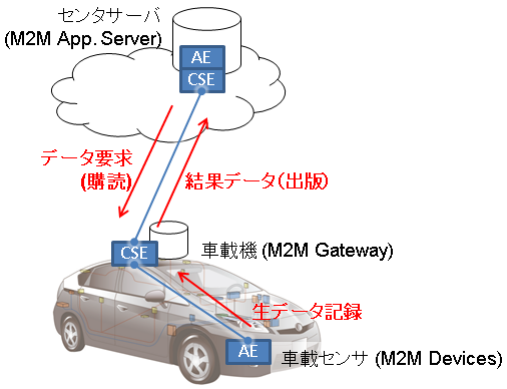

図 1 車両情報収集サービスのイメージ

2. 車両情報収集サービスの試作

センタサーバ（インフラ）が車載機（デバイス）に情報を要求し，車載機は要求に応じて保有する情報を処理してから，結果を応答するシステムを oneM2M 規格に基づいて試作した．センタサーバや車載機は PC で模擬し，PC 間の通信には無線 LAN を用いる．oneM2M 規格では CSE（Common Service Entity）が通信ミドルウェアとなり，AE（Application Entity）がアプリケーションのインタフェースとなる．アプリケーション間のデータ伝送には図 2 に示す 3 つのステップを踏む必要がある：i) AE から CSE への登録処理，ii) CSE 間の接続処理，iii) AE 間の CSE を介したデータ伝送．実線は Request，破線は Response，赤線は long polling による疑似サーバプッシュに関するメッセージである．単なる ACK や long polling の再設定などは簡略的に短

1（株）トヨタ IT 開発センター
Toyota InfoTechnology Center, Co., Ltd.

く記述する．

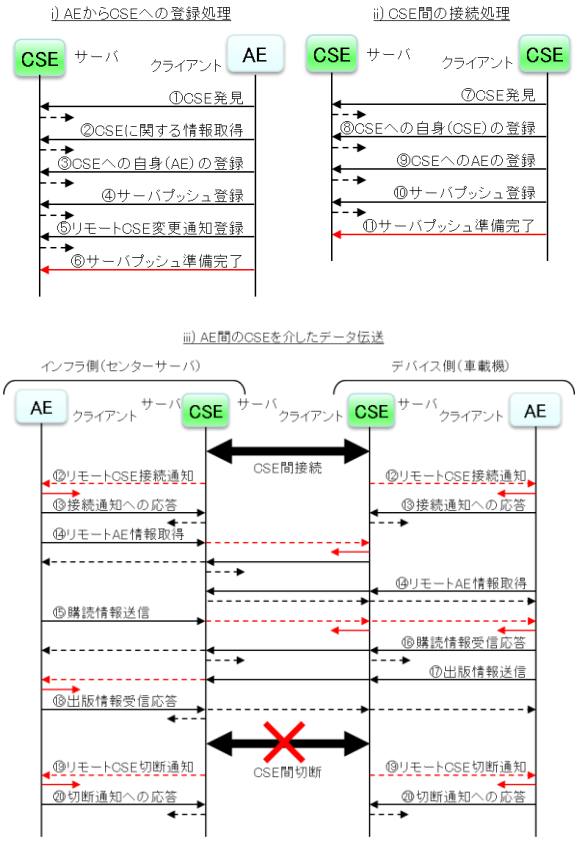

図 2 試作システムのメッセージシーケンス

3. 車両情報収集サービスの評価

本章では，車両情報収集サービスにおけるメッセージシーケンスごとの処理時間を評価した．

3.1 評価概要

i) AE から CSE への登録処理，ii) CSE 間の接続処理，iii) AE 間の CSE を介したデータ伝送の 3 つのメッセージシーケンスごとの処理に要した時間を実機評価する．

また，各 Entity 間のアプリケーション層の通信プロトコルとして oneM2M がサポートする規格のうち，最も普及している HTTP と，M2M アプリケーション向けに IETF で策定された標準規格である CoAP（Constrained Application Protocol）[2]の 2 種類を用意し，通信プロトコルごとの時間特性も比較評価する．

測定を 10 回繰り返し，その平均値を求める．

表 1 評価環境

対象	仕様
OS	Ubuntu 14.04.3 LTS
CPU	Intel Core i5-2520M (2.50GHz)
メモリ	8GB
無線 LAN モジュール	Intel Centrino Advanced-N 6205

3.2 評価環境

表 1 に記載したスペックの PC を 2 台用意し，1 台にはインフラ用の AE と CSE を，もう 1 台にはデバイス用の AE と CSE を実装した．AE と CSE の開発言語を

表 2 に，使用したミドルウェアを表 3 に，実装した HTTP 方式と CoAP 方式のプロトコルスタックをそれぞれ図 3 と図 4 に示す．想定する車両情報収集サービスでは，AE-CSE 間は同一ホスト内での有線接続を，CSE-CSE 間は異なるホスト間の無線接続を前提としている．そのため，AE-CSE 間は安定した通信が期待できるが，CSE-CSE は接続時間が限られていることや，マルチパスフェージングなどの影響により通信が不安定であることが想定される．通信の安定性の観点から， HTTP の実装では， AE-CSE 間にはトランスポート層に TCP を， CSE-CSE 間は UDP を用いて情報伝達を行う．なお，TCP と UDP のプロトコル変換は，UDT（UDP-based data transfer）[3]を実装した Proxy 上で実施する．一方， CoAP の実装では，すべての Entity 間の通信のトランスポート層に UDP を用いる．また，CoAP 方式では 1034Byte ごとにパケットが分割され，それぞれに制御ヘッダが付与される．

また，CSE-CSE 間の無線通信方式は，インフラ側の CSE をアクセスポイント，デバイス側の CSE をステーションとした 2.4GHz 帯の無線 LAN 接続とする．

評価対象の車両情報収集サービスでは，インフラ側の AE がデバイス側の AE に対して，エンジン回転数，車両速度，点火時期，スロットル開度，吸気温度など 60 種類の車両情報についての購読情報を送信し，デバイス側の AE はインフラから発行された購読情報に基づき，生のセンサデータを統計処理し，結果データを IETF の標準規格である JSON（JavaScript Object Notation）[4]形式で表現した出版情報（8054Byte）として，インフラ側の AE に送信する．

表 2 開発言語

対象	開発言語
CSE	Ruby 2.2.0
AE	Python 3.4.0

表 3 使用ミドルウェア

対象	ミドルウェア
UDT	InstantWebP2P/node-httpp (Version0.8.28)[5]
CoAP (CSE)	nning/coap（version 0.1.1)[6]
CoAP (AE)	aiocoap（version 0.1）[7]

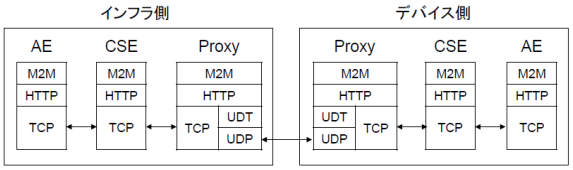

図 3 HTTP のプロトコルスタック

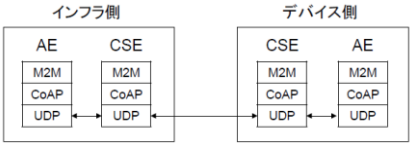

図 4 CoAP のプロトコルスタック

3.3 評価結果

AE から CSE への登録処理に要した時間を HTTP 方式，CoAP 方式についてそれぞれ表 4，表 5 に示す．表中の数値は処理開始からの経過時間の平均値（ミリ秒）を表している．HTTP 方式では登録完了までの時間がインフラ側で 548 ミリ秒，デバイス側で 563 ミリ秒であり，CoAP 方式ではインフラ側で 372 ミリ秒，デバイス側で 407 ミリ秒であり，両通信方式ともにインフラ側とデバイス側で大きな所要時間の差は見られなかった．これはインフラ側で行われる登録処理と，デバイス側で行われる登録処理が同等であるため，顕著な差が出なかったと考えられる．一方，通信方式ごとの所要時間を比較すると，CoAP 方式が HTTP 方式よりも，インフラ側で約 180 ミリ秒，デバイス側で 160 ミリ秒程度処理時間が短い結果となった．これは，CoAP が軽量な通信プロトコルであるため，通信制御用のオーバヘッドなどが省略されているためであると考えられる．

表 4 AE から CSE への登録処理（HTTP）

メッセージシーケンス	インフラ（ミリ秒）	デバイス（ミリ秒）
①CSE 発見	0	0
②CSE に関する情報取得	40	37
③CSE への自身（AE）の登録	173	172
④サーバプッシュ登録	323	338
⑤リモート CSE 変更通知登録	423	440
⑥サーバプッシュ準備完了	548	563

表 5 AE から CSE への登録処理（CoAP）

メッセージシーケンス	インフラ（ミリ秒）	デバイス（ミリ秒）
①CSE 発見	0	0
②CSE に関する情報取得	64	59
③CSE への自身（AE）の登録	149	162
④サーバプッシュ登録	259	280
⑤リモート CSE 変更通知登録	307	344
⑥サーバプッシュ準備完了	372	407

次に，インフラ側の CSE とデバイス側の CSE との接続処理の所要時間を表 6 に示す．HTTP 方式では接続完了までの時間が 995 ミリ秒，CoAP 方式では 278 ミリ秒であり，CoAP 方式は，HTTP 方式の 1/3 以下の時間で接続が完了している結果となった．AE から CSE への登録処理と同様に，CoAP ではオーバヘッドなどが省略されているためであると考えられる．

表 6 CSE 間の接続処理

メッセージシーケンス	インフラ（ミリ秒）	デバイス（ミリ秒）
⑦CSE 発見	0	0
⑧CSE への自身（CSE）の登録	274	72
⑨CSE への AE の登録	468	132
⑩サーバプッシュ登録	691	205
⑪サーバプッシュ準備完了	995	278

さらに，インフラ側の AE と，デバイス側の AE 間データ伝送に要した時間を表 7，表 8 に示す．データ伝送が完了するまでの所要時間では HTTP 方式で 2650 ミリ秒，CoAP 方式で 2606 ミリ秒と大きな差は見られないが，⑮購読情報送信までの経過時間を比較すると，HTTP 方式で 1091 ミリ秒，CoAP 方式で 520 ミリ秒と，CoAP 方式の方が処理時間が短い結果となった．

一方で，⑮購読情報送信から，⑰出版情報受信応答完了までの所要時間を比較すると，HTTP 方式で 1559 ミリ秒，CoAP 方式で 2081 ミリ秒と，HTTP 方式の方が処理時間が短い結果となった．これはデータ伝送をする出版情報が 8054Byte と大きいたため，一度により大きなメッセージサイズのパケットを送信できる HTTP の方が有利であったと考えられる．

なお，本実装では，⑬接続通知応答と⑭リモート AE 情報取得は並列に実行されており，⑭リモート AE 情報取得の方が⑬接続通知応答よりも早く完了している．

表 7 AE 間の CSE を介したデータ伝送（HTTP）

メッセージシーケンス	インフラ（ミリ秒）	デバイス（ミリ秒）
⑫リモート CSE 接続通知	0	0
⑬接続通知応答	797	259
⑭リモート AE 情報取得	5	5
⑮購読情報送信	1091	-
⑯購読情報受信応答	-	1282
⑰出版情報送信	-	2040
⑱出版情報受信応答	2650	-

表 8 AE 間の CSE を介したデータ伝送（CoAP）

メッセージシーケンス	インフラ（ミリ秒）	デバイス（ミリ秒）
⑫リモート CSE 接続通知	0	0
⑬接続通知応答	212	89
⑭リモート AE 情報取得	5	4
⑮購読情報送信	520	-
⑯購読情報受信応答	-	1276
⑰出版情報送信	-	2038
⑱出版情報受信応答	2606	-

3.4 評価まとめ

検討中の車両情報収集システムでは，AE から CSE への登録は起動時のみ実施すればよいため，車両がアクセスポイントに繋がってからデータ伝送が完了するまでの時間は，⑦CSE 発見から⑱出版情報受信応答までを評価すればよい．測定結果，接続からデータ伝送完了までの時間は HTTP 方式で約 3.7 秒，CoAP 方式で約 2.9 秒で完了することが分かった．さらに，通信プロトコルの特性を比較すると，CoAP は接続処理などのデータ量が比較的小さいメッセージシーケンスの送受に向いており，HTTP は出版購読などのデータ量が比較的大きいメッセージの送受に向いていることが分かった．

4. 出版情報のバイナリシリアライズ

第 3 章で述べた車両情報収集サービスでは，出版する情報を JSON 形式で表現していたが，本章では，出版する情報にバイナリ表現を用いること（バイナリシリアライズ）を検討・評価する．このことにより，伝送するデータ量の削減が期待される．バイナリシリアライズによるデータ量の圧縮率と AE 間のデータ伝送の所要時間を評価項目とした．

4.1 評価環境

評価対象のバイナリシリアライズ方式として，IETF の標準規格である CBOR（Concise Binary Object Representation）[8][9]を用いた．デバイス側の AE はインフラ側の AE からの購読情報に応じて，出版情報を生成し CBOR 形式に変換して送信する．それ以外の実装方法は第 3 章で述べた車両情報収集サービスと同様である．

4.2 評価結果

出版情報における各データ表現形式のデータ量を表 9 に示す．JSON 形式は 8054Byte であるのに対し，CBOR 形式では 6668Byte であり，バイナリシリアライズにより約 8 割にデータ量が圧縮されたことになる．

さらに，CBOR 形式で出版情報を伝送したときの AE 間の CSE を介したデータ伝送に要した時間をそれぞれ表 10 と表 11 に示す．第 3 章で示した JSON 形式の結果（表 7）と比較すると，インフラ側とデバイス側の AE 間のデータ伝送に要した時間は，HTTP 方式では JSON 形式で 2650 ミリ秒，CBOR 形式で 2832 ミリ秒，CoAP 方式では JSON 形式で 2606 ミリ秒，CBOR 形式で 2833 ミリ秒という結果であり，HTTP 方式では 180 ミリ秒，CoAP 方式では 230 ミリ秒程度，CBOR 形式の方が JSON 形式よりも処理時間が長くなる結果となった．これは，データ形式変換にかかる内部処理が発生したためであると考えられる．

表 9 データ形式ごとの出版情報量

データ形式	サイズ（byte）	JSON 比
JSON	8054	1
CBOR	6668	0.83

表 10 AE 間の CSE を介したデータ伝送（HTTP）

メッセージシーケンス	インフラ（ミリ秒）	デバイス（ミリ秒）
⑫リモート CSE 接続通知	0	0
⑬接続通知応答	838	257
⑭リモート AE 情報取得	4	4
⑮購読情報送信	1142	-
⑯購読情報受信応答	-	1420
⑰出版情報送信	-	2159
⑱出版情報受信応答	2832	-

表 11 AE 間の CSE を介したデータ伝送（CoAP）

メッセージシーケンス	インフラ（ミリ秒）	デバイス（ミリ秒）
⑫リモート CSE 接続通知	0	0
⑬接続通知応答	29	135
⑭リモート AE 情報取得	447	4
⑮購読情報送信	712	-
⑯購読情報受信応答	-	1384
⑰出版情報送信	-	2135
⑱出版情報受信応答	2833	-

5. まとめ

本稿では，通信網やセンタサーバの負荷低減，外部のデバイスとの相互接続性などを目的とした oneM2M 規格に基づいた車両情報収集サービスを提案し，性能評価を行った．評価の結果，車両情報収集の一連の動作（車両のサーバへの接続，サーバから車両へのデータの要求，要求に基づいた車両でのデータ処理，車両からサーバへの結果データの伝送）が 3 秒以内で完結することを確認した．

さらに，通信方式を比較した結果，通信プロトコルの特

性を比較すると，メッセージシーケンスにおけるデータサイズなどの特徴を考慮した通信方式を検討する必要があることが分かった．

また，出版情報のデータ形式にバイナリ表現を用いると伝送データ量を削減できるが，処理時間が増加する可能性があるため，通信と計算を総合的に最適化していく必要がある．

今後は更なる議論の発展のために，ソースコードの提供等も検討したい．

参考文献

[1] oneM2M Release 1 specifications，oneM2M partners (2015)，入手先〈http://www.onem2m.org/technical/published-documents〉．

[2] The Constrained Application Protocol (CoAP)，IETF RFC7252，入手先〈https://tools.ietf.org/html/rfc7252〉．

[3] UDT: UDP-based Data Transfer Protocol，IETF Internet Draft，入手先〈https://tools.ietf.org/html/draft-gg-udt-03〉．

[4] The JavaScript Object Notation (JSON) Data Interchange Format，IETF RFC7159，入手先〈https://tools.ietf.org/html/rfc7159〉．

[5] InstantWebP2P/node-httpp (Version0.8.28)，Github ，入手先〈https://github.com/InstantWebP2P/node-httpp〉．

[6] nning/coap（version 0.1.1) source code，Github ，入手先〈https://github.com/nning/coap〉．

[7] aiocoap（version 0.1）source code，Github ，入手先〈https://github.com/chrysn/aiocoap〉．

[8] Concise Binary Object Representation (CBOR)，IETF RFC7049，入手先〈https://tools.ietf.org/html/rfc7049〉．

[9] CBOR source code，Github，入手先〈https://bitbucket.org/bodhisnarkva/cbor〉．