自動車ネットワークサービスの連携アーキテクチャ

2007MI028　濱千代　正弥　　2007MI093　片桐　雅仁

指導教員　青山　幹雄

1. はじめに

近年，車載システムと外部ネットワークシステムが連携する自動車ネットワークサービスが実用化されている．しかし，この連携にはインタフェースやプロトコルの違いと，車載システムが組み込みシステムであるため処理能力の制約という問題がある．

本稿では，これらの問題を解決するために SOA (Service-Oriented Architecture)[5]を適用する．自動車の外部ネットワークシステムと車載システムの機能を標準インタフェースを介して提供するサービスととらえ，連携可能なソフトウェアアーキテクチャを提案する．

2. 研究の背景

本稿では，車外サービス連携に着目する．車外サービス連携とは，車載システムと自動車の外部ネットワークシステムの連携と定義する．

現状の車外サービス連携では，サービスを提供するプロバイダごとにインタフェースやプロトコルが固有である．そのため，サードパーティサービスの利用や異なるサービス間のサービス連携も困難である(図 1)．

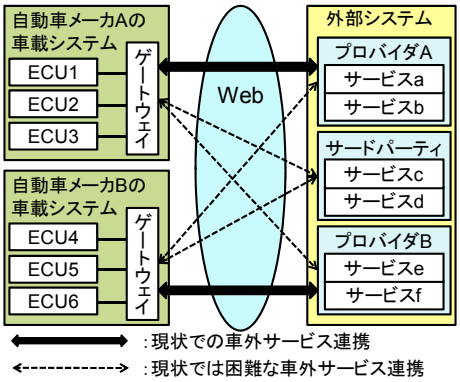


図 1　車外サービス連携の現状

3. 研究課題

サードパーティサービスの利用や異なるサービス間のサービス連携を容易にするため，SOAP や REST (REpresentational State Transfer)などの Web サービスで標準化されたメッセージプロトコルを適用することが考えられる．また，SOAP に準処したメッセージを SOAP メッセージ，REST に準処したメッセージの一つを POX メッセージと呼ぶ．これらを車外サービス連携に適用するにはプロトコルの変換が必要となる．

しかし，現状の ECU(Electronic Control Unit)では，処理能力の制約が厳しいため，CAN(Controller Area Network)や MOST(Media Oriented Systems Transport)などの車載ネットワークに対して，SOAP メッセージや POX メッセージのプロトコル変換を個別の ECU で行うことは困難である．よって ECU で行うのではなく，新たにプロトコル変換を行うシステムを追加する必要がある．

4. 関連研究

4.1. OSGi(Open Service Gateway initiative)

OSGi[7]は家電製品をインターネットに接続し，パソコンや機器間で相互にサービスを提供するための仕様であり，ホームネットワークでのゲートウェイの機能を持っている．OS 上で JVM(Java Virtual Machine)がプロセスとして実行され，そのプロセス上で Java 言語により記述されたミドルウェアである OSGi フレームワークが動作する．この上に Bundle と呼ばれるプラグ&プレイ可能なソフトウェアコンポーネントがインストールされ実行される．

また，OSGi 仕様では，多くの開発者にとって利用価値が高い共通的なサービスとユーティリティクラスを定義している．これらを Bundle の形態で提供されたものを Basic Service Bundle と呼ぶ．一方，各事業者や機器に依存し標準化されていない Bundle を Application Bundle と呼ぶ．

4.2. SCSN(Smart Car Sensor Network)

SCSN プラットフォーム[8]は，車載システムの中でも追加や変更が特に多いセンサに対応するために提案されている．また，SCSN プラットフォームは OSGi フレームワークを基盤としたアーキテクチャである．

4.3. 軽量 Web サービスアーキテクチャ

Web サービスのメッセージ交換では，主に SOAP が利用されている．しかし，SOAP 構文の生成や解析に多くの処理が必要となるため，性能低下が問題となる．この問題に対し，メッセージ交換の手段として REST を利用したアーキテクチャを提案している[3]．また，提案を検証するためにプロトタイプを開発し，応答時間を比較している．

5. アプローチ

本稿では，プロトコル変換の必要性に着目し，ゲートウェイの機能を持つ車載サービスブローカを提案する(図 2)．

車載サービスブローカでプロトコル変換を行うことにより，SOAP/REST を用いた車外サービス連携が可能となり，サ

ードパーティサービスの利用や異なるサービス間のサービス連携を容易にする.

プロトタイプ実装により SOAP メッセージと POX メッセージを比較し，サービス間連携の性能評価を行うことで，提案するアーキテクチャの妥当性を示す.

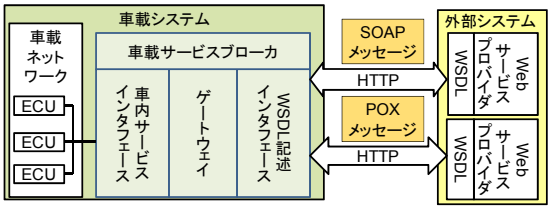


図 2　車外サービス連携モデル

6. 提案方法

6.1. 車載サービスブローカのアーキテクチャ

プロトコル変換を行う車載サービスブローカ[6]を用いたソフトウェアアーキテクチャを提案する(図 3).

また，この車載サービスブローカには OSGi フレームワークを用いる．OSGi では，プロトコル変換の機能を持つ Bundle が標準化されて，提供されており，SOAP/POX メッセージの利用が容易である．さらに，複雑化している車載ソフトウェア開発に対して，Bundle を用いたモジュール化により開発効率，運用効率の向上が図れる．車載サービスブローカの OSGi フレームワーク上に以下に挙げる機能を持つ四つの Bundle を実装する.

a) テレマティクスサービスの機能を持つ Telematics Application Bundle
b) SOAP/POX メッセージの生成と解析を行う SOAP/POX Proxy Processor Bundle
c) HTTP による通信を行う HTTP Server Bundle
d) CAN Gateway を通して車載ネットワークへ通信を行う Interface Gateway Bundle

これらの Bundle が連携して動作することで車載ネットワークと外部システムのサービス連携を実現する.

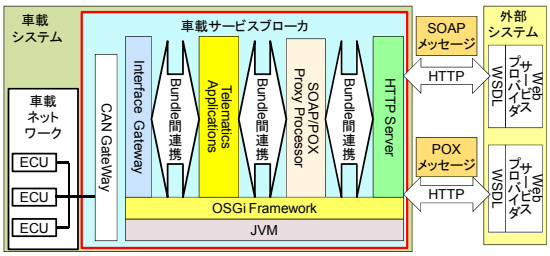


図 3　車載サービスブローカのアーキテクチャ

6.2. Bundle 間連携

Bundle 間連携とは，車載サービスブローカ内の OSGi フレームワーク上の異なる Bundle 同士の連携である．OSGi の機能であるパッケージの公開と読み込み，サービスの登録，検索と取得を利用することで実現される(図 4).

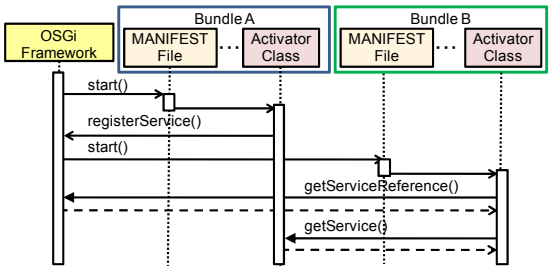


図 4　Bundle 間連携

6.3. 外部システムとの連携

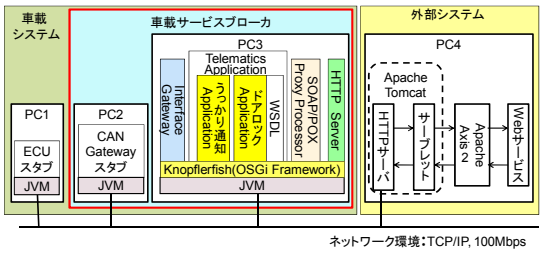
外部システムとの連携とは，車載サービスブローカと外部システムとの連携である．Telematics Application Bundle が SOAP/POX Proxy Processor Bundle，HTTP Server Bundle と連携し，サービスプロバイダが提供する外部システムのサービスと連携を行う．また，SOAP/POX Proxy Processor Bundle と HTTP Server Bundle は 4.1 節で説明した Basic Service Bundle を複数まとめた総称である.

6.4. 車載ネットワークとの連携

車載ネットワークとの連携とは，車載サービスブローカと車載ネットワークの連携である．Telematics Application Bundle が Interface Gateway Bundle と連携して CAN Gateway と連携を行う．Interface Gateway Bundle を実現することで車載ネットワーク内の実装が変更された場合にも Interface Gateway Bundle を変更するだけで対応可能である．Interface Gateway Bundle と CAN Gateway により Telematics Application Bundle と ECU 間の連携が可能となる.

7. 車載サービスブローカのプロトタイプ

Web サービスのインタフェースを用いた接続性と応答時間の性能を評価するためオープンソースの OSGi フレームワークである Knoplerfish を用いてプロトタイプを開発した[1,4]．CAN Gateway と ECU はスタブとして Java で実装を行い，Interface Gateway と CAN Gateway 間，CAN Gateway と ECU 間には TCP 接続を用いる(図 5).



	PC1 (ECU)	PC2(CAN Gateway)	PC3(OSGi Framework)	PC4 (外部システム)
OS	Windows XP SP3	Windows XP SP3	Windows XP SP3	Windows XP SP3
メモリ	2.00GB	1.00GB	1.00GB	2.00GB
CPU	Intel Celeron(R)M 2.00GHz	Intel Pentium(R)4 2.00GHz	Intel Pentium(R)4 2.00GHz	Intel Celeron(R)M 2.00GHz

図 5　プロトタイプ構成

テレマティクスサービスの例として，以下の二つのサービスを実装した．

(1)　うっかり通知サービス

うっかり通知サービスは，自動車の鍵の閉め忘れを検知し，所有者に通知する(図 6)．

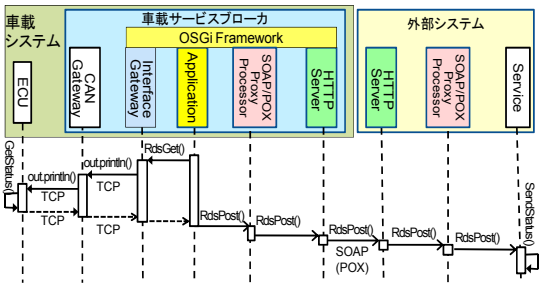


図 6　うっかり通知サービスのシーケンス図

(2)　リモートドアロックサービス

リモートドアロックサービスは，携帯電話などの機器を用いて，外部からドアロックを行う(図 7)．

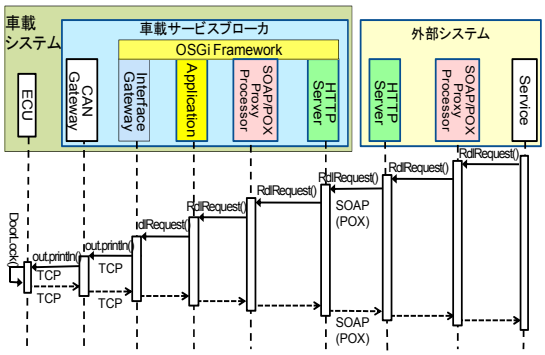


図 7　リモートドアロックサービスのシーケンス図

8. プロトタイプによる評価

8.1. Web サービスのインタフェースを用いた接続性

提案した車載サービスブローカを用いて，車載サービスブローカと外部の Web サービスが接続可能であり，車載システムと外部システム間の双方向の通信が可能であることを確認した．

8.2. 外部システムとの連携の性能評価

うっかり通知サービスの外部システムとの連携処理の性能を測定した．そのため，実行回数，データ量による応答時間を測定するために，外部システムとの連携を非同期ではなく，受け取った文字列をそのまま返す同期に変更を行い，応答時間を測定した．

8.2.1. 実行回数による応答時間の変化

サービスリクエスタが文字列データ(100Byte)を送信し，返り値としてデータをそのまま受信する．実行回数を変化させて応答時間を測定した．図 8 に SOAP メッセージ，図 9 に POX メッセージの実行回数による応答時間を示す．

図 8，図 9 よりほぼ一定周期で異常に大きな遅延が発生している．これは JVM のガベージコレクションによると推定される．そこで平均値の約 2 倍以上を特異値とし，SOAP メッセージは 20 ミリ秒以上，POX メッセージは 10 ミリ秒以上を特異値とした．この特異値を除いた平均応答時間を補修値とした．測定値と共に補修値を図 10 に，補修値の平均応答時間と標準偏差を表 1 に示す．平均応答時間に対し，標準偏差が約 5%であることからデータのばらつきが少ないといえる．

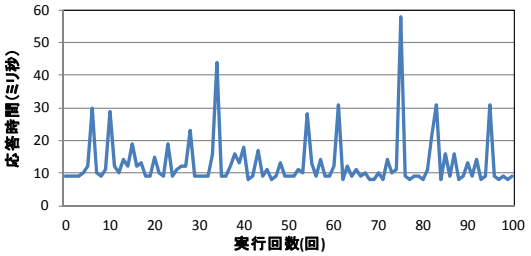


図 8　SOAP メッセージによる応答時間

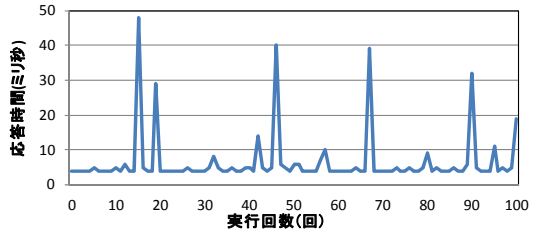


図 9　POX メッセージによる応答時間

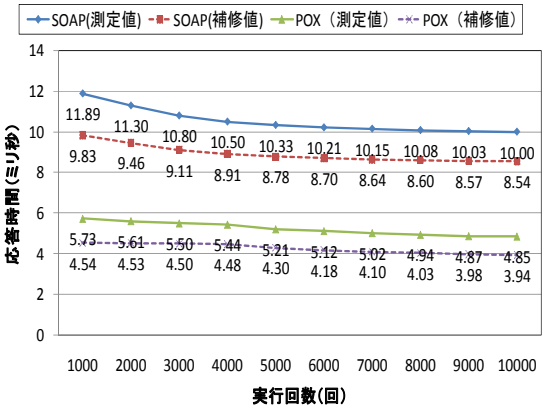


図 10　実行回数による応答時間

表 1　平均応答時間と標準偏差

	平均応答(ミリ秒)	標準偏差
SOAP（補修値）	8.91	0.41
POX（補修値）	4.26	0.23

8.2.2. データ量に対する応答時間の変化

送信するデータ量の増加による SOAP メッセージと POX メッセージの応答時間の変化を測定した．測定には，

100Byteごとに100回実行し，ガベージコレクションによる特異値を除いた平均応答時間を補修値として，線形近似直線と共に図11に示す．

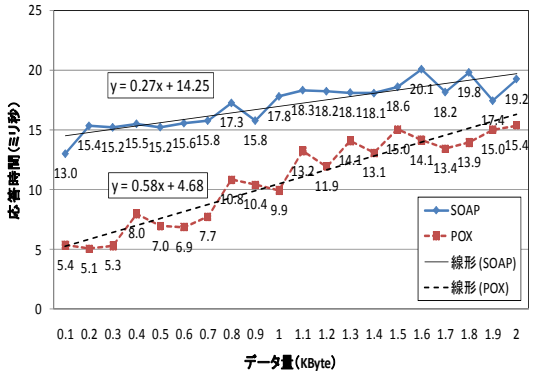


図11　データ量における応答時間の差

8.2.3. 性能評価

性能測定より以下の二点が明らかになった．

(1) 実行回数に依存せず POX メッセージの応答時間は SOAP メッセージに対して約50%短い．

(2) データ量が約 3Kbyte まで POX メッセージの応答時間は SOAP メッセージに対して短い．しかし，線形近似直線の傾きが SOAP メッセージに比べ POX メッセージが約2.5倍であることからデータ量の増大に従い応答時間の差は縮小する．

これは POX メッセージには SOAP エンベロープのような構造がなく，簡潔な構造のためメッセージの生成，解析処理が SOAP メッセージに比べ少ないためだと思われる．しかし，データ量の増加により相対的にデータ量に対する SOAP エンベロープのオーバヘッドによる割合が縮小するので応答時間の差も縮小したと思われる．

さらに，データ量が2KByteまでの応答時間はSOAPメッセージ，POX メッセージ共に平均 20 ミリ秒以内である．そのため提案するアーキテクチャはうっかり通知サービスやリモートドアロックサービスなどの自動車ネットワークサービスの応答時間の要求を満たしている．

9. 考察

9.1. 車載サービスブローカを用いた車外サービス連携

現状の車外サービス連携では利用できるサービスは，自動車メーカの提供するサービスに限定される．提案したアーキテクチャでは Web サービスとして標準化されているメッセージプロトコルである SOAP/REST を用いることにより外部システムとの連携が容易となり，サードパーティサービスが利用できる．さらにサービスの組み合わせが可能であり，組み合わされた新たなサービスの利用も可能である．

9.2. 外部システムとの連携の性能評価

応答時間を用いて性能評価を行い，外部システムとの連携の応答時間がうっかり通知サービスやリモートドアロックサービスなどの自動車ネットワークサービスの応答時間の要求を満たしていることを示した．よって応答時間の観点から提案するアーキテクチャは妥当性があるといえる．

ただし，JVM のガベージコレクションにより一定周期に異常に大きな遅延が発生するためリアルタイム性の要求が厳しい自動車ネットワークサービスには Java 言語のリアルタイム仕様である RTSJ(Real Time Specification for Java)[2]の適用が必要である．

10. 今後の課題

10.1. 車載ネットワークとの連携においての応答時間

外部システムとの連携部分のみでは性能評価が不十分である．CAN Gateway と ECU を用いてサービス全体の応答時間を測定することが必要である．

10.2. 車外サービス連携におけるセキュリティ

セキュリティの観点からの検証が必要である．さらに，セキュリティ機能を付加したうえでの応答時間の評価が必要である．

10.3. ガベージコレクションによる遅延の解決

リアルタイム性の要求が厳しい自動車ネットワークサービスを利用するには RTSJ の適用によりガベージコレクションによる遅延を無くすための構造が必要である．

11. まとめ

本稿では SOA を用いて，車載システムと外部ネットワークシステムのサービス連携可能なアーキテクチャを提案した．さらに，OSGi を基盤とし，プロトコル変換を可能にする車載サービスブローカを提案した．車載サービスブローカのプロトタイプを開発し，応答時間を測定し，妥当性を確認した．

参考文献

[1]Apache axis2, http://axis.apache.org/axis2/java/core/.

[2]P. C. Dibble, Real-Time Java Platform Programming, PrenticeHall, 2002.

[3]池崎 崇, 軽量サービス指向アーキテクチャ設計方法の提案と評価, 2008 年度南山大学修士論文, 2009.

[4]Knoplerfish OSGi-Open Source OSGi Service Platform, http://www.knopflerfish.org/index.html.

[5]D. Krafzig, et al., Enterprize SOA, Prentice Hall, 2005.

[6]水谷 拓人, 自動車ネットワークサービスのサービス連携アーキテクチャの提案と評価, 2009 年度南山大学修士論文, 2010.

[7]OSGi(Open Service Gateway Initiative) Alliance, http://www.osgi.org/Main/HomePage.

[8]P. Park, et al., An OSGi Based In-Vehicle Gateway Platform Architecture for Improved Sensor Extensibility and Interoperability, Proc. IEEE COMPSAC 2009, pp.140-147.