

車載向けサービスプラットフォームの構築と 評価

岩井明史[†]

現在、自動車分野において、高信頼かつ安全なサービス統合システムの実現が課題となって来ている。本研究では、SOAをレファレンスとした車載向けサービスプラットフォームを構築し、車内外サービス連携に関する機能と性能評価を行った。

Designing and Evaluation of Automotive Service Integration Platform

Akihito Iwai[†]

Today, in automobile industry, it becomes the challenge that we are able to realize highly reliable and safety service integrated systems. In this study, we designed the SOA-based service integration platform for automotive and evaluated the functionality and real-time performance regarding the association between in-vehicle and out-vehicle services.

1. はじめに

近年、ACC (Adaptive Cruise Control) や LKS (Lane Keeping System) 等の高度な安全制御システムの普及に伴い、車載電子制御システムは、益々、大規模化・複雑化の一途を辿っている。更に、最近では、車外の交通インフラや iPhone や Android 等のモバイル情報機器とも繋がり、クルマと車外の複数の情報サービスとが密に連携・融合する、所謂、サービス統合システムへと発展している。

また、将来的には、低炭素かつ安全・快適・便利な新モビリティ社会の実現に向け、クルマは、PHV/EV の電動化と合わせ、家・ビル・店・街等あらゆる社会の構成要素と繋がり、サービス連携範囲が、更に拡大していくと予測されている。

本発表では、そうした状況において、今後の広範囲かつ付加価値の高い車内外のサービス連携の仕組み実現に向け、IT 業界の標準技術の一つである SOA (Service Oriented Architecture) に着目し、サービス統合システムの核となる安全かつ信頼性の高い車載向けのサービスプラットフォームを構築・評価したものである。

2. 研究背景

本章では、本研究の背景となる自動車分野でのサービス統合システムの課題について述べる。

2.1 自動車分野でのサービス統合システムの課題

車内と車外の夫々の機能やサービスを連携・統合したサービス統合システムを構築する場合、車内の車載電子制御機器 (ECU : Electronic Control Unit) と車外のパソコンやモバイル情報端末等の情報機器との特性の違いに着目する必要がある。両者は、夫々の機器に求められる特性がかなり異なる。

例えば、一般的に、車載電子制御機器の方が、情報機器に対して、リアルタイム性 (時間制約)、安全性、信頼性、品質、コスト等の非機能要件が厳しい。また、開発プロセスや製品サイクルも、車載電子制御機器の方が長い。

従って、情報機器と同様の常時接続を前提とした、クルマの情報通信ネットワークシステム、即ち、自動車分野でのサービス統合システムを構築しようとした場合、情報機器の場合とは異なる課題が生じる。

(1) リアルタイム性の保証 (時間制約の厳守)

車載電子制御システムは、時間制約が厳しいリアルタイム組込みシステムである。センサで取り込んだ信号を、決まった時間内に処理し、アクチュエータを適切なタイ

[†] 株式会社デンソー
DENSO Corporation

ミングで制御しなければならない。もし、何らかの影響で決められた時間を過ぎてしまった場合は、制御性が損なわれ、最悪、システムが誤動作する場合がある。つまり、サービス統合システムにおいては、例えば、車外の情報機器からインターネット経由で、センサ情報を取得、或いは、アクチュエータを制御しようとした場合等、リアルタイム性の保証（時間制約の厳守）が課題となる。

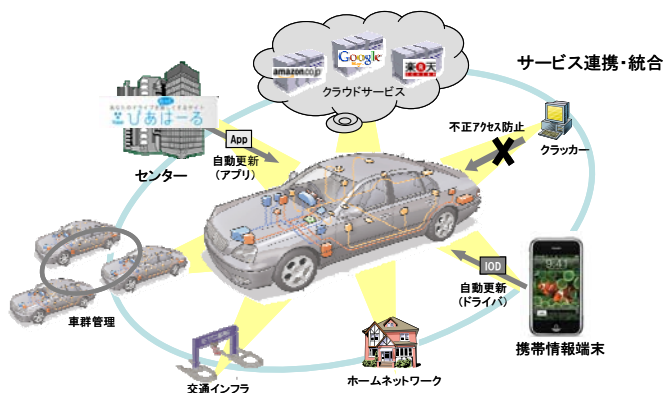


図 1 サービス統合システム

(2) 安全性・信頼性の確保

また、車載電子制御システムは、人命に関わるクリティカルシステムである。そのため、システム構築上、安全性・信頼性が最も重視される。サービス統合システムにおいて、信頼性が不十分な車外の情報機器・情報インフラと、高い安全性と信頼性が要求される車載制御機器をシステム統合する場合、全体のシステムとしての安全性・信頼性の確保が課題となる。つまり、車外の情報機器が故障した場合においても、車載制御機器には、何ら影響を与えない仕組み（Isolation）が必要となる。情報セキュリティ面においても同様である。悪意を持ったクラッカーからの不正アクセスに対して、防御する情報セキュリティの仕組み（Fire Wall）も不可欠である。

(3) 車外の早い製品サイクルへの追従

情報機器は、車載制御機器と違って、製品サイクルが早い。また、ソフトウェアのアップデートが容易であり、製品リリース後も機能アップが可能である。インターネット上のサービスプロバイダから、オンデマンドでサービスも利用できる。これは、上述のリアルタイム性・安全性・信頼性が、車載制御機器に比べ、あまり厳しくない

という理由もあるが、情報通信分野での技術が多く標準化されていることが大きい。

例えば、インターネット上のサービスの場合は、既に、HTTP や FTP 等の通信プロトコルが標準化されているため、誰でも簡単に世界中のサイトにアクセスできる。SOAP や REST 等の Web サービスの高位プロトコルも標準化されているため、サービス連携も容易である。

しかし、一方、自動車分野においては、このようなサービス系の標準化は、まだ始まったばかりである。Telematics 向けの NGTP (New Generation Telematics Protocol) [1] や、Genevi[2]等があるが、実用段階には至っていない。

3. 車載向けサービスプラットフォームによる解決アプローチ

3.1 狙い・目的

本研究では、自動車分野でのサービス統合システムの課題を解決するために、車載電子機器向けの制御系プラットフォームと情報機器向けの情報系プラットフォームに、サービスレイヤを追加した新しい車載向けサービスプラットフォーム（開発コードネーム：DARWIN）の構築を考案した。その際、制御系プラットフォームは、現在、自動車業界標準に成りつつある AUTOSAR[3]を前提とした。また、情報系プラットフォームは、IT 業界の標準である Windows や Linux を想定した。

この車載向けサービスプラットフォームは、大規模・複雑化するサービス統合システムにおいて、車内外のサービスを安全かつ迅速に連携・統合する事を目的としている。また、車載電子制御システムで、重要なリアルタイム性の保証も狙いとしている。

3.2 SOA (Service Oriented Architecture) アプローチ

数年前より、通信サービスと情報サービスとを Web を介してシームレスに統合するための基盤アーキテクチャである SOA (Service Oriented Architecture) [4]の研究が盛んになっている。SOA では、全ての情報を”サービス”として扱う事で、物理的なシステム構成や種々の通信プロトコルを仮想化（抽象化）し、一貫した情報サービスに対するインターフェイスを提供している。また、その構成要素は、XML や SOAP, WSDM, BPML 等の業界標準を利用しているため、既存システムとの親和性も良い。最近では、リアルタイム SOA と称し、時間制約があるシステムへの適応のため、リアルタイム性の拡張に関する研究も進んでいる[5][6][7]。

本研究では、この SOA をレファレンスとして、車載向けサービスプラットフォームを構築した。その際、OMG での階層モデル[8]をベースに、階層毎に、自動車分野の構成要素をマッピングした。

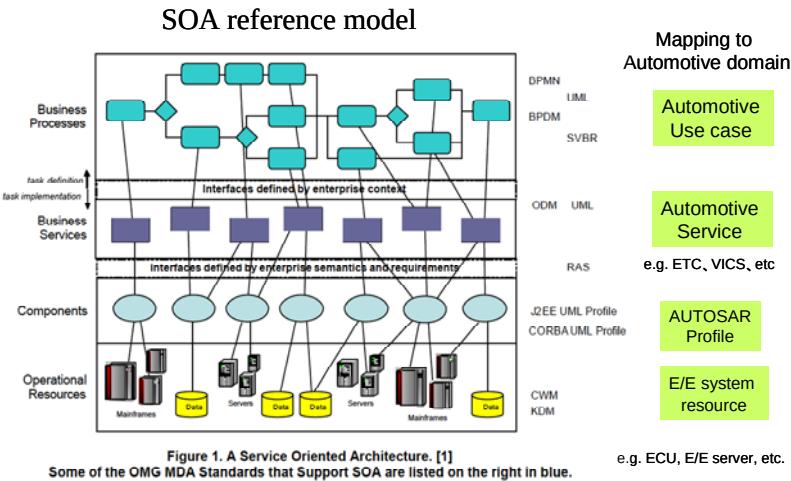


図 2 SOA レファレンスモデルへの自動車マッピング

3.3 車載向けサービスプラットフォームに対する要求

今回、複数のサービス統合システムを想定し、車載向けサービスプラットフォームに対する要求を抽出した。その際、特に、既存の SOA 技術には不足しているリアルタイム性や信頼性の観点を中心に分析した。

表 1 車載向けサービスプラットフォームに対する固有の要求（抜粋）

分類	要求
サービス連携・統合	- 車内の AUTOSAR SWC 同士が双方向連携できる。 - 車外の情報サービスと AUTOSAR SWC が、時間制約内にかつ安全に双方向連携できる。 - 様々なユーザーニーズや車両状況に応じた最適なサービスプロセスを定義できる。 - 通信途絶中や電源遮断時等の不安定なネットワーク状態でも、サービスプロセスが継続可能。
情報共有	- 複数のクルマのユーザー情報・車両情報を、車外より通信回線を経由して安全に呼び出し、一括管理・共有できる。 - クルマの販売後にユーザー情報・車両情報が、車外より安全に追加・削除・変更が可能。
その他	- 車載側の端末コストが低い。 - 障害に対する耐故障性が高い

4. Darwin プロジェクト

本章では、上述の車載向けサービスプラットフォーム（開発コードネーム：DARWIN）についての基本コンセプトとアーキテクチャ、及びコア技術を解説する。

4.1 基本コンセプト

本研究プロジェクト（DARWIN プロジェクト）にて構築した、車載向けサービスプラットフォーム DARWIN は、車外のライフサイクルに合わせて、動的かつ自律的に進化するクルマを基本コンセプトとしている。将来的には、DARWIN は、クルマの購入から 10 年間、ソフトウェア更新のみで車外のライフサイクルに追従でき、また、ユーザーの多様化に対応するため、機能最小セットのクルマを購入し、ニーズに合わせ、必要な機能を拡張できることを目指している。

4.2 DARWIN サービス

DARWIN で扱うサービス（DARWIN サービス）は、通常の Web サービスとは異なる特徴を有している。クルマは移動体であり、移動した場所や時間によって、ドライバーもしくは同乗者に必要なサービスは変化する。DARWIN は、Web サービスのような、いつでも、どこでも利用できるサービスではなく、状況とニーズのマッチングによって、いまここで最適なサービスを提供可能とする事の特徴とする。今後、自動車ビジネスにおいて、クルマのユーザーに対して、移動による付加価値を提供することが重要と成ってくる。

4.3 Darwin アーキテクチャ

(1) 前提とするシステム構成

実際に、サービス統合システムを実現する場合、クルマとインターネットの接続形態や、車両内のネットワーク構成等、様々なシステム構成が考えられる。今回は、一般的な以下のシステム構成を前提とした。

- クルマは、インターネットには直接接続されない。インターネットのサービスを利用する場合は、専用の通信回線を経由する。
- クルマは、専用の車載情報端末（本研究では、DARWIN Gateway）を介して車外のサービスと連携する。
- 車載情報端末は、基幹バスを経由して各車載電子制御機器（ECU）と接続され、車両内ネットワークを構成している。
- 車両内ネットワークは、CAN や LIN, FlexRay 等の車載向け通信プロトコルで

構成されている。

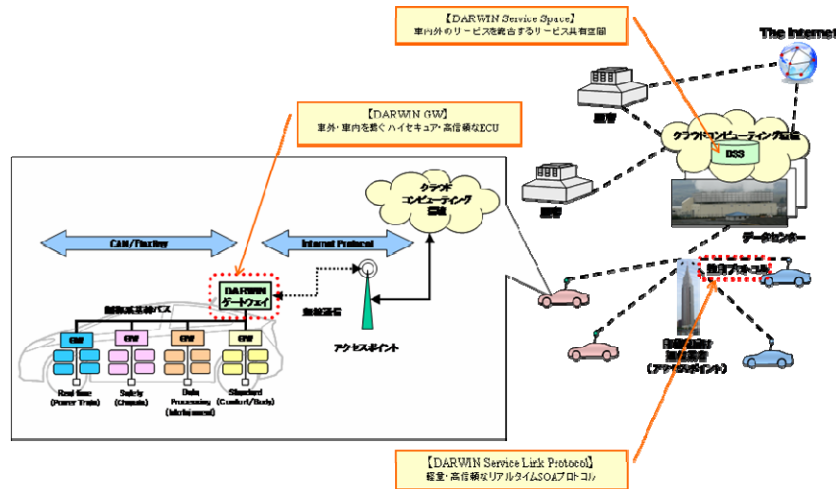


図 3 前提とするシステム構成

(2) 機能アーキテクチャ

SOA をレファレンスとした機能アーキテクチャを図 4 のように定義した。各層は、SOA のリファレンスモデルに対応している。横断的な機能として、各サービスを統合するフレームワークである DSS (DARWIN Service Space) と QoS や Security 管理、システム診断、リカバリー等の非機能要件を含んだ高信頼性 OS がある。機能アーキテクチャは、DARWIN のソフトウェアアーキテクチャを、DARWIN の機能面から抽象化したものであり、実際の各機器に実装されるソフトウェアアーキテクチャとは異なる。

(3) コンセプトアーキテクチャ

コンセプトアーキテクチャとは、車内外のサービスが連携する DARWIN Platform の仕組みを概念的に表現した構成図である。Service Process でサービス統合されたアプリケーションが、SPM (Service Process Manager) によって起動されると、構成要素である車外のサービスを、DSS (DARWIN Service Space) から、SOAP, REST 又は専用サービス連携プロトコル DSP (Darwin Service link Protocol) のインターフェイスを介して呼び出す。また、同時に車内の ECU ソフトウェアコンポーネント (ECU SWC)

を、DSS から、AUTOSAR VFB アダプタを経由して AUTOSAR VFB のインターフェイスを介して呼び出す。本研究において、コアとなる技術が、これら DSS, DSP と SPM である。VFB とは、Virtual Functional Bus の略であり、AUTOSAR で定義されている AUTOSAR SWC を結合する仮想的な機能バスである。

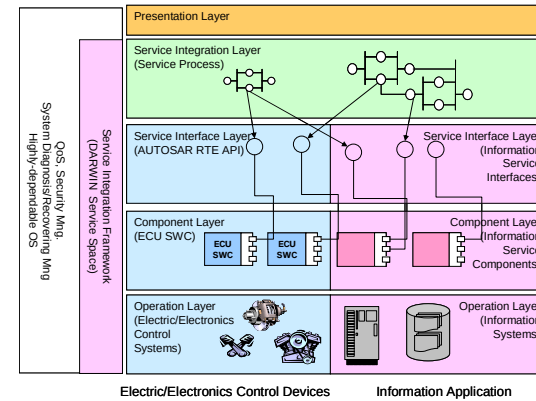


図 4 DARWIN 機能アーキテクチャ

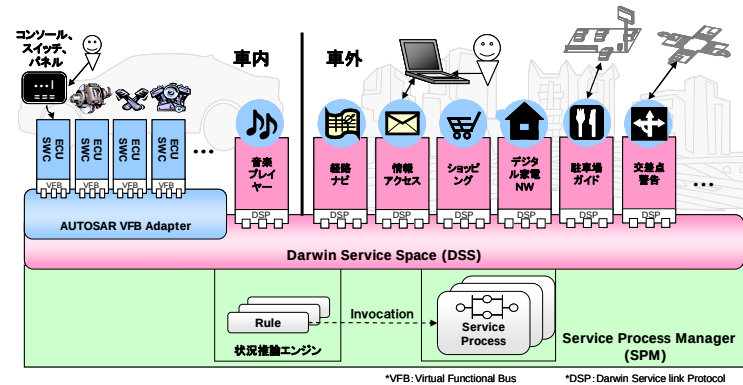


図 5 DARWIN コンセプトアーキテクチャ

4.4 DARWIN Platform のコア技術

(1) DSS (DARWIN Service Space)

現在、ネットワークシステムにおいて、複数の機器に分散した情報やサービスを、共有化する技術として Java Space や CORBA 等がある。しかし、車載向けサービス統合システムの様な、時間制約が厳しく、かつ、高い安全性や信頼性が要求されるシステムにおいては、そのままでは扱えない。従って、本研究では、新たに、これら既存技術をベースに、情報サービス共有空間である DSS (DARWIN Service Space) を開発した。DAWRIN が扱うあらゆるサービス (通常の Web サービスも含めた DARWIN サービス) は、この DSS を介して連携・統合される (図 6)。また、QoS 確保のため、サービスに優先順位を付与し、重要サービスを優先的に実行させる機構を有している。

サービスへのアクセスは、Public/Subscribe セマンティックスで行われる。実際の DSS は、データベースを用いて、複数の機器やサーバへ分散して実装される。また、本研究では、以下の車載向け機能を拡張した。

- AUTOSAR VFB 通信機能 (AUTOSAR VFB アダプタ)
- サービス優先制御
- 通信途絶対応

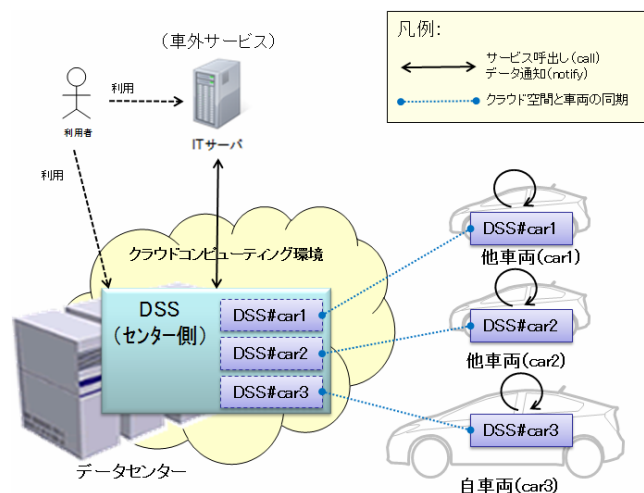


図 6 DSS (DARWIN Service Space) の役割

(2) DSP (Darwin Service link Protocol)

DSP は、REST/NGTP のサブセットであり、REST や NGTP と互換性がある。REST に対して、データ符号化技術を利用したデータ圧縮により軽量化を図っている。REST (Representational State Transfer) とは、HTTP を利用した汎用通信プロトコル仕様であり、サーバが管理するリソース (プロトコル上で扱う論理的な資源) を、URL として表現する特徴を持つ。また、HTTP プロトコル上の操作 (PUT, GET, POST, DELETE) に応じてリソースの操作を行うことになる。通常はリソースの更新 (PUT)、リソースの削除 (DELETE)、リソースの取得 (GET) が利用される。

DSS は、ユーザー情報、自動車情報、ECU サービス情報、車外サービス情報を管理するので、これらをそれぞれ REST のリソースとして扱い、URL で表現する。URL には、ユーザー ID、自動車 ID、サービス ID、インターフェイス ID を設定することで、DSS が管理するリソースを一意に特定することができる。例えば、ECU サービス呼出を行うための URL は、以下のように記述する。

URL 記述例

```
http://darwin.jp/api/001/vehicles/{自動車 ID}/services/{サービス ID}/interfaces/{インターフェイス ID}/call
```

(3) SPM (Service Process Manager)

SPM は、DARWIN サービスのコンセプトを実現するサービスプロセス管理モジュールである。クルマの運転状況に応じたサービスプロセスの起動条件を記述したルールファイル、それを選択し、対応するサービスプロセスを起動するエンジン (SIE: Situation Interface Engine) で構成される。サービス提供者が、予めこのルールファイルを記述しておけば、SPM がクルマの運転状況とルールファイルを照合させ、自動的に必要なサービスを起動する仕組みとなっている。本研究では、ルールファイルを予め全てをサービス提供者が記述する方法を取っているが、推論エンジンや機械学習等の推論機構を応用すれば、もっと自律的にサービスを起動する仕組みができると考えている。

また、本研究でベースとした技術は、Apache ODE, BPEL (Business Process Execution Language) である。BPEL は、サービスプロセスの記述に利用している。

図 7 に、Ruby 言語で記述した「ライト点灯サービス」のプログラム例を示す (本サービス例は、後述する 5.2-(2)節の評価においても使用)。また、サービス呼出しは、上述の通り、REST のリソースとして扱い、URL を指定する。なお、本プログラムは、ユーザーが直接コーディングする必要は無く、ツールにより定義ファイルから自動生成される。

```
require 'net/http'

/* ①「自動車 ID:PRI0001」「サービス ID:SwcLoc」「インターフェイス ID:MainLight」
を URI で指定 */

MainLightUri=
  '/darwin.jp/api/001/vehicles/PRI0001/services/SwcLoc/interfaces/MainLight/call'

/* ②インターフェイスに対する操作内容を指定(XML 形式：1・・・ライト ON)
*/

lightOn=<<-requestData
<Datum>
  <Data>
    <DataTypeID>200</DataTypeID>
    <Value1></Value1>
  </Data>
</Datum>

/* ③DSS に URI と操作内容を送信し、車両操作を実行 */

requestData
def call_car(operation, data)
  Net::HTTP.start('dssserver', 80) do |http|
    headers = {'Content-Type'=>'text/xml; charset=utf-8'}
    response = http.send_request('PUT', operation, data, headers)
  end
end
```

図 7 Ruby 言語による「ライト点灯サービス」のプログラム記述例

5. プロトタイピング実装による評価

サービス統合プラットフォームの要求に対する DARWIN の妥当性を評価するため、プロトタイピング実装を行った。その際、実装期間を短縮する為、標準技術がそのまま使える部分は、オープンソースを中心に、出来るだけ既存のソフトウェアモジュール

ルを流用する方針とした。また、評価項目を、机上検討でも評価できる項目と、実際に実装して見なければ、分からない評価に層別した。特に、リアルタイム性の評価に関しては、後者に当たる為、優先して実施した。本章では、その評価要領と評価結果のポイントを抜粋して解説する。

5.1 評価システム

図 8 にプロトタイプ実装した評価システムを示す。車内には、標準の AUTOASR プラットフォーム (PF) が搭載された複数の ECU (本図では 1 つ) が、CAN 通信、及び FlexRay 通信を用いて車載 LAN に接続されている。夫々の ECU 内には、AUTOASR PF 上で動作する複数の AUTOSAR SWC が実装されている。更に、車外と車内は、Darwin Platform (GW 用) を搭載した DARWIN Gateway で接続される。

一方、車外は、データセンターを模擬したサーバと、サービスプロバイダの運用を想定した Web サーバ、或いは、携帯端末で構成される。今回は、ドライバが携帯する Android 端末とした。Android 端末からは、HMI アプリを使用して、データセンターを介し、DARWIN Gateway 経由で、車内の ECU にアクセスが可能である。

また、データセンターのサーバには、Darwin Platform (サーバ用) が搭載され、車内の機能と車外のサービスを統合した Service Process を実行する事ができる。なお、サービスリポジトリには、市販の組込みデータベースを流用し、通信回線遮断時のリカバリー処理や、トランザクション処理を任せた。また、実験用に、PC は、Intel Core 2 Duo の 1.83GHz クラスを使用した。

表 2 評価システムで流用したソフトウェアモジュール一覧

機器	ソフトウェアモジュール
車載 ECU	AUTOSAR Basic Software Release 3.0 準拠ソフトウェア
DARWIN Gateway	Ubuntu 9.0.4
データセンター	Ubuntu 9.0.4, Sun Java Ver. 1.6.0_14, Apache Tomcat Ver. 6.0.20
Web サーバ	同上
携帯端末	Android R1.5

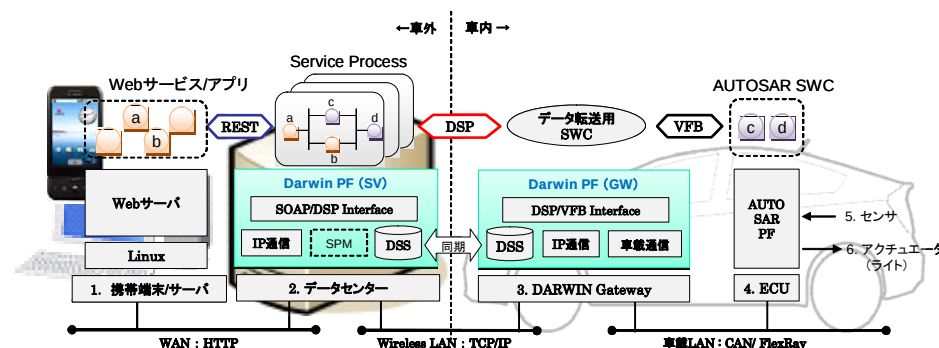


図 8 評価システム構成

5.2 評価結果

(1) DSP (Darwin Service link Protocol) の単体性能評価

図 9 に、今回開発した DSP と REST 及び SOAP のサービス呼び出しの処理時間を示す。条件によって多少のばらつきはあるが、何れのデータサイズにおいても、DSPの方が高速である結果となった。なお、処理時間は、サンプルデータを送信データ形式に変換して要求を発行し、サーバ側で元のサンプルデータに複合するまでに要する時間である。

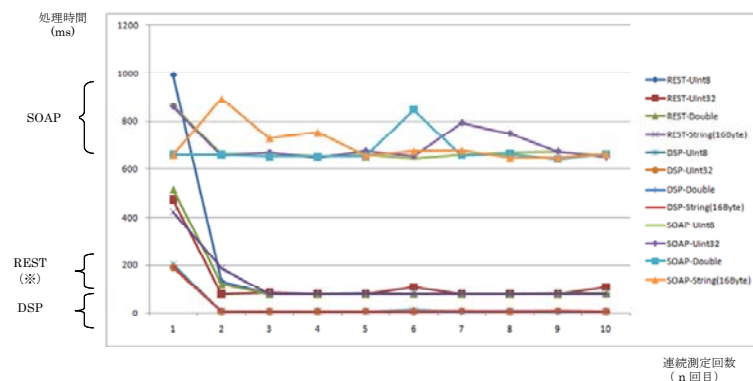


図 9 サービス連携プロトコル別処理時間

なお、初回は、各モジュールともに初期化処理時間が生じるため処理速度が遅い結果となっている。

(2) DSS (DARWIN Service Space) のリアルタイム性能評価

以下、評価シナリオにより、DSS の性能を含めた DSP の End to End のリアルタイム性能を評価した。今回、車載 ECU に接続されたライトを携帯端末から点灯するシナリオと、センサ情報を携帯端末に表示させるシナリオを用いた。なお、参考として、詳細シーケンスを載せるが、ポイントとなる処理のみを記述している。

●車外サービスから車内サービスを呼び出す (図 10) (図 12)

データセンター内のサービスより、DARWIN Gateway を介して、車載 ECU 上の車内サービスを呼び出す。なお、図 12 の処理シーケンスでの、「2. ライト点灯要求」は、図 7 で定義したライト点灯プログラムを呼び出している。

●車内から外部サービスに車両情報を提供する (図 11) (図 13)

車載 ECU から、DARWIN Gateway を介して、データセンターに車両情報が通知される。

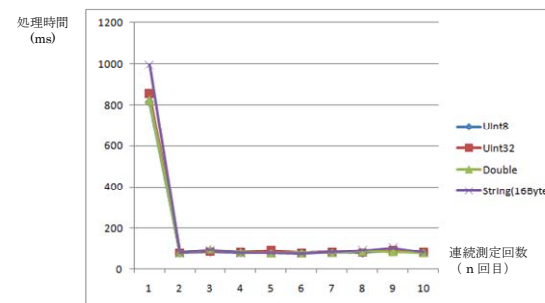


図 10 End to End リアルタイム性能
車外サービスから車内サービスの呼び出し

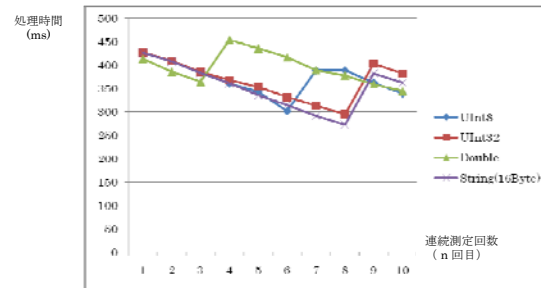


図 11 End to End リアルタイム性能
車内から外部サービスに車両情報を提供する

測定結果より、車外サービスからの車内サービス呼び出しに関しては、目標とした 200ms 以内（初回は除く）のリアルタイム性は確保されている事が確認できた。しかし、車内からの外部サービスへの車両情報提供に関しては、目標性能に達しなかった。なお、この原因は、後述する優先制御の問題と同様と考えられる。

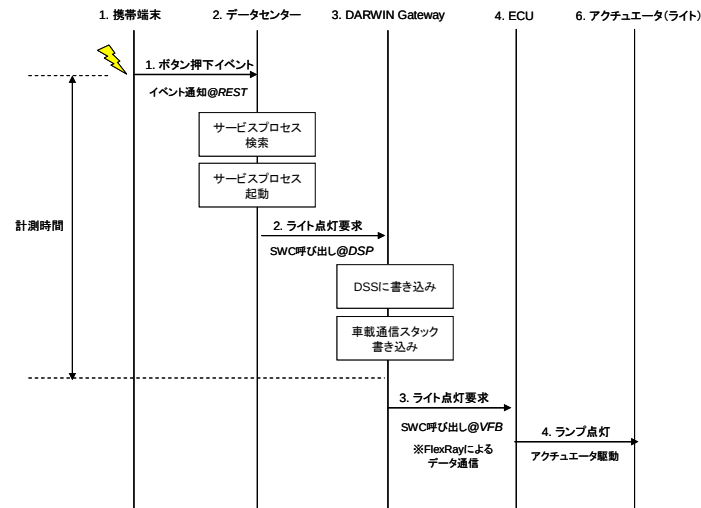


図 12 車外サービスからの呼び出しシーケンス

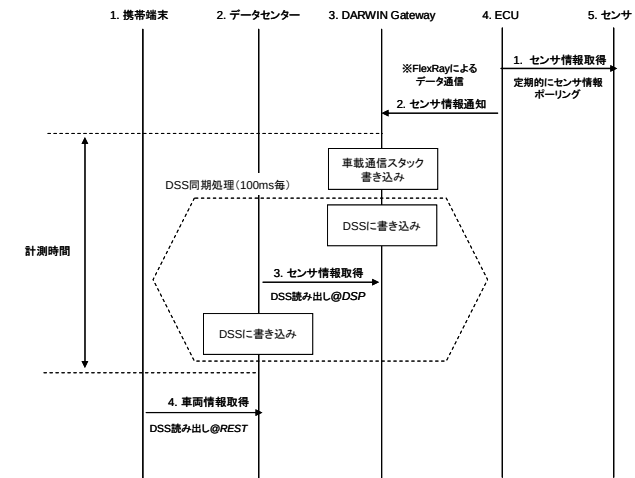


図 13 車内からの車両情報提供シーケンス

(3) DSS の優先制御の性能評価

複数の端末から車内サービス呼び出しが発生している状況下で車載 ECU のサービスを呼び出し、優先制御（4.4-(1)節参照）の応答性を評価した。また、車内から継続的に DSS の呼び出しが発生している状況下での車両情報の更新時間も計測した。

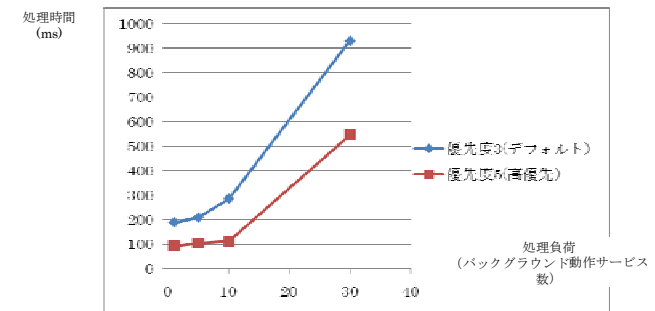


図 14 複数のサービス呼び出し時の処理時間

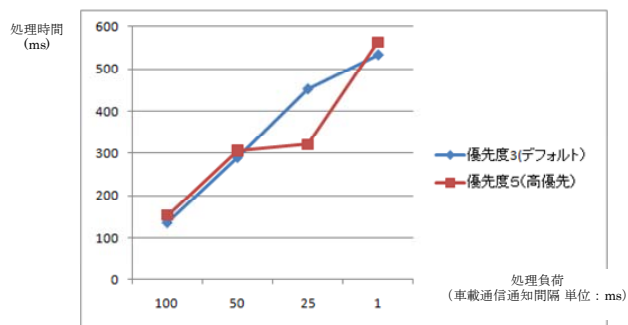


図 15 車載通信が高負荷時の車両情報の更新時間

本評価より、優先制御は、車外サービスからの車内サービス呼び出しに対しては有効であるが、車内からの DSS の呼び出しに関しては、余り、有効でない結果となった。その原因としては、以下のものが考えられる。

- Java 環境の制約：実行コンテキストの優先度が、実際の処理時間に反映されない
- DB の IO ネック：DB への入出力がボトルネックとなり待ち合わせが生じて、優先制御設定に効果が出ていない（DSS の同期処理）
- 負荷のばらつき：ガベージコレクションなど非同期で実行される Java VM 処理に実行資源が使われるため、均一の負荷状況で測定が行えていない

5.2.1 その他の評価

プロトタイプ実装による評価では、その他、車載向けプラットフォームに対する要件に関しての様々な評価を実施し、DARWIN の有効性を確認した。今回は、紙面の都合上、詳細は割愛する。

5.2.2 考察

今回の評価では、最初の原理試作という位置付けで、特にリアルタイム性に着目し、SOA アプローチによる解決の現実性を評価した。その為、出来るだけ IT 分野での汎用のソフトウェアモジュールを流用するという方針をとり、実用化の目処付けを行った。その結果、最初としては、概ね、期待した効果が得られ、原理的な検証は出来たものの、製品化に向けては、やはり解決すべき課題が多いことを再認識した。次のステップでは、実際の製品スペックを考慮したプロトタイプ実装を行

い評価する。

例えば、Java の VM に、リアルタイム制御に適合された RTSJ 準拠（Real-Time Specification for Java）[9]の VM を利用する等、組込み分野に特化したソフトウェアの使用も検討する。また、今回、DARWIN Gateway に用いたプロセッサは、汎用のペンティアム 1GHz クラスのチップセットであったが、次回は、ARM Cortex A9 等の組込みマイコンの採用も検討する。

6. まとめ

本研究において、自動車分野でのサービス統合プラットフォームの課題を明らかにし、SOA をリファレンスとする車載向けサービスプラットフォーム（開発コードネーム：DARWIN）による解決アプローチに関して検討を行った。特に、本報告においては、車内外のサービスを共有する仮想的なサービス共有空間である DSS（Darwin Service Space）と DSP（Darwin Service link Protocol）のリアルタイム性に関して評価を行い、有効性の確認と今後の課題を抽出した。

6.1 今後の課題

今後は、Java 環境や DB の性能改善を検討し、DSS のリアルタイム性の改善に取り組む。また、今回、評価工数不足のため、机上しか出来なかった、もう一つのコア技術である SPM（Service Process Manager）の評価を、状況推論エンジンの改善と合わせて行う。更に、車載システムにおいて最も重要である、セキュリティ管理、システム診断等の安全性・信頼性に関する検討・評価も、充分ではないため、リアルタイム性の改善と合わせて、今後の研究テーマとして、深堀を進めていく。

参考文献

- 1) Next Generation Telematics Protocol (NGTP) Web site, <http://www.ngtp.org/>
- 2) GENIVI Web site, <http://genivi.org/>
- 3) AUTOSAR Web site, <http://www.autosar.org/>
- 4) 青山 幹雄, サービス指向アーキテクチャの誕生と進化, ソフトウェアエンジニアリング最前線 2008 (情報処理学会ソフトウェアエンジニアリングシンポジウム 2008 論文集), 近代科学社, Sep. 2008, pp. 9-16.
- 5) Mark Panahi, Weiran Nie, Kwei-Jay Lin, "A Framework for Real-Time Service-Oriented Architecture," cec, pp.460-467, 2009 IEEE Conference on Commerce and Enterprise Computing, 2009
- 6) 宮脇 聡, 中道 上, 青山 幹雄, 佐藤 洋介, 岩井 明史, 車載ソフトウェアのためのイベント駆動サービス指向アーキテクチャの提案と評価, 情報処理学会 創立 50 周年記念(第 72 回)全国大会 講演論文集 (1), No. 3P-4, Mar. 2010, pp. 481-482.
- 7) 永東 丈寛, 中道 上, 青山 幹雄, 佐藤 洋介, 岩井 明史, サービス指向車載ソフトウェアの協調制御におけるタイミング設計方法の提案と評価, 情報処理学会 第 167 回 ソフトウェア工学研究会, Vol. 2010-SE-163, No. 14, Mar. 2010, pp. 1-8[電子出版].
- 8) The OMG and Service Oriented Architecture,
<http://www.omg.org/attachments/pdf/OMG-and-the-SOA.pdf>
- 9) Real-Time Specification for Java (RTSJ) Web site, <https://rtsj.dev.java.net/>