Vol.2017-ARC-225 No.5
Vol.2017-SLDM-179 No.5
Vol.2017-EMB-44 No.5
2017/3/9

次世代車載システム向けサービス指向ミドルウェア SOME/IP-SDの性能評価

水谷 太貴[1,a)]　松原 豊[1]　高田 広章[1]

概要：先進運転支援システムや自動運転の実装により環境の変化が非常に加速している自動車業界では，車載システムの柔軟性や再利用性が求められるようになってきた．それらをサポートするため，サービス指向通信ミドルウェアである SOME/IP（ScalableService-Oriented MiddlewarE over IP）の標準化が進められている．車載ネットワークの通信ミドルウェアに対する性能要件として，サービスの利用確立までの遅延や，通信メッセージのリアルタイム性などがある．サービス指向はサービスをネットワーク上で連携させてシステムの全体を構築していくため，動的なネットワークの設計を行う必要がある．従来，静的にネットワーク設計を行っていた車載システムに比べてオーバーヘッドが増加することは明らかである．本研究では，SOME/IP の Service Discovery における通信確立遅延を測定し，考察する．

Performance Evaluation of Service Oriented Middleware SOME/IP-SD for Next Generation Automotive Systems

Mizutani Taiki[1,a)]　Matsubara Yutaka[1]　Takada Hiroaki[1]

Abstract: Flexibility and reusability are becoming increasingly demanded in the automobile industry where changes in the environment are accelerating due to implementation of advanced driving support systems and automatic driving. In order to support them, standardization of SOME/IP (ScalableService-Oriented MiddlewarE over IP) which is a service-oriented communication middleware is proceeding. Performance requirements for communication middleware of in-vehicle network include delay until establishment of utilization of service and real time property of communication message. In service orientation, it is necessary to dynamically design the network in order to construct the whole system by linking the services on the network. It is obvious that overhead is generated in an in-vehicle system which has conventionally designed a network statically. In this research, we measure and consider the communication establishment delay in Service Discovery of SOME/IP.

1. はじめに

先進運転支援システムが普及期に入り，完全自動運転も大きな注目を集めている自動車業界は，環境の変化が非常に早く，システムが複雑化している．そのため，環境の変化に迅速に対応できるように，柔軟性や拡張性がアプリケーションに強く求められるようになってきている．そこで注目され始めたのがサービス指向という考え方である．

サービス指向とは，コンピュータ・システムを構築する際の手法の一つであり，業務上の一処理に相当するソフトウェアの機能をサービスと見立て，そのサービスをネットワーク上で連携させてシステムの全体を構築していくことを指す言葉である．よって，再利用性の高いサービスを組み合わせてアプリケーションを柔軟に構築できるアーキテクチャと言える．サービス指向の実現をサポートするため，サービス指向通信ミドルウェアである SOME/IP（Scalable Service-Oriented MiddlewarE over IP）の標準化が進められている [1].

車載システムの通信ミドルウェアに対する性能要件として，サービスの利用確立までの遅延や，通信メッセージのリ

[1] 名古屋大学大学院 情報科学研究科
Grduate School of Information Science,Nagoya University
[a)] taiki421@ertl.jp

Vol.2017-ARC-225 No.5
Vol.2017-SLDM-179 No.5
Vol.2017-EMB-44 No.5
2017/3/9

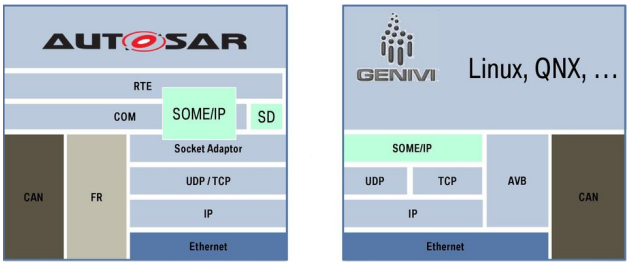

図 1　SOME/IP の位置づけ

アルタイム性などがある．SOME/IP は比較的最近にプロトタイプ実装である vSomeIP が公表された [2] ため，性能評価が十分にされていない．サービス指向では動的にネットワーク設計を行う場合がある．従来，静的にネットワーク設計を行っていた車載システムにおいてオーバーヘッドが増加することは明らかである．本研究では，そのオーバーヘッドを調査し，明らかにする．

2. SOME/IP

2.1 概要

SOME/IP とは “Scalable Service-Oriented MiddlewarE over IP” の略称であり，メッセージの制御に用いられるサービス指向の車載ミドルウェアの仕様である．インフォテイメント機能や先進運転支援システム用の通信システムとして BMW が中心となり，2011 年に開発された [3]．

SOME/IP は以下の目標特性を達成している [1]．

- 組込みシステムにおいて，資源消費に関するハードウェアの要件を満たす
- できるだけ多くのユースケースや通信相手と互換性を持つ
- AUTOSAR と互換性がある
- 幅広い規模のプラットフォームを想定した拡張性
- 異なる OS（AUTOSAR，Linux，OSEK など）において実装可能であり，また OS のないデバイスにおいても埋め込むことができる

図 1 に SOME/IP の位置づけを示す．SOME/IP は TCP/IP プロトコルに基づき通信をサポートし，AUTOSAR や Linux などさまざまな OS に対応している．

2.2 サービス指向

サービス指向とは，コンピュータ・システムを構築する際の手法の一つである．業務上の一処理に相当するソフトウェアの機能をサービスと見立て，そのサービスをネットワーク上で連携させてシステムの全体を構築していくことでソフトウェアを実装する．サービス指向の必要条件として以下のことが挙げられる．[4]

- サービスは人間にとって意味のある粒度であり，それを構成単位とする．
- オープンで標準化されている技術仕様を用いてサービスのインタフェースが定義され，それに従った呼び出し，応答が可能である．
- サービスをネットワーク上で連携させてシステムの全体を素早く構築できる．

ソフトウェアを部品化して呼び出し規約を標準化し，その組み合わせでシステムを構築していく手法は分散オブジェクト技術など従来から存在するが，部品化の単位は細かいプログラム上の機能であるのに対して，サービスは人間の視点で意味のあるものの単位で部品化されたものを指す．

サービス指向の特徴はプログラムの柔軟性と再利用性であるといえる．サービスの実装方法が自由なことから各サービスは疎結合であり，システムの変更や機器の交換，改良が柔軟に行える．また個々のサービスは自由に組み替えることができ，実装方法は自由であるので違うソフトウェアであってもサービスを再利用することが可能である．以上のことからサービス指向のメリットは以下のように考えられる．[4]

- 既存のアプリケーションの機能をサービス化して新規システムに組み込んで利用することが可能であるため，既存アプリケーション資産の有効活用ができる．
- 既存アプリケーションの機能を再利用することで，新規システムの開発から展開までの期間が短縮できる．
- アプリケーション間の依存度を低減できるため，修正箇所が局所化され，システムの構築／変更／運用コストが下がる．

これよりサービス指向は通信速度や処理速度などの性能ではなくソフトウェアの開発側の利点が大きいことがわかる．自動運転やコネクテッド技術など自動車業界は，環境の変化が非常に早く，複雑化している．そのため，アプリケーションにも，環境の変化に迅速に対応できるように柔軟性や拡張性が強く求められるようになってきている．サービス指向は 20 年ほど前から存在した手法であるが，このような側面より現在，自動車業界で注目されている．

次に，サービス指向で実現されるアプリケーションの例を図 2 に示す．サービス指向において，サービスを提供するサービスアプリケーションとサービスを利用するクライアントアプリケーションが存在する．この二つが通信することでサービスを利用し，サービス指向を実現する．一連の流れとしてはまずクライアントアプリケーションは必要とするサービスを呼び出すためのリクエストが送信する．リクエストを受け取ったサービスアプリケーションはリクエストに基づき処理を行い，結果をレスポンスとして返す．この結果からクライアントアプリケーションは再び処理を進める．このやり取りを必要に応じて繰り返すことでサービスによりソフトウェアを構築することができる．

Vol.2017-ARC-225 No.5
Vol.2017-SLDM-179 No.5
Vol.2017-EMB-44 No.5
2017/3/9

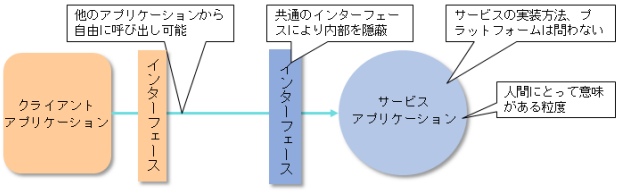

図 2　サービス指向概要

3. Service Discovery

3.1 概要

Service Discovery(以下「SD」という) とは利用可能なサービスを探索し通信を確立する機能である．SOME/IP はサービス指向ミドルウェアであるため，ソフトウェア機能を独立した「サービス」という単位で実装し，それらを組み合わせ，システムを作り上げる．よってサービスの状況を把握し，通信を確立する機能は非常に重要になってくるのである．サービスを提供するサービスアプリケーションは offer message，サービスを利用するクライアントアプリケーションは find message をマルチキャスト通信をし，通信確立を試みる．

3.2 SD におけるサービスとクライアントの動作

サービスとクライアントには以下の 3 つのフェーズが存在する．[5]

(1) Initial Wait Phase
(2) Repetition Phase
(3) Main Phase

3.2.1 Initial Wait Phase

クライアント

このフェーズでは指定された時間待つ．クライアントが探していたサービスの offer message を受信した場合，クライアントは Repetition Phase を飛ばし，Main Phase に直接移る．

サービス

クライアントと同様に，指定された時間待つ．クライアントと異なるのは，受信した find message は無視される．よって，サービスが Repetition Phase を飛ばすことはない．

3.2.2 Repetition Phase

このフェーズでは find message と offer message を短い周期で送信し，短い通信確立遅延を可能にする．

クライアント

Repetition Phase に入るとすぐに，クライアントはマルチキャスト通信にて find message を送信する．送信されたメッセージのすべては *FndCnt* にてカウントされる．*FndCnt* が *CltRepMax* と一致した場合，またはサービスから offer message を受信した場合，Repetition Phase は終了する．find message の送信間隔は以下の式で計算できる．

$$2^{FndCnt} \times CltRepDel \tag{1}$$

数式（1）より find message の送信間隔は，ベース間隔である *CltRepDel* に送信するたびに 2 を乗算したものとなる．

サービス

サービスでの offer message の送信間隔はクライアントとほぼ同様の振る舞いをする．find message を受信した場合，*SdServerTimerRequestResponseMinDelay* と *SdServerTimerRequestResponseMaxDelay* の間で乱数を取り，その時間待ち，find message の送信者に対して，offer message をユニキャスト通信する． サービスでは find message を受信したとしても，offer message の最大数である *SvcRepMax* に届くまでサービスは offer message をマルチキャスト通信し続ける．

3.2.3 Main Phase

このフェーズの目的は SD によってできるネットワーク負荷をなるべく低くさせることである．

クライアント

find message は送信しない．

サービス

SvcCycDel の周期で offer message を送信し続ける．find message を受信した場合，Repetition Phase と同様に *SdServerTimerRequestResponseMinDelay* と *SdServerTimerRequestResponseMaxDelay* の間で乱数を取り，その時間待ち，find message の送信者に対して，offer message をユニキャスト通信する．

3.3 SD の動作モード

SD はサービスとクライアントに各 2 種類ずつ通信確立に関するモードが存在する [5].

クライアント

request mode

Repetition Phase にて find message を送信する．

listen mode

find message は送信せず，サービスから offer message が送信されるのを待機する．

サービス

offer mode（Service）

Repetition Phase と Main Phase にて offermessage を送信する．

silent mode mode（Service）

クライアンからト find message を受信した場合のみ，offer message を送信する．

Vol.2017-ARC-225 No.5
Vol.2017-SLDM-179 No.5
Vol.2017-EMB-44 No.5
2017/3/9

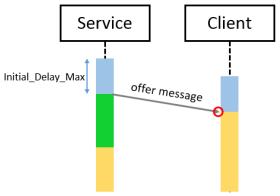

図 3　通信確立パターン 1

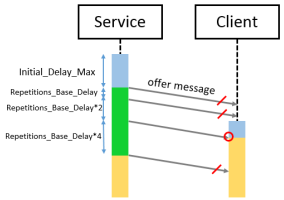

図 4　通信確立パターン 2

3.4　SD における通信確立パターン

Phase によって offer message の送信周期が異なるため，クライアントとサービスの起動タイミングにより通信確立遅延は変わっていくと考えられる．今回，条件として，クライアントは listen mode，サービスは offer mode とし，その場合の通信確立時のパターンを考える．

1 つ目の通信確立の状況を図 3 に示す．これは，サービスが先に起動し，サービスの Repetition Phase の最初の offer message にて通信確立が行われる場合である．この場合では，クライアントは最大で Initial Wait Phase の時間分，offer message を待つことになる．

2 つ目の通信確立の状況を図 4 に示す．これは，サービスが先に起動し，サービスの Repetition Phase の 2 回目以降の offer message にて通信確立が行われる場合である．この場合はクライアントは最大で Repetition Phase のメッセージ送信周期分待つことになる．周期は数式（1）を参考にすると，メッセージ送信回数が増えるごとに周期が 2 倍になっていくので，クライアントの最大待ち時間も同様に増加していく．

3 つ目の通信確立の状況を図 5 に示す．これは，サービスが先に起動し，サービスの Main Phase に送信される offer message にて通信確立が行われる場合である．この場合はクライアントは最大で Main Phase のメッセージ送信周期分待つことになる．

4 つ目の通信確立の状況を図 6 に示す．これは，クライアントが先に起動し，サービスの Repetition Phase の最初の offer message にて通信確立が行われる場合である．この場合はサービスの Initial Wait Phase の時間分，通信確立まで必ず時間がかかる．

4.　評価

4.1　評価環境

SOME/IP の性能を評価するにあたって，GENIVI が配布する SOME/IP プロトコルの実装である vSomeIP を使用した．今回の評価で使用したバージョンは 2.0.1 である．測定環境としては，以下を使用した．

- ボード：MinnowBoard Turbot
  - CPU：Intel Atom E3826 1.46 GHz Dual Core
  - メモリ：2GB DDR3L RAM
  - OS：Linux 3.19.5

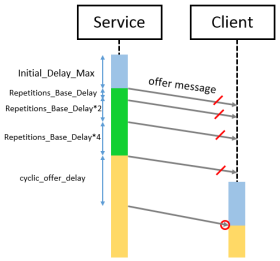

図 5　通信確立パターン 3

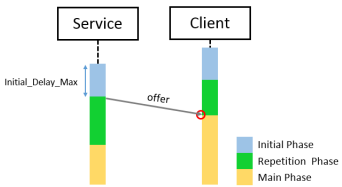

図 6　通信確立パターン 4

表 1　vSomeIP の設定値

変数	設定値
repetitions_base_delay	200[ms]
repetitions_max	3[回]
cyclic_offer_delay	200[ms]
initial_delay_max	100[ms]

このボードを 2 つ用意し，Ethernet 規格の LAN ケーブルによって接続，通信を行う．

vSomeIP は SOME/IP の SD に関する設定値を設定ファイルにて定義する．今回使用した変数と設定値の対応を表 1 に示す．*repetitions_base_delay* は数式（1）の *CltRepDel* に対応しており，Repetition Phase の周期を決定するベース周期を表す．*repetitions_max* は Repetition Phase における最大メッセージ送信回数を示す．*cyclic_offer_delay* は Main Phase におけるメッセージ送信周期を示す．*initial_delay_max* は Initial Wait Phase の時間を示す．

使用した SOME/IP プロトコルのオープン実装 vSomeIP は，割り込みやプロセススケジューリングの影響を受ける．この影響をなるべく小さくするためにプロセスの優先度を変更し，時間測定内において関係のないプロセスが入り込まないように考慮した．Linux のスケジューリングにおける優先度は 140 段階であり，カーネル内の表現方法では，値が小さいほうが優先度が高く，0〜99 はリアルタイムタスク，100〜139 はノンリアルタイムタスクとなっている．今回，使用するアプリケーションをそれぞれ優先度が最大である 0 とした．また，コアのマイグレーションによるオーバーヘッドが測定結果に影響しないよう，それぞれアプリケーションに使用するコアを割り当て，固定した．

4.2　評価対象

測定には，vSomeIP 2.0.1 のサンプルプログラムである request-response を使用する．request サンプルはクライアントアプリケーションであり，listen mode で動作する．また，respone サンプルはサービスアプリケーションであり，offer mode で動作する．vSomeIP における通信確立の流れを図 7 に示す．まずアプリケーションは起動後，start

Vol.2017-ARC-225 No.5
Vol.2017-SLDM-179 No.5
Vol.2017-EMB-44 No.5
2017/3/9

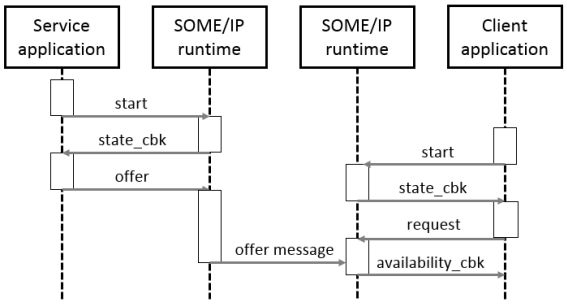

図 7　通信確立のシーケンス図

関数より SOME/IP の runtime を起動させる．runtime の初期化が完了したら，state_cbk というコールバック関数が呼ばれる．その後，クライアントは要求したいサービスを request 関数にて要求する．また，サービスは offer 関数により利用可能なサービスを申請し，クライアントに offer message が送信される．offer message により通信が確立した場合，availability_cbk というコールバック関数が呼ばれ，クライアントは要求したサービスが利用可能であることが通知される．

これより，後に起動したアプリケーションの立ち上がり時刻を測定開始時間とし，availabilty_cbk によりサービス利用可能の通知が来るまでの時間を通信確立遅延とする．

4.3　評価結果

4.3.1　起動タイミングのズレによる遅延

3.4 節で述べたように，起動タイミングによって通信確立遅延は異なる．よって起動時間差を 20ms ずつずらしていき，通信確立遅延を測定することで通信確立遅延と起動タイミングの関係性を評価する．

サービスを先に起動した測定結果を図 8，クライアントを先に起動した測定結果を図 9 に示す．図 8 中の①は図 3 で示された通信確立パターン 1 の場合である．Initial Wait Phase の時間である *initial_delay_max* の設定値 100ms が最大遅延となっているのが図から読み取れる．これはクライアントとサービスが同時に起動した場合であり，offer message が送信されるまでのサービスの Initial Wait Phase 時間分待たなければいけないからである．

図 8 中の②は図 4 で示された通信確立パターン 2 の場合である．Repetition Phase のベースメッセージ送信周期である *repetitions_base_delay* の設定値の 200ms とそれを基準に 2 倍ずつされた 400ms，800ms が周期内での最大遅延になっている．これは offer message が受信可能になる直前に offer message が送信されてしまった場合，その周期分，最大で待つことになるからである．以上のことから，この範囲の最大遅延を一般的に示すと

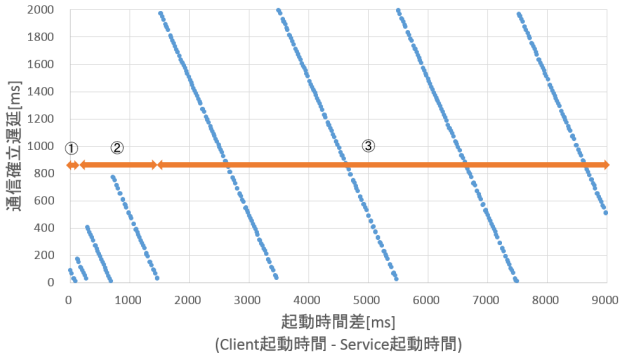

図 8　サービスが先に起動した場合の起動時間差と通信確立遅延

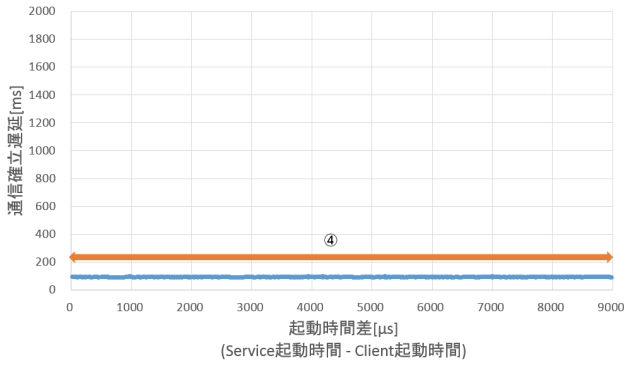

図 9　クライアントが先に起動した場合の起動時間差と通信確立遅延

$2^{initial\_delay\_max} \times repetitions\_base\_delay$ と表せる．

図 8 中の③は図 5 で示された通信確立パターン 3 の場合である．Main Phase のメッセージ送信周期である *cyclic_offer_delay* の設定値の 2000ms が最大遅延となっている．これも同様に offer message が受信可能になる直前に offer message が送信されてしまった場合，その周期分，最大で待つことになるからである．

図 9 中の④は図 6 で示された通信確立パターン 4 の場合である．Initial Wait Phase の時間である *initial_delay_max* の設定値 100ms が遅延となっているのが図から読み取れる．これは通信確立の起因となる offer message が送信されるまでのサービスの Initial Wait Phase 時間分必ず待たなければいけないからである．

4.3.2　SD のオーバーヘッド

4.3.1 で測定した結果は起動タイミングによって offer message を待つ時間が大きく影響していた．この測定ではクライアントが待つ時間を除いた SD の処理によるオーバーヘッドについて測定する．サービスが先に起動した場合，タイミングによってはクライアントがほぼ待つことな

Vol.2017-ARC-225 No.5
Vol.2017-SLDM-179 No.5
Vol.2017-EMB-44 No.5
2017/3/9

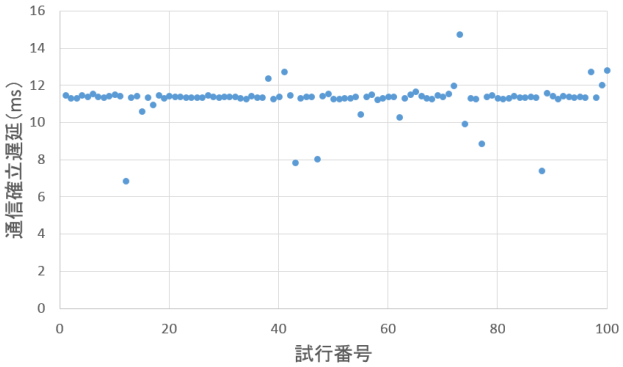

図 10 SD のオーバーヘッド

表 2 SD のオーバーヘッド集計結果

平均時間 [ms]	最悪時間 [ms]	最小時間 [ms]	標準偏差
11.27	14.78	6.86	9.44

く offer message を受信できるタイミングがある．そのタイミングでの通信確立遅延を測定することで，SD の処理によるオーバーヘッドを測定できると考える．4.3.1 の測定結果より，サービスが先に起動した場合の起動時間差が 5480ms〜5500ms の範囲でクライアントがほぼ待つことなく offer message を受信できているとわかった．以上より，起動時間差 5480ms〜5500ms を 1ms 刻みで通信確立遅延測定し，その範囲での最小遅延を SD 自体のオーバーヘッドとする．図 10 に測定を 100 回行った散布図を示す．また，表 2 に集計結果を示す．

今回，1ms ずつ起動タイミングをずらして測定したため，条件における誤差は 1ms 以下である．よってシステムにおけるバラつきは 3ms ほどであると言える．

4.4 考察

図 8，図 9 より SOME/IP の設定値は通信確立遅延に大きく影響することがわかった．ここでは，適切な設定値について考察を行う．

まずは Initial Wait Phase の時間を設定する *initial_delay_max* について考える．Initial Wait Phase の目的は，バスの混雑を防ぐために指定時間待つことである．よって ECU によって待ち時間をそれぞれ設定する必要があるのだが，あまりに大きすぎると通信遅延時間を大きくすることに繋がってしまう．特にクライアントが先に起動した場合，通信確立遅延は Initial Wait Phase の時間とほぼ一致する．よって，この値は ECU が同時に起動しない程度にずらして設定し，ほかの設定値よりも小さく設定するのが最適だと考えられる．

次に，Repetitiom Phase のベース周期である *repetitions_base_delay*，Repetitiom Phase でのメッセージ送信回数である *repetitions_max*，Main Phase のメッセージ送信周期である *cyclic_offer_delay* について考える．Repetition Phase の目的はアプリケーション起動後のすばやい通信確立，Main Phase の目的はアプリケーションの動作・通信に影響しない程度で通信確立を目指すことである．よって，Repetition Phase のメッセージ周期が Main Phase のメッセージ周期を超えることは設計意図に反することとなる．以上のことから数式 (2) を満たす必要があると考える．左辺は Repetition Phase の最大メッセージ周期の計算式であり，右辺が Main Phase のメッセージ周期である．

$$2^{repetitions\_max} \times repetitions\_base\_delay < cyclic\_offer\_delay \quad (2)$$

以上が設定値に関する考察であるが，具体的な数値は現時点で算出するのは難しい．設定値を小さくすれば通信遅延は，4.3.2 で測定したオーバーヘッドに近づけることができる．しかし，offer message はマルチキャスト通信によって送信されるため，帯域幅の占領につながる．このことから，通信確立遅延と帯域幅の確保はトレードオフの関係にあると考える．よって，具体的な設定値を算出するには ECU の個数や通信データ量などを実際の自動車に模擬した測定が必要であると考える．

5. おわりに

本稿では車載システム向けサービス指向ミドルウェア SOME/IP の Service Discovery による通信確立の遅延時間を評価した．評価により，通信確立遅延と SOME/IP の設定値の関係を示し，設定値の満たすべき条件について考察した．

今回，測定環境として 1 対 1 の通信を仮定した．しかし，SD により送信される offer message や find message はマルチキャスト通信により送信されるため，通信相手が増加するにつれて処理が大きくなると考えられる．今後の課題として通信相手の数と通信確立遅延の関係性について調査する必要がある．

参考文献

[1] Kirsten Matheus and Thomas Knigseder: *Automotive Ethernet*, pp.159-166, (2014).
[2] vsomeip, 入手先 ⟨http://docs.projects.genivi.org/vSomeIP/1.3.0/html/README.html⟩ (2016.01.29 参照).
[3] someip, 入手先 ⟨http://some-ip.com/⟩ (2016.01.29 参照).
[4] 米持 幸寿: 基礎からわかる SOA(サービス指向アーキテクチャ), 日経 BP 社 (2005).
[5] Jan R. Seyler,Thilo Streichert,Michael Gla,Nicolas Navet,Jrgen Teich: *Formal analysis of the startup delay of SOME/IP service discovery*, (2014).