

次世代車載向け通信プロトコル SOME/IP と DDS 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 ADAS や自動運転の実現に向けて、車載制御システムに対する機能要件や性能、アーキテクチャの柔軟性や再利用性が求められている。これらに対する車載ネットワークを実現するため、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。一方、航空交通システムや金融システムなどで採用実績のあるデータ中心指向である DDS(Data Distribution Service) も候補として挙げられる。本論文では、SOME/IP と DDS を対象に、機能と通信性能の両面から、比較・考察する。

Performance evaluation of next-generation automotive communication protocol SOME / IP and DDS

MIZUTANI TAIKI^{1,a)} MATSUBARA YUTAKA¹ TAKADA HIROAKI¹

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、アプリケーション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。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、システムの全体を構築する手法である [1]。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。車載システムにおいて、サービス指向アーキテクチャをサポートするため、車載システム向け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 [2]。一方で、金融システムや航空交通システムなどの分散リアルタイムシステム

で採用実績のあるデータ中心ミドルウェアプラットフォーム DDS(Data Distribution Service) の採用も検討されている。本論文では、SOME/IP と DDS を対象に、機能と通信性能の両面から、比較・考察する。

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。インフォテイクメント機能や先進運転支援システムのメッセージを制御するために、BMW が中心となり、2011 年に開発された。図 1 に SOME/IP の位置づけを示す。SOME/IP は TCP/UDP プロトコルに基づく通信をサポートし、主に車載電子制御用の AUTOSAR Classic Platform や自動運転向けの Adaptive Platform、またはインフォテイクメント向けの Linux など様々な車載ソフトウェアプラットフォームで採用されている [3]。

SOME/IP は通信機能として Pub/Sub 通信と RPC に対応している。図 2 に SOME/IP の Pub/Sub 通信の例を示す。まず通信相手を探索する SOME/IP-SD(Service Dis-

¹ 名古屋大学大学院 情報科学研究科
Graduate School of Information Science, Nagoya University
^{a)} taiki421@ertl.jp

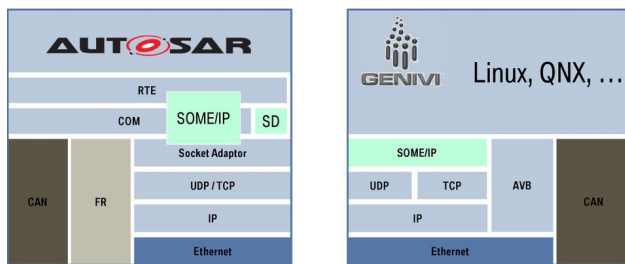


図 1 SOME/IP の位置づけ

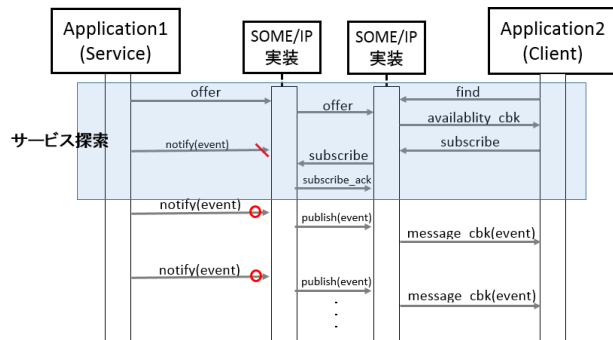


図 2 SOME/IP の Pub/Sub 通信例

covery) の機能を用いる。Service アプリケーションが offer message, Client アプリケーションが find message を送信しあうことで、通信相手を探索する。マッチングが完了した後に subscribe を行い、イベントの通信が行われる。

2.2 サービス指向

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。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、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。サービス指向のメリットは以下のよう考えられる。

- 既存アプリケーション資産の再利用によるコスト削減
- 開発、テスト、更新はサービスごとに独立して可能
- システムの部分的アップデートの容易化

以上より、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がわかる。

自動運転やコネクテッド技術など車載ソフトウェアの環境の変化は非常に早く、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。

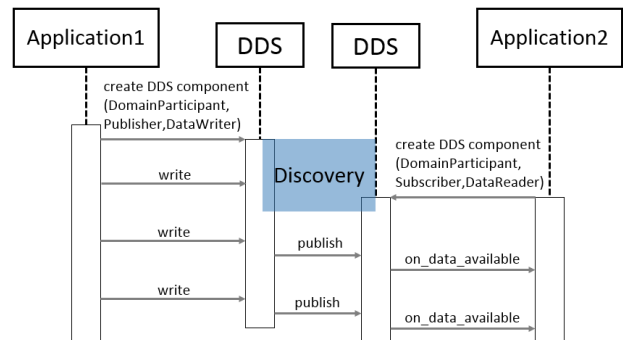


図 3 DDS の Pub/Sub 通信例

3. DDS(Data Distribution Service)

3.1 概要

QoS 対応のデータ中心ミドルウェアプラットフォームの需要に応える形で 2004 年に OMG (Object Management Group) が DDS の標準仕様を策定した。DDS の標準仕様は、コンポーネント・インターフェース、QoS ポリシー、およびプロトコルを定義する [4]。同時に、ソフトウェアベンダーに対して、製品のパフォーマンスに影響を与えるさまざまな実装オプションを採用する自由度を意図的に残している。その結果、基本的な仕様は統一されているが、実装によって機能や性能が異なる。

DDS 実装は複雑なコンピュータネットワークに関連したプログラミングを単純化する通信ミドルウェアであり、ノード間の送受信を Pub/Sub 通信モデルで実装している。DDS の通信例を図 3 に示す。DDS では構成要素を作成した時点で、ミドルウェアが自動で探索の通信を行い、マッチングを行う。その後、通信するデータの種類の情報を持つ Topic を中心に Pub/Sub 通信が行われる。

また、DDS は Pub/Sub 通信以外にも、メッセージのアドレッシング、データのマーシャリング、配布、フロー制御、再送などをサポートする。特に重要な利点は、DDS を通信に利用するアプリケーション群は完全に切り離されているという点である。アプリケーション間の相互作用についての考慮や、別のアプリケーションの情報を必要とすることはなく、場所も存在するか否かさえも知る必要がない。

3.2 QoS

QoS ポリシーは、データ可用性、配信およびタイミングを含む、Pub/Sub 通信に関する非機能的特性に影響する。QoS ポリシーは、通信構成要素である DomainParticipant, Publisher, DataWriter, Subscriber, DataReader, Topic それぞれに対して設定する。Pub/Sub 通信を行う要素全体において QoS 設定の整合性を確認できた場合のみ、通

信が成立する。

QoSの内容は、DDSの仕様書にて30項目ほど定められているが、そのほとんどが詳細まで決められておらず、実装に自由度を持たせている[4]。また、QoSの項目自体についても自由度があり、仕様に対して違反がなければ各実装独自のQoSを実装することもできる。今回行った測定に関係するQoSについていくつか説明する[5]。

RESOURCE LIMIT

DDSでは、同じデータ型のTopicに対して、種類の異なるデータごとにインスタンスが生成される。対象のインスタンスのサンプルが更新されると通信が行われる。このQoSポリシーは、DataWriterまたはDataReaderが扱うインスタンスやサンプルの数を指定する。具体的には、1つのDataWriterまたはDataReaderが扱える最大サンプル数(max_sample)、1つのDataWriterまたはDataReaderが扱える最大インスタンス数(max_instances)、1つのDataWriterまたはDataReaderが扱えるインスタンス1つあたりの最大サンプル数(max_samples_per_instance)を指定する。

HISTORY

サンプルの保持について制御する。Publisher側は、DataWriterによって維持されるべきサンプルを制御する。Subscriber側では、アプリケーションがサービスからデータ取得するまで維持する必要があるサンプルを制御する。KEEP_LASTを指定した場合、設定値depthの数だけサンプルを保持する。KEEP_ALLを指定した場合、全てのサンプルを保持する。ただし、RESOURCE LIMITSによりできる数は制限される。

RELIABILITY

通信の信頼性のレベルを指定する。指定するレベルはBEST EFFORTとRELIABLEの2つである。BEST EFFORTを指定した場合、ネットワーク障害によりデータが送信できなかったとしても、再送などは行わず、常に新しい値を送信する。RELIABLEを指定した場合、DataWriterでは、RESOURCE LIMITSで指定された制限を超過すると、最大で指定した時間だけ書き込み操作を待機させる。ネットワーク競合および障害にてロストしたメッセージはSubscriberがNACKメッセージを送信し、Publisherはロストしたシーケンス番号の更新を再送する。再送するために保持しておくサンプル数はHISTORY QoSに依存する。

4. 評価

4.1 評価目的

本論文では、3つの性能評価について述べる。それぞれの評価に対する目的を以下に示す。

表 1 機能比較

プロトコル	SOME/IP	DDS
主な対象	次世代車載システム	分散リアルタイムシステム
実装	vSomeIP	Connex DDS, OpenSplice, OpenDDS etc.
内部通信	ソケット通信	実装次第 (Connex DDS : 共有メモリ)
外部通信	UDP/TCP	UDP/TCP
QoS	なし	信頼性, 送信期限, 帯域幅, リソース資源 etc.
セキュリティ	なし	認証, アクセス制御, 暗号化 etc.

応答時間測定

車載システムにおいて通信ミドルウェアを導入することで、通信のたびにミドルウェア処理としてオーバーヘッドが発生する。制御関係の機能と通信する可能性もあるため、通信の応答性能を把握しておく必要がある。

ストレステスト

自動運転技術の実装により高いCPU負荷がかかる中、高いスループットが望まれる場面が増えてくる。そのような状況下での性能を把握しておく必要がある。

メモリ測定

この2つのミドルウェアを利用したアプリケーションは幅広いECUに導入される可能性がある。性能の低いECUに組み込まれる場合、リソースに対する要求は厳しく、メモリサイズは重要な項目である。

4.2 機能比較

ここではSOME/IPとDDSの機能を比較する。比較結果を表1にまとめる。大きな違いとして、まず内部通信の通信方式が挙げられる。SOME/IPの実装であるvSomeIP[6]はBoost.Asioライブラリを使ったunix domain sockets通信を行う。DDSは仕様で定められていないため実装によって異なるが、Connex DDS[7]では共有メモリが採用されている。次にサポートされる機能が大きく異なる。vSomeIPはQoSやセキュリティに関して定義されていないが、DDSは30種類以上のQoSと認証、アクセス制御、暗号化などセキュリティに関する仕様が策定されている[8]。

4.3 性能評価環境

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[6]を使用した。今回の評価で使用したバージョンは2.6.4である。DDSの実装として、Real-Time Innovations, Inc.が提供するConnex DDS[7]を使用した。今回の評価で使用したバージョンは5.2.3である。

表 2 評価環境

内部通信・外部通信送信用 PC	
機種名称	Let's note CF-F8
OS	Ubuntu 14.04 LTS
Linux Kernel Version	3.13.0-24-generic
CPU	Intel Core2 Duo P9400 2.40GHz
メモリ	4GB
LAN	CAT6 1Gbps
外部通信受信用 PC	
機種名称	ThinkPad X300
OS	Ubuntu 14.04 LTS
Linux Kernel Version	3.13.0-24-generic
CPU	Intel Core2 Duo L7100 1.20GHz
メモリ	2GB
LAN	CAT6 1Gbps

表 2 に性能評価に用いたコンピュータの情報を示す。ノートパソコン 2 台を用意し、そのうち 1 台を内部通信と外部通信の送信側として使用し、もう一台を外部通信の受信用として使用した。外部通信の測定は CAT6 の LAN で接続した。

4.4 応答時間測定

4.4.1 測定内容

この評価項目では、測定方法としては各実装においてアプリケーションを 2 つ用意し、Pub/Sub 通信が往復する時間を測定する。

通信の条件として、内部通信と外部通信を想定する。内部通信においては、vSomeIP は Boost.Asio の API を使ったソケット通信、ConnexDDDS は共有メモリで通信し、外部通信においてはどちらの実装も UDP と TCP で通信する。また比較対象として vSomeIP の内部通信で採用している Boost.Asio のソケット通信そのものの応答時間も測定した。メッセージ長を 64B から 2 倍ずつ長くしていき、512KB までのメッセージを 1 秒間隔で送信し、それを 1000 回繰り返すことで測定を行う。測定範囲を大きくしたのは、この 2 つの通信ミドルウェアは車載システムに採用を検討されている状況であり、ユースケースが不透明であるためである。ただし、vSomeIP の UDP 通信はペイロード長に 1400B の制限、ConnexDDDS の UDP 通信は 63KB の制限がある。

4.4.2 測定方法

今回の測定は汎用 OS 上で行っているため、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、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

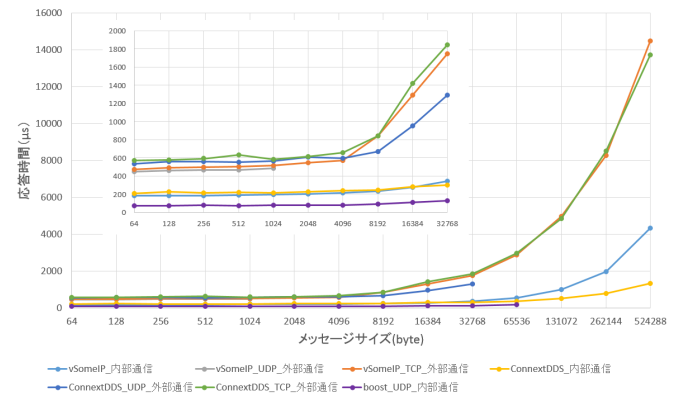


図 4 応答時間

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。

4.4.3 測定結果

測定結果を図 4 に示す。まず内部通信を比較すると、64B~16KB においてその差は小さいが vSomeIP のほうが早い。しかし、32KB 以降は ConnexDDDS のほうが高速であり、メッセージサイズが大きくなるにつれてその差は大きくなる。これはメッセージサイズが大きくなるにつれて共有メモリの通信速度の速さが影響していると考えられる。次に外部通信を比較する。UDP は仕様上、大きなメッセージサイズでの測定はできなかったが、1024B までの UDP 通信においては vSomeIP の方が ConnexDDDS より 100μs ほど高速である。同じ UDP 通信であることから、この差はミドルウェアの処理の差である。TCP 通信では、1024B までの UDP 通信と同様に vSomeIP の方が ConnexDDDS より 100μs ほど高速である。応答時間自体が 500μs であったので明らかな差であったが、メッセージサイズが大きくなるにつれて応答時間に対する時間差の割合は小さくなった。これは TCP 通信にかかる時間が大きいため、ミドルウェア実装としての性能差の影響が小さかったからであると考えられる。

4.5 ストレステスト

4.5.1 測定内容

この評価項目では、SOME/IP と DDS の CPU 負荷状況におけるスループットとパケットロス率を測定する。測定方法としては各実装において Publish アプリケーションと Subscribe アプリケーションの 2 つを用意する。内部通信では全体に、外部通信では送信側または受信側のみに CPU 負荷を 0, 50, 70, 90 % かけ、間隔なしに 100 万パケット送信する。メッセージ長を 64B から 2 倍ずつ長くしていき、64KB までのメッセージを送信する。受信側で時間を計測し、スループットを計測する。また受信できたメッセージの数をカウントし、パケットロス率を算出する。

vSomeIP の UDP 通信はペイロード長に 1400B の制限、ConnexDDDS の UDP 通信は 63KB の制限がある。内部通

表 3 QoS のパラメータ設定

パラメータ	設定
max_sample	100
max_sample_per_instance	100
HISTORY	KEEP_ALL
RELIABILITY	RELIABLE

信においては、vSomeIP は Boost.Asio の API を使ったソケット通信、ConnexDDDS は共有メモリで通信し、外部通信においてはどちらの実装も UDP と TCP で通信する。

4.5.2 測定方法

ストレステストでは、プロセス優先度設定も実行コアの固定も行わない。理由としてはプロセス優先度の変更やコアの固定を行った場合、高負荷の状態でもマルチスレッド実装がうまく動作せず、クラッシュしたからである。

次にストレステストに関する QoS と設定値について説明する。各 QoS 内容の説明は 3.2 で行った。RESOURCE LIMIT と HISTORY, RELIABILITY QoS に関して、測定で使った値を表 3 に示す。

4.5.3 測定結果

スループット

内部通信におけるスループットを図 5 に、UDP 通信におけるスループットを図 6 に、TCP 通信におけるスループットを図 7 を示す。

まず内部通信について述べる。ConnexDDDS ではメッセージサイズが大きくなるにつれて、スループットが向上しているのが分かる。それに対して vSomeIP はメッセージサイズが大きくなってもスループットはあまり伸びていない。これは ConnexDDDS が通信方式に共有メモリを採用しており、その性能差であると考えられる。

次に UDP 通信について述べる。今回使用したイーサネットケーブルは回線速度が 1 Gbps である。ConnexDDDS はメッセージサイズが大きくなるにつれてスループットを伸ばし、8192B では回線速度の限界で頭打ちになっているのが分かる。これより、ConnexDDDS は効率よく帯域幅を消費できていると言える。vSomeIP は仕様により 1024B までしか測定できていないが、ConnexDDDS と比べ CPU 負荷の影響を受けやすいことが分かる。これは ConnexDDDS の RELIABLE により RESOURCE LIMIT で指定したサンプル数を超える過剰な書き込みがあった場合、書き込みがブロックされ、データ送信周期が調整されているのに対して、vSomeIP は間隔なくメッセージを送信しているため、CPU 負荷による影響が大きくなると考える。

最後に TCP 通信について述べる。ConnexDDDS では、UDP 通信の結果と同様に 8192B では回線速度の限界により頭打ちになっており、帯域幅を有効に利用できている。それに対して vSomeIP では、メッセージサイズが大きくなっても、スループットが伸びていない。ConnexDDDS は UDP 通信の結果で述べたようにデータ送信周期が調整され

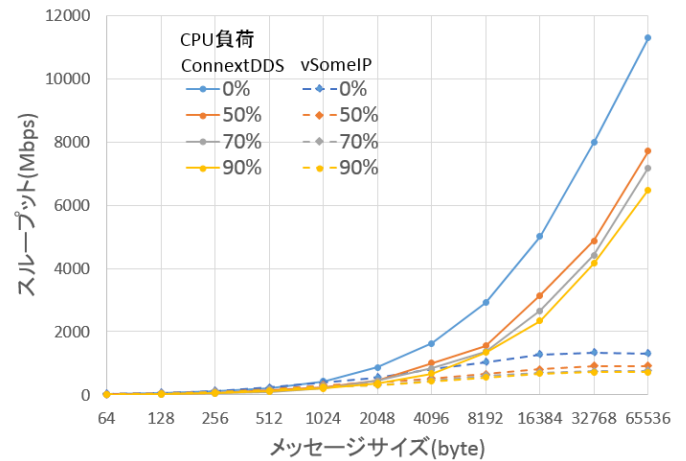


図 5 内部通信におけるスループット

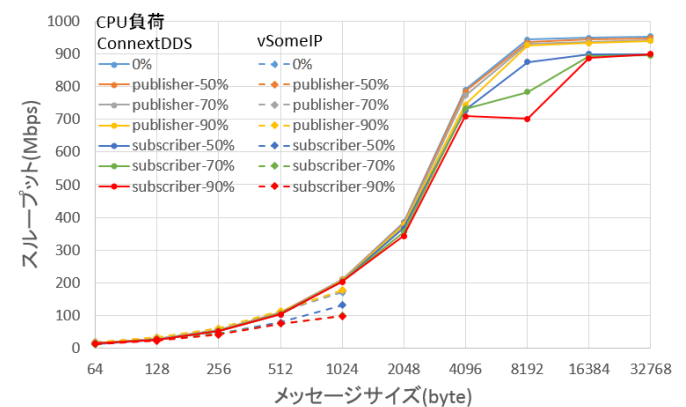


図 6 UDP 通信におけるスループット

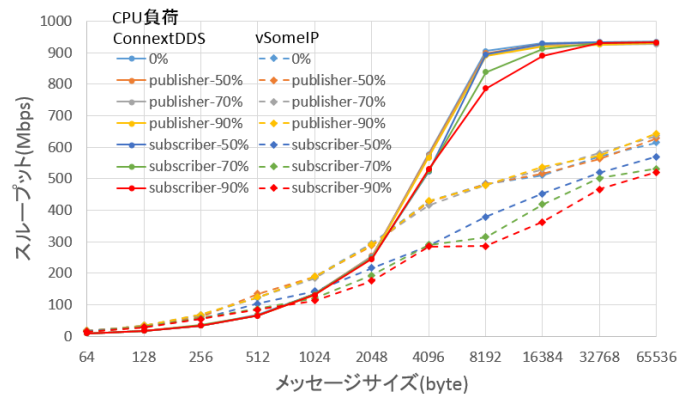


図 7 TCP 通信におけるスループット

るため、効率のよい通信ができています。一方で、メッセージサイズが小さい場合、vSomeIP よりスループットが劣る結果となった。これも同様に過剰な書き込みに対してブロックする機能が原因である。過剰と判断する材料としては、送信完了が確認できていない書き込みデータの数である。よって帯域幅や処理能力に余裕があったとしても書き込みがブロックされることがある。これによりデータの送信量が減少し、スループットが vSomeIP より劣ったと考えられる。

表 4 ストレステストにおけるパケットロス率

実装	CPU負荷	パケットロス率[%]											
		メッセージサイズ(バイト)											
		64	128	256	512	1024	2048	4096	8192	16384	32768	65536	
内部通信	vSomeIP	0%	0	0	0	0	0	0	0	0	0	0	0
		50%	0	0	0	0	0	0	0	0	0	0	0
		70%	0	0	0	0	0	0	0	0	0	0	0
		90%	0	0	0	0	0	0	0	0	0	0	0
	ConnnextDDS	0%	0	0	0	0	0	0	0	0	0	0	0.35
		50%	0	0	0	0	0	0	0	0	0	0	0.77
		70%	0	0	0	0	0	0	0	0	0	0	0.96
		90%	0	0	0	0	0	0	0	0	0	0	0.2
	UDP通信	0%	47.92	53.13	50.69	50.77	58.10	-	-	-	-	-	-
		Pub 50%	34.66	39.47	38.6	34.82	47.55	-	-	-	-	-	-
		Pub 70%	28.43	36.21	39.3	31.66	35.06	-	-	-	-	-	-
		Pub 90%	27.19	34.61	34.81	24.18	34.56	-	-	-	-	-	-
		Sub 50%	57.89	60.82	65.64	65.37	67.59	-	-	-	-	-	-
		Sub 70%	65.47	63.28	66.69	66.92	75.9	-	-	-	-	-	-
		Sub 90%	62.81	66.21	68.07	67.04	76.36	-	-	-	-	-	-
		0%	0	0	0	0	0	0	0.0032	0.0025	0.0019	-	-
UDP通信	vSomeIP	50%	0	0	0	0	0	0	0.0068	0.028	0.035	-	-
		70%	0	0	0	0	0	0	0.0073	0.017	0.016	-	-
		90%	0	0	0	0	0	0	0.0036	0.025	0.014	-	-
		Sub 50%	0	0	0	0	0	0	2.06	1.02	1.9	-	-
	ConnnextDDS	Sub 70%	0	0	0	0	0	0	5.17	1.37	2.16	-	-
		Sub 90%	0	0	0	0	0	0	8.45	1.71	2.16	-	-
		0%	0	0	0	0	0	0	0.0032	0.0025	0.0019	-	-
		50%	0	0	0	0	0	0	0.0068	0.028	0.035	-	-
		70%	0	0	0	0	0	0	0.0073	0.017	0.016	-	-
		90%	0	0	0	0	0	0	0.0036	0.025	0.014	-	-
		Sub 50%	0	0	0	0	0	0	2.06	1.02	1.9	-	-
		Sub 70%	0	0	0	0	0	0	5.17	1.37	2.16	-	-
		Sub 90%	0	0	0	0	0	0	8.45	1.71	2.16	-	-

パケットロス率

パケットロス率の結果について表 4 に示す。TCP については再送制御によりパケットロスが発生しないため省略する。内部通信においてはどちらもパケットロス率がほとんどない。UDP 通信においては、vSomeIP では非常に多くのパケットがロスしているのに対して、ConnnextDDS は 10% 以下の少ないパケットロス率を実現している。これは QoS 機能である RELIABILITY が RELIABLE に設定されているからである。RELIABLE が設定された場合、過剰な書き込みをブロックし、送信周期が調整される。また、届かなかったメッセージは HISTORY QoS によりデータが残っていた場合、再送される。これらにより、高い送信成功率が実現されている。

4.6 メモリ測定

各実装において 1 秒間隔で 1 バイトのメッセージを通信するアプリケーションを用意し、その静的メモリサイズと実行時の動的メモリサイズを測定した。結果を表 5 に示す。

ROM/RAM のサイズを比較すると ConnnextDDS のほうが vSomeIP と比べ 50 倍以上大きいことがわかった。また実行時の動的メモリサイズを比較しても ConnnextDDS のほうが 4 倍ほど多くメモリを消費している。ConnnextDDS は QoS 機能など多くの機能があるゆえにプログラムサイズが大きく、またリソースも多く消費することが分かった。

以上のことから、リソースの乏しい ECU においては、メモリ消費量の観点で vSomeIP が適すると言える。

4.7 考察

応答時間測定では、メッセージサイズが小さい場合は、ミドルウェアが軽量であることから応答時間が短く、vSomeIP が優位であることが分かった。またストレステストでは、ConnnextDDS の QoS 機能により高い CPU 負荷において、メッセージサイズが大きくなっても高いスループットと信頼性があることが分かった。これらの結果より、性能面の

表 5 消費メモリ

ミドルウェア	ノード	ROM[KB]	RAM[KB]	動的メモリ [KB]
vSomeIP	送信側	132.080	1.792	3464
vSomeIP	受信側	111.532	1.728	3220
ConnnextDDS	送信側	8817.923	66.368	12564
ConnnextDDS	受信側	8821.368	66.976	12820

観点では、SOME/IP は制御関係などの通信量の小さいシステムに適している言え、DDS は自動運転関連の高度な処理による高い CPU 負荷と大きなデータ通信が伴うシステムに適していると考えられる。また、メモリ測定より vSomeIP はメモリ消費量が低く、リソースの乏しい ECU においても適用可能である。リソースの管理や通信設計などの調整は開発者に委ねられるが幅広い環境で適用可能であると考えられる。一方、DDS はパフォーマンスの高いマシンであれば QoS 機能などによりリソース管理や通信設計などの開発者の負担を軽減できると考える。

5. おわりに

本論文では、機能比較と 3 つの性能評価を行った。これらより、SOME/IP はリソース管理や通信設計などの調整が開発者に委ねられるが、幅広い環境において適用可能である。一方、DDS は性能の高いマシンであれば、開発者の負担なく効率的な通信を実現できることがわかった。

今後の課題としては、通信相手が複数に増加した場合の各評価の変化と DDS の QoS パラメータ設定における性能の変化などが挙げられる。

参考文献

- [1] 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.
- [2] AUTOSAR: SOME/IP Protocol Specification. https://www.autosar.org/fileadmin/user_upload/standards/foundation/1-3/AUTOSAR_PRS_SOMEIPProtocol.pdf, (参照 2018.02.05).
- [3] Kirsten Matheus and Thomas Knigseder: *Automotive Ethernet*. pp.159-166, 2014.
- [4] Object Management Group: Data Distribution Service (DDS) Version 1.4. 2015-03. <http://www.omg.org/spec/DDS/1.4/PDF>, (参照 2018-02-05).
- [5] Real-Time Innovations: RTI ConnnextDDS CoreLibraries User's Manual. https://community.rti.com/static/documentation/connnext-dds/5.3.0/doc/manuals/connnext-dds/RTI_ConnnextDDS_CoreLibraries_UsersManual.pdf, (参照 2017-2).
- [6] BMW AG: GENIVI/vsomeip: An implementation of Scalable service-Oriented MiddlewarE over IP. <https://github.com/GENIVI/vsomeip>, (参照 2018-02-05).
- [7] Real-Time Innovations: Develop and Intergrate Systems with Connnext DDS Professional — RTI Products. <https://www.rti.com/products/dds>, (参照 2017-2).
- [8] Object Management Group: About the DDS Security Specification Version 1.1 beta. <http://www.omg.org/spec/DDS-SECURITY/1.1/Beta1/>, (参照 2017-2).