空間データを対象とした分散MQTTシステムGammaの提案と実装

上田 高寛[1,a)]　片山 徹郎[1,b)]

概要：近年，スマートフォンの処理能力の向上やカメラなどのセンサーの性能向上，画像処理技術の向上により，スマートフォンを使って，物体の種類や位置などの空間データをリアルタイムに取得することが可能になった．空間データをより有効に活用するためには，必要な空間データのみを多くのアプリケーション間でリアルタイムに共有することが必要である．そこで本論文では，空間データを対象とした分散MQTTシステムであるGammaを提案し，実装する．実験では，Gammaは単一のMQTTブローカよりも多くのメッセージを処理できることがわかった．また，GammaはGatewayと分散MQTTブローカの数を増やすことでスケーラビリティが確保できることを確認した．

キーワード：ネットワークミドルウェア， クライアントサーバアーキテクチャ，MQTT

Proposal and Implementation of Gamma Which Is a Distributed MQTT System for Spatial Data

Abstract: In recent years, with improvements in the processing ability of smartphones, the performance of cameras and other sensors, and improvements in image processing technology, it has become possible to use smartphones to acquire spatial data such as the type and location of objects in real time. In order to use more effectively of spatial data, it is necessary to share only the necessary spatial data among many applications in real time. Therefore, this paper proposes and implements Gamma, which is a new distributed MQTT system, to improve the usefulness of distributed MQTT systems for sharing spatial data. In the experiment, it has been found that Gamma can process more messages than a single MQTT broker. It is confirmed that Gamma achieves scalability by increasing the number of Gateways and distributed MQTT brokers.

1. はじめに

空間データとは「モノの位置情報と意味情報 (状態，形状，サイズなど) の双方を要素として持つデータ」であり，具体的には「自転車がA地点を30km/hで走行している」のようなデータを指す [1]．近年，スマートフォンの処理能力の向上やカメラなどのセンサの性能向上，画像処理技術の向上に伴い，スマートフォンのカメラに映った物体の種類や位置のような空間データをリアルタイムに取得することが可能になった [2]．空間データをより有効に活用するためには，多くのアプリケーション同士で必要な空間データのみをリアルタイムに共有する必要がある．MQTT(Message Queuing Telemetry Transport)[3] は，IoT(Internet of Things) のアプリケーション層の通信プロトコルとして知られている．基本的なMQTTは1つのシステムに1つのメッセージを仲介するブローカを定義しているため，メッセージを送信するPublisherやメッセージを受け取るSubscriberといったクライアントの数は限られる．そこで，複数のブローカ間で連携し，負荷分散を行う分散MQTTの実装がある．空間データを対象とした分散MQTTの先行研究としては，Ryo Kawaguchiらの研究 [4] がある．この研究では，空間データを扱うことに適したトピック構造を導入している．また，ブローカの負荷を軽減し，異なる種類のブローカをサポートするために分散MQTTブローカとGatewayを導入している．数値計算を用いて，特にSubscriberの数が多い場合やSubscribeの変更が頻繁に発生する場合に，既存の分散MQTTシス

[1] 宮崎大学
University of Miyazaki
[a)] uedat@earth.cs.miyazaki-u.ac.jp
[b)] kat@cs.miyazaki-u.ac.jp

テムよりも，各ブローカが単位時間あたりに受信するメッセージ数が少ないことを確認している．しかし，この分散 MQTT システムを実際に利用することを想定した場合，以下の 2 つの課題がある．

- 事前に分散 MQTT ブローカの IP アドレスとポート番号を，Gateway に登録する必要があるため，システム稼働中に分散 MQTT ブローカの構成を変更できない点
- Gateway に担当トピックの情報が設定されていないため，分散 MQTT ブローカと Gateway 間のメッセージの共有を効率よく行えない点

そこで本論文では，空間データを対象とした分散 MQTT システムの有用性の向上を目的として，分散 MQTT システム Gamma の提案と実装を行う．具体的には，上記 2 つの課題を解決するために，先行研究 [4] に Manager を追加した新たなシステムを提案し，5 つの機能を実装する．

2. Gamma の機能

本章では，Gamma とそのクライアントライブラリの機能について説明する．本研究で実装する Gamma は，システムの負荷が大きくなった際に，分散 MQTT ブローカや Gateway を追加できる．これにより，より多くのメッセージの処理にも対応できる．以降，本研究で実装した 5 つの機能について説明する．

- システム稼働中に分散 MQTT ブローカを追加する機能
  システム管理者が，システム稼働中に分散 MQTT ブローカを追加する機能である．この機能は，Gamma で新たに追加する Manager が主に担う．また，Manager は，分散 MQTT ブローカを新たに追加した際に，Gateway が持つ分散 MQTT ブローカの接続情報の更新処理を行う．
- システム稼働中に Gateway を追加する機能
  システム管理者が，システム稼働中に Gateway を追加する機能である．この機能は，Gamma で新たに追加する Manager が主に担う．Manager はシステム起動後に，新たに追加した Gateway から当該 Gateway の接続情報を受け取り管理する．また，新たに追加した Gateway に，分散 MQTT ブローカの接続情報を通知する役割もある．
- Gateway の担当トピックを設定する機能
  システム管理者が，Gateway の担当トピックを設定する機能である．この機能は，Gamma で新たに追加する Manager が主に担う．Manager は Gateway の担当トピックの設定リクエストを，システム管理者から受け付け，設定する．また，Gateway の担当トピックに関する情報を，クライアントに伝える機能がある．1 つの Gateway に複数の担当トピックを設定することができる．

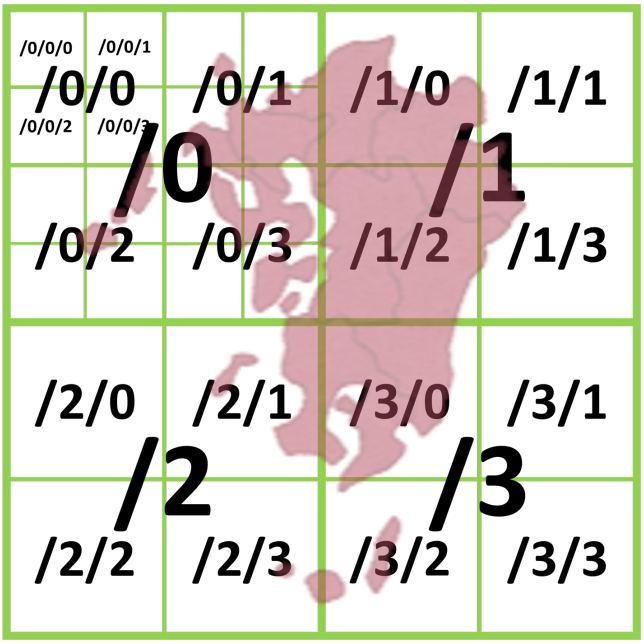

図 1　トピックと位置情報の対応例

- 位置情報に基づいて接続する Gateway を選択する機能
  クライアントが Publish/Subscribe する空間データの位置情報を基に，接続する Gateway を選択する機能である．この機能は，Gamma で新たに追加する Manager とクライアントライブラリが主に担う．クライアントは，Gateway へ接続する前に Manager に接続し，Gateway への接続情報や各 Gateway の担当トピックに関する情報を取得する．したがって，クライアントは，Manager への接続情報を知っているだけでよい．また，クライアントは Manager から取得した各 Gateway の担当トピックと，クライアント自身が Publish/Subscribe するトピックを比較することによって，最適な Gateway を選択し接続できる．
- 位置情報に基づいて空間データを Publish/Subscribe する機能
  クライアントが位置情報を基に，空間データの Publish/Subscribe を行う機能である．この機能は，Gamma のクライアントライブラリが主に担う．クライアントは，空間データの緯度と経度から Publish するためのトピックを算出し，空間データを Publish することができる．また，与えられた緯度と経度と半径からその範囲の空間データを Subscribe するためのトピックを算出し，Subscribe することができる．これらの計算には S2 Geometry[5] を使用している．

3. 実装

Gamma は評価ツールやクライアントライブラリを含め

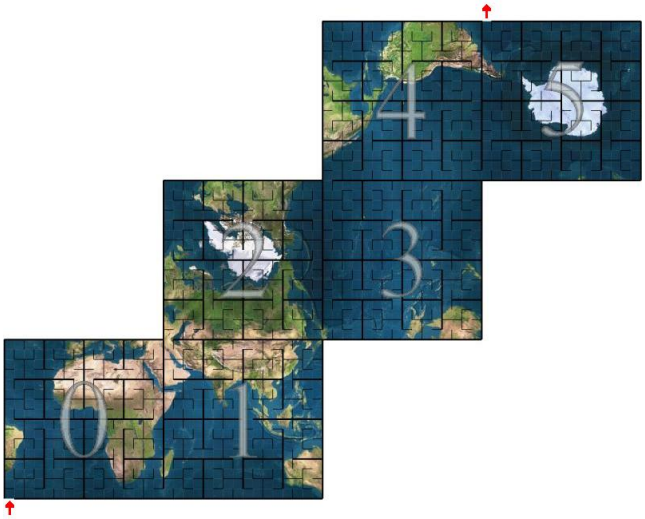

図 2　S2 Geometry における 6 つの平面と地球表面との対応 (S2 Cells[9] より)

Go 言語 [6] を用いて実装した．また本論文では MQTT ブローカに Mosquitto[7] を利用している．MQTT のクライアントライブラリには Paho[8] を利用している．

本章では，空間データを位置情報に基づいて Publish したり，指定した範囲の空間データのみを Subscribe するといった機能を持つクライアントの実装に着目し解説する．

3.1　位置情報からトピックを算出する際の処理

Gamma や先行研究 [4] では，空間データを Publish/Subscribe する際に使用するトピックを空間データに紐づいた位置情報に基づいて決定する．先行研究 [4] では，**図 1** のようにシステムが扱う領域を Z の書き順で 4 つに再帰的に分割し番号を割り当てることでトピックを決定していた．Gamma でもこの方針は変わらない．しかし，実際に地球上で使用することを考慮すると，地球の形状を考慮する必要がある．そこで，Gamma では地球上の位置情報 (緯度と経度) からトピックを決定するために S2 Geometry の S2 Cells[9] を使用している．S2 Cells では，立方体の 6 つの面を単位球面上に投影し，それぞれの面について再帰的に 4 つに分割し Cell ID という番号を割り当てることができる．これにより，地球上の空間データを Publish/Subscribe する際に使用するトピックを，先行研究と同様の構造のまま決定することが出来る．**図 2** は，S2 Geometry における 6 つの平面と地球表面との対応を示したものである．

また，S2 Geometry では中心点と半径を指定することでその範囲を表す S2 Cells を算出することが出来る．この機能を利用することで，Subscribe する際に必要な範囲の空間データのみを Subscribe することが可能になる．

3.2　クライアントオブジェクト生成時の処理

クライアントオブジェクトは，Gateway に接続し，空間データを Publish/Subscribe する．クライアントオブジェクトを生成する際の処理を，以下に示す．

なお，この処理を呼び出す際，Manager の MQTT ブローカへの接続情報と現在地の緯度経度，空間データを取得する際の範囲，接続処理の際にタイムアウトするまでの時間を，指定する必要がある．

(1) クライアントが Manager の MQTT ブローカへ接続する
(2) クライアントが「/api/gateway/info/all」トピックを Subscirbe し，Gateway の MQTT ブローカに接続するためのメッセージを受信する
(3) クライアントがメッセージを受信した後，Manager の MQTT ブローカとの接続を終了する
(4) クライアントが受信したメッセージをパースする．以降，このメッセージを Gateway 接続情報リストと記述する
(5) クライアントが Gateway 接続情報リストをランダムに並べ替える
これは，ランダムに並べ替えなかった場合，同じ担当トピックの Gateway が複数存在する場合でも，1 つの Gateway に接続が集中してしまうためである
(6) クライアントがランダムに並べ替えた Gateway 接続情報リストを担当トピックの長さを基に，降順でソートする
(7) 現在地から現在地の情報を Publish 又は Subscribe する際に必要となるトピックを生成する (以下，生成トピックと記述する)
(8) 「接続先情報」変数を作成する
(9) 各 Gateway 接続情報リストの要素に対して，以下の処理を行う
  (a) 「接続先情報」変数に取り出した接続リストの要素を代入する
  (b) 取り出した接続情報リストの要素の担当トピックと，生成トピックの先頭部分が一致した場合，処理を終了する
(10) ランダムに並べ替え，担当トピックの長さの降順にソートした Gateway 接続情報リストの担当トピックと生成トピックの先頭部分が一致するかどうかを確認する
(11) 一致を確認した場合，当該 Gateway ブローカへ接続する． 一致するものがなかった場合，接続情報リストの末尾の接続情報をもとに Gateway へ接続する
(12) クライアントオブジェクトを生成し，MQTT の接続オブジェクトや空間データを取得する際の範囲の値をクライアントオブジェクトに設定する

表 1　特定の範囲を表す Cell ID 文字列生成の際のパラメータの例

緯度	31.828061073646502
経度	131.4153025207035
半径 [km]	1.0
最大 Cell 数	4
最大 Level	20

表 2　特定の範囲を表す Cell ID 文字列の例

Cell ID 文字列
1/222130121002
1/222130121003
1/222130121010111
1/222130121013

表 3　特定の範囲の空間データを取得する際に Subscribe するトピックの例

トピック
/1/2/2/2/1/3/0/1/2/1/0/0/2/#
/1/2/2/2/1/3/0/1/2/1/0/0/3/#
/1/2/2/2/1/3/0/1/2/1/0/1/0/1/1/1/#
/1/2/2/2/1/3/0/1/2/1/0/1/3/#

3.3　空間データを Publish する際の処理

空間データを Publish する際の処理を，以下に示す．

(1) 空間データを取得する
(2) 空間データの位置情報から Publish するトピックを算出する
(3) 算出したトピックの先頭に文字列「/forward」を付加する
(4) 空間データを算出したトピックへ Publish する

3.4　指定した範囲の空間データを Subscribe する際の処理

指定した範囲の空間データを Subscribe する際の処理を以下に示す．なお，位置情報からトピックを算出する際に，S2 Geometry [5] を使用する．ここで，最大 Cell 数とは，S2 Geometry において任意の範囲を表現する際に使用する Cell ID の最大数である．また，最大 Level とは，Cell ID の Level(長さ) を表す．この値が大きいほど，Cell ID 文字列も長くなり，狭い範囲の表現が可能になる．**表 1** に，特定の範囲を表す Cell ID 文字列生成の際のパラメータの例を示す．緯度と経度は，宮崎大学工学部の値を用いている．**表 2** に，実際に生成した特定の範囲を表す Cell ID 文字列の例を示す．**表 3** に，特定の範囲の空間データを取得する際に Subscribe するトピックの例を示す．

(1) 表 1 に示すように，緯度と経度，半径，最大 Cell 数，最大 Level を指定し，S2 Geometry の指定した範囲の Cell ID オブジェクトを列挙する機能を呼び出し，表 2 に示すように指定した範囲を表す Cell ID 文字列に変換する

図 3　実際の Cell ID が示す範囲 (S2 Demo[10] を用いて可視化)

(2) 列挙した Cell ID 文字列をトピックに変換する．この際，表 3 に示すようにワイルドカードトピックにする
(3) 変換したトピックと，自身の持つ「Subscribe トピックリスト」変数のトピックを比較して，Subscribe していないトピックを Subscribe する
(4) 自身の持つ「Subscribe トピックリスト」変数のトピックから，変換したトピックに含まれていないトピックを Unsubscribe する
(5) 自身の持つ「Subscribe トピックリスト」から Unsubscribe したトピックを削除する
(6) 自身の持つ「Subscribe トピックリスト」に新たに Subscribe したトピックを追加する

図 3 は，表 2 の実際の Cell ID が示す範囲を可視化したものである．

4.　Gamma の適用例

Gamma を稼働したまま，新たな分散 MQTT ブローカを追加できることを確認する．本章で使用するシステムの構成を，**図 4** に示す．水色で示した部分が，動的に追加する分散 MQTT ブローカである．具体的には，Gamma を稼働したまま，新たな分散 MQTT ブローカの追加リクエストを Manager へ送信し，分散 MQTT ブローカを追加できることを，分散 MQTT ブローカを追加する前後の画面を比較することで確認する．

分散 MQTT ブローカ 01 の追加リクエストを Manager へ送信する前後の画面の比較を，**図 5** に示す．図 5 の画面 1 は，既存の分散 MQTT ブローカ 00 に Publish されたメッセージとそのトピックを確認するための画面である．

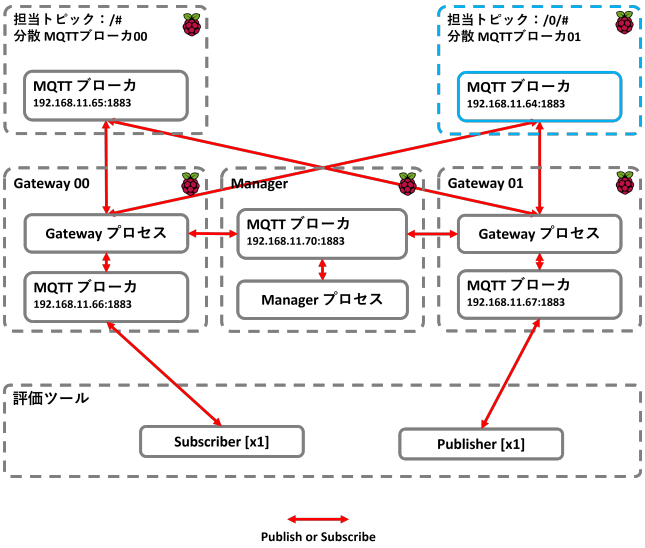

図 4　分散 MQTT ブローカを追加する際に使用するシステムの構成
(追加する分散 MQTT ブローカを水色で表示)

図 5 の画面 2 は，新たに追加する分散 MQTT ブローカ 01 に Publish されたメッセージとそのトピックを確認するための画面である．分散 MQTT ブローカ 01 を追加前は，トピックの先頭が「/0」で始まるメッセージを，分散 MQTT ブローカ 00 に転送していることが，図 5 から確認できる．また，担当トピックが「/0」である分散 MQTT ブローカ 01 を追加後は，トピックの先頭が「/0」で始まるメッセージを，分散 MQTT ブローカ 01 に転送していることが，図 5 から確認できる．よって，システムの稼働中に分散 MQTT ブローカを追加する機能は，正しく動作したといえる．

なお，今回実装した他の 4 つの機能についても，正しく動作することを確認した．

5. Gamma の評価

本研究で実装した Gamma の有用性を評価する．まず，単体の MQTT ブローカとの比較を行う．次に Gateway の数を増やすことによって得られるスケーラビリティを評価するための実験を行う．同時に，Gateway に担当トピックを設定することによる効果を評価するための実験も行い，それぞれ考察を行う．なお，以下に示す実験は，送信メッセージ長を 150 文字，Publisher の数を 100 個，Subscriber の数を 100 個，各 Publisher の Publish 間隔を 100 ミリ秒，測定時間を 100 秒という条件で行った．また，Publisher と Subscriber を 4 つのエリアに等しく配置した．Manager や分散 MQTT ブローカ，Gateway は全て Raspberry Pi[11] 上で，Publisher や Subscriber といったクライアントは全て同一の PC 上で動作させた．評価の際，クライアントを動作させている PC のタスクマネージャからネットワークの使用帯域や CPU 使用率を確認し，評価に使用する PC がボトルネックとならないことを確認した．

5.1 単体の MQTT ブローカと Gamma の比較に関する実験と考察

単体の MQTT ブローカと Gamma の比較を行う．評価に利用した単体 MQTT ブローカのシステム構成を**図 6** に，Gamma のシステム構成を**図 7** に示す．また，実験の結果を**表 4** に示す．表 4 より，Gamma のほうが平均遅延時間が小さいことがわかる．このことから，本論文で実装した Gamma は，単体 MQTT ブローカと比べて多くのメッセージを処理できると言える．

5.2 分散 MQTT ブローカの数を増やすことによって得られるスケーラビリティに関する実験と考察

分散 MQTT ブローカの数を変化させて，スケーラビリティの確認を行う．Gateway の数は 4 つに固定する．また，Gateway に担当トピックを設定した場合と，設定しない場合の平均遅延時間の違いも比較する．Gateway に担当トピックを設定した際のシステムの構成例は，**図 8** に示した通りである．また，Gateway に担当トピックを設定しなかった場合のシステムの構成例は，**図 9** に示した通りである．なお，実験の際に分散 MQTT ブローカの CPU 使用率の上限を，cputool というコマンドラインツール [12] を用いて 10%に設定した．本研究のシステム全体で見た際の負荷の割合は，分散 MQTT ブローカの負荷よりも Gateway が占める負荷の割合が高い．また，本研究で用意できた Gateway は 4 台だけである．しかし，分散 MQTT ブローカ 1 つのみに対して Gateway が 4 つの場合でさえ，分散 MQTT ブローカよりも先に Gateway の処理能力が不足するという問題が生じた．そこで，この実験では分散 MQTT ブローカの処理能力を cputool[12] を用いて意図的に下げ，分散 MQTT ブローカの数を増やすことによって得られるスケーラビリティの確認を行う．

実験結果を，**表 5** に示す．表 5 から，分散 MQTT ブローカの数に応じて平均遅延時間が小さくなることを確認できる．このことから，分散 MQTT ブローカの数を増やすことによって，システム全体の処理能力が向上することが分かった．よって，分散 MQTT ブローカの数を増やすことによって，スケーラビリティを確保していることを確認できた．また，表 5 から，「担当トピック設定有」の場合と，「担当トピック設定無」の場合を比較すると，Gateway の数に関わらず，「担当トピック設定有」のほうが平均遅延時間が小さいことがわかる．このことから，本論文で実装した Gamma の機能の 1 つである Gateway に担当トピックを設定することによって，システム全体の効率が良くなったといえる．

5.3 Gateway の数を増やすことによって得られるスケーラビリティに関する実験と考察

Gateway の数を変化させて，スケーラビリティの確認を

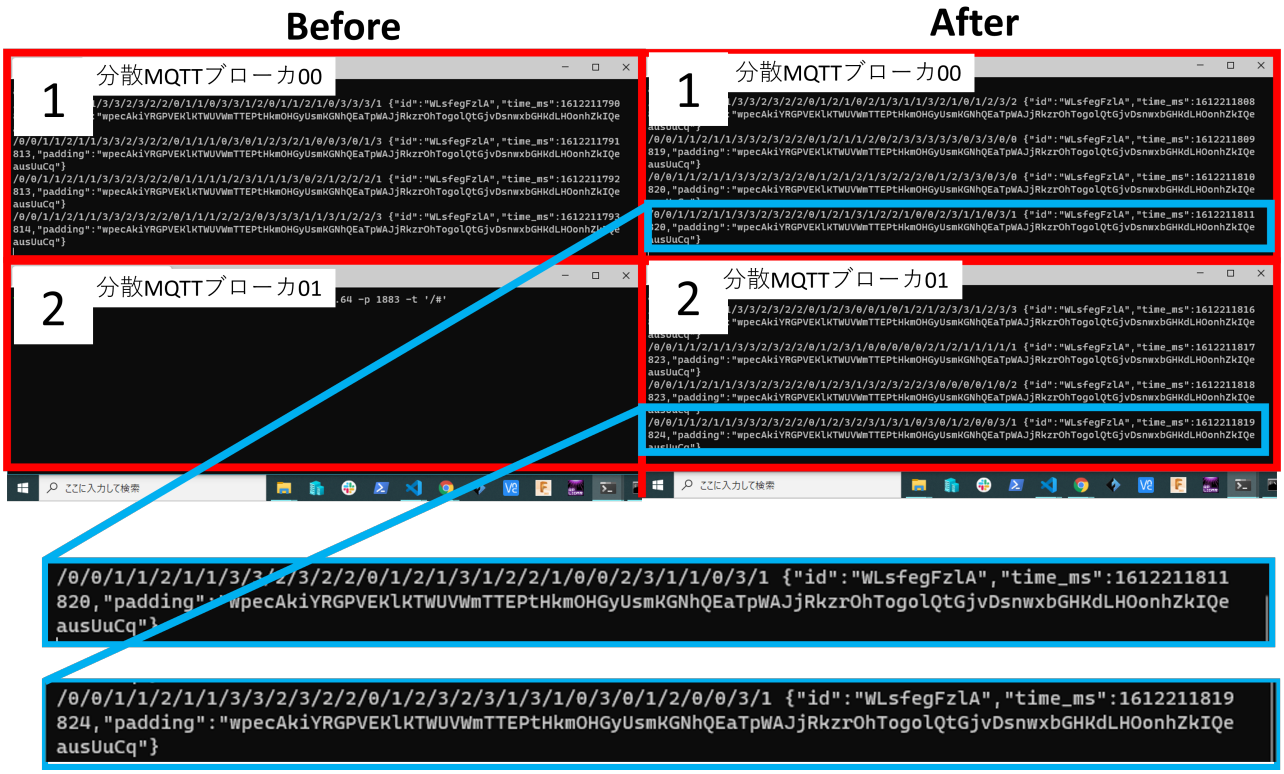

図 5　分散 MQTT ブローカ 01 を追加する前後の画面

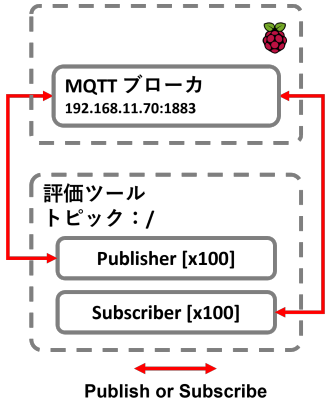

図 6　単体の MQTT ブローカと Gamma の比較における単体の MQTT ブローカのシステム構成

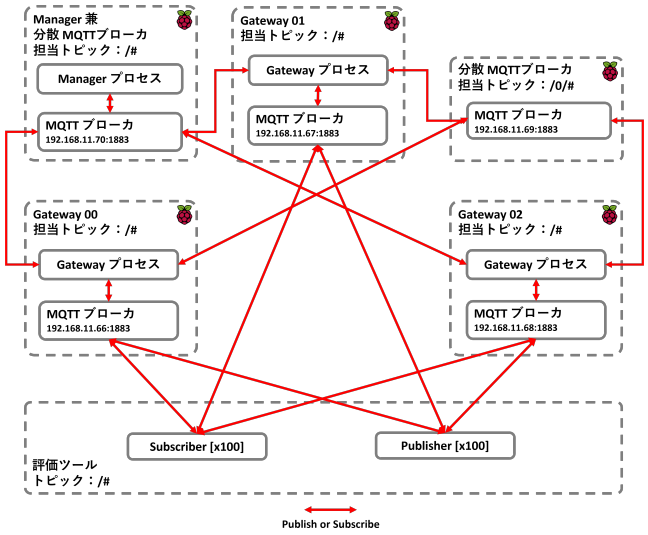

図 7　単体の MQTT ブローカと Gamma の比較における Gamma のシステム構成

行う．分散 MQTT ブローカの数は 1 つに固定する．また，Gateway に担当トピックを設定した場合と，設定しない場合の平均遅延時間の違いも比較する．Gateway に担当トピックを設定した際のシステムの構成例を，図 8 に示す．Gateway に担当トピックを設定しなかった場合のシステムの構成例を，図 9 に示す．Gateway における担当トピック設定有の際の担当トピックの設定例を**図 10** に示す．なお，Gamma では担当トピックは担当エリアと言い換えることもできる．担当トピックを複数設定できるため，Gateway が 2 つの際それぞれの Gateway で等しいサイズのエリアを担当することができる．

表 6 に，Gateway の数を変化させた際の実験結果を示す．表 6 から，Gateway の担当トピックの設定の有無にかかわらず，Gateway の数に応じて平均遅延時間が小さくなることを確認できる．すなわち，Gateway の数を増やすことによって，システム全体の処理能力が向上することが分かった．したがって，本研究で実装した Gamma は，Gateway の数を増やすことによって，スケーラビリティを

表 4　単体の MQTT ブローカと Gamma の比較

	単体の MQTT ブローカ	Gamma
平均遅延時間 [ms]	13871.2	14.6832

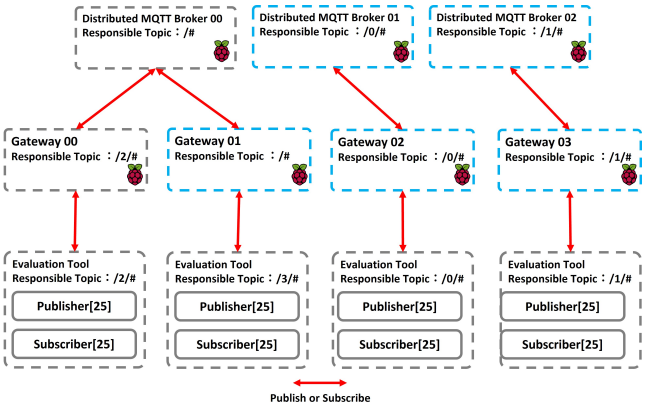

図 8　システム構成例 (Gateway の担当トピック設定有)

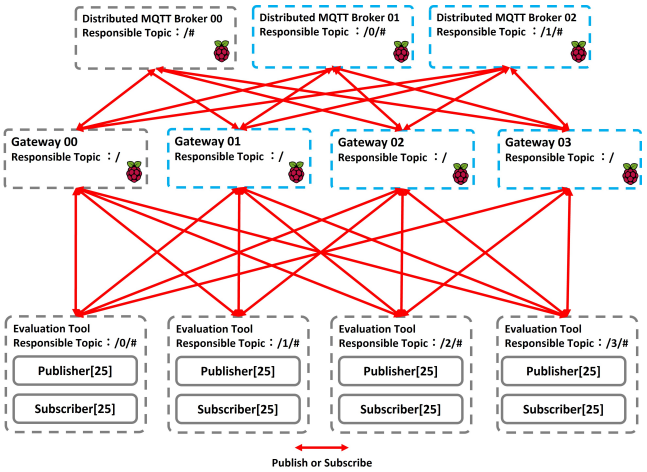

図 9　システム構成例 (Gateway の担当トピック設定無)

表 5　分散 MQTT ブローカの数を増やすことによって得られるスケーラビリティの確認 (平均遅延時間 [ms])

分散 MQTT ブローカの数	1	2	3
担当トピック設定有	294.35	127.05	46.75
担当トピック設定無	14925.22	1163.05	118.92

確保していることを確認できた．また，表 6 から，図 8 のように「担当トピック設定有」の場合と，図 9 の場合では，「担当トピック設定有」の場合の方が平均遅延時間が小さいことが分かる．このことから，本論文で実装した Gamma の機能の 1 つである Gateway に担当トピックを設定することによって，システム全体の効率が良くなったといえる．

6. まとめ

本論文では，空間データを対象とした分散 MQTT システムの有用性の向上を目的として，新たな分散 MQTT システム Gamma の提案と実装を行なった．空間データを対象とした分散 MQTT の先行研究としては，Ryo Kawaguchi らの研究 [4] がある．この，先行研究には，1 章で述べた 2 つの課題がある．そこで，この 2 つの課題を解決するために，本研究では，先行研究 [4] に Manager を追加した新たなシステムを提案し，5 つの機能を実装した．

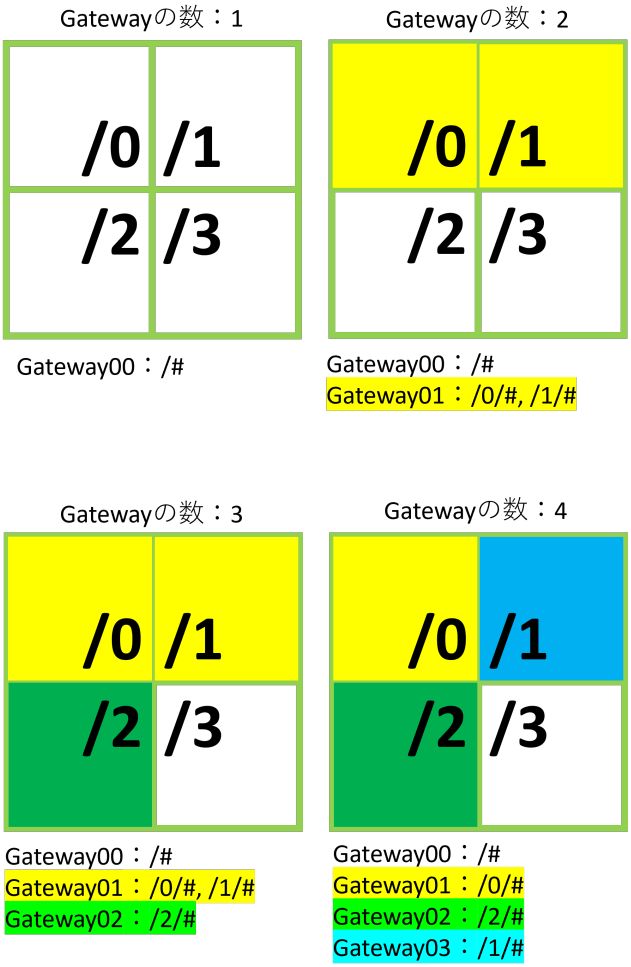

図 10　Gamma の Gateway の担当トピックの設定例

表 6　Gateway の数を増やすことによって得られるスケーラビリティの確認 (平均遅延時間 [ms])

Gateway の数	1	2	3	4
担当トピック設定有	11423.46	116.14	57.30	15.96
担当トピック設定無	11423.46	233.15	118.73	42.65

実装した Gamma の 5 つの機能が正常に機能することを，実際に動作させることで確認した．このことから，本研究で新たに提案したシステム Gamma は先行研究の，システム稼働中に分散 MQTT ブローカの構成を変更できないという課題の解決を確認した．また，実験で Gamma は，単体の MQTT ブローカと比べて多くのメッセージの処理が可能なことが分かった．Gateway や分散 MQTT ブローカの数を増やすことによってスケーラビリティを確保できることを確認した．更に，Gateway の担当トピックを複数設定できるようにし，特定の Gateway に負荷が偏らないように設定することによって，システム全体の効率が良くなることを確認した．このことから，Gateway に担当トピックの情報が設定されていないため，分散 MQTT ブローカと Gateway 間のメッセージの共有を効率よく行えないという課題の解決を確認した．

以上から，本研究で提案し実装した Gamma は，以下の 2 点を実現し，先行研究の 2 つの課題を解決した．

- システム稼働中に分散 MQTT ブローカを追加できる点
- 複数存在する Gateway の情報や Gateway の担当トピックをクライアントに通知できる点

以上のことから，本研究で提案し実装した Gamma は有用であることを確認できた．以下に，本研究における今後の課題を示す．

- 負荷に応じた Gateway や分散 MQTT ブローカの自動追加への対応
- Gateway や分散 MQTT ブローカの死活監視機能への対応
- 単一レベルワイルドカードへの対応
- Manager の冗長化

参考文献

[1] NTT データ：空間データ活用 〜モノと位置で見える未来〜，入手先 ⟨https://www.nttdata.com/jp/ja/data-insight/2020/033001/⟩（参照 2022-02-12）．
[2] ニューラルポケット株式会社：スマートくん | AI 搭載型スマホドライブレコーダー，入手先 ⟨https://smartkun.neuralpocket.com/⟩（参照 2022-02-12）．
[3] OASIS：*MQTT Version 3.1.1*, 入手先 ⟨http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html⟩（参照 2022-02-12）．
[4] Ryo Kawaguchi and Masaki Bandai: *A Distributed MQTT Broker System for Location-based IoT Applications*, 2019 IEEE International Conference on Consumer Electronics (ICCE), pp.1-4, 2019.
[5] S2 Geometry：*S2 Geometry*, 入手先 ⟨https://s2geometry.io/⟩（参照 2022-02-14）．
[6] Go：*The Go Programming Language* 入手先 ⟨https://golang.org/⟩（参照 2022-02-14）．
[7] Eclipse：*Mosquitto* 入手先 ⟨https://mosquitto.org/⟩（参照 2022-02-14）．
[8] The Eclipse Foundation：*Eclipse Paho* 入手先 ⟨https://www.eclipse.org/paho/⟩（参照 2022-02-14）．
[9] S2 Geometry：*S2 Cells*, 入手先 ⟨https://s2geometry.io/devguide/s2cell$_h ierarchy$⟩（参照 2022-02-14）．
[10] S2 Demo：*S2 Demo*, 入手先 ⟨https://s2.sidewalklabs.com/⟩（参照 2022-02-14）．
[11] Raspberry Pi Foundation：*Buy a Raspberry Pi - Raspberry Pi*, 入手先 ⟨https://www.raspberrypi.com/products/⟩（参照 2022-02-14）．
[12] Ubuntu Manpage：*cputool*, 入手先 ⟨http://manpages.ubuntu.com/manpages/bionic/man8/cputool.8.html⟩（参照 2022-02-12）．