OPC UA Publisher プロバイダ

Version 1.1.0

ユーザーズ ガイド

December 16, 2024

【備考】

本書で使用している画像の一部は，参考文献1，2から引用しています．

本書の付録 A は，参考文献1から引用しています．

本書の付録 B の一部は，参考文献2から引用しています．

【改版履歴】

バージョン	日付	内容
1.0.0	2024-3-18	初版.
1.1.0	2024-12-16	内部使用ライブラリの更新.

目次

1. はじめに

本書は OPC UA Publisher プロバイダのユーザーズガイドです.

OPC UA Publisher プロバイダでは, OPC UA UDP メッセージ(以下メッセージ)を発行し, メッセージ指向ミドルウェアに対して送信することができます.

第 2 章に OPC UA Publisher プロバイダの概要, 変数の詳細を記載しています.

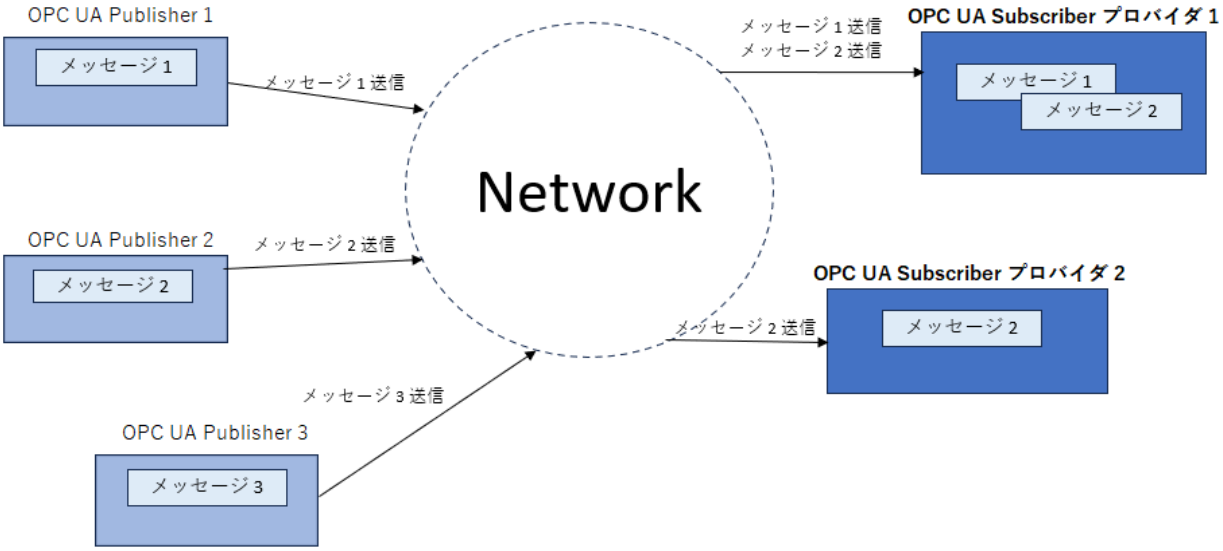

図 1-1 OPC UA Publisher プロバイダの概要

2. プロバイダの概要

2.1. 概要

OPC UA Publisher プロバイダは，メッセージの送信機能を提供する CAO プロバイダです．

そのファイル形式は DLL（Dynamic Link Library）であり，CAO エンジンから使用時に動的にロードされます．メッセージ指向ミドルウェアに対してメッセージを送信して，Subscriber にメッセージを通知することを目的として使用します．

表 2-1 OPC UA Publisher プロバイダ

ファイル名	CaoProvOPCUAPublisher.dll
ProgID	CaoProv.OPCUA.Publisher
レジストリ登録	regsvr32 CaoProvOPCUAPublisher.dll
レジストリ登録の抹消	regsvr32 /u CaoProvOPCUAPublisher.dll

2.1.1. OPC UA

OPC Unified Architecture（UA）は，各 OPC Classic 仕様の機能性全てを，拡張可能なフレームワークに統合した，プラットフォーム非依存のサービス指向アーキテクチャです．

2.1.2. OPC UA UDP

NetworkMessage の転送に使用される UDP ベースのプロトコルで，Message Oriented Middleware（メッセージ指向ミドルウェア）の一つです．UDP 転送プロトコル URL は，次の書式で記述されます．

書式：opc.udp://<host>[:<port>]

- OPC UA Publisher プロバイダでは，URL の<host>を IP アドレスで指定時は，IPv4 で指定します．

詳細は，参考文献1を参照してください．OPC UA PubSub 通信の概要は 3.付録 B を参照してください．

2.1.3. OPC UA Publisher プロバイダの機能

OPC UA Publisher プロバイダは以下の機能を提供しています．

➢ メッセージの送信

- メッセージ指向ミドルウェアを通じて，Subscriber に対して周期的にメッセージを送信することができます．
- メッセージの送信は Variable で行います．
  Variable は複数作成することができ，それぞれで別々のメッセージを送信することが可能です．メッセージを区別するための情報は PublisherId， WriterGroupId， DataSetWriterId の組み合わせで指定

できます.

- メッセージを送信する際に周期間隔を指定します.
  - PublishingInterval, KeyFrameCount を使用して周期間隔を指定します.
    - ✧ 周期間隔は PublishingInterval×KeyFrameCount で決定されます.
    - ✧ KeyFrameCount が 1 の場合は PublishingInterval の間隔でメッセージを送信し続けます.
    - ✧ KeyFrameCount が 2 以上の場合は初回のメッセージは PublishingInterval の時間が経過時に送信され, それ以降は周期間隔内で PublishingInterval の時間が経過するごとに差分チェックを行います. 差分があった場合には周期間隔の時間が経過する前でもメッセージを送信し, 差分がない場合は周期間隔でメッセージを送信し続けます. 詳細については 2.2.4.2 を参照してください.
- 未サポートの型を使用してメッセージを送信しようとすると, 未サポートエラーが発生します.
  サポートしている型については 2.2.6 を参照してください.
- メッセージの送信は AddVariable を行ったタイミングでは行われず, put_Value を行ったタイミングで開始されます. ただし, AddVariable 時に WaitDataSetWriter や StartPublish を指定していた場合はこの限りではありません. 詳細については 2.2.4.3 と 2.2.4.4 を参照してください.
- メッセージ長は, 最大 65535 バイトです. これは IP ヘッダや UDP ヘッダ, UADP メッセージの各種ヘッダやオプション等の長さを含みます.

➢ メタ情報(DataSetMetaData)の生成と送信

- OPC UA Publisher プロバイダは, 送信するメッセージから OPC UA Subscriber が正しく情報を取得するためのメタ情報を生成します.
- OPC UA Publisher プロバイダは, OPC UA Subscriber からメタ情報のリクエストを受け取ることがあり, その際はメタ情報が送信されます.
- メタ情報には作成時の時間がバージョンとして保持されています.

➢ 最後に設定されたメッセージの読み出し

メッセージの読み出し(CaoVariable::get_Value)が行われると, 最後に送信用データとして設定されたデータを読み出します.

- 設定されたメッセージがない場合, VT_EMPTY を返します.

➢ 注意事項

- KeyFrameCount を 1 以外で指定した場合, その WriteGroup に紐づく DataSetWriter は単一となり他の DataSetWriter は紐づけできなくなります.

2.1.4. OPC UA Publisher プロバイダと OPC UA Subscriber プロバイダのデータ送受信の流れ

OPC UA Publisher プロバイダとは別に受信側となる OPC UA Subscriber プロバイダ(CaoProvOPCUASubscriber.dll)が別途, 実装されています.

OPC UA Subscriber プロバイダとのデータのやり取りについて記載します.

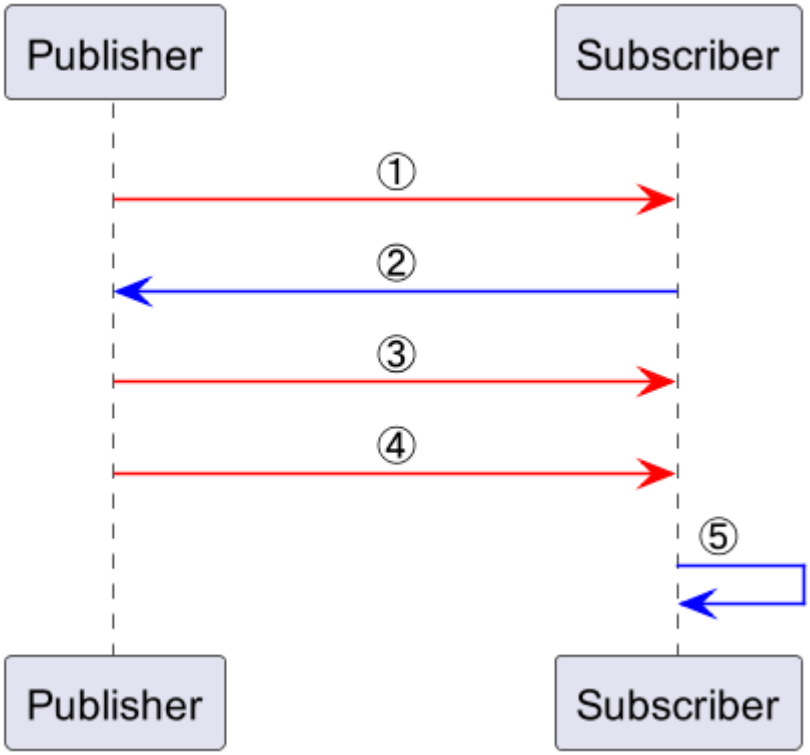

図 2-1 起動時に Subscriber が既に PubsubConnection に登録されている場合の流れ

①Publisher 起動時に WriterData を送信
②WriterData を受け取ってメタデータの要求
③Subscriber 側から要求を受け取ってメタデータを送信
④ネットワークメッセージの送信
⑤受信したメタデータを使用してメッセージを変換

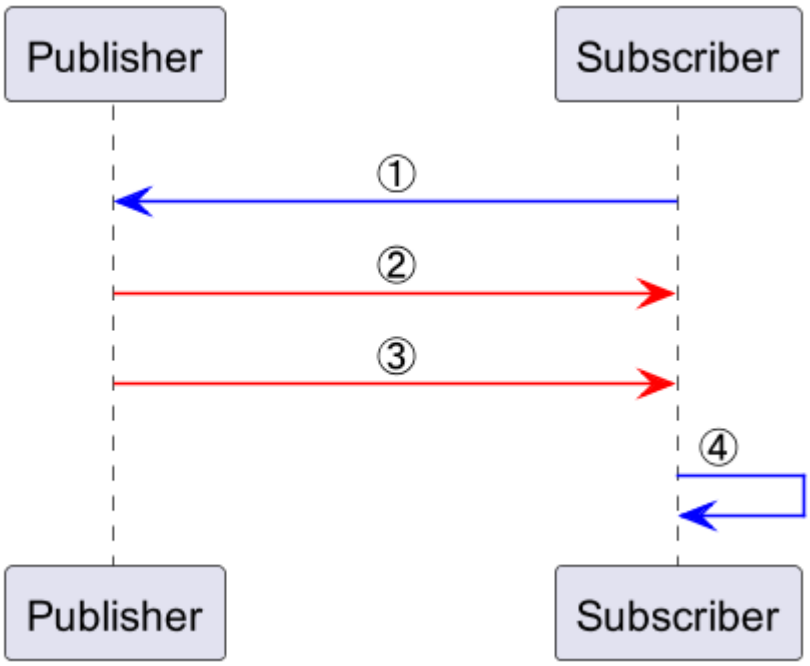

図 2-2 Subscriber より先に PubsubConnection に登録されていた場合の流れ

①Subscriber 起動時にメタデータの要求
②Subscriber 側から要求を受け取ってメタデータを送信
③ネットワークメッセージの送信
④受信したメタデータを使用してメッセージを変換

2.2. メソッド・プロパティ

2.2.1. CaoWorkspace::AddController メソッド

OPC UA Publisher プロバイダは AddController 時に通信用の接続パラメータを参照し，PubSubConnection を作成します．

書式 AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,
<bstrPCName:BSTR>,<bstrOption:BSTR>))

bstrCtrlName : [in] コントローラ名
bstrProvName : [in] プロバイダ名．固定値=”CaoProv.OPCUA.Publisher”
bstrPcName : [in] プロバイダの実行マシン名
bstrOption : [in] オプション文字列

以下にオプション文字列に指定するリストを示します．

表 2-2 CaoWorkspace::AddController のオプション文字列

オプション名[1]	意味[2]
UADPUrl=<送信先の IP アドレス>	必須指定項目． 接続パラメータを指定します． （参照：2.2.1.1）
DiscoveryUrl[=<DataSetMetaData の Request 応答送信先>]	省略可能項目． DataSetMetaData の Request 受信および応答送信用の IP アドレス． （デフォルト：opc.udp://224.0.2.14:4840）[3] （参照:2.2.1.2）
NetworkInterface[=<送信先の NIC の IP アドレス>]	省略可能項目． 送信先のネットワークインターフェースを指定します． （デフォルト：0.0.0.0） （参照：2.2.1.3）
PublisherIdDataType=<データタイプ>	必須指定事項． PublisherId のデータタイプを指定します． （参照：2.2.1.4）

[1] 角括弧（“[ ]”）内は省略可能を示します．
[2] 省略可能項目はオプション名自体を省略可能です．
[3] 224.0.2.14 は IANA に登録された OPC UA 検出用の IPv4 マルチキャストアドレスです．

PublisherId=<ID 数字/文字列>	必須指定事項. PublisherId を指定します. （参照:2.2.1.5）

2.2.1.1. UADPUrl オプション

以下に UADPUrl オプションのパラメータ文字列を示します.

"UADPUrl=opc.udp://<host>[:<port>]"[4]

例:"UADPUrl=opc.udp://239.0.0.1:4840"

例:"UADPUrl=opc.udp://239.0.0.1"

<host> : 接続先ホスト名または接続先 IP アドレス.
送信先のマルチキャストアドレス[5]またはユニキャストアドレスから使用したい通信形式のアドレスを指定します.
例:"239.0.0.1"

<port> : 送信先の接続ポート番号を指定します.（デフォルト:4840）
例:"4840"

2.2.1.2. DiscoveryUrl オプション

以下に DiscoveryUrl オプションのパラメータ文字列を示します.

"DiscoveryUrl[=<opc.udp://<host>[:<port>]]"

<host> : DataSetMetaData の Request 受信および応答送信用の IP アドレス.
マルチキャストアドレスを指定します. [6]
（デフォルト:224.0.2.14）

<port> : 接続ポート番号を指定します.
（デフォルト:4840）

2.2.1.3. NetworkInterface オプション

以下に NetworkInterface オプションのパラメータ文字列を示します.

"NetworkInterface[=<送信先の NIC の IP アドレス>]"

<送信先の NIC の IP アドレス> : 送信先の NIC の IP アドレス.
0.0.0.0 ですべての NIC を対象に動作します.（デフォルト:0.0.0.0）

[4] "opc.udp"部分は, "opc"と"udp"をピリオドで繋いでいます.
[5] マルチキャストアドレスの範囲は 224.0.0.0～239.255.255.255.
[6] DiscoveryUrl ではユニキャストでの指定はできません.

2.2.1.4. PublisherIdDataType オプション

PublisherId の型を番号で指定します.

表 2-3 型番号と対応するデータ型

番号	データ型	入力範囲
1	Byte	0～255
2	UInt16	0～65535
3	UInt32	0～4294967295
4	UInt64	0～18446744073709551615
5	String	無制限[7]

2.2.1.5. PublisherId オプション

PublisherIdDataType オプションで指定した型の範囲内の値を指定します. (表 2-3 参照)

2.2.1.4 で設定した PublisherIdDataType と PublisherId が同一の組み合わせのものを設定した場合はエラーになります.

[7] Std::string.size()の戻り値の型 size_t の最大値に制限されます. 本プロバイダでの最大値は 4294967295 です.

2.2.2. CaoController::AddExtension メソッド

CaoController クラスの AddExtension メソッドは，CaoExtension オブジェクトを使用して，それぞれのプロバイダで Publish を行うために使用する情報を WriterGroup 単位で管理します．

Extension を作成後に Variable を作成して，Variable は Extension で管理されている情報を参照して使用します．Extension の削除[8]で WriterGroup を削除します．

書式　AddExtension(<bstrExtensionName:VT_BSTR>[,<bstrOption:VT_BSTR>])

<bstrExtensionName>　：　[in] 変数名

<bstrOption>　：　[in] オプション文字列

以下にオプション文字列に指定するリストを示します．

表 2-4 CaoController::AddExtension のオプション文字列

オプション名 [1]	意味 [2]
WriterGroupId=<ID 数字>	必須指定事項． WriterGroupId を指定します． （参照：2.2.2.1）
PublishingInterval[=<データを送信する間隔>]	省略可能事項 周期的に送信する場合に使用 メッセージを送信する間隔をミリ秒で指定します． （デフォルト:1000） (参照：2.2.2.2)
KeepAliveTime[=<KeepAliveMessage の送信間隔>]	省略可能事項． KeepAliveMessage の送信間隔をミリ秒で指定します． (デフォルト:10000) (参照:2.2.2.3)
MaxNetworkMessageSize[=<NetworkMessage の最大サイズ>]	省略可能事項． 送信する NetworkMessage の最大サイズをバイト数で指定します． （デフォルト：1400）

[8] 削除時には接続が一時的に OFF になるため，削除が完了するまでの間は同一の Controller から作成された全ての Variable でメッセージが送信できない状態になります．

	（参照:2.2.2.4）
WaitWriterGroup[=<TRUE/FALSE>]	省略可能事項. WriterGroup の作成タイミングを指定します. （デフォルト:FALSE） （参照:2.2.2.5）
AutoRebootWriterGroup[=<TRUE/FALSE>]	省略可能事項. メタデータの変更時に自動的にWriterGroup を再起動するかを指定します. （デフォルト:TRUE） (参照:2.2.2.6)

2.2.2.1. WriterGroupId オプション

指定可能範囲内(1～65535)の値を指定します.

同一コントローラ内での重複があった場合, エラーになります.

2.2.2.2. PublishingInterval オプション

送信を周期的に行う際に指定します.

指定した数値に従って, データを送信する間隔が決定されます.

PublishingInterval[=<データを送信する間隔>]

< データを送信する間隔> : 1[9]～2147483647

送信されるデータの種類, タイミングについての詳細は 3.付録 C を参照してください.

2.2.2.3. KeepAliveTime オプション

本オプションで PublishingInterval オプションより小さい値を指定した場合はエラーになります.

本オプションを指定していない状態で, PublishingInterval が KeepAliveTime のデフォルト値の 10000 ミリ秒より大きくなる場合, 本オプションの値は PublishingInterval と同等の値に自動で変更されます.

本オプションを指定すると以下のタイミングと条件で KeepAliveMessage が送信されます.

[9] 短い送信間隔を指定すると, OS の性能によっては指定した送信間隔でメッセージが送信できない場合があります.

タイミング	前回データ送信時から,KeepAliveTime の時間が経過後，最初の PublishingInterval の時間が経過したタイミング
条件	データの差分チェックで差分がなかった場合 (差分がある場合は DeltaFrameMessage が送信される)

KeepAliveTime[=<KeepAliveMessage の送信間隔>]

<KeepAliveMessage の送信間隔> ： 1～2147483647

送信されるデータの種類，タイミングについての詳細は 3.付録 C を参照してください．

2.2.2.4. MaxNetworkMessageSize オプション

送信するデータの最大サイズ（バイト）を指定します．

設定値より大きなサイズのデータは送信されません．[10]

最大サイズを超えたデータは分割して送信されます．

MaxNetworkMessageSize[=<送信するデータの最大サイズ>]

<送信するデータの最大サイズ> ： 1400[11]～65535[12]

2.2.2.5. WaitWriterGroup オプション

WriterGroup を作成するタイミングを指定します．

TRUE を指定した場合は，WriterGroup の作成が待機状態となります．

待機状態の間はメッセージが送信できない(Publish されない)状態になります．

待機状態の WriterGroup を作成してメッセージを送信できる状態にする場合は CreateWriterGroup コマンドを実行してください．

待機状態で put_Value が行われた場合，Publish は開始されず，CreateWriterGroup コマンドを実行したタイミングで put_Value にて設定した送信データで Publish が開始されます．

FALSE を指定した場合は，WriterGroup が即時作成されます．

ただし WriterGroup の作成時には接続が一時的に OFF になるため，作成が完了するまでの間は同一の Controller から作成された全ての Variable でメッセージが送信できない状態になります[13]．

2.2.2.6. AutoRebootWriterGroup オプション

送信データの型を変更した際に Subscriber に対して DataSetMetaData の変更を通知するために

[10] エラー通知などは行われません．
[11] WriterData(Publish 開始時に送信する構成情報)や，メッセージのヘッダ部分など，一部の分割できないデータのサイズを下回る値を指定した場合，正常に送信が行えないことがあります．
[12] 実際にはヘッダなども含んでの値です．
[13] 接続が一時的に OFF になるため，送信間隔の経過時間がリセットされます．

WriterGroup の再起動が必要な場合があります. 本オプションで WriterGroup の再起動を自動で行うか, 手動で行うかを選択することができます.

TRUE を選択した場合, 送信データの型を変更した際に自動的に WriterGroup が再起動されて Subscriber 側に DataSetMetaData の変更を通知します.

FALSE を選択した場合, Subscriber に DataSetMetaData の変更通知が送られないため, Subscriber 側がデータの受信ができなくなる場合があります. Subscriber に対して DataSetMetaData の変更を通知するためには RebootWriterGroup コマンドを実行して WriterGroup を再起動してください.

ただし WriterGroup の再起動時には接続が一時的に OFF になるため, 再起動が完了するまでの間は同一の Extension から作成された全ての Variable でメッセージが送信できない状態になります [13].

なお送信データの型が変更されると MetaData 変更通知メッセージ（参照 : 2.2.8.2）が通知されます. RebootWriterGroup コマンド実行時に指定する Extention 名はこのメッセージから取得することができます.

2.2.3. CaoController::Execute メソッド

指定コマンドを汎用的に実行します.

使用できるコマンド名と詳細は 2.3.1 を参考にしてください.

書式 Execute(<bstrCommand:VT_BSTR>[,<vntParam:VARIANT>[,<pVal:VARIANT>]])

<bstrCmd>	:	[in] コマンド名
<vntParam>	:	[in] パラメータ
<pVal>	:	[out] 実行結果

2.2.4. CaoExtension::AddVariable メソッド

CaoExtension クラスの AddVariable メソッドは，それぞれのプロバイダで Publish を行うための変数オブジェクトを作成するメソッドです．

OPC UA Publisherプロバイダは変数の作成後に送信するデータ型とデータを指定しput_Valueを実行後，Publish を開始します．変数の削除[14]で UnPublish を実行します．

1 つの Extension に複数 Variable を追加する場合，KeyFrameCount は 1 でなければいけません．

書式　AddVariable(<bstrVariableName:VT_BSTR>[,<bstrOption:VT_BSTR>])

<bstrVariableName> : [in] 変数名
<bstrOption> : [in] オプション文字列

以下にオプション文字列に指定するリストを示します．

表 2-5 CaoExtension::AddVariable のオプション文字列

オプション名 [1]	意味 [2]
DataSetWriterId=<ID 数字>	必須指定事項. DataSetWriterId を指定します. （参照:2.2.4.1）
KeyFrameCount[=<PublishingInterval が期限切れになる最大回数>]	省略可能事項. PublishingInterval が期限切れになる最大回数を指定します. 本オプションを 2 以上に指定すると，PublishingInterval が指定した回数経過するか，メッセージに変更があった場合のみメッセージを送信するようになります. 1 つの Extension に複数の Variable を紐づけたい場合はデフォルト値を使用してください. (デフォルト:1) (参照:2.2.4.2)

[14] 削除時には接続が一時的に OFF になるため，削除が完了するまでの間は同一の Extension から作成された全ての Variable でメッセージが送信できない状態になります．

WaitDataSetWriter[=<TRUE/FALSE>]	省略可能事項. DataSetWriter の作成タイミングを指定します. （デフォルト:FALSE） （参照:2.2.4.3）
StartPublish[=<TRUE/FALSE>]	省略可能事項 Publish の開始有無を指定します. (デフォルト:TRUE) （参照:2.2.4.4)
SendByteString[=<TRUE/FALSE>]	省略可能事項 VT_UI1|VT_ARRAY の送信形式を指定します. (デフォルト:FALSE) (参照:2.2.4.5)

2.2.4.1. DataSetWriterId オプション

指定可能範囲内(1～65535)の値を指定します.

同一コントローラ内での重複があった場合，エラーになります.

2.2.4.2. KeyFrameCount オプション

PublishingInterval が期限切れになる最大回数を指定します.

本オプションを指定すると送信周期が以下のタイミングに設定されます.

ただし，初回メッセージは最初の PublishInterval の時間が経過したタイミングで送信されます.

送信周期 ＝ PublishingInterval × KeyFrameCount

本オプションを 2 以上で指定している場合，送信周期内に PublishingInterval の時間が経過すると，データの差分チェックを行うようになり，差分があった場合には DeltaFrameMessage として差分のみのデータが送信されます. [15]

1 つの Extension に複数の Variable を紐づけたい場合は，本オプションは 1(デフォルト値)を使用してください.

KeyFrameCount[=<PublishingInterval が期限切れになる最大回数>]

< PublishingInterval が期限切れになる最大回数> ： 1～4294967295

[15] DeltaFrameMessage は変更されたデータを送信するために追加のインデックスが必要となるため，KeyFrameMessage よりデータが大きくなる可能性があり，その場合は KeyFrameMessage が送信されます.

送信されるデータの種類，タイミングについての詳細は 3.付録 C を参照してください．

2.2.4.3. WaitDataSetWriter オプション

DataSetWriter を作成するタイミングを指定します．

TRUE を指定した場合は，DataSetWriter の作成が待機状態となります．

待機状態の間はメッセージが送信できない(Publish されない)状態になります．

待機状態の DataSetWriter を作成してメッセージを送信できる状態にする場合は CreateDataSetWriter コマンドを実行してください．

待機状態で put_Value が行われた場合，Publish は開始されず，CreateDataSetWriter コマンドを実行したタイミングで put_Value で設定した送信データで Publish が開始されます．

FALSE を指定した場合は，DataSetWriter が即時作成されます．

ただし DataSetWriter の作成時には接続が一時的に OFF になるため，作成が完了するまでの間は同一の Extension から作成された全ての Variable でメッセージが送信できない状態になります[16]．

2.2.4.4. StartPublish オプション

Publish の開始有無を指定します．

TRUE を指定した場合は，put_Value を行ったタイミングで Publish を開始します．

FALSE を指定した場合は，Publish を行いません．

それぞれ，StartPublish コマンドと StopPublish コマンドを実行することでも指定できます．

本オプションが FALSE の状態で put_Value が行われた場合，Publish は開始されず，本オプションが TRUE の状態になったタイミングで put_Value で設定した送信データで Publish が開始されます．

2.2.4.5. SendByteString オプション

特定の型(VT_ARRAY | VT_UI1)を含む内容で送信を実施する際に，ByteString 型として送信するか，VT_UI1 型の配列として送信するかを選択する際に指定します．

TRUE を指定した場合は ByteString 型として送信されます．

FALSE を指定した場合は VT_UI1 の配列として送信されます．

2.2.5. CaoExtension::Execute メソッド

指定コマンドを汎用的に実行します．

使用できるコマンド名と詳細は 2.3.2 を参考にしてください．

書式　Execute(<bstrCommand:VT_BSTR>[,<vntParam:VARIANT>[,<pVal:VARIANT>]])

[16] 接続が一時的に OFF になるため，送信間隔の経過時間がリセットされます．

<bstrCmd> : [in] コマンド名

<vntParam> : [in] パラメータ

<pVal> : [out] 実行結果

2.2.6. CaoVariable::put_Value プロパティ

送信データを設定して送信を開始します.

設定された VARIANT 型から, データ型を OPC UA の組み込みデータ型へ変換, 対応する DataSetMetaData の作成を行い, Publish を実行します.

DataSetMetaData は作成時の時間をバージョンとして保持しています.

最後に送信した Publish データと比べてデータ型が変更されていない場合は, DataSetMetaData の作成は行わず, 値のみを更新して Publish を実行します.

制限事項

下表に記載のないデータ型を含む場合は, 設定できません.

二次元以上の配列は対応していません.

表 2-6 対応しているデータタイプ

OPC UA の組み込みデータ型	VARIANT のデータ型	説明
SByte	VT_I1	1 バイト整数型
Int16	VT_I2	2 バイト整数型
Int32	VT_I4	4 バイト整数型
Int64	VT_I8	8 バイト整数型
Byte	VT_UI1	1 バイト符号なし整数型
UInt16	VT_UI2	2 バイト符号なし整数型
UInt32	VT_UI4	4 バイト符号なし整数型
UInt64	VT_UI8	8 バイト符号なし整数型
Float	VT_R4	4 バイト浮動小数点型
Double	VT_R8	8 バイト浮動小数点型
DateTime	VT_DATE	日付型
String	VT_BSTR	文字列型
Boolean	VT_BOOL	ブール型
ByteString	VT_UI1 | VT_ARRAY	バイトストリング型

2.2.7. CaoVariable::get_Value プロパティ

最後に送信した Publish データを取得します.

データが設定されていない場合は VT_EMPTY が返ります.

2.2.8. CaoController::OnMessage イベント

以下の発生契機で CaoController クラスの OnMessage イベントが発生します.

表 2-7 メッセージ種別

メッセージ種別		発生契機
0	送信動作中エラー	何らかの要因により送信動作中にエラーが生じた際に発生します.
1	MetaData 変更通知	送信されるデータの型が変更された場合に通知されます.

2.2.8.1. 送信動作中エラー

送信動作中エラーメッセージで得られるデータ形式を以下に示します.

Number : メッセージ種別 (0)

Value : 内部エラーコード

DateTime : タイムスタンプ

Destination : AddExtension メソッドの<bstrExtensionName>

Source : AddVariable メソッドの<bstrVariableName>

Description : エラー説明情報[17]

2.2.8.2. MetaData 変更通知

MetaData 変更通知で得られるデータ形式を以下に示します.

Number : メッセージ種別 (1)

Value : AutoRebootWriterGroup オプションの設定値 (VT_BOOL)[18]

DateTime : タイムスタンプ

Destination : AddExtension メソッドの<bstrExtensionName>

Source : AddVariable メソッドの<bstrVariableName>

Description : メタデータ更新通知情報

[17] 内部エラーコードに関連する説明情報が無い場合は空文字列を格納

[18] AddExtension 時に AutoRebootWriterGroup オプションに指定した設定値

2.3. コマンド一覧

2.3.1. CaoController クラス

表 2-8 CaoController::Execute コマンド一覧

コマンド	機能	参照
CreateWriterGroup	2.2.2.5 で TRUE を指定して待機状態にある WriterGroup を作成します. WriterGroup の作成時には接続が一時的に OFF になるため, 作成が完了するまでの間は同一の Controller から作成された全ての Variable でメッセージが送信できない状態になります.	P. 22
StartVariablesPublish	Extension に紐づく全ての Variable の Publish を開始します.	P. 22
StopVariablesPublish	Extension に紐づく全ての Variable の Publish を停止します.	P. 23
RebootWriterGroup	指定した Extension に対応する WriterGroup を再起動します. 有効な DataSetWriter が存在しない WriteGroup を指定した場合は, エラーになります. WriterGroup の再起動時には接続が一時的に OFF になるため, 再起動が完了するまでの間は同一の Extension から作成された全ての Variable でメッセージが送信できない状態になります.	P. 23

CreateWriterGroup

構文	object.CreateWriterGroup()
引数	なし.
戻り値	なし.
説明	2.2.2.5 で TRUE を指定して待機状態にある WriterGroup を作成します. WriterGroup の作成時には接続が一時的に OFF になるため, 作成が完了するまでの間は同一の Controller から作成された全ての Variable でメッセージが送信できない状態になります.

StartVariablesPublish

構文	object.StartVariablesPublish(Data)
引数	Data.VT=VT_BSTR Data.bstrVal=ExtensionName
戻り値	なし.

説明	指定した Extension に紐づく全ての Variable で Publish を開始します.

StopVariablesPublish

構文	object.StopVariablesPublish(Data)
引数	Data.VT=VT_BSTR Data.bstrVal=ExtensionName
戻り値	なし.
説明	指定した Extension に紐づく全ての Variable で Publish を停止します.

RebootWriterGroup

構文	object.RebootWriterGroup(Data)
引数	Data.VT=VT_BSTR Data.bstrVal=ExtensionName
戻り値	なし.
説明	指定した Extension に対応する WriterGroup を再起動します. 有効な DataSetWriter が存在しない WriteGroup を指定した場合は, エラーになります. WriterGroup の再起動時には接続が一時的に OFF になるため, 再起動が完了するまでの間は同一の Extension から作成された全ての Variable でメッセージが送信できない状態になります.

2.3.2. CaoExtension クラス

表 2-9 CaoExtension::Execute コマンド一覧

コマンド	機能	参照
CreateDataSetWriter	2.2.4.3 で TRUE を指定して待機状態にある DataSetWriter を作成します. DataSetWriter の作成時には接続が一時的に OFF になるため, 作成が完了するまでの間は同一の Extension から作成された全ての Variable でメッセージが送信できない状態になります.	P. 24
StartPublish	指定した Variable で Publish を開始します.	P. 24
StopPublish	指定した Variable で Publish を停止します.	P. 24

CreateDataSetWriter

構文	object.CreateDataSetWriter()
引数	なし.
戻り値	なし.
説明	2.2.4.3 で TRUE を指定して待機状態にある DataSetWriter を作成します. DataSetWriter の作成時には接続が一時的に OFF になるため, 作成が完了するまでの間は同一の Extension から作成された全ての Variable でメッセージが送信できない状態になります.

StartPublish

構文	object.StartPublish(Data)
引数	Data.VT=VT_BSTR Data.bstrVal=VariableName
戻り値	なし.
説明	指定した Variable で Publish を開始します.

StopPublish

構文	object.StopPublish(Data)
引数	Data.VT=VT_BSTR Data.bstrVal=VariableName
戻り値	なし.
説明	指定した Variable で Publish を停止します.

2.4. 変数一覧

2.4.1. CaoController クラス

表 2-10 CaoController クラス システム変数一覧

変数名	データ型	説明	属性	
			get	put
@VERSION	VT_BSTR	プロバイダバージョン情報.	○	―

2.4.2. CaoExtension クラス

表 2-11 CaoExtension クラス ユーザー変数一覧

変数名	データ型	説明	属性	
			get	put
任意	put_Value で指定できるデータ型(参照 2.2.6)	変数の作成後, put_Value を行ったタイミングで指定されたデータ型と値でデータを作成し, 送信します. (参照:2.2.4)(参照:2.2.6)	○	○

2.5. エラーコード

OPC UA Publisher プロバイダでは, 以下の固有エラーコードが定義されています.

ORiN2 共通エラーについては,「ORiN2 プログラミングガイド」のエラーコードの章を参照してください.

表 2-12 固有エラーコード

エラー名	エラー番号	説明
E_FAILED_PUB_INVALID_PARAMS_UADPURL	0x80100001	送信処理に必要なパラメータ(UADPUrl オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_PUBLISHERIDDATATYPE	0x80100002	送信処理に必要なパラメータ(PublisherIdDataType オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_PUBLISHERID	0x80100003	送信処理に必要なパラメータ(PublisherId オプション)が不正値です.

E_FAILED_PUB_DUPLICATE_PARAMS_PUBLISHERID	0x80100004	PublisherIdDataTypeオプションとPublisherId オプションの組み合わせが重複しています.
E_FAILED_PUB_INVALID_PARAMS_WRITERGROUPID	0x80100005	送信処理に必要なパラメータ(WriterGroupId オプション)が不正値です.
E_FAILED_PUB_DUPLICATE_PARAMS_WRITERGROUPID	0x80100006	WriterGroupId オプションが重複しています.
E_FAILED_PUB_INVALID_PARAMS_PUBLISHINGINTERVAL	0x80100007	送信処理に必要なパラメータ(PublishingInterval オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_KEYFRAMECOUNT	0x80100008	送信処理に必要なパラメータ(KeyFrameCount オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_KEEPALIVETIME	0x80100009	送信処理に必要なパラメータ(KeepAliveTime オプション)が不正値です.
E_FAILED_PUB_KEEPALIVETIME_UNDER_PUBLISHINGINTERVAL	0x8010000A	送信処理に必要なパラメータ(KeepAliveTime オプション)がPublishingInterval オプションより小さいです.
E_FAILED_PUB_INVALID_PARAMS_MAXNETWORKMESSAGESIZE	0x8010000B	送信処理に必要なパラメータ(MaxNetworkMessageSize オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_WAITWRITERGROUP	0x8010000C	送信処理に必要なパラメータ(WaitWriterGroup オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_DATASETWRITERID	0x8010000D	送信処理に必要なパラメータ(DataSetWriterId オプション)が不正値です.
E_FAILED_PUB_DUPLICATE_PARAMS_DATASETWRITERID	0x8010000E	DataSetWriterId オプションが重複しています.
E_FAILED_PUB_INVALID_PARAMS_WAITDATASETWRITER	0x8010000F	送信処理に必要なパラメータ(WaitDataSetWriter オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_STARTPUBLIS	0x80100010	送信処理に必要なパラメータ

H		(StartPublish オプション)が不正値です.
E_FAILED_PUB_INVALID_PARAMS_SENDBYTESTRING	0x80100011	送信処理に必要なパラメータ(SendByteString オプション)が不正値です.
E_FAILED_PUB_INVALID_VALUEDATATYPE	0x80100012	対応してないデータ型です.
E_FAILED_PUB_CONVERT_DATA	0x80100013	送信するデータの変換処理に失敗しました.
E_FAILED_PUB_STARTUP_PUBS	0x80100014	送信するための情報設定に失敗しました. AddController や AddExtension, AddVariable のオプションの設定値を確認してください.
E_FAILED_PUB_LINK_DATASETWRITER	0x80100015	KeyFrameCount が 2 以上の Variable が存在している Extension に Variable を追加しようとしました. もしくは既に Variable が存在する Extension に KeyFrameCount が 2 以上の Variable を追加しようとしました.
E_FAILED_PUB_INVALID_PARAMS_AUTOREBOOTWRITERGROUP	0x80100016	送信処理に必要なパラメータ(AutoRebootWriterGroup オプション)が不正値です.
E_FAILED_PUB_NO_VALID_DATASETWRITER_EXISTS	0x80100017	有効な DataSetWriter が存在しない WriteGroup に対してコマンドを実行しました。

3. 付録

付録A. OPC UA PubSub 用語集

以下に OPC UA PubSub を扱う上で登場する用語を参考文献1より引用します.

- **DataSetClass**
  DataSet の内容を宣言するテンプレートです. DataSetClassId(GUID)をもっています.(参照 参考文献1 3.1.1)
- **DataSetMetaData**
  DataSet の内容と意味を説明します. DataSetClassId(GUID)を持つことができます.(参照 参考文献1 3.1.2)
- **DataSetReader**
  メッセージ指向ミドルウェアから DataSetMessage を受信するエンティティです.(参照 参考文献1 3.1.3)
- **DataSetWriter**
  DataSet から DataSetMessage を作成し, メッセージ指向ミドルウェアを通じて公開するエンティティです. PublisherId において一意の ID(DataSetWriterId)をもっています.(参照 参考文献1 3.1.4)
- **PublishedDataSet**
  DataSet として発行されるアプリケーションデータの構成です.(参照 参考文献1 3.1.5)
- **SubscribedDataSet**
  受信した DataSets をディスパッチするための構成です.(参照 参考文献1 3.1.7)
- **NetworkMessage**
  DataSetMessages を収めるコンテナです. DataSetMessages 間で共有される情報を含んでいます.(参照 参考文献1 5.3.4)
- **DataSetMessage**
  DataSet から作成され, ヘッダと DataSet のエンコードされたフィールドで構成されたものです.(参照 参考文献1 5.3.3)
- **DataSet**
  イベントまたは変数値のリストを表す名前と値のペアのリストです.(参照 参考文献1 5.2.1)
- **Publisher**
  NetworkMessage をメッセージ指向ミドルウェアに送信する PubSub エンティティです. (参照 参考文献1 5.4.1.1)
- **Subscriber**
  メッセージ指向ミドルウェアからの NetWorkMessages のコンシューマです. OPC UA のクライアントであったり, OPC UA のサーバーであったり, はたまたクライアントやサーバーでなく OPC UAPubSub のメッセージ構造を理解するためのアプリケーションであることもあります.(参照 参考文献1 5.4.2.1)

- **Message Oriented Middleware**
  分散システム間での NetworkMessage の送受信をサポートするインフラストラクチャです.（参照 参考文献1 5.4.4.1）
- **PubSubConnection**
  プロトコル選択，プロトコル設定，およびアドレス指定情報の組み合わせです.（参照 参考文献1 9.1.5.2）
- **WriterGroup**
  DataSetWriters のリストをグループ化したものです.（参照 参考文献1 6.2.5.6.1 ）
- **KeyFrameMessage**
  PublishedDataSet の現在の値がすべて含まれるメッセージです.（参照 参考文献1 6.2.3.3 ）
- **DeltaFrameMessage**
  前回送信メッセージからの差分のみのメッセージです.（参照 参考文献1 6.2.3.3 ）
- **KeepAliveMessage**
  長期間データが送られない場合に Subscriber 側に Publisher の接続が失われてないことを知らせるためのメッセージです.（参照 参考文献1 6.2.5.3 ）

付録B. OPC UA PubSub の概要

参考文献1, 参考文献2を参考に, OPC UA PubSub の概要を紹介します.

OPC UA Publisher がデータを送信し, OPC UA Subscriber がデータを受信します.

- 数に依存した接続情報の管理が不要です.
- 複数同時データ交換が可能です.
- OPC UA 通信セキュリティ使用可能です.
  - ➢本プロバイダは未対応です.

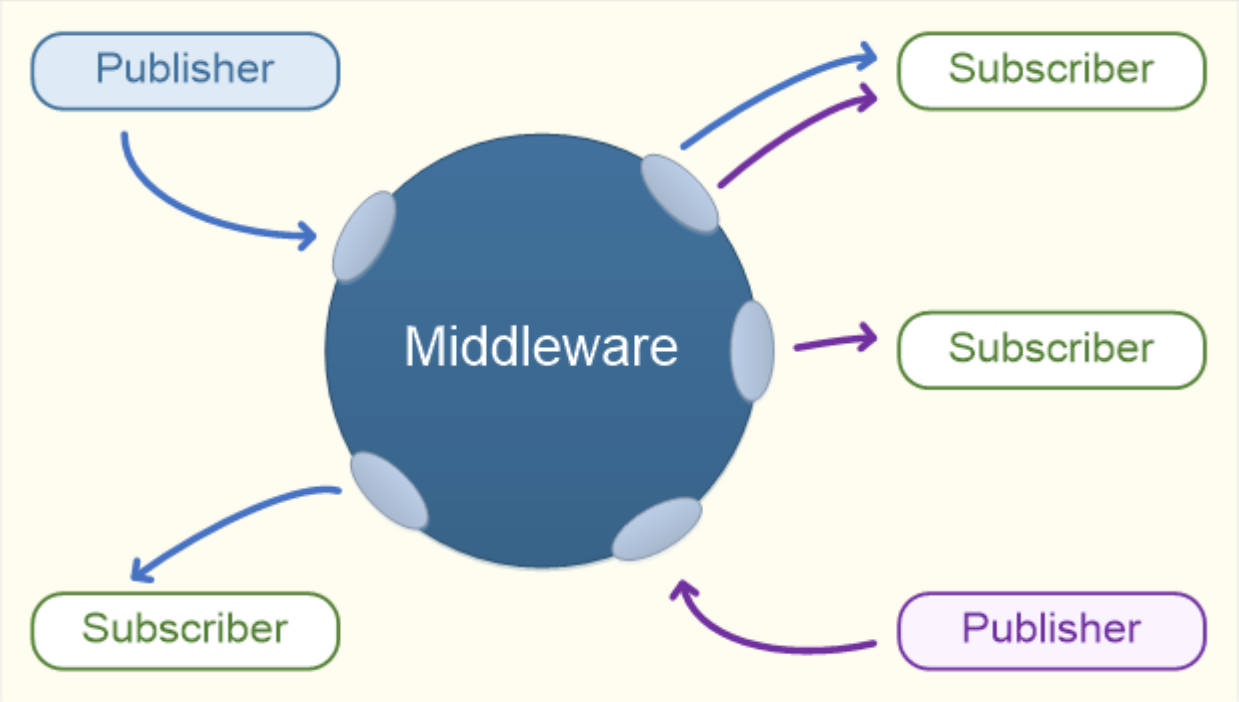

図 3-1 Publish Subscribe Model Overview（参考文献1より）

Publisher によるデータ送信は次の特徴をもっています.

- OPC UA UDP（UADP）マルチキャストによる高速通信です.
  - ➢ユニキャストによる通信も可能です.
- Broker（メッセージ指向ミドルウェア）によるセッションレス通信です.

Subscriber によるデータ受信は次の特徴をもっています.

- 必要なデータを指定して受信します.
  - ➢本プロバイダは UADP マルチキャストと UADP ユニキャストに対応します.

OPC UA Publisher は，ネットワークにおいて一意な PublisherId を持っています．また，Publisher において一意な DataSetWriterId，WriterGroupId を DataSetWriter,WriterGroup に付与しています．DataSetWriter らは，それぞれ DataSetMetaData と収集したデータから DataSetMessage にエンコードします．1 つ以上の DataSetWriter からなる集合を表す WriterGroup は，関連付けられた DataSetMessage らから NetworkMessage にエンコードします．Publisher はこの NetworkMessage を送信します．

OPC UA Subscriber は，OPC UA Publisher の{ PublisherId, [WriterGroupId,] DataSetWriterId }によって NetworkMessage をフィルタリングします．OPC UA Subscriber は，NetworkMessage を処理するために必要な DataSetMetaData を OPC UA Publisher から入手します．

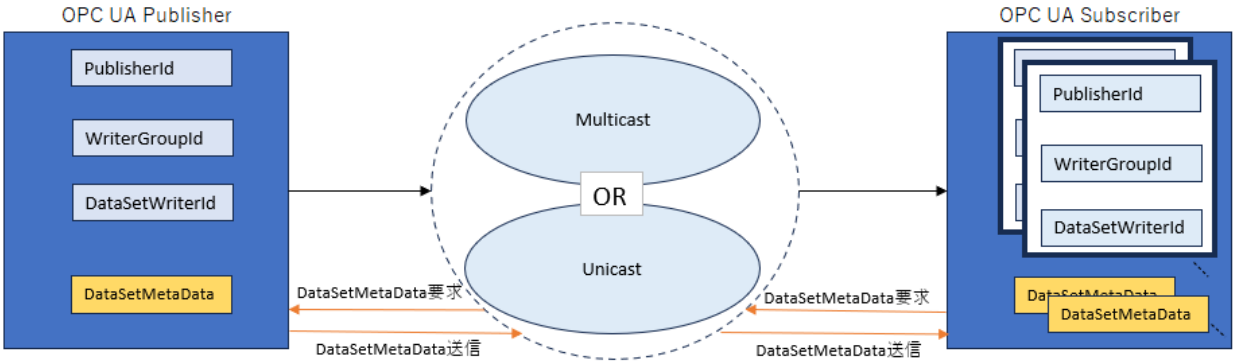

図 3-2 NetworkMessage のフィルタリングと処理に必要な情報

付録C. データの送信タイミングについて

データの送信タイミング及び送信されるデータの内容は「2.2.2.2.PublishingInterval オプション」,「2.2.2.3.KeepAliveTime オプション」,「2.2.4.2.KeyFrameCount オプション」の設定値により異なります.

各種データの送信タイミングについての詳細は以下のタイミングです.

KeyFrameMessage	送信周期(PublishingInterval と KeyFrameCount の乗算値)の時間が経過したタイミング
DeltaFrameMessage	KeyFrameMessage の送信周期内で PublisherInterval の時間が経過したタイミングで前回のデータと差分チェックを行い, 差分が発見された場合
KeepAliveMessage	KeyFrameMessage の送信周期内で前回のデータ送信時から KeepAliveTime の時間が経過後且つ, PublisherInterval の時間が経過したタイミングで前回のデータと差分チェックを行い, 差分が発見されなかった場合

以下の図にパターン例を記載します. 図の当てはまるパターンを参照してください.

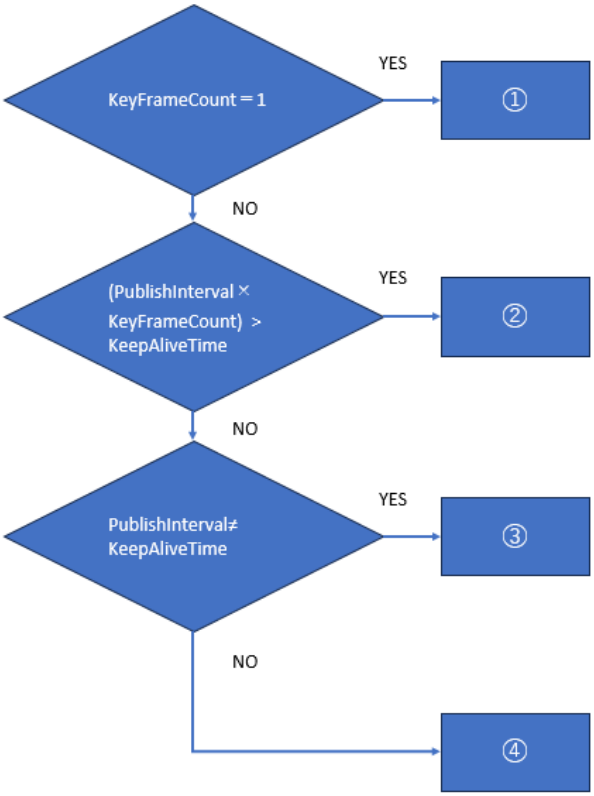

図 3-3 パターンフロー

- **パターン① KeyFrameCount オプションの設定値が 1(デフォルト値)の場合**

このパターンの場合，KeepAliveMessage や DeltaFrameMessage は送信されず，PublishingInterval 毎にデータが送信されます.

PublishInterval	100ms
KeyFrameCount	1(default)
KeepAliveTime	10000ms(default)

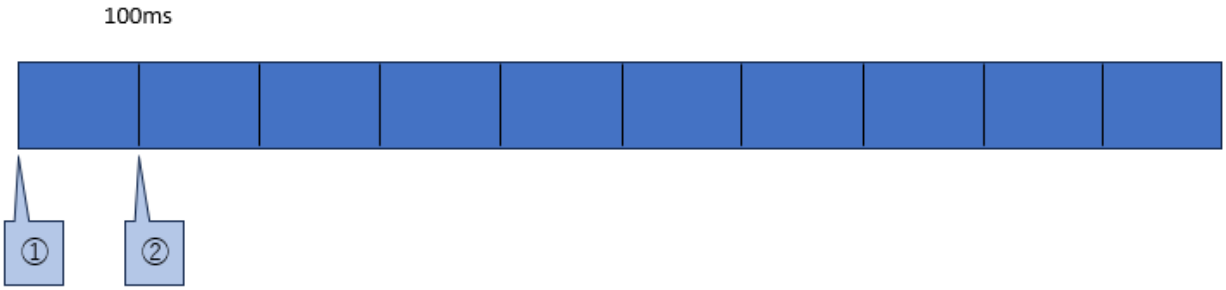

図 3-4 パターン図①

① 初回メッセージ送信

② 100 ミリ秒経過毎に KeyFrameMessage が送信されます

- **パターン② KeyFrameCount オプションを 2 以上に設定していて KeepAliveTime オプションが送信周期内にこない場合**

このパターンの場合，送信周期が PublishingInterval オプションと KeyFrameCount オプションの乗算値に設定され，送信周期内で PublishingInterval の時間が経過する度に前回送信メッセージとの差分チェックを行います．

その際，差分が確認されれば差分のみが DeltaFrameMessage として送信されます．

また，このパターンの場合，KeepAliveMessage は送信されません．

PublishInterval	100ms
KeyFrameCount	5
KeepAliveTime	10000ms(default)

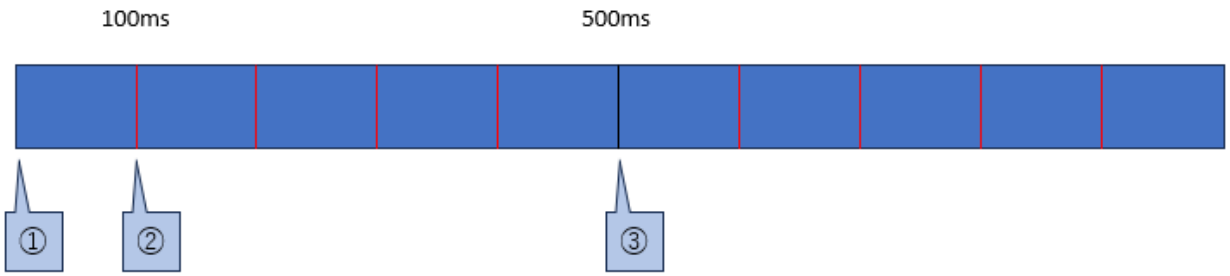

図 3-5 パターン図②

① 初回メッセージ送信．

② 100 ミリ秒経過毎に前回送信データと差分チェックを行い，差分があった場合，差分のみを DeltaFrameMessage として送信されます．[15]

③ 500 ミリ秒経過毎に KeyFrameMessage が送信されます．
※途中で KeyFrameMessage が送信された場合はそこから再び 500 ミリ秒カウントするので注意してください．

- **パターン③ KeyFrameCount オプションを 2 以上に設定していて KeepAliveTime オプションが送信周期内にくる場合**

このパターンの場合，送信周期が PublishingInterval オプションと KeyFrameCount オプションの乗算値に設定され，送信周期内で PublishingInterval の時間が経過する度に前回送信メッセージとの差分チェックを行います．

その際，差分が確認されれば差分のみを DeltaFrameMessage として送信されます．

また，差分チェック時に前回送信メッセージとの差分がなく，前回メッセージ送信から KeepAliveTime の時間が経過している場合は，KeepAliveMessage が送信されます．

PublishInterval	100ms
KeyFrameCount	5
KeepAliveTime	210ms

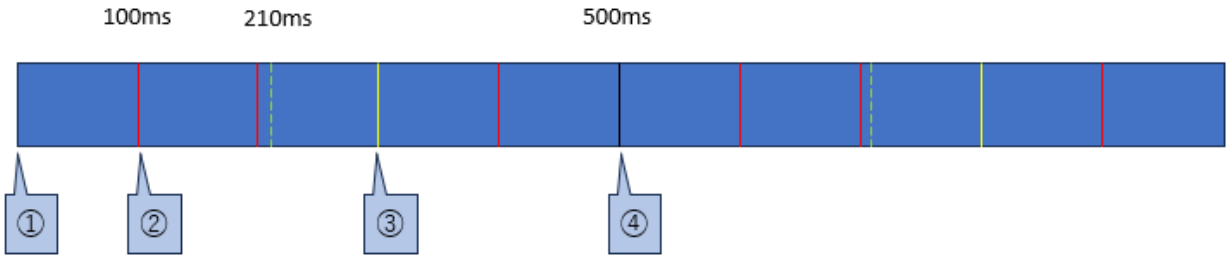

図 3-6 パターン図③

① 初回メッセージ送信

② 100 ミリ秒経過毎に前回送信データと差分チェックを行い，差分があった場合，差分のみが DeltaFrameMessage として送信されます．[15]
このタイミングでは KeepAliveTime の時間である 210 ミリ秒が経過してないので差分がなくても KeepAliveMessage は送信されません

③ 前回データ送信時から KeepAliveTime の時間である 210 ミリ秒が経過した後(※)，最初の差分チェックにて差分がなかった場合，KeepAliveMessage が送信されます．(差分があった場合は DeltaFrameMessage が送信されます．)
※KeyFrameMessage や DeltaFrameMessage もしくは KeepAliveMessage が送信された場合はそこから再び 210 ミリ秒カウントするので注意してください

④ 500 ミリ秒経過毎にデータが送信されます
※途中で KeyFrameMessage が送信された場合はそこから再び 500 ミリ秒カウントするので注意してください．

- **パターン④ KeyFrameCount オプションを 2 以上に設定していて KeepAliveTime オプションが PublishingInterval オプションと同等の場合**

このパターンの場合，送信周期が PublishingInterval オプションと KeyFrameCount オプションの乗算値に設定され，送信周期内で PublishingInterval の時間が経過する度に前回送信メッセージとの差分チェックを行います．

その際，差分が確認されれば差分のみを DeltaFrameMessage として送信されます．

また，差分チェック時に前回送信メッセージとの差分がなく，前回メッセージ送信から KeepAliveTime の時間が経過している場合は，KeepAliveMessage が送信されます．

ただし，このパターンの場合，処理タイミングの関係で KeepAliveTime の時間経過前に差分チェックが終わってしまい，KeepAliveMessage の送信がされない場合があります．

その場合は次の PublishingInterval の時間経過タイミングで KeepAliveMessage を送信するかの判定を行います．

KeepAliveTime の時間が経過していた場合の挙動

PublishInterval	100ms(default)
KeyFrameCount	5
KeepAliveTime	100ms

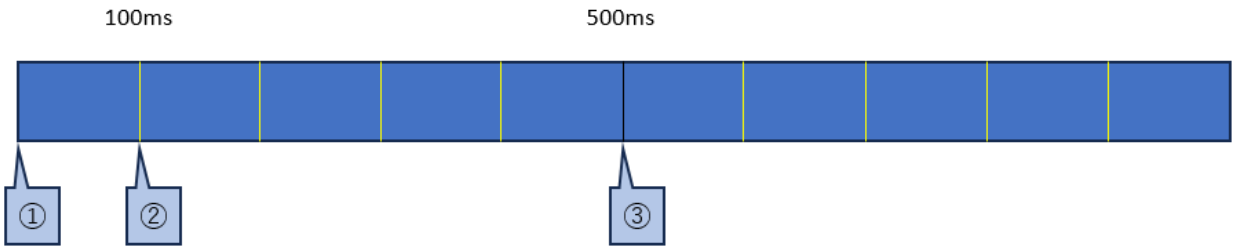

図 3-7 パターン図④-1

① 初回メッセージ送信．

② 100 ミリ秒経過毎に前回送信データと差分チェックを行い，差分があった場合，差分のみを DeltaFrameMessage として送信されます．[15]
差分がなかった場合，前回送信メッセージから KeepAliveTime の時間が経過していれば KeepAliveMessage が送信されます．

③ 500 ミリ秒経過毎にデータが送信されます．
※途中で KeyFrameMessage が送信された場合はそこから再び 500 ミリ秒カウントするので注意してください．

KeepAliveTime の時間が経過していなかった場合の挙動

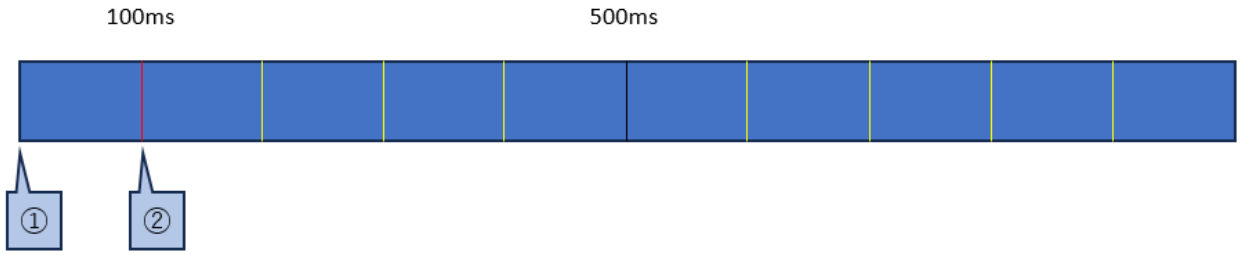

図 3-8 パターン図④-2

① 初回メッセージ送信.

② 本来は前回メッセージ送信から PublishingInterval の時間である 100 ミリ秒が経過した時点で差分チェックが行われ, 差分がなければ KeepAliveMessage が送信されるはずですが, メッセージ送信タイミングに数ミリ秒程度の差が出ることによって前回メッセージ送信時から KeepAliveTime の時間が経過していないケースがあり, KeepAliveMessage が送られない場合があります.
この場合は次回の PublishingInterval の時間経過時に差分がなければ KeepAliveMessage が送信されます.

付録D. 参考・引用文献一覧

- 参考文献1 :
  - OPC Foundation. OPC 10000-14: UA Part 14:PubSub v1.04. OPC Foundation.2018-02-06
  - https://reference.opcfoundation.org/Core/Part14/v104/docs
- 参考文献2:
  - OPC Foundation 里村虎和. OPC UA Pub/Sub の紹介と デモンストレーション. OPC Foundation.2018-12-14
  - https://jp.opcfoundation.org/wp-content/uploads/sites/2/2018/12/5_Satomura_PubSub.pdf