BACloud-GA-AZ インタフェース仕様書

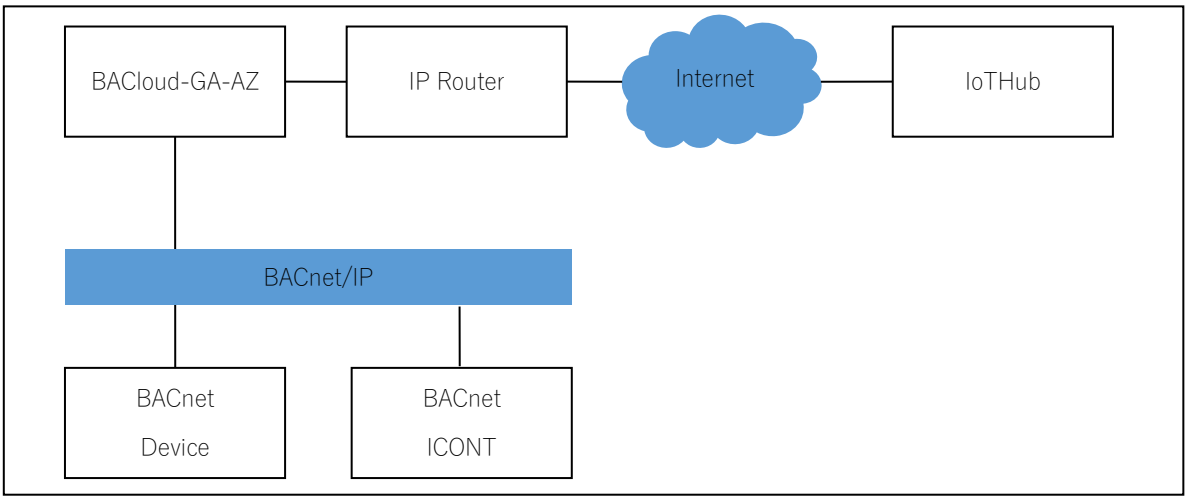
1. はじめに

本書は BACloud-GA-AZ のインタフェース仕様について記したものである。

2. 構成

2.1. システム構成

BACloud-GA-AZ は Windows10 を使用する（プラットホーム製　OpenBlocks IoT VX2/W)。



3. BACloud-GA-AZ の機能

3.1. 概要

3.1.1. 定期収集

ScanPoint ファイルで指定した BACnet デバイス内の BACnet オブジェクト（指定できるオブジェクトタイプは限定される）のプロパティ（このプロパティはオブジェクトタイプで決められている）の値を、指定された周期で BACnet サービス(ReadProperty、もしくは ReadPropertyMultiple)を使用して読み込み、Azure の IoTHub へテレメトリ通信を使用して送付する。

Azure 通信コストを削減するための機能として、BACnet から読み込んだデータをまとめて IoTHub へ送信する蓄積テレメトリ通信を行うように設定することができる。

3.1.2. COV 受信処理

ScanPoint ファイルで指定した BACnet デバイス内の BACnet オブジェクト（テレメトリ通信で使用可能なオブジェクトタイプに限定される）に BACnet サービス(SubscribeCOV)を指定された周期で送信する。BACnet デバイスはこのサービスを受け取ると、その指定されたオブジェクトの特定プロパティの値が変化すると COV を送信する。BACloud-GA-AZ はこの COV を受信すると IoTHub にテレメトリを送信する。

上記の SubscribeCOV 送信を設定していない BACnet デバイス内の BACnet オブジェクト（テレメトリ通信で使用可能なオブジェクトタイプに限定される）からの COV を受信した場合、COV 受信を指定するための CSV ファイル（COV 指定ファイル）に記述したものと一致すると IoTHub にテレメトリを送信する。

3.1.3. Event 受信処理

BACloud-GA-AZ が BACnet デバイス内の BACnet オブジェクト（テレメトリ通信で使用可能なオブジェクトタイプに限定される）からのイベントを受信した場合、その送信元が CSV ファイル（Event 指定ファイル）に記述されているものと一致すると IoTHub にテレメトリを送信する。

3.1.4. 発停・設定処理

ScanPoint ファイルで指定した BACnet デバイスに、スマホや PC から IoTHub 経由で発停情報や設定情報を書き込むことができる。これはダイレクトメソッドによる通信で実現される。

3.1.5. 遠隔設定編集機能

IoTHUB に接続するブラウザなどから BACnetGateway の各種設定情報を確認したり修正したりすることができる。これはダイレクトメソッドによる通信で実現される。

4. IoTHub とのインタフェース

4.1. テレメトリ通信

BACloud-GA-AZ はユーザが CSV ファイルで指定した BACnet デバイス内の BACnet オブジェクト（指定できるオブジェクトタイプは限定される）のプロパティ（このプロパティはオブジェクトタイプで決められている）の値を指定された周期で BACnet サービス(ReadProperty、もしくは ReadPropertyMultiple)を使用して読み込み Azure の IoTHub へテレメトリ通信を使用して送付する。

Azure 通信コストを削減するための機能として、CSV ファイルに Quick が用意されている。

CSV ファイルに Quick が指定されていると BACnet からデータを受け取ると直ちに IoTHub に送信する。

Quick が指定されていない場合、BACnet から読み込んだデータをまとめて IoTHub に送信する蓄積テレメトリ通信を行う。

BACnet から読み込んだデータは一旦 BACloud-GA-AZ 内の送信キューに蓄積される。送信キュー内の一番古いデータが送信キューに格納されてからパケットタイム時間(config ファイルの PacketTime にて設定)経過した場合、その時の送信キューに溜めこまれている全データが IoTHub に送信される。

また送信キューにたまっているデータの総バイト数がパケットサイズ(config ファイルの PacketSize にて設定)を超えるとその時の送信キューに溜めこまれている全データが IoTHub に送信される。

4.2. 発停・設定通信

スマホや PC から IoTHub 経由で BACloud-GA-AZ に接続している BACnet デバイスに発停情報や設定情報を書き込むことができる。この機能は IoTHub で規定されているダイレクトメソッド通信を使用する。

スマホや PC はサービス接続文字列で BACloud-GA-AZ を指定し、BACloud-GA-AZ に接続する BACnet デバイスとそのデバイス内のオブジェクト・プロパティ情報、発停データや設定情報は JSON データで指定する。

IoTHub 経由で BACloud-GA-AZ に伝えられた JSON データにより BACloud-GA-AZ は BACnet デバイスに WriteProperty を発行し、その実行結果をダイレクトメソッドの応答として IoTHub 経由でスマホや PC に伝える。

当初はテレメトリ通信で指定可能なオブジェクトタイプのプロパティに ReadProperty や WriteProperty を可能とするが、今後オブジェクトタイプとプロパティは拡張していくものとする（拡張予定は未定である)。

5. ファイル仕様

5.1. 設定ファイル

設定情報は AzureDevice 接続文字列ファイル、config ファイルおよび CSV ファイルで指定される。

以下の CSV ファイルは設定を動的に変更することができる。BACloud-GA-AZ はおよそ 60 秒周期で各 CSV ファイルのタイムスタンプを確認しており、CSV ファイルを更新すると、その変更が自動的に反映される。

- ScanPoint ファイル
- COV 指定ファイル
- Event 指定ファイル

5.1.1. AzureDevice 接続文字列ファイル

デバイス接続文字列を新規設定および変更する場合、"C:¥Users¥user¥Documents¥unitec¥conf"に本ファイルを格納する。本ファイルのファイル名は「DeviceCon.txt」とすること。BACloud-GA-AZ はこのファイルの存在を確認すると、指定された新しいデバイス接続文字列で IoTHub との接続を行う。

正常に接続できた場合、"C:¥Users¥user¥Documents¥unitec¥conf"から同ファイルを削除し、次回以降の接続のために同ファイルをユーザー非公開領域に保存する。

正常に接続できなかった場合、"C:¥Users¥user¥Documents¥unitec¥conf"から同ファイルは削除せず、IoTHub との通信を行わない状態となる。

フォーマット

```
{
        "DeviceConnection": デバイス接続文字列
}
```

5.1.2. Gateway 用 config ファイル仕様

Gateway 用 config ファイルのパスはプログラムが起動された場合の引数で指定される。引数で指定されない場合はプログラム実行ディレクトリに Gateway.conf という名前で格納されていなければならない。

太字のイタリック体の部分は設定されるデータであり、パラメータ表で規定されている。

フォーマット

```
{
        "IOTHub": {
                "PacketSize":          PacketSize
                ,"PacketTime":         PacketTime
                ,"System_id":          SystemId
                ,"Device_id":          DeviceId
                ,"RetryInterval":      RetryInterval
                ,"NumberOfRetries":    NumberOfRetries
                ,"Interface":          IoTHubInterface
                ,"IPAddress":          IoTHubIPAddress
                ,"SubNetMask":         IoTHubSubNetMask
                ,"DefaultGateway":     IoTHubDefaultGateway
                ,"DNSServer":          IoTHubDNSServer
        },
        "BACnet": {
                "IPAddress":           BACnetIPAddress
                ,"SubNetMask":         BACnetSubNetMask
                ,"PortNo":             PortNo
                ,"DeviceID":           DeviceID
                ,"PropertyCount":      PropertyCount
                ,"ReadInterval ":      ReadInterval
                ,"TimeSynchronizationEnable": TimeSynchronizationEnable
        },
        "File": {
                "ScanPointPath":       ScanPointPath
                ,"COVPath":            COVPath
                ,"EventPath":          EventPath
                ,"OutputFilePath":     OutputFilePath
                ,"RestartRecipients":  RecipientsPath
                ,"BACnetLogPath":      BACnetLogPath
                ,"IoTHubLogPath":      IoTHubLogPath
        }
}
```

パラメータ名	内容
IOTHub	未指定／エラー時：アプリケーション終了
PacketSize	送信すべきデータの最大バイト数 範囲：0～256000 未指定／エラー時：256000
PacketTime	送信を待たせる最大時間 (秒) 範囲：0～4294967295 未指定／エラー時：60
System_id	システムを一意に特定する文字列 未指定／エラー時："unSpecifiedSystemID"
Device_id	BACloud-GA-AZ の名称の文字列 未指定／エラー時："unSpecifiedDeviceID"
RetryInterval	IoTHub への送信のリトライを判断するまでの時間 (秒) 範囲：1～4294967295 未指定／エラー時：60
NumberOfRetries	IoTHub への送信をリトライする回数 範囲：0～2147483647 未指定／エラー時：2147483647
IoTHubInterface	IoTHub との通信に使用するインタフェースを指定する 範囲：以下の文字列のいずれか "Mobile", "Wi-Fi", "Ethernet" 未指定／エラー時：前回の通信設定を使用する。IPAddress, SubNetMask, DefaultGateway, DNSServer の値は使用されない。
IoTHubIPAddress	IoTHub との通信に使用する IP アドレスを指定する 範囲：以下の文字列のいずれか IP アドレス形式の文字列（例："192.168.0.1"）, "DHCP" 未指定／エラー時：前回の通信設定を使用する。Interface, SubNetMask, DefaultGateway, DNSServer は使用されない。 "DHCP"の場合、SubNetMask, DefaultGateway, DNSServer の値は使用されない。
IoTHubSubNetMask	IoTHub との通信に使用するサブネットマスクを指定する 範囲：サブネットマスク形式の文字列（例："255.255.255.0"） 未指定／エラー時：前回の通信設定を使用する。Interface, IPAddress, DefaultGateway, DNSServer は使用されない。
IoTHubDefault Gateway	IoTHub との通信に使用するデフォルトゲートウェイを指定する 範囲：IP アドレス形式の文字列（例："192.168.0.1"） 未指定／エラー時：前回の通信設定を使用する。Interface, IPAddress, SubNetMask, DNSServer は使用されない。
IoTHubDNSServer	IoTHub との通信に使用する DNS サーバーを指定する 範囲：IP アドレス形式の文字列（例："192.168.0.1"） 未指定／エラー時：前回の通信設定を使用する。Interface, IPAddress, SubNetMask, DefaultGateway は使用されない。
BACnet	未指定／エラー時：アプリケーション終了
BACnetIPAddress	BACloud-GA-AZ の BACnet デバイスとしての IP アドレス 未指定／エラー時：アプリケーション終了

	BACnetSubNetMask	BACloud-GA-AZ の BACnet デバイスとしてのサブネットマスク 未指定／エラー時：アプリケーション終了
	PortNo	BACloud-GA-AZ の BACnet デバイスとしてのポート番号 範囲：1～65535 未指定／エラー時：アプリケーション終了
	DeviceID	BACloud-GA-AZ の BACnet デバイスとしてのデバイスインスタンス番号 範囲：0～4194302 未指定／エラー時：アプリケーション終了
	PropertyCount	ReadPropertyMultiple にまとめるプロパティ数の最大値 範囲：0～4294967295 未指定／エラー時：20
	ReadInterval	ReadPropertyMultiple 送信後次の ReadPropertyMultiple を送信するまでに入れるディレー時間（ミリ秒） 範囲：0～4294967295 未指定／エラー時：10
	TimeSynchronization Enable	BACnet サービス（UTC）TimeSynchronization 受信時の時刻同期の可否 範囲：true または false（文字列ではなく、BOOL 型のパラメータ） 未指定／エラー時：false true の場合、OS の時刻同期機能は無効化され、BACnet サービスによる時刻の変更が行われる。 false の場合、OS の時刻同期機能が有効化され、BACnet サービスによる時刻同期は無視される。時刻同期有効化処理時、時刻同期を実行する。
File		未指定／エラー時：アプリケーション終了
	ScanPointPath	定周期スキャン情報を記述する CSV ファイルへのフルパス名 未指定／エラー時："C:¥Users¥user¥Documents¥unitec¥conf¥ScanPoint.csv"
	COVPath	COV を選別するための CSV ファイルへのフルパス名 未指定／エラー時："C:¥Users¥user¥Documents¥unitec¥conf¥COV.csv"
	EventPath	Event を選別するための CSV ファイルへのフルパス名 未指定／エラー時："C:¥Users¥user¥Documents¥unitec¥conf¥Event.csv"
	RecipientsPath	Restart 通告先のための CSV ファイルへのフルパス名 未指定／エラー時：BACnet 仕様に則った設定（ローカルブロードキャスト）
	OutputFilePath	定周期スキャン情報の送信データの保管用ファイルのフルパス名 試験時にデータを確認するための目的にのみ使用し、運用に入ったら、この項目は指定しないこと。ファイルサイズが巨大化する恐れがあるため。 未指定／エラー時：""（データを出力しない）
	BACnetLogPath	BACnet 通信ログファイルへのフルパス名 未指定／エラー時：""（ログを出力しない）
	IoTHubLogPath	IoTHub 通信ログファイルへのフルパス名 未指定／エラー時：""（ログを出力しない）

5.1.3. BACnet 用 config ファイル仕様

BACnet 用 config ファイルのパスはプログラムが起動された場合の引数で指定される。引数で指定されない場合はプログラム実行ディレクトリに BACnet.conf という名前で格納する。本ファイルが存在しない場合は、各パラメータは未指定時と同等の動作を行う。

太字のイタリック体の部分は設定されるデータであり、パラメータ表で規定されている。

フォーマット

```
{
        "WhoisSendInterval": WhoisSendInterval
        ,"WhoisRange": WhoisRange
}
```

パラメータ名	内容
WhoisSendInterval	Whois を送信する周期を秒単位で指定する。 この値が 0 ならば Whois は定周期送信しない。 範囲：0～120 未指定／エラー時：10
WhoisRange	WhoisRange は JSON の Array であり、その要素は以下のとおりである。 {"LowDeviceInstanceNo": *LowDeviceInstanceNo* ,"HighDeviceInstanceNo": *HighDeviceInstanceNo*} *LowDeviceInstanceNo*,*HighDeviceInstanceNo* は符号無整数（0 ~4194302） Low-High で 1 ペアであること。片方のみの場合、エラーとなる。 未指定／エラー時：範囲指定なしの Who-Is を通知する

5.1.4. ScanPoint ファイル仕様

このファイルは CSV 形式である。CSV ファイルの 1 行目はヘッダである。

列番号	項目	備考
1	BACnet デバイスインスタンス番号	範囲：0～4194302
2	BACnetObjecType	BACnet オブジェクトタイプ番号 0 AnalogInput 1 AnalogOutput 2 AnalogValue 3 BinaryInput 4 BinaryOutput 5 BinaryValue 13 MultiStateInput 14 MultiStateOutput 19 MultiStateValue 23 Accumulator 128 計量 130 電力デマンド監視 131 電力デマンド制御 132 発電機負荷制御
3	BACnetObject インスタンス番号	範囲：0～4194302
4	収集周期	秒単位(0 は収集しない) 範囲：0～4294967295
5	COVSubscribe 周期	秒単位(0 は subscribe しない) 範囲：0～2147483647 ※エラーの場合、60 秒周期でリトライする
6	Quick	1 ならば収集後直ちに Azure に書き込む、 0 ならば蓄積テレメトリ通信を行う 範囲：0 または 1
7	書き込み Priority	ダイレクトメソッドにおける WriteProperty の動作を制御 範囲：0～16 0：書き込み不可(収集のみ) 1～16: 書き込み時の優先順位を指定
8	Confirmed	SubscribeCOV サービスの Confirmed パラメタ 範囲：0 または 1 0;Unconfirmed 1:Confirmed
9	予約領域	0 固定

オブジェクトタイプにより収集するプロパティが決められている。

BACnetObjecType			収集するプロパティ
AnalogInput, MultiStateInput, Accumulator, 電力デマンド制御,	AnalogOutput, MultiStateOutput, 計量, 発電機負荷制御	AnalogValue MultiStateValue 電力デマンド監視	PresentValue StatusFlags
BinaryInput,	BinaryOutput,	BinaryValue	PresentValue StatusFlags ElapsedActiveTime ChangeOfStateCount

5.1.5. COV 指定ファイル仕様

このファイルは CSV 形式である。CSV ファイルの 1 行目はヘッダである。

COV を受信した場合、このファイルに指定されている BACnet デバイスのオブジェクトならば IoTHub の蓄積ではないテレメトリ送信を行う、そうでなければその COV は廃棄する。

列番号	項目	備考
1	BACnet デバイスインスタンス番号	範囲：0～4194302
2	BACnetObjectType	BACnet オブジェクトタイプ番号 0 AnalogInput 1 AnalogOutput 2 AnalogValue 3 BinaryInput 4 BinaryOutput 5 BinaryValue 13 MultiStateInput 14 MultiStateOutput 19 MultiStateValue 23 Accumulator 128 計量 130 電力デマンド監視 131 電力デマンド制御 132 発電機負荷制御
3	BACnetObject インスタンス番号	範囲：0～4194302

5.1.6. Event 指定ファイル

このファイルは CSV 形式である。CSV ファイルの 1 行目はヘッダである。

Event を受信した場合、このファイルに指定されている BACnet デバイスのオブジェクトならば IoTHub の蓄積ではないテレメトリ送信を行う。そうでなければその Event は廃棄する

列番号	項目	備考
1	BACnet デバイスインスタンス番号	範囲：0～4194302
2	BACnetObjectType	BACnet オブジェクトタイプ番号 0 AnalogInput 1 AnalogOutput 2 AnalogValue 3 BinaryInput 4 BinaryOutput 5 BinaryValue 13 MultiStateInput 14 MultiStateOutput 19 MultiStateValue 23 Accumulator 128 計量 130 電力デマンド監視 131 電力デマンド制御 132 発電機負荷制御
3	BACnetObject インスタンス番号	範囲：0～4194302

5.1.7. Recipients 指定ファイル

このファイルは CSV 形式である。CSV ファイルの 1 行目はヘッダである。

BACloud-GA-AZ の RestartNotification に使用される宛先を指定する。

列番号	項目	備考
1	通告先 BACnet デバイスのネットワーク番号	0 固定
2	通告先 BACnet デバイスの IP アドレス	IP アドレス形式の文字列 (例："192.168.1.1")
3	通告先 BACnet デバイスのポート番号	範囲：0～65535 0 は Config ファイルで指定した BACloud-GA-AZ と同じポート番号を使用する

5.2. エラー仕様

設定ファイルのエラー情報は、起動時のコンソール出力で確認することができる。

5.2.1. config ファイル

(1) ファイルが JSON 形式ではない場合

コンソール画面にエラー情報が出力される。

フォーマット

```
Read Conffile Error err=<例外データ>
```

(2) データエラー

下記のような場合、コンソール画面にエラー情報が出力される

- 必要なデータが不足している
- 指定されたデータが範囲外である
- 指定されたデータが想定外の形式である

フォーマット

```
ReadInifile error <例外データ>
```

5.2.2. AzureDevice 接続文字列ファイル

JSON 形式ではない、もしくはデータが不足しているか、データ形式が不正な場合、コンソール画面にエラー情報が出力される。指定された接続文字列が間違っている場合、エラー情報は出力されない。

上記のようなエラーが起きた場合、BACloud-GA-AZ は動作を開始するが、IoTHub との通信は行われない。

フォーマット

```
readDeviceConnection err=<例外データ>
```

5.2.3. CSV ファイル

正しく読み込めたレコードのみ、その情報がコンソール画面に出力される。エラーとなったレコードはコンソール画面に出力されない。

5.3. 通信ログファイル

config ファイルの「OutputFilePath」および「BACnetLogPath」、「IoTHubLogPath」で指定するファイルに出力されるログファイルの使用を記す。

5.3.1. ファイル名

config ファイルで指定されたファイル名に識別情報を付加する。

config ファイルで拡張子が指定された場合、その拡張子を使用する。拡張子が指定されなかった場合、「.txt」を使用する。なお、指定された拡張子に応じてファイル内容が変化することは無い。

フォーマット

<ファイル名>_<年(4 桁)><月(2 桁)><日(2 桁)><時(2 桁)><分(2 桁)><秒(2 桁)>.<拡張子>

例　config ファイルで「BACnetLog.txt」を指定した場合

BACnetLog_20201106123456.txt

5.3.2. ファイルサイズとファイル数

ログ毎に、以下のファイルサイズ、ファイル数のログ情報を保持し、それ以上のログファイルを出力する場合には最も古いサイズを削除する。

項目	内容
ファイルサイズ	10MB
ファイル数	150

5.3.3. ファイルフォーマット

UTF-8 のテキストファイルとして出力する。

(1) BACnetLogPath

レコードフォーマット

<タイムスタンプ>,<PDU タイプ>,<送受信方向>,<アドレス>,<InvokeID>,<デバイス>,<PDU タイプ別情報>CRLF

項目	内容
タイムスタンプ	<年(4 桁)><月(2 桁)><日(2 桁)><時(2 桁)><分(2 桁)><秒(2 桁)>
PDU タイプ	ReadProperty 等の名称
送受信方向	Send　　Recv
アドレス	送信時の宛先、または受信時の送信元の IP アドレスとポート番号 例：192.168.1.2:47808
InvokeID	サービスの Invoke ID 範囲：0～255 UnconfirmedRequest 等一部のサービスタイプの場合は存在しないため省略される
デバイス	Device=<インスタンス番号> デバイスが特定できない場合は省略される（デバイスを指定しないブロードキャスト送信をした場合や未認識のデバイスから初めてのサービスを受信した場合など）
PDU タイプ別情報	● 正常時、以下の情報をカンマ区切りで記述する ➢ オブジェクト ID (同一オブジェクトはセミコロン区切りで記述する) ✧ プロパティ ID ✧ 存在すれば、プロパティ値、Priority、ArrayIndex ● 異常時 ➢ 発生した事象やエラーに関する文字列 非サポートの通信は PDU タイプ別情報に"No data"または"No data (segmented message)"を出力する。

サポートするアプリケーションサービス

ReadProperty	ReadPropertyMultiple
WriteProperty	WritePropertyMultiple
ConfirmedCOVNotification	UnconfirmedCOVNotification
ConfirmedEventNotification	UnconfirmedEventNotification
SubscribeCOV	WhoIs
WhoHas	IAm
IHave	SimpleACK
ComplexACK (ReadProperty)	ComplexACK (ReadPropertyMultiple)
SegmentACK	Abort
Error	Reject

(2) IoTHubPath

レコードフォーマット

<タイムスタンプ>,<コマンド>,<送受信方向>,<コマンド別情報>CRLF

項目	内容
タイムスタンプ	<年(4 桁)><月(2 桁)><日(2 桁)><時(2 桁)><分(2 桁)><秒(2 桁)>
コマンド	Write　COV　Event ReadProperty　WriteProperty TelemetryStatus　DirectMethodStatus
送受信方向	Send　Recv
コマンド別情報	● 正常時 ➢ IoTHub と送受信した JSON 形式文字列 ● 異常時、またはコマンドが Telemetry あるいは DirectMethodStatus の場合 ➢ 発生した事象やエラーに関する文字列 ➢ IoTHub と送受信した JSON 形式文字列（無い場合もある）

コマンドの「TelemetryStatus」および「DirectMethodStatus」は BACloud-GA-AZ と IoTHub との接続状態を表すコマンドであり、IoTHub とのデータの送受信を伴わない。また、コマンド別情報に「Status」および「Reason」情報が出力された場合、詳細は以下を参照すること。

- Status
  https://docs.microsoft.com/ja-jp/dotnet/api/microsoft.azure.devices.client.connectionstatus?view=azure-dotnet
- Reason
  https://docs.microsoft.com/ja-jp/dotnet/api/microsoft.azure.devices.client.connectionstatuschangereason?view=azure-dotnet

(3) OutputFilePath

レコードフォーマット

<タイムスタンプ>:<テレメトリ>CRLF

項目	内容
タイムスタンプ	<年(4 桁)>/<月(2 桁)>/<日(2 桁)> <時(2 桁)>:<分(2 桁)>:<秒(2 桁)>
テレメトリ	IoTHub に送信した JSON 形式文字列

6. 通信データ仕様

太字のイタリック体の部分は「7.JSON データ仕様」で規定される定義に置き換えられる。

6.1. テレメトリデータ仕様

正常時のテレメトリデータ仕様を記す。

エラーが発生した場合、テレメトリデータの一部がエラー情報に置き換えられる。詳細は「6.1.4BACnet サービスエラー時のテレメトリデータ」を参照すること。

フォーマット

```
{
        "TimeStamp": "yyyy-mm-ddThh:MM:SS.ffffff+zz:zz"形式の文字列
        ,"BACnetDevice": 符号無整数 (0..4194302)
                                                -- BACnet デバイスのインスタンス番
号
        ,"BACnetObject": {
                "_base": "BACnetObjectIdentifier"
                ,"_value": {
                        "ObjectType": 符号無整数 (0..1023)
                        ,"InstanceNo": 符号無整数 (0..4194302)
                }
        }
        ,"Properties": {
                "PresentValue": 単精度不動小数点数、文字列または符号無整数       -- *1
                                        -- Analog Input/Output/Value の場合
                                                -- 単精度不動小数点数または文字列
                                        -- それ以外のオブジェクトタイプの場合
                                                -- 符号無整数 (0..)
                ,"PulseRate": 符号無整数 (0..4294967295)                        -- *1
                ,"StatusFlags": {
                        "_base": "BACnetStatusFlags"
                        ,"_value": [bit0,bit1,bit2,bit3]
                                                -- bit0 (0..1)       in-alarm
                                                -- bit1 (0..1)       fault
                                                -- bit2 (0..1)       overridden
                                                -- bit3 (0..1)       out-of-service
                }
                ,"ElapsedActiveTime": 符号無整数 (0..4294967295)                -- *2
                ,"ChangeOfStateCount": 符号無整数 (0..4294967295)               -- *2
                ,"EventState": 符号無整数 (0..5)                                -- *3
```

```
                                                    -- normal          (0)
                                                    -- fault                (1)
                                                    -- offnormal            (2)
                                                    -- high-limit           (3)
                                                    -- low-limit            (4)
                                                    -- life-safety-alarm (5)
                ,"EventTimeStamp": "yyyy-mm-ddThh:MM:SS.ffffff+zz:zz"形式の文字列
                                        または符号無整数 (0..65535)          -- *3
        }
}
```

*1 Event のテレメトリかつ ObjectType が Accumulator (23) の場合のみ、"PresentValue"が無く、"PulseRate"が含まれる。
その他の場合は"PresentValue"のみが含まれる。

*2 BACnetObject の ObjectType が以下のタイプかつ ScanPoint のテレメトリの場合のみ付加される。

- BinaryInput (2)
- BinaryOutput (3)
- BinaryValue (4)

*3 Event のテレメトリの場合のみ付加される。日付情報が存在しない場合"yyyy-mm-dd"部分が"****-**-**"に置き換えられる。Event 送信元 BACnet デバイスが時刻を取り扱う機能を持たない場合、符号無整数が用いられる。

6.1.1. ScanPoint データ仕様

フォーマット

```
{
        "Command": "Write"
        ,"System_id": 文字列        -- config ファイルで規定する System_id
        ,"Device_id": 文字列        -- config ファイルで規定する Device_id
        ,"ValueString": [
                テレメトリデータ
                ,…
        ]
}
```

ValueString はテレメトリデータ (6.1 テレメトリデータ仕様) を要素に持つ配列である。

データを収集するとテレメトリデータが配列に付加される。そのため通信総データバイト数が更新される。Azure 通信を効率よく行うために、この総データが一定値(config ファイルの PacketSize)を超えないように蓄積し、送信回数を減らすことにより、Azure 通信コストを削減させることができる。

また、収集データが少ない場合 PacketSize までに達する時間が長くかかりすぎて Azure でのデータ更新が遅くなりすぎる可能性があるため、未送信のデータが送信されずに蓄積されてもよい最大秒数（config ファイルの PacketTime）経過すると PacketSize に到達しなくても送信される。

例


```
{
        "Command": "Write",
        "System_id": "sample-system-id",
        "Device_id": "sample-device-id",
        "ValueString": [
                {
                        "TimeStamp": "2020-10-14T15:35:28.123225+09:00",
                        "BACnetDevice": 2,
                        "BACnetObject": {
                                "_base": "BACnetObjectIdentifier ",
                                "_value": {
                                        "ObjectType": 0,
                                        "InstanceNo": 0
                                }
                        },
                        "Properties": {
                                "PresentValue": 28.0,
                                "StatusFlags": {
                                        "_base": "BACnetStatusFlags",
                                        "_value": [0,0,0,0]
                                }
                        }
                }
        ]
}
```

6.1.2. COV データ仕様

フォーマット

```
{
        "Command": "COV"
        ,"System_id": 文字列        -- config ファイルで規定する System_id
        ,"Device_id": 文字列        -- config ファイルで規定する Device_id
        ,"ValueString": [
                テレメトリデータ
        ]
}
```

例

```
{
        "Command": "COV",
        "System_id": "sample-system-id",
        "Device_id": "sample-device-id",
        "ValueString": [
                {
                        "TimeStamp": "2020-10-14T17:04:48.748167+09:00",
                        "BACnetDevice": 2,
                        "BACnetObject": {
                                "_base": "BACnetObjectIdentifier",
                                "_value": {
                                        "ObjectType": 0,
                                        "InstanceNo": 0
                                }
                        },
                        "Properties": {
                                "PresentValue": 0,
                                "StatusFlags": {
                                        "_base": "BACnetStatusFlags",
                                        "_value": [0,0,0,0]
                                }
                        }
                }
        ]
}
```

6.1.3. Event データ仕様

フォーマット

```
{
        "Command": "EVENT"
        ,"System_id": 文字列        -- config ファイルで規定する System_id
        ,"Device_id": 文字列        -- config ファイルで規定する Device_id
        ,"ValueString": [
                テレメトリデータ
        ]
}
```

例

```
{
  "Command": "EVENT",
  "System_id": "sample-system-id",
  "Device_id": "sample-device-id",
  "ValueString": [
        {
          "TimeStamp": "2020-10-14T17:08:59.324053+09:00",
          "BACnetDevice": 2,
          "BACnetObject": {
                "_base": "BACnetObjectIdentifier",
                "_value": {
                  "ObjectType": 0,
                  "InstanceNo": 1
                }
          },
          "Properties": {
                "PresentValue": 5,
                "StatusFlags": {
                  "_base": "BACnetStatusFlags",
                  "_value": [0,0,0,0]
                },
                "EventTimeStamp": "2020-10-14T17:09:02.460000+09:00",
                "EventState": 0
          }
        }
  ]
}
```

6.1.4. BACnet サービスエラー時のテレメトリデータ

BACnet サービスにエラーが発生した場合、「6.1 テレメトリデータ仕様」に示すテレメトリデータの"Properties"の値が以下のように置き換えられる。

(1) BACnet サービスがエラーになった場合

"Properties"の値には"PresentValue"キーとその値のみが含まれる。"PresentValue"の値にはエラー情報に置き換えられる。

"result"の値は、AbortPDU 受信時に"Abort"、RejectPDU 受信時に"Reject"、その他の場合に"Error"となる。

フォーマット

```
"Properties": {
        "PresentValue": {
                "result": 文字列
                , "_value": BACnetAbortReason, BACnetRejectReason または BACnetError
                                -- "result"が"Abort"の場合   BACnetAbortReason
                                -- "result"が"Reject"の場合  BACnetRejectReason
                                -- その他の場合                BACnetError
        }
}
```

例　ReadPropertyMultiple サービスが非サポート等の理由により拒否された場合

```
"Properties": {
  "PresentValue": {
        "result": "Reject",
        "_value": {"_base": "RejectReason","_value": 9}
  }
}
```

例　宛先 BACnet デバイスが発見できず、BACnet サービスを送信できなかった場合

```
"Properties": {
  "PresentValue": {
        "result": "Error",
        "_value": {"error-class": 7, "error-code": 70}
  }
}
```

(2) BACnet サービスの応答に部分的なエラーが含まれていた場合

"Properties"の値に含まれるエラーが発生したプロパティは、その値がエラー情報に置き換えられる。

フォーマット

```
"Properties": {
        "<プロパティ名称>": {
                "result": "Error"
                , "_value": BACnetError
        }
        , ...
}
```

例　PresentValue と StatusFlags を読み出し、前者が正常、後者がエラーの場合

```
"Properties": {
        "PresentValue": 1,
        "StatusFlags": {
                "result": "Error",
                "_value": {"error-class": 2, "error-code": 27}
        }
}
```

6.2. ダイレクトメソッド通信仕様

ダイレクトメソッドにはメソッド名とメソッドデータが指定される。

メソッド名の一覧を以下に示す。

メソッド名	内容	備考
WriteProperty	指定された BACnetDevice のオブジェクトのプロパティに指定されたデータを書き込む。	
ReadProperty	指定された BACnetDevice のオブジェクトのプロパティのデータを読み込む。	
WritePropertyMultiple	将来用	
TimeSynchronization	将来用	
DeviceRestart	将来用	
addList	将来用	
removeList	将来用	

サポートするオブジェクトタイプ、プロパティ及びそのデータ型を以下に示す。

オブジェクトタイプ	プロパティ	データ型
AnalogInput AnalogOutput AnalogValue	PresentValue	REAL
	StatusFlags	BACnetStatusFlags
BinaryInput BinaryValue	PresentValue	BACnetBinaryPV
	StatusFlags	BACnetStatusFlags
BinaryOutput	PresentValue	BACnetBinaryPV
	FeedbackValue	
	StatusFlags	BACnetStatusFlags
MultiStateInput MultiStateValue Accumulator 計量 電力デマンド監視 電力デマンド制御 発電機負荷制御	PresentValue	Unsigned
	StatusFlags	BACnetStatusFlags
MultiStateOutput	PresentValue	Unsigned
	FeedbackValue	
	StatusFlags	BACnetStatusFlags

ダイレクトメソッドの device-identifier で指定する BACnet デバイスは ScanPoint ファイルに記述されている BACnet デバイスでなければならない。その BACnet デバイスに対して Scan する必要がない場合でも ScanPoint ファイルに記述しておかなければならない。

例　ダイレクトメソッドだけで指定するデバイスインスタンス番号 1001 を設定する

1001,0,0,0,0,0,8,0,0

6.2.1. WriteProperty

(1) メソッドデータ

フォーマット

BACnetDeviceObjectPropertyValue

例　「7.2.2.BACnetDeviceObjectPropertyValue」参照

"priority"が指定されていない場合、ScanPointFile の書き込み Priority が BACnet の WritProperty サービスに使用される。

(2) 動作

BACnetDeviceObjectPropertyValue で指定される BACnetDevice（device-identifier）の BACnetObject（object-identifier）のプロパティ（property-identifier）に property-value で示される値を書き込む。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) AbortPDU 受信時

フォーマット

```
{
        "result": "Abort"
        , "_value": BACnetAbortReason
}
```

(c) RejectPDU 受信時

フォーマット

```
{
        "result": "Reject"
        , "_value": BACnetRejectReason
}
```

(d) ErrorPDU 受信時

フォーマット

```
{
        "result": "Error"
        , "_value": BACnetError
}
```

(e)ダイレクトメソッド　WriteProperty のパラメータが不正の場合

フォーマット

```
{
        "result":"WriteProperty parameter error"
}
```

6.2.2. ReadProperty

(1) メソッドデータ

フォーマット

BACnetDeviceObjectPropertyReference

例　「7.2.1.BACnetDeviceObjectPropertyReference」参照

(2) 動作

BACnetDeviceObjectPropertyReference で指定される BACnetDevice（device-identifier）の BACnetObject（object-identifier）のプロパティ（property-identifier）に対して ReadProperty を実行しその結果を戻す。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
        ,"_value": BACnetDeviceObjectPropertyValue
}
```

(b) エラー時

「6.2.1(3)リターンデータ」の(b), (c), (d)を参照。

(c)ダイレクトメソッド　ReadProperty のパラメータが不正の場合

フォーマット

```
{
        "result":"Read Property parameter error"
}
```

6.3. 設定用ダイレクトメソッド

ダイレクトメソッドにはメソッド名とメソッドデータが指定される。

設定用メソッド名の一覧を以下に示す。

メソッド名	内容	備考
WriteConfig	付属するデータで Config ファイルを更新する。	
ReadConfig	現在の Config ファイルを読み込む。	
ChangeConfig	現在の Config ファイルの一部を変更する	
WriteBACnetConfig	付属するデータで BACnetConfig ファイルを更新する。	
ReadBACnetConfig	現在の BACnetConfig ファイルを読み込む。	
ChangeBACnetConfig	現在の BACnetConfig ファイルの一部を変更する	
WriteScanPoint	付属するデータで ScanPoint ファイルを更新する。	
ReadScanPoint	ScanPointCSV ファイルを読み込む	
AddScanPoint	ScanPointCSV ファイルに付属するレコードデータを追加する。	
RemoveScanPoint	ScanPointCSV ファイルから指定されたレコードデータを削除する。	
WriteCOV	付属するデータで COVCSV ファイルを更新する。	
ReadCOV	COVCSV ファイルを読み込む	
AddCOV	COVCSV ファイルに付属するレコードデータを追加する。	
RemoveCOV	COVCSV ファイルから指定されたレコードデータを削除する。	
WriteEvent	付属するデータで EventCSV ファイルを更新する。	
ReadEvent	EventCSV ファイルを読み込む	
AddEvent	EventCSV ファイルに付属するレコードデータを追加する。	
RemoveEvent	EventCSV ファイルから指定されたレコードデータを削除する。	
WriteRecipients	付属するデータで RecipientsCSV ファイルを更新する。	
ReadRecipients	RecipientsCSV ファイルを読み込む	
AddRecipients	RecipientsCSV ファイルに付属するレコードデータを追加する。	
RemoveRecipients	RecipientsCSV ファイルから指定されたレコードデータを削除する。	
ReadLog	Log ファイルを読み込む。	
CheckLog	Log ファイルを調べる	
RemoveLog	Log ファイルを削除する。	
RemoveLogAll	指定された LogType の Log ファイルをすべて削除する	

※行末記号は CRLF とする。

6.3.1. WriteConfig

(1) メソッドデータ

フォーマット

config ファイル中身の文字列

例


```
{
	"IOTHub":{
		"PacketSize":32000,
		"PacketTime":60,
		"System_id":"System_id",
		"Device_id":"209AA-A_0009",
		"RetryInterval":60,
		"NumberOfRetries":2147483647,
		"Interface":"Ethernet",
		"IPAddress":"192.168.1.254",
		"SubNetMask":"255.255.255.0",
		"DefaultGateway":"192.168.1.1",
		"DNSServer":"8.8.8.8"
	},
	"BACnet":{
		"IPAddress":"192.168.0.254",
		"SubNetMask":"255.255.255.0",
		"PortNo":47808,
		"DeviceID":254,
		"PropertyCount":40,
		"ReadInterval":10
,		"TimeSynchronizationEnable":true
	},
	"File":{

	"ScanPointPath":"c:¥¥Users¥¥user¥¥Documents¥¥unitec¥¥conf¥¥ScanPoint.csv",
		"COVPath":"c:¥¥Users¥¥user¥¥Documents¥¥unitec¥¥conf¥¥COV.csv",
		"EventPath":"c:¥¥Users¥¥user¥¥Documents¥¥unitec¥¥conf¥¥Event.csv",

	"RestartRecipients":"c:¥¥Users¥¥user¥¥Documents¥¥unitec¥¥conf¥¥RestartRecipients.
csv",
```

```
                "BACnetLogPath":"c:¥¥Users¥¥user¥¥Documents¥¥unitec¥¥BACnetLog.txt",
                "IoTHubLogPath":"c:¥¥Users¥¥user¥¥Documents¥¥unitec¥¥IoTHub.txt"
        }
}
```

ここで IOTHub、BACnet、File のことをセクション名と称す。

PacketSize などセクションの下の JSON オブジェクトのプロパティ名をパラメータ名と称す。

その下の 32000,60 などの値をパラメータ値と称す。

(2) 動作

受信したデータで config ファイルを書き換える。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.2. ReadConfig

(1) メソッドデータ

なし

(2) 動作

config ファイルを読み出し IoTHUB に戻す。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
        ,"_value": “Config ファイルの中身”
}
```

ファイルの中に特殊文字含まれる場合は、¥“や¥¥のように JSON 文字列規約に従ってエスケープされたデータが戻される。

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.3. ChangeConfig

(1) メソッドデータ

フォーマット

```
{
        “SectionName“:”セクション名
        ,“ParameterName”:”パラメータ名”
        ,"ParemeterValue": パラメータ値
}
```

(2) 動作

受信したデータで config ファイルの指定されたセクションのパラメータ部分のみ書き換える。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.4. WriteBACnetConfig

(1) メソッドデータ

フォーマット

BACnet 用 config ファイル中身の文字列

例

```
{
        " WhoisSendInterval":60
        ,"WhoisRange":[
                {”LowDeviceInstanceNo”: 1,”HighDeviceInstanceNo”: 5}
                ,{”LowDeviceInstanceNo”: 11,”HighDeviceInstanceNo”: 15}
                ]
}
```

WhoisSendInterval と *LowDeviceInstanceNo, HighDeviceInstanceNo* は符号無整数

(2) 動作

受信したデータで config ファイルを書き換える。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.5. ReadBACnetConfig

(1) メソッドデータ

なし

(2) 動作

BACnet 用 config ファイルを読み出し IoTHUB に戻す。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
        ,"_value": “BACnetConfig ファイルの中身”
}
```

ファイルの中に特殊文字含まれる場合は、¥“や¥¥のように JSON 文字列規約に従ってエスケープされたデータが戻される。

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.6. ChangeBACnetConfig

(1) メソッドデータ

フォーマット

```
{
        "ParameterName":"パラメータ名"
        ,"ParemeterValue":パラメータ値
}
```

(2) 動作

受信したデータで config ファイルの指定されたセクションのパラメータ部分のみ書き換える。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.7. WriteScanPoint

(1) メソッドデータ

フォーマット

```
タイトルレコード
CSV ファイルのレコード 1
CSV ファイルのレコード 2
CSV ファイルのレコード 3
CSV ファイルのレコード 4
    ・・・・・・・・・
CSV ファイルのレコード N
```

上記はレコード数が N の場合例である。

例

Device,ObjectType,IntanceNo,Period,SubscribePeriod,QuickSend,Priority,Confirmed

2,0,0,2,10,0,0,1 ――レコード 1

2,0,1,2,10,0,0,1

2,0,2,2,10,0,0,1

2,0,3,2,10,0,0,1

2,0,4,2,10,0,0,1

2,0,5,2,10,0,0,1

2,0,6,2,10,0,0,1

2,0,7,2,10,0,0,1

(2) 動作

受信したデータで ScanPoint.csv ファイルを書き換える。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.8. ReadScanPoint

(1) メソッドデータ

フォーマット

```
{
        "Start": 開始行番号
        ,"Count": 読み込み行数
}
```

CSV ファイルの先頭行は 0 番とする。

(2) 動作

ScanPoint.csv ファイルの開始行番号から読み込み行数で示される個数のみ読み出し IoTHUB に戻す。実際の行数より Count が大きい場合、実際の行数分のみ戻す。

(3) リターンデータ

(a) 正常時

フォーマット（開始行番号（S＝１００）から（X＝５）個読み込んだ場合）

```
{
```

```
          "result": "OK"
          ,"_value":[
                    "行番号 S（＝100）の行の CSV データ"
                    ・・・・・・・・・
                    ,"行番号(S+X-1＝104)の行の CSV データ"
          ]
}
```

例

```
{
        result:"OK"
        ,"_value":[
                  "2,0,0,2,10,0,0,1" ――レコード 1
                  ,"2,0,1,2,10,0,0,1"
                  ,"2,0,2,2,10,0,0,1"
                  ,"2,0,3,2,10,0,0,1"
                  ,"2,0,4,2,10,0,0,1"
                  ,"2,0,5,2,10,0,0,1"
                  ,"2,0,6,2,10,0,0,1"
                  ,"2,0,7,2,10,0,0,1"
        ]
}
```

(b) 異常時

フォーマット

```
{
          "result": "Error"
          ,"_value": “エラー情報”
}
```

Start と Count で指定された行の全部もしくは一部が存在しない場合はエラーとなる。

6.3.9. AddScanPoint

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

2,0,0,2,10,0,0,1

(2) 動作

受信したデータを ScanPoint.csv ファイルの最後部に追加する。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.10. RemoveScanPoint

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

2,0,0,2,10,0,0,1

(2) 動作

受信したデータと同じレコードを持つ行を ScanPoint.csv ファイルから削除する。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.11. WriteCOV

(1) メソッドデータ

フォーマット

```
タイトルレコード
CSV ファイルのレコード 1
CSV ファイルのレコード 2
CSV ファイルのレコード 3
CSV ファイルのレコード 4
    ・・・・・・・・
CSV ファイルのレコード N
```

上記はレコード数が N の場合例である。

例

```
Device,ObjectType,IntanceNo
2,0,0   ――レコード 1
2,0,1
2,0,2
2,0,3
2,0,4
2,0,5
2,0,6
2,0,7
```

(2) 動作

受信したデータで COV.csv ファイルを書き換える。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.12. ReadCOV

(1) メソッドデータ

フォーマット

```
{
        "Start": 開始行番号
        ,"Count": 読み込み行数
}
```

CSV ファイルの先頭行は 0 番とする。

(2) 動作

COV.csv ファイルの開始行番号から読み込み行数で示される個数のみ読み出し IoTHUB に戻す。実際の行数より Count が大きい場合、実際の行数分のみ戻す。

(3) リターンデータ

(a) 正常時

フォーマット（開始行番号（S＝１００）から（X＝５）個読み込んだ場合）

```
{
        "result": "OK"
        ,"_value":[
                "行番号 S（＝100）の行の CSV データ"
                ・・・・・・・・
                ,"行番号(S+X-1＝104)の行の CSV データ"
```

```
        ]
}
```

例

```
{
        result:"OK"
        ,"_value":[
                "2,0,0"    ――レコード 1
                ,"2,0,1"
                ,"2,0,2"
                ,"2,0,3"
                ・・・・
                ,"2,0,7"
        ]
}
```

(b) 異常時

フォーマット

```
{
        "result":"Error"
        ,"_value": "エラー情報"
}
```

Start と Count で指定された行の全部もしくは一部が存在しない場合はエラーとなる。

6.3.13. AddCOV

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

2,0,1

(2) 動作

受信したデータを COV.csv ファイルの最後部に追加する。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.14. RemoveCOV

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

2,0,1

(2) 動作

受信したデータと同じレコードを持つ行を COV.csv ファイルから削除する。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.15. WriteEvent

(1) メソッドデータ

フォーマット

```
タイトルレコード
CSV ファイルのレコード 1
CSV ファイルのレコード 2
CSV ファイルのレコード 3
CSV ファイルのレコード 4
    ・・・・・・・・・
CSV ファイルのレコード N
```

上記はレコード数が N の場合例である。

例

Device,ObjectType,IntanceNo
2,0,0 ――レコード 1
2,0,1
2,0,2
2,0,3
2,0,4
2,0,5
2,0,6
2,0,7

(2) 動作

受信したデータで Event.csv ファイルを書き換える。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
```

```
        ,"_value": “エラー情報”
}
```

6.3.16. ReadEvent

(1) メソッドデータ

フォーマット

```
{
        "Start": 開始行番号
        ,"Count": 読み込み行数
}
```

CSV ファイルの先頭行は 0 番とする。

(2) 動作

Event.csv ファイルの開始行番号から読み込み行数で示される個数のみ読み出し IoTHUB に戻す。実際の行数より Count が大きい場合、実際の行数分のみ戻す。

(3) リターンデータ

(a) 正常時

フォーマット（開始行番号（S＝１００）から（X＝５）個読み込んだ場合）

```
{
        "result": "OK"
        ,"_value":[
                “行番号 S（＝100）の行の CSV データ”
                ・・・・・・・・・
                ,“行番号(S+X-1＝104)の行の CSV データ”
        ]
}
```

例

```
{
        result:"OK"
        ,"_value":[
                "2,0,0"　――レコード 1
                ,"2,0,1"
                ,"2,0,2"
                ,"2,0,3"
                ・・・・
```

```
                ,"2,0,7"
        ]
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": "エラー情報"
}
```

Start と Count で指定された行の全部もしくは一部が存在しない場合はエラーとなる。

6.3.17. AddEvent

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

2,0,1

(2) 動作

受信したデータを Event.csv ファイルの最後部に追加する。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.18. RemoveEvent

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

2,0,1

(2) 動作

受信したデータと同じレコードを持つ行を Event.csv ファイルから削除する。およそ 60 秒後 Gateway プログラムは書き換えた内容で実行開始する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.19. WriteRecipients

(1) メソッドデータ

フォーマット

```
タイトルレコード
CSV ファイルのレコード 1
CSV ファイルのレコード 2
CSV ファイルのレコード 3
CSV ファイルのレコード 4
    ・・・・・・・・・
CSV ファイルのレコード N
```

上記はレコード数が N の場合例である。

例

NetworkNo,IPAddress,PortNo

0,"192.168.1.2",47808　——レコード 1

1,"192.168.1.3",47808

2,"192.168.1.4",47808

(2) 動作

受信したデータで RestartRecipients.csv ファイルを書き換える。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

6.3.20. ReadRecipients

(1) メソッドデータ

フォーマット

```
{
        "Start": 開始行番号
        ,"Count": 読み込み行数
}
```

CSV ファイルの先頭行は 0 番とする。

(2) 動作

RestartRecipients.csv ファイルの開始行番号から読み込み行数で示される個数のみ読み出し IoTHUB に戻す。実際の行数より Count が大きい場合、実際の行数分のみ戻す。

(3) リターンデータ

(a) 正常時

フォーマット（開始行番号（S＝１）から（X＝3）個読み込んだ場合）

```
{
        "result": "OK"
        ,"_value": [
                "行番号 S（＝1）の行の CSV データ"
                ・・・・・・・・・
                ,"行番号(S+X-1＝3)の行の CSV データ"
        ]
}
```

ファイルの中に特殊文字含まれる場合は、¥“や¥¥のように JSON 文字列規約に従ってエスケープされたデータが戻される。

例

```
{
        result:"OK"
        ,"_value":[
                ,"0,￥"192.168.1.2￥",47808"        ――レコード 1
                ,"1,￥"192.168.1.3￥",47808"
                ,"2,￥"192.168.1.4￥",47808"
        ]
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": “エラー情報”
}
```

Start と Count で指定された行の全部もしくは一部が存在しない場合はエラーとなる。

6.3.21. AddRecipients

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

0,"192.168.1.2",47808

(2) 動作

受信したデータを RestartRecipients.csv ファイルの最後部に追加する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": "エラー情報"
}
```

6.3.22. RemoveRecipients

(1) メソッドデータ

フォーマット

CSV ファイル 1 レコード

例

0,"192.168.1.2",47808

(2) 動作

受信したデータと同じレコードを持つ行を RestartRecipients.csv ファイルから削除する。

(3) リターンデータ

(a) 正常時

フォーマット

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value":"エラー情報"
}
```

6.3.23. ReadLog

(1) メソッドデータ

フォーマット

```
{
        "Choice":choice
        ,"Start":start
        ,"Count":RecordCount
        ,"FileName":"FileName"
}
```

Choice は”StartLineNo”を指定する。

Start は先頭行番号を符号なし整数で指定する。省略すると 0 行目が指定される。

RecordCount は読み込むレコード数を指定する。ただし、ダイレクトメソッドの 1 回分の最大サイズに制限される。

FileName は読み込むファイル名をフルパスで指定する。この名前は CheckLog で取得可能である。

(2) 動作

FileName で指定された Log ファイルを対象として、Start で指定した先頭レコードから Count レコード数分を送り返す。実際の行数より Count が大きい場合、実際の行数分のみ戻す。

(3) リターンデータ

(a) 正常時

```
{
        "result": "OK"
        ,"RecordNo": RecordNo…///読み出された先頭レコード番号（ファイル内で 0 から開始する符号なし整数）
        ,"_value":[
                "Log レコード 1 "
```

```
                ・・・・・・・・
                ,"Log レコード N"
        ]
}
```

ファイルの中に特殊文字含まれる場合は、¥“や¥¥のように JSON 文字列規約に従ってエスケープされたデータが戻される。

例

```
{
        result:"OK"
        ,"RecordNo":5
        ,"_value":[
                ,"xxxxxxxxxxxxxx" ――レコード 1
                ・・・・・・・・・・・・
                ,"xxxxxxxxxxxxxx" ――レコード N
        ]
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value": "エラー情報"
}
```

6.3.24. CheckLog

(1) メソッドデータ

フォーマット

```
{
        "Type":LogType
}
```

LogType は文字列であり以下の 2 種類を指定可能

LogType	内容	備考
"IOTHub"	IoTHUB との通信ログ	
”BACnet"	BACnet 通信ログ	

(2) 動作

LogType で指定されたログファイルの名称を送り戻す。

(3) リターンデータ

(a) 正常時

指定された LogType のログファイルが N 個存在した場合は以下のようになる。

```
{
        "result": "OK"
        ,"_value":[
                "ログファイル名 1 "
                ・・・・・・・・・
                ,"ログファイル名 N"
        ]
}
```

ファイルの中に特殊文字含まれる場合は、¥“や¥¥のように JSON 文字列規約に従ってエスケープされたデータが戻される。

例

```
{
        result:"OK"
        ,"_value":[
                "IOTHubLog1.txt"          ――レコード 1
                ・・・・・・・・
                ,"IOTHubLogN.txt"
        ]
}
```

(b) 異常時

フォーマット

```
{
        "result":"Error"
        ,"_value":"エラー情報"
}
```

6.3.25. RemoveLog

(1) メソッドデータ

フォーマット

```
{
        "Type":LogType
        ,"FileName": "FileName"
}
```

LogType は文字列であり以下の 2 種類を指定可能

LogType	内容	備考
"IOTHub"	IoTHUB との通信ログ	
"BACnet"	BACnet 通信ログ	

FileName は削除するファイル名をフルパスで指定する。この名前は CheckLog で取得可能である。

(2) 動作

FileName で指定された Log ファイルを削除する。

(3) リターンデータ

(a) 正常時

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result":"Error"
        ,"_value": "エラー情報"
}
```

6.3.26. RemoveLogAll

(1) メソッドデータ

フォーマット

```
{
        "Type":LogType
}
```

LogType は文字列であり以下の 3 種類を指定可能

LogType	内容	備考
"IOTHub"	IoTHUB との通信ログ	
"BACnet"	BACnet 通信ログ	

(2) 動作

LogType で指定された Log ファイルをすべて削除する。

(3) リターンデータ

(a) 正常時

```
{
        "result": "OK"
}
```

(b) 異常時

フォーマット

```
{
        "result": "Error"
        ,"_value":"エラー情報"
}
```

7. JSON データ仕様

本節で太字の斜体は Primirive または Base データ型を意味し、実際にはその型で規定される JSON データで置き換えられる。

7. 1. PrimitiveData の JSON 仕様

PrimitiveData は、本章に規定されている通りに展開される。

7.1.1. Unsigned

フォーマット

```
{
        "_base": "Unsigned"
        ,"_value": 符号無整数 (0..)
}
```

例　1234 の場合

```
{
        "_base": "Unsigned",
        "_value": 1234
}
```

7.1.2. REAL

フォーマット

```
{
        "_base": "REAL"
        ,"_value": 単精度浮動小数点数または文字列              -- *1
}
```

*1　　特殊な数は下記のように記述する
Positiveinfinity は"∞"、negativeinfinity は"-∞"、非数は"NaN"の文字列となる。
なお、-0 は 0 として扱われる。+∞は∞として扱われる。

例　-12.345 の場合

```
{
        "_base": "REAL",
        "_value": -12.345
}
```

7.1.3. BACnetStatusFlags

フォーマット

```
{
        "_base": "StatusFlags"
        ,"_value": ビット配列または文字列
                                -- ビット配列表現の場合　[Bit0,Bit1,Bit2,Bit3]
                                        -- Bit0 (0..1)      in-alarm
                                        -- Bit1 (0..1)      fault
                                        -- Bit2 (0..1)      overridden
                                        -- Bit3 (0..1)      out-of-service
                                -- 文字列表現の場合
                                        -- Bit0             "InAlarm"
                                        -- Bit1             "Fault"
                                        -- Bit2             "OverRidden"
                                        -- Bit3             "OutOfService"
}
```

例　ビット配列表現　in-alarm のみ true、fault, overridden, out-of-service は false

```
{
        "_base": "StatusFlags",
        "_value": [1,0,0,0]
}
```

例　文字列表現　in-alarm と out-of-service は true、fault と overridden は false

```
{
        "_base": "StatusFlags",
        "_value": "InAlarm;OutOfService"
}
```

7.1.4. BACnetAbortReason

フォーマット

```
{
        "_base": "AbortReason"
        ,"_value": 符号無整数 (0..11)
                                -- other                                  (0)
                                -- buffer-overflow                (1)
                                -- invalid-apdu-in-this-state     (2)
                                -- preempted-by-higher-priority-task (3)
                                -- segmentation-not-supported           (4)
                                -- security-error                 (5)
                                -- insufficient-security                (6)
                                -- window-size-out-of-range       (7)
                                -- application-exceeded-reply-time (8)
                                -- out-of-resources               (9)
                                -- tsm-timeout                          (10)
                                -- apdu-too-long                        (11)
}
```

例　buffer-overflow の場合

```
{
        "_base": "AbortReason",
        "_value": 1
}
```

7.1.5. BACnetBinaryPV

フォーマット

```
{
        "_base": "BinaryPV"
        ,"_value":符号無整数 (0..1) または文字列
                                -- 数値表現の場合
                                        -- inactive       (0)
                                        -- active  (1)
                                -- 文字列表現の場合
                                        -- inactive       "Inactive"
                                        -- active  "Active"
}
```

例　数値表現　active (1)の場合

```
{
        "_base": "BinaryPV",
        "_value": 1
}
```

例　文字列表現　active の場合

```
{
        "_base": "BinaryPV",
        "_value": "Active"
}
```

7.1.6. BACnetPropertyIdentifier

フォーマット

```
{
        "_base": "PropertyIdentifier"
        ,"_value": 符号無整数 (0..491)
                                        -- ANSI/ASHRAE 135-2016 の 21 章に記載されている
                                        -- BACnetPropertyIdentifier を参照
}
```

例　PresentValue (85)の場合

```
{
        "_base": "PropertyIdentifier",
        "_value": 85
}
```

7.1.7. BACnetRejectReason

フォーマット

```
{
        "_base": "RejectReason"
        ,"_value": 符号無整数 (0..9) -- other                              (0)
                                        -- buffer-overflow                 (1)
                                        -- inconsistent-parameters         (2)
                                        -- invalid-parameter-data-type     (3)
                                        -- invalid-tag                     (4)
                                        -- missing-required-parameter      (5)
                                        -- parameter-out-of-range          (6)
                                        -- too-many-arguments              (7)
                                        -- undefined-enumeration           (8)
                                        -- unrecognized-service            (9)
}
```

例　unrecognized-service の場合

```
{
        "_base": "RejectReason",
        "_value": 9
}
```

7.1.8. BACnetObjectIdentifier

フォーマット

```
{
        "_base": "ObjectIdentifier"
        ,"_value": {
                "object-type": 符号無整数 (0..1023)
                ,"instance-number": 符号無整数 (0..4194303)
        }
}
```

例　AnalogOutput-123

```
{
        "_base": "ObjectIdentifier",
        "_value": {
                "object-type":1,
                "instance-number":123
        }"
}
```

7. 2. BaseTypes データの JSON 仕様

BaseTypes は、本章に規定されている通りに展開される。

7.2.1. BACnetDeviceObjectPropertyReference

フォーマット

```
{
        "_base": "DeviceObjectPropertyReference"
        ,"_value": {
                "object-identifier": BACnetObjectIdentifier
                ,"property-identifier": BACnetPropertyIdentifier
                ,"property-array-index": 符号無整数
                                        -- オプションであり存在しない場合もある
                ,"device-identifier": BACnetObjectIdentifier
        }
}
```

例　Device-2 の AO-3 の PresentValue を指定する

```
{
        "_base":"DeviceObjectPropertyReference",
        "_value": {
                "device-identifier": {
                        "_base":"ObjectIdentifier",
                        "_value": {
                                "object-type": 8,
                                "instance-number": 2
                        }
                },
                "object-identifier": {
                        "_base":"ObjectIdentifier",
                        "_value": {
                                "object-type": 1,
                                "instance-number": 3
                        }
                },
                "property-identifier": {
                        "_base":"PropertyIdentifier",
                        "_value":85
                }
        }
}
```

7.2.2. BACnetDeviceObjectPropertyValue

フォーマット

```
{
        "_base": "DeviceObjectPropertyValue"
        ,"_value": {
                "device-identifier": BACnetObjectIdentifier
                ,"object-identifier": BACnetObjectIdentifier
                ,"property-identifier": BACnetPropertyIdentifier
                ,"property-array-index": 符号無整数
                                        -- オプションであり存在しない場合もある
                ,"property-value": object-identifier と property-identifier と
                                        property-array-index で取りうるデータ型
                ,"priority": 符号無整数 (1..16)
                                        -- オプションであり存在しない場合もある
        }
}
```

例　Device-2 の AO-0 の PresentValue に 123.456 を指定する

```
{
        "_base": "DeviceObjectPropertyValue",
        "_value": {
                "device-identifier": {
                        "_base": "ObjectIdentifier",
                        "_value": {
                                "object-type": 8,
                                "instance-number": 2
                        }
                },
                "object-identifier": {
                        "_base": "ObjectIdentifier",
                        "_value": {
                                "object-type": 1,
                                "instance-number": 0
                        }
                },
                "property-identifier": {
                        "_base": "PropertyIdentifier",
                        "_value": 85
                },
                "property-value": {
                        "_base": "REAL",
                        "_value": 123.456
                },
                "priority": 8
        }
}
```

7.2.3. BACnetError

フォーマット

```
{
        "error-class": 符号無整数 (0..7)          -- device (0)
                                                  -- object (1)
                                                  -- property (2)
                                                  -- resources (3)
                                                  -- security (4)
                                                  -- services (5)
                                                  -- vt (6)
                                                  -- communication (7)
        ,"error-code": 符号無整数 (0..138)
                                                  -- ANSI/ASHRAE 135-2016 の 21 章
                                                  -- に記載されている error-code 参照
}
```

例　error-class が object、error-code が unknown-object の場合

```
{
        "error-class": 1,
        "error-code": 31
}
```

8. BACnet 仕様

BACnet 仕様を記載する。

8.1. BACnet 規格

ANSI/ASHRAE Standard 135-2016

8.2. デバイスプロファイル

B-GENERAL（他に該当するプロファイルが無いデバイスのプロファイル）

8.3. データリンク

BACnet/IP

8.4. サポート BIBB

BIBB 区分	サポート内容	備考
Data Sharing	ReadProperty-A	DS-RP-A
	ReadProperty-B	DS-RP-B
	ReadPropertyMultiple-A	DS-RPM-A
	ReadPropertyMultiple-B	DS-RPM-B
	WriteProperty-A	DS-WP-A
	WriteProperty-B	DS-WP-B
	WritePropertyMultiple-B	DS-WPM-B
	Change Of Value-A	DS-COV-A
	Change Of Value Unsubscribed-A	DS-COVU-A
Alarm and Event	Notification-A	AE-N-A
Device Management	Dynamic Device Binding-A	DM-DDB-A
	Dynamic Device Binding-B	DM-DDB-B
	Dynamic Object Binding-B	DM-DOB-B
	TimeSynchronization-B	DM-TS-B
	UTCTimeSynchronization-B	DM-UTC-B
	Restart-B	DM-R-B

8.5. サポートオブジェクトとプロパティ

サポートオブジェクトおよびプロパティを記載する。

8.5.1. サポートオブジェクト

下表のオブジェクトをサポートする。これ以外の BACnet オブジェクトは持たず、作成もできない。

オブジェクトタイプ	数量	備考
Device	1	インスタンス番号は Config ファイルで設定
NetworkPort	1	インスタンス番号は 1 で固定

8.5.2. サポートプロパティ

(1) Device オブジェクトタイプ

ID	プロパティ名	R/W	プロパティ値
10	APDU_Segment_Timeout	R/W	初期値：6000 範囲：Unsigned 型の範囲 ※　Number_Of_APDU_Retries が 0 の時のみ、0 が許される
11	APDU_Timeout	R/W	初期値：6000 範囲：Unsigned 型の範囲 ※　Number_Of_APDU_Retries が 0 の時のみ、0 が許される
12	Application_Software_Version	R	※バージョンにより異なる
24	Daylight_Savings_Status	R	初期値：false
28	Description	R/W	初期値："" 範囲：CharacterString 型の範囲
30	Device_Address_Binding	R	※BACnet デバイスの認識状況により異なる
44	Firmware_Revision	R	※ Application_Software_Version と同じ
56	Local_Date	R	※日時により異なる
57	Local_Time	R	※日時により異なる
58	Location	R/W	初期値："" 範囲：CharacterString 型の範囲
62	Max_APDU_Length_Accepted	R/W	初期値：1024

			範囲：50～1476
70	Model_Name	R	"BACloud-GA-AZ" 固定
73	Number_Of_APDU_Retries	R/W	3 範囲：Unsigned 型の範囲
75	Object_Identifier	R	Device-X ※インスタンス番号は Config ファイルで設定
76	Object_List	R	Device-X NetworkPort-1 ※Device のインスタンス番号は Config ファイルで設定
77	Object_Name	R/W	初期値：" Device-X" ※インスタンス番号は Config ファイルで設定 範囲：1～255 文字の文字列
79	Object_Type	R	8 (device) 固定
96	Protocol_Object_Types_Supported	R	以下のみ true で固定 (60 ビット) Device NetworkPort
97	Protocol_Services_Supported	R	以下のみ true で固定 (44 ビット) ReadProperty ReadPropertyMultiple WriteProperty WritePropertyMultiple ConfirmedCOVNotification UnconfirmedCOVNotification ConfirmedEventNotification UnconfirmedEventNotification Who-Is Who-Has I-Am TimeSynchronization UTCTimeSynchronization
98	Protocol_Version	R	1 固定
107	Segmentation_Supported	R/W	0 (segmented-both) 範囲：0～3
112	System_Status	R	0 (operational) 固定
119	UTC_Offset	R/W	初期値：-540 範囲：-1440～1440
120	Vendor_Identifier	R	154

121	Vendor_Name	R	初期値："Unitec corporation"
139	Protocol_Revision	R	19 固定
155	Database_Revision	R	0
167	Max_Segments_Accepted	R	初期値：64 範囲：Unsigned 型の範囲
168	Profile_Name	R/W	初期値："" 範囲：CharacterString 型の範囲
196	Last_Restart_Reason	R	1 (coldstart) 固定
202	Restart_Notification_Recipients	R	※RestartNotification ファイルの指定により異なる ※ファイルが指定されなかったりファイルにアクセスできなかったりした場合、BACnet 仕様に則った初期値（ローカルブロードキャスト）が設定される ※指定されたファイルに有効なレコードが含まれなかった場合、要素無しに設定される
203	Time_Of_Device_Restart	R	※日時により異なる
371	Property_List	R	※75, 77, 79, 371 を除く、本表のプロパティ ID 固定

(2) NetworkPort オブジェクトタイプ

ID	プロパティ名	R/W	プロパティ値
28	Description	R/W	初期値："" 範囲：CharacterString 型の範囲
75	Object_Identifier	R	NP-1 固定
77	Object_Name	R/W	初期値："NetworkPort-1" 範囲：1～255 文字の文字列
79	Object_Type	R	56 (network-port) 固定
81	Out_Of_Service	R	0 (false) 固定
103	Reliability	R	0 (no-fault-detected)
111	Status_Flags	R	[0,0,0,0] (全て false)
371	Property_List	R	※75, 77, 79, 371 を除く、本表のプロパティ ID
399	APDU_Length	R	1476 固定
400	IP_Address	R	※Config ファイルおよび通信イン

			タフェースで設定
401	IP_Default_Gateway	R	0x00000000 固定
402	IP_DHCP_Enable	R	0 (false) 固定
403	IP_DHCP_Lease_Time	R	0 固定
404	IP_DHCP_Lease_Time_Remaining	R	0 固定
405	IP_DHCP_Server	R	0x00000000 固定
406	IP_DNS_Server	R	0x00000000 固定
408	BACnet_IP_Mode	R	0 (normal) 固定
409	BACnet_IP_Multicast_Address	R	0x00000000 固定
411	IP_Subnet_Mask	R	※Config ファイルおよび通信インタフェースで設定
412	BACnet_IP_UDP_Port	R	※Config ファイルで設定
416	Changes_Pending	R	0 (false) 固定
420	Link_Speed	R	0 固定 (unknown を意味する)
423	MAC_Address	R	※ IP_Address と BACnet_IP_UDP_Port を組み合わせた 6 オクテット
425	Network_Number	R	0
426	Network_Number_Quality	R	0 (unknown)
427	Network_Type	R	5 (ipv4) 固定
482	Protocol_Level	R	2 (bacnet-application) 固定

8.5.3. プロパティの範囲制限

(1) Unsigned 型

0～4294967295（プロパティ固有の範囲が規定されていない場合）

(2) CharacterString 型

CharacterSet＝0 (ISO 10646 = UTF-8)のみサポート
文字数：0～255 文字（プロパティ固有の範囲が規定されていない場合）