

CANoe.Connectivity

IoTアプリケーションをシミュレーション・テスト

Agenda

▶ 概要

CANoe.Connectivityの設定

プロトコル別の対応機能

CANoe.Connectivityとは (1)

CANoe.Connectivityとは……

- ▶ コネクティビティアプリケーションを解析・シミュレーション・テスト
- ▶ ローカル環境・クラウド環境に対応

対応プロトコル……

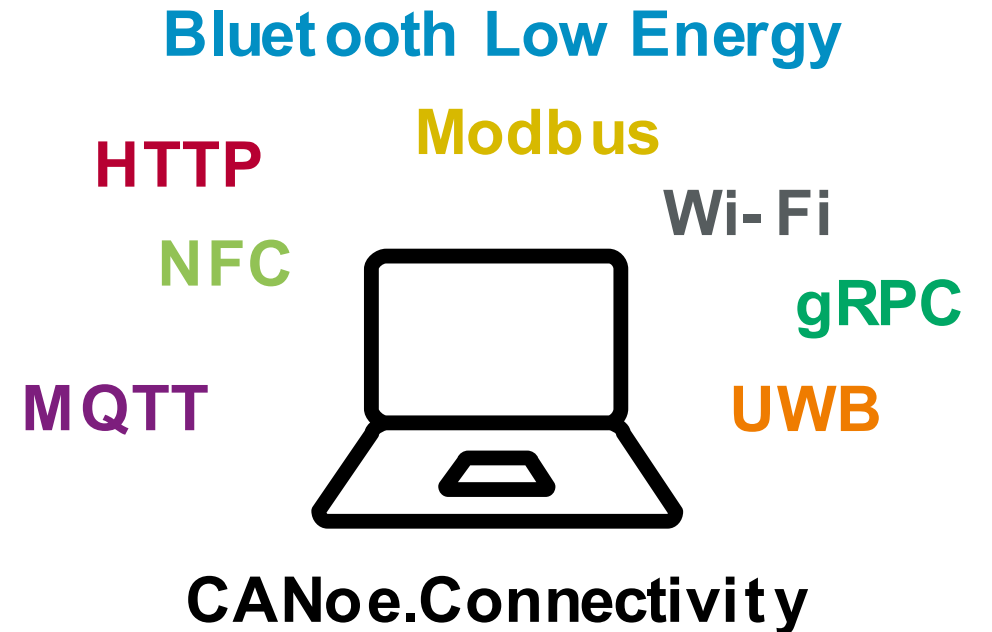
- ▶ 有線通信……MQTT、HTTP、Modbus、gRPC
- ▶ 無線通信……BLE、Wi-Fi、NFC、UWB (VH4110が必要)

対象分野……

- ▶ インダストリー4.0
- ▶ 医療・輸送システム
- ▶ 自動車 (クラウド、インフォテイメント、テレマティクス)

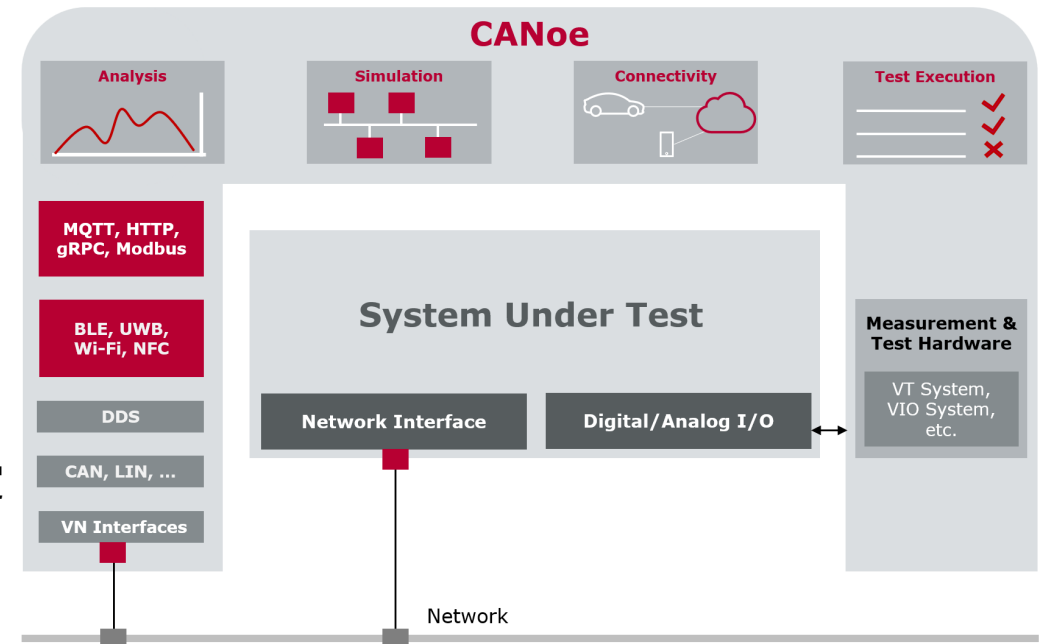
機能……

- ▶ コネクティビティアプリケーションのシミュレーション・テスト
- ▶ 多数の解析機能



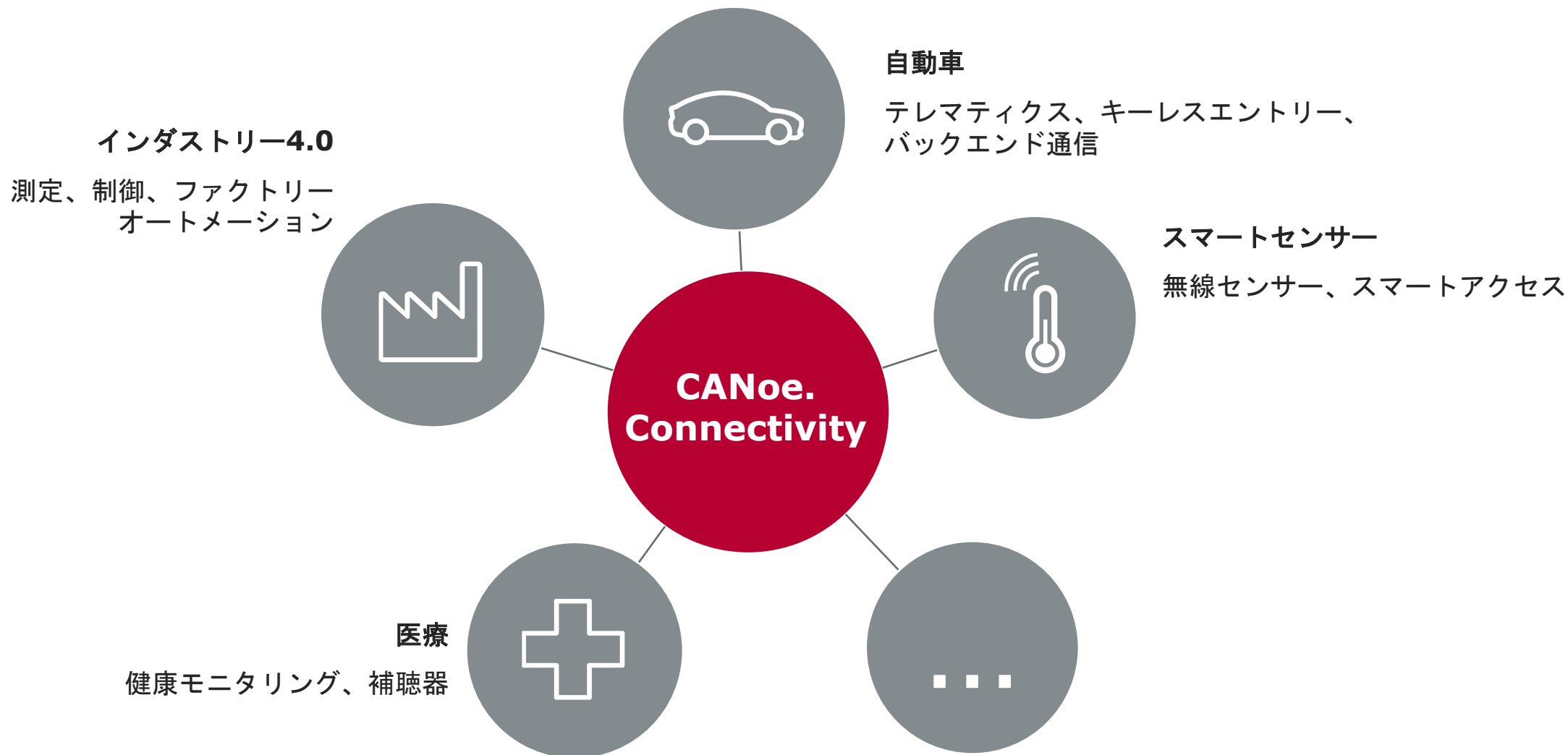
CANoe.Connectivityとは (2)

- ▶ 1つのツールでコネクティビティ・IoTアプリケーションの解析・シミュレーション・テストに対応
- ▶ テスト・シミュレーション用APIとして、C#、Python、CAPL^{A)}を用意
 - ▶ APIを使用して、開発サイクルを自動化
- ▶ vTESTstudio^{B)}による開発支援
- ▶ インタラクティブなシミュレーション環境
- ▶ 通信内容を可視化
- ▶ テキストベースでデータフォーマット・通信形式を記述・設定
 - ▶ VS Codeに対応 (補完・構文チェック)



- A) CANoe上で通信シミュレーションやテストロジックを記述する専用スクリプト言語
- B) CAPLやテストケースをGUIベースで効率的に作成・管理できるテスト開発環境

CANoe.Connectivityの対象分野



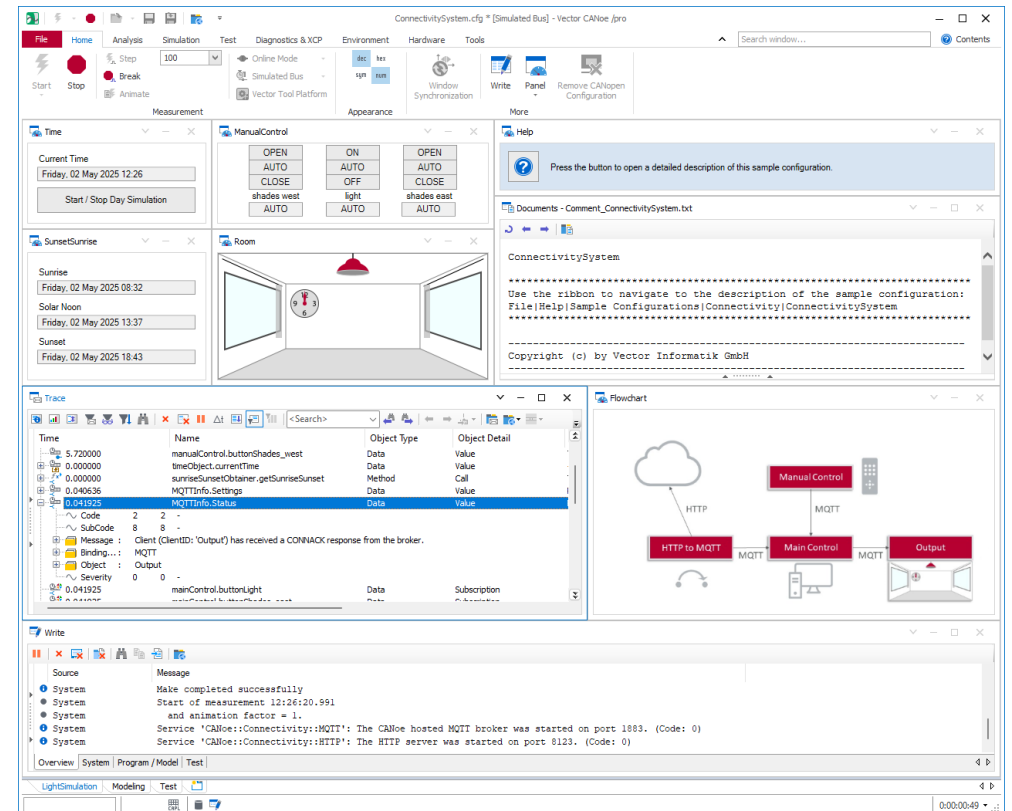
CANoe.Connectivityの対象分野詳細(1) – 有線通信

有線通信のシミュレーションとテスト……

- ▶ コネクティビティ・テレマティクスアプリケーションのテスト
 - ▶ 機能・性能テスト
 - ▶ フロントエンド/バックエンドのシミュレーション

- ▶ 産業用制御システムのテスト
 - ▶ 機能・性能テスト
 - ▶ クライアント/サーバー/ゲートウェイのシミュレーション

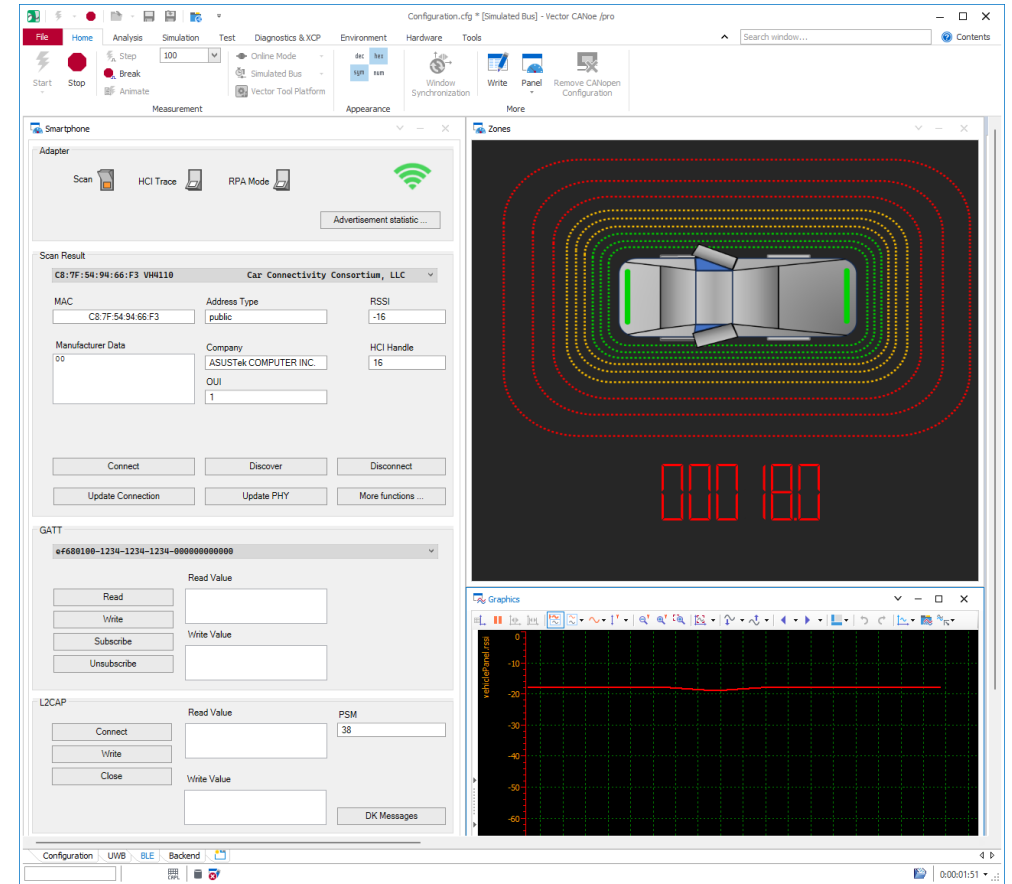
- ▶ RESTベースインターフェースとの統合
 - ▶ バックエンドシステムとの連携
 - ▶ サードパーティハードウェアとの統合



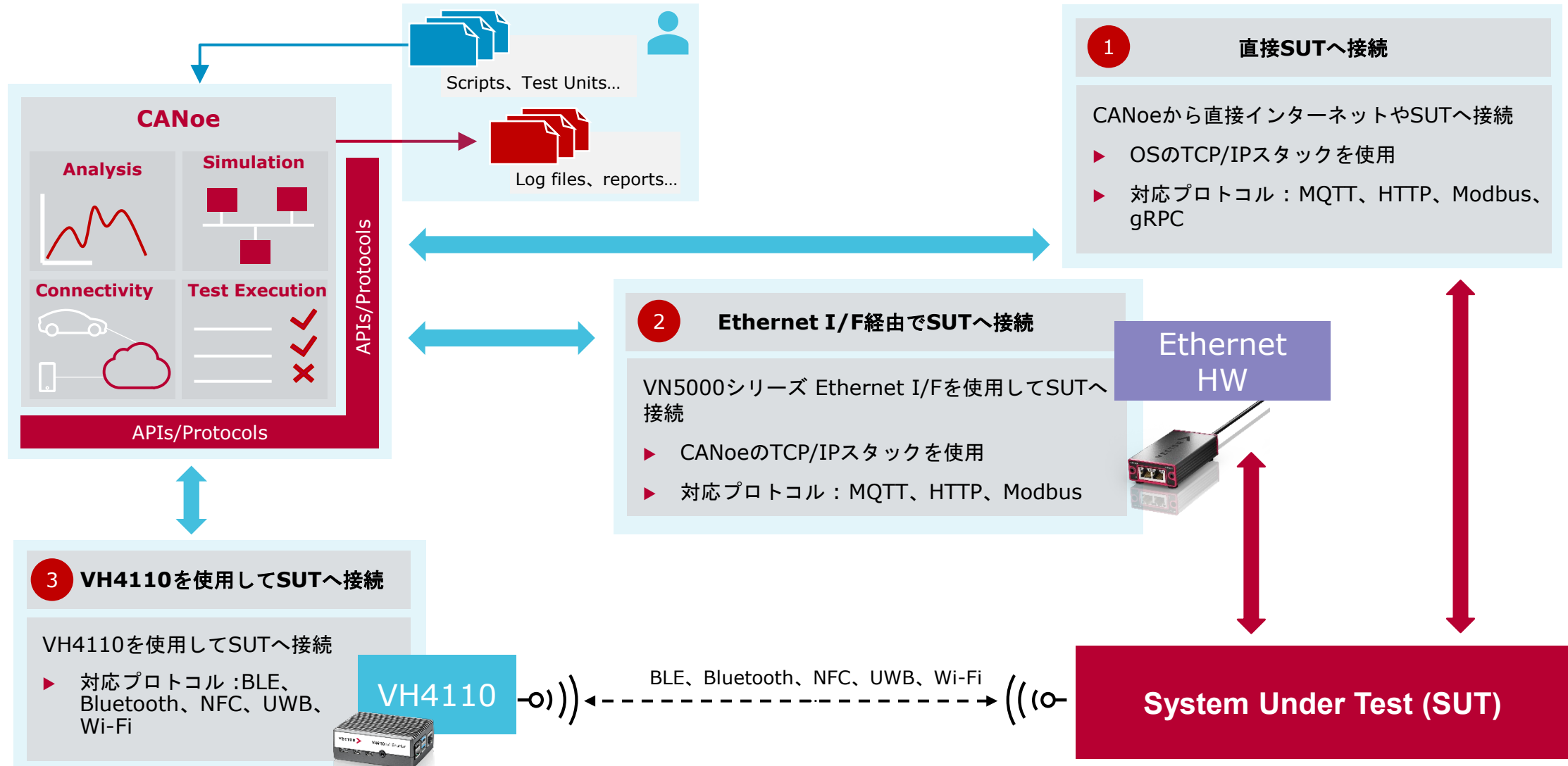
CANoe.Connectivityの対象分野詳細(2) - 無線通信

VH4110を使用した、無線通信のシミュレーションとテスト……

- ▶ スマートアクセス・キーレスゴーシステムのテスト
 - ▶ 車両やキーのシミュレーション
 - ▶ CCC(Car Connectivity Consortium)規格に準拠したテスト
- ▶ インフォテインメントECUのテスト
 - ▶ スマートフォンやタブレットのシミュレーション
 - ▶ メーカー固有パラメータのテスト
- ▶ インダストリー4.0 アプリケーションのテスト
 - ▶ 無線センサーやアクチュエータのテスト



CANoe.Connectivityを使用したSUTへの接続例



Agenda

概要

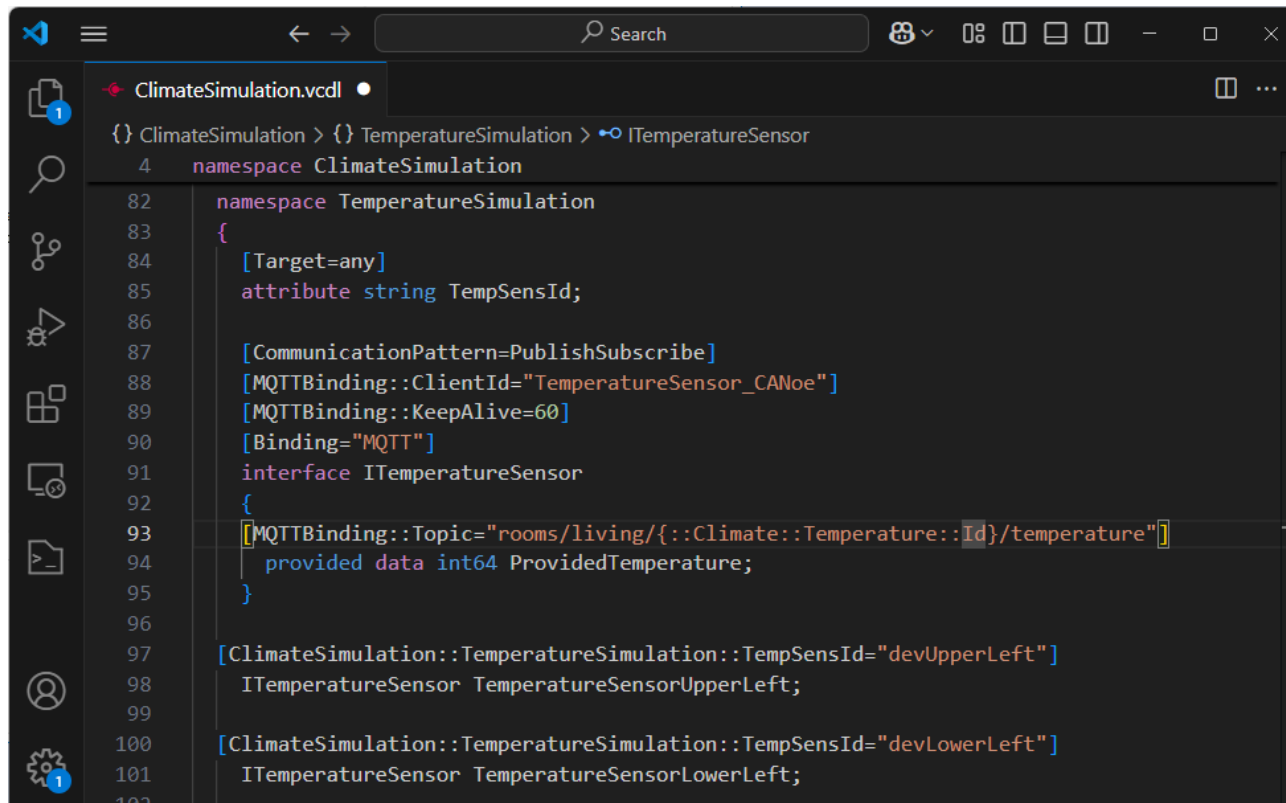
▶ **CANoe.Connectivity**の設定

プロトコル別の対応機能

CANoe.Connectivityの設定方法

- ▶ 通信内容は**vCDL**※で設定
 - ▶ データ型・通信パラメータを定義
 - ▶ VS Code対応（構文チェック、コード補完）

- ▶ CI/CDパイプラインや、バージョン管理システムとの統合に対応可能



```
ClimateSimulation.vcdl
{} ClimateSimulation > {} TemperatureSimulation > ITemperatureSensor
4 namespace ClimateSimulation
82 namespace TemperatureSimulation
83 {
84 [Target=any]
85 attribute string TempSensId;
86
87 [CommunicationPattern=PublishSubscribe]
88 [MQTTBinding::ClientId="TemperatureSensor_CANoe"]
89 [MQTTBinding::KeepAlive=60]
90 [Binding="MQTT"]
91 interface ITemperatureSensor
92 {
93 [MQTTBinding::Topic="rooms/living/{::Climate::Temperature::Id}/temperature"]
94 provided data int64 ProvidedTemperature;
95 }
96
97 [ClimateSimulation::TemperatureSimulation::TempSensId="devUpperLeft"]
98 ITemperatureSensor TemperatureSensorUpperLeft;
99
100 [ClimateSimulation::TemperatureSimulation::TempSensId="devLowerLeft"]
101 ITemperatureSensor TemperatureSensorLowerLeft;
102
```

※ サービス指向通信システム向けのドメイン固有言語(DSL)

Agenda

概要

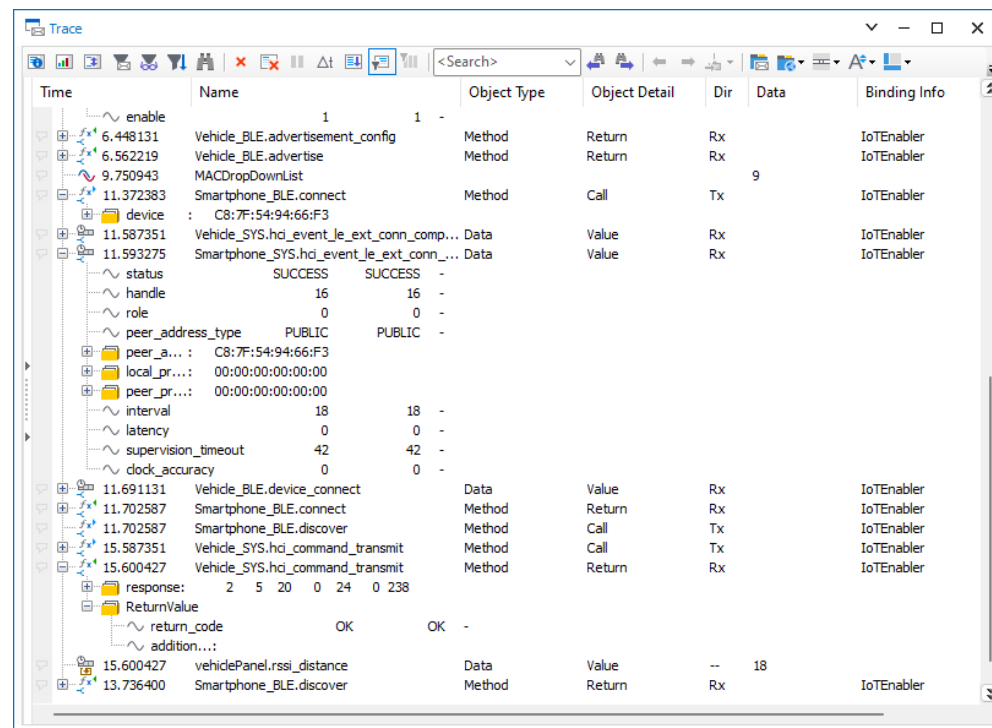
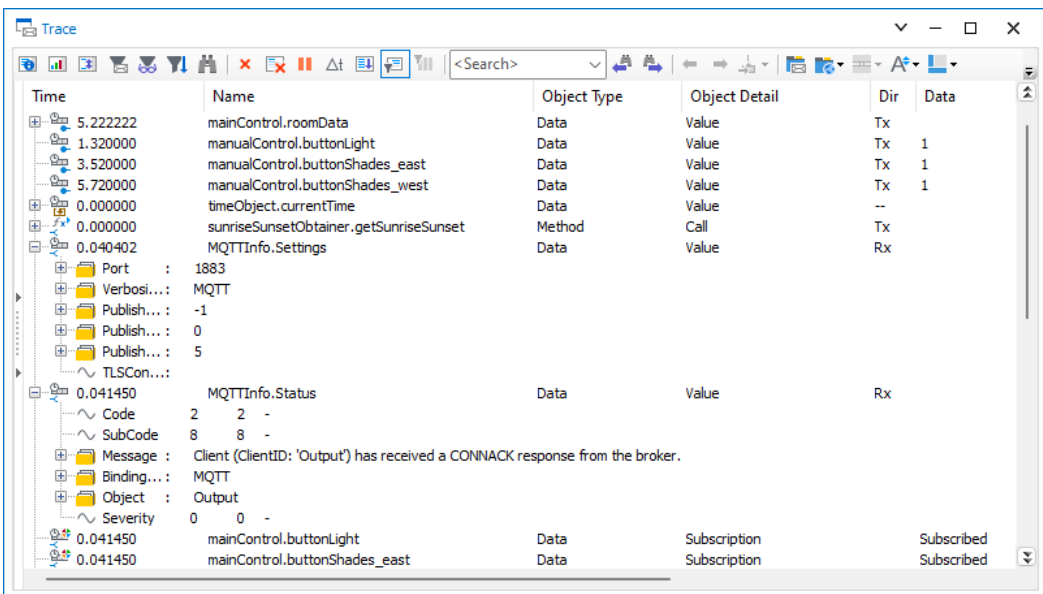
CANoe.Connectivityの設定

▶ **プロトコル別の対応機能**



CANoe.Connectivityの解析機能

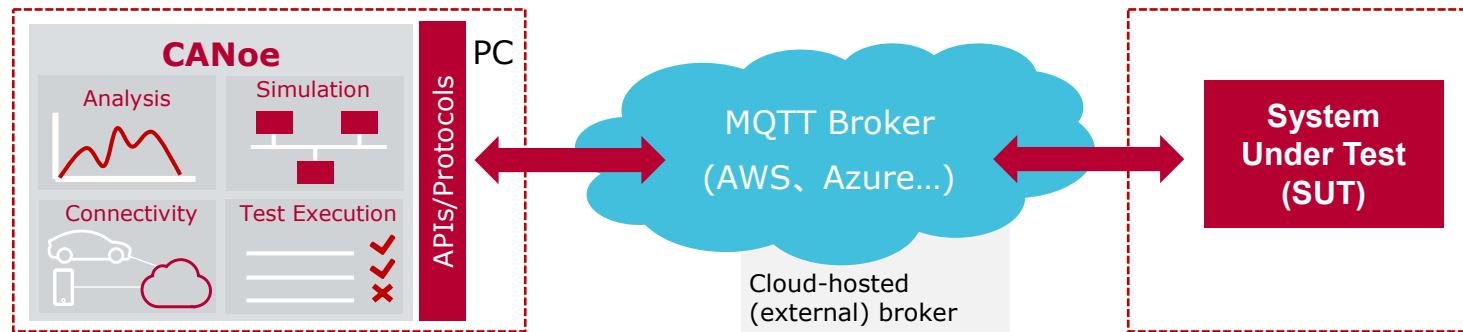
- ▶ Trace Windowでプロトコルの詳細をモニター
 - ▶ MQTTメタ情報、BLE HCIコマンド、HTTPヘッダーなど
 - ▶ Ethernet通信の詳細モニターも可能 (MAC、IP、UDP)
- ▶ ログの保存に対応
 - ▶ ファイル形式 : BLF、ASC
- ▶ ログの再生も可能



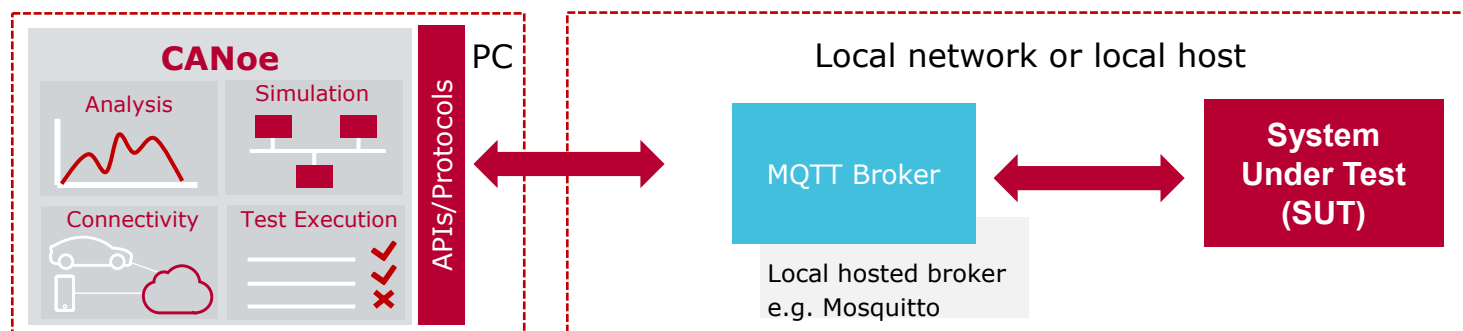
MQTTのシミュレーションとテスト (1)

使用できるMQTTブローカーの種類……

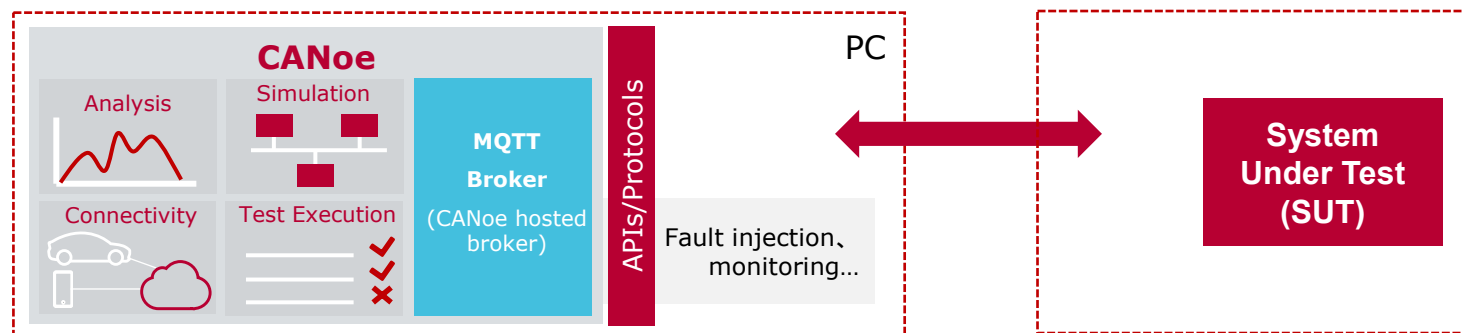
1. クラウド・リモートPC上のMQTTブローカーに接続



2. ローカルネットワークもしくはローカルホストのMQTTブローカーに接続



3. CANoeでMQTTブローカーをシミュレーション (フォールトインジェクション可能)



MQTTのシミュレーションとテスト (2)

- ▶ Clientシミュレーション用のAPIを用意・・・
 - ▶ Connect / Disconnect
 - ▶ Publish / Subscribe
 - ▶ コールバック
 - > OnConnect、OnDisconnect、OnPublish、OnSubscribe、...

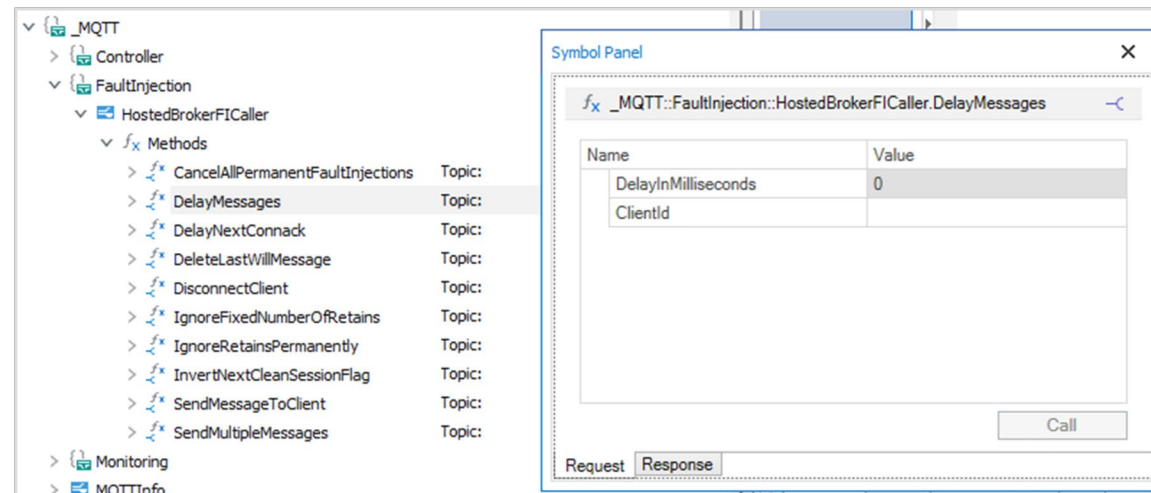
- ▶ CANoeでMQTTブローカーをシミュレーションする場合、ブローカー用のAPIも使用可能・・・
 - ▶ brokerの開始・停止
 - ▶ フォールトインジェクションAPI・・・
 - > Clientの強制切断
 - > 任意のメッセージをClientへ送信
 - > フラグの変更
 - > Retainオプションの無視
 - > メッセージスキップ (last willなど)

```
import vector.canoe
import application_layer
from random import randrange

@vector.canoe.measurement_script
class MQTTTester:

    def start(self):
        self.timer = vector.canoe.MsTimer(1000, self.send_values)
        self.timer.start()

    def send_values(self):
        Demo.client_pub.providedData.data_a = randrange(0, 10)
        self.timer.start()
```



The screenshot shows the Vector CANoe interface. On the left, a tree view displays the MQTT configuration structure:

- √ _MQTT
 - > Controller
 - √ FaultInjection
 - √ HostedBrokerFICaller
 - √ Methods
 - > CancelAllPermanentFaultInjections Topic:
 - > DelayMessages Topic:
 - > DelayNextConnack Topic:
 - > DeleteLastWillMessage Topic:
 - > DisconnectClient Topic:
 - > IgnoreFixedNumberOfRetains Topic:
 - > IgnoreRetainsPermanently Topic:
 - > InvertNextCleanSessionFlag Topic:
 - > SendMessageToClient Topic:
 - > SendMultipleMessages Topic:
 - > Monitoring
 - > MQTTInfo

On the right, the Symbol Panel is open, showing the configuration for the `f_x _MQTT::FaultInjection::HostedBrokerFICaller.DelayMessages` method. It contains a table with the following data:

Name	Value
DelayInMilliseconds	0
ClientId	

At the bottom of the Symbol Panel, there are tabs for 'Request' and 'Response', and a 'Call' button.

HTTPのシミュレーションとテスト

- ▶ HTTPクライアント・サーバーのシミュレーション
- ▶ JSONオブジェクト対応

- ▶ 2つのAPIに対応
 - ▶ ハイレベルAPI
 - > 詳細なHTTP知識不要で実装可能
 - > 標準的なユースケースに対応
 - ▶ ローレベルAPI
 - > HTTPヘッダーやボディの詳細な設定やシミュレーションが可能

- ▶ VN5000シリーズ Ethernet I/Fを使用すると、OSI下位層の解析やフォールトインジェクションも可能



The screenshot shows the Vector CANoe software interface. The main window displays a test suite titled "Test Light and Shades" with a list of test cases and their results. The test cases include "Try to Turn Light On", "Try to Turn Light Off", "Try to Close Shades East", "Try to Open Shades East", "Try to Close Shades West", and "Try to Open Shades West". The results show that all tests passed successfully. The "Runtime" column indicates the execution time for each test case, with the total runtime being 0.591s.

The "Trace" window shows a detailed log of MQTT and HTTP messages. The messages include MQTTInfo.Settings, MQTTInfo.Status, mainControl.buttonLight, mainControl.buttonShades_east, mainControl.buttonShades_west, mainControl.sunriseSunsetData, output.roomData, HostedBrokerMonitor.IsBrokerRunning, HostedBrokerMonitor.LastMQTTMessage, output.roomData, mainControl.buttonLight, mainControl.lightAutomatic, mainControl.buttonShades_east, mainControl.shades_eastAutomatic, mainControl.buttonShades_west, mainControl.shades_westAutomatic, output.shadesEastAnimationState, and output.shadesWestAnimationState.

The "Flowchart" window illustrates the system architecture. It shows a cloud icon connected to "HTTP to MQTT", which is connected to "Main Control". "Main Control" is connected to "Manual Control" and "Output". "Manual Control" is also connected to "Main Control".

The "_HTTP::Client" window shows the configuration for an HTTP client. The configuration includes the following fields:

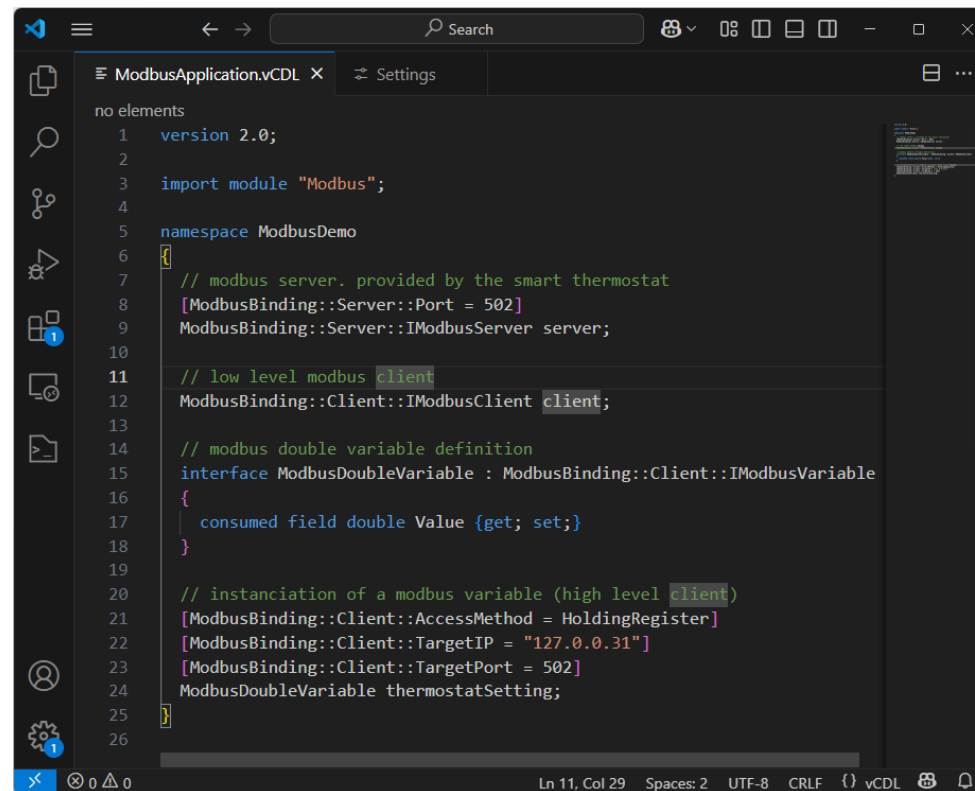
Name	Op..	Value
RequestMessage		HTTPRequestMessage
HTTPMethod		
RequestUri		
Header (min=0,max=hasNoMaxLength)	<input checked="" type="checkbox"/>	HeaderField[0]
Body (min=1,max=hasNoMaxLength)	<input checked="" type="checkbox"/>	[1]
AuthConfig	<input checked="" type="checkbox"/>	AuthConfig
HTTPClientEndpoint	<input checked="" type="checkbox"/>	

Modbus, gRPCのシミュレーションとテスト

- ▶ Modbus/TCP.....
 - ▶ クライアント、サーバー、ゲートウェイのシミュレーション
 - ▶ VN5000シリーズ Ethernet I/Fを使用すると、OSI下位層の解析やフォールトインジェクションも可能



- ▶ gRPC.....
 - ▶ Unaryリクエストのクライアントシミュレーション
 - ▶ Protocol Buffers自動変換



```
no elements
1  version 2.0;
2
3  import module "Modbus";
4
5  namespace ModbusDemo
6  {
7    // modbus server. provided by the smart thermostat
8    [ModbusBinding::Server::Port = 502]
9    ModbusBinding::Server::IModbusServer server;
10
11   // low level modbus client
12   ModbusBinding::Client::IModbusClient client;
13
14   // modbus double variable definition
15   interface ModbusDoubleVariable : ModbusBinding::Client::IModbusVariable
16   {
17     consumed field double Value {get; set;}
18   }
19
20   // instantiation of a modbus variable (high level client)
21   [ModbusBinding::Client::AccessMethod = HoldingRegister]
22   [ModbusBinding::Client::TargetIP = "127.0.0.31"]
23   [ModbusBinding::Client::TargetPort = 502]
24   ModbusDoubleVariable thermostatSetting;
25 }
26
```

無線通信のシミュレーションとテスト

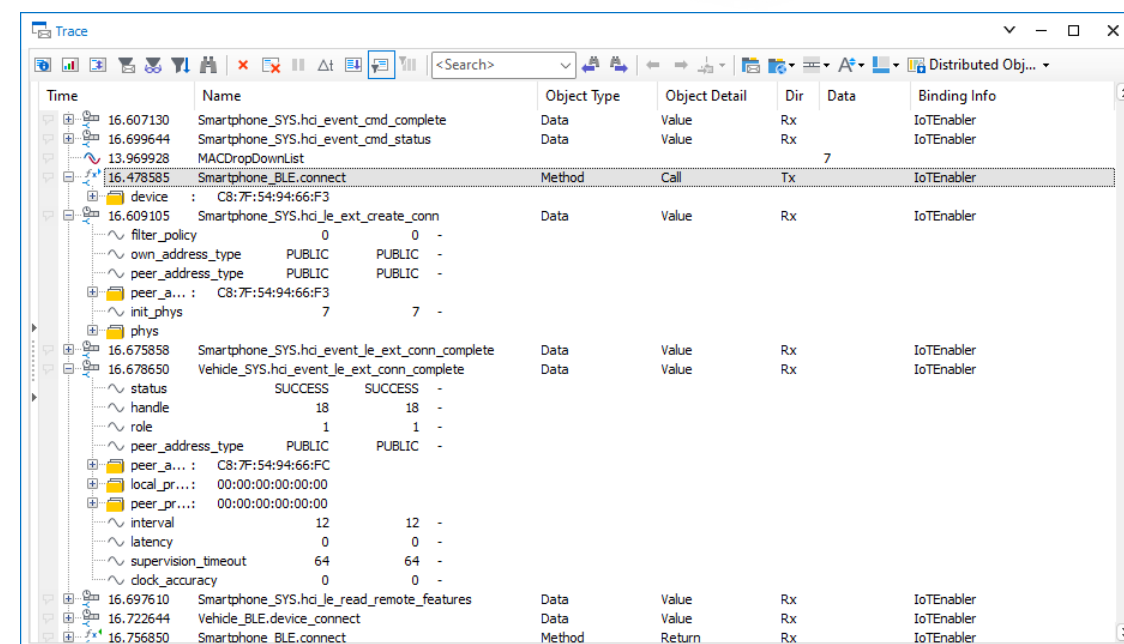
VH4110を使用して、無線通信をシミュレーション・テスト

- ▶ BLE.....
 - ▶ Central / Peripheralのシミュレーション
 - ▶ GATT/L2CAP、ペアリング、HCIトレース機能
 - ▶ サンプルコンフィギュレーション付属

- ▶ NFC.....
 - ▶ リーダー・カード／タグモード (ISO 7816 APDU)

- ▶ Wi-Fi.....
 - ▶ Access point／ステーションモード
 - ▶ パッシブモニタリング

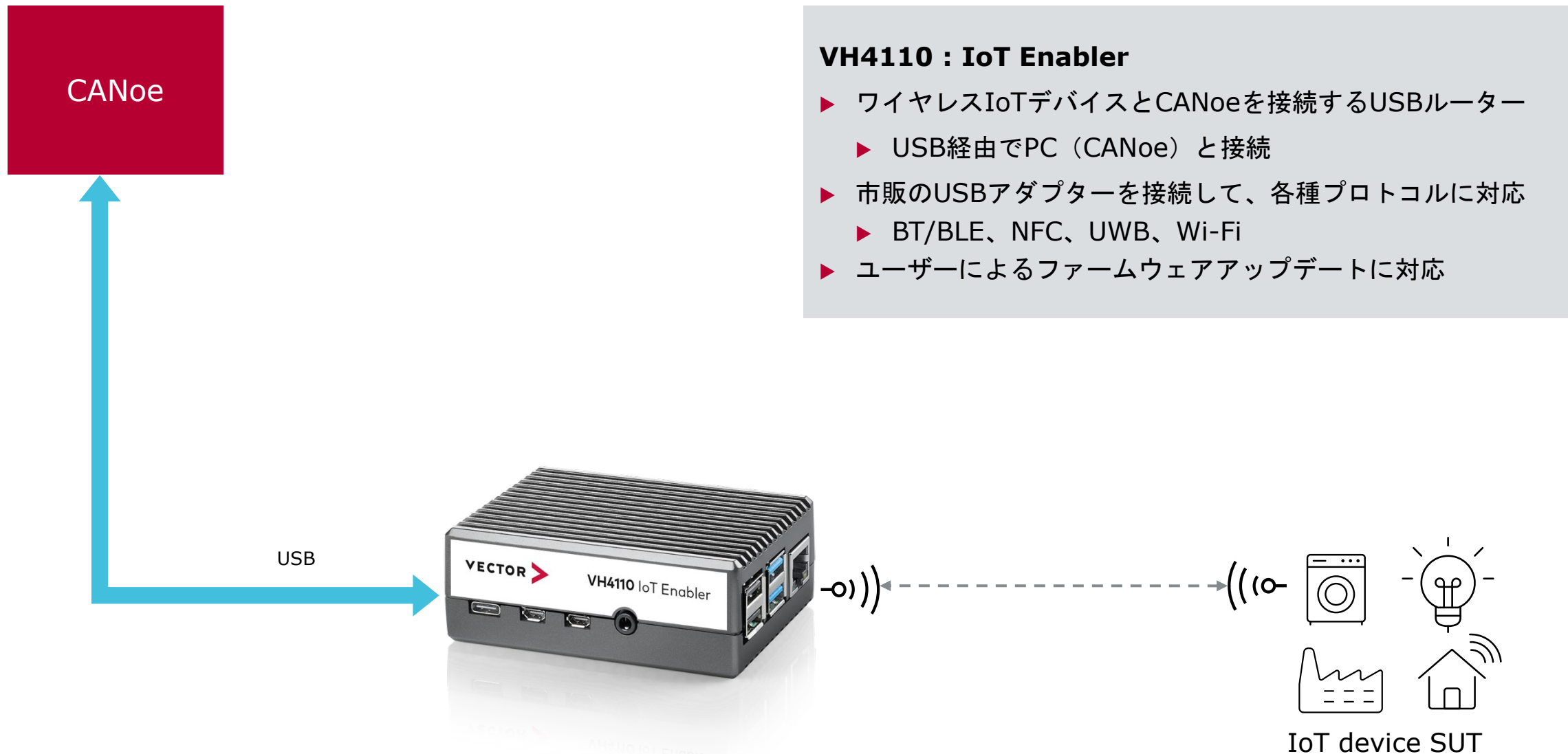
- ▶ UWB.....
 - ▶ UCIインターフェース
(イニシエーター・レスポnderモード)

The screenshot shows a trace window with a table of communication events. The table has columns for Time, Name, Object Type, Object Detail, Dir, Data, and Binding Info. The selected event is at time 16.478585, named 'Smartphone_BLE.connect', with Object Type 'Method' and Dir 'Tx'.

Time	Name	Object Type	Object Detail	Dir	Data	Binding Info
16.607130	Smartphone_SYS.hci_event_cmd_complete	Data	Value	Rx		IoTEnabler
16.699644	Smartphone_SYS.hci_event_cmd_status	Data	Value	Rx		IoTEnabler
13.969928	MACDropDownList				7	
16.478585	Smartphone_BLE.connect	Method	Call	Tx		IoTEnabler
	device : C8:7F:54:94:66:F3					
16.609105	Smartphone_SYS.hci_le_ext_create_conn	Data	Value	Rx		IoTEnabler
	filter_policy 0 0 -					
	own_address_type PUBLIC PUBLIC -					
	peer_address_type PUBLIC PUBLIC -					
	peer_a... : C8:7F:54:94:66:F3					
	init_phys 7 7 -					
	phys					
16.675858	Smartphone_SYS.hci_event_le_ext_conn_complete	Data	Value	Rx		IoTEnabler
16.678650	Vehicle_SYS.hci_event_le_ext_conn_complete	Data	Value	Rx		IoTEnabler
	status SUCCESS SUCCESS -					
	handle 18 18 -					
	role 1 1 -					
	peer_address_type PUBLIC PUBLIC -					
	peer_a... : C8:7F:54:94:66:FC					
	local_pr... : 00:00:00:00:00:00					
	peer_pr... : 00:00:00:00:00:00					
	interval 12 12 -					
	latency 0 0 -					
	supervision_timeout 64 64 -					
	dock_accuracy 0 0 -					
16.697610	Smartphone_SYS.hci_le_read_remote_features	Data	Value	Rx		IoTEnabler
16.722644	Vehicle_BLE.device_connect	Data	Value	Rx		IoTEnabler
16.756850	Smartphone_BLE.connect	Method	Return	Rx		IoTEnabler

VH4110 IoT Enablerとは

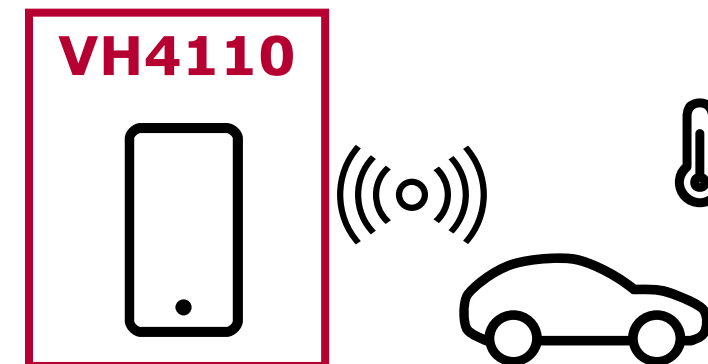
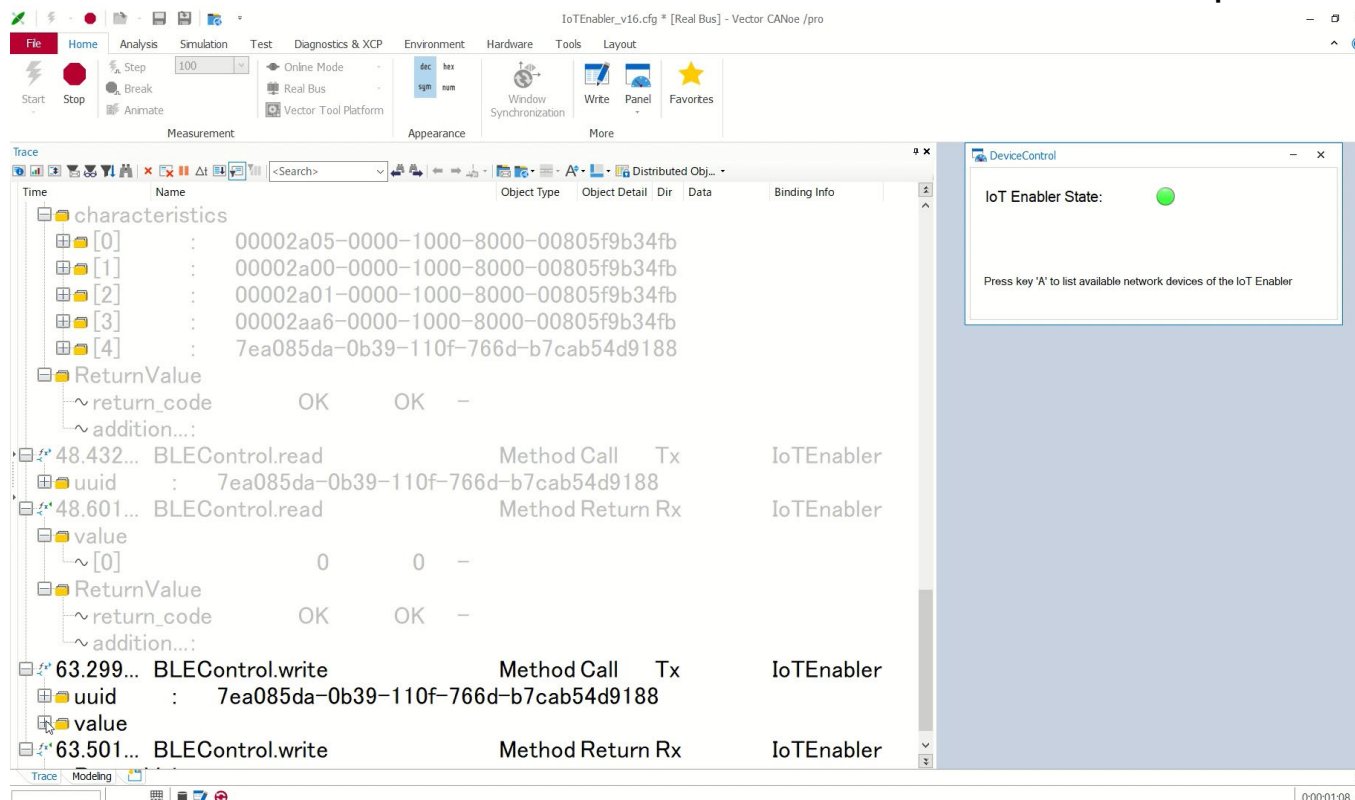


VH4110 : IoT Enabler

- ▶ ワイヤレスIoTデバイスとCANoeを接続するUSBルーター
- ▶ USB経由でPC（CANoe）と接続
- ▶ 市販のUSBアダプターを接続して、各種プロトコルに対応
 - ▶ BT/BLE、NFC、UWB、Wi-Fi
- ▶ ユーザーによるファームウェアアップデートに対応

VH4110を使用したBLE Centralのシミュレーション

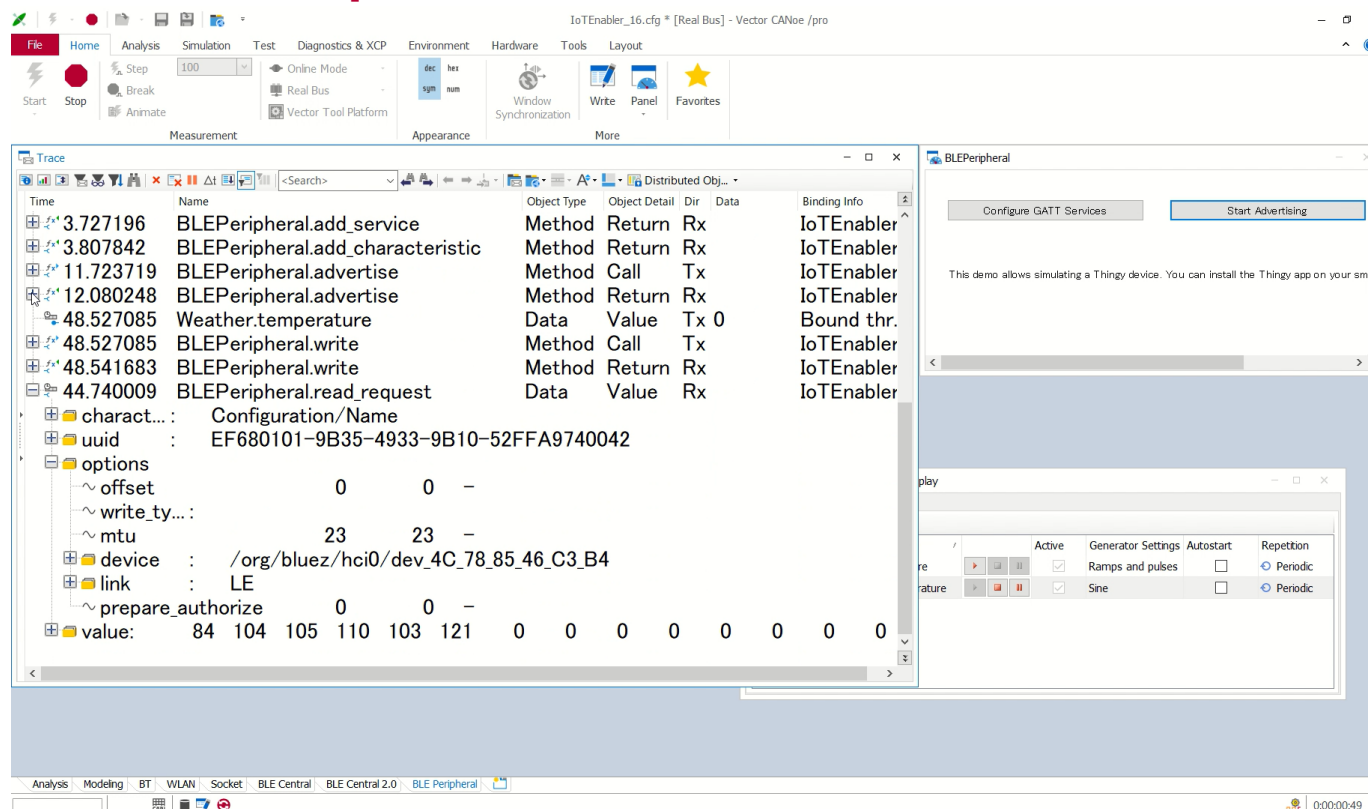
- ▶ VH4110で**BLE Central機能**をシミュレーションして、BLE Peripheralアプリケーションを動作テスト



- ▶ 対応するBLE Central機能・・・
 - ▶ デバイスのScan、Connect
 - ▶ Peripheralが持つCharacteristicのRead、Write など

VH4110を使用したBLE Peripheralのシミュレーション

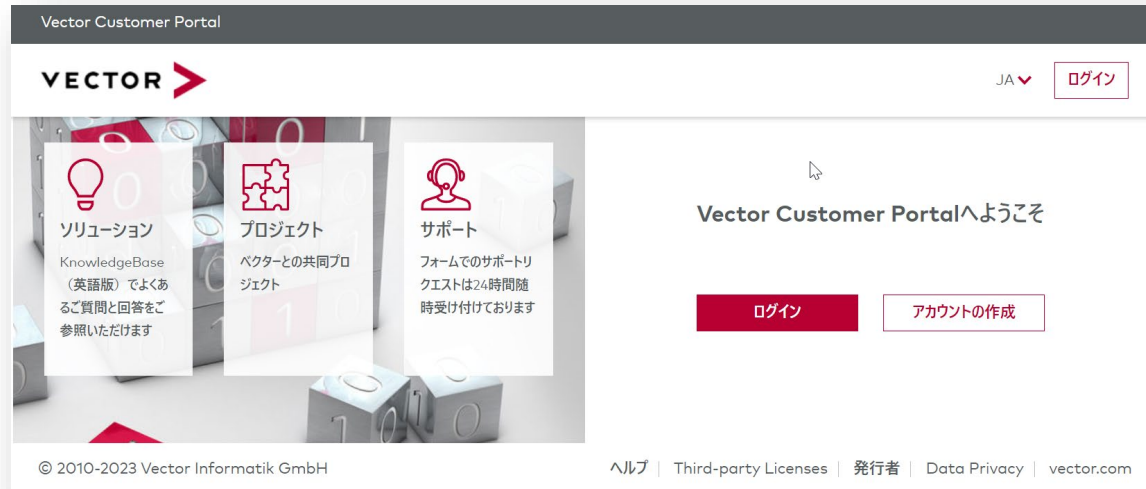
- ▶ VH4110で**BLE Peripheral機能**をシミュレーションして、BLE Centralアプリケーションを動作テスト



- ▶ 対応するBLE Peripheral機能・・・
 - ▶ Advertiseの送信
 - ▶ Notificationの送信
など

Vector Customer Portal

- ▶ ベクターWebサイト「[Vector Customer Portal](#)※」から、製品説明スライドやアドバンストマニュアル等の資料をダウンロード可能

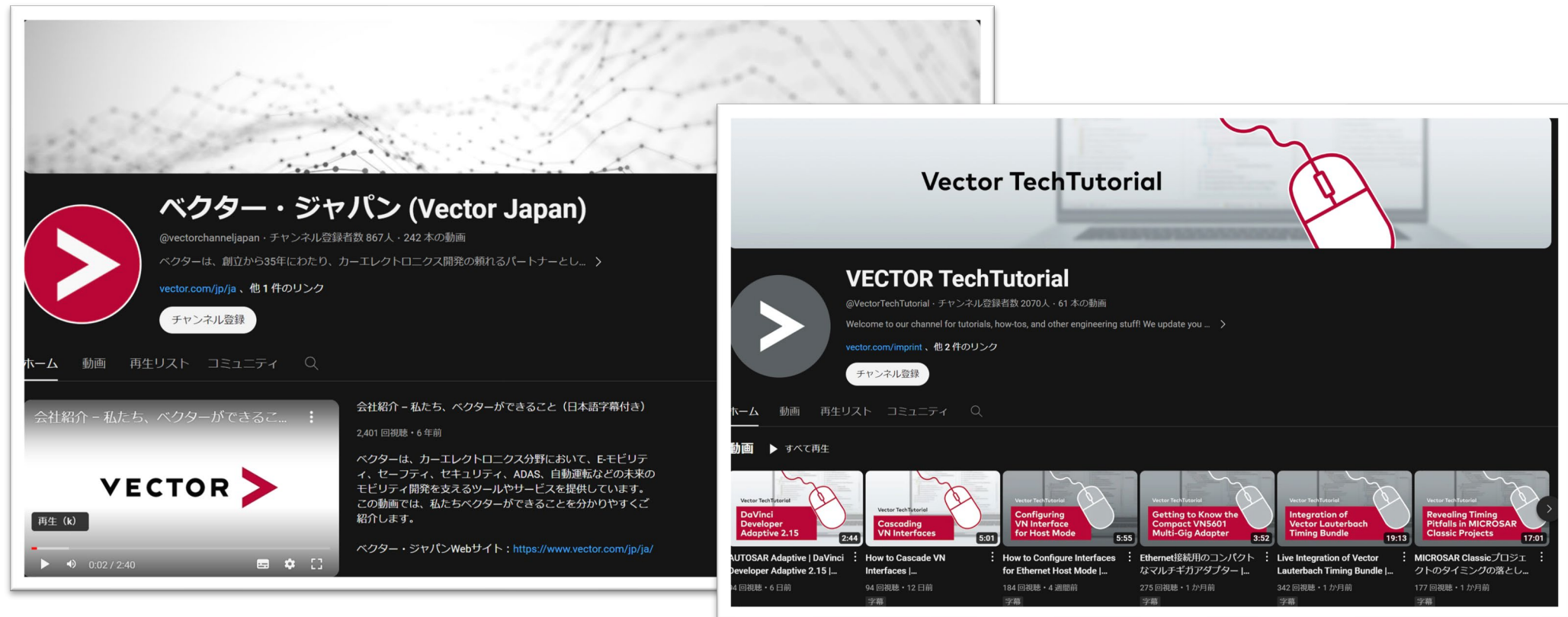


※ Customer Portalは、ベクターの製品をご使用頂いているお客様向けの無料会員サイトです。

- ▶ 製品説明スライド (概要)
 - ▶ CANoe/CANalyzerの紹介
 - ▶ CANoe.Connectivityの紹介
 - ▶ CANoe.DDSの紹介
 - ▶ VN5000シリーズ Ethernet I/Fの紹介
 - etc.
- ▶ アドバンストマニュアル
 - ▶ CANoe/CANalyzer.Ethernet トラブルシューティング
 - ▶ Vector Hardware Managerの設定方法
 - ▶ SOME/IPのシミュレーション・解析
 - ▶ CANoe.Connectivity/DDSを使用したIoTアプリケーションの解析・シミュレーション
 - etc.

サポート資料の紹介

- ▶ ベクター・ジャパンYoutubeチャンネルやVECTOR TechTutorial Youtubeチャンネルにて、ベクター製品の使用方法や各種通信プロトコルなどを解説した動画を閲覧可能
- ▶ ユースケースの紹介動画「IoT・医療・鉄道システムのシミュレーション環境でのテスト」や、実装方法の説明動画「あらゆる業界向けのプロトコルをサポート」も公開中



ベクター・ジャパン株式会社

www.vector.com/jp/ja/

- ▶ 営業部および各種お問い合わせ（Eメール）
- ▶ サポートリクエスト

※記載内容については予告なく変更されることがありますので、あらかじめご了承ください。