Cloud Connected Sensor Data-as-a-Service

Solutions de démarrage

Manuel utilisateur

Traduction de la notice originale



YAGEO GROUP Brings you

Telemecanique Sensors

Ce document contient des informations générales et techniques relatives aux produits vendus par TMSS France, ses filiales et autres sociétés affiliées.

Les informations et caractéristiques contenues dans ce document ne sont en aucun cas destinées à se substituer aux analyses et tests devant être effectués par l'utilisateur ou l'intégrateur des produits. Il est de la responsabilité de chaque utilisateur ou intégrateur d'effectuer les analyses de risque et tests nécessaires sur les produits pour s'assurer de leur adéquation avec leurs besoins spécifiques et la performance attendue. Ni TMSS France, ni aucune de ses filiales ou autres sociétés affiliées ne peuvent être tenues pour responsables d'une mauvaise utilisation des informations contenues dans le présent document.

TelemecaniqueTM Sensors est une marque de Schneider Electric Industries SAS utilisée sous licence par TMSS France. Toutes les autres marques citées dans ce document sont la propriété TMSS France, de ses filiales ou autres sociétés affiliées ou, le cas échéant, de ses concédants de licence.

Table des matières

Consignes de sécurité 5
A propos de ce document 6
Présentation de Cloud Connected Sensor Data-as-a-Service 7
Offre héritée Cloud Connected Sensor 7
Offre Cloud Connected Sensor Data-as-a-Service 8
Messagerie Publier / S'abonner et processus ETL 10
Messagerie Publier / S'abonner et processus ETL appliqués à Cloud Connected Sensor Data-as-a-Service 10
Data-as-a-Service dans Cloud Connected Sensor 12
Comment activer Data-as-a-Service dans Cloud Connected Sensor 12
Fonctionnalités liées à Cloud Connected Sensor Data-as-a-Service 12
Guide d'activation étape par étape de Cloud Connected Sensor Data-as-a-Service 13
Format des messages des émetteurs 14
Format des messages 14
Trame de configuration produit (`RecordType` = 16 (10_h)) 17
Trame de configuration réseau (`RecordType` = 32 (20_h)) 18
Trame d'état (`RecordType` = 48 (30_h)) 19
Trame de mesure (`RecordType` = 64 (40_h)) 20
Trame de géolocalisation (RecordType 255 (FFh)) 21
Transmission des données par radio 22
Détails des trames radio 22
Cloud Connected Sensor Data-as-a-Service - Solutions de démarrage 25
Vue conceptuelle des solutions de démarrage 26
Solution de démarrage pour un fichier simple 28
Packages NuGet requis, SDK et bibliothèques 28
Configuration préalable à l'exécution 28
Comportement de la solution de démarrage 28
Solution de démarrage pour WonderWare Historian 30
Packages NuGet requis, SDK et bibliothèques 30
Configuration préalable à l'exécution 30
Comportement de la solution de démarrage 31
Solution de démarrage pour Modbus over TCP 32
Packages NuGet requis, SDK, et bibliothèques 32
Configuration préalable à l'exécution 32
Comportement de la solution de démarrage 33
Annexes 37
Principe de la messagerie Publier / S'abonner et des processus ETL 38
Processus ETL 39
Présentation des processus ETL 39
Processus d'extraction 39
Processus de transformation 39
Processus de chargement 40
Messagerie Publier / S'abonner 41
Concept de base 41

Remarque relative à la cybersécurité 42
Informations relatives à la technologie utilisée 43
Microsoft Azure Service Bus 43
Microsoft Visual Studio 43
Glossaire 45

Consignes de sécurité

Informations importantes

Lisez attentivement ces instructions et examinez le matériel pour vous familiariser avec l'appareil avant de tenter de l'installer, de le faire fonctionner, de le réparer ou d'assurer sa maintenance. Les messages spéciaux suivants que vous trouverez dans cette documentation ou sur l'appareil ont pour but de vous mettre en garde contre des risques potentiels ou d'attirer votre attention sur des informations qui clarifient ou simplifient une procédure.



La présence de ce symbole sur une étiquette “Danger” ou “Avertissement” signale un risque d'électrocution qui provoquera des blessures physiques en cas de non-respect des consignes de sécurité.



Ce symbole est le symbole d'alerte de sécurité. Il vous avertit d'un risque de blessures corporelles. Respectez scrupuleusement les consignes de sécurité associées à ce symbole pour éviter de vous blesser ou de mettre votre vie en danger.

⚠ DANGER

DANGER signale un risque qui, en cas de non-respect des consignes de sécurité, **provoque** la mort ou des blessures graves.

⚠ AVERTISSEMENT

AVERTISSEMENT signale un risque qui, en cas de non-respect des consignes de sécurité, **peut provoquer** la mort ou des blessures graves.

⚠ ATTENTION

ATTENTION signale un risque qui, en cas de non-respect des consignes de sécurité, **peut provoquer** des blessures légères ou moyennement graves.

AVIS

AVIS indique des pratiques n'entraînant pas de risques corporels.

Remarque Importante

L'installation, l'utilisation, la réparation et la maintenance des équipements électriques doivent être assurées par du personnel qualifié uniquement. Schneider Electric décline toute responsabilité quant aux conséquences de l'utilisation de ce matériel.

Une personne qualifiée est une personne disposant de compétences et de connaissances dans le domaine de la construction, du fonctionnement et de l'installation des équipements électriques, et ayant suivi une formation en sécurité leur permettant d'identifier et d'éviter les risques encourus.

A propos de ce document

Objet du document

Ce document a pour objectif de fournir au client des informations sur la solution Cloud Connected Sensor Data-as-a-Service, son activation, son utilisation et les possibilités qu'elle offre.

Note de validité

Ce document a été mis à jour avec les solutions de démarrage Data-as-a-Service V1.1.

Pour plus d'informations sur la conformité des produits avec les normes environnementales (RoHS, REACH, PEP, EOLI, etc.), consultez le site www.se.com/ww/en/work/support/green-premium/.

Les caractéristiques techniques des équipements décrits dans ce document sont également fournies en ligne. Pour accéder aux informations en ligne, allez sur la page d'accueil de Schneider Electric www.se.com/ww/en/download/.

Les caractéristiques présentées dans ce manuel devraient être identiques à celles fournies en ligne. Toutefois, en application de notre politique d'amélioration continue, nous pouvons être amenés à réviser le contenu du document afin de le rendre plus clair et plus précis. Si vous constatez une différence entre le manuel et les informations fournies en ligne, utilisez ces dernières en priorité.

Documents à consulter

Titre de la documentation	Numéro de référence
XIOT11SE••• Émetteur autonome - Fiche d'instructions	QGH83382 (ENG) QGH83382 (ITA) QGH83382 (SPA) QGH83382 (FRE) QGH83382 (DAN) QGH83382 (GER) QGH83382 (SWE) QGH83382 (DUT)
Cloud Connected Sensor Data-as-a-Service - Solution de démarrage WonderWare	XIOTDaaSStarterSolution_WonderWare
Cloud Connected Sensor Data-as-a-Service - Solution de démarrage Visual Studio	XIOTDaaSStarterSolution_Visual Studio

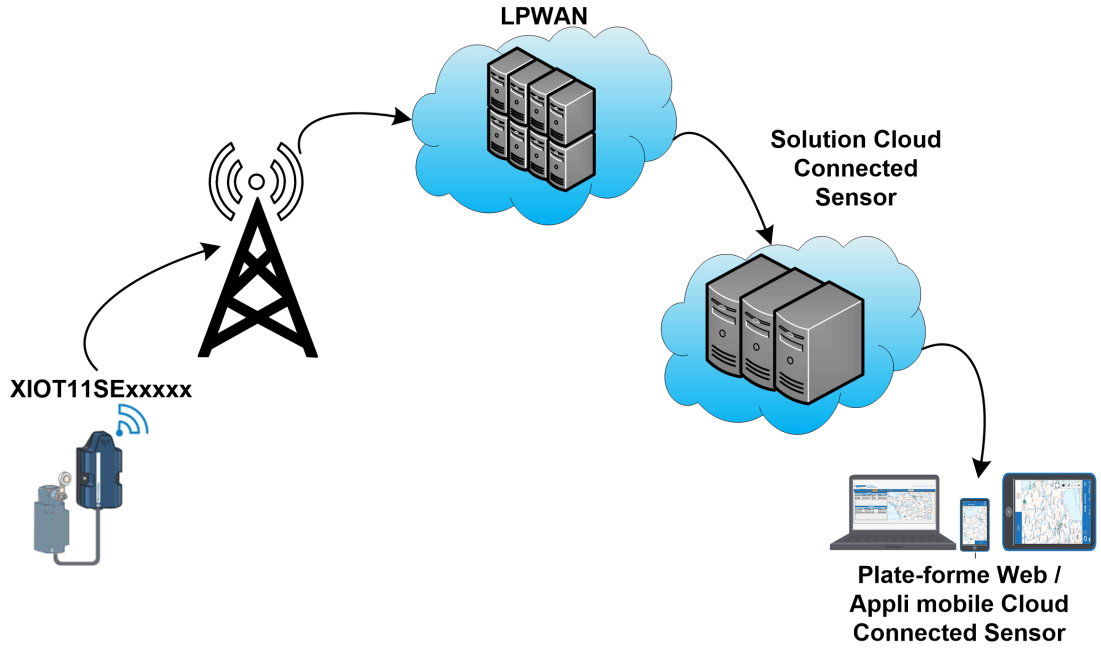
Vous pouvez télécharger ces publications, le présent manuel et autres informations techniques depuis notre site web à l'adresse : www.se.com/en/download/.

Présentation de Cloud Connected Sensor Data-as-a-Service

Offre héritée Cloud Connected Sensor

Présentation

Présentation de Cloud Connected Sensor hérité :



L'émetteur autonome XIOT11SE••••• permet d'exploiter les changements d'état d'un ou deux contacts secs via une connexion LPWAN (Sigfox). Les informations transmises sont mises à disposition sur le Web, via la plate-forme Cloud Connected Sensor (https://XIOT.Tesensors.com) ou à partir des applications mobiles IOS / Android (appli Cloud Connected Switch).

NOTE: Cloud Connected Sensor Data-as-a-Service vient compléter et s'ajouter aux fonctionnalités Cloud Connected Sensor héritées. Que Data-as-a-Service soit activé ou non, les données de l'émetteur restent disponibles sur la plate-forme Web Cloud Connected Switch de l'appli mobile.

Offre Cloud Connected Sensor Data-as-a-Service

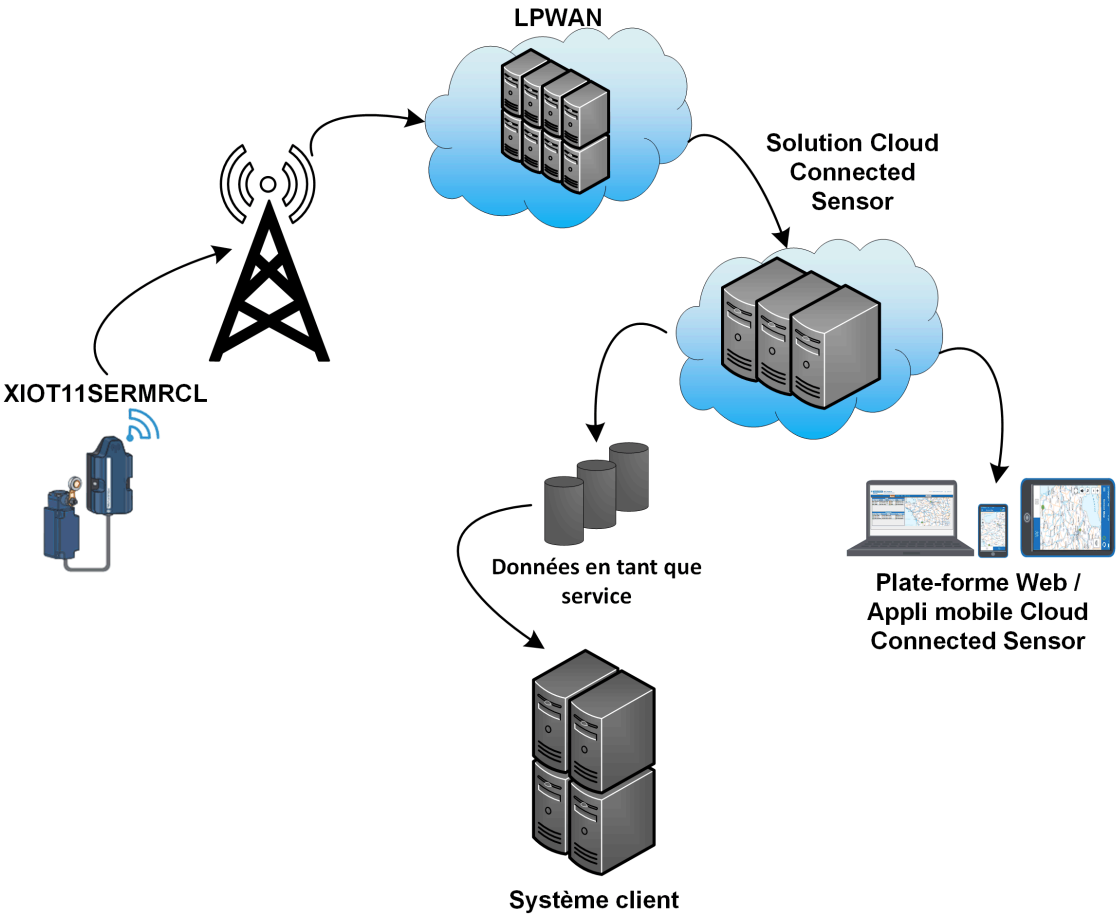
Présentation

Cloud Connected Sensor Data-as-a-Service (DaaS) offre des capacités d'intégration pour stocker des données et gérer les équipements Cloud Connected Sensor avec les systèmes du client, par exemple les systèmes SCADA ou les systèmes d'application client spécifiques.

NOTE: la solution d'intégration Cloud Connected Sensor Data-as-a-Service est uniquement disponible pour la référence XIOT11SERMRCL.

Cloud Connected Sensor offre une solution orientée messages, en quasi-temps réel, pour Data-as-a-Service.

Présentation de Cloud Connected Sensor avec Data-as-a-Service :



Solution Cloud Connected Sensor

Les solutions d'intégration permettent d'extraire les données du système source et de les charger dans le système de destination.

Dans le cas de Cloud Connected Sensor Data-as-a-Service :

- le système source est la solution Cloud Connected Sensor,
- le système de destination est toujours le système client.

Cloud Connected Sensor Data-as-a-Service permet de récupérer les informations d'un émetteur à partir de la plate-forme Cloud Connected Sensor et offre la possibilité de stocker et de surveiller les données des systèmes client, quels qu'ils soient.

Ces processus sont souvent appelés **ETL** :

- **E** pour **Extract** : extraient les données du système source au format d'extraction.
- **T** pour **Transform** : transforment les données du format d'extraction au format de chargement.
- **L** pour **Load** : chargent les données dans le système de destination

Le principe de messagerie Publier / S'abonner est utilisé pour mettre les données à disposition sur le système de destination.

Selon le concept ETL, le fait de récupérer les données à partir du système source est appelé processus d'extraction. Cependant, dans le concept de messagerie Publier / S'abonner, le fait de récupérer les données à partir du stockage de données est appelé processus de consommation.

Dans les chapitres suivants, la terminologie Extraire / Consommer décrit la même action de récupération des données à partir d'une source de données. Dans la mesure où le concept de Cloud Connected Sensor Data-as-a-Service repose principalement sur la messagerie Publier / S'abonner, l'expression « processus de consommation » est plus adaptée.

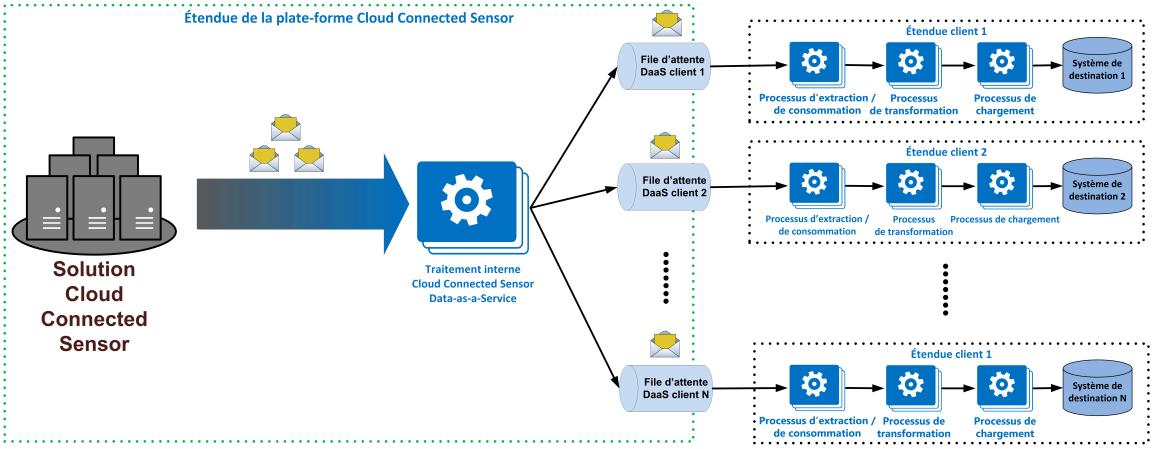
Pour plus d'informations sur les concepts orientés messages, reportez-vous à la section Principe de la messagerie Publier / S'abonner et des processus ETL, page 38.

Messagerie Publier / S’abonner et processus ETL

Messagerie Publier / S’abonner et processus ETL appliqués à Cloud Connected Sensor Data-as-a-Service

Présentation

Vue conceptuelle de la messagerie Publier / S’abonner et des processus ETL appliqués à la solution Cloud Connected Sensor Data-as-a-Service :



Principe

Dans le cas de Cloud Connected Sensor Data-as-a-Service :

- La plate-forme Cloud Connected Sensor est l’éditeur du message.
- DaaS Customer Queue est la plate-forme de publication / d’abonnement basée sur la technologie Microsoft Azure Service Bus Queue.
- Les mécanismes de communication sont les API Microsoft Azure Service Bus API (application program interface).
- Le consommateur du message est le client qui a activé Cloud Connected Sensor Data-as-a-Service.
- Il s’agit du consommateur du message qui implémente le processus d’extraction (consommation), le processus de transformation et le processus de chargement.

Une fois le processus d’extraction exécuté, les données sont chargées sur le système de destination et ne sont plus stockées sur la plate-forme Cloud Connected Sensor.

⚠AVERTISSEMENT

RISQUES POTENTIELS POUVANT PORTER ATTEINTE À LA DISPONIBILITÉ, À L’INTÉGRITÉ ET À LA CONFIDENTIALITÉ DU SYSTÈME

- Placez les équipements en réseau derrière plusieurs couches de cyberdéfense (telles que les pare-feux, la segmentation du réseau, ainsi que la détection et la protection contre les intrusions sur le réseau).
- Appliquez les bonnes pratiques en matière de cybersécurité (par exemple : privilèges restreints, séparation des tâches) pour éviter l’exposition à des personnes non autorisées, la perte, la modification des données et des journaux ou l’interruption des services.

Le non-respect de ces instructions peut provoquer la mort, des blessures graves ou des dommages matériels.

L’activation des Data-as-a-Service à partir de la plate-forme Cloud Connected Sensor donne accès à la chaîne de connexion.

Une chaîne de connexion est une chaîne qui spécifie des informations à propos d’une source de données et les méthodes permettant de s’y connecter. Dans le cas de Cloud Connected Sensor Data-as-a-Service, la chaîne de connexion fournie est destinée à Microsoft Azure Service Bus Queue.

Pour plus d’informations sur l’utilisation de la chaîne de connexion fournie, reportez-vous aux solutions de démarrage, et notamment à la fonction de consommation, page 25.

Consultez le site Web Microsoft Azure pour plus d’informations sur l’utilisation des chaînes de connexion et sur la façon de procéder pour se connecter à Microsoft Azure Service Bus Queue à l’aide d’une API dédiée. Reportez-vous également aux informations sur les technologies utilisées, page 43.

Data-as-a-Service dans Cloud Connected Sensor

Comment activer Data-as-a-Service dans Cloud Connected Sensor

Fonctionnalités liées à Cloud Connected Sensor Data-as-a-Service

Présentation

Cloud Connected Sensor Data-as-a-Service est uniquement disponible pour la référence XIOT11SERMRCL. La page **Services** de la plate-forme Web Cloud Connected Sensor est uniquement accessible si un équipement XIOT11SERMRCL a été déclaré.

La page **Services** de la plate-forme Web Cloud Connected Sensor permet d'effectuer les actions suivantes :

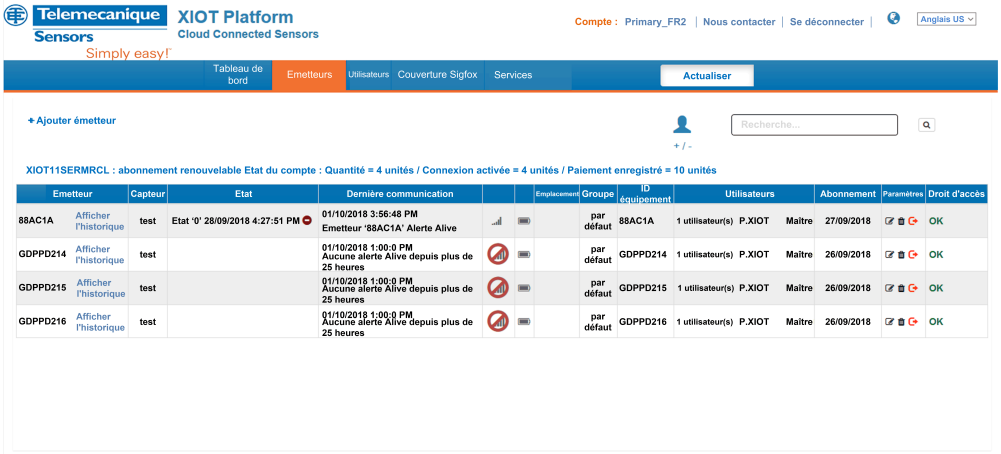
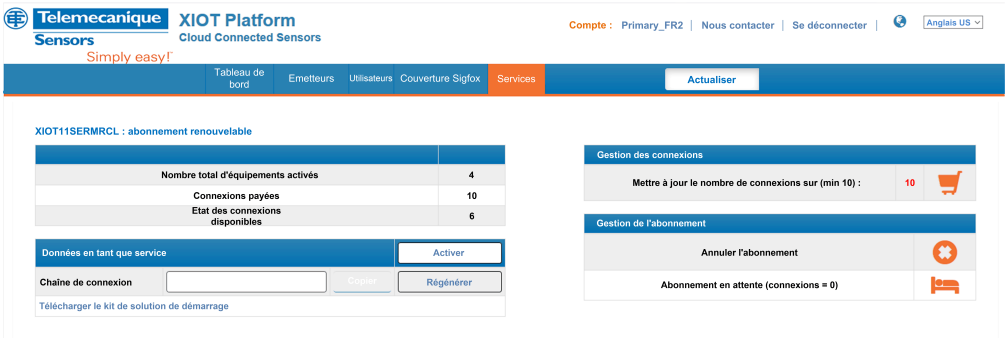
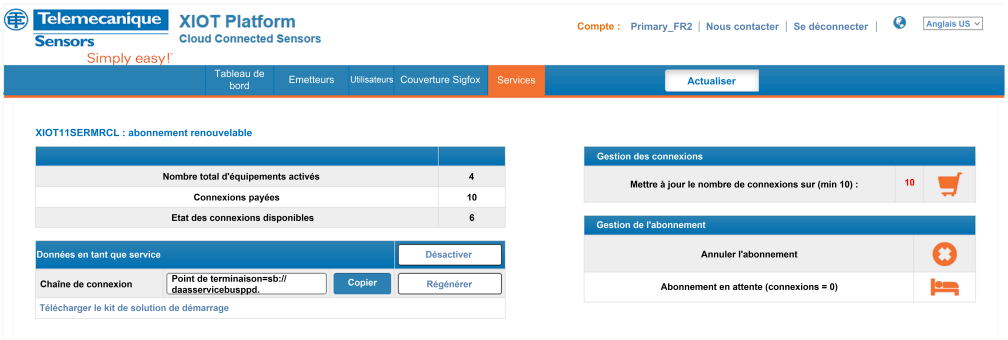
- Acheter des abonnements (connexions) pour les équipements XIOT11SERMRCL, à l'aide de l'icône **Panier**.
- Suspendre ou interrompre tous les abonnements pour les équipements.
- Activer / désactiver Data-as-a-Service.
- Régénérer la chaîne de connexion.
- Accéder aux fichiers des solutions de démarrage.

Comportement de Cloud Connected Sensor Data-as-a-Service

Informations sur le comportement de Cloud Connected Sensor Data-as-a-Service :

- Une fois que Data-as-a-Service est activé, toutes les données en provenance des équipements XIOT11SERMRCL associés à un abonnement valide sont disponibles en quasi-temps réel dans la file d'attente du client.
- Dans le cas où Cloud Connected Sensor Data-as-a-Service est déjà activé et où le client déclare un nouvel émetteur, prévoyez un délai de configuration de 15 minutes avant que les données en provenance de cet équipement soient disponibles dans la file d'attente du client.
- Toutes les données de la file d'attente du client seront disponibles pendant 15 jours.
- Lorsque le client se connecte à sa file d'attente et consomme un message, ce dernier n'est plus disponible dans la file d'attente.
- Dans la file d'attente du client DaaS, l'ordre FIFO (premier entré, premier sorti) est respecté.
- Si la connexion à la file d'attente (file d'attente DaaS) a été compromise, il est possible de régénérer la chaîne de connexion correspondante à l'aide du bouton dédié sur la page **Services**. La chaîne de connexion précédente deviendra non valide, et la file d'attente du client sera accessible uniquement via la nouvelle chaîne de connexion. Toutes les données incluses dans la file d'attente au moment de la régénération de la chaîne de connexion sont conservées.
- Seuls les messages en provenance d'équipements dont le client est maître seront transmis à la file d'attente du client. C'est pourquoi les messages en provenance d'équipements dont le client est administrateur ou utilisateur ne seront pas disponibles dans la file d'attente du client.

Guide d'activation étape par étape de Cloud Connected Sensor Data-as-a-Service

Étape	Action
1	Une fois qu'un XIOT11SERMRCL est déclaré, l'onglet **Services** de la plate-forme Cloud Connected Sensor devient accessible. Cloud Connected Sensor Data-as-a-Service est géré depuis cette page. 
2	Pour activer Cloud Connected Sensor DaaS, cliquez sur le bouton **Activer**. 
3	Cloud Connected Sensor DaaS est activé et la chaîne de connexion (informations de connexion) requise pour accéder à la file d'attente du client est disponible dans la zone dédiée. 
4	Les données destinées à l'émetteur sont accessibles dans la file d'attente DaaS à l'aide de la chaîne de connexion fournie.

Format des messages des émetteurs

Format des messages

Présentation

Les données de l'émetteur sont enregistrées dans la file d'attente du client au format JSON.

Divers messages JSON peuvent être présents dans la file d'attente du client DaaS. Chaque message JSON présente des attributs communs, ainsi que quelques attributs qui les différencient les uns des autres.

L'attribut JSON `RecordType` permet de savoir quel modèle JSON est utilisé.

Les émetteurs Cloud Connected Sensor peuvent envoyer 5 types de trame de données :

- Une trame de configuration produit, page 17
- Une trame de configuration réseau, page 18
- Une trame d'état, page 19
- Une trame de mesure, page 20
- Une trame de géolocalisation, page 21

Diagramme temporel

Trames transmises par l'émetteur :

Étape	Action	Trames envoyées
1	Activation du produit après la détection d'un aimant	Trame de configuration produit 10_h Trame de configuration réseau 20_h Trame de mesure 40_h
2	Changement d'état d'une des deux entrées	Trame de mesure 40_h
3	Changement d'état d'une des deux entrées pendant la transmission de la trame précédente **NOTE:** Temps de garde de 20 s entre deux transmissions	Trame de mesure 40_h
4	Une fois par jour	Trame d'état 30_h

NOTE: les trames transmises sont au format little endian.

NOTE: Une trame de géolocalisation FF_h est générée par le Sigfox Backend à chaque réception d'un message de l'émetteur.

Interprétation des messages JSON

Le tableau suivant explique comment interpréter chaque champ des messages JSON :

Champ JSON	Description	Valeur	Type de trame				
			10_h	20_h	30_h	40_h	FF_h
`device`	ID d'équipement unique Sigfox (ID d'émetteur unique).	4 octets	✓				
`time`	Horodatage GMT (date de réception).	AAAA-MM-JJ**T**HH:MM:SS	✓				-
`avgSnr`	Rapport signal-bruit moyen calculé à partir des 25 derniers messages. **NOTE:** l'équipement doit avoir envoyé au moins 15 messages.	En dB, valeur flottante avec au maximum deux décimales ou N/A. Plus fournie par Sigfox (toujours "null").	✓				-
`data`	Réservé	-	✓				-
`duplicate`	Réservé	-	✓				-
`snr`	Rapport signal-bruit.	En dB, valeur flottante avec au maximum deux décimales.	✓				-
`station`	Identifiant de la station de base Sigfox.	2 octets	✓				-
`lat`	Latitude, arrondie à l'entier le plus proche, de la station de base qui a reçu le message.	En degrés, avec une décimale. Plus fournie par Sigfox (toujours "null").	✓				-
`lng`	Longitude, arrondie à l'entier le plus proche, de la station de base qui a reçu le message.	En degrés, avec une décimale. Plus fournie par Sigfox (toujours "null").	✓				-
`rssi`	Force du signal reçu.	En dBm, valeur flottante avec au maximum deux décimales. S'il n'y a aucune donnée à renvoyer, la valeur est nulle.	✓				-
`seqNumber`	Numérotation de la séquence de trames, le cas échéant.	-	✓				-
`RecordType`	Type de la trame envoyée par l'émetteur Cloud Connected Server. • Trame de configuration produit. • Trame de configuration réseau. • Trame d'état. • Trame de mesure. • Trame de géolocalisation.	• 16 (10_h) • 32 (20_h) • 48 (30_h) • 64 (40_h) • 255(FF_h)			✓		
`FrameCnt1`	Numérotation des trames envoyées par l'émetteur. Incrémentation à chaque transmission. Permet de savoir si une trame a été perdue.	0…7, puis nouvelle boucle	✓				-
`CommandDone`	Réservé	-	✓				-
`HWError`	Réservé	-	✓				-
`LowBatError`	Réservé	-	✓				-
`ConfigOK`	Réservé	-	✓				-
`S1ClosedCnt`	Nombre de fois où l'entrée 1 est passée à l'état fermé depuis sa dernière transmission vers la plate-forme Cloud Connected Sensor.	0…255	-			✓	-
`S2ClosedCnt`	Nombre de fois où l'entrée 2 est passée à l'état fermé depuis sa dernière transmission vers la plate-forme Cloud Connected Sensor.	0…255	-			✓	-
`S3ClosedCnt`	Nombre de fois où l'entrée 3 est passée à l'état fermé depuis sa dernière transmission vers la plate-forme Cloud Connected Sensor.	0…255	-			✓	-

Champ JSON	Description	Valeur	Type de trame				
			10_h	20_h	30_h	40_h	FF_h
S4ClosedCnt	Nombre de fois où l'entrée 4 est passée à l'état fermé depuis sa dernière transmission vers la plate-forme Cloud Connected Sensor.	0…255	-			✓	-
S4PreviousState	État précédent de l'entrée 4 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S4State	État actuel de l'entrée 4 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S3PreviousState	État précédent de l'entrée 3 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S3State	État actuel de l'entrée 3 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S2PreviousState	État précédent de l'entrée 2 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S2State	État actuel de l'entrée 2 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S1PreviousState	État précédent de l'entrée 1 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S1State	État actuel de l'entrée 1 connectée à l'émetteur Cloud Connected Sensor.	1 : ouvert 0 : fermé	-			✓	-
S202	Réservé	Valeur fixe (2).	-	✓	-		
S300	Réservé	Valeur fixe (144).	✓	-			
S301	Réservé	Valeur fixe (72).	✓	-			
S302	Réservé	Valeur fixe (70).	✓	-			
S303	Réservé	Valeur fixe (70).	✓	-			
S304	Réservé	Valeur fixe (70).	✓	-			
S305	Réservé	Valeur fixe (70).	✓	-			
S306	Réservé	Valeur fixe (1).	✓	-			
FrameCnt2	Réservé	Même valeur que FrameCnt1.	✓				-
FrameCnt3	Réservé	Même valeur que FrameCnt1.	✓				-
FrameCnt4	Réservé	Même valeur que FrameCnt1.	✓				-
S1OpenCnt	Nombre de fois où l'entrée 1 est passée à l'état ouvert depuis sa dernière transmission vers la plate-forme Cloud Connected Sensor.	0…255	-			✓	-
S2OpenCnt	Nombre de fois où l'entrée 2 est passée à l'état ouvert depuis sa dernière transmission vers la plate-forme Cloud Connected Sensor.	0…255	-			✓	-
lat	Latitude de l'émetteur qui a transmis la trame.	En Deg - valeur flottante 32 bits.	-				✓
lng	Longitude de l'émetteur qui a transmis la trame.	En Deg - valeur flottante 32 bits.	-				✓

Trame de configuration produit (`RecordType` = 16 (10_h))

Présentation

Pour une trame de configuration produit, `RecordType` est égal à 16 (10_h) dans le message JSON.

Les trames de configuration produit sont envoyées par l'émetteur Cloud Connected Sensor uniquement après leur activation.

La réception d'une trame de ce type pendant le fonctionnement normal du produit peut être interprétée comme le signe d'un comportement anormal.

Exemple de trame de configuration produit


```
{
  "device": "88536A",
  "time": "2018-10-04T13:39:30",
  "avgSnr": "null",
  "data": "100090484646464601",
  "duplicate": "false",
  "snr": "6.00",
  "station": "0FBE",
  "lat": "null",
  "lng": "null",
  "rssi": "-135.00",
  "seqNumber": "1254",
  "RecordType": 16,
  "FrameCnt1": 0,
  "CommandDone": 0,
  "HWError": 0,
  "LowBatError": 0,
  "ConfigOK": 0,
  "S300": 144,
  "S301": 72,
  "S302": 70,
  "S303": 70,
  "S304": 70,
  "S305": 70,
  "S306": 1,
  "FrameCnt2": 0,
  "FrameCnt3": 0,
  "FrameCnt4": 0,
  "SwitchError": 0,
}
```

Trame de configuration réseau (`RecordType` = 32 (20_h))

Présentation

Pour une trame de configuration réseau, `RecordType` est égal à 32 (20_h) dans le message JSON.

Les trames de configuration réseau sont envoyées par l'émetteur Cloud Connected Sensor uniquement après leur activation.

La réception d'une trame de ce type pendant le fonctionnement normal du produit peut être interprétée comme le signe d'un comportement anormal.

Exemple de trame de configuration réseau


```
{
  "device": "88536A",
  "time": "2018-10-04T13:40:06",
  "avgSnr": "null",
  "data": "202002",
  "duplicate": "false",
  "snr": "8.40",
  "station": "0FBE",
  "lat": "null",
  "lng": "null",
  "rssi": "-136.00",
  "seqNumber": "1255",
  "RecordType": 32,
  "FrameCnt1": 1,
  "CommandDone": 0,
  "HWError": 0,
  "LowBatError": 0,
  "ConfigOK": 0,
  "S202": 2,
  "FrameCnt2": 1,
  "FrameCnt3": 1,
  "FrameCnt4": 1,
  "SwitchError": 0
}
```

Trame d'état (`RecordType` = 48 (30_h))

Présentation

Pour une trame d'état, `RecordType` est égal à 48 (30_h) dans le message JSON.

Les trames d'état sont envoyées par l'émetteur Cloud Connected Sensor toutes les 24 heures.

Exemple de trame de configuration réseau


```
{
  "device": "88AC1A",
  "time": "2018-09-28T13:57:01",
  "avgSnr": "null",
  "data": "3060",
  "duplicate": "false",
  "snr": "13.21",
  "station": "05A1",
  "lat": "null",
  "lng": "null",
  "rssi": "-135.00",
  "seqNumber": "723",
  "RecordType": 48,
  "FrameCnt1": 3,
  "CommandDone": 0,
  "HWError": 0,
  "LowBatError": 0,
  "ConfigOK": 0,
  "FrameCnt2": 3,
  "FrameCnt3": 3,
  "FrameCnt4": 3,
  "SwitchError": 0
}
```

Trame de mesure (`RecordType` = 64 (40_h))

Présentation

Pour une trame de mesure, `RecordType` est égal à 64 (40_h) dans le message JSON.

Les trames de mesure sont envoyées par Cloud Connected Sensor chaque fois que l'état de sortie du capteur est modifié.

Exemple de trame de configuration réseau


```
{
  "device": "88AC1A",
  "time": "2018-09-28T13:28:38",
  "avgSnr": "null",
  "data": "40800001000000000000001",
  "duplicate": "false",
  "snr": "8.20",
  "station": "05A1",
  "lat": "null",
  "lng": "null",
  "rssi": "-137.00",
  "seqNumber": "716",
  "RecordType": 64,
  "FrameCnt1": 4,
  "CommandDone": 0,
  "HWError": 0,
  "LowBatError": 0,
  "ConfigOK": 0,
  "S1ClosedCnt": 1,
  "S2ClosedCnt": 0,
  "S3ClosedCnt": 0,
  "S4ClosedCnt": 0,
  "S4PreviousState": 0,
  "S4State": 0,
  "S3PreviousState": 0,
  "S3State": 0,
  "S2PreviousState": 0,
  "S2State": 0,
  "S1PreviousState": 0,
  "S1State": 0,
  "FrameCnt2": 4,
  "FrameCnt3": 4,
  "FrameCnt4": 4,
  "SwitchError": 0
  "S1OpenCnt": 1,
  "S2OpenCnt": 0,
}
```

Trame de géolocalisation (RecordType 255 (FFh))

Présentation

Pour une trame de géolocalisation, `RecordType` est égal à 255 (FF_h) dans le message JSON.

Les trames de géolocalisation sont générées par le Sigfox Backend à chaque réception d'un message de transmetteur.

Exemple de trame de géolocalisation

```
{
  "device": "88AC1A",
  "RecordType": 255,
  "lat": "46.751289126",
  "lng": "1.0254862135",
}
```

Transmission des données par radio

Détails des trames radio

Présentation

L'émetteur envoie quatre types de trame sur le réseau SIGFOX® :

- Une trame d'activation produit (code 10_h).
- Une trame d'activation réseau (code 20_h).
- Une trame d'état (code 30_h).
- Une trame d'événement pour les entrées du capteur (code 40_h).

Détails de la trame de configuration produit transmise

Trame `Data` de configuration produit 10_h :

Nombre d'octets		Données	Description
0		Code de type d'enregistrement	10_h (16)
1	Bit 0	État	00_h
	Bit 1		= 1 si la tension de la batterie est ≤ 2,5 Vcc
	Bit 2		= 1 si une erreur matérielle a été détectée
	Bits 3… 4		Réservés
	Bits 5… 7		Compteur de trames transmises (00_h…07_h)
2		-	90_h
3		-	48_h
4		-	46_h
5		-	46_h
6		-	00_h
7		-	00_h
8		-	01_h
9		-	00_h
10		-	00_h

Détails de la trame de configuration réseau transmise

Trame `Data` de configuration réseau 20_h :

Nombre d'octets		Données	Description
0		Code de type d'enregistrement	20_h (32)
1	Bit 0	État	00_h
	Bit 1		= 1 si la tension de la batterie est ≤ 2,5 Vcc
	Bit 2		= 1 si une erreur matérielle a été détectée
	Bits 3… 4		Réservés
	Bits 5… 7		Compteur de trames transmises (00_h…07_h)
2		-	01_h
3		-	01_h

Nombre d'octets	Données	Description
4	-	00_h
5	-	00_h
6	-	00_h
7	-	00_h
8	-	00_h
9	-	00_h
10	-	00_h

Détails de la trame d'état transmise

Trame `Data` d'état 30_h :

Nombre d'octets		Données	Description
0		Code de type d'enregistrement	30_h (48)
1	Bit 0	État	00_h
	Bit 1		= 1 si la tension de la batterie est ≤ 2,5 Vcc
	Bit 2		= 1 si une erreur matérielle a été détectée
	Bits 3…4		Réservés
	Bits 5…7		Compteur de trames transmises (00_h…07_h)

Détails de la trame de mesure transmise

Trame `Data` de mesure 40_h :

Nombre d'octets		Données	Description
0		Code de type d'enregistrement	40_h (64)
1	Bit 0	État	00_h
	Bit 1		= 1 si la tension de la batterie est ≤ 2,5 Vcc
	Bit 2		= 1 si une erreur matérielle a été détectée
	Bits 3…4		Réservés
	Bits 5…7		Compteur de trames transmises (00_h…07_h)
2…3		Capteur 1	Compteur d'événements (00_h…FF_h)
4…5		Capteur 2	Compteur d'événements (00_h…FF_h)
6…9		Réservés	-

Nombre d'octets		Données	Description
10	Bit 0	État du capteur 1 : ouvert, 2 : fermé	Capteur 1 - État actuel
	Bit 1		Capteur 1 - État dans la trame précédente
	Bit 2		Capteur 2 - État actuel
	Bit 3		Capteur 2 - État dans la trame précédente
	Bit 4		Capteur 3 - État actuel
	Bit 5		Capteur 3 - État dans la trame précédente
	Bit 6		Capteur 4 - État actuel
	Bit 7		Capteur 4 - État dans la trame précédente

Cloud Connected Sensor Data-as-a-Service - Solutions de démarrage

Présentation

Les solutions de démarrage sont constituées d'un code source logiciel documenté, fourni comme une application de console .NET Core. Elles incluent des exemples d'implémentation de la messagerie Publier / S'abonner et des processus d'extraction, de transformation et de chargement.

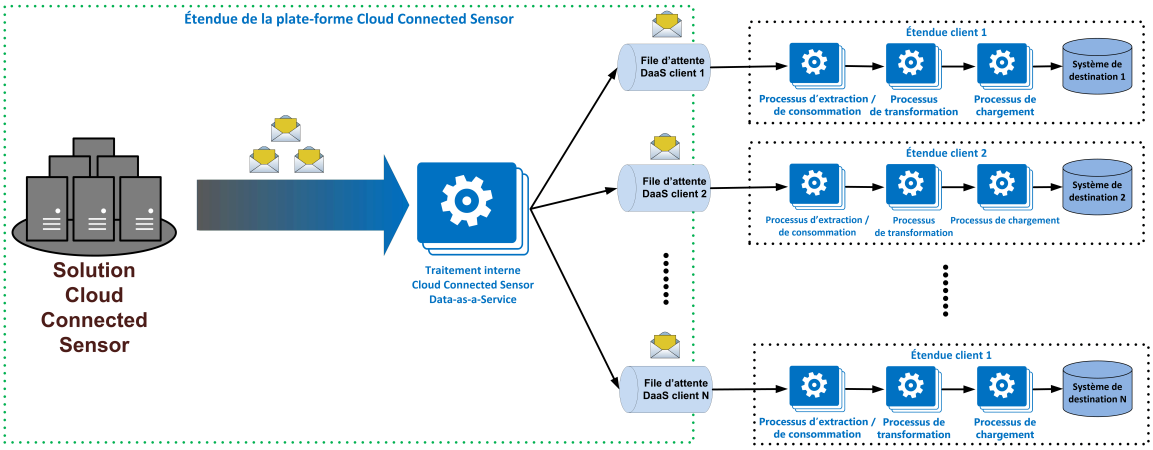
Les solutions de démarrage couvrent deux systèmes de destination :

- Un fichier simple enregistré dans un dossier Windows spécifique
- WonderWare Historian (à ne pas confondre avec WonderWare Online)
- Modbus over TCP

Vue conceptuelle des solutions de démarrage

Présentation

Vue conceptuelle des solutions de démarrage pour l'intégration de Cloud Connected Sensor Data-as-a-Service :



Les solutions de démarrage permettent de mieux comprendre le fonctionnement et l'implémentation de chaque processus. Les solutions de démarrage peuvent être personnalisées afin de consommer, transformer et charger les données dans Historian.

D'un point de vue architectural, toutes les solutions de démarrage sont conçues de la même manière :

- Une première partie, dédiée aux configurations, dans laquelle les connexions au système source (la file d'attente du client DaaS) et au système de destination sont effectuées.
- Une partie centrale, dans laquelle les trois processus ETL (Extract, Transform, Load [Extraire, Transformer, Charger]) sont implémentés.
- Une dernière partie, dans laquelle s'effectue la déconnexion du système source et du système de destination.

En fonction du système de destination, le processus de transformation peut être facultatif. Le processus de transformation est toujours présent dans les solutions de démarrage, mais il peut s'agir d'un opérateur nul.

Toutes les solutions de démarrage sont fournies comme une solution Microsoft Visual Studio autonome, rédigée en langage C# pour une structure .NET Core ou .NET. .NET Core (v2.0) et .NET Framework (v4.6) peuvent être indispensables à l'exécution des solutions de démarrage. Ces packages peuvent être installés à l'aide de Microsoft Visual Studio Installer.

Avant d'exécuter une solution de démarrage, vous devez obligatoirement apporter quelques modifications au code source. Des informations relatives à ce qui doit être modifié sont fournies en fonction de la solution de démarrage.

Fonction de consommation

Cette fonction permet de s'abonner à la file d'attente du client et aux messages de consommation qui en proviennent. L'abonnement à la file d'attente nécessite que le client dispose d'informations sur la file d'attente, notamment le point de terminaison et les identifiants sécurisés pour la lecture. Ces informations sont fournies par la chaîne de connexion.

La fonction de consommation de messages est le même code pour tous les clients. La seule différence d'un client à l'autre est la chaîne de connexion, qui est unique pour chaque client.

Fonction de transformation

Cette fonction sert à extraire l'élément payload du message reçu lors de la consommation du message et à le convertir dans un format exploitable par la fonction de chargement.

Par exemple, cette fonction convertit l'élément payload JSON en un format d'ensemble de données intermédiaire adapté au chargement dans le système Historian du client. L'ensemble de données est ensuite utilisé par la fonction de chargement des données et chargé dans le système Historian du client.

Fonction de chargement

Cette fonction permet de récupérer l'ensemble de données reçu de la part de la fonction de transformation, et de l'insérer dans le système de destination du client. Le processus de chargement est celui qui est le plus spécialisé du système : il doit implémenter la logique ésotérique d'insertion des données du système de destination à l'aide des bibliothèques et des classes natives.

Solution de démarrage pour un fichier simple

Présentation

Cette solution de démarrage est accessible dans le fichier . zip `XIOTDaaSStarterSolution_Visual Studio.zip` et fournit un exemple de consommation, de transformation et de chargement des données dans un fichier JSON d'un système Windows.

Packages NuGet requis, SDK et bibliothèques

Présentation

Cette solution de démarrage est fournie en tant qu'application .NET Core.

C'est pourquoi seuls deux packages doivent être installés. Il s'agit de Microsoft. Azure.ServiceBus (v3.1.0) et de Microsoft.NET Core.App (v2.0).

Configuration préalable à l'exécution

Présentation

Avant d'exécuter cette solution de démarrage, vous devez modifier la valeur de deux éléments locaux constants :

- `ServiceBusConnectionString` doit être mis à jour avec la chaîne de connexion du client (disponible sur le site Web Cloud Connected Sensor, à la page **Services**).

  Par exemple :

  ```
  const string ServiceBusConnectionString = "Endpoint=
  sb://dass-test.servicebus.windows.net/;
  SharedAccessKeyName=RootManageSharedAccessKey;
  SharedAccessKey=slVnJLa7QXMm8qEc9/pGm3xtrs1wKzf
  +q3aImwq1iZI=;EntityPath=dass-queue1";
  ```

- `FilePath` doit être mis à jour avec le chemin du fichier dans lequel seront stockés les messages de données.

  Par exemple :

  ```
  const string FilePath = "C:\\Users\\Public\\DaaSTest.
  json";
  ```

Comportement de la solution de démarrage

Présentation

`XIOTDaaSStarterSolution_Visual Studio` effectuera les actions suivantes :

Étape	Action
1	Se connecter à une file d'attente Azure Service Bus (la file d'attente du client DaaS) à l'aide de la chaîne de connexion fournie dans l'élément local constant `ServiceBusConnectionString`.
2	Consommer les messages de la file d'attente du client DaaS.
3	Afficher les messages reçus dans la console.
4	Stocker les messages reçus tels quels (format JSON) dans un dossier spécifique défini par l'élément local constant `FilePath`.
5	Effacer les messages de la file d'attente.
6	Fermer la connexion à la file d'attente du client DaaS.

Cette solution de démarrage est prévue pour être utilisée comme suit :

Étape	Action
1	Vérifiez que Data-as-a-Service est activé sur la plate-forme Web Cloud Connected Sensor.
2	Configurez la solution de démarrage en mettant à jour les éléments locaux constants (`ServiceBusConnectionString` et `FilePath`).
3	Déclenchez l'envoi de messages en activant l'un des deux contacts secs connectés à l'émetteur XIOT11SERMRCL.
4	Exécutez la solution de démarrage.
5	Les messages envoyés par l'émetteur s'affichent dans une fenêtre de la console. Les messages envoyés par l'émetteur sont également disponibles dans le fichier référencé par `FilePath`.

Solution de démarrage pour WonderWare Historian

Présentation

Cette solution de démarrage est accessible dans le fichier zip `XIOTDaaSStarterSolution_WonderWare.zip` et fournit un exemple de consommation, de transformation et de chargement des données dans un système WonderWare Historian.

Packages NuGet requis, SDK et bibliothèques

Présentation

Cette solution de démarrage est fournie sous la forme d'une application .NET Framework, dans la mesure où elle utilise un package NuGet tiers pour WonderWare Historian.

Pour cette solution de démarrage, deux packages doivent être installés

- Microsoft.Azure.ServiceBus (v3.1.0) et .Net Framework (v4.7.1)
- SDK WonderWare Historian 2017 (v2.0)

En outre, un serveur WonderWare Historian 2017 doit être disponible.

Configuration préalable à l'exécution

Présentation

Avant d'exécuter cette solution de démarrage, vous devez modifier les éléments suivants :

- `ServiceBusConnectionString` doit être mis à jour avec la chaîne de connexion du client (disponible sur le site Web Cloud Connected Sensor, à la page **Services**).

  Par exemple :

  ```
  const string ServiceBusConnectionString = "Endpoint=
  sb://dass-test.servicebus.windows.net/;
  SharedAccessKeyName=RootManageSharedAccessKey;
  SharedAccessKey=slVnJLa7QXMm8qEc9/pGm3xtrs1wKzf
  +q3aImwq1iZI=;EntityPath=dass-queue1";
  ```

- Dans la méthode `setUpWonderwareHistorianConnection()`, les identifiants de connexion Historian doivent être mis à jour :
  - `connectionArgs.ServerName` doit correspondre à la valeur du nom du serveur Historian.
  - `connectionArgs.UserName` et `connectionArgs.Password` doivent être mis à jour avec les identifiants utilisés pour se connecter à WonderWare Historian.

Comportement de la solution de démarrage

Présentation

`XIOTDaaSStarterSolution_File` effectuera les actions suivantes :

Étape	Action
1	Se connecter à une file d'attente Azure Service Bus (la file d'attente du client DaaS) à l'aide de la chaîne de connexion fournie dans l'élément local constant `ServiceBusConnectionString`.
2	Se connecter au serveur WonderWare Historian à l'aide des informations de connexion fournies (`connectionArgs.ServerName`, `connectionArgs.UserName`, `connectionArgs.Password`).
3	Consommer les messages de la file d'attente du client DaaS.
4	Afficher les messages reçus dans la console.
5	Enregistrer les messages reçus tels quels (format JSON) dans le serveur Historian en utilisant le format balisé.
6	Effacer les messages de la file d'attente.
7	Fermer la connexion à la file d'attente du client DaaS et au serveur Historian.

Cette solution de démarrage est prévue pour être utilisée comme suit :

Étape	Action
1	Vérifiez que Data-as-a-Service est activé sur la plate-forme Web Cloud Connected Sensor.
2	Configurez la solution de démarrage en mettant à jour les informations de connexion pour la file d'attente du client DaaS et WonderWare Historian.
3	Déclenchez l'envoi de messages en activant l'un des deux contacts secs connectés à l'émetteur XIOT11SERMRCL.
4	Exécutez la solution de démarrage.
5	Les messages envoyés par l'émetteur s'affichent dans une fenêtre de la console. Les messages envoyés par l'émetteur sont également disponibles dans WonderWare Historian.

Personnalisation du code

La solution de démarrage pour WonderWare Historian est proposée comme un exemple de base. Le code doit être adapté au cas d'utilisation / à l'environnement.

Par exemple, la gestion des balises WonderWare et les propriétés étendues des balises peuvent nécessiter des changements.

NOTE: Cette solution de démarrage a été mise en œuvre afin d'analyser uniquement les trames de mesure récupérées dans la file d'attente du client DaaS. Pour analyser d'autres trames, vous devez modifier le code existant ou suivre une autre méthodologie.

Solution de démarrage pour Modbus over TCP

Présentation

Cette solution de démarrage est accessible dans un fichier zip et fournit un exemple de consommation, de transformation et de chargement des données dans un Modbus TCP server exécuté en local.

Packages NuGet requis, SDK, et bibliothèques

Présentation

Cette solution de démarrage est fournie sous la forme d'une application .NET Framework, car elle utilise un package NuGet tiers pour Modbus TCP Server.

2 fichiers zip de la solution de démarrage sont disponibles :

- Pour la solution App : `XIOT_DaaS_SS_ModbusTCP_App_vx-y.zip`
- Pour la solution Full : `XIOT_DaaS_SS_ModbusTCP_Full_vx-y.zip`

Solution de démarrage App

Toutes les bibliothèques (dll) nécessaires à l'exécution de l'application sont fournies.

Pour cette solution de démarrage, un package doit être installé :

- .Net Framework (v4.7.1)

L'assistant d'installation hors ligne de .Net Framework (v4.7.1) est fourni dans le fichier zip de l'application.

Solution de démarrage Full

Le package fourni inclut :

- le code source de l'application,
- le projet Visual Studio associé,
- les bibliothèques nécessaires (dll),
- le fichier exécutable du programme.

Le code source est personnalisable et son utilisation nécessite une compilation.

Pour cette solution de démarrage, trois packages doivent être installés :

- Microsoft Azure Servicebus (v3.1.0)
- .Net Framework (v4.7.1)
- EasyModbusTCP (v5.5)

Configuration préalable à l'exécution

Présentation

Avant d'exécuter cette solution de démarrage :

- Si vous utilisez la solution de démarrage App, le client doit lancer le programme d'installation `NDP471-KB4033342-x86-x64-AllOS-ENU.exe`.
- Le client doit procéder à l'installation du contenu du package.
- Le client doit mettre à jour les deux fichiers de configuration suivants :

- `CONNECTION_STRING_CFG.txt` sert à enregistrer la chaîne de connexion de la file d'attente DaaS du client. Copiez-collez votre chaîne de connexion dans ce fichier.
- Ensuite, dans le fichier `DEVICE_ID_TABLE_CFG.txt`, le client doit fournir la liste de tous ses émetteurs XIOT11SERMRCL qui enverront des trames. Ce tableau est également utilisé pour mapper les données des émetteurs dans les registres du serveur Modbus TCP.

  NOTE: le format attendu dans ce fichier est un format hexadécimal en lettres majuscules (`00AABBCC`), avec un ID d'émetteur par ligne.

Ensuite, le client lance l'application en cliquant sur le fichier exécutable.

NOTE: Si vous utilisez la solution de démarrage App, le fichier exécutable `XIOTDaaSStarterSolution_ModbusTCP.exe` se trouve dans le dossier `Application\Release`.

Comportement de la solution de démarrage

Présentation

`XIOTDaaSStarterSolution_ModBusTCP` effectuera les actions suivantes :

Étape	Action
1	Se connecter à une file d'attente Azure Service Bus (la file d'attente du client DaaS) à l'aide de la chaîne de connexion fournie dans le fichier de configuration.
2	Configurer un serveur Modbus TCP local.
3	Consommer les messages de la file d'attente du client DaaS.
4	Afficher les messages reçus dans la console.
5	Stocker les données des messages reçus dans plusieurs registres Modbus.
6	Effacer les messages de la file d'attente.
7	Fermer la connexion à la file d'attente du client DaaS et arrêter le serveur Modbus TCP si le client appuie deux fois sur n'importe quelle touche.

Cette solution de démarrage est prévue pour être utilisée comme suit :

Étape	Action
1	Vérifiez que Data-as-a-Service est activé sur la plate-forme Web Cloud Connected Sensor.
2	Configurez la solution de démarrage en mettant à jour les informations de connexion pour la file d'attente du client DaaS et le tableau des ID des équipements (fichiers de configuration).
3	Exécutez la solution de démarrage.
4	Déclenchez l'envoi de messages en activant l'un des deux contacts secs connectés à l'émetteur XIOT11SERMRCL.
5	Les messages envoyés par l'émetteur s'affichent dans une console et les données sont disponibles dans plusieurs registres du serveur Modbus TCP.
6	Le client peut alors initier une communication Modbus TCP avec son système de surveillance / SCADA, afin de lire les données dans les registres correspondants.

Personnalisation du code

La solution de démarrage pour Modbus TCP est proposée comme un exemple de base. Le code doit être adapté au cas d'utilisation / à l'environnement.

Par exemple, il peut être intéressant pour le client de personnaliser le mappage du registre en fonction de ses besoins.

NOTE: Cette solution de démarrage a été mise en œuvre afin d'analyser uniquement les trames de mesure et les trames d'état récupérées dans la file d'attente du client DaaS. Pour analyser d'autres trames, vous devez modifier le code existant ou suivre une autre méthodologie.

Configuration du serveur Modbus et mappage du registre

Le serveur Modbus initié par le programme commence à écouter l'adresse IPv4 active, sur le port 502. Par défaut, l'ID d'unité du serveur Modbus est 1. L'ID d'unité et le port peuvent être modifiés dans le code source. Il est néanmoins impossible de sélectionner l'adresse IPv4 pour le moment.

Les données envoyées par un émetteur sont enregistrées dans les registres du serveur Modbus. À cette fin, l'application utilise uniquement des registres d'exploitation. Le serveur Modbus présente une capacité de 65 535 registres d'exploitation.

Un émetteur est représenté à l'aide de 50 registres. En théorie, cette application peut donc gérer plus de 1 200 émetteurs.

Si besoin, le client peut ajuster le code source afin de sélectionner et de stocker uniquement les données pertinentes.

Pour chaque nouvel émetteur, les registres dans lesquels les données de cet émetteur se trouvent sont décalés de 50. Par exemple, si le client a déclaré les deux ID d'équipement (ID d'émetteur) 00AABBCC et 00DDEEFF dans le fichier `DEVICE_ID_TABLE_CFG.txt`, les données appartenant à l'équipement 00AABBCC se trouvent dans les registres 40001 à 40050, et les données appartenant à l'équipement 00DDEEFF se trouvent dans les registres 40051 à 40100.

Les données sont enregistrées dans les 50 registres comme suit :

Décalage de l'adresse	Données	Format	Attribut JSON correspondant
1	ID d'émetteur	Représentation ASCII des deux premiers caractères de l'ID d'émetteur (**00**AABBCC).	Équipement
2	ID d'émetteur	Représentation ASCII des deux caractères suivants de l'ID d'émetteur (00**AA**BBCC).	Équipement
3	ID d'émetteur	Représentation ASCII des deux caractères suivants de l'ID d'émetteur (00AA**BB**CC).	Équipement
4	ID d'émetteur	Représentation ASCII des deux derniers caractères de l'ID d'émetteur (00AABB**CC**).	Équipement
5	Horodatage	Représentation 16 bits de l'année dans l'horodatage.	Heure
6	Horodatage	Représentation 16 bits du mois dans l'horodatage.	Heure
7	Horodatage	Représentation 16 bits du jour dans l'horodatage.	Heure
8	Horodatage	Représentation 16 bits des heures dans l'horodatage.	Heure
9	Horodatage	Représentation 16 bits des minutes dans l'horodatage.	Heure
10	Horodatage	Représentation 16 bits des secondes dans l'horodatage.	Heure
11	SNR moyen	Représentation 16 bits du SNR moyen de l'émetteur. La valeur est tronquée et représentée comme une valeur non signée. Plus fournie par Sigfox (toujours "null").	avgSnr
12	SNR	Représentation 16 bits du SNR de l'émetteur. La valeur est tronquée et représentée comme une valeur non signée.	snr
13	ID de la station	Représentation ASCII des 2 premiers caractères de la station de base Sigfox qui a reçu le message Sigfox (**AA**BB).	Station
14	ID de la station	Représentation ASCII des 2 derniers caractères de la station de base Sigfox qui a reçu le message Sigfox (AA**BB**).	Station
15	Latitude	Représentation 16 bits de la latitude de la station de base Sigfox. La valeur est tronquée et représentée comme une valeur signée.	lat

Décalage de l'adresse	Données	Format	Attribut JSON correspondant
		Plus fournie par Sigfox (toujours "null").	
16	Longitude	Représentation 16 bits de la longitude de la station de base Sigfox. La valeur est tronquée et représentée comme une valeur signée. Plus fournie par Sigfox (toujours "null").	lng
17	RSSI	Représentation 16 bits du RSSI de l'émetteur. La valeur est tronquée et représentée comme une valeur signée.	rssi
18	Numéro de séquence	Voir la description des trames JSON.	seqNumber
19	Type d'enregistrement	Voir la description des trames JSON.	RecordType
20	Compteur de trames 1	Voir la description des trames JSON.	FrameCnt1
21	Erreur matérielle	Voir la description des trames JSON.	HWError
22	Erreur Batterie faible	Voir la description des trames JSON.	LowBatError
23	Compteur fermé capteur 1	Voir la description des trames JSON.	S1ClosedCnt
24	Compteur fermé capteur 2	Voir la description des trames JSON.	S2ClosedCnt
25	État précédent capteur 2	Voir la description des trames JSON.	S2PreviousState
26	État actuel capteur 2	Voir la description des trames JSON.	S2State
27	État précédent capteur 1	Voir la description des trames JSON.	S1PreviousState
28	État actuel capteur 1	Voir la description des trames JSON.	S1State
29	Compteur ouvert capteur 1	Voir la description des trames JSON.	S1OpenCnt
30	Compteur ouvert capteur 2	Voir la description des trames JSON.	S2OpenCnt
31	Octet de poids fort (MSB) de la latitude de l'émetteur	Représentation flottante 32 bits de la latitude de l'émetteur. Dans ce registre, seuls les 16 premiers bits (octet de poids fort) sont présents.	lat
32	Octet de poids faible (LSB) de la latitude de l'émetteur	Représentation flottante 32 bits de la latitude de l'émetteur. Dans ce registre, seuls les 16 derniers bits (octet de poids faible) sont présents.	lat
33	Octet de poids fort (MSB) de la longitude de l'émetteur	Représentation 16 bits de la longitude de la station de base Sigfox. Dans ce registre, seuls les 16 premiers bits (octet de poids fort) sont présents.	lng
34	Octet de poids faible (LSB) de la longitude de l'émetteur	Représentation 16 bits de la longitude de la station de base Sigfox. Dans ce registre, seuls les 16 derniers bits (octet de poids faible) sont présents.	lng
35	État précédent capteur 3	Voir la description des trames JSON.	S3PreviousState
36	État actuel capteur 3	Voir la description des trames JSON.	S3State
37	État précédent capteur 4	Voir la description des trames JSON.	S4PreviousState
38	État actuel capteur 4	Voir la description des trames JSON.	S4State
39	RÉSERVÉ	RÉSERVÉ POUR USAGE FUTUR	N/A
…	…	…	…
49	RÉSERVÉ	RÉSERVÉ POUR USAGE FUTUR	N/A
50	Drapeau sans état	Valeur définie sur 1 lorsqu'aucune trame (d'état ou de mesure) n'a été reçue pendant les 25 dernières heures pour cet équipement.	N/A

Les informations figurant dans ce document et dans le code source sont susceptibles d'être modifiées. Nous recommandons au client de considérer cet exemple comme un « prototype de code » ou une « démonstration de faisabilité ».

Gestion des données :

- Les clients Modbus/TCP peuvent accéder aux registres du serveur en mode lecture seule.
- Le serveur Modbus/TCP n'enregistre pas l'historique des données.
- Les valeurs de tous les registres sont réinitialisées après chaque démarrage du serveur Modbus/TCP.

Annexes

Contenu de cette partie

Principe de la messagerie Publier / S’abonner et des processus ETL 38
Informations relatives à la technologie utilisée 43

Principe de la messagerie Publier / S’abonner et des processus ETL

Contenu de ce chapitre

Processus ETL....39
Messagerie Publier / S’abonner....41

Processus ETL

Présentation des processus ETL

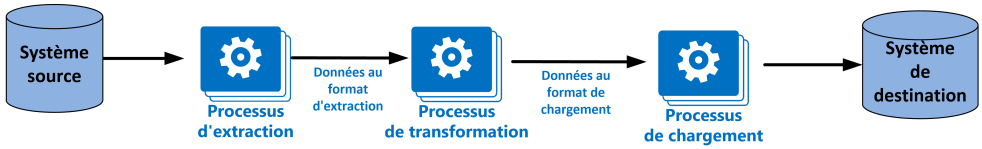
Présentation

Cette section présente les concepts et la terminologie de base que vous devez connaître pour utiliser la solution Cloud Connected Sensor Data-as-a-Service.

La figure suivante détaille les concepts ETL de base, notamment les processus, les systèmes et le flux de données (représenté par des flèches) :

- Le **processus d’extraction** extrait les données du système source et les transforme dans un format d’extraction structuré.
- Le **processus de transformation** transforme les données au format d’extraction en données au format de chargement.
- Le **processus de chargement** charge les données au format de chargement dans le système de destination.

Composants logiques ETL



Quelle que soit la manière dont ces processus sont mis en œuvre, l’extraction, la transformation et le chargement sont des étapes classiques de tous les scénarios d’intégration. Les processus ETL suivent toujours le même ordre : d’abord l’extraction, puis la transformation, et enfin le chargement.

Processus d’extraction

Présentation

Le rôle du processus d’extraction (également appelé processus de consommation) est d’extraire les données du système source et de les récupérer au format d’extraction, exploitable par le processus de transformation.

Le système source peut être n’importe quel type de stockage de données, notamment les suivants :

- Base de données (base de données relationnelle, base de données NoSQL, base de données Graph, etc.)
- SCADA Historian
- Système de fichiers (journaux, sorties analytiques, etc.)
- Système de messagerie

Processus de transformation

Présentation

Il peut être nécessaire de transformer les données du format d’extraction au format de chargement avant de les charger dans un système de destination. C’est le but du processus de transformation.

Le processus de transformation est souvent mis en œuvre dans le cadre du processus de chargement et n’est pas considéré comme un processus distinct. Il est cependant indépendant, quelle que soit la manière dont il est codé ou déployé. Une fois les données transformées dans un format de chargement exploitable par le système de destination (expression compatible avec la base de données, opérations sur le fichier, etc.), elles peuvent être chargées.

Le format d’extraction est nécessaire à la mise à l’échelle du développement, afin de prendre en charge divers systèmes de destination. Dans la mesure où le format d’extraction est indépendant du système de destination, le processus d’extraction n’est couplé à aucune caractéristique du système de destination. Résultat : le processus d’extraction est extrêmement simple et l’architecture d’intégration est extensible, ce qui signifie qu’il est possible de charger un format d’extraction indépendant du système de destination dans un système de destination. Le propriétaire du système de destination ou l’intégrateur système est responsable de la création de processus de transformation et de chargement spécialisés.

En plus de transformer les données du format d’extraction au format de chargement, le processus de transformation doit permettre de valider les données. En effet, des données non valides (mais bien formatées) peuvent engendrer une corruption et une perte de données dans les systèmes de destination.

Voici quelques éléments clés de la validation des données :

- Type de données
- Vérification des plages
- Contraintes
- Intégrité référentielle (bases de données relationnelles, par exemple)
- Structure
- Autorisation des valeurs nulles
- Duplication

À la fin du processus de transformation, vous obtenez un ensemble de données prêt à être chargé dans le système de destination.

Processus de chargement

Présentation

Après la transformation, les données sont lues par le processus de chargement des données, puis chargées dans le système de destination. Le processus de chargement des données exécute des opérations de base de données classiques en s’appuyant sur les données prévalidées et préformatées, avec un traitement réduit au minimum.

Le processus de chargement des données doit être simple et robuste. Si le chargement des données ne se déroule pas correctement, vous pouvez supposer que cela est dû à un motif externe (ce que les journaux devraient prouver), par exemple un comportement incorrect du processus de transformation, une coupure du réseau ou de la connexion, ou encore un problème lié au système de destination (disque plein, par ex.).

L’enregistrement de l’activité dans un journal est souvent requis dans le cadre des processus de transformation et de chargement, à des fins d’audit.

Messagerie Publier / S’abonner

Présentation

La messagerie Publier / S’abonner (ou tout simplement, la messagerie) est une stratégie d’intégration Push dans laquelle un éditeur de données pousse préventivement les messages de données vers un client. La messagerie offre de nombreux avantages en matière d’intégration. À l’inverse, elle présente peu d’inconvénients. Elle est à la fois robuste, éprouvée et sécurisée dans le cadre d’une communication d’entreprise à entreprise.

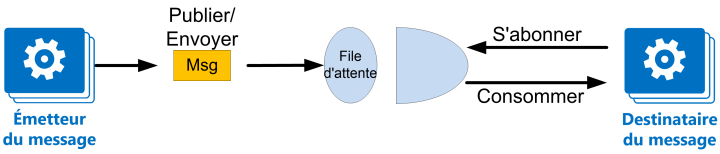
Concept de base

Présentation

Le concept de base de la messagerie est le suivant :

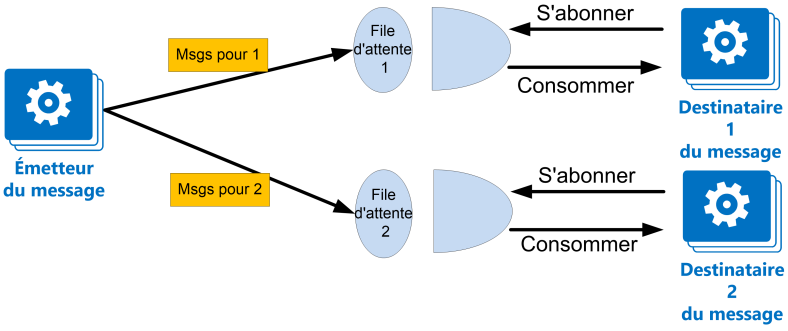
- L’éditeur héberge un mécanisme de stockage des données et de communication. C’est ce qu’on appelle une file d’attente.
- Le client s’abonne à cette file d’attente.
- L’éditeur des données pousse les données dans cette file d’attente sous la forme d’un message.
- Le client reçoit le message et traite les données.

Messagerie Publier / S’abonner :



Dans les scénarios publics, la file d’attente est une ressource réseau disponible au public, par exemple un fil d’actualités ou un flux météo, accessible à tous les clients qui veulent recevoir des messages de sa part. Dans un scénario d’intégration Data-as-a-Service, la file d’attente n’est pas disponible au public, et seuls les clients DaaS peuvent s’abonner à la file d’attente. De plus, les clients DaaS reçoivent uniquement les messages qui leur appartiennent ; ils ne peuvent pas recevoir les messages appartenant à d’autres clients.

Publier / S’abonner avec plusieurs clients :



Chaque client (ou client DaaS) s’abonne à une file d’attente logique qui lui est dédiée. Il reçoit uniquement les messages envoyés à sa file d’attente. Les clients n’ont aucune visibilité sur les files d’attente de messagerie des autres clients, ne disposent pas d’identifiants pour y accéder et ne peuvent pas s’y abonner.

Remarque relative à la cybersécurité

Présentation

Le client de messagerie utilise des connexions réseau sortantes vers la file d’attente de messagerie et n’a pas besoin d’exposer un point de terminaison de réseau public. Les risques de cybersécurité sont donc limités pour l’intégration basée sur la messagerie, en particulier pour les clients qui consomment les messages sur site.

Dans les architectures de messagerie, l’éditeur du message (les capteurs Schneider Electric / Telemecanique dans le cas de Cloud Connected Sensor) assume le risque pour la sécurité : hébergement, sécurisation, gestion et exposition de la ressource réseau (la file d’attente).

Pour recevoir ses messages, le client de messagerie doit lancer une session dans la file d’attente et en faire la demande. Plutôt que de connecter la file d’attente au client de messagerie via un point de terminaison de réseau public, le client de messagerie lance une connexion sortante depuis son réseau vers le point de terminaison du réseau de la file d’attente. Malgré quelques différences, ce fonctionnement est similaire à celui d’un client Web qui se connecte à un serveur Web pour demander le contenu d’un site Web. Le risque lié à la sécurité du réseau est donc limité pour le client.

Il est également possible pour le client de messagerie de se connecter à la file d’attente pour une durée courte avant de se déconnecter et de se reconnecter. Cela réduit encore davantage le risque de cyberattaque, dans la mesure où le client de messagerie utilise une séquence non déterminante de ports en sortie volatils.

Informations relatives à la technologie utilisée

Contenu de ce chapitre

Microsoft Azure Service Bus 43
Microsoft Visual Studio 43

Microsoft Azure Service Bus

Présentation

La file d'attente du client DaaS est une file d'attente Azure Service Bus. Pour plus d'informations sur cette technologie, consultez le site Web de Microsoft.

Voici un lien vers la documentation relative à la messagerie Azure Service Bus.

https://docs.microsoft.com/en-us/azure/service-bus-messaging/

Vous trouverez sur cette page toutes les informations clés concernant Azure Service Bus :

- Qu'est-ce qu'Azure Service Bus ?
- Fonctionnement
- Didacticiels et exemple de code pour interagir avec Azure Service Bus
- Autres informations clés

Microsoft Visual Studio

Présentation

Les solutions de démarrage sont fournies sous la forme d'applications pour la console .NET Core Microsoft Visual Studio. Pour plus d'informations sur cet environnement de développement intégré, consultez le site Web de Microsoft.

Voici deux liens vers la documentation relative à Visual Studio :

- Informations générales sur Visual Studio : https://visualstudio.microsoft.com/
- Documentation Visual Studio : https://docs.microsoft.com/en-us/visualstudio

Glossaire

F

Format d’extraction:

Format dans lequel le processus d’extraction écrit les données pour leur exploitation par le processus de transformation.

Format de chargement:

Format dans lequel le processus de transformation écrit les données pour leur exploitation par le processus de chargement.

J

JSON:

JavaScript Object Notation est un format de données léger, lisible et structuré, utilisé pour le transfert de messages entre les équipements et le cloud, et entre le cloud et les équipements. De nombreuses applications de traitement du flux sont conçues pour consommer les structures JSON en natif.

P

Partenaires d’intégration:

Partenaires impliqués dans une solution d’intégration. Ils peuvent être appelés partenaires source ou partenaires de destination. Dans le cas présent, Cloud Connected Sensor est le partenaire source et le client DaaS est le partenaire de destination.

Processus d’extraction:

Processus qui extrait les données du système source et les écrit au format d’extraction.

Processus de chargement:

Processus qui charge les données au format de chargement dans le système de destination.

Processus de transformation:

Processus qui transforme les données au format d’extraction en données au format de chargement.

Processus ETL:

Les trois processus mis en œuvre par le consommateur de messages :

- Processus d’extraction
- Processus de transformation
- Processus de chargement

S

Système de destination:

Système détenu par le partenaire de destination et dans lequel les données au format de chargement sont chargées par le processus de chargement.

Système source:

Système détenu par le partenaire source et qui contient les données d’origine extraites par le processus d’extraction.

www.tesensors.com

En raison de l’évolution des normes et du matériel, les caractéristiques indiquées par les textes et les images de ce document ne nous engagent qu’après confirmation par nos services.

TMSS France
SAS capital: 366 931 214 €
Tour Eqho, 2 avenue Gambetta,
92400 Courbevoie – France
908 125 255 RCS Nanterre

©2024, TMSS France, Tous droits réservés

TESEUG000310FR
09/2024