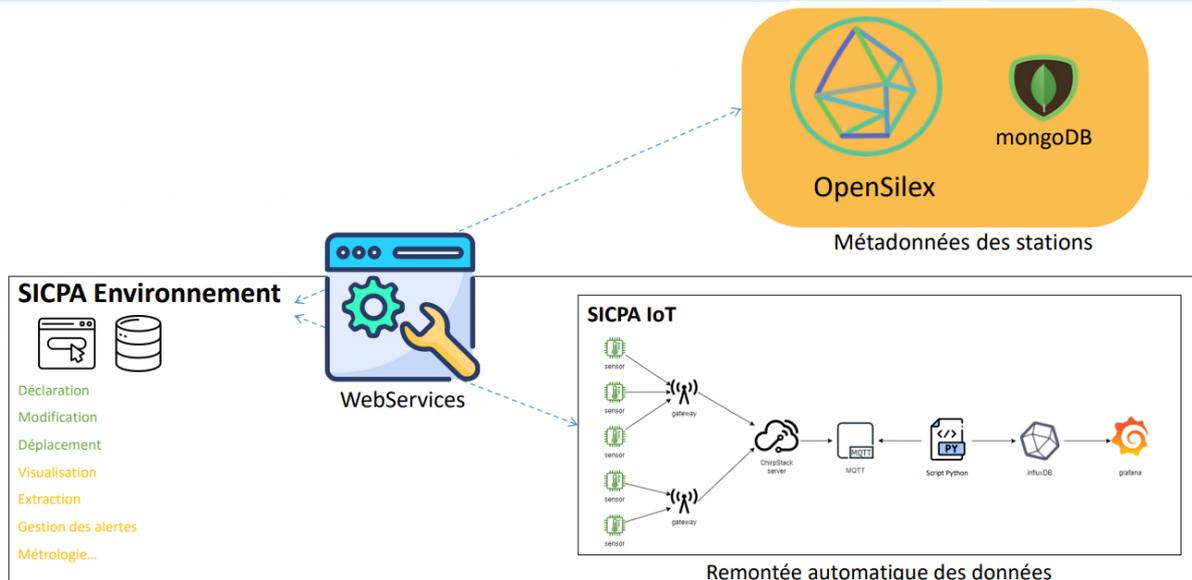


Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024
		Rédacteur(s) : Adrien Pasquesoone

Fonctionnement :



Localisation des différents éléments :

Éléments sur la machine <https://sicpa-iot.cati.inrae.fr> :

- Base de données environnement
- Chirpstack et MQTT associé
- Script Python dans /data/scripts/python/iot.telegraf.python/
- Base influxDB
- Grafana

Appli web Environnement sur les machines :

- <https://sicpa-webapps-preprod.cati.inrae.fr/SicpaEnvironnementWeb/> (preprod)
- <https://sicpa-webapps.cati.inrae.fr/SicpaEnvironnementWeb/> (prod)

API Environnement sur les machines interop1 et interop2

Instance OpenSilex accessible à l'adresse : <http://147.100.203.92:8081/>

Les repos Git pour l'API et l'appli Web Environnement sont disponibles sur la Forge dans le projet *SICPA Environnement*

Les repos Git de sicpa-iot et plus précisément le script Python sont disponibles sur la Forge dans le projet *IoT* :

Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024 Rédacteur(s) : Adrien Pasquesoone

<https://forge-dga.jouy.inra.fr/projects/iot/repository/sicpa-iot-telegraf-python>

Projet Python

Pour travailler en Python il faut un venv. Premièrement il vous faut installer la librairie :

```
pip install virtualenv
```

Pour créer un venv, se positionner au même endroit que le fichier main puis :

Lancer la commande :

```
python<version> -m venv <virtual-environment-name>
```

Puis activer le venv

```
source env/bin/activate
```

Téléchargez les librairies dépendantes du projet

```
Pip install -r requirements.txt
```

Nouveau site géographique :

Dans Chirpstack :

- Créer l'application, conservez temporairement l'applicationId créé

Dans influxDB :

- Créer une base de données pour ce site
- Créer un utilisateur en écriture pour cette nouvelle base
- Créer un utilisateur en lecture pour cette nouvelle base

Dans Script Python :

- Ajouter dans le fichier
/data/scripts/python/iot.telegraf.python/configuration/configuration.ini : l'applicationID, le nom de la nouvelle base de données créée, le nom d'utilisateur en écriture, le mot de passe de l'utilisateur en écriture ; de la même manière que les autres sites déjà renseignés dans ce fichier

Dans Grafana :

- Créer une organisation pour ce nouveau site
- Ajouter la datasource correspondant à l'utilisateur en lecture de la nouvelle base de données créée, conservez temporairement l'uid de la datasource
- Créer un compte de service et un token associé à ce compte dans l'onglet « *Administration/Utilisateurs et droits d'accès/Comptes de service* », conservez temporairement ce token

Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024
		Rédacteur(s) : Adrien Pasquesoone

Dans la base de données Environnement :

- Dans la table CorrespondanceSite, ajouter une ligne avec : l'IdSiroco* correspondant au site géographique, l'ApplicationId et l'uid conservés précédemment

*vous pouvez retrouver l'IdSiroco associé au site géographique grâce à l'API Siroco à cette adresse : <https://sicpa-interop.cati.inrae.fr/Siroco/doc/>

Dans l'API Environnement :

- Dans le fichier `src/main/java/fr/inrae/sicpa/helper/Constantes.java` : ajoutez, de la même manière que les autres, le nouveau token Grafana créé précédemment et l'ajouter à la liste `GRAFANA_KEYS`

Nouveau bâtiment :

Pour les bâtiments se référer au fichier `Batiments_zones` du Nextcloud. Les bâtiments/zones/sous-zones sont à renseignés au niveau de la BD Siroco, pour les accès, demander à Alexandre Journaux.

Il faut déclarer dans l'ordre suivant :

- Dans la table `CS_BATIMENT` : les bâtiments associés à un site
- Dans la table `CS_ZONE` : les zones associées aux bâtiments
- Dans la table `CS_SOUS_ZONE` : les sous-zones associées aux zones

Faire attention à bien déclarer dans les 2 bases (recette et prod) !

Folder Grafana :

Pour le bon fonctionnement de la création automatique des dashboards dans Grafana à partir de l'appli Web, les folders doivent respecter un nommage.

Ce nommage est lié aux différents élevages présents sur le site géographique.

Il faut donc que les éléments de la colonne `ELEVAGE_NOM_ENVIRONNEMENT` de la table `CS_ELEVAGE` de la base Siroco correspondent aux noms des Folders de Grafana.

Nous pouvons modifier ces noms s'ils ne conviennent pas à la réalité, mais ils doivent correspondre au nom de l'élevage dans Siroco.

Ce qu'il doit se passer quand un utilisateur crée une station

Dans Chirpstack

Station créée dans Chirpstack dans l'application correspondante au site géographique renseigné dans le formulaire

Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024
		Rédacteur(s) : Adrien Pasquesoone

Dans OpenSilex

Station créée dans « Organisation Scientifique » -> Appareils

Événement de déplacement créé dans la liste des événements de la station

Événement d'association avec les capteurs créé dans la liste des événements de la station

Capteurs créés dans « Organisation Scientifique » -> Appareils

Dans la base Environnement

Station créée dans la table Station avec l'URI OpenSilex associé ; l'URI de déplacement est également associé

Capteurs créés dans la table Capteur avec les URIs OpenSilex associés

Dans Grafana

Si l'utilisateur a coché la case « Créer un dashboard sur Grafana », dans l'organisation associée au site géographique sélectionné dans le formulaire, dans le folder associé à l'élevage sélectionné, un nouveau dashboard avec le nom de la station doit apparaître.

Ce qu'il doit se passer quand un utilisateur déplace une station

Dans Chirpstack

Si le site géographique a changé alors, la station ne doit plus apparaître dans l'ancienne application et doit apparaître dans la nouvelle application

Si la station a été déplacé dans le bâtiment « Stockage », la station doit être disable.

Dans OpenSilex

Un nouvel événement de déplacement doit être créer

Si la station a été déplacé dans le bâtiment « Stockage » alors un événement d'arrêt doit également être créé

Dans la base Environnement

L'URI OpenSilex associé à la station doit avoir changé et être celle du nouvel événement de déplacement.

Ce qu'il doit se passer quand un utilisateur crée un événement

Dans Chirpstack

Si c'est un événement d'arrêt alors la station doit être disable

Si c'est un événement de redémarrage alors la station doit être enable

Dans OpenSilex

L'événement associé au type sélectionné dans le formulaire est créé

Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024
		Rédacteur(s) : Adrien Pasquesoone

Localisation des stations :

Lors de la remontée des données de MQTT vers influxDB, dans le script python, nous faisons appel à l'API Environnement pour récupérer la localisation de la station. Pour ce faire nous avons besoin d'un token, il faut donc renouveler le token du script Python dans ce fichier :

- `/data/scripts/python/iot.telegraf.python/configuration/configuration.ini` dans la section `API_ENVIRONNEMENT` la valeur de `JWT_Token`

Ce token est à renouveler manuellement tous les ans. Token actuel valide jusqu'au **24/06/2025**

Nouveau modèle de station :

Dans Chirpstack :

- Faire le développement du Codec dans l'application « dev » (id=16, la remontée des données sera visible dans l'organisation `SICPA_IOT_Dev` sur Grafana)
- Créer un profile respectant la nomenclature suivante : `marque-nom_modele`. Dans le codec de ce profile, il faut respecter un certain nombre de règles.

Les éléments du codec commençant par :

- o « `t_` » seront stockés dans influxDB comme tags pour la base utilisateur
- o « `m_` » seront stockés dans influxDB comme des fields pour la base maintenance
- o « `mt_` » seront stockés dans influxDB comme des tags pour la base maintenance
- o « `d_` » seront stockés dans influxDB comme des fields pour les deux bases
- o « `dt_` » seront stockés dans influxDB comme des tags pour les deux bases
- o Par défaut, les éléments sont stockés comme fields dans la base utilisateur

Veillez à réutiliser un maximum les éléments déjà utilisés dans les autres codecs, par exemple `d_percentBattery`, `temperature...` Si vous ne trouvez pas, veuillez respecter la nomenclature : en anglais en lowerCamelCase

Dans le script Python :

- Si une information transmise en dehors du champ « objet » de la trame MQTT est à stocker dans la base influxDB, veuillez alors compléter le fichier `/data/scripts/python/iot.telegraf.python/configuration/infos_trame_mqtt.json` dans les rubriques `user.fields`, `user.tags`, `maintenance.fields` et `maintenance.tags` respectivement selon de quel manière vous voulez stocker l'information. Si vous souhaitez que ça aille dans les deux bases, il faut alors renseigner dans les deux champs. Il faut renseigner sous forme de chemin l'accès à l'information dans la trame MQTT.

Exemple :

- o Exemple de trame :


```
{
  "applicationID": "12",
  "applicationName": "Guadeloupe-duclos",
  "deviceName": "seeed-s2101-0776",
  "deviceProfileName": "seeed-s2101",
  "deviceProfileID": "c1306695-d892-4dc5-85da-9393c95367f7",
  "devEUI": "2cf7f1c044300776",
```

Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024
		Rédacteur(s) : Adrien Pasquesoone

```

"rxInfo": [
  {
    "gatewayID": "5031395312414750",
    "uplinkID": "d9023eb8-ef1d-4512-ba6e-a3548219bd78",
    "name": "mkt-ltap-lte6-kit-duclos-414750",
    "time": "2024-06-17T12:52:00.638653Z",
    "rssi": -108,
    "loRaSNR": 4.25,
    "location": {
      "latitude": 16.200400558194715,
      "longitude": -61.66556789785327,
      "altitude": 0
    }
  }
],
"txInfo": {
  "frequency": 868500000,
  "dr": 5
},
"adr": true,
"fCnt": 19581,
"fPort": 2,
"data": "AQEQno4AAAECEBL3AAA0Pg==",
"object": {
  "bytes": "0101109e8e000001021012f70000343e",
  "humidity": 63.25,
  "temperature": 36.51,
  "valid": true
}
}

```

- Si nous voulons stocker : le champ *data* en field de maintenance, le champ *frequency* en tag de maintenance, et le champ *name* en tag des deux bases

Alors le fichier

/data/scripts/python/iot.telegraf.python/configuration/infos_trame_mqtt.json

ressemblerait maintenant à ceci :

```

{
  "time": "[\"rxInfo\"][0][\"time\"]",
  "devEUI": "[\"devEUI\"]",
  "object": "[\"object\"]",
  "user": {
    "fields": [

    ],
    "tags": [
      "[\"deviceName\"]",
      "[\"rxInfo\"][0] [\"name\"]"
    ]
  }
}

```

Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024
		Rédacteur(s) : Adrien Pasquesoone

```

},
"maintenance": {
  "fields": [
    ["rxInfo\"][0]["rssi\"],
    ["data\"]
  ],
  "tags": [
    ["applicationName\"],
    ["deviceName\"],
    ["txInfo\"]["frequency\"],
    ["rxInfo\"][0] ["name\"]
  ]
}
}
}

```

Dans Grafana :

- Créer un dashboard, qui servira de référence pour ce modèle, avec une station de ce nouveau modèle en suivant les autres templates de modèle déjà existant.
- Exporter le json de ce dashboard
- Remplacer :
 - o la valeur de *id* par *null*
 - o toutes les occurrences de *uid* de la datasource par *template_datasource*
 - o toutes les occurrences du nom de la station par *template_nomStation*
 - o la valeur de *uid* par ""
 - o le numéro de la *version* par 0
- Sauvegardez temporairement ce json modifié et vous pouvez supprimer le dashboard de Grafana si vous le souhaitez

Dans la base Environnement :

- Créer une nouvelle ligne dans la table *ModeleStation* en renseignant la marque, le nom du modèle, et le template Grafana créé précédemment. Il faut veiller à ce que la marque et le nom du modèle renseignés soient identiques à ceux dans Chirpstack. Sauvegarder l'id de la ligne créée.
 - Pour chaque capteur de la station regardez si la mesure est présente dans la table *TypeMesureCapteur* :
 - o Si oui, sauvegarder l'id de la ligne du type de la mesure puis créer une ligne dans la table *ModeleCapteur* en associant l'id du modele de la station à l'id du type de mesure.
 - o Si la mesure n'existe pas, il faut d'abord créer une nouvelle variable sur OpenSilex (pour ceci voir la rubrique suivante), puis créer une nouveau type de mesure en renseignant le nom du type de mesure, l'URI OpenSilex de l'appareil de mesure, le nom anglais de l'appareil, la propriété *vocabulary:measures* ainsi que l'URI de la variable créée.
- On peut enfin créer le nouveau modèle de capteur dans la table *ModeleCapteur*

Inrae - Cati Sicpa Systèmes d'Informations et Calcul pour le Phénotypage Animal	Environnement	Code : Sicpa-Environnement-2024-08-08
	Documentation technique	Date : 08/08/2024
		Rédacteur(s) : Adrien Pasquesoone

Dans OpenSilex :

- Dans l'onglet *Appareils* de la section *Vocabulaire*, regarder si un appareil correspond au capteur, sauvegarder son URI ainsi que son nom en anglais (disponible en changeant de langue en haut à droite de l'écran), sinon il faut le créer.
- Dans l'onglet *Variables* de la section *Informations scientifiques*, dans la page *Entité* vérifier qu'une entité est adapté au capteur, sinon la créer, de même que pour la page *Caractéristique* et *Unité/Echelle*
- Créer enfin la variable avec les éléments nouvellement créés et sauvegarder son URI
- Revenir à la partie création de type de capteur

Transfert de l'historique

Pour transférer l'historique, il faut avoir le code disponible sur la forge à ce lien :

<https://forge-dga.jouy.inra.fr/projects/iot/repository/sicpa-iot-migration-python>

Il faut changer le site géographique de destination à la ligne 18 du fichier *main.py*

Dans le fichier *configuration/correspondance_noms_stations.json* il faut renseigner les capteurs qu'il faut transférer, vous avez une liste complète de toutes les stations dans le fichier *configuration/correspondance_noms_stations_full.json*

Il faut vous assurer que les fichiers présents dans le dossier *configuration/profiles* sont corrects. Ces fichiers servent à faire la correspondance des anciens noms de fields pour les nouveaux measurements. On peut notamment retrouver les nouveaux noms dans les codecs des profiles sur Chirpstack.