

tags: DAISEE Atlansun SunShare bus sémantique

[Atlansun] Propositions de SunShare & Data-Players pour le hackathon (<https://frama.link/hackathon-atlansun-propositions>) - Challenge “Autoconsommation collaborative”

0- À propos

Ce document - complémentaire du carnet de bord (<https://frama.link/hackathon-atlansun-logbook>) - réunit les propositions de Simon Louvet (Assemblée Virtuelle, Data-Players, Bus Sémantique), J. Moreau (Sunshare (<http://sunshare.fr>)), G. Moreau (SYDELA) pour le hackathon Atlansun (<https://www.eventbrite.fr/e/billets-numerisons-le-solaire-hackathon-conference-47162413047>) (organisé à Nantes, les 15 et 16 Septembre 2018).

1- Propositions

Sunshare, le SYDELA et ENERCOOP ont **besoin** de connaître plusieurs flux de données concernant l’autoconso et la production des consommateurs pour des finalités différentes.

Etat des intentions

- **Sunshare** : souhaite développer un portail personnel pour autoconsommateurs et un portail d’agrégation de l’électricité produite / consommée de la communauté. Communiquer le statut de fonctionnement (injection, soutirage, autoconsommation) (**Note 1**)
- **Data-Players** : veut démontrer la pertinence du “bus sémantique” pour les opérations de manipulation de données de l’énergie (notamment dans l’autoconso. collective).
- **SYDELA** : information des collectivités du suivi de leurs projets de production / autoconsommation et état de la tension du réseau public aux points de livraison (**Note 2**).
- **Enercoop / Alecs** : Est dans l’obligation de proposer un affichage de la consommation des clients sous LINKY (API ENEDIS), ou affichage déporté de la consommation (Article L337-3-1 du code de l’énergie). Réflexions sur l’autoconsommation collective d’énergie en cours.

Tâches envisagées

- Tache 1 : Déployer et/ou développer des outils pour se connecter à l’API d’ENEDIS pour récupérer les données brutes afin de les transformer en donnée “haut niveau” (métier). Permettre aux particuliers de consulter les informations sur le portail, de télécharger leur courbe de charge. Envisager avec ENEDIS les conditions pour la remontée par LINKY

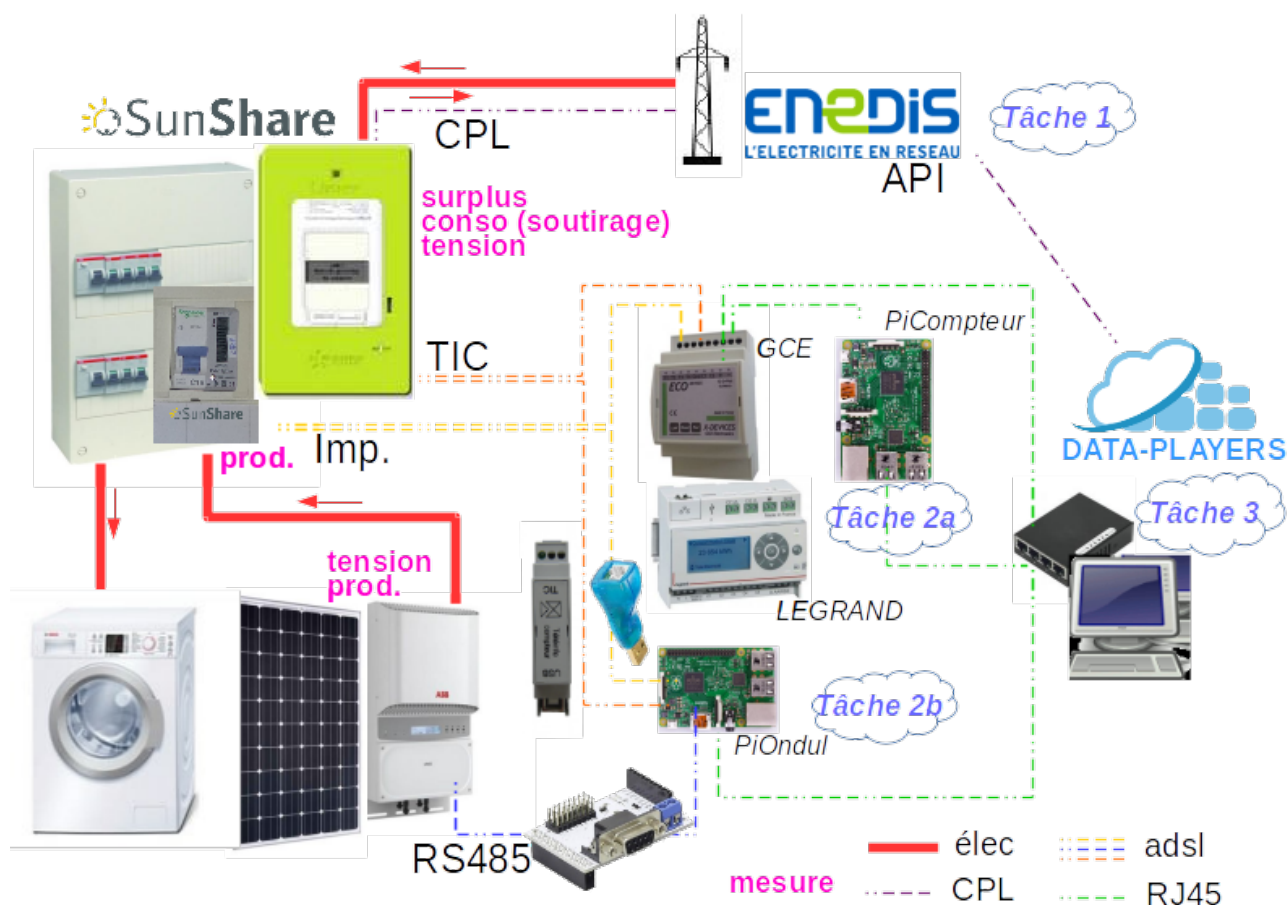
d'informations de production (en provenance de l'onduleur ou d'un compteur à impulsion)

Une prochaine mise à jour du portail présentera l'injection sur le réseau. La remontée d'information par LINKY n'est pas envisageable selon ENEDIS. Sujet à reconsidérer ?

- **Tache 2** : Déployer et/ou développer un module en capacité de lire, en temps réel ou à J+1, les sorties numériques "bas niveau" des compteurs LINKY (particuliers (https://www.enedis.fr/sites/default/files/Enedis-NOI-CPT_54E.pdf), PME/PMI (https://www.enedis.fr/sites/default/files/Enedis-NOI-CPT_02E.pdf)) pour mettre à disposition de la donnée "haut niveau" (métier). Ce module devra recevoir l'information en provenance d'un sous compteur pour la partie "production" ou "consommation" aval compteur. Cette équipe pourrait être située chez J. Moreau pour profiter d'une installation consom/prod et de son LINKY. Des briques existent ici (<https://geo80blog.wordpress.com/2017/02/25/recuperer-les-informations-du-compteur-edf-avec-un-esp8266/>) ou ici (<https://www.jeedom.com/market/index.php?v=d&p=market&type=plugin>) et ici (https://github.com/jeedom/documentation/blob/master/installation/fr_FR/index.md).

Le matériel et le programme sont détaillés ici : <https://pad.lamyne.org/s/H1AyUjml7#>
(<https://pad.lamyne.org/s/H1AyUjml7#>)

- Tester plusieurs **architectures matérielles** (ecodevice+raspberry, legrand+raspberry, ecodevice seul, TIC+RS485+Raspberry)
- Tester plusieurs **architectures logicielles** (raspbian, raspbian lite, jeedom)
évaluer les options de réduction de puissance (raspberry pi 0, Orange Pi...)
- Faut-il installer un sous compteur de consommation pour boucler la mesure ?



“Montage du rack d’essai”

- **Tâche 3** : Déployer et/ou développer les outils de traitement et de visualisation des données issues des équipes 1 et 2. Des briques se trouvent dans les liens ci-dessus.

Une esquisse de rendu à télécharger ici (https://drive.google.com/file/d/1oZZQmFVfda911wctKC2bIT_bm6ReN1P9/view?usp=sharing)

- **Bonus** : construire la page personnelle d’un particulier autoconsommateur avec les paramètres et le graphe à installer sur raspberry pi, Exporter (en sécurité) vers un serveur data-players, Développer une application jeedom pour faciliter l’installation à domicile.

3- Ressources

- Schémas (https://drive.google.com/file/d/1_uIPlgvEiESTSxOeNZmPQ9tcHQYv29Jj/view?usp=sharing)
- Montage (http://perso.sunshare.fr/montage_SunShare.png)
- Matériels (<https://pad.lamyne.org/MYFghgJgHA7AjBAtAZgiJIAMrFQGxRSLABmMmccAnKCAEwRA>)
- Esquisse (https://drive.google.com/file/d/1oZZQmFVfda911wctKC2bIT_bm6ReN1P9/view?usp=sharing)

4- Licences

La licence mentionnée pour les notices est CC-BY-SA. Développements sous GNU3.

5- Notes

(Note 1) Mesure et remontée de la production : le compteur électrique ne compte que le flux qui arrive du réseau chez le consommateur (flux soutiré) ou le flux qui sort du producteur/consommateur vers le réseau (flux injecté). Hors ce flux perçu par le réseau est constitué en permanence de la différence des flux de prod. et de conso. en aval compteur. Pour reconstruire la courbe de charge d'autoconsommation de l'utilisateur il est nécessaire de coupler les données fournies par le compteur avec une donnée tierce qui mesure uniquement le flux de prod. ou de conso. en aval compteur (chez l'utilisateur). (cf. schéma (https://drive.google.com/file/d/1_uIPlgvEiESTSxOeNZmPQ9tcHQYv29Jj/view?usp=sharing))

(Note 2) Mesure de la tension : ENEDIS doit respecter un **cadre réglementaire** (<https://www.cre.fr/Electricite/Reseaux-d-electricite/Qualite-de-l-electricite>) qui spécifie la qualité de l'électricité fournie chez les usagers. Ce cadre stipule, entre autre chose, que la tension nominale efficace aux bornes des compteurs usagers doivent être compris entre 220 et 240 V en monophasé (autour de 400 V en triphasé). L'objectif de ce cadre est de garantir le bon fonctionnement de l'ensemble des appareils branchés chez les usagers en aval compteurs. Dans le cas d'un (consommateur et) producteur, ENEDIS doit donc s'assurer, avant la mise en oeuvre de la production chez le client, que l'ajout d'une production ne va pas venir mettre en contrainte (faire dépasser le niveau de tension max) le réseau au point de livraison, au compteur (http://www.photovoltaique.info/IMG/pdf/ESPRIT_Raccordement_des_centrales_PV_au_RPD_BT_en_France.pdf). Cette contrainte conduit régulièrement ENEDIS a déclaré des projets non faisables (c'est très rare chez les "petits producteurs" < 6kWc mais courant pour les plus gros projets), c'est à dire avec un niveau de puissance maximum de la production qui conduirait à un niveau d'élévation de tension au point de livraison non compatible avec le respect du décret qualité. Néanmoins les moments où cette contrainte surviendrait sont souvent extrêmement réduits dans le temps (autour du 15 aout classiquement, la production est maximum, la consommation minimum), mais la contrainte conduit quand même à sous dimensionner le projet (j'ai la place pour installer 80 kWc mais la contrainte me dit pas plus de 20 kWc, donc j'installe du PV que sur 1/4 de mon toit).

Mettre en oeuvre une stratégie de récupération du niveau de tension grâce à la sortie TIC du compteur permettrait donc de développer une boucle de rétroaction sur les onduleurs de la centrale PV pour brider en direct la production et donc l'élévation du niveau de tension pour les quelques moments où cela serait nécessaire, sans limiter le productible.

La tension peut aussi être rapportée par l'onduleur qui adapte son point de fonctionnement à la tension du réseau.

5- Session de travail 15/09

- Contexte (rappel) : les données de l'API Enedis peuvent suffire pour un service de base mais pas de possibilité de voir les consommations et productions en temps réel.

- Besoin : traiter les données du Linky, y compris les données de l'API et enrichir avec des données complémentaires qui ne sont pas encore dans l'API.
Notamment sortir la puissance consommée.
Développement d'un module individuel d'interprétation des données de consommation et production.
- Réalisé en amont :
 - Branchement à l'**Eco-device** et remontée des données de l'API dans le bus sémantique. Limite : pas d'opensource.
 - Rencontre avec **GCE** (fabricant de l'Eco-Device et du RT2).
Note : GCE propose de réaliser un shield qui permettrait de rassembler le matériel de la tâche 2.b. et permettrait de réaliser les calculs sans processeur. Cependant, GCE ne travaillerait pas dans une logique Open Source : les plans du shield resteraient à la propriété de GCE. Néanmoins, compte-tenu de leur savoir-faire & dispositions matériels, GCE permettrait de réaliser le shield de façon industrielle au prix de 15~30€
[@Julien] Proposition d'un partenariat avec un Lycée-BTS qui pourrait réaliser les cartes (prototype)
 - Echanges avec Nantes Métropole suite à leur proposition d'ouverture d'accès à un espace "bac à sable" Opendatasoft
- Éléments à disposition au début de la session de travail :
 - Données de l'**Eco-device** (sous-comptage Linky) - branché chez J. Moreau - des 3 jours précédents
 - Données de **Nantes Métropole**.
Utilisation des données Enedis de Nantes Métropole pour avoir à disposition 1 an d'historique (tout 2018).
Note : Enedis n'a pas mis toutes les informations : numéro de livraison, type de tarifv(jaune, HP/HC, ...). Pour pouvoir décoder le "TIC" du Linky il faut avoir accès aux **index**. Importance notamment pour les fournisseurs d'énergie.
@Julien : Enedis n'est pas capable de relever la production pour un particulier, ni pour la centrale de production de Beaulieu. @Guillaume : Enedis a la donnée, mais il s'agit d'une question de mise à disposition.
 - Composant GCE RT2 (version "améliorée" de l'Eco-Device annoncée comme compatible avec le mode STANDARD Linky) à brancher sur l'installation
- Objectifs :
 - Acculturation et montée en compétences sur le bus sémantique
 - RT2 :
 - Branchement
 - Paramétrage
 - Récupération de données en mode pull
 - Récupération de données en mode push

- API x bus sémantique
 - Connexion à l'API Eco-Device et remontée des données dans le bus (pull/push)
 - Connexion à l'API RT2 et remontée des données dans le bus (pull/push)
 - Connexion à l'API Enedis J.Moreau et remontée des données dans le bus (pull/push)
 - Connexion à l'API Enedis pour Nantes Métropole et remontée des données dans le bus (pull/push)
- Data visualisation : données Eco-Device, RT2, Nantes Métropole
 - Test de Plot.ly (<https://plot.ly/>)
 - Développements web dans le bus sur la base de technos "web components", modulaire + côté client.
- Raspberry
 - Récupération données RS485 (onduleur)
 - Récupération données TIC-USB
 - Hachage, cryptage, envoi données, interprétation trame
- Discussion autour des perspectives :
 - Shield (GCE)
 - Fabrication d'un module/prototype par les contributeurs.trices et sympathisant.e.s DAISEE
 - Le RT2 peut faire du cryptage (à vérifier) mais pas l'Ecodevice

Actions de la journée

Les modes opératoires des actions réalisées listées ci-dessous est disponible plus bas.

- **Acculturation et montée en compétence avec le bus sémantique**

- Déborah et Guillaume ont manipulé le bus et pris en main différents workflows

- **Ecodevice RT2**

- Branchement réussi
 - Paramétrage : en cours. Pas de récupération de l'impulsion et de la trame du TIC. Récupération des données de soutirage effective.
 - Interrogation du RT2 par le bus par une requête HTTP Get (**mode pull**). A priori ok côté bus sémantique. Récupération de la donnée de soutirage. Limite : pas de soutirage durant l'expérimentation (journée : consommation de la production PV présumée)

![] (/uploads/upload_468ae44352815dd49f9aa5b7bdd0616b.PNG)

Duree (ms)	Nombre de record
1623	1314
▲ [1314] ▲ 0 {} conso : 006602.240,0.00 ▲ 1 {} conso : 006602.240,0.00	

- Appel d'une API HTTP Post sur le bus sémantique par le RT2 (mode **push**) : bus sémantique pas prêt (composé HTTP Post développé durant la première résidence de Prats-De-Mollo (https://pad.lamyne.org/s/r1_DSuw6Z) en novembre 2017 à revoir)

- **Branchement en sortie de l'API OpenDataSoft du portail Open Data de Nantes Métropole** : ok



Duree (ms)
0
▲ [100] ▲ 0 {} horodate : 2018-09-09T21:50:00+00:00 conso : 76000 id : 30001421924698 ▲ 1 {} horodate : 2018-09-09T21:40:00+00:00 conso : 115000 id : 30001421924698 ▲ 2 {} horodate : 2018-09-09T21:30:00+00:00 conso : 73000 id : 30001421924698

- **DataVisualisation**

- plot.ly (<http://plot.ly>) : connexion des 2 flux de données (production RT2 / consommation Nantes Métropole)
- Comparaison avec entre les fonctionnalités de Node-RED (<https://nodered.org/>) et celles du bus sémantique

Modes opératoires

Portail Opendata Nantes Métropole x Bus sémantique

- Mise à disposition des données de consommation de 4 bâtiments de Nantes Métropole sur une version privée de leur portail OpenDataSoft (pour le hackathon Atlansun)

Côté Portail Opendata

- Génération de la clé d'identification de l'API

Votre compte

Paramètres Clés d'API Quota Applications Visualisations Notifications

Clés d'API


Vous pouvez générer des clés d'API pour permettre à des applications tierces d'accéder aux données en votre nom. Les clés d'API peuvent être révoquées à tout moment.

Vous n'avez aucune clé d'API.

+ Générer une nouvelle clé

Clé d'API

Label

Valeur de la clé 

Révoquer cette clé

+ Générer une nouvelle clé

- Récupération des informations de la clé d'API (lien en bas de page)

Cartes Graphiques API Documentation

deborah.thebault Déconnexion

Consommation électrique de 4 bâtiments de Nantes Métropole

Informations Tableau Carte Analyse Export API

Ce jeu de données peut être utilisé via une API qui autorise la recherche et le téléchargement d'enregistrements par plusieurs critères, exposés ci dessous. Jetez un oeil à la [documentation de l'API](#) et utilisez la [console d'API complète](#) pour essayer les autres API !

dataset: consommation-electrique-de-4-batiments-de-nantes-metropole

ID du jeu de données

q:

Requête en texte intégral

lang:

Code langue de 2 lettres

rows: 10

Nombre de lignes de résultat (10 par défaut)

```
{
  "nhits": 121364,
  "parameters": {
    "dataset": [
      "consommation-electrique-de-4-batiments-de-nantes-metropole"
    ],
    "timezone": "UTC",
    "rows": 10,
    "sort": [
      "horodate"
    ],
    "format": "json",
    "facet": [
      "horodate",
      "annee",
      "mois",
      "jour",
      "puissance_souscrite_max_w"
    ]
  }
}
```

/api/records/1.0/search/?dataset=consommation-electrique-de-4-batiments-de-nantes-metropole&sort=horodate&facet=horodate&facet=annee&facet=mois&facet=jour&facet=puissance_souscrite_max_w&facet=code_postal&facet=identifiant_du_site&facet=

- Analyse de la documentation pour utilisation de la clé d'identification de l'API

Accès et Authentification

L'accès aux API d'un Domaine peut-être accessible à tous de façon anonyme, ou protégé et restreint aux utilisateurs authentifiés du Domaine. Dans le cas d'un Domaine protégé, l'authentification peut se faire de deux façons:

- [HTTP Basic Authentication](#) en utilisant le login et le mot de passe de l'utilisateur.
- Une clé d'API, à l'aide du paramètre `apikey`. Pour générer une clé d'API, il suffit d'ouvrir la page de préférences une fois connecté à la console.

`http://<DOMAIN>/api/datasets/1.0/search/?apikey=<APIKEY>`

L'accès aux API peut se faire en HTTP ou HTTPS. Cependant, dans le cas d'un appel authentifié par Basic Auth, il est recommandé d'employer HTTPS, à la fois pour protéger les informations d'authentification, mais également les données renvoyées. Dans le cas d'un appel JSONP depuis un navigateur, faites également attention à d'éventuelles restrictions (par exemple JSONP HTTP depuis une page HTTPS).

- Lecture des informations sur le paramètre "apikey". D'autres paramètres ont leurs caractéristiques renseignées dans l'onglet "API" du jeu de données des 4 bâtiments.

Exemple avec les paramètres facet annee, mois, jour, etc.

facet	horodate	×
	annee	×
	mois	×
	jour	×
	puissance_souscrite_max_w	×
	code_postal	×
	identifiant_du_site	×
	type_de_compteur	×
		+

Côté Bus sémantique

- **Composant 1** : Rest Get JSON - Connexion à l'API
 - Ajout d'un composant "Rest Get JSON" pour interroger l'API du Portail Open Data



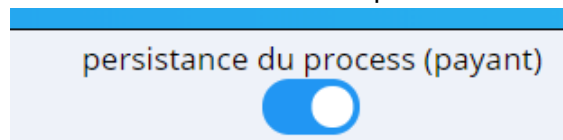
- Edition du composant



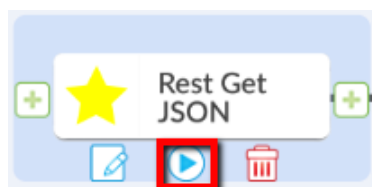
- Utilisation de la clé d'API et ajout du paramètre `apikey` pour l'identification suite à analyse de la documentation

https://sandbox-nantesmetropole.opendatasoft.com/api/records/1.0/search/?dataset=consommation-electrique-de-4-batiments-de-nantes-metropole&sort=horodate&facet=horodate&facet=puissance_souscrite_max_w&facet=code_postal&facet=identifiant_du_site (https://sandbox-nantesmetropole.opendatasoft.com/api/records/1.0/search/?dataset=consommation-electrique-de-4-batiments-de-nantes-metropole&sort=horodate&facet=horodate&facet=puissance_souscrite_max_w&facet=code_postal&facet=identifiant_du_site) &apikey=<votre_cle_API>

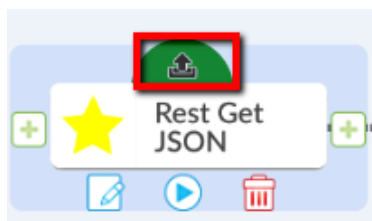
- Sélection de l'option persistance pour conserver les logs lors des exécutions. Un log donne des informations sur les résultats d'une opération.



- Exécution du composant



- Ouverture du journal de log



Nom du composant	Date debut
Ca marche	16/09/2018
Duree (ms)	Nombre de record
341	1
<pre> { facet_groups [8] records [100] 0 { record_timestamp : 2018-09-11T11:47:44+00:00 fields { annee : 2018 jour : 9 puissance_souscrite_max_w : 279000 identifiant_du_site : 30001421924698 type_de_compteur : ICE adresse : RUE LEFEVRE UTILE horodate : 2018-09-09T21:50:00+00:00 mois : 9 code_postal : 44000 puissance_w : 76000 recordid : e4ab986e6ca37a868e3bdf3d4091e65caac251e1 datasetid : consommation-electrique-de-4-batiments-de-nantes-metropole } } } </pre>	


Les éléments nous intéressant dans notre cas de figure sont renseignés dans le tableau "records"

- **Composant 2** : Value from path - Récupération de valeurs à partir d'une nouvelle racine. Le but est de conserver le "sous-groupe" des données qui nous intéressent pour la suite des traitements.

- Ajout du composant

Value From Path

extraire une valeur par son chemin

 Default

- Edition du composant et sélection du "sous-groupe" que l'on souhaite conserver. Ici, le sous-groupe "records" contenant les informations nous intéressant : année, horodatage, puissance souscrite, etc.

nom du composant

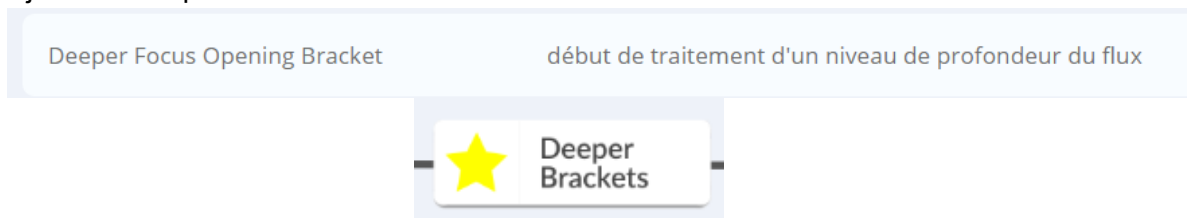
persistance du process (payant) ☒

records

chemin vers la valeur à récupérer et mettre à a racine

- **Composant 3** : Deeper brackets - Séquencage des données pour découper les traitements suivants en paquet. Le but est de lisser et d'éviter les surcharges. Ce composant est notamment utile pour traiter les gros volumes de données.

- Ajout du composant



- Edition du composant

nom du composant	persistance du process (payant) <input checked="" type="checkbox"/>
chemin à inspecter pour les traitements qui suivent (vide=racine)	
nombre de traitements parallèles	
le chemin designe une structure de tableau à conserver en tableau (décomposé en objet par défaut) <input type="checkbox"/>	

Chemin à inspecter : sélection du groupe de données qui va servir de base de référence pour le traitement

Nombre de séquences envoyées en parallèle au composant suivant. Par défaut : 1
 Taille des paquets envoyés. Par défaut les données sont envoyées par paquet de 1, donc 1 par 1 et sont traitées chacun leur tour par le composant suivant. Augmenter la taille des paquets permet de traiter plus de données en même temps.

Possibilité de choisir l'envoi sous forme de tableau au composant suivant sans le décomposition.

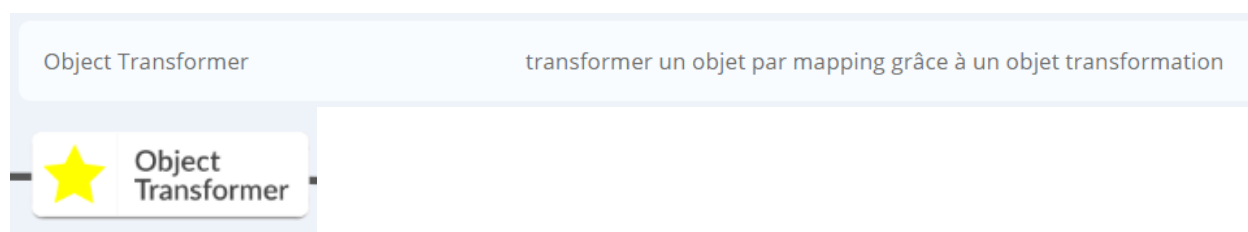
Envoi du groupe de données sélectionné à envoyer sous forme de tableau de données ou décomposition de chaque résultat avant envoi.

Structure à conserver en tableau. Par défaut : propriété par propriété. Coche : prise en compte de tout le tableau.

- **Composant 4** : Object Transformer - Transformation d'un objet par mapping.

Passage d'un objet source à un objet voulu avec un mapping. Il s'agit d'une sorte de recette de cuisine permettant de faire passer un objet d'une structure originale (A) à une structure cible (B). *Ex : pour obtenir le champs voulu "conso" on récupère la valeur dans le champs source "fields.puissance_w)*

- Ajout du composant



- Edition du composant

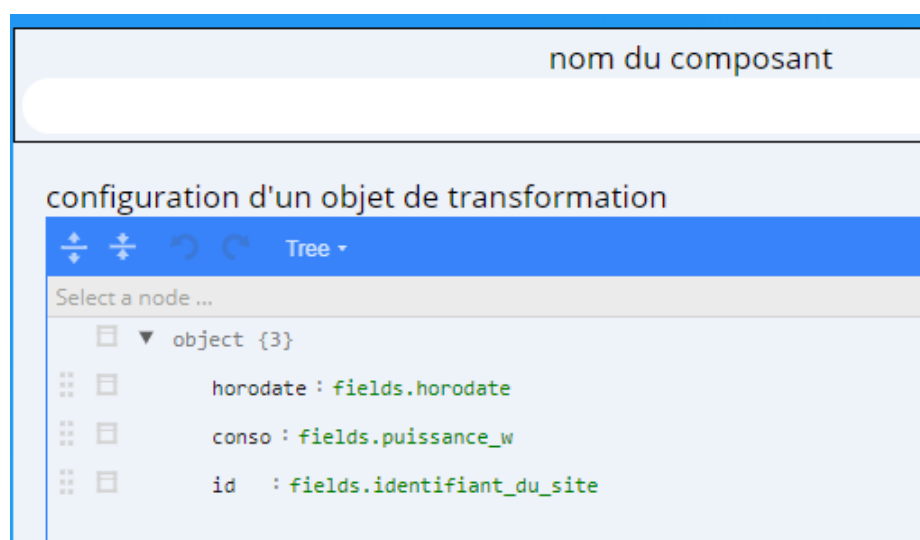
Définition de l'objet cible dans "fields" (champs gauche) et de l'objet source dans "value" (champs droit)

Rappel de la structure initiale :

Nom du composant	Date debut
Ca marche	16/09/2018
Duree (ms)	Nombre de record
341	1
<pre> { facet_groups [8] records [100] 0 { record_timestamp : 2018-09-11T11:47:44+00:00 fields { annee : 2018 jour : 9 puissance_souscite_max_w : 279000 identifiant_du_site : 30001421924698 type_de_compteur : ICE adresse : RUE LEFEVRE UTILE horodate : 2018-09-09T21:50:00+00:00 mois : 9 code_postal : 44000 puissance_w : 76000 recordid : e4ab986e6ca37a868e3bdf3d4091e65caac251e1 datasetid : consommation-electrique-de-4-batiments-de-nantes-metropole } } } </pre>	

Suite à l'utilisation du composant 2 "Value from Path", le "sous-groupe" records avait été sélectionné comme racine, comme "dossier principal". Nous avons donc abouti à une arborescence type (records).fields.annee, etc.

Dans notre cas, pour le mapping de la puissance, il faut donc sélectionner fields.puissance_w



- Récupération des données du log

```

[100]
  0 {}
    id : 30001421924698
    conso : 76000
    horodate : 2018-09-09T21:50:00+00:00
  1 {}
    id : 30001421924698
    conso : 115000
    horodate : 2018-09-09T21:40:00+00:00

```



Cache
NoSQL

nom du composant	
mettre en cache les data et les réinterroger	
<input type="checkbox"/>	Historisation
<input type="checkbox"/>	Sortie avec historique



Rest API
Get

nom du composant	
description de l'api	
hackathon-nantes-metro	key
http://semantic-bus.org/data/api/hackathon-nantes-metro	url
application/vnd.ms-excel	content-type

Exposition d'une API pour récupérer les données au format Excel à l'adresse indiquée

- Flux intégral bus (<http://semantic-bus.org/ihm/application.html#workspace/5b9cdeb1fab4e000908cac96/component>)
- Flux global



Visualisation sur plot.ly (<http://plot.ly>)

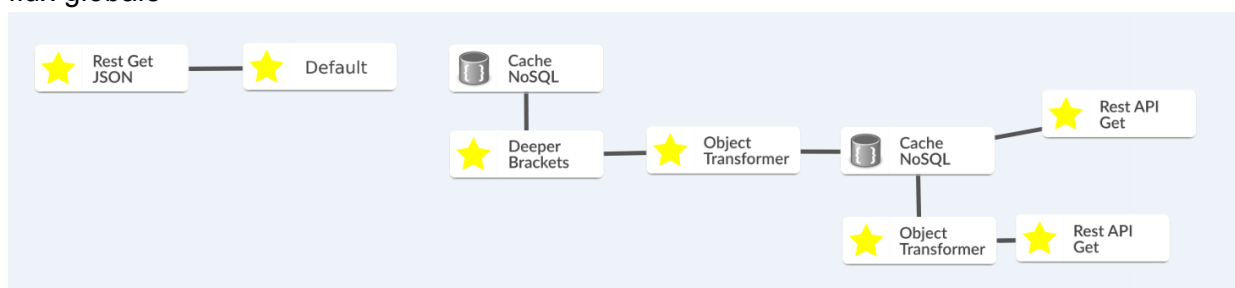
```

* edition : https://plot.ly/create/?fid=zenate:6
* view : https://plot.ly/~zenate/6/
![] (/uploads/upload_dedabaaefc0638fd89209383a462881.png)
* étapes
  * se créer un compte
  * créer un jeux de données
    *![] (/uploads/upload_30e027c952fcc02ef0bf06bc41faad86.png)

```

Compteur de julien par l'ancien eco-Device (petit materiel diférent de RT2)

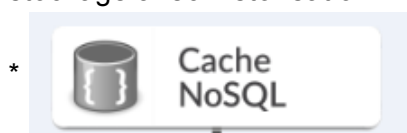
- <http://semantic-bus.org/ihm/application.html#workspace/5b97c526d0add804ef18981d/component> (<http://semantic-bus.org/ihm/application.html#workspace/5b97c526d0add804ef18981d/component>)
- flux globale



- Get sur API EcoDevice

- Timer pour jouer le traitement du flux toute les minutes

- stockage avec historisation



mettre en cache les data et les réintégrer

Historisation

Sortie avec historique

▸ [4522]

▸ [4522]

▸ 0 {}

▸ _id : 5b97c581d0add804ef18982e

▸ __v : 0

▸ data {}

▸ response {}

▸ salr [1]

▸ version [1]

▸ dnsstatus [1]

▸ c1_fuel [1]

▸ c0_fuel [1]

▸ c1day [1]

▸ c0day [1]

▸ count1 [1]

▸ count0 [1]

▸ meter3 [1]

▸ meter2 [1]

▸ meter1 [1]

▸ meter0 [1]

▸ T2_IMAX3 [1]

T2_IMAX3 [1]

- introspection de chaque objet et transformation de chaque objet



configuration d'un objet de transformation

Tree ▼

Select a node...

object {7}

countProd : data.response.count0.0

meterProd : data.response.meter2.0

date : data.response.date.0

datecalc1 : =parseInt({\$.data.response.date.0}.substring(1,3))

datecalc3 : =new Date(parseInt({\$.data.response.date.0}.substring(1,3)),parseInt({\$.data.response.date.0}.substring(3,5)),2018)

datecalc2 : =parseInt({\$.data.response.date.0}.substring(3,5))

heure : data.response.time0.0

- stockage sans historisation




mettre en cache les data et les réintégrer

Historisation
Sortie avec historique

```

[4522]
  0 {}
    heure : 15:39
    datecalc2 : 9
    datecalc3 {}
    datecalc1 : 11
    date : 11109
    meterProd : 1559
    countProd : 162433
  1 {}
  2 {}
  
```

- préparation des donnée pour une API XML



configuration d'un objet de transformation

Select a node ...

- object {1}
- root {1}
 - data : \$..

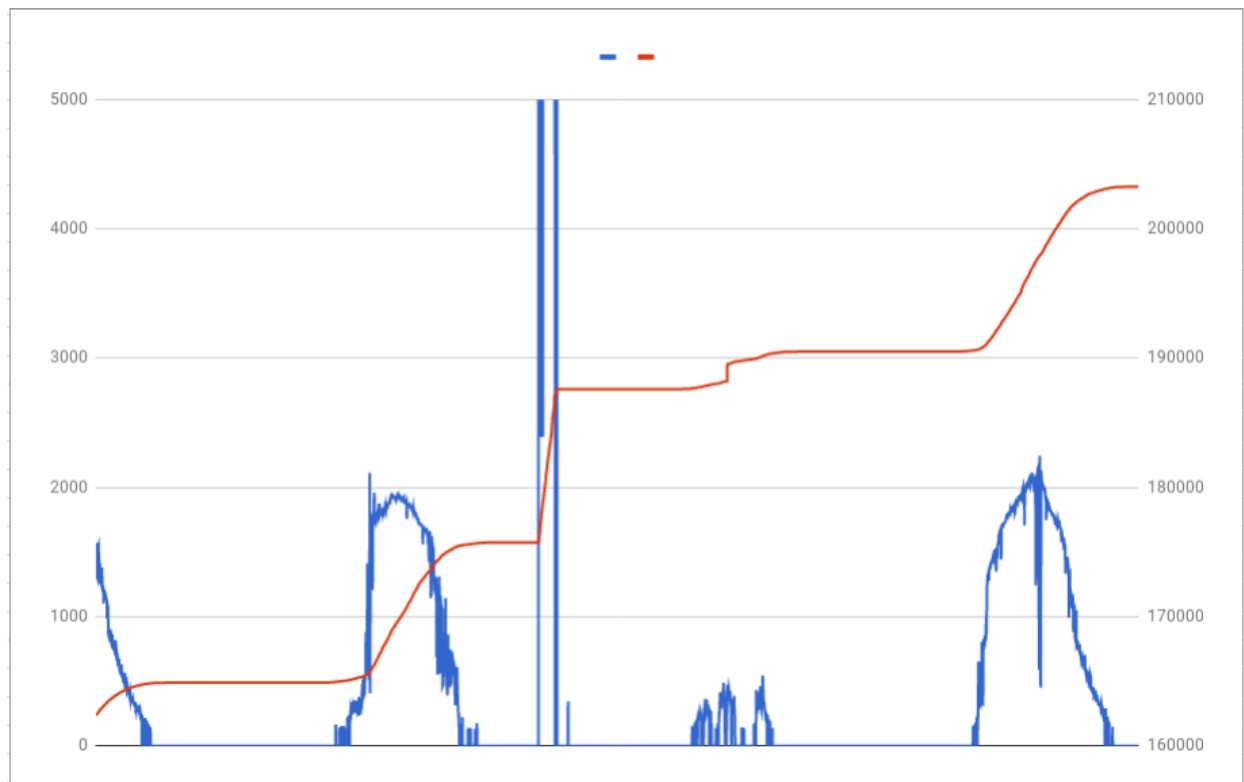
- exposition en API XML



description de l'api

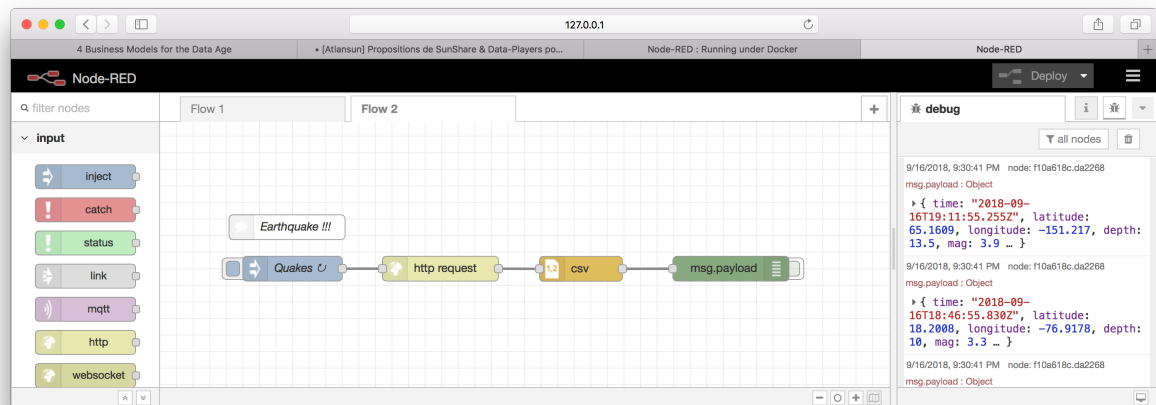
key	value
sunshare_Julien_pull	url
http://semantic-bus.org/data/api/sunshare_Julien_pull	content-type
application/xml	

- Plot.ly (<http://Plot.ly>) refuse de charger plus de 500ko de data et les datas traitée font plus ette limite
 - * visualisation sous google drive : <https://docs.google.com/spreadsheets/d/1BarZf02wD9-KLjqIL1AUUV0fkEzgy9i86CTzuJu6yM/edit#gid=1441179688> (<https://docs.google.com/spreadsheets/d/1BarZf02wD9-KLjqIL1AUUV0fkEzgy9i86CTzuJu6yM/edit#gid=1441179688>)



Réflexion : Bus sémantique / Node-RED (<http://nodered.org>)

- Comparaison : le bus sémantique est présenté ci-dessus ;
capture d'écran de Node-RED :



avis :

- Node-RED & le bus sémantique fonctionnent sur le même principe de flux graphique basé sur un système à 2 composants : les **blocs** et les **liaisons**.
- en terme d'**UX** (User eXperience), Node-RED est plus confortable / simple à utiliser ; l'interface est plus intuitive et l'utilisateur peut se l'approprier (
- plus) facilement.
- les **blocs** possèdent plusieurs entrées / sorties sur Node-RED.
- Node-RED propose également une **console de logging**, c'est-à-dire un espace (à droite sur la capture d'écran) pour afficher les résultats. Cela permet de savoir ce qu'il se passe.

- Concurrence bus sémantique / Node-RED ?
 - Node-RED est un outil plus complet (+ de fonctionnalités) et plus abouti (UX).
 - Le bus sémantique est peut-être plus adapté pour les traitements de données massives ("big data"). Le bus sémantique est peut-être plus inter-opérable au niveau du SI - ce qui permet de faire évoluer l'outil plus facilement. Par exemple, je ne sais pas si Node-RED pourra être utilisé pour le web sémantique.
 - Il faut regarder si Node-RED utilise une bibliothèque Open Source pour gérer son interface graphique et voir si celle-ci peut être ré-utilisée.
- Faites-vous votre avis ?

- Code source du schéma à copier-coller sur Node-RED :

```
[{"id":"c0ea781c.0f87c8","type":"tab","label":"Flow
1","disabled":false,"info":""},{ "id":"a52ee96c.61d08","type":"http
in","z":"c0ea781c.0f87c8","name":"","url":"/conso","method":"get","uplo
ad":false,"swaggerDoc":"","x":200,"y":320,"wires":
[["7f78096c.8ef26"]]},
{"id":"390b835e.5bbdfc","type":"template","z":"c0ea781c.0f87c8","name":
"","field":"payload","fieldType":"msg","format":"handlebars","syntax":"
mustache","template":"info0 : {{payload.info0}} <br>\ninfo3 :
{{payload.info3}}","output":"str","x":580,"y":320,"wires":
[["3313c6b1.7887c2"]]}, {"id":"3313c6b1.7887c2","type":"http
response","z":"c0ea781c.0f87c8","name":"","statusCode":"","headers":
{},"x":750,"y":320,"wires":[{}], {"id":"7f78096c.8ef26","type":"http
request","z":"c0ea781c.0f87c8","name":"","method":"GET","ret":"obj","ur
l":"http://perso.sunshare.fr:8001
/user/io.json","tls":"","x":390,"y":320,"wires":[["390b835e.5bbdfc"]]]}
```

6- Feedback

[@Jonathan]

- Bidouille avec du matos qu'on ne maîtrise pas forcément + pas forcément Open Source.
=> Faire à la mano et monter en compétence ?

[@Julien]

- Il exisdt une carte qui fait de l'impulsion (et TIC?) qui se branche sur un Rasperry PI ici (<http://www.yoctopuce.com/FR/products/capteurs-electriques-usb/yocto-knob>). Carte avec 4 entrées.
- Faire un shield avec 4 entrées

[@Jonathan]

- Branchements simples. Il a fallu 1/2 journée de formation (matos spécifique) mais finalement on a réussi à avoir 3 points d'entrée.

- Limite : on ne peut pas savoir si ça fonctionne : on ne sait pas ce qu'on mesure. RT2 -> outil pas adapté au besoin ?
- Le paramétrage pourrait se faire par le technicien qui installe le PV. Mais loin du système plug&play.

[@Julien]

- Mode d'emploi difficile pour décoder les trames.
- => Autant travailler pour la communauté que déchiffrer un système fermé.
- EcoDevice à condition qu'il lise la trame Linky standard. EcoDevice : 1200 baud, Linky : 9600 baud. Fonctionnel ?
- Toute la partie soft peut être développée sur le Raspi en attendant le shield.

[@Guillaume]

- Intéressant de travailler sur le bus sémantique, surtout en rebond des discussions de la veille
- Dû au fait que nous étions dans la pratique
- Echanges sur la partie design du bus sémantique et comparaison avec Node-RED intéressants.

[@Déborah]

- Intéressant malgré les frustrations. Mise en valeur des limites du RT2 par rapport à l'EcoDevice. Ça permet d'avancer dans la réflexion et d'ouvrir de nouvelles pistes d'expérimentation
- Pas nombreux mais bilan satisfaisant :
 - Prise en main Node-RED
 - Veille DataViz
 - Branchements électroniques
 - ...
- => Et sans avoir eu à formaliser l'organisation des tâches, qui fait quoi et la contribution et documentation ...
- Intéressant pour la suite avec des opportunités de continuation du projet avec d'autres personnes / dans d'autres types de formats
- Ça pourrait faire l'objet d'un weekend en résidence à l'image de la résidence de Prats (e.g. réalisation d'OpenEnergyMonitor (qui utilise Node-RED)).

[@Julien]

- Laisse les branchements à disposition et Raspberry accessibles à distance. Si DAISEE est intéressé
- [@Déborah]
 - Outils collaboratifs où tu es à l'aise ?
 - Par exemple, se tenir en mettant en copie dans les mails / avec un channel Rocket.Chat
 - ...

- Utilisons l'instance **Rocket.Chat** de la MYNE ?
- [@Julien] Essayons

[@Déborah]

- Attention au manque d'informations qui a pu créer de la frustration
- Il y a eu des frictions culturelles

[@Simon]

- Très content de l'ambiance de la journée. Avoir pris le temps de s'acculturer.
- Limite de l'outil "Plot.ly (<http://Plot.ly>)" qui n'autorise que 500Ko de data (< 5000 lignes).
- => Il faudra passer sur Web Components qui utilise la bibliothèque plot.ly (<http://plot.ly>) qui est open source. Ce développement me paraît prioritaire.
- Autre développement prioritaire, l'API Rest en Post qui avait été développée à prats mais qui ne fonctionne plus.
- Discussion avec Node-RED à approfondir et notamment explorer la bibliothèque de graphe et voir si elle peut être ré-utilisée.

Plot.ly (<http://Plot.ly>) -> migration de GNU plot en JS.

- Développer un device custom, avec la communauté DAISSEE et le faire avec des composants plus bas-niveau et faire de l'acculturation (par ex: problématiques physiques électroniques) pour aboutir à un cahier des charges juste et cohérent de manière à ce que GCE soit en mesure de développer un shield qui correspond. Essayer d'être bas-niveau (dans l'idéal pas de processeur) pour arriver
 1. USB
 2. à la main
 3. shield

[@Julien]

- Identifier à Nantes des personnes susceptibles de contribuer au projet. Par ex: le lycée Nicolas Apper qui pourrait imprimer la carte.

[@Déborah]

- Dans l'idée d'une résidence, celle-ci serait dans tous les cas ouverte.

[@Simon]

- Node-RED pourrait être la brique que l'on mettrait dans le Raspi pour traiter les signaux des pins & ports série.
Cela nécessite de poser des questions sur le stockage, le transfert, etc.
- Cela revient à faire OEM mais avec un shield ? Enjeux de coût : OEM 200€ / Raspbi-Shield 100€.
- Enjeux de **cyber-sécurité** à prendre en compte. Sans passer par la blockchain.

Davy Marchand Sun'Air : aggrégation de barrages hydroélectriques

La licence juridique de ce document est CC by SA

-> on ne précise pas quelle version