

03/05/2022

Environnemental Sensing

Présentation

Environmental Sensing



Confidential C

TABLE DES MATIERES

1	Introduction	3
1.1	Données environnementales	3
1.2	Contexte	3
1.3	Attentes	4
1.4	Objectifs	4
1.5	Contenu	5
2	Interopérabilité	6
2.1	Principes	6
2.1.1	Niveau Technique	6
2.1.2	Niveau syntaxique	6
2.1.3	Niveau sémantique	7
2.2	Situation actuelle	8
2.2.1	Niveau sémantique	8
2.2.2	Niveau syntaxique	9
3	Cas d'usage et bénéfices	11
3.1	Mesure d'exposition – interopérabilité horizontale	11
3.2	Cycle de vie des données – interopérabilité verticale	12
3.3	Assimilation de données – interopérabilité horizontale	13
3.4	Bénéfices attendus	13
4	Présentation	16
4.1	Principes généraux	16
4.2	Structure de données	18
4.3	Opérations	19
4.4	Connecteurs	19
4.5	Utilisation	19
4.6	Exemples	20
4.6.1	Mesure simple	21
4.6.2	Mesure sur un trajet	21
4.6.3	Mesure multiple	22
5	Structuration	23

5.1	Modèle de données	23
5.1.1	Observation.....	23
5.1.2	Ilist.....	24
5.1.3	ESValue.....	24
5.2	Traitements	25
6	Mise en œuvre	27
6.1	Principes	27
6.2	Acquisition par capteur	27
6.3	Echange de données	28
6.4	Stockage et partage de données.....	28
6.5	Traitement des données	28
6.6	Représentation des données	28
6.7	Exemples	29
6.7.1	Capteur.....	29
6.7.1	Stockage en Base de données.....	29
6.7.2	Extraction d'une Base de données	29
6.7.3	Visualisation d'une Observation	30
7	FAQ.....	32
8	Annexes.....	34
I.	Annexe 1 : Modèle de classes	35
II.	Annexe 2 : Mapping Bluetooth	36
III.	Annexe 3 : Représentation binaire.....	37
IV.	Annexe 4 : Représentation textuelle.....	39
V.	Annexe 5 : API	40
VI.	Annexe 6 : Mapping formats XD	41

1 INTRODUCTION

1.1 DONNEES ENVIRONNEMENTALES

Le Conseil National du Numérique dans son avis de juillet 2020 intitulé « **Faire des données environnementales des données d'intérêt général** » définit les données environnementales de la façon suivante :

« Les données environnementales peuvent être définies de manière extensive comme toute donnée, par nature ou par destination, relative à l'environnement, à son état et/ou à ses flux d'interaction. Cette définition présente l'avantage de pouvoir qualifier plusieurs types de données – à l'instar des données agricoles, des données naturalistes ou des données relatives à la mobilité – comme des données environnementales. »

« L'interopérabilité et la qualité des données, ainsi que leurs structure authenticité et intégrité, sont essentielles pour le partage des données d'intérêt général. Dès lors, le régime d'ouverture des données d'intérêt général implique la prise en compte de la lisibilité et l'interopérabilité des formats et des données. »

Cette définition donne un cadre étendu à la notion de données environnementales.

1.2 CONTEXTE

On constate une grande diversité de solutions traitant de données environnementales et paradoxalement, l'interopérabilité entre ces solutions reste très faible, ce qui oblige à mettre en œuvre des interfaces spécifiques et conduit à dénaturer les données d'origine par des retraitements successifs.

Cette absence d'interopérabilité a pour conséquence une absence de solution transversale aussi bien au niveau des cas d'usage (intégration des contextes personnels, familiaux, professionnels, régionaux) qu'au niveau du cycle de vie des données environnementales : (1) de la production des données (acquisition, observation, modélisation), (2) de leur partage (réseaux, stockage), (3) de l'exploitation des informations (restitution, analyse).

La mise en place d'une interopérabilité forte, est conditionnée par :

- L'existence d'une structure de données unique et partagée (standard)
- La disponibilité d'outils ouverts d'exploitation de cette structure (connecteurs),

Trois conditions supplémentaires sont nécessaires pour que cette structuration soit adoptée et appliquée :

- Une mise en œuvre simple et adaptée à chaque besoin,
- Une construction de « convergence » plutôt que de « remise en cause » de l'existant,

- Une compatibilité avec la qualification « d'intérêt général » des données environnementales

1.3 ATTENTES

Les données environnementales (y compris mobilité et énergie) sont identifiées comme données « d'intérêt général » et sont donc soumises à une exigence d'interopérabilité croissante.

Le développement des métropoles et plus généralement de systèmes intelligents, plus adaptatifs et efficaces nécessite une gestion technique renforcée s'appuyant sur des données ouvertes de plus en plus nombreuses.

L'accès à une information environnementale localisée, fiable et compréhensible est une demande citoyenne relayée au travers de collectifs toujours plus importants.

1.4 OBJECTIFS

Ce document propose une solution d'interopérabilité des données environnementales applicable aussi bien au niveau de données unitaires (ex. émises par un capteur) que de données agrégées à différentes échelles de temps ou d'espace.

Elle permet de :

- Faciliter l'usage et le partage des données environnementales
- Optimiser le volume des données échangées
- Banaliser aussi bien les équipements d'acquisition de données (capteurs) que les applications de traitement,
- Mettre en œuvre une architecture logicielle remplaçant toutes les opérations de codage/décodage (interfaces) par l'utilisation de connecteurs standards,
- Respecter et s'appuyer sur les principaux standards existants
- Partager et développer collectivement un ensemble de connecteurs open-source répondant à toutes les situations (plateforme)

La structure de données proposée s'appuie sur les standards existants qu'elle complète en assurant une convergence.

Cette structure est implémentée au travers de connecteurs utilisables sur toute la chaîne de traitement (du capteur produisant une donnée au système d'analyse et de restitution d'un ensemble de données) incluant les interfaces avec les équipements réseau, les systèmes de calcul et les bases de données.

Les modèles de données et les connecteurs sont partagés et accessibles en open-source (licence Creative Commons BY-SA).

1.5 CONTENU

Ce document précise le contexte lié aux notions d'interopérabilité appliquées aux données environnementales (chapitre 2).

Il décrit les bénéfices attendus d'une solution sur la base d'exemples de cas d'usage (chapitre 3).

Il présente ensuite les principes mis en œuvre et nécessaires à l'atteinte des objectifs ci-dessus (chapitre 4 et 5) et en particulier :

- la structure de données permettant de prendre en compte les différentes données environnementales définies,
- les opérations de traitement sur ces structures (création, formatage, assemblage, extraction),
- les formats d'échanges (fichiers, textuels, binaires),
- les connecteurs pour les interfaces (réseaux, stockage, visualisation, traitement)

Il présente enfin quelques exemples et principes de mise en œuvre.

2 INTEROPERABILITE DES DONNEES ENVIRONNEMENTALES

« L'interopérabilité est la capacité que possède un produit ou un système, dont les interfaces sont intégralement connues, à fonctionner avec d'autres produits ou systèmes existants ou futurs et ce sans restriction d'accès ou de mise en œuvre. » (définition Wikipedia)

Elle se décline à trois niveaux :

- Technique « pouvoir communiquer »
- Syntaxique « savoir communiquer »
- Sémantique « savoir se comprendre »

Deux autres niveaux externes complètent cette décomposition :

- Niveau organisationnel
- Niveau légal

Dans ce chapitre, seuls les trois premiers niveaux sont abordés.

2.1 PRINCIPES

2.1.1 Niveau Technique

Il concerne principalement la définition des interfaces, les formats de données et les protocoles utilisés.

L'objectif de ce projet n'est pas de créer de nouveaux produits ou équipements relatifs aux données environnementales mais de s'intégrer aux infrastructures existantes.

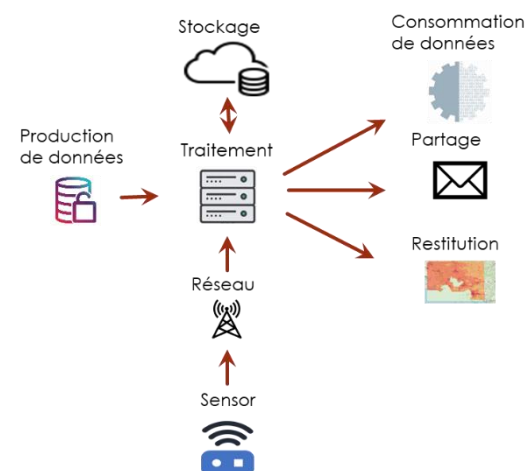
Il s'agit donc d'un existant à prendre en compte (contrainte).

2.1.2 Niveau syntaxique

Il concerne le mode de formatage et de codage des données en fonction des interfaces techniques définies.

Il est spécifique du niveau où l'on se situe dans la chaîne de traitement des données environnementales (cf schéma ci-contre) :

- Equipements de production de données (capteurs) :
 - Intégration avec les systèmes de mesure embarqués
 - Interface avec les réseaux de communication
- Réseaux :



- Plusieurs types de réseaux : personnel (PAN), local (LAN), étendu (LPWAN)
- Interface avec les protocoles de communication (ex. LoRa, Bluetooth, TCP/IP)
- Traitement :
 - Prise en compte de différents langages informatiques (ex. C++, Python, JavaScript)
 - Utilisation des outils disponibles dans chaque langage
- Production et consommation de données :
 - Intégration (API) avec les applications tierces (ex. outil de calcul) ou de mise à disposition de données (ex. open-data)
 - Capacité à échanger entre structures de données différentes
- Stockage :
 - Prise en compte des formats de stockage de type fichier, documents (base de données NoSQL), tables (base de données SQL)
 - Intégration des API avec les bases de données ou les systèmes d'exploitation
- Restitution de données :
 - Intégration avec les outils adaptés à la nature des données environnementales (spatiale, temporelle, physique)
- Partage de données :
 - Intégration avec les outils de partage et d'échange de données

2.1.3 Niveau sémantique

Il concerne la nature et l'usage des données environnementales.

Les données environnementales peuvent être caractérisées selon différentes dimensions :

- Une dimension spatiale qui peut être :
 - restreinte (ex endroit identifié) ou étendue (ex ville, pays)
 - caractérisée (ex. par une limite) ou informelle (ex. zone humide, un embouteillage)
 - fixe (ex un lieu) ou mouvante (ex. un panache de fumée, une ombre)
 - en deux dimensions (ex sur une carte) ou en trois dimensions (ex. un étage, une profondeur)
- Une dimension temporelle qui peut être :
 - ponctuelle (un instant donné) ou étendue (ex journée, saison, décennie)
 - caractérisée (ex. horodatage) ou imprécise (ex. heures de pointe)
 - fixe ou mouvante (ex. heure du coucher du soleil)
- Une dimension représentant un phénomène, une propriété, une perception qui peut être décrite par :
 - Une ou plusieurs mesure ou observation d'une propriété
 - Une caractérisation (ex. ressenti)
- Une dimension contextuelle qui explicite :
 - Le contexte d'obtention de la donnée

- Le mode d'obtention de la donnée
- Sa validité et son niveau de confiance

En termes d'usages, on distingue les besoins suivants qui conditionnent également le niveau sémantique :

- Analyse : L'objectif est de pouvoir traiter un ensemble de données qui peut être important pour en extraire une information de plus haut niveau (ex. une mesure de pression atmosphérique peut être utilisée pour identifier un risque d'orage)
- Restitution : Elle consiste à fournir sous une forme adaptée à l'utilisateur un ensemble d'informations (ex. des courbes de niveau pour représenter un relief, des cartes animées pour représenter un trafic routier)
- Archivage : Il s'agit ici de stocker les informations pour une exploitation ultérieure non encore définie.

Ces usages, renvoient donc sur deux notions complémentaires :

- Une notion d'échelle, de granularité ou d'agrégation des données
- Une notion d'information dérivée

2.2 SITUATION ACTUELLE

2.2.1 Niveau sémantique

Le standard ISO 19156 – « Observation and Measurement » dont le champ d'application est rappelé ci-dessous permet de fixer la plupart des concepts et dimensions associées aux données environnementales.

"This International Standard defines a conceptual schema for observations, and for features involved in sampling when making observations. These provide models for the exchange of information describing observation acts and their results, both within and between different scientific and technical communities.

Observations commonly involve sampling of an ultimate feature-of-interest. This International Standard defines a common set of sampling feature types classified primarily by topological dimension, as well as samples for ex-situ observations. The schema includes relationships between sampling features (subsampling, derived samples).

This International Standard concerns only externally visible interfaces and places no restriction on the underlying implementations other than what is needed to satisfy the interface specifications in the actual".

Ainsi, les principaux concepts associés aux données environnementales sont définis et trouvent une déclinaison dans les formats d'échanges :

- La dimension spatiale est représentée par les coordonnées géographiques (géodésiques ou projections) et des entités géographiques (polygones et formes élémentaires) sont intégrées dans les formats d'échange ou de stockage (ex. GeoJSON),
- La dimension temporelle est également représentée par la notion de TimeStamp qui permet de représenter un instant en fonction de différents fuseaux horaires. Les intervalles, durées et périodicités sont également codifiés (ISO 8601 mise à jour en 2019 et sa déclinaison informatique RFC3339),
- Les propriétés étudiées font l'objet de nombreux catalogues et classifications mais on ne dispose pas d'un référentiel unique,
- Les mesures numériques font référence à un système d'unités bien établi (SI).

Cependant, la notion complète d'Observation ne fait l'objet d'aucun standard applicable transversalement. On trouve notamment une déclinaison de l'ISO 19156 au niveau de l'OGC (Open Geospatial Consortium) qui est adaptée aux grandes instances internationales mais peut difficilement être appliquée à des utilisations basiques. A l'opposé, Bluetooth SIG (Special Interest Group) définit des standards qui sont directement implémentables (ex. Environmental Sensing Service).

Il est à noter également qu'il n'existe aucun standard d'échange traitant à la fois des dimensions temporelles, spatiales et physiques.

2.2.2 Niveau syntaxique

Au niveau syntaxique des solutions existent et sont en place. Le sujet est donc de mettre en correspondance la structure de données définie au niveau sémantique avec les solutions existantes au niveau syntaxique au travers de connecteurs.

Plus précisément, si l'on reprend le découpage effectué au chapitre 2.1

- Equipements de production de données (capteurs) :
 - Les capteurs « programmable » disposent de leur propre langage de programmation qui leur permet d'interagir avec les systèmes de mesure embarqués. Le connecteur doit donc être créé au niveau de ce langage (ex. C++ pour les microcontrôleurs les plus simples de type ESP) et gérer l'interface avec le protocole de communication.
- Réseaux :
 - Trois types de réseaux sont envisagés :
 - Bluetooth : Le protocole et les données échangées sont définies dans l'ESS (Environmental Sensing Service) et dans le « Profile » associé. Celui-ci est supporté par des bibliothèques Bluetooth. Le connecteur à mettre en place doit donc appeler ces bibliothèques et établir la correspondance (mapping) entre la structure de données Bluetooth et celle du standard.

- LoRa, Sigfox : ces réseaux LPWAN transportent des ensembles de données binaires non structurés (« payload »). Le connecteur pour ces réseaux doit donc générer une image binaire des données à transporter (codage) et à l'inverse reconstruire la structure de données à partir d'un « payload » (décodage).
 - Réseaux TCP/IP : Pour ces réseaux, le standard le plus commun est l'utilisation de requêtes http dans une architecture REST qui permet de transporter également un « payload ». Par contre, contrairement aux réseaux LPWAN, le Payload peut être structuré. L'usage le plus courant (sans contrainte forte de volume de données) est d'utiliser le format JSON. Les requêtes http et le format JSON existent dans la majorité des langages de programmation. Le connecteur doit donc appeler les bibliothèques associées à ces requêtes et disposer d'un codage / décodage JSON.
- Traitement :
 - Les traitements s'effectuent directement à partir de la structure de données définie au niveau sémantique. Deux points sont donc à prendre en compte :
 - Pour permettre la mise en place de traitements avec différents langages informatiques (ex. C++, Python, JavaScript), la structure de données doit être suffisamment simple pour pouvoir être transcrite sur les principaux langages de programmation.
 - Pour faciliter les traitements la structure de données doit pouvoir être convertie en structure de plus haut niveau (ex. pour Python : matrices indexées multi-dimensionnelle Xarray)
- Production et consommation de données :
 - Ces fonctions correspondent à des interfaces d'export ou d'import de données. Au niveau le plus basique, il s'agit de disposer d'une part d'API couvrant les principaux besoins (recherche, sélection, envoi, réception...) et d'autre part de formats d'échange (ex. fichier s'appuyant sur les formats binaire ou textuel déjà existants).
 - Chaque interface est à construire sur la base des API disponibles et des formats de données souhaités (ex. une interface maquette numérique peut nécessiter l'utilisation d'un format IFC).
- Stockage :
 - Plusieurs modes de stockage sont à prendre en compte :
 - Stockage « boîte noire » par fichier : celui-ci découle des formats binaires et textuels définis
 - Stockage « boîte blanche » en base de données. Deux cas de figure principaux sont à prendre en compte :
 - Base de données relationnelle : La structure de données définie est compatible avec un découpage en tables (une table par dimension).

- Base de données NoSQL : Le stockage s'effectue sur la base du format JSON défini complété de données de synthèse facilitant la recherche et la sélection.
- Stockage mixte « métadonnées » en base de données et « données » en fichiers.
- Les connecteurs de stockage disposent donc :
 - Des API de tri, de recherche et de sélection adaptées au support de stockage,
 - Des fonctions de conversion au format supporté par le système de stockage,
- Restitution de données :
 - La restitution des données est un sous-ensemble des fonctions de traitement. Néanmoins, des fonctions de base doivent permettre de restituer
 - D'une part des indicateurs permettant d'évaluer la qualité ou la quantité des données,
 - D'autre part des représentations simples permettent de comprendre les données suivant leurs différentes dimensions spatiale, temporelle, et physique (ex. Représentation sous forme de courbes, de graphique, de cartographies...)
- Partage de données :
 - Ce point est à traiter dans un second temps.

3 CAS D'USAGE ET BENEFICES

Ce chapitre illustre à partir d'exemples les bénéfices d'une interopérabilité renforcée.

3.1 MESURE D'EXPOSITION – INTEROPERABILITE HORIZONTALE

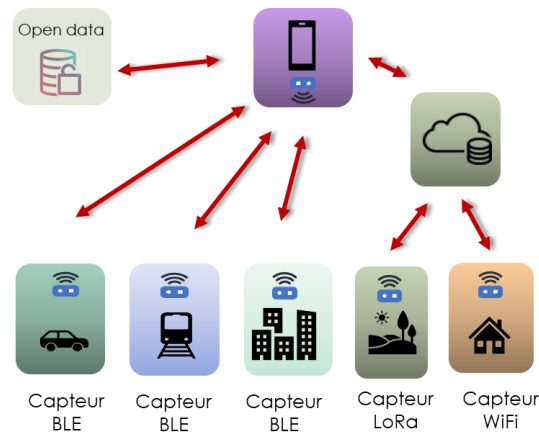
L'exposition d'une personne à un paramètre environnemental nécessite une mesure de cette exposition pour toutes ses activités quotidiennes (au domicile, dans les transports, sur son lieu de travail, dans les lieux publics, en extérieur).

Exemple de cas d'usage :

Un scénario de mesure de cette exposition pourrait par exemple être le suivant :

- *J'installe des capteurs à mon domicile (intérieur, extérieur),*
- *Mon véhicule est équipé par le constructeur d'un capteur,*
- *Les lieux publics mettent à disposition les informations (issues de capteurs ou de modélisation),*
- *Pour les lieux extérieurs non couverts par des capteurs, je prends en compte les données open-data disponibles*

Aujourd'hui, aucune solution ne couvre l'ensemble de ce scénario et l'absence d'interopérabilité entre capteurs et applications nous interdit un « mixage » de solutions.



La mise en œuvre de connecteurs standards permet de répondre à ce type de scénario :

- Les capteurs Bluetooth disposés dans les véhicules ou les lieux publics fournissent les données suivant le service Bluetooth ESS (Environnemental Sensing Service) mis à niveau en septembre 2021.
- Les capteurs individuels utilisant un réseau LPWAN (extérieur) ou TCP/IP (intérieur) envoient leurs données au format standard défini,
- Elles sont stockées sur une base de données respectant le format standard,
- Une application indépendante à la fois des capteurs et des dispositifs de stockage accède à l'ensemble des données émises via les connecteurs définis.
- Les traitements applicatifs effectués sont également indépendants de la nature des données traitées.

3.2 CYCLE DE VIE DES DONNEES – INTEROPERABILITE VERTICALE

L'exploitation des données environnementales nécessite de pouvoir gérer l'ensemble du cycle de vie (acquisition / transmission / stockage / traitement / analyse / restitution).

Exemple de cas d'usage :

Je mets en place un réseau citoyen de mesures environnementales, pour cela, je communique aux personnes de mon réseau :

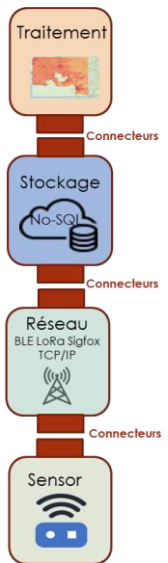
- *Quelques références de capteurs à acquérir en fonction des réseaux retenus,*
- *Les coordonnées de la base de données partagée que j'ai créée (configuration Cloud standard),*

- Quelques références d'applications de restitution des informations.

Sur ce périmètre, les solutions proposées actuellement sont « fermées », ou bien ne couvrent qu'une partie du cycle de vie.

Les connecteurs proposés permettent de faire communiquer les principaux composants :

- Les capteurs intègrent le connecteur permettant de générer la structure de données,
- Les connecteurs réseaux assurent le codage / décodage suivant le format d'échange propre au réseau,
- Les connecteurs base de données stockent les données selon le format de la base de données et les restituent suivant les API adaptés,
- Les connecteurs de traitement génèrent les structures de données adaptées aux traitements à réaliser et produisent les informations utilisateurs souhaitées



3.3 ASSIMILATION DE DONNEES – INTEROPERABILITE HORIZONTALE

Les données environnementales sont disponibles sous différents formats, à différentes échelles spatiales ou temporelles. L'intégration de ces données est souvent complexe et nécessite une expertise et des outils spécifiques.

Exemple de cas d'usage :

Je souhaite connaître le niveau de concentration d'un polluant donné au voisinage de mon domicile. Je dispose :

- D'une valeur moyenne annuelle sur ma région par maille de 100m x 100m (open-data)
- De relevés de mesures horaires de plusieurs stations fixes situées entre 1 km et 10 km (open data)
- De mesures ponctuelles relevées lors de mes déplacements alentours mais avec une précision très faible (capteur mobile)

Ces données sont de natures différentes. Il n'existe pas d'outils simples permettant d'assembler ces données et d'en extraire des informations de synthèse.

La structure de données proposée permet de représenter au sein d'un même objet ces trois types d'informations. Les techniques d'analyse disponibles peuvent alors être appliquées automatiquement à cette structure. Dans le cas présent, l'utilisation d'un procédé « d'assimilation de données » permet de fournir la réponse optimale à la question posée.

3.4 BENEFICES ATTENDUS

La mise en œuvre de la structure de données et des connecteurs associés permet d'obtenir une interopérabilité à différents niveaux :

- Interopérabilité matérielle entre les équipements d'acquisition, les équipements réseaux et les équipements de stockage,
- Interopérabilité entre données de nature différente,
- Interopérabilité des outils d'analyse et de traitement de ces données

Une première segmentation permet d'identifier les bénéfices suivants :

- Chaîne d'acquisition (capteurs, microcontrôleurs, réseau) :
 - Compte tenu de l'absence de standardisation des données, chaque contributeur de cette chaîne traite les données dans un format spécifique. Ceci oblige de mettre en œuvre des fonctions de codage/décodage des informations reçues.
 - Certains fournisseurs étendent leur périmètre à des plateformes d'accès à leurs données,
 - **L'intégration de connecteurs standard permet de banaliser les équipements et leur intégration dans un système complet.**
- Plateformes IoT :
 - Les plateformes généralistes proposent de nombreux services couvrant l'ensemble des fonctions liées aux données mais n'offrent pas de fonctions directement utilisables pour les données environnementales (absence de standards).
 - Les plateformes spécialisées qui s'appuient sur ces plateformes généralistes, des outils open-source ou encore sur des outils de plus bas niveaux, proposent une structuration propriétaire et développent de nombreuses interfaces.
 - Les plateformes internationales dédiées aux enjeux climatiques utilisent des outils spécialisés liés aux standards géomatiques (OGC). Ces outils sont complexes et ne peuvent être déclinés pour des besoins spécifiques.
 - **L'intégration de connecteurs standard intégrés aux plateformes du marché permet de réduire le nombre d'interfaces et de mettre en place des traitements sur des structures de données de plus haut niveau.**
- Editeurs de solutions de conception ou d'aménagement :
 - Les éditeurs (ex. Dassault Systèmes, Autodesk) s'appuient sur des structures de données internes,
 - Des standards existent pour faciliter l'interopérabilité entre éditeurs (ex. IFC : Industry Foundation Classes),
 - **L'intégration dans les IFC du standard défini permet de déployer les connecteurs standards dans les solutions de conception et d'aménagement.**
- Intégration applicative :
 - De nombreux acteurs sont présents sur ce marché en ciblant des cas d'usage identifiés. Ces acteurs utilisent les services apportés par les fournisseurs de capteurs ou de plateformes pour mettre en place des solutions spécifiques.
 - **L'intégration de connecteurs standard permet d'élargir le périmètre des solutions proposées tout en en réduisant le coût de mise en œuvre.**
- Exploitation de données (acteurs publics, privés ou citoyens) :

- Les gestionnaires / exploitants de données environnementales construisent, exploitent et mettent à disposition des ensembles de données toujours plus étendus.
- Ils intègrent des passerelles vers les solutions de leur structure (ex. SIG pour les métropoles, systèmes de monitoring et de pilotage) et vers les acteurs locaux et régionaux (ex. open data)
- **L'usage de solutions standardisées et open-source est un élément de politique technique qui permet de faciliter l'intégration, l'agrégation et le partage de nouvelles données.**
- Activités d'étude et d'aménagement :
 - Les acteurs de ces secteurs sont confrontés aux difficultés d'accès à des données fiables et précises et également à l'absence d'interopérabilité des données échangées.
 - **La mise en place de standard de données de plus haut niveau permet d'obtenir des données mieux structurées et de faciliter les échanges.**
- Citoyen et utilisateur final :
 - Les utilisateurs finaux sont confrontés à l'éparpillement des solutions proposées et sont demandeurs d'interopérabilité et d'accès à des données plus complètes et de plus haut niveau que celles accessibles aujourd'hui.
 - **La solution proposée répond à cette attente.**

4 PRESENTATION

4.1 PRINCIPES GENERAUX

La structure de données mise en place est issue du concept « d'*Observation* » tel qu'il est défini dans la norme ISO19156. Elle permet de représenter par exemple :

- Des données unitaires issues de capteurs,
- Des résultats de modélisation,
- Des répartitions géographiques,
- Des historiques temporels ou sur un trajet,
- ...

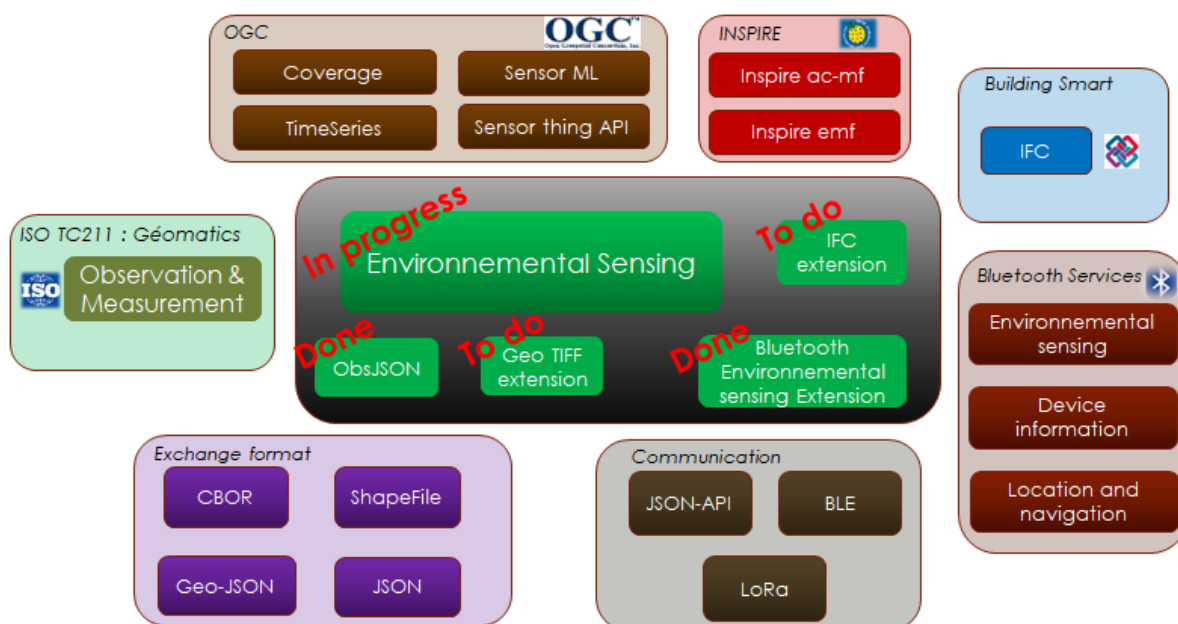


Une Observation est représentée sous la forme d'un objet informatique unique. Celui-ci peut alors être traité ou bien exporté sous différents formats neutres sans avoir à adapter les traitements à la nature des données qui composent cet objet.

Standard

La structuration retenue s'appuie notamment sur les standards et outils suivant :

- ISO 19156:2011 - Observations and Measurement (TC211 - géomatique)
- Référentiel OGC - SWE, (en particulier SensorThings API Part 1: Sensing, SensorML, Coverage, WaterML Part 1: TimeSeries) et ODM2 : Observation Data Model
- Format de données textuelles JSON, GeoJSON
- TIFF 6.0, NetCDF pour les formats de fichiers
- Format de données binaires : CBOR, Bluetooth (Environnemental Sensing Service)
- Structures Xarray, Pandas et GeoPandas de traitement des données multidimensionnelles



Le travail de mise en cohérence mené avec Bluetooth SIG rend maintenant celui-ci compatible avec cette structuration (intégration des données air à « l'Environnemental Sensing Service » adoptée le 14 septembre 2021 : <https://www.bluetooth.com/specifications/specs/gatt-specification-supplement-5/>).

Cas d'usage

La structure retenue est permissive et permet de traiter tous les niveaux de complexité et de complétude liés aux dimensions définies. Elle reste également ouverte sur la nature des mesures effectuées.

Interopérabilité

La solution retenue est applicable au niveau :

- Des équipements à faible complexité logicielle (ex. sensor) : solution accessible en C++ avec un nombre restreint de bibliothèques externes,
- Des différents types de réseaux (LoRa, Bluetooth, SigFox, TCP/IP) : les données sont utilisables sous format binaire (LoRa, Bluetooth) et sous format textuel JSON (TCP/IP)
- Des systèmes de traitements de données : La structure proposée est compatible avec les structures de haut niveau (ex. Xarray, Pandas, GeoPandas)
- Des Bases de données : L'intégration est native avec des bases de données NoSQL (ex. Mongo DB)
- Des stockages par Fichiers : Les principaux formats de fichiers sont pris en compte (ex. JSON / BSON, Tiff / geoTiff, csv, NetCDF, Shape file).

Modularité

Plusieurs mécanismes sont mis en place pour intégrer de nouveaux types de données et de nouvelles fonctionnalités :

- Ajout de « classes filles » à des classes déjà existantes,
- Ajout de données dans les dictionnaires,
- Utilisation de classes d'objets dédiées à des usages spécifiques,
- Ajout de connecteurs open-source ou privés.

4.2 STRUCTURE DE DONNEES

Une Observation est caractérisée par :

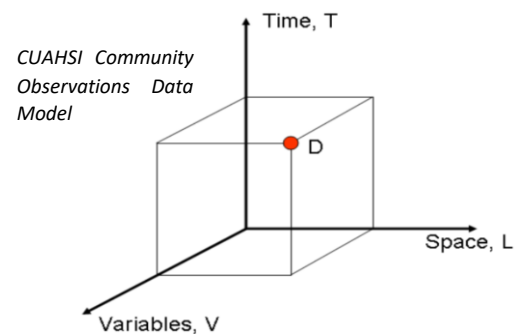
- "observed property" : la propriété observée,
- "feature of interest" : l'objet (le plus souvent un lieu) de l'observation,
- "procedure" : le mode d'acquisition de l'information (capteur, modèle...)
- "result" : résultat de l'observation ou de la mesure



Le résultat est un ensemble de valeurs référencées selon les 3 dimensions :

- Temporelle,
- Spatiale,
- Physique (propriété observée)

Il peut être converti en une matrice à 3 dimensions, chaque résultat étant indexée par des valeurs temporelles, spatiales et physiques.



Nota : Cette représentation indexée « domain range » est préférée à une représentation tabulaire « interleaved » qui associe à chaque valeur du résultat les valeurs temporelles, spatiales et physiques.

Des propriétés communes (indicateurs) sont associées à chaque Observation. Elles permettent d'effectuer des traitements sur les Observations sans avoir à en connaître la composition (ex. boîtes englobantes, type d'observation, volumétrie...).

interleave

Time	Space	Variable	Data

domain-range

Time	Space	Variable	Data



4.3 OPERATIONS

Une Observation peut être décrite partiellement et faire l'objet de compléments ultérieurs. On distingue plusieurs types d'opérations :

Les opérations directement liées aux Observations

- Fonctions de construction (création, modification, fusion, restriction).
- Fonctions de structuration (Indexation, normalisation)
- Fonctions d'analyse (complétude, typologie, indicateurs).

Les opérations liées à des formats de données

- Format textuel : format JSON
- Format binaire : format Bluetooth, LoRa
- Format fichier : TIFF, GeoTIFF, NetCDF, Shape file
- Format objet : Xarray, GeoPandas

Les opérations liées au traitement de ces données

- Recherche et extraction,
- Ajustement à un point de vue (granularité spatiale et temporelle, vue suivant des dimensions spécifiques),
- Traitements multiples et globaux (statistique, machine learning, assimilation)



4.4 CONNECTEURS

Les connecteurs sont les interfaces entre un objet Observation et une application ou un format d'échange.

La liste des connecteurs est extensible en fonction des besoins et concerne :

- Les capteurs,
- Les réseaux,
- Les formats neutres
- Les bases de données
- Les traitements de restitution (ex. visualisation)
- Les applications spécifiques (en entrée / sortie)
- Les modules d'analyse et de retraitement

4.5 UTILISATION

Les résultats d'une Observation peuvent être de différentes natures (ex. valeur numérique, texte, données structurées).

Les cas d'usage sont multiples et découlent des trois dimensions spatiales, temporelles et physiques.

On distingue notamment les usages suivants :

resultat	location	datation	usage	type
aucune	aucune	aucune	configuration : données de paramétrage	config
		unique	top d'indication de fonctionnement	top
		multiple	multi-top d'information	multiTop
	unique	aucune	geo localisation de l'émetteur	point
		unique	localisation ponctuelle de l'émetteur (tracking)	track
		multiple	localisation fixe	fixedTrack
	multiple	aucune	zoning : délimitation d'une zone	zoning
		unique	localisation d'un device réparti	multiTrack
		multiple	zone spatiotemporelle (déformable dans le temps) localisations multiples sur un trajet	timeLoc
unique	aucune	aucune	information sans dimension spatiale ou temporelle	measure
		unique	relevé de valeur	record
		multiple	relevé de valeur moyen	meanRecord
	unique	aucune	caractéristique locale	feature
		unique	mesure ou information ponctuelle	obsUnique
		multiple	mesure moyenne d'un capteur fixe	obsMeanFixed
	multiple	aucune	caractéristique d'une zone géographique	areaFeature
		unique	mesure moyenne sur une zone / grille (device réparti)	obsMeanArea
		multiple	mesure moyenne sur un trajet	meanTimeLoc
multiple	aucune	aucune	serie de mesures sans contexte	multiMeasure
		unique	mesure échantillonnée non localisée	multiRecord
		multiple	historique de mesures	measureHistory
	unique	aucune	mesures locales variables	featureVariation
		unique	mesure échantillonnée ponctuelle et localisée	obsSampled
		multiple	séquence de mesure d'un capteur fixe	obsSequence
	multiple	aucune	mesure sur une zone (device réparti)	measureLoc
			mesure sur une grille (device réparti)	
			mesure sur un profil vertical	
		unique	observation sur une zone (device réparti)	obsLoc
			observation sur une grille (device réparti)	
			observation sur un profil vertical	
		multiple	observation d'un trajet	obsTimeLoc
			observation sur une zone dans le temps	
			observation sur un profil vertical dans le temps	
			observation d'une grille dans le temps	

4.6 EXEMPLES

Pour plus de lisibilité, les exemples sont donnés dans le format JSON.

4.6.1 Mesure simple

Format réduit :

```
{
  "type": "observation",
  "dateTime": "2021-05-04T12:05:00",
  "coordinates": [14, 42.2],
  "propertyList": { "PropertyType": "PM10", "unit": "ppm" },
  "value": 120
}
```

Format complet (informations générées) :

```
{
  "type": "observation",
  "attributes": {
    "typeobservation": "obsUnique",
    "datation": {
      "typedatation": "date",
      "dateTime": "2021-05-04T12:05:00",
    },
    "location": {
      "typelocation": "point",
      "coordinates": [14, 42.2]},
    "property": {
      "typeproperty": "list",
      "propertyList": {"PropertyType": "PM10", "unit": "ppm"}},
    "result": {
      "typeresult": "real_int_str",
      "value": [120, [0,0,0]]},
    "information": {
      "nvalloc": 1,
      "nvaldat": 1,
      "nvalprop": 1,
      "nvalres": 1,
      "boundingBoxMin": [14, 42.2],
      "boundingBoxMax": [14, 42.2],
      "timeBoxMin": "2021-05-04T12:05:00",
      "timeBoxMax": "2021-05-04T12:05:00",
      "complet": true,
      "score": 111,
      "tauxMeasure": 1,
      "tauxEchantillon": 1,
      "dim": 0,
      "nEch": 1}
    }
  }
}
```

4.6.2 Mesure sur un trajet

Exemple d'une mesure en 3 point sur un trajet

Format réduit :

```
{
  "type": "observation",
  "dateTime": [
    "2021-05-04T12:05:00",
    "2021-07-04T12:05:00",
    "2021-02-04T12:05:00"
  ],
  "coordinates": [
    [14, 42.2],
    [24, 22.9],
    [20, 25]
  ],
  "propertyList": { "PropertyType": "PM10"},
  "value": [45, 5, 40]
}
```

4.6.3 Mesure multiple

Exemple d'une mesure en 3 point à 3 instants différents

Format réduit :

```
{
  "type": "observation",
  "dateTime": [
    "2021-05-04T12:05:00",
    "2021-07-04T12:05:00",
    "2021-02-04T12:05:00"
  ],
  "coordinates": [
    [14, 42.2],
    [24, 22.9],
    [20, 25]
  ],
  "propertyList": {"PropertyType": "PM10"},
  "value": [5,25,50,4,20,40,2,10,20]
}
```

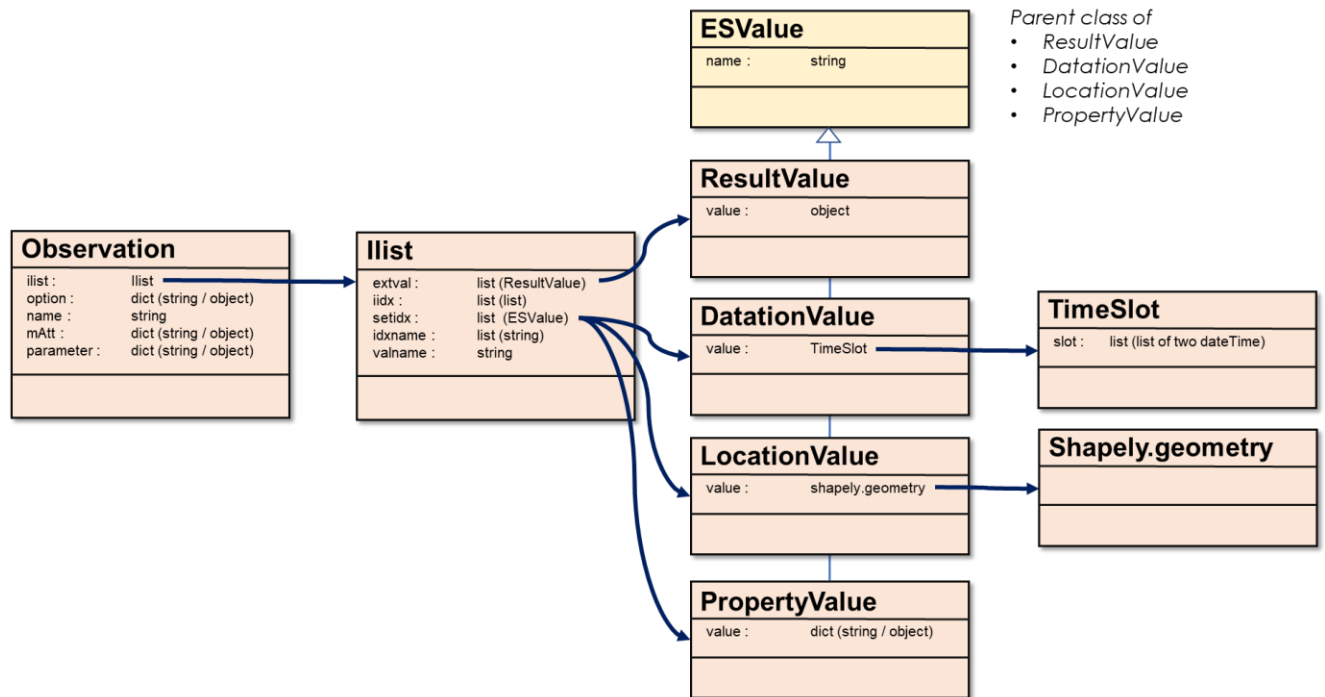
Format Xarray :

```
<xarray.Dataset>
Dimensions: (loc: 3, prop: 1, time: 3)
Coordinates:
  lon      (loc) float64 42.2 22.9 25.0
  lat      (loc) float64 14.0 24.0 20.0
  * loc     (loc) <U12 '[14.0, 42.2]' '[24, 22.9]' '[20, 25]
  * time    (time) datetime64[ns] 2021-05-04T12:05:00 ... 2021-02-04T12:05:00
  timestr   (time) <U19 '2021-05-04T12:05:00' ... '2021-02-04T12:05:00'
  * prop    (prop) object {'PropertyType': 'PM10'}
Data variables:
  value     (time, loc, prop) float64 5.0 25.0 50.0 4.0 20.0 40.0 2.0 10.0 20.0
```

5 STRUCTURATION

5.1 MODELE DE DONNEES

La structure des observations est la suivante :



Une Observation est associée à un objet Ilist regroupant à la fois le résultat de l'Observation (extval) et les caractéristiques observées (setidx).

Ces objets sont eux-mêmes composés d'éléments (ESValue) correspondant aux valeurs prises par les caractéristiques.

Le modèle général des classes d'objets est fourni en annexe.

5.1.1 Observation

Cette classe est constituée des attributs correspondants aux caractéristiques externes de l'Observation :

- Name : nom de l'observation effectuée
- ilist : caractéristiques observées et résultats de l'observation
- mAtt : attributs particuliers
- Parameter : Informations utilisateur
- Option : attributs définissant le mode de restitution et d'utilisation

Un objet Observation peut représenter aussi bien des objets simples de dimension 0 (ex une mesure unique ponctuelle) que des objets de dimensions 5 (3 dimensions au niveau spatial + 1 dimension temporelle + 1 dimension liée aux propriétés observées).

5.1.2 Ilist

Cette classe intègre les éléments suivants :

- Liste des valeurs observées (extval) et nom associé (valname)
- Liste des caractéristiques observées (setidx) et noms associés (idxname)
- Liens entre valeurs observées et caractéristiques (extidx)
- Des éléments internes (iidx, ival)

Les caractéristiques prises en compte sont les suivantes :

- Location :
 - ensemble d'objets « locationValue » (voir chapitre suivant),
 - GridLocation : objet décrivant un découpage spatial matriciel d'une zone (dans une future version)
- Datation :
 - ensemble d'objets « datationValue » (voir chapitre suivant),
 - GridDatation : objet décrivant un découpage temporel linéaire d'un intervalle (dans une future version)
- Result :
 - ensemble d'objets de type « resultValue » (voir chapitre suivant),
- Property :
 - ensemble d'objets de type « propertyValue » (voir chapitre suivant),

Ces types d'objets permettent de représenter la majorité des cas d'usage :

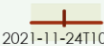



- Mesures ponctuelles issues de capteurs fixes ou mobiles,
- Mesures sur des zones non ponctuelles (ex. découpage géographique)
- Relevés matriciels (ex. relevé satellite)
- Mesures sur des intervalles de temps,
- Evolutions complexes (ex. panache de fumée)

5.1.3 ESValue

Cette classe intègre les objets terminaux des Observations.

Les objets ESValue sont constitués de plusieurs représentations selon des niveaux de complexité (voir figure ci-dessous) :

- Représentation textuelle,
- Représentation simplifiée
- Représentation détaillée

	Datation	Location	Property	Result
Textual	« morning »	« Paris »	« Air pollutant »	«High»
Simplified	 2021-11-24T10:43:20	 [2.35, 48.87]	type : PM25 Unit : µg/m3	25
Detailed	 [[[d1,d2], [d3,d4]]]	 [[[[30,20],[45,40],[10,40],[30,20]]], [[[15,5],[40,10],[10,20],[5,10],[15,5]]]]]	type : PM25 Unit : µg/m3 Sampling : instantaneous Period : 600 Uncertainty : 10%	Anything
Common properties	Boundary Conversion Representation	:	min – max detailed -> simplified json , binary	
Specific properties	Location Datation Property Result	:	Open Location Code (e.g. 8F W4 V7 5V + 8F) Year, Month, Day, Weekday... Catalog classification Conversion to float	

Ils sont composés des éléments suivants :

- **PropertyValue** : Caractérisation de la propriété observée
 - Simplifiée : identifiant de la propriété (issu d'un catalogue) et unité associée,
 - Détaillée : Ensemble des propriétés descriptives
- **ResultValue** : Résultat de la mesure
 - Simplifié : Une valeur numérique
 - Détaillée : Tout type d'objet
- **LocationValue** : Caractérisation d'un lieu
 - Simplifiée : point (coordonnées géodésiques ou projetées)
 - Détaillée : entité surfacique (ensemble de polygone). Ceci permet par exemple de représenter l'évolution d'un panache de fumée qui se transforme et se déplace ou encore d'identifier des zones géographiques (ex découpage cartographique administratif).
- **DatationValue** : Caractérisation d'un espace temporel
 - Simplifiée : instant (horodatage)
 - Détaillée : entité temporelle (ensemble d'intervalles temporels), ce qui permet de distinguer des mesures instantanées de mesures sur une durée.

5.2 TRAITEMENTS

Ils sont mis à disposition sur le site GitHub hébergeant les informations et développements réalisés (deux langages sont pris en compte actuellement : C++, Python).

On distingue :

- Les fonctions de construction :

- Création : à partir de composants élémentaires ou bien par structures textuelles au format Json
- Modification : ajout de données (mécanismes similaires à la création) ou suppression de données
- Extension : ajout d'une partie des données d'une autre Observation pour simplifier la construction d'une Observation
- Addition : L'addition de deux Observations consiste à cumuler les résultats des deux Observations sous un même format
- Réduction : Fonction inverse de réduction du nombre de résultats par filtrage
- Les fonctions de structuration :
 - Indexation / désindexation : représentation sous forme matricielle ou « à plat »
 - Complétion : ajout de données résultats vides (null ou nan) pour disposer d'un jeu de résultat complet
 - Normalisation : suppression des données inutiles et ordonnancement
- Les fonctions d'analyse :
 - Mesure de complétude (taux de mesure, taux d'échantillonnage)
 - Typologie d'observation (cas d'usage, dimension)
 - Indicateurs (nombre de valeurs)
 - Boîtes englobantes (temporelle ou spatiale)
- Les interfaces :
 - Format textuel : format JSON, GEOJSON, csv, IFC
 - Format binaire : format Bluetooth, LoRa
 - Format fichier : TIFF, GeoTIFF, Shapefile
 - Format objet : Xarray
- Les connecteurs :
 - LoRa, SigFox
 - Bluetooth,
 - RestFul
 - MongoDB
 - Pandas, Numpy, Xarray
 - LeafLet
 - ...

La liste des connecteurs est extensible en fonction des besoins.

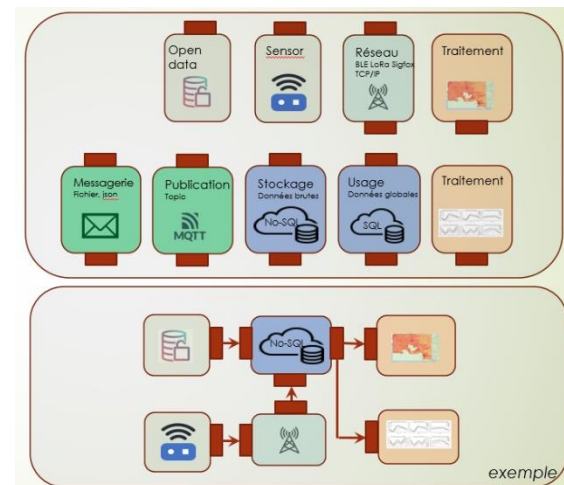
6 MISE EN ŒUVRE

6.1 PRINCIPES

L'utilisation des connecteurs permet de s'affranchir de la façon dont les données sont codées, ce codage étant masqué au sein de l'objet « Observation ».

Le partage et l'accès aux données s'effectue alors par un assemblage de connecteurs.

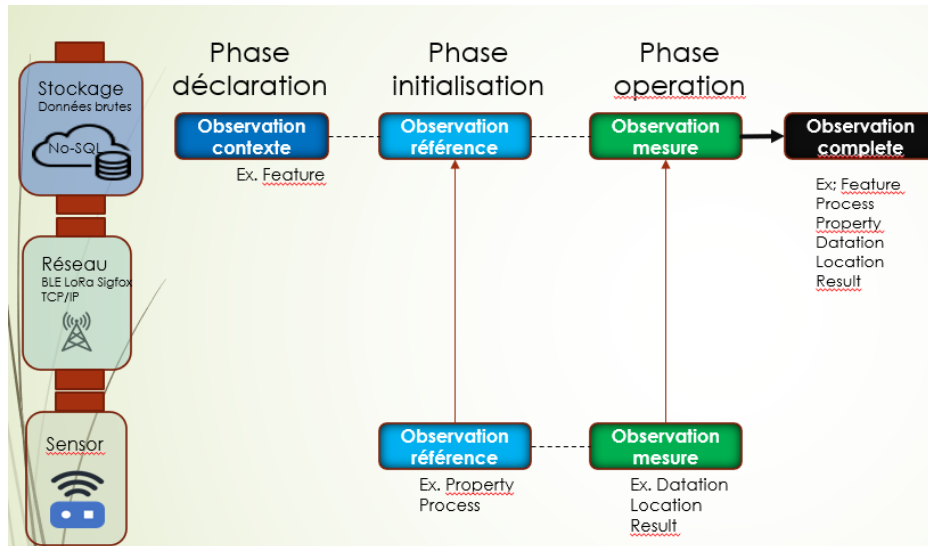
Le chapitre ci-dessous illustre cette mise en œuvre sur quelques exemples.



6.2 ACQUISITION PAR CAPTEUR

L'acquisition de données par capteur doit permettre de minimiser les flux de données tout en conservant une observation complète. Une observation complète est alors construite en plusieurs étapes :

- Phase de déclaration : Une observation décrivant le contexte observé est construite (ex. uniquement constituée d'un objet de type « interest »). Cette observation peut être créée au moment de la déclaration ou de l'activation du capteur.
- Phase d'initialisation : Une observation initiale est créée par le capteur. Celle-ci définit les propriétés mesurées et le mode d'acquisition (ex. objet de type « property » et « process »).
- Phase opération : Une observation faisant référence à l'observation initiale est créée pour chaque ensemble de mesures constituant l'observation (ex. objet de type « datation », « location », « result »).



L'observation complète est construite en agrégeant les trois observations créées (fonction d'extension).

L'application de ces principes permet d'être compatible avec une « charge utile » très faible. Par exemple, pour un usage du réseau Sigfox limité à 12 octets par message, on pourrait avoir :

- Une observation de référence constituée d'une propriété à mesurer,
- Une observation de mesure constituée d'une valeur mesurée et d'une localisation GPS.

6.3 ECHANGE DE DONNEES

A compléter (réseaux LPWAN, LAN, PAN, format de données)

6.4 STOCKAGE ET PARTAGE DE DONNEES

A compléter (fichiers, BD, API)

6.5 TRAITEMENT DES DONNEES

A compléter (environnements logiciels, outils d'analyse)

6.6 REPRESENTATION DES DONNEES

A compléter (dimension, granularité, graphique)

6.7 EXEMPLES

6.7.1 Capteur

L'exemple ci-dessous illustre un capteur mobile qui enregistre une information (ex. température) de façon périodique et la transmet dès qu'il est connecté à un réseau.

Le capteur (exemple : microcontrôleur programmable en C++) utilise les fonctions suivantes :

```
Observation obs = Observation(); // création d'un objet observation vide
int nprop1 = obs.init(PropertyValue("Temp", "°C")); // ajout d'une propriété correspondante aux mesures à effectuer
...
for (int i = 0; i < 6; i++) { // simule une boucle de mesure
    int nres = obs.addValueSensor(RealValue(25+i), TimeValue(2021, 5, 4+i, 12, 5, 0), Coordinates(14+i, 42), nprop1);
    // ajoute à l'objet obs une mesure constituée d'une valeur mesurée, de l'instant de mesure et des coordonnées
    // et l'associe à la propriété prop1
}
```

Le capteur envoie ensuite l'observation constituée des différentes mesures (ex format json pour un envoi via API Rest) :

```
string json_a_envoyer = obs.json();
```

Composition de la donnée Json générée :

```
{ "type": "observation", "propertyList": { "property": "Temp", "unit": "deg C", "coordinates": [[14, 42], [15, 42], [16, 42], [17, 42], [18, 42], [19, 42]], "dateTime": ["2021-05-04T12:05:00", "2021-05-05T12:05:00", "2021-05-06T12:05:00", "2021-05-07T12:05:00", "2021-05-08T12:05:00", "2021-05-09T12:05:00"], "value": [25, 26, 27, 28, 29, 30] }
```

6.7.1 Stockage en Base de données

La base de données doit pouvoir stocker tout type d'observation correspondant aux différents cas d'usage. C'est le cas des bases « NoSQL » de type « document » qui stockent directement des formats Json.

A titre d'exemple, l'envoi d'une Observation dans une base MongoDB s'effectue très simplement par une requête HTTP POST (ex. Python ci-dessous) :

```
url = "https://webhooks.mongodb-realm.com/api/client/v2.0/app/observ
r = rq.post(url, data=obs.json())
```

Pour faciliter les recherches, la donnée Json peut être enrichie d'informations complémentaires (exemple identique au cas ci-dessus) :

```
{ "type": "observation", "dateTime": ["2021-05-04T12:05:00", "2021-05-05T12:05:00", "2021-05-06T12:05:00", "2021-05-07T12:05:00", "2021-05-08T12:05:00", "2021-05-09T12:05:00"], "coordinates": [[14, 42], [15, 42], [16, 42], [17, 42], [18, 42], [19, 42]], "propertyList": { "property": "Temp", "unit": "deg C", "value": [25, 26, 27, 28, 29, 30], "information": { "typeobs": "obsPath", "typeloc": "multipoint", "typedat": "multidate", "typeprop": "list", "typeres": "multireal_int_str", "nvalloc": 6, "nvaldat": 6, "nvalprop": 1, "nvalres": 6, "boudingBoxMin": [14, 42], "boudingBoxMax": [19, 42], "timeBoxMin": "2021-05-04T12:05:00", "timeBoxMax": "2021-05-09T12:05:00", "complet": true, "score": 222, "tauxMeasure": 1, "tauxEchantillon": 1, "dim": 1, "nEch": 6} }
```

6.7.2 Extraction d'une Base de données

La recherche d'Observation dans une base NoSQL permet d'exploiter plusieurs mécanismes :

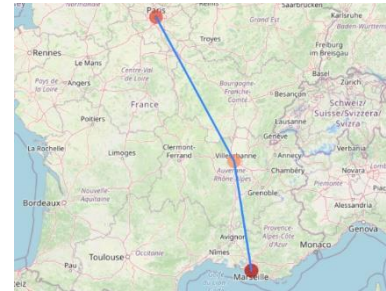
- Recherche textuelle (base indexée au format texte)
- Recherche par boîte englobante spatiale ou temporelle (informations boundingbox et timebox)
- Recherche par attributs sur la base des attributs stockés

Le résultat de la recherche est une liste d'Observation qui peuvent être fusionnées en une seule pour faciliter les traitements ultérieurs.

6.7.3 Visualisation d'une Observation

Une observation peut être restituée de plusieurs façons :

- représentation 1D : courbe, familles de courbes,
- représentation 2D : surfaces et familles de surfaces,
- représentation 3D : volumes et familles de volumes,
- représentation sur des supports cartographiques,
- représentation sous forme de vidéos temporelles

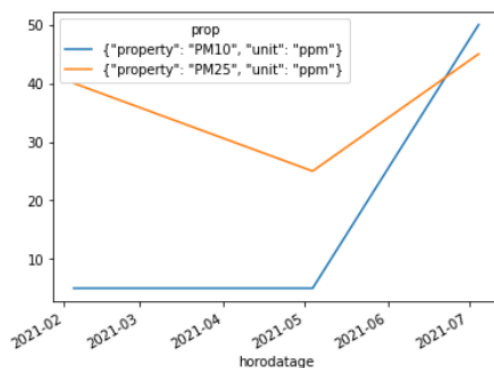


Exemple d'une trajectoire :

```
{ "type": "observation", "dateTime": ["2021-05-04T00:00:00", "2021-07-04T12:05:00", "2021-02-04T12:05:00"], "coordinates": [[48.87, 2.35], [45.76, 4.83], [43.3, 5.38]], "propertyList": [{"property": "PM10", "unit": "ppm"}, {"property": "PM25", "unit": "ppm"}], "value": [5, 25, 50, 45, 5, 40] }
```

La visualisation s'effectue par une simple commande :

```
ob = Observation(text_json)
ob.plot()
```

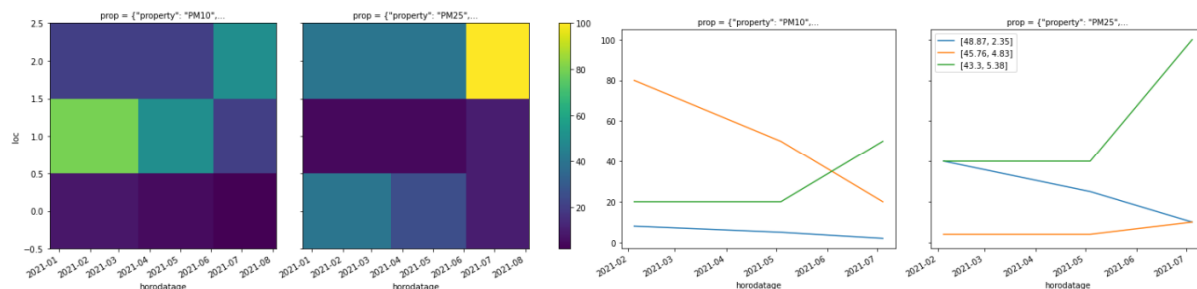


Exemple d'une grille :

```
{ "type": "observation", "dateTime": ["2021-05-04T00:00:00", "2021-07-04T12:05:00", "2021-02-04T12:05:00"], "coordinates": [[48.87, 2.35], [45.76, 4.83], [43.3, 5.38]], "propertyList": [{"property": "PM10", "unit": "ppm"}, {"property": "PM25", "unit": "ppm"}], "value": [5, 25, 50, 4, 20, 40, 2, 10, 20, 10, 50, 100, 8, 40, 80, 4, 20, 40] }
```

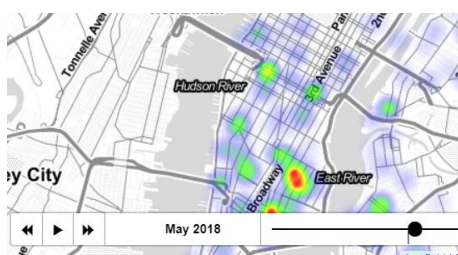
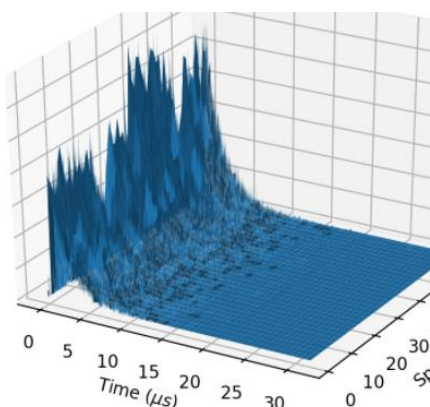
La visualisation s'effectue par la même commande :

```
ob = Observation(text_json)
ob.plot()
```

Exemple de représentation spatiale :

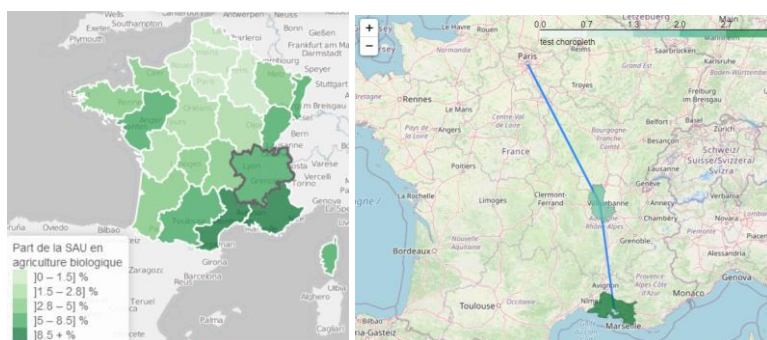
Des représentations spatiales intégrant également la dimension temporelle sont également intégrées (ex. via Pyplot ou bien via Leaflet).



Exemple d'une choroplèthe :

Il s'agit de représentations intégrant des espaces géographiques définis (ex. ci-dessous avec des espaces correspondant à des découpages administratifs).

La seule différence avec les exemples précédents est de remplacer les points (définis par ses coordonnées) par des polygones (listes de points) délimitant les espaces.



7 FAQ

Quelles sont les données concernées ?

Toutes les données numériques ou alphanumériques associées à un thème et/ou une localisation et/ou une datation. C'est un périmètre très large qui dépasse le cadre des données environnementales.

Cela concerne par ailleurs aussi bien des données unitaires (ex. issues d'un capteur) que des données groupées spatialement (ex. issue d'un relevé satellite) ou temporellement (historique de mesures).

Pourquoi un nouveau standard alors qu'il en existe déjà ?

Il n'existe pas de standards opérationnels adressant aussi bien des données simples (ex. issues de capteurs) que des données complexes intégrant les trois dimensions (spatiale, temporelle, physique) et couvrant l'ensemble du cycle de vie des données.

Les standards existants sont génériques (ex. ISO « Observation & Measurement ») ou bien sont spécialisés soit sur des données particulières (ex. données géographiques), soit sur des étapes spécifiques du cycle de vie (ex. communication Bluetooth), soit sur des structures de données génériques (ex. Xarray / Pandas) pour les données multidimensionnelles, soit sur des outils spécifiques (ex bibliothèque graphique Leaflet).

Il s'agit donc plus de mettre en cohérence les différents standards et de les rendre applicables à tout type de situation. Par exemple, le standard Bluetooth « Environmental Sensing » n'intégrait pas les données liées à la qualité de l'air, celles-ci sont maintenant intégrées.

Pourquoi ce standard serait-il utilisé ?

C'est lié à une demande d'interopérabilité croissante :

- Il est difficilement compréhensible de devoir accéder à des données environnementales avec des moyens et des applications différentes si l'on se trouve par exemple au domicile, sur son lieu de travail, dans un lieu public ou en pleine nature. La seule réponse à cette attente est de s'appuyer sur un standard de données commun.
- Le standard est intégré dans des « connecteurs » publics et open-source utilisables pour chaque type d'environnement logiciel (ex. C++, Python, javascript), chaque type de réseau LAN, LPWAN, PAN et chaque type de cas d'usage
- Les connecteurs simplifient les activités de développement, d'intégration ou de commercialisation puisqu'ils prennent en charge les opérations fastidieuses

de codage/décodage (ex entre capteur, réseau, base de données, visualiseur, carte, logiciel de modélisation, fichiers...)

Pourquoi s'intéresser à des capteurs dont le cout est de quelques euros ?

Ce sont les matériels qui ont la plus forte contrainte puisqu'ils disposent de fonctions logicielles réduites. Si cela fonctionne sur ce type de matériel, c'est généralisable à tout type de matériel.

Par ailleurs, la majeure partie du cout de mise en œuvre d'un capteur est liée à son intégration dans un système complet qui peut être fortement réduit dans un environnement standardisé.

Pourquoi structurer les résultats par des index vers chacune des dimensions plutôt qu'une représentation matricielle ?

La représentation individuelle par index présente l'avantage de pouvoir traiter avec un faible volume de données les cas d'usage présentant peu de mesures au regard des dimensions temporelles et spatiales concernées (notion de « matrice creuse »). Néanmoins, les deux représentations sont équivalentes et on peut passer de l'une à l'autre sans perte d'information. Ceci permet l'utilisation des outils matriciels pour exploiter ces données (ex. interface Xarray).

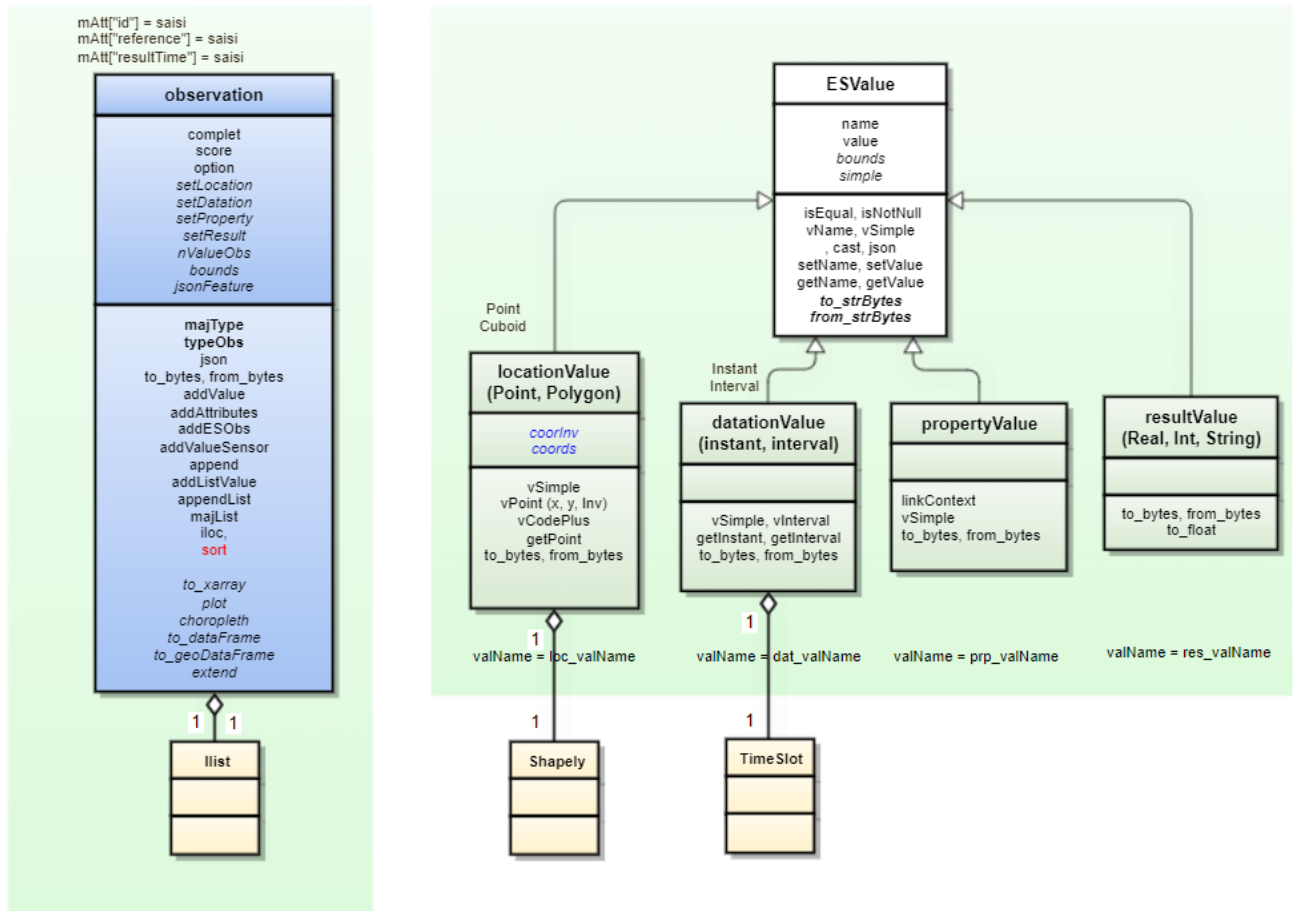
8 ANNEXES

- I. Annexe 1 : Modèle de classes
- II. Annexe 2 : Mapping Bluetooth
- III. Annexe 3 : Représentation binaire
- IV. Annexe 4 : Représentation textuelle
- V. Annexe 5 : API
- VI. Annexe 6 : Mapping formats XD

I. Annexe 1 : Modèle de classes

Le diagramme ci-dessous représente :

- En blanc : les classes abstraites
- En bleu, vert et rose : Les classes instanciables

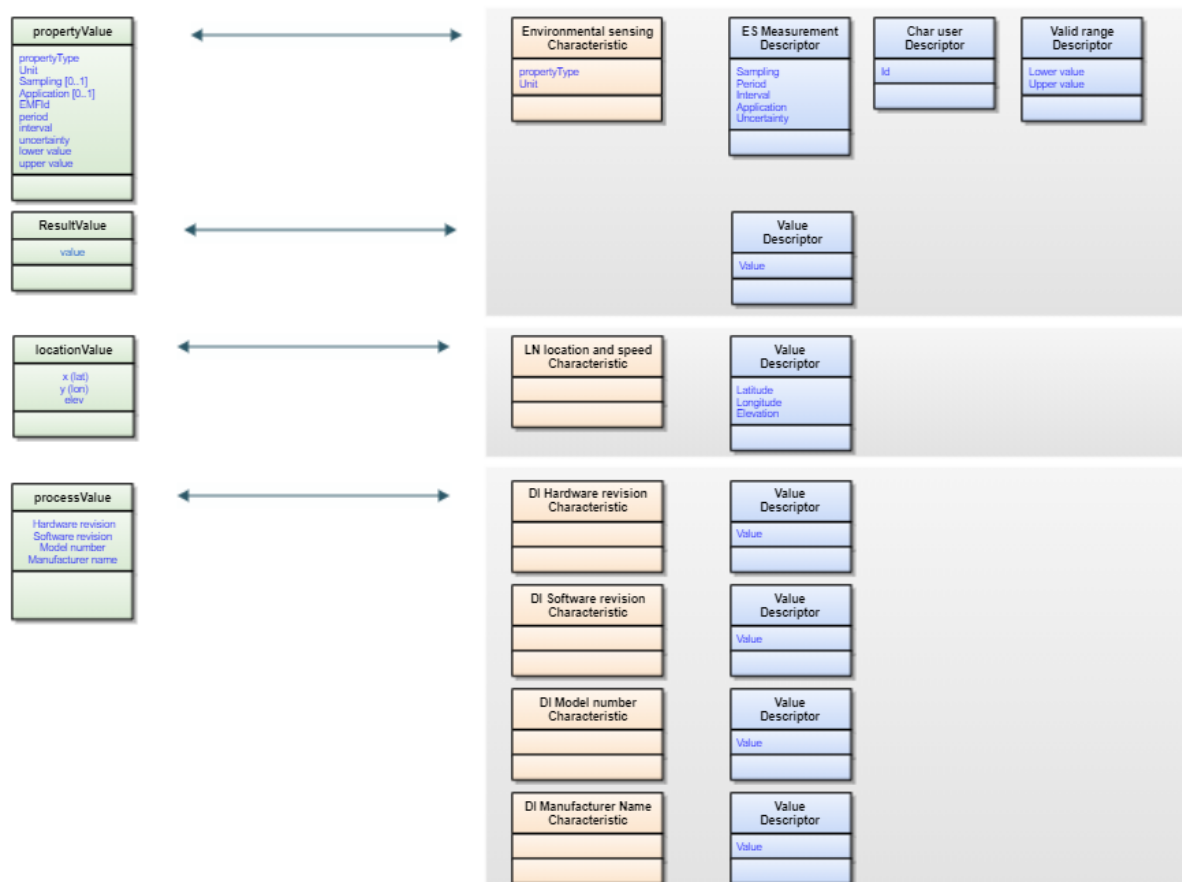


II. Annexe 2 : Mapping Bluetooth

Le diagramme ci-dessous représente la correspondance du modèle avec les données Bluetooth décrites dans les Services Bluetooth suivants :

- Environmental Sensing Service
- Location and Navigation Service
- Device Information Service

Mapping Bluetooth



III. Annexe 3 : Représentation binaire

Le diagramme ci-dessous représente la représentation binaire d'une observation.

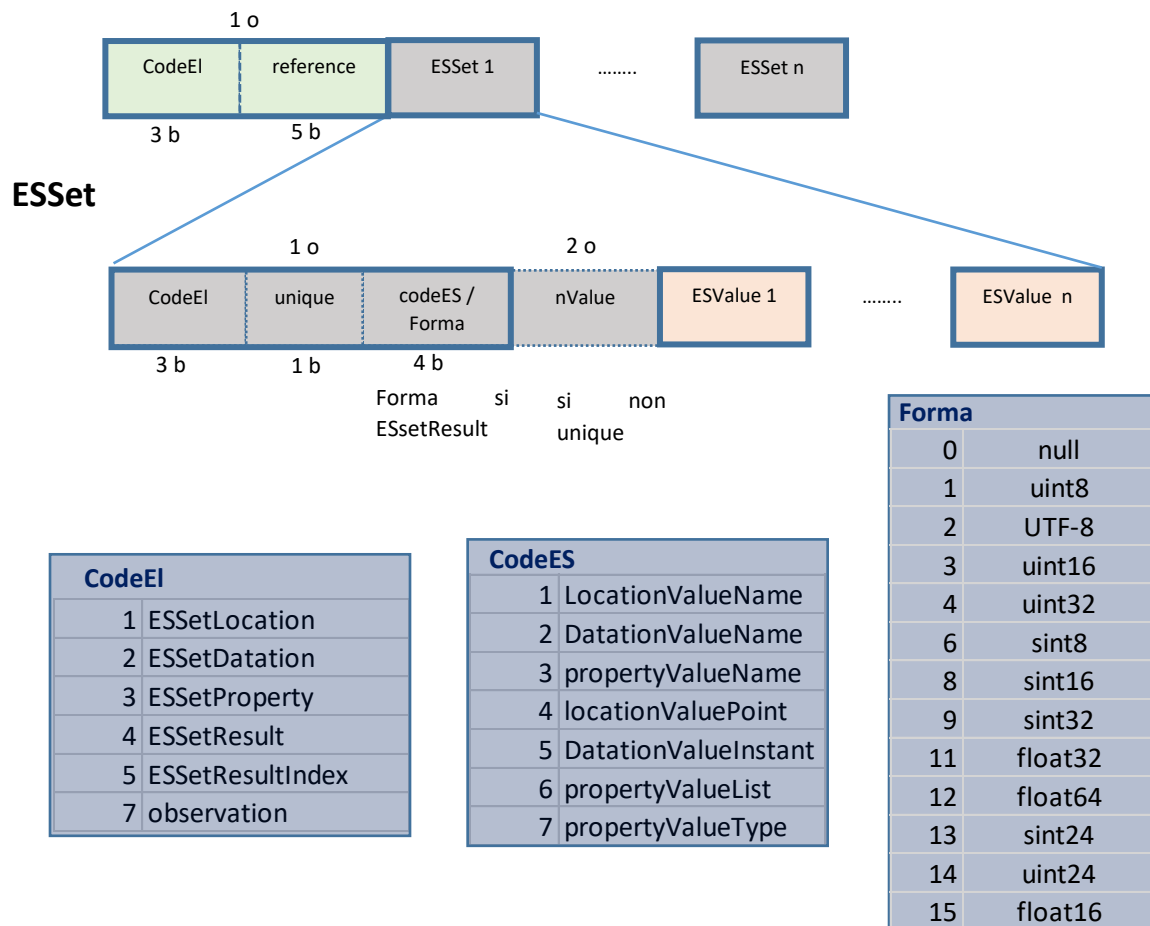
Une observation est constituée de :

- 1 octet de déclaration de l'observation (type d'élément sur 3 bits et référence de l'observation sur 5 bits)
- Une liste d'objets de type ESObjs

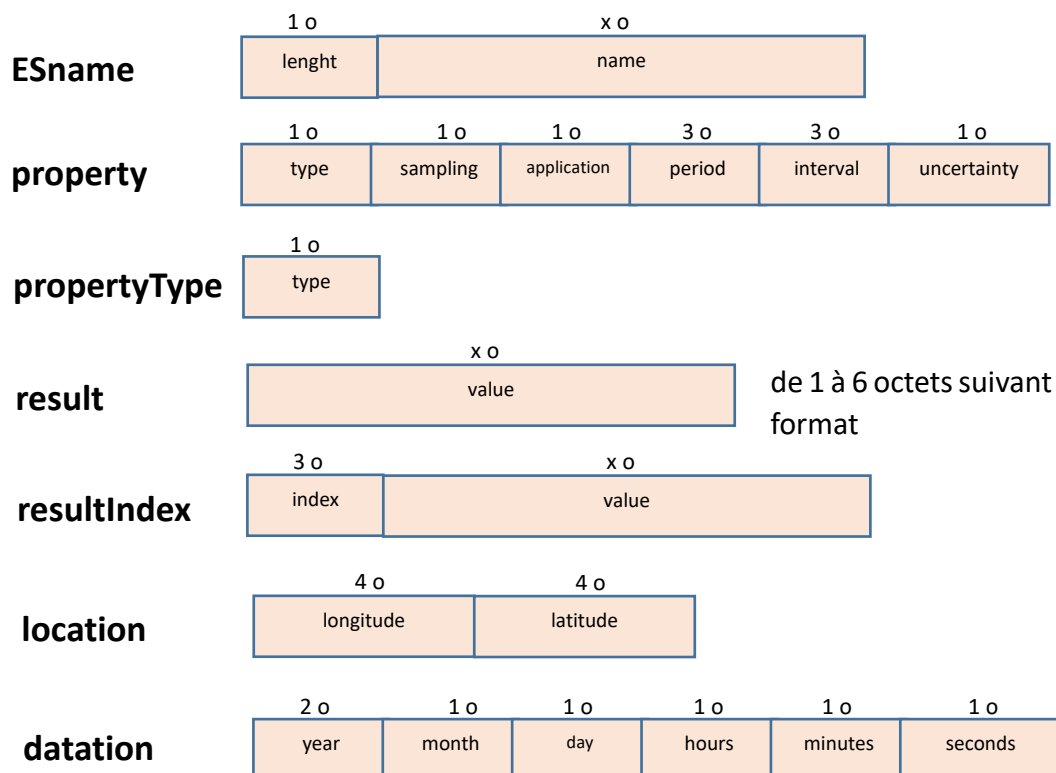
Un objet ESObjs de type ESSet est constitué de :

- 1 octet de déclaration de l'ESObjs (type d'élément sur 3 bits, nature l'ensemble (unique ou non) sur 1 bit, format des ESValue qui le composent sur 4 bits)
- 2 octets indiquant le nombre de valeurs ESValue
- Une liste d'ESValue

Observation



Un objet ESValue à un format spécifique suivant la nature de la valeur représentée :



Les formats définis pour les objets property, result, location, datation sont identiques à ceux définis par le service Environnemental Sensing de Bluetooth.

IV. Annexe 4 : Représentation textuelle

Le format JSON adapté à la structure de données (format ObsJSON) est décrit dans un document spécifique (voir site de partage des documents).

V. Annexe 5 : API

A compléter : requêtes simples, niveaux de complexité

VI. Annexe 6 : Mapping formats XD

A compléter : formats Xarray, Pandas, Numpy, GeoPandas