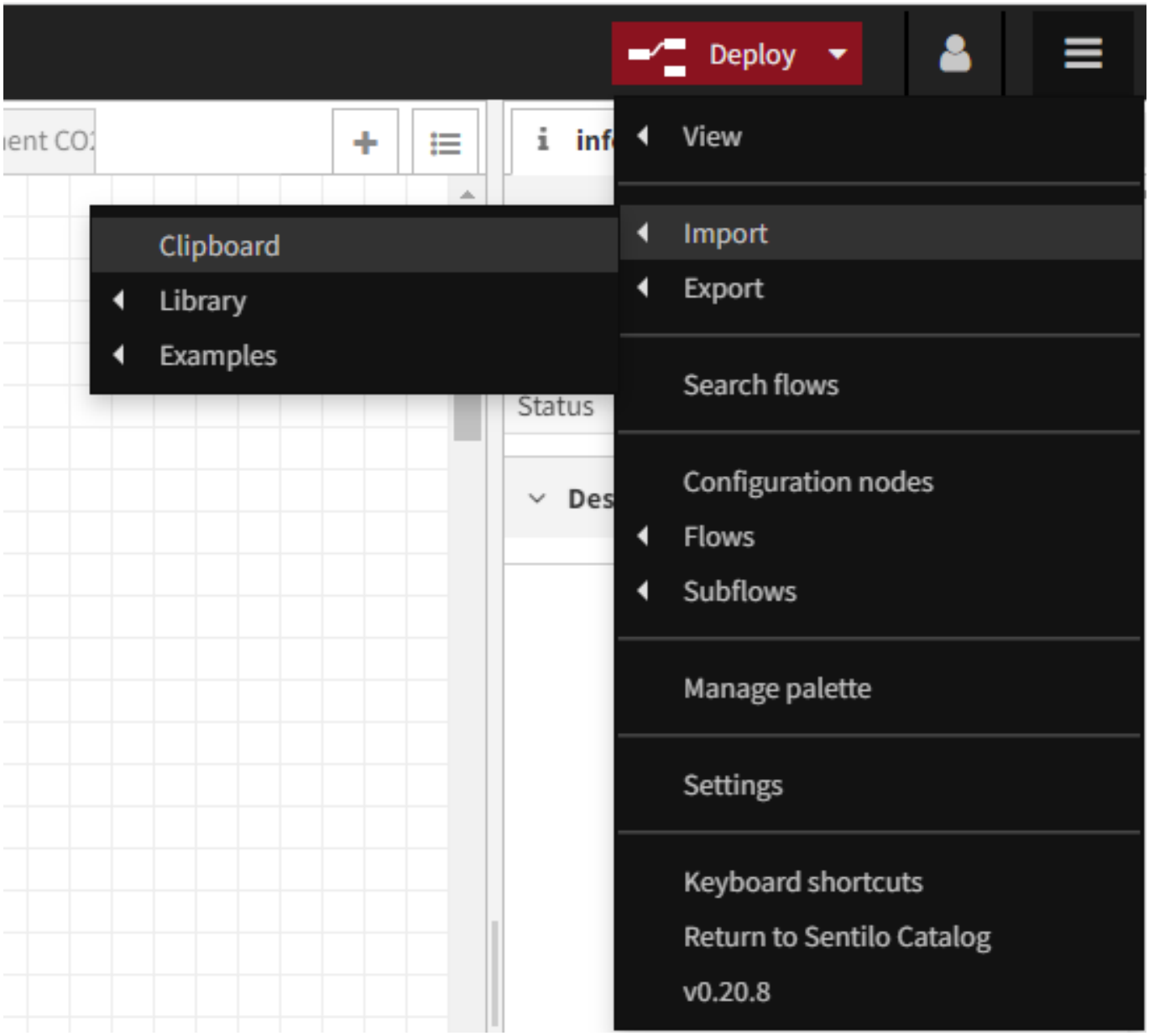
5. Fluxos Node-RED

Taula de continguts

1. Importar fluxos Node-RED de la Wiki

El flux s'importa a la instància Node-RED accedint a:

"Menú hamburguesa -> Import -> Clipboard" i marquem l'opció *"Import to new flow"*.

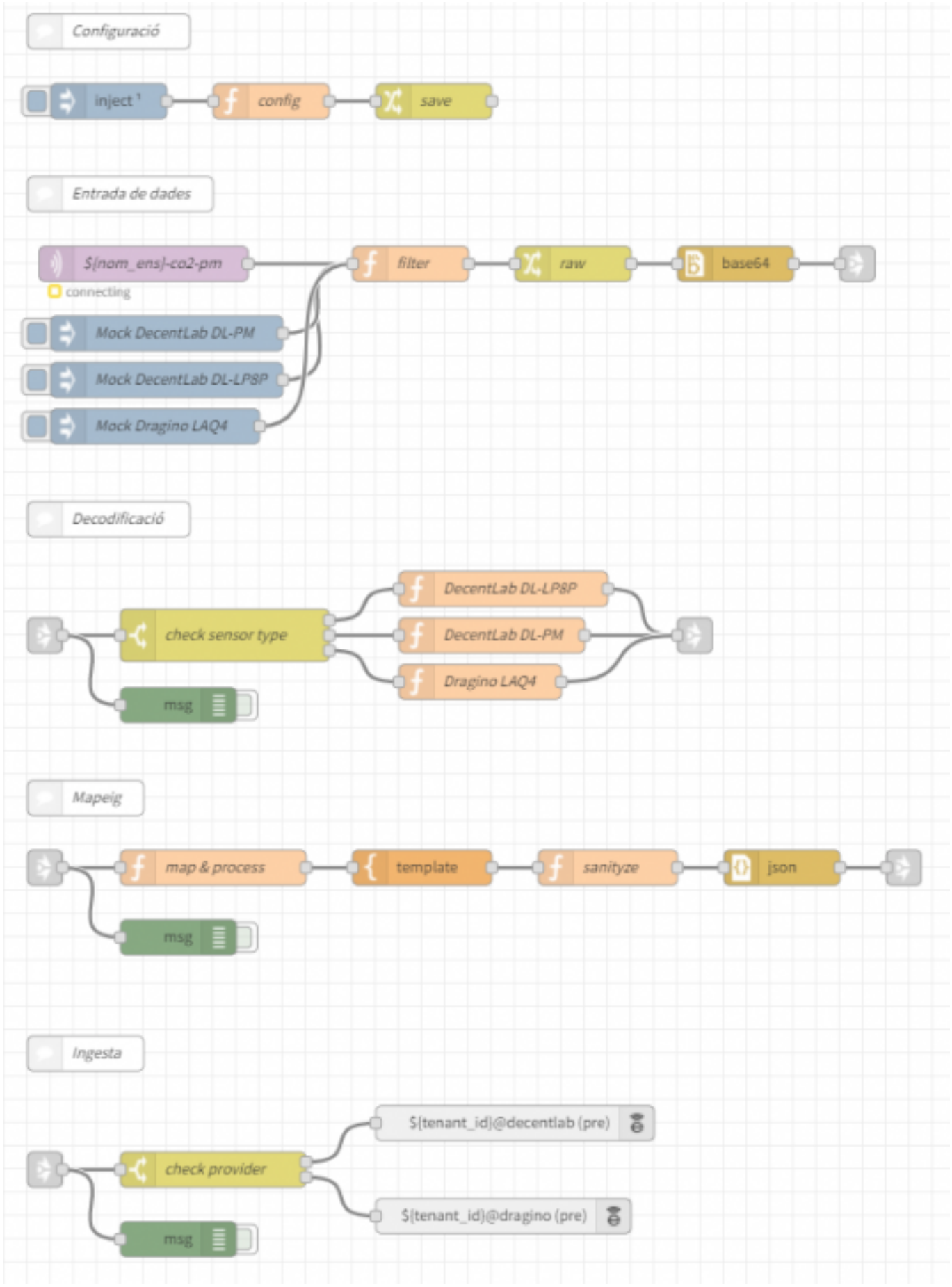


2. Sostenibilitat i medi ambient

2.1. LoraWan - Flux d'integració de dades de sensors de qualitat de l'aire (descarrega [1])

Agraïr l'aportació desinteresada de l'Ajuntament d'Arenys de Munt amb aquest flux. En concret s'integren dades de dispositius dels fabricants DecentLab i Dragino.

Amb la col·laboració de Femprocomuns han fet un esforç d'adaptació important del flux per a que sigui reutilitzable per altres municipis i altres dispositius de diferents fabricants.



El flux està dividit en 5 etapes:

1. **Configuració**, s'injecta la configuració amb els dispositius que es volen monitorar. Per cada dispositiu es registra l'identificador, tipus de dispositiu, proveïdor associat a Sentilo i un "mapeig" de camps entre els que torna el dispositiu i com es diuen els sensors associats a Sentilo.
2. **Entrada de dades**. Es reben les dades des del servidor LoraWAN, es filtren per processar només els dispositius que volem i preprocessa l'entrada de dades per tenir un "buffer" de bytes.
3. **Descodificació**. En funció del tipus de dispositiu es fa passar el "buffer" de bytes per un descodificador o un altre. Aquests descodificadors són els proporcionats pel fabricant del dispositiu (en el codi s'inclou la referència) però modificats per tal que la sortida sigui homogènia entre diferents fabricants. A aquestes alçades tindrem un llistat de camps i valors.
4. Es "**mapegen**" els camps que torna el descodificador del dispositiu a sensors a Sentilo basant-nos en la informació de configuració. La sortida es passa per una plantilla i es neteja per obtenir un missatge preparat pel node de publicació de Sentilo.
5. S'**ingereixen** les dades a Sentilo en funció del proveïdor associat en aquest. Un únic missatge genera una única crida al node "publish" de Sentilo amb dades dels N sensors associats a aquell dispositiu.

S'ha dissenyat modularment amb l'objectiu de homogeneïtzar els fluxos per afavorir la replicació i manteniment del codi.
Permet identificar quins són els nodes i variables específics per cada sensor i fabricant.

Autoria: xoic@femprocomuns.coop [2]

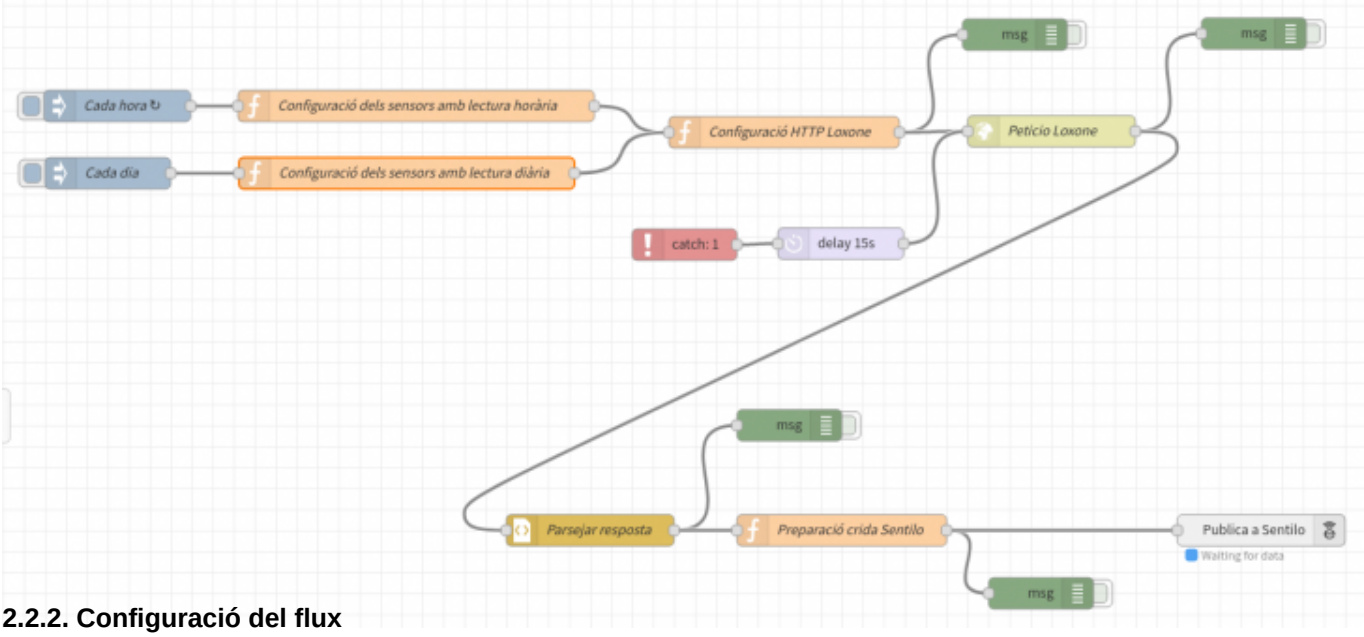
2.2. PLC's Loxonne - Flux d'integració de dades (descarrega [3])

2.2.1. Descripció del funcionament

El flux utilitza l'API de Miniserver de Loxone per obtenir les dades actuals d'un sensor concret. Existeix un servei de núvol de Loxone que ofereix accés al Miniserver sense necessitat de conèixer la IP de Miniserver prèviament - només és necessari saber el ID de Miniserver.

Bàsicament, el flux segueix aquests passos:

1. Llegeix la configuració dels sensors, cada hora o una vegada al día
2. Per cada sensor fa una crida a l'API de Loxone i recupera la dada actual del sensor en format XML
3. Si la connexió en el punt anterior falla, torna a provar
4. Parseja la resposta XML i obté el valor
5. Publica el valor a Sentilo.



2.2.2. Configuració del flux

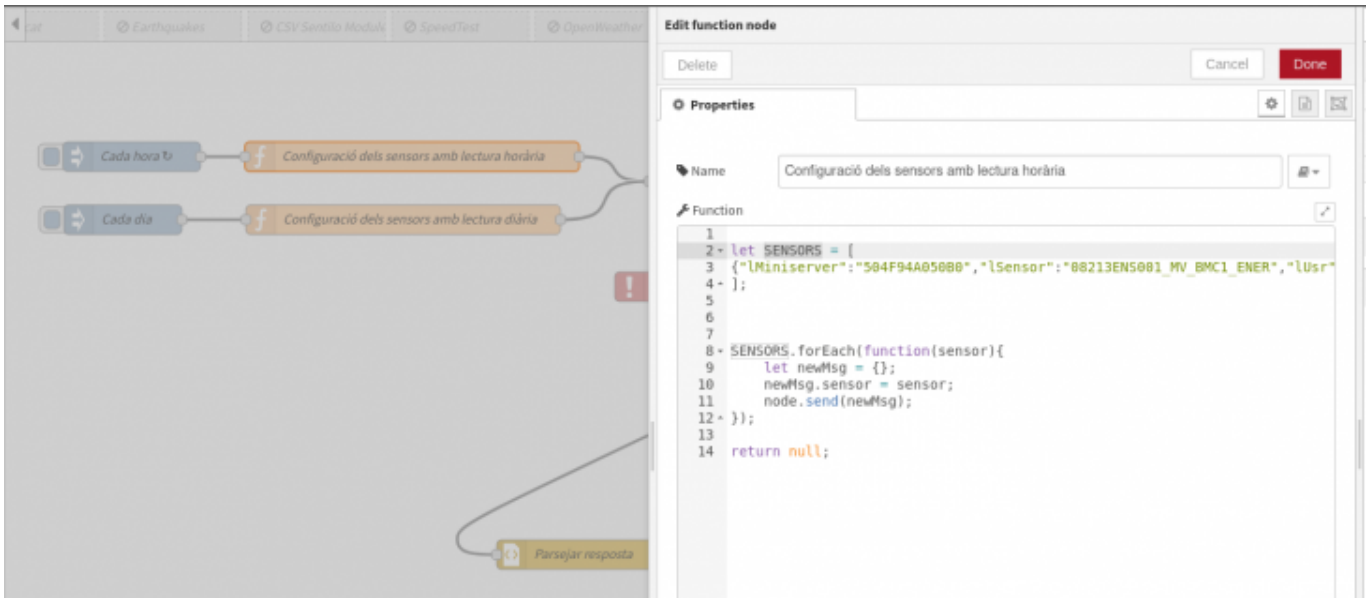
Configuració de la connexió a Sentilo

Si fem un doble clic sobre el node *"Publica a Sentilo"*, podem escollir una connexió a Sentilo de les existents o configurar una nova.

Alternativament, podem esborrar la connexió importada i crear o reutilitzar una connexió que ja tenim dins de la nostra instància de Node-RED.

Configuració dels sensors a importar

Dins dels nodes Function *"Configuració dels sensors amb lectura horària"* i *"Configuració dels sensors amb lectura diària"* hi ha una array javascript SENSORS.



Cada element de la array és una configuració de sensor amb les següents atributs:

- Miniserver: ID de Miniserver Loxone
- Sensor: ID de Sensor Loxone
- Usr: Usuari d'accés a Miniserver
- Pass: Contrasenya d'accés a Miniserver
- sSensor: ID de sensor de Sentilo

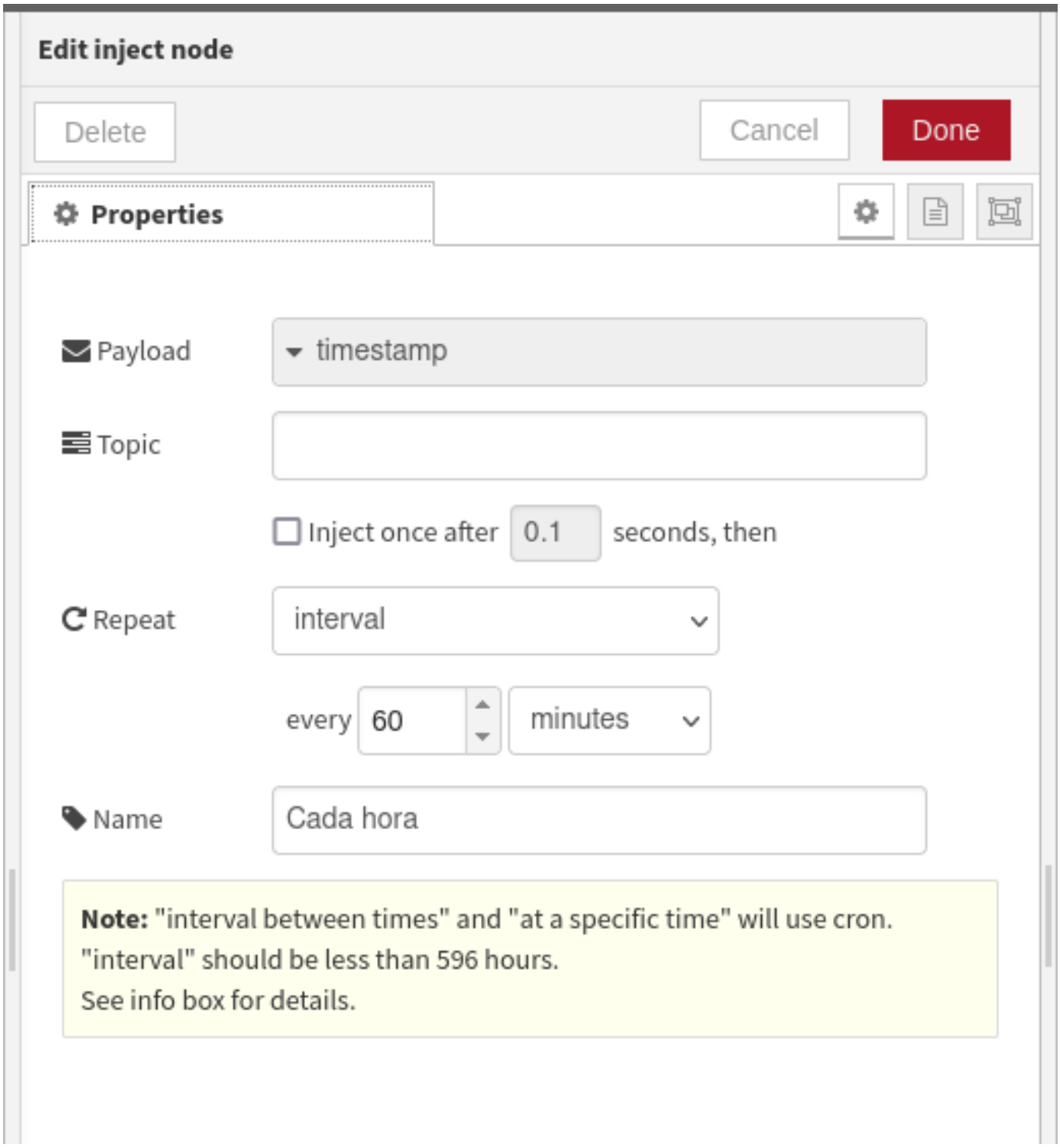
Per exemple, la definició de sensor seria:

{
"lMiniserver":"504F94A050B0",
"lSensor":"08213ENS001_MV_BMC1_ENER",
"lUsr":"x",
"lPass":"x",
"sSensor":"08213ENS001_MV_BMC1_ENER"
}

El flux processarà tots els elements de la array SENSORS, si està activat el seu Inject Node (veure el següent apartat).

Programació periódica de l'execució

Dins del flux hi ha 2 injectors, cadascun pensat per planificar l'execució amb una freqüència different. Per defecte estan desactivats. Si volem activar la execució de cada hora, hem de accedir a la configuració d'Inject Node *"Cada hora"* i configurar-ho d'aquesta forma:



Desplegament del flux

Finalment publiquem les nostres canvis amb el botó *"Deploy"*.

En cas d'incidència, podem activar els nodes *"Debug"* i revisar el contingut dels missatges en la pestanya log.

2.3. Meteocat - Flux d'integració de dades meteorològiques

Des d'Octubre 2024 aquesta integració de les estacions de la XEMA de Meteocat es realitza de forma agrupada des de Smart Region: si esteu interessats en integrar aquestes dades sobre la vosta instància de la Plataforma Smart Region, envieu un correu a smartregion@diba.cat **[4]** i us activarem aquesta integració de dades directa.

2.4. Qualitat de l'aire - Flux d'integració de dades de Qualitat de l'Aire

Des de Novembre 2024 aquesta integració de les estacions de la XPVCA de dades obertes es realitza de forma agrupada des de Smart Region: si esteu interessats en integrar aquestes dades sobre la vosta instància de la Plataforma Smart Region, envieu un correu a **smartregion@diba.cat** [5] i us activarem aquesta integració de dades directa.

2.5. Punts de recàrrega - Flux d'integració de dades dels punts de recàrrega de vehícles elèctrics

Aquesta integració de les estacions o punts de recàrrega de vehícles elèctrics es realitza de forma agrupada des de Smart Region: si esteu interessats en integrar aquestes dades sobre la vosta instància de la Plataforma Smart Region, envieu un correu a smartregion@diba.cat **[6]** i us activarem aquesta integració de dades directa.

3. Adaptacions necessàries dels fluxos per actualització de la versió del Node-Red a v0.5.1

L'actualització de **Node-RED a la versió 2.2.2**, també ha aportat actualitzacions d'alguns nodes, com és el cas dels de ***Sentilo***, que passen de la **versió v0.1.5 a la v0.5.1**.

Per tal de poder continuar fent servir els nostres fluxos que utilitzen aquests nodes, caldrà fer una petita intervenció.

A continuació es detalla com cal actuar per tal d'adaptar els fluxos a la nova versió si és necessari.

3.1. Detalls de la versió v0.1.5

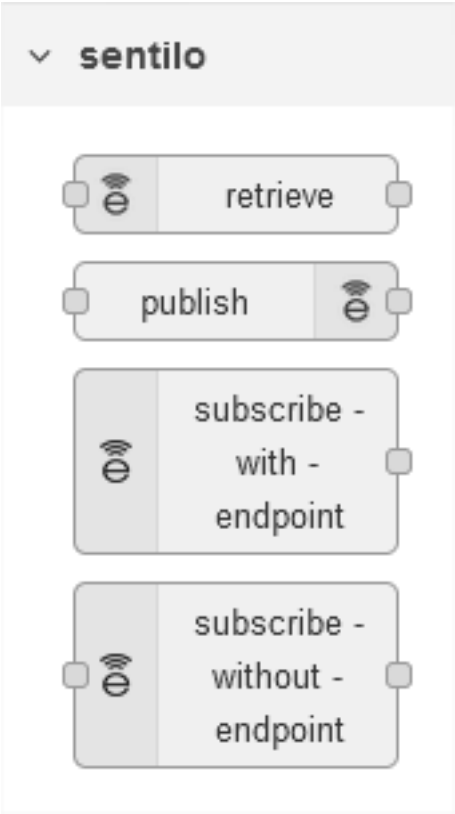
Si recordem els nodes de la versió anterior, teníem la següent col·lecció:

que estava formada per 3 nodes funcionals i un de configuració del servidor (node intern):

- ***retrieve:*** permet obtenir dades de Sentilo, i les ofereix a la seva sortida
- ***publish:*** permet publicar dades a Sentilo, injectades com a entrada en format *observation*
- ***subscribe:*** permet subscriure's a un recurs de Sentilo (en el moment del desplegament del fluxe), i ens ofereix la dada publicada a la seva sortida, en format *observació*
- ***server:*** ofereix un espai de configuració per a la connexió a Sentilo, que es pot fet servir a la resta de nodes del fluxe

3.2. Detalls de la versió v0.5.1

La nova versió de nodes de Sentilo, la v0.5.1, ofereix els mateixos nodes funcionals, i el node de servidor. A part, s'ha introduït un nou node de subscripció:



un cop més tenim:

- ***retrieve:*** permet obtenir dades de Sentilo, i ofereix dos dades de sortida
  - sortida 1: resposta de l'api de Sentilo, amb la dada sol·licitada
  - sortida 2: codi http de resposta del servidor de Sentilo
- ***publish:*** permet publicar dades a Sentilo, injectades com a entrada en format *observation*, i ofereix dos sortides:
  - sortida 1: només en cas d'error, emetrà el missatge retornat per l'api de Sentilo, o un missatge buit si tot ha anat bé
  - sortida 2: codi http de resposta del servidor de Sentilo
- ***subscribe with endpoint:*** permet subscriure's a un recurs de Sentilo (en el moment del desplegament del fluxe) i rebre la dada al mateix node, tenim 3 sortides:
  - sortida 1: ofereix la dada a la qual estem subscrits en el moment de la seva publicació (actua com a *endpoint*)
  - sortida 2: ofereix la resposta de Sentilo a l'event de subscripció (només un cop, quan fem el desplegament del fluxe)

    - sortida 3: ofereix el codi de resposta http que ens retorna el servidor en el moment de fer la subscripció
- ***subscribe without endpoint:*** aquest node és una modificació del node *subscription with endpoint* de anterior, però amb la particularitat que accepta dades d'entrada (de configuració), i no genera dades de sortida quan es publica una dada a la que estem subscrits (no crea un *endpoint*), sinò que les reenvia cap a una adreça remota (de fet, el que fa és generar una subscripció cap a la url que informarem a la dada *Callback URL* de la seva configuració).
  El funcionament segueix sent molt semblant al cas anterior, però com a sortida disposem de dos canals que podem explotar:
    - sortida 1: ofereix la resposta de Sentilo a l'event de subscripció (només un cop, quan fem el desplegament del fluxe)
    - sortida 2: ofereix el codi de resposta http que ens retorna el servidor en el moment de fer la subscripció
- ***server:*** ofereix un espai de configuració per a la connexió a Sentilo, que es pot fet servir a la resta de nodes del fluxe

Com veiem, els nodes *publish* i *retrieve* mantenen la seva funcionalitat, i afegeixen una nova sortida informativa.

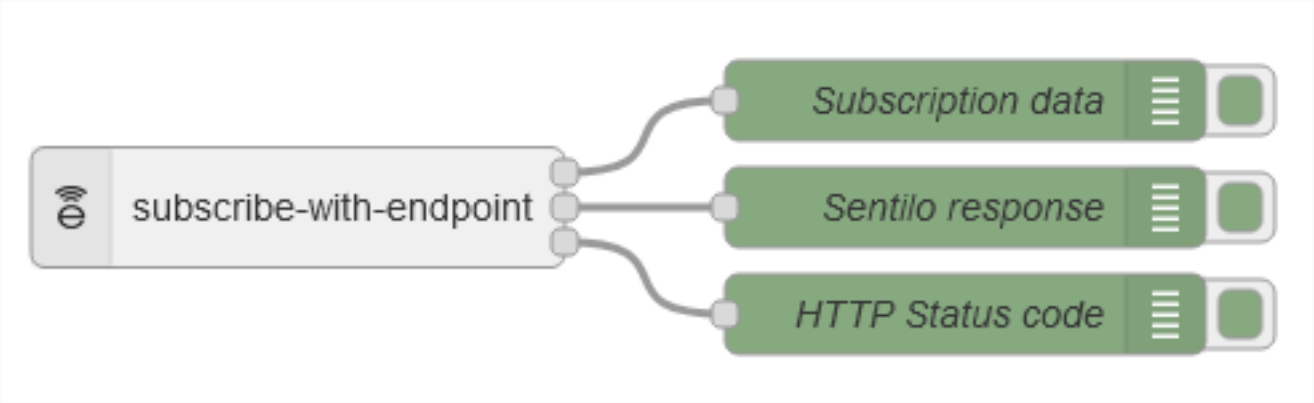
Per contra, els dos nodes de *subscripció* han canviat. El primer, *subscribe with endpoint* és l'equivalent a l'original de la versió anterior, mentres que el *subscribe without endpoint* és un node nou que afegeix una nova funcionalitat.

A continuació es mostra un conjunt d'imatges amb l'estructura funcional de cadascun d'ells, i com explotar la seva sortida:

Retrieve

Publish

Subscribe with endpoint



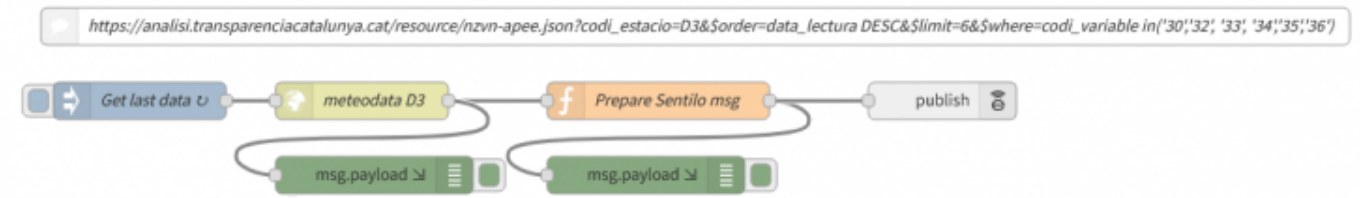
Subscribe without endpoint

En aquest darrer cas cal tenir en compte el node *function* anomenat *Prepara subscripció* que és l'encarregat d'injectar les dades de configuració del node de subscripció.

3.3. Com adaptar fluxos a la nova versió dels nodes de Sentilo

Per tal d'adaptar els nostres fluxos a la nova versió de Sentilo v0.5.1, cal seguir unes passes molt senzilles que tot seguit explicarem.

Prendrem com a referència el fluxe de test ***Meteocat Simple***:



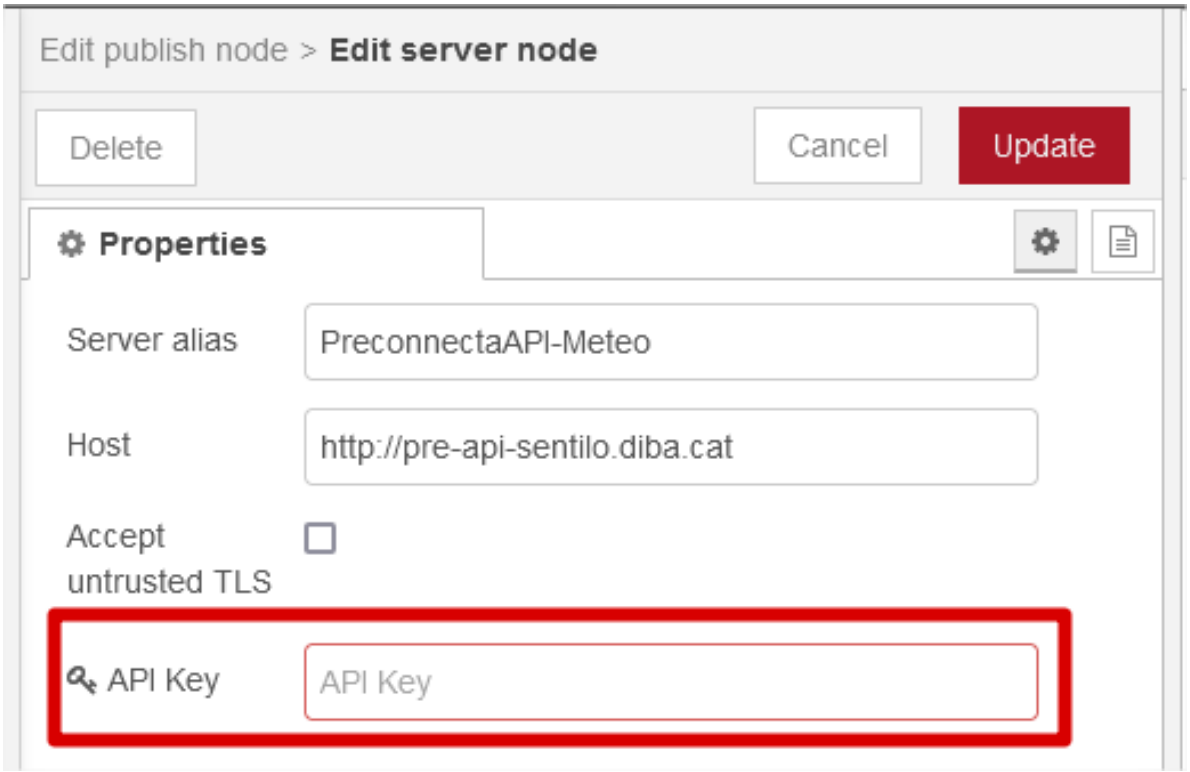
Com podem veure, el node *publish* està connectat a un node funcional que li prepara les dades a enviar (node *Prepare Sentilo msg*).

En arrencar la instància normalment detectarà que aquest node ha estat actualitzat, i el substituirà per la nova versió. En aquest cas no caldrà fer res més (si no fos així, només cal esborrar el node i tornar a insertar el node de la nova versió al seu lloc).

Per tant, ens trobem amb aquest escenari:

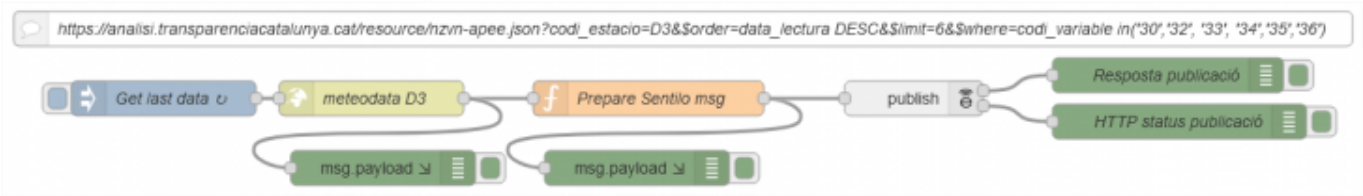
1. Arrenquem la nova instància i accedim al nostre fluxe
2. Revisem els nodes de Sentilo existents (en l'exemple, és el node *publish*) i mirem que hagin estat correctament actualitzats (verificar que el node no està en format transparent i no tenim cap error per pantalla, indicaria que cal substituir el node manualment)
    - si hi ha incompatibilitats, esborrem el node, i el tornem a inserir, aquesta vegada amb la versió actual (actualització del node de forma manual)
3. Cal tornar a configurar el node Server que estem fent servir per a les nostres connexions
    - és possible que Node-RED ens indiqui que hi ha errors de configuració al node (icona vermella o

missatge d'error que diu ***SETTINGS ERROR!***)
- cal revisar i inserir les dades de connexió a Sentilo, en especial el *token*, que per seguretat haurà estat esborrat del node *server*



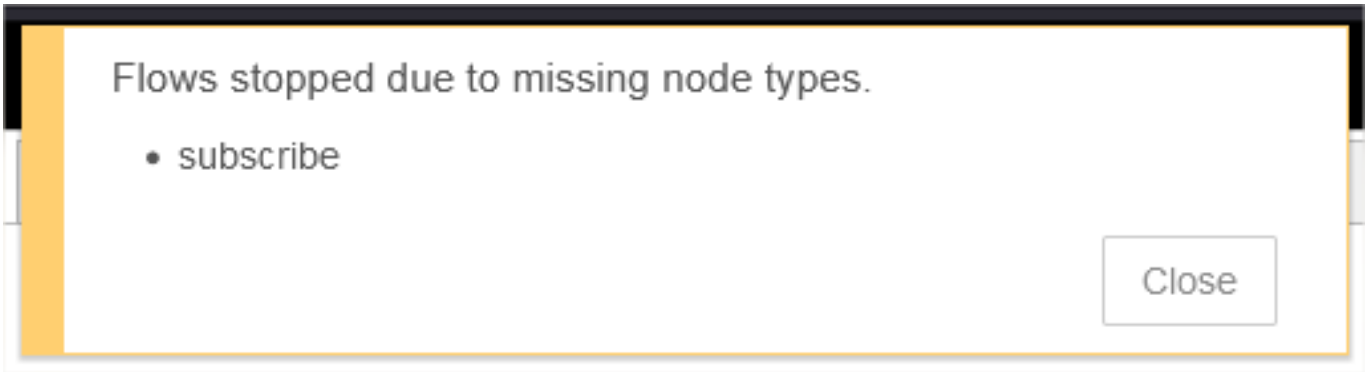
4. Un cop hem actualitzat el node, podem fer servir les sortides segons ens convingui
   - en l'exemple que estem explicant (veure imatge inferior) hem fet servir les sortides a mode de debug
     - sortida 1: debug de resposta de Sentilo
     - sortida 2: debug de codi d'estat http de Sentilo

Un cop finalitzat aquest procediment, ja hauríem de poder tornar a fer servir el nostre fluxe normalment:



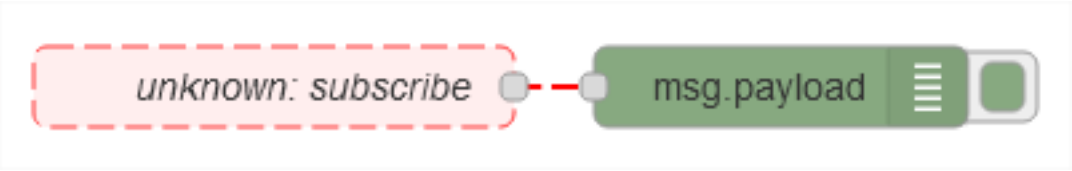
3.3.1. Casos d'incompatibilitat

Un cop arranquem i entrem per primer cop a la nova instància de Node-RED, és molt probable que ens trobem amb aquest missatge per pantalla (en el cas que algun dels nostres fluxos estiguin fent servir un node de subscripció):



És un *missatge totalment normal*, i ens indica que, en algun dels nostres fluxos, estem fent servir un node que ja no existeix.

Ens trobarem una situació com aquesta:

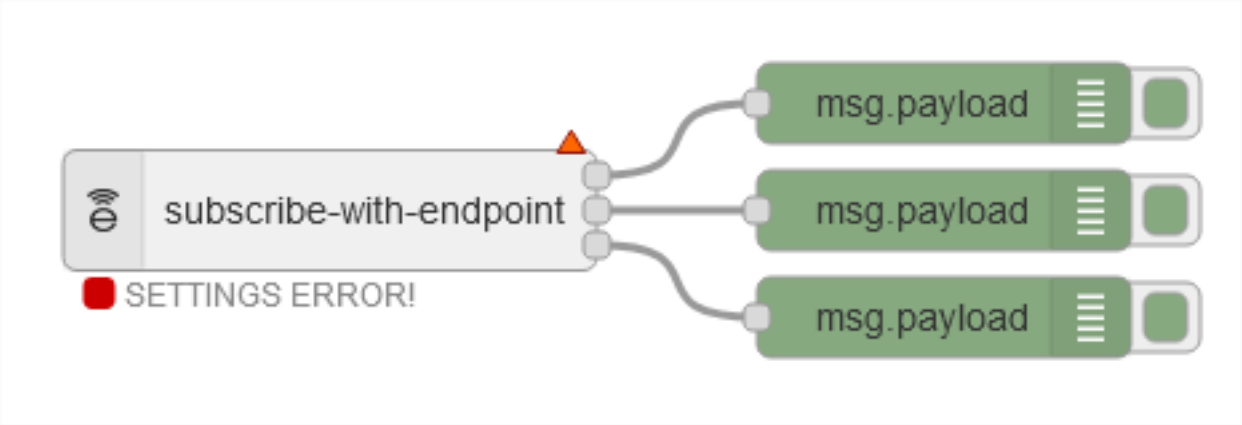


En aquest cas, és el node ***subscribe*** que no existeix. Com ja hem dit abans, ha canviat de nom i ara hi ha dos tipus de node de subscripció.

El que correspon a la versió anterior de forma directa seria el ***subscribe with enpoint***, que podríem substituir-lo sense més problemes, tot seguint les passes comentades anteriorment.

Per fer-ho, cal esborrar el node antic (marcat en vermell) i inserir el nou node, tot tornant a configurar les seves dades de connexió com s'ha explicat anteriorment.

Finalment, el fluxe quedaria així:



NOTA: les indicacions i errors ens avisen que cal configurar novament el node, ja que hem fet una substitució

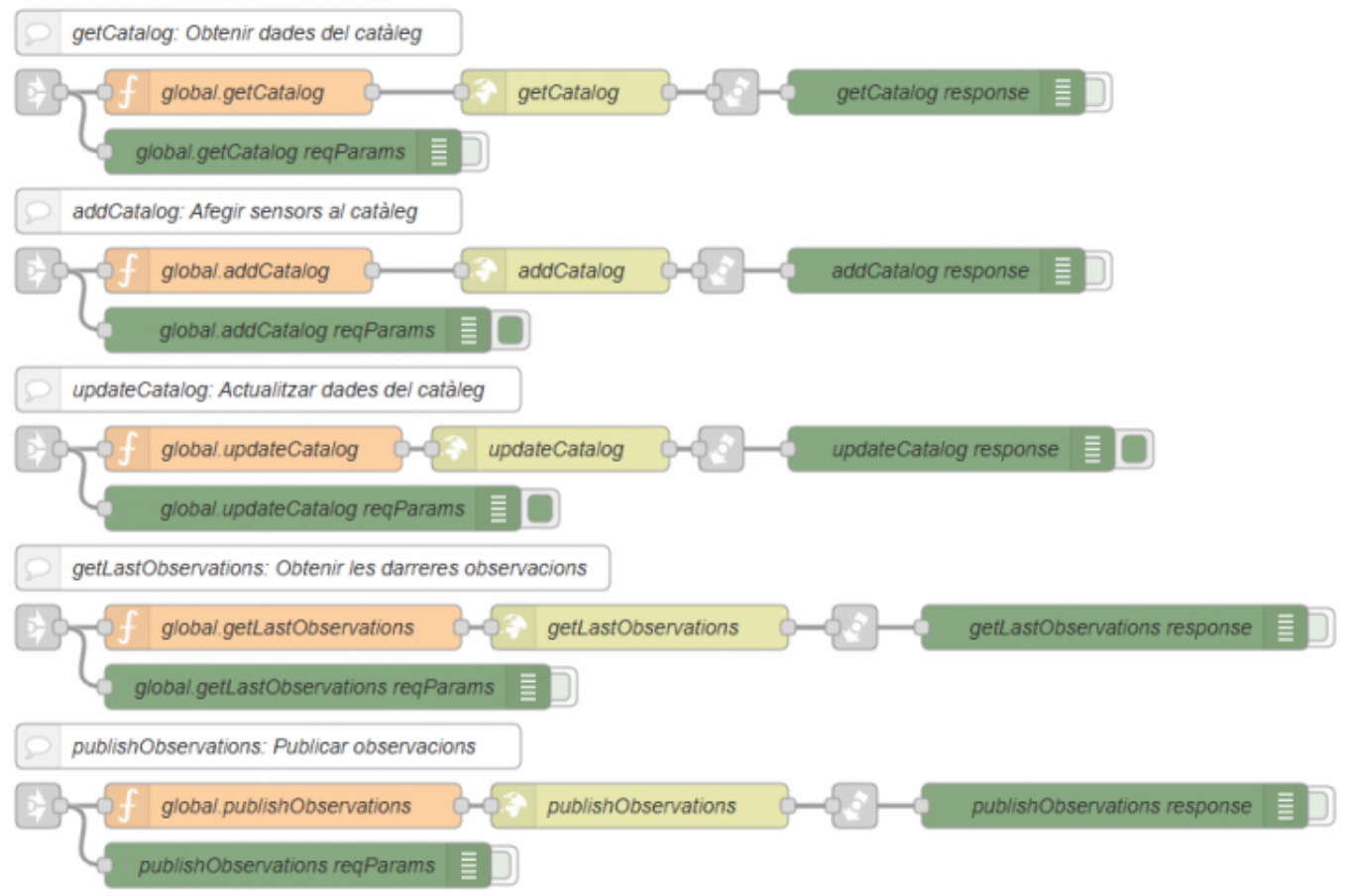
4. Recomanacions en el desenvolupament de fluxos personalitzats

4.1. Flux global

El flux global és una plantilla d'inici per a proveïdors que necessitin començar de forma ràpida i amb l'aplicació de bones pràctiques assolides durant el desenvolupament dels fluxes transversals.

Es composa de una petita col·lecció de **sub fluxos**, i un **flux principal**.

Aquests són els sub fluxos bàsics:



El flux global aporta funcionalitats transversals que es poden fer servir des d'altres fluxos de la instància Node-RED, mitjançant aquest sub fluxos.

El pas de paràmetres es fa a través de l'objecte msg.reqParams, en el que es configuren tots els valots necessaris per a poder fer les crides.

Podeu descarregar la col·lecció de sub fluxos o funcionalitats del flux global des d'aquest **link** [7].

4.1.1. Sub fluxos

4.1.1.1. getCatalog: Obtenir dades del catàleg

Obtenció de les dades del catàleg de Sentilo sobre els que el proveïdor identificat té permisos.

Segueix la documentació oficial de Sentilo: Retrieve providers / sensors list [8]

Invocació

Per a invocar-lo cal cridar la funcionalitat global.getCatalog des d'un node de crida a sub flux.

Dades d'entrada

```
msg.reqParams = {
    sentilo: {
        host: &lt;API_HOST&gt;,
        token: &lt;PROVIDER_TOKEN&gt;,
        filters: {
            type: &lt;SENSOR_TYPE&gt;,
```

```
            component: &lt;COMPONENT_ID&gt;,
            componentType: &lt;COMPONENT_TYPE&gt;
        }
    }
}
```

- host: (obligatori) adreça ip o nom de host de l'API de Sentilo
- token: (obligatori) credencials del proveïdor per fer la crida a l'API de Sentilo
- filtres: (opcionals)
    - type: tipus de sensor
    - component: identificador de component
    - componentType: tipus de component

Dades de sortida

El sub flux dona com a resultat el llistat de proveïdors i els seus sensors sobre els que té permisos, en el msg.payload.

4.1.1.2. addCatalog: Afegir sensors al catàleg

Creació de components i sensors al catàleg de Sentilo.

Segueix la documentació oficial de Sentilo: Adding sensors or components to the catalog [9]

Invocació

Per a invocar-lo cal cridar la funcionalitat global.addCatalog des d'un node de crida a sub flux.

Dades d'entrada

```
msg.reqParams = {
    sentilo: {
        host: <API_HOST>,
        token: <PROVIDER_TOKEN>,
        provider: <PROVIDER>,
        sensorsToCreate: [SENSORS_TO_CREATE]
    }
}
```

- host: (obligatori) adreça ip o nom de host de l'API de Sentilo
- token: (obligatori) credencials del proveïdor per fer la crida a l'API de Sentilo
- provider: (obligatori) proveïdor al qual aplica la crida
- sensorsToCreate: (obligatori) llistat de components i sensors a crear (el format ha de ser el mateix que s'exposa a la documentació oficial [10])

Dades de sortida

El sub flux no retorna cap dada, però en cas d'error podeu examinar l'objecte msg.payload per tal de verificar els motius.

4.1.1.3. updateCatalog: Actualitzar dades del catàleg

Actualització de components i sensors al catàleg de Sentilo.

Segueix la documentació oficial de Sentilo: Update data of a component / sensor [11]

Invocació

Per a invocar-lo cal cridar la funcionalitat global.updateCatalog des d'un node de crida a sub flux.

Dades d'entrada

```
msg.reqParams = {
    sentilo: {
        host: <API_HOST>,
        token: <PROVIDER_TOKEN>,
        provider: <PROVIDER>,
        componentsToUpdate: [COMPONENTS_TO_UPDATE],
        sensorsToUpdate: [SENSORS_TO_UPDATE],
    }
}
```

- host: (obligatori) adreça ip o nom de host de l'API de Sentilo
- token: (obligatori) credencials del proveïdor per fer la crida a l'API de Sentilo
- provider: (obligatori) proveïdor al qual aplica la crida
- componentsToUpdate: (opcional) llistat de sensors a actualitzar (el format és el que s'especifica a la documentació oficial de Sentilo [12])
- sensorsToUpdate: (opcional) llistat de sensors a actualitzar (el format és el que s'especifica a la documentació oficial de Sentilo [13])

Dades de sortida

El sub flux no retorna cap dada, però en cas d'error podeu examinar l'objecte msg.payload per tal de verificar els motius.

4.1.1.4. getLastObservations: Obtenir les darreres observacions

Obtenció de les darreres observacions del sensors del proveïdor de Sentilo.

Segueix la documentació oficial de Sentilo: Read observations from provider's sensors [14]

Invocació

Per a invocar-lo cal cridar la funcionalitat global.getLastObservations des d'un node de crida a sub flux.

Dades d'entrada

```
msg.reqParams = {
    sentilo: {
        host: <API_HOST>,
        token: <PROVIDER_TOKEN>,
        provider: <PROVIDER>,
        filters: {
            from: <FROM_DATE>,
            to: <TO_DATE>,
            limit: <LIMIT>
        }
    }
}
```

- host: (obligatori) adreça ip o nom de host de l'API de Sentilo
- token: (obligatori) credencials del proveïdor per fer la crida a l'API de Sentilo
- provider: (obligatori) proveïdor al qual aplica la crida
- filters: (opcional)

- from: data inici
- to: data fi
- limit: nombre d'observacions a recuperar (per defecte 1)

Dades de sortida

El sub flux retorna a l'objecte msg.payload totes les observacions del(s) sensor(s) sobre els que té permisos, i filtrat segons els filtres opcionals. El format de la sortida és l'especificat per la documentació oficial de Sentilo [15].

4.1.1.5. publishObservations: Publicar observacions

Publicació d'observacions dels sensors del proveïdor de Sentilo.

Segueix la documentació oficial de Sentilo: Publishing observations from different sensors [16]

Invocació

Per a invocar-lo cal cridar la funcionalitat global.publisghObservations des d'un node de crida a sub flux.

Dades d'entrada

```
msg.reqParams = {
    sentilo: {
        host: <API_HOST>,
        token: <PROVIDER_TOKEN>,
        provider: <PROVIDER>,
    }
}
```

- host: (obligatori) adreça ip o nom de host de l'API de Sentilo
- token: (obligatori) credencials del proveïdor per fer la crida a l'api de Sentilo
- provider: (obligatori) proveïdor al qual aplica la crida

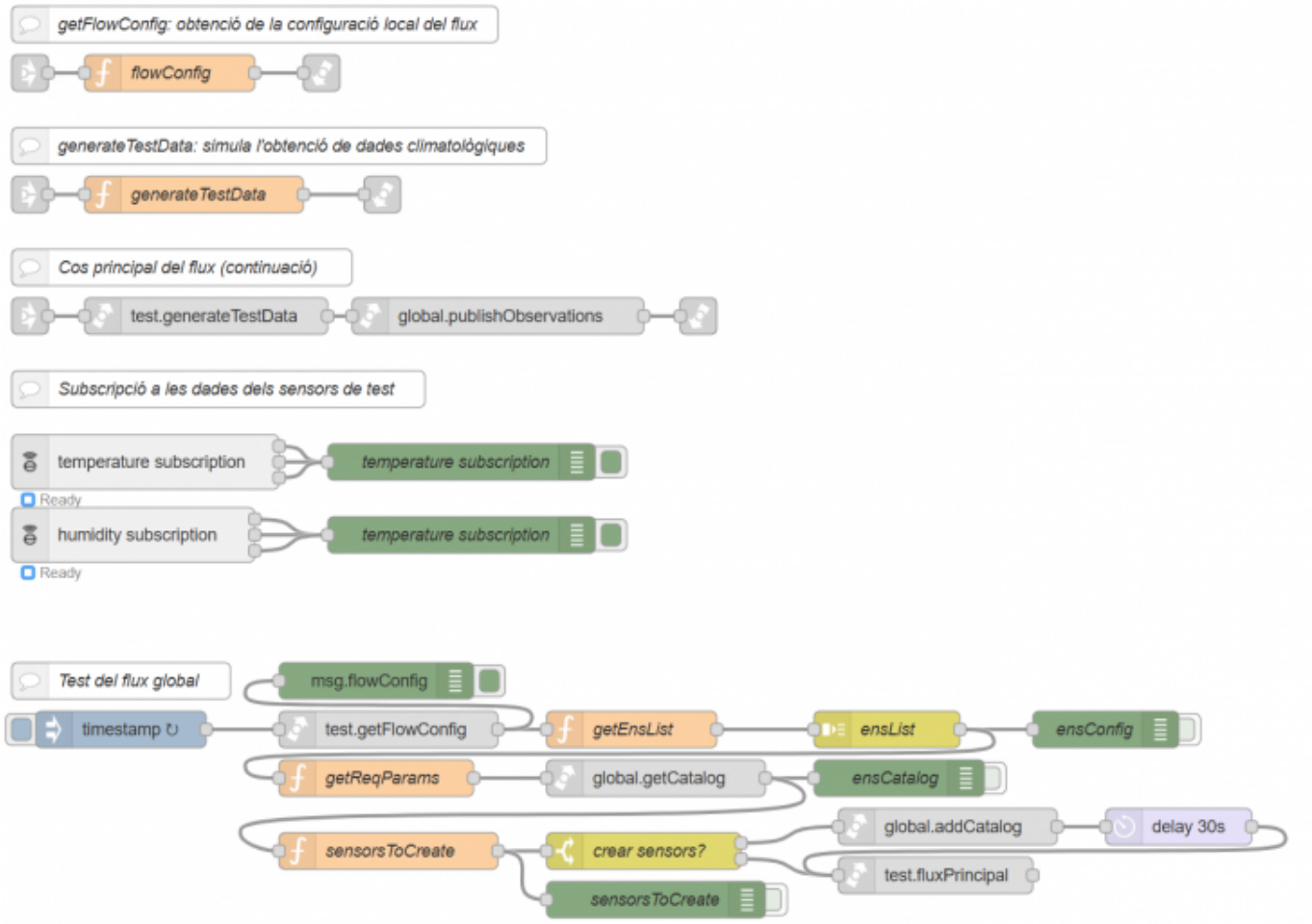
Dades de sortida

El sub flux no retorna cap dada, però en cas d'error podeu examinar l'objecte msg.payload per tal de verificar els motius.

4.1.2. Flux principal

Com part de la plantilla del flux global, s'ofereix a més aquest flux inicial que posa en pràctica la crida als sub fluxos globals, i exposa la metodologia de com podem treballar amb ells, tot fent servir les bones pràctiques i tècniques de desenvolupament propossades.

Aquest flux es pot fer servir com a plantilla inicial per a crear un flux de treball que aporta unes funcionalitats bàsiques: itera diferents ens i recupera i/o publica dades a Sentilo.



Aquest flux principal es pot importar tants cops com necessitem i modificar-lo per poder començar amb una base estable i fiable.

Podeu descarregar el flux des d'aquest **link** [17].

El pas de paràmetres es fa a través de l'objecte msg.reqParams, en el que es configuren tots els valots necessaris per a poder fer les crides.

Hi ha algunes parts ben definides:

- test.getFlowConfig: injecció i obtenció de la configuració bàsica del flux (retorna msg.flowConfig)
- test.generateTestData: simula l'obtenció de dades climatològiques (podria ser una crida a un servei extern, en el que obtenim dades de tercers, les tractem i les preparem per a ser publicades a Sentilo)
- test.fluxPrincipal: en aquesta part aniria tot la part core del flux que no és genèrica (continuació o crida a la part final del flux principal)

*Hi ha una part de subscripcions, que no formen part del flux, però que han estat incloses per a mostrar el funcionament dels nodes de subscripció de Sentilo. Es tracta d'*un petit exemple de com podem subscriure'ns a les dades d'un sensor i obtenir-les cada vegada que es publiquen dades a aquest sensor.

Finalment trobem el flux principal, que fa servir els *sub fluxos del flux global* i les seves funcionalitats *"privades"* del propi flux.

4.1.2.1. Funcionament del flux principal

Aquesta plantilla base ens permet:

1. Configurar les dades necessàries per fer-les servir al flux (injecció de msg.flowConfig)
2. Iteració per ens per a:
    - Obtenir del catàleg els sensors i components d'un proveïdor de l'ens
    - Crear els sensors i components que estan definits al fitxer de configuració però no existeixen al catàleg
3. Cridar / continuar amb el flux principal un cop hem generat tota l'estructura de components i sensors necessària a Sentilo (crida a test.fluxPrincipal)
    - Cridar a un sistema remot (simulació) per a obtenir les dades dels sensors
    - Publicar les dades dels sensors a Sentilo

4.1.2.2. Configurció del flux principal

A continuació es presenta el contingut bàsic de la configuració del flux de test, anomentat msg.flowConfig, que es fa servir durant tota la seva execució:

```
const flowConfig = {
    sentilo: {
        // Credencials de l'API
        api: {
            host: "<URL_API_SENTILO>",
            token: "<MASTER_TOKEN>"
        },
        catalog: {
            // Llistat d'ens per a iterar
            ens: [
                {
                    id: "<ID_ENS>",
                    provider: "<ID_ENS>@TEST_GLOBALS",
                    token: "<PROVIDER_TOKEN>"
                }
            ],
            sensors: [
                {
                    "sensor": "TESTG-TEMP",
                    "description": "Temperature sensor",
                    "type": "temperature",
                    "dataType": "number",
                    "unit": "ºC",
                    "component": "TESTG-METEO-001",
                    "publicAccess": "true",
                    "additionalInfo": {
                        "accuracy": "1.0%",
                        "voltage": "2.1-3.6"
                    }
                },
                {
                    "sensor": "TESTG-HUM",
                    "description": "Humidity sensor",
                    "type": "humidity",
                    "dataType": "number",
                    "unit": "%",
                    "component": "TESTG-METEO-001",
                    "publicAccess": "true",
                    "additionalInfo": {
                        "accuracy": "4.5%",
                        "voltage": "2.1-3.6"
                    }
                }
            ],
            components: {
                "TESTG-METEO-001": {
                    "component": "TESTG-METEO",
                    "componentType": "meteo",
                    "componentDesc": "Global flow meteo test component",
                    "componentPublicAccess": "true",
                    "location": "41.39479 2.148768",
```

```
                    "componentAdditionalInfo": {
                        "brand": "My brand",
                        "electrical-connection-type": "shuko",
                        "electrical-connection-voltage": "230V"
                    }
                }
            }
        }
    }
}

// Guardem l'objecte flowConfig al msg
msg.flowConfig = flowConfig;

return msg;
```

4.1.2.3. Anotacions finals

- El flux global és una plantilla d'iniciació, no és un flux funcional, sinó una petita recopilació de recursos que es poden fer servir de forma global des de qualsevor flux de la instància Node-RED
- Cada cop que importem aquesta plantilla és important canviar els noms de les funcionalitats internes (totes elles comencen per test.) per tal que puguin ser identificades com privades al propi flux
- Es recomana fer servir sempre l'objecte msg.reqParams com a mètode de pas de paràmetres com a bona pràctica
- El concepte de *token mestre* no és més que un punt en el que declarar un token que té permís sobre tots els proveïdors / ens de la configuració (pot ser una app amb els permisos necessaris, per exemple)
- Es recomana tenir una pestanya a la instància Node-RED amb els fluxos globals i anar creant diferents fluxos que els facin servir, i ampliar-los amb possibles necessitats, amb la idea de reutilitzar codi
- El flux de test és un seguit de bones pràctiques adoptades i propostes fetes per tal d'homoogenitzar la creació de fluxos Node-RED en DIBA, es recomana fer-lo servir com a base per a futurs fluxos que tinguin la mateixa base i funcionalitats (obtenció i actualització de catàleg de forma predeterminada, i posterior execució del flux principal, per exemple)
- Tots els fluxos i exemples contenen nodes de debug en punts d'interès, per poder observar dades d'entrada i sortida de les principals funcionalitats. Només cal activar-los o desactivar-los segons necessitats (en temps real).

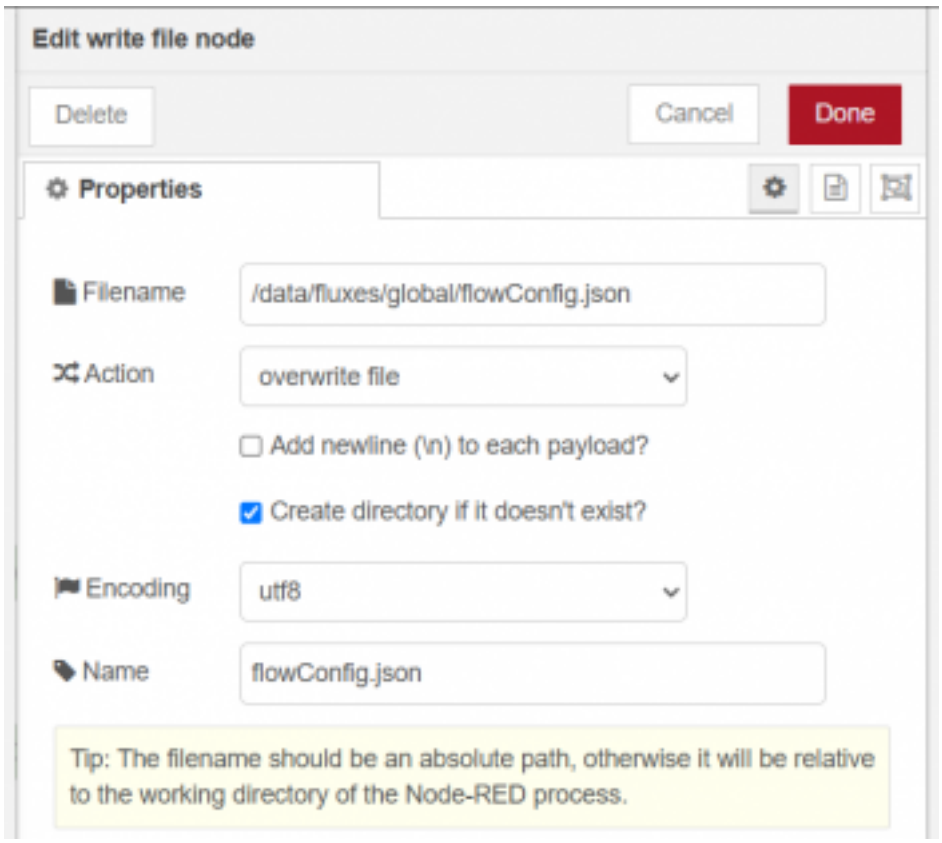
4.2. Persistència de dades a disc

Els fluxos tenen un espai de disc reservat per a poder escriure de forma segura. Aquest espai roman dintre del contexte de la instància Node-RED, i només podrem escriure les nostres dades dintre del mateix.

Aquest espai es troba al path absolut /data.

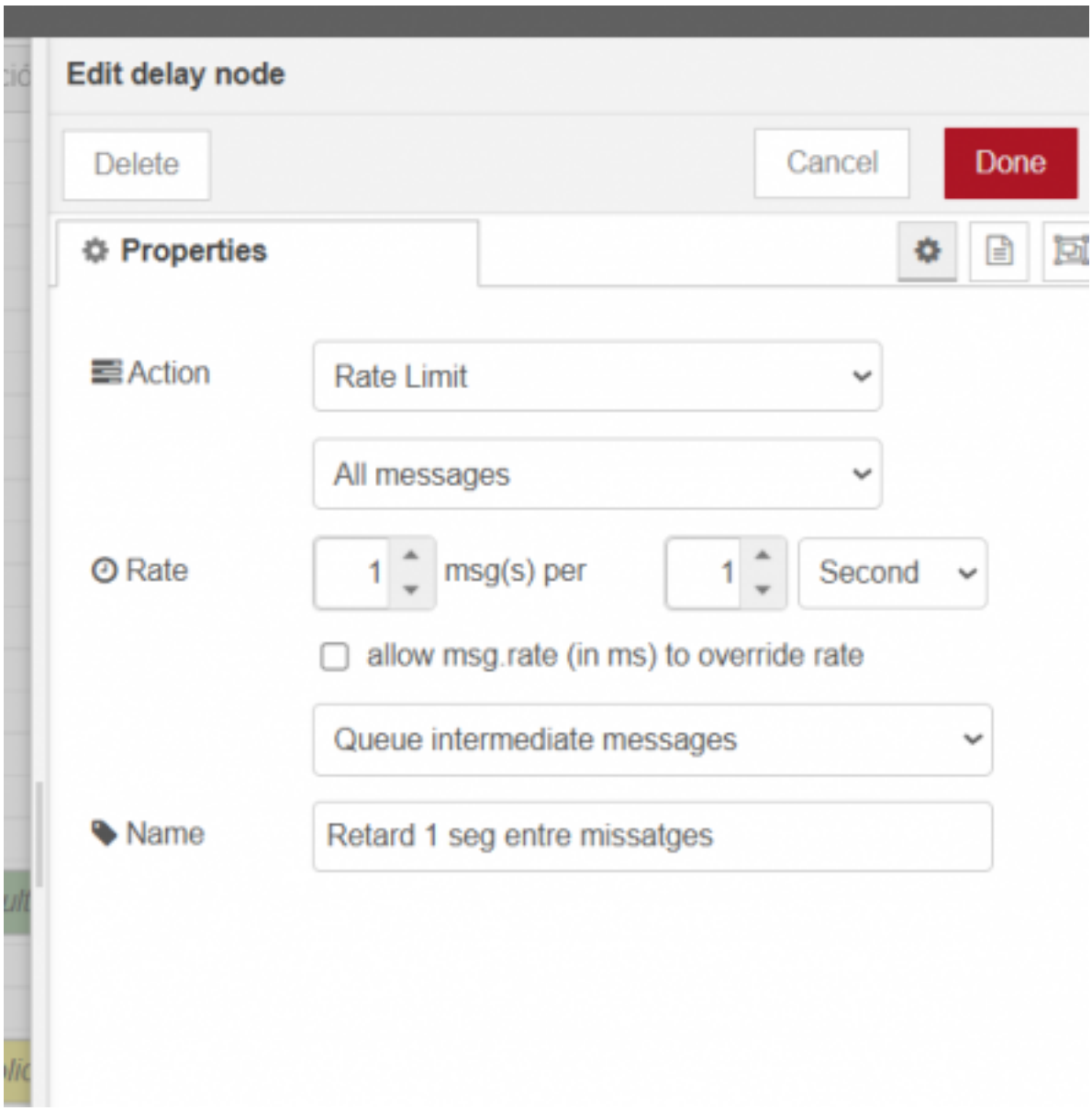
Es recomanda, doncs, fer servir sempre aquesta adreça com a base del path del fitxer que volguem escriure (o llegir), per tal d'accedir correctament i sense problemes al seu contingut.

Exemple d'escriptura a /data/fluxes/global/flowConfig.json



4.3. "Rate limiting" i timings recomenats

Per tal de no saturar els servidors, les peticions cal espaiar-les. Recomanem fer servir el node Delay node de Node-Red i configurar-lo d'aquesta manera abans de publicar a Sentilo:



Adjunt	Mida
Flux Meteocat [18]	6.45 KB
Flux Loxone [3]	5.37 KB
Flux Lorawan - sensors qualitat aire [1]	29.64 KB
Fluxos globals [19]	30.2 KB

URL d'origen: https://smartregion.diba.cat/wiki/5-fluxos-node-red

Enllaços:
[1] https://smartregion.diba.cat/sites/smartregion.diba.cat/files/sensors_de_qualitat_de_laire.json_.txt
[2] mailto:xoic@femprocomuns.coop
[3] https://smartregion.diba.cat/sites/smartregion.diba.cat/files/flux-loxone-sentilo-node-red.json_.txt
[4] mailto:smartregion@diba.cat?subject=Sol%C2%B7licitud%20integraci%C3%B3%20Meteocat
[5] mailto:smartregion@diba.cat?subject=Sol%C2%B7licitud%20integraci%C3%B3%20XPVCA
[6] mailto:smartregion@diba.cat?subject=Sol%C2%B7licitud%20integraci%C3%B3%20PRVE
[7] https://smartregion.diba.cat/sites/smartregion.diba.cat/files/userfiles/comunitat/nodered/flux-global.json_.txt
[8] https://sentilo.readthedocs.io/en/latest/api_docs/services/catalog/retrieve_authorized_entities.html
[9] https://sentilo.readthedocs.io/en/latest/api_docs/services/catalog/create_sensors.html
[10] https://sentilo.readthedocs.io/en/latest/api_docs/services/catalog/create_sensors.html#adding-several-sensors
[11] https://sentilo.readthedocs.io/en/latest/api_docs/services/catalog/update_sensors.html
[12] https://sentilo.readthedocs.io/en/latest/api_docs/services/catalog/update_sensors.html#request-to-update-the-component-data
[13] https://sentilo.readthedocs.io/en/latest/api_docs/services/catalog/update_sensors.html#request-to-update-the-

sensor-data
[14] https://sentilo.readthedocs.io/en/latest/api_docs/services/data/retrieve_provider_sensor_data.html
[15] https://smartregion.diba.cat/book/export/html/942
[16] https://sentilo.readthedocs.io/en/latest/api_docs/services/data/publish_provider_sensor_data.html
[17] https://smartregion.diba.cat/sites/smartregion.diba.cat/files/userfiles/comunitat/nodered/test-flux-global.json_.txt
[18] https://smartregion.diba.cat/sites/smartregion.diba.cat/files/meteocatsimple.json__0.txt
[19] https://smartregion.diba.cat/sites/smartregion.diba.cat/files/global-flows.json_.txt