Repte Blockchain i Aigua

PROPOSTA DE SOLUCIÓ

Blockchain 4SDG

Equip BCDS

Sergi Bergillos Pedraza

sergibergillospedraza@gmail.com

Marc Cosgaya Capel

u1959174@campus.udg.edu

Martí Madrenys Masferrer

u1953866@campus.udg.edu

13 de maig de 2022

Taula de continguts:

1 Introducció

1.1 Repte

En aquest segon repte organitzat pel CBCat, amb la col·laboració del Consorci d'Aigües Costa Brava Girona, es demana la integració d'una solució basada en tecnologia *blockchain* en el funcionament d'una planta depuradora pilot a Roses.

L'objectiu és garantir la integritat de les dades obtingudes dels sensors enfront d'agents externs maliciosos i així poder garantir que les decisions que han pres els encarregats de la planta han estat amb les dades correctes.

1.2 CBCat

El Centre de Blockchain de Catalunya (CBCat) té com a finalitat promoure l'adopció de tecnologies descentralitzades a Catalunya. Això s'aconsegueix mitjançant la difusió i l'alfabetització digital d'entitats i individus arreu del territori amb iniciatives i projectes com el projecte *Blockchain 4SDG*, en el qual forma part aquest repte.

1.3 Consorci d'Aigües

El Consorci d'Aigües Costa Brava Girona és una entitat que té per objectiu donar resposta a la problemàtica de la gestió dels recursos hidràulics de la Costa Brava i en la preservació de la qualitat de les seves aigües.

1.4 L'equip BCDS

L'equip BCDS està format per tres estudiants que col·laborem amb el grup de recerca Comunicacions i Sistemes Distribuïts (BCDS) de la Universitat de Girona: en Sergi Bergillos Pedraza, estudiant del màster de ciència de dades; en Marc Cosgaya Capel, estudiant de quart del grau d'enginyeria informàtica; i en Martí Madrenys Masferrer, estudiant també de quart del grau d'enginyeria informàtica.

Les dues branques principals d'investigació del grup són l'estudi de la robustesa de xarxes de telecomunicacions i l'aprenentatge enriquit amb la tecnologia, però més recentment s'ha participat amb projectes de transferència tecnològica de *blockchain*, com el desenvolupament d'una plataforma de vot electrònic amb éKratos, o amb projectes nacionals relacionats amb l'aigua, com CLEaN-TOUR amb ICRA.

2 Impacte en els Objectius de Desenvolupament Sostenible

Els Objectius de Desenvolupament Sostenible són 17 punts aprovats per l'ONU el 2015 que tenen per objectiu des d'eliminar la pobresa fins a combatre el canvi climàtic, l'educació, la igualtat de la dona o la defensa del medi ambient.

El projecte Blockchain 4SDG, mencionat breument en l'apartat anterior, consisteix en 17 reptes, un per cada Objectiu de Desenvolupament Sostenible. En concret, aquesta segona edició està lligat al punt **6. Clean Water and Sanitation**.

Encara que aquest punt pot semblar estrany d'aplicar al nostre país perquè a la descripció de l'objectiu, l'ONU menciona que una de cada tres persones no té accés a aigua potable salubre, dos de cada cinc persones no disposen d'una instal·lació bàsica per netejar-se les mans i més de 673 milions de persones defequen a l'aire lliure.

Tanmateix, la sequera és un fet habitual a la regió i, malauradament, s'espera el seu agreujament al llarg del segle XXI a causa del canvi climàtic. És per aquest motiu, que aquest punt ens resulta d'igual importància a Catalunya i el motiu pel qual des del Consorci d'Aigües Costa Brava Girona estan desenvolupant el projecte d'una planta pilot a Roses per recuperar aigua que amb els mètodes tradicionals no hi hauria cap altre remei que abocar al mar. Així, a més, es pot minimitzar el risc de restriccions d'aigua futures.

En concret, les tres metes en què s'engloba el repte són:

- 6.3 Per a 2030, millorar la qualitat de l'aigua reduint la contaminació, eliminant l'abocament i minimitzant l'alliberament de productes químics i materials perillosos, reduint a la meitat la proporció d'aigües residuals no tractades i augmentant substancialment el reciclatge i la reutilització segura a escala mundial.
- 6.4 Per a 2030, augmentar substancialment la utilització eficient dels recursos hídrics en tots els sectors, i assegurar la sostenibilitat de l'extracció i del subministrament d'aigua potable per a fer front a l'escassetat d'aigua i reduir substancialment el nombre de persones que sofreixen d'escassetat d'aigua.
- 6.6 Per a 2020, protegir i restablir els ecosistemes relacionats amb l'aigua, inclosos boscos, muntanyes, aiguamolls, rius, aqüífers i llacs.

Un segon objectiu que considerem rellevant, encara que en menor mesura, per aquest projecte és el **9. Industry, Innovation and Infrastructure**, ja que la nostra intenció és ajudar al Consorci d'Aigües Costa Brava Girona a desenvolupar una infraestructura de sanejament d'aigua fiable, sostenible, resilient i de qualitat en adoptar una nova tecnologia com és la *blockchain* i utilitzant metodologies de ciència de dades que facilitin l'optimització dels tractaments a realitzar.

3 Metodologia

La metodologia que hem escollit per a planificar i dissenyar la nostra proposta de solució és *The Data Science Hierarchy of Needs*, inspirada en la Jerarquia de Necessitats de Maslow. Aquesta, per exemple, és la metodologia que utilitzen a Som Energia per al monitoratge de les seves plantes fotovoltaiques.

Tal com es pot veure a la figura 1, la Jerarquia de les Necessitats defineix les diferents etapes que ha de seguir tot projecte de ciència de dades per ser assolit amb èxit. De forma resumida són:

1. Recollir: adquirir les dades dels instruments, sensors, registres,...
2. Moure i emmagatzemar: definir infraestructura, *pipelines*, bases de dades estructurades o no estructurades,...
3. Explorar i transformar: netejar les dades i detectar anomalies.
4. Agregar i etiquetar: realitzar un estudi estadístic de les dades, segmentar,...
5. Aprendre i optimitzar.
6. Utilitzar algorismes d'intel·ligència artificial i *machine learning*.

.

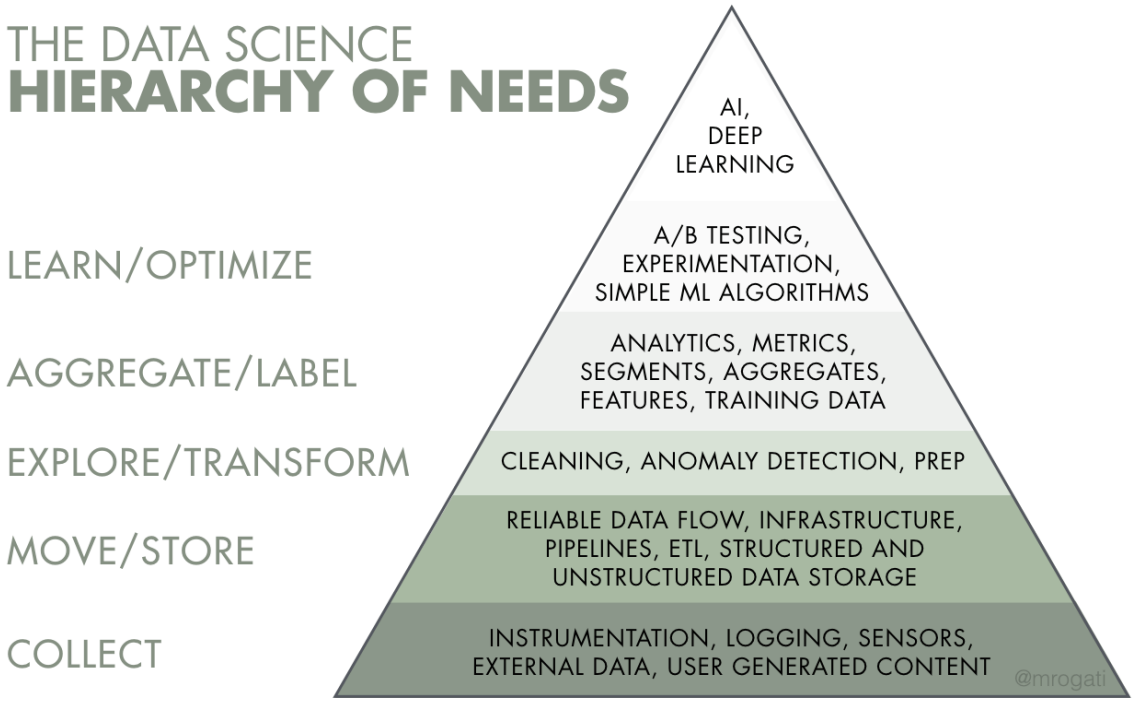

Figura 1: Diagrama de The Data Science Hiearchy of Needs per Monica Rogarti.

Encara que en aquest moment l'abast del projecte només arriba al nivell d'agregar i etiquetar, considerem que és una eina útil i necessària per a l'èxit de qualsevol classe de projectes en el que les dades juguen un paper fonamental perquè la seva adhesió des del principi permetrà després aprofitar aquesta feina ja feta per aprendre i optimitzar els processos de la planta.

4 Proposta

4.1 Objectius

Els nostres objectius són dos: aplicar la *blockchain* per a garantir la veracitat de les dades amb les quals es treballen i dissenyar una solució de *software* que segueixi totes les etapes de la metodologia explicada en la secció 3.

A més, a l'hora de desenvolupar la nostra solució, hem tingut en compte tres principis bàsics sobre els quals hem construït la proposta:

1. **Sostenibilitat**: La proposta haurà de tenir en compte la sostenibilitat prenent per base els ODS mencionats a la secció 2.
2. **Viabilitat**: La proposta haurà de ser viable econòmicament a llarg termini. Quan s'hagi de triar quina cadena fer servir, tindrem en compte els possibles costos en fer l'elecció.
3. **Seguretat**: La solució desenvolupada es basarà en els principis bàsics de seguretat informàtica sempre dins uns paràmetres d'acceptabilitat raonables.

Aquests principis són els que hem extret de les tutories realitzades durant la fase de desenvolupament.

4.2 Definicions

A continuació hem fet una llista de definicions de termes necessàries per entendre la nostra solució:

- *Gas*: Tarifa o preu que cal pagar per realitzar transaccions o executar funcions d'un *Smart Contract*. No totes les funcions dels *Smart Contract* s'han de pagar, només les que modifiquen l'estat de la *blockchain*.
- *Hash*: Una funció criptogràfica *hash* o *hash* és un algorisme matemàtic que transforma qualsevol bloc de dades en una nova sèrie de caràcters amb una longitud fixa.
- Microordinador: Ordinador basat en un microprocessador. En el context actual, i del nostre treball, es refereix a un ordinador petit.
- *Parachain*: Cadena de blocs paral·lela a la principal.
- *Plugin*: *Software* que ajuda a un programa a expandir les seves funcionalitats.
- *Proof of Work*: Protocol de consens distribuït en què els minadors competeixen per ser els primers a aconseguir resoldre un problema computacionalment complex. Per fer-ho cal usar força bruta, és a dir, prova i error. Això implica que les *blockchains* amb aquest algoritme tenen un impacte ecològic molt alt.
- *Proof of Stake*: Protocol de consens distribuït que funciona triant validadors en funció de la quantitat d'actius bloquejats. D'aquesta manera s'evita el cost computacional que tenen els protocols *Proof of Work*.
- *Pruning*: Tècnica de podar les dades perquè no ocupin tant.
- *Smart Contract*: Contracte intel·ligent. Defineix les crides que es poden fer a la *blockchain*.
- *Sudo*: Mòdul (o pallet) de Polkadot que proporciona les eines necessàries per a la gestió i manteniment de la *blockchain* a un administrador únic. Normalment, es fa servir en les primeres fases de desenvolupament on la gestió encara no es pot descentralitzar.

- *Switch*: Commutador de xarxa Ethernet.
- *Xip TPM*: El xip TPM (*Trusted Plataform Module*) és un processador especialitzat en operacions criptogràfiques que emmagatzema de forma segura les claus de xifratge.

4.3 Recollida de dades

A la recollida de dades cal diferenciar les dels sensors repartits per la planta i les dels tractaments que s'hi fa a l'aigua.

4.3.1 Dades dels sensors

Proposem connectar els sensors a microordinadors perquè aquests després facin les operacions a la *blockchain* i guardin les mesures preses a la base de dades.

Això permet la possibilitat de connectar múltiples sensors a un únic microordinador, depenent de la localització d'aquests dins la planta, per què un microordinador té capacitat més que suficient per gestionar diferents sensors alhora. Això també permet reduir costos d'instal·lació i manteniment.

La connexió entre els sensors i els microordinadors es farà mitjançant un bus i utilitzant el protocol estàndard Modbus per simplificar la configuració dels microordinadors.

També és interessant la possibilitat de fer servir xips TPM per emmagatzemar de forma segura les claus privades que els microordinadors faran servir per signar les transaccions a la *blockchain*.

Encara que hi hagi lectures errònies a causa d'errors del *hardware*, preferim no corregir-les prèviament. Volem insistir que hem de garantir la integritat de les dades i poder detectar aquests errors a posteriori. Per això, un cop publicades les dades a la *blockchain* veiem possible una operació de postprocessament per eliminar els possibles errors de la lectura.

Una assumpció que hem fet és que la connexió entre els sensors i els microordinadors és segura per què des del Consorci D'Aigües Costa Brava Girona poden garantir la seguretat física de la planta pilot i, per tant, no considerem altres tècniques per evitar la substitució d'un sensor per un altre que doni dades errònies.

4.3.2 Dades dels tractaments

Proposem entrar els tractaments al sistema mitjançant una interfície web. És a dir, un operari s'identificarà a la pàgina web i entrarà la informació referent a un tractament (qui, què, quan i perquè) per més tard poder fer l'associació entre les dades dels sensors i les característiques del tractament.

En tractar-se d'una pàgina web amb poca càrrega computacional creiem que un altre microordinador pot ser suficient per fer l'hosteig.

4.4 Emmagatzematge de les dades

En un principi la nostra intenció era publicar totes les dades recollides a la *blockchain*, però ràpidament ens vam adonar que això no era viable per què les *blockchains* no estan pensades per treballar amb un volum tan gran de dades.

Així, després d'investigar les diferents aproximacions possibles, vam veure que la solució més habitual a aquest problema és fer servir la tecnologia *blockchain* de la següent forma: per una

banda, es guarden totes les dades en brut a una base de dades; i, per altra banda, per garantir la fiabilitat de les dades, cada cert període de temps es calcula el *hash*, definit a l'apartat 4.2, de les dades i es penja a la *blockchain*.

Fer-ho d'aquesta manera ens aporta diversos avantatges:

- Com que les funcions de *hash* generen sempre una cadena de caràcters de mida fixe, aconseguim reduir la mida de les transaccions que s'ha de publicar a la *blockchain* i d'aquesta forma també reduïm el cost per transacció, tant energètic com econòmic.
- Una altra de les propietats dels algoritmes de *hash* és que un petit canvi en les dades d'entrada provoca un gran canvi en les dades de sortida, per tant, si algun actor maliciós manipulés les dades per poc que sigui, podrem detectar-ho sempre i actuar en conseqüència.

Això també permetrà que qualsevol entitat o persona pugui validar la fiabilitat del sistema en comprovar que el *hash* generat a partir de les dades publicades a la base de dades.

A més a més, en un sistema de monitoratge d'aquest tipus només ens interessen les dades més recents així que per estalviar en capacitat de la base de dades recomanem implementar un programa en Python que s'encarregui del manteniment. És a dir, aquelles dades que, per exemple, siguin més velles d'un any, es poden comprimir i emmagatzemar en disc.

Finalment, nosaltres proposem fer servir TimescaleDB perquè és una base de dades codi obert especialitzada en sèries temporals.

4.4.1 Càlcul del hash

El càlcul del *hash* hem decidit fer-lo el més a prop possible dels sensors per raons de seguretat. Així, cada microordinador serà l'encarregat de calcular el *hash* per cada sensor i cada interval de temps definit. A més, aquests microordinadors també seran els encarregats de signar les transaccions de la *blockchain* pel que cadascú tindrà la seva pròpia parella de claus criptogràfiques que, per millorar la seguretat, s'emmagatzemaran en xips TPM.

Una opció que creiem també interessant és la possibilitat que un operari o l'administrador del sistema pugui forçar el càlcul del *hash* i la seva publicació si detecta, per exemple, valors molt estranys d'un sensor i vol assegurar-se que aquests han estat generats pel sistema.

L'objectiu d'utilitzar una funció *hash* és ajuntar múltiples mesures en una sola cadena de caràcters de mida reduïda sense perdre la capacitat de detectar possibles modificacions. Això també permet afegir i treure sensors sense afectar els *hashos* existents. En el cas dels tractaments, ja que creiem que la seva freqüència serà molt menor que de les mesures, hem decidit que es faci un *hash* per a cada tractament.

Un altre avantatge d'aquesta proposta és la possibilitat d'assignar un interval de temps diferent per penjar el *hash* de cada sensor segons les seves necessitats. Caldrà, doncs, tenir en compte que a major freqüència tindrem una finestra més petita de treballar amb dades manipulades a canvi d'un cost més elevat. En canvi, a menor freqüència tindrem un cost més reduït a canvi de perdre integritat de les dades. A l'estudi del pressupost analitzem la viabilitat de les diferents freqüències.

Per al càlcul d'aquest *hash* hem fet servir l'algorisme *SHA-256* després de concatenar totes les mesures de l'interval de temps del sensor de la següent manera:

valormesura1_datamesura1#...#valormesuraN_datamesuraN#idsensor#idinterval

Per als tractaments, proposem fer la concatenació de la següent manera executant el mateix algorisme:

tractament#datatractament#idtreballador

4.4.2 Node Repetidor

Per raons de seguretat, hem decidit que els microordinadors no tinguin accés directe a Internet sinó que ho facin a través d'un node repetidor que formarà part de la xarxa *blockchain* escollida.

Aquest actuarà també com a *firewall* dels microordinadors i així afegirà una capa extra de seguretat al sistema. D'aquesta manera podem habilitar només els ports imprescindibles per al correcte funcionament dels microordinadors.

Els microordinadors es connectaran al node a través d'un *switch*, que a la vegada estarà connectat al node. Aquest pot ser un altre microordinador. També hem decidit que la web per introduir els tractaments pugui estar executant-se dins d'aquest node.

4.5 Elecció de la blockchain

4.5.1 Candidats de la Blockchain

Les *blockchains* candidates que hem tingut en compte han estat:

- BigChainDB: Base de dades distribuïda basada en *blockchain*, immutable. Treballa amb *Proof of Stake*.
- Cardano: *Blockchain* de 3^a generació. S'hi poden executar Smart Contracts, però han de ser desenvolupats en llenguatges específics.
- Ethereum: *Blockchain* de 2^a generació. Actualment, funcionen amb un algoritme de consens basat en *Proof of Work* tot i que s'està treballant en una nova versió basada en un mètode *Proof of Stake*.
- Moonbeam: *Parachain* de Polkadot que permet implementar *Smart Contracts* amb Solidity.
- Symbol: *Blockchain* dissenyada per desenvolupar solucions per a empreses i organitzacions. Basada en *Proof of Stake*.
- Teranyina: *Blockchain* basada en Polkadot, creada pel CBCat i pretén ser una tecnologia de caràcter públic per tal que tot aquell qui vulgui la pugui fer servir. Buscarà incentivar la investigació i recerca. L'algoritme de consens és *Sudo* tot i que s'espera que acabi en un *Proof of Stake*.

4.5.2 Consideracions al triar Blockchain

A l'hora de triar quina *blockchain* hem fet servir per publicar les dades, hem tingut en compte les següents consideracions que es deriven dels principis mencionats a l'apartat 4.1:

- Impacte ecològic que té la *blockchain* sobretot pel que respecta al cost energètic.
- El cost econòmic de desenvolupar la solució amb cada candidat tenint en compte el volum de dades i transaccions que s'hauran de fer.

Finalment, també es tindran en compte altres conceptes estratègics com el contacte amb els desenvolupadors, facilitat de la documentació i desenvolupament.

4.5.3 Elecció del candidat definitiu

En un començament es descarta la cadena Ethereum pel gran cost energètic i el consegüent impacte ambiental que suposa.

Per entendre millor aquest impacte i la diferència entre una *blockchain* amb *proof of work* respecte a una amb *proof of stake*, podem veure quin cost energètic té fer una transacció amb alguns dels candidats, i amb Visa per comparar, a la taula 1.

Blockchain	**Cost per transacció**
Ethereum	692.820.000 J
Bitcoin	6.995.592.000 J
Cardano	44.604 J
Visa	12.888 J
Moonbeam	13.608 J

Taula 1: Cost energètic per transacció de diferents *blockchains*.

Aquesta taula ha estat calculada a partir de les dades de les següents fonts: [Platt et al., 2021], [SolanaDevelopers, 2021], [DigiEconomist, 2022].

Una segona eliminació és BigChainDB per què no s'ajusta a la solució que volem desenvolupar. Busquem una plataforma d'*Smart Contracts* i BigChainDB s'adapta més al concepte de base de dades.

A més a més, també descartem Symbol perquè no permet desenvolupar fàcilment els *Smart Contracts*, hauríem d'implementar la funcionalitat a partir de *plugins* i treballar amb una *blockchain* híbrida entre privada i pública.

La següent *blockchain* que hem eliminat ha estat Cardano perquè al moment de redactar aquest document, el preu de les transaccions és més elevat respecte Moonbeam, tal com es pot veure a la taula 2 on també hi ha Ethereum per comparar. Un segon motiu és que la programació dels *Smart Contracts* a Cardano es fa amb Haskell. Haskell és un llenguatge de programació funcional segur, però amb una corba d'aprenentatge lenta.

Blockchain	**Preu gas en *token***	**Gas requerit**	**Gas * Preu Gas**	**Conversió a €**
Ethereum	0,00000007227	27.103	0,00195873381	5,1504513787188
Cardano	X	27.103	0,156789	0,156789
Moonbeam	0,000000101	27.103	0,002737403	0,00802059079

Taula 2: Cost econòmic per transacció de diferents *blockchains*.

Les dades que s'han utilitzat per a l'elaboració de la taula 2 han estat:

- El cost que té (gas) penjar un *hash* a la *blockchain*: 27.103 gas. Això depèn bàsicament del cost computacional que té executar la funció que guarda aquest *hash* al contracte.
- La mitjana del preu del gas a Ethereum del 8 de febrer al 8 de març de 2022: 72,27 gwei.
- La conversió mitjana d'Ethereum a euros el mes de febrer: 2.629,48€.

- El preu del gas a Moonbeam el 9/03/2022: 102. No disposem d'històric.
- La conversió mitjana de GMLR (*token* natiu de Moonbeam) a euros el gener de 2022: 3,78€.
- Per calcular el cost de la transacció amb Cardano s'ha fet servir la fórmula de cost mínim documentada a la web dels desenvolupadors: $0,155381$ ADA + $0,000043946$ ADA/Byte $*$ 32 Bytes.
- La conversió mitjana d'ADA a euros el gener de 2022: 1,04€.

Quedant dos candidats i donat que tecnològicament són força semblants perquè tots dos permeten treballar amb Solidity, hem decidit que treballarem amb Teranyina perquè tenim contacte directe amb els desenvolupadors i per donar suport a una xarxa creada al territori.

Tanmateix, es pot canviar entre les dues *blockchain* de manera molt senzilla, només cal canviar un fragment d'un arxiu de text. Això pot resultar útil per si Teranyina no està completament operativa una vegada s'hagi d'implementar el projecte.

4.5.4 Smart Contract

Una de les principals característiques que hem tingut a l'hora d'escollir la *blockchain*, a part del seu cost energètic i econòmic, és la possibilitat de realitzar *smart contracts*. La Teranyina ens ofereix aquesta opció al poder treballar amb Solidity per definir el contracte que ens permet interactuar amb la *blockchain* per publicar les dades i validar-las.

Aquest contracte té les següents funcions:

- *authorize(address)*: el creador del contracte autoritzar un actor, identificat per la seva adreça IP o nom DNS, a publicar dades a la *blockchain*.
- *revoke(address)*: el creador revoca el permís d'escriptura d'un actor autoritzat prèviament, identificat per l'adreça.
- *checkAutorized(address)*: funció de consulta que retorna cert si l'actor, identificat per l'adreça, està autoritzat a publicar dades a la *blockchain*, altrament retorna fals.
- *add(bytes32)*: un actor autoritzat publica una llista de 32 bytes (256 bits) i la guarda a la *blockchain*. Aquests 32 bytes corresponen al *hash* explicat a 4.4.1.
- *validate(address,bytes32)*: funció de consulta que retorna cert si l'actor amb l'adreça especificada ha penjat la llista de 32 bytes, altrament retorna fals. Aquesta funció permet validar si un microordinador ha penjat un *hash* concret i la pot fer qualsevol actor, autoritzat o no.

El funcionament habitual de l'*Smart Contract* implementat seria el següent:

1. El gestor del sistema autoritza un microordinador a publicar dades a la *blockchain* (en el nostre cas l'únic actor autoritzat seria el node repetidor explicat a 4.4.2) amb la funció *authorize(addrMicroordinador)*.
2. El microordinador autoritzat penja els *hashos* que ha calculat per cada sensor i per cada interval de temps amb la funció *add(hashComputat)*.
3. El Consorci d'Aigües Costa Brava Girona, l'administració pública, un grup ecologista o qualsevol persona que ho desitgi, pot validar que el sistema funciona correctament

comparant les dades que hi ha a la base de dades amb les de la *blockchain* amb *validate(addrMicroordinador, hashQueEsVolValidar)*. Aquesta consulta és gratuïta pel fet que no genera cap cost. Tanmateix, això requerirà que es posi a disposició del públic tant les adreces autoritzades com les seves claus públiques.

4. Si pel motiu que sigui s'ha compromès la seguretat d'una clau privada d'un microordinador caldria revocar el permís de penjar dades, ho hauria de fer el gestor del sistema amb *revoke(addrMicroordinadorCompromesa)*.

4.6 Processament de les dades

El processament de les dades és una etapa indispensable pel bon funcionament d'un projecte de ciència de dades, ja que ens permet transformar-les per adaptar-les a les nostres necessitats.

Tanmateix, aquesta etapa està molt lligada amb els requeriments i les funcionalitats que els experts o els consumidors finals, en aquest cas els treballadors del Consorci d'Aigües, necessiten. Una vegada definits els requeriments, ens podem fer 5 preguntes que ens ajudaran a definir el què hem de fer en aquesta etapa:

1. Quina és la font?
2. Quina ha de ser la freqüència de mesura i la freqüència de visualització?
3. Quina agregació s'ha de fer a aquestes dades?
4. Quina és la fórmula de l'indicador?
5. Quin tipus de visualització volen?

Una vegada ja se sap això, es pot fer el processament de les dades en brut que obtenim dels sensors. Un punt molt important a tenir en compte en aquest moment és que les dades en brut no s'han de modificar o eliminar, sinó que aquestes transformacions s'han d'emmagatzemar en noves taules, perquè això provocaria que els *hashos* que hem publicat a la *blockchain* no concordin amb els de la base de dades.

Algunes de les transformacions habituals que es fan en el processament de les dades són:

- Generar taules derivades amb arrodoniments o interpolació de valors per emplenar els forats buits dels sensors o simplement utilitzar la mitjana d'altres sensors del mateix tipus per aproximar aquells valors que no es tenen.
- Una segona tècnica és realitzar agregacions de les dades cada X temps: per exemple una mitjana.

4.7 Lectura de dades

4.7.1 Visualització

Per visualitzar l'estat del sistema cal consultar la base de dades per obtenir en temps real l'estat del sistema. Algunes de les dades encara no poden ser validades perquè el seu corresponent *hash* encara no s'ha penjat.

Proposem fer servir Redash perquè permet fàcilment connectar una base de dades i visualitzar les dades en gràfics. La plataforma permet afegir gràfics modularment i en general és molt flexible.

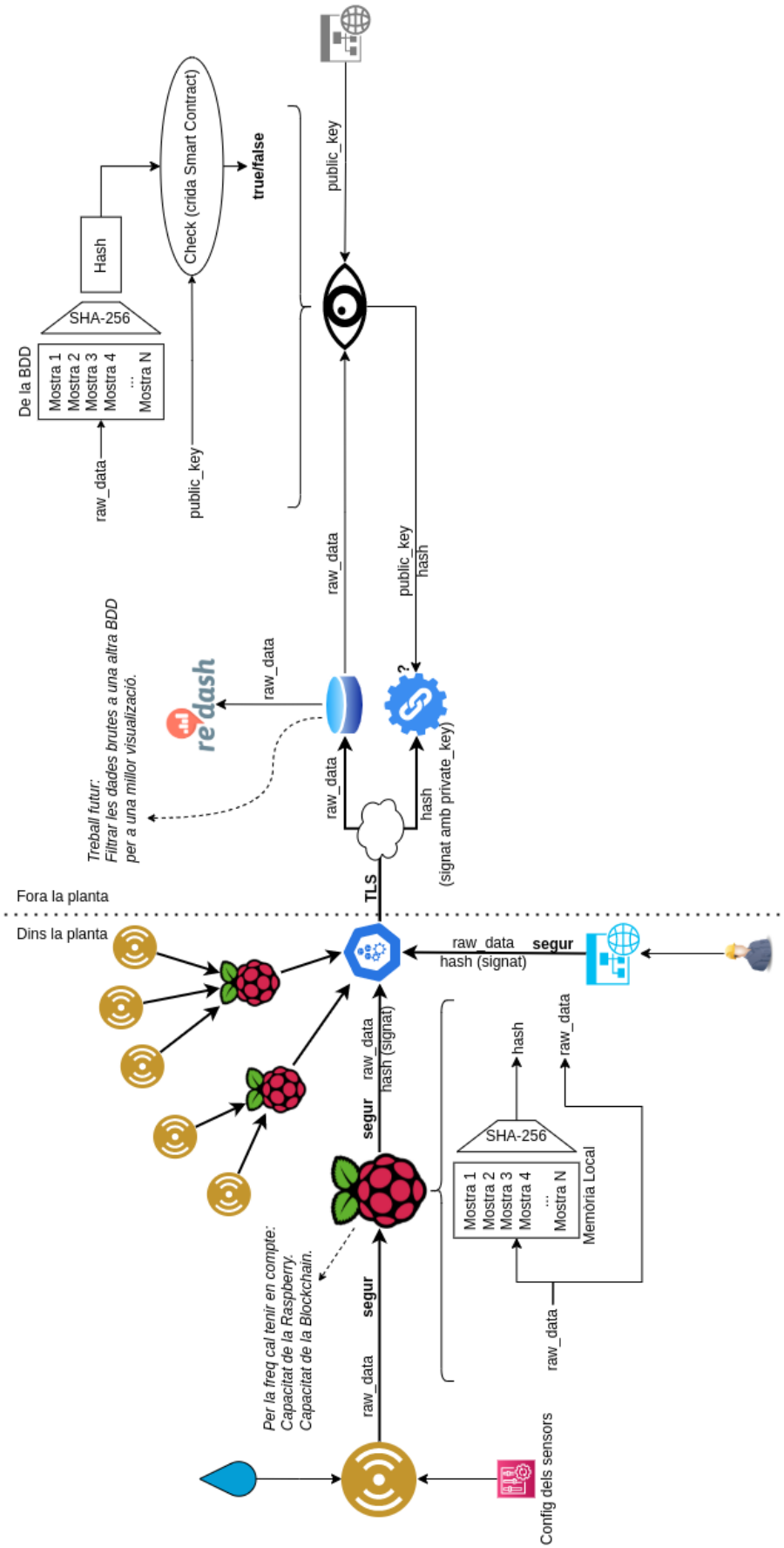

Figura 2: Digrama complet de la nostra proposta.

4.7.2 Validació

Per validar les dades que s'han penjat a la base de dades, una entitat externa ha de poder tenir accés a la lectura d'aquestes. És a dir, que la lectura de la base de dades sigui pública. D'aquesta manera l'entitat pot llegir-ne el contingut i calcular el *hash* de la mateixa manera que en la publicació. Finalment, pot executar la crida al *smart contract* per saber si el *hash* existeix i, per tant, que les dades corresponents estan validades.

4.8 Diagrama complet

La figura 2 és el diagrama general de la nostra proposta. Hem escollit utilitzar Raspberries com a microordinadors. Veiem que és una bona opció perquè són flexibles per agregar múltiples sensors i de molts tipus diferents, inclosos els *Scann*. També són barates, uns 50€la versió de gama mitjana. I també consumeixen relativament poc, 3W en repòs i uns 6W quan treballa en alt rendiment.

La informació té dos recorreguts o capes. La primera és la de les dades i la segona és la de la verificació. Aquests dos recorreguts succeeixen a la vegada.

Dins la capa de les dades, els sensors primer recullen les dades i després els microordinadors les envien a la base de dades. Posteriorment les dades poden ser processades per netejar i arreglar possibles errors en les mesures per ser guardades en una base de dades a part. Finalment, es poden visualitzar els valors de les mesures mitjançant un programa de visualització. En el nostre cas hem escollit Redash tal com ja hem mencionat abans.

Dins la capa de la verificació, els sensors primer recullen les dades i després els microordinadors en calculen el *hash* i signen les transaccions. El node repetidor s'encarrega de comunicar-ho a la resta de la xarxa *blokchain*. Finalment, una entitat externa pot validar les dades mitjançant un validador, un programa especialitzat, o a través de les seves pròpies eines.

4.9 Pressupost

4.9.1 Pressupost inicial

Per a la posada en marxa de la proposta hem calculat el pressupost inicial. No tenim en compte el preu d'enviament del *hardware*.

- Les Raspberries: hem estimat que necessitarem 11, 10 per a la recollida dels sensors i 1 amb accés a internet per a la web dels tractaments i node local de la *blockchain*. Aquesta estimació l'hem fet partint del document que se'ns va fer arribar en una tutoria on es projectaven 64 sensors en les diferents fases de la planta, la distribució i clusterització de les Raspberries segons la fase és la següent:
  - Dipòsit de capçalera: 4 sensors 1 Raspberry.
  - Microfiltració: 3 sensors 1 Raspberry.
  - Osmosi inversa: 43 sensors 6 Raspberries.
  - UV-AOP: 6 sensors 1 Raspberry
  - Dipòsit sortida: 6 sensors 1 Raspberry
  - Gestió (web, node de la *blockchain* i servidor de reenviament): 1 Raspberry

Com que el preu per cadascuna és de 51,01€[1] tenim un cost total de 561,11€.

No obstant això, a l'haver desenvolupat la solució usant un sistema modular, si el nombre de sensors acabés essent diferent del projectat, la infraestructura que proposem s'hi podria adaptar molt fàcilment afegint o eliminant el nombre de microcomputadors.

- Les targetes SD dels microordinadors: per a les Raspberries de les mesures hem escollit targetes de 64 GB i Categoria 10 (120 MB/s). Per al node amb accés a l'exterior hem decidit fer servir una de 256 GB i Categoria 10, ja que estimem que es requerirà més espai. Les de 64 GB tenen un preu de 13,50€[2] i la de 256 GB té un preu de 48,31€. Per tant, el cost total és de 183,31€.
- Els xips TPM per a les Raspberry Pi 4, aquests xips són integrables al microordinador Raspberry i ens permeten guardar de manera segura les claus privades. El seu cost unitari és de 25,54€[3] i, per tant, el cost total és de 280,94€.
- Fonts d'alimentació de les Raspberries: En aquest cas es tracta de fonts de 15 W que tenen un preu de 6,82€[4] i, per tant, el cost total és de 75,02€.
- Cablejat Ethernet: Per a la connexió de les Raspberries amb el *switch* i el node local es necessitarà cable Ethernet. El preu per a una bobina de 10 0m és de 91,55€[5] i estimem que en necessitarem 2. Per tant, el cost total és de 183,10€.
- *Switch* Ethernet: Per poder crear una xarxa local per a les Raspberries de manera segura, pressupostem un *switch* de 16 ports que es connectaria a través d'un tallafoc amb el node de Teranyina. Les Raspberries no estaran en cap cas connectades directament a Internet.
- Mà d'obra per a la instal·lació del *hardware*: Estimem que es necessitaran 20 hores per a la instal·lació del *hardware*. Hem estimat que el preu serà de 17,50€/h tenim un cost total de 350,00€.
- Mà d'obra per a la instal·lació i la configuració del *software*: Estimem que es necessitaran 270 h de mà d'obra. Assumim 100 h més de marge pels possibles problemes que pudin sorgir a l'adaptar el codi que ja s'ha implementat. En total 370h de mà d'obra a 20 €/h desglossat de la següent forma:
  - 50 h per la posada en marxa del Redash i implementació de les consultes SQL i dels *dashboard*, configuració del servidor de correu electrònic per les alertes, configuració del HTTPS.
  - 100 h per desenvolupar el software que processi les dades d'acord amb el que s'ha definit a la subsecció 4.6.
  - 30 h per fer la web amb *AJAX* partint de la prova de concepte i implementant la gestió d'usuaris que no s'ha fet.
  - 90 h per instal·lar i configurar les Raspberris: sistema operatiu, sensors i xarxa.

El cost total de posada en producció de la nostra solució seria de: 10.905,91€.

[1] `https://es.farnell.com/raspberry-pi/rpi4-modbp-4gb/raspberry-pi-4-model-b-4gb/dp/3051887`
[2] `https://www.amazon.es/SanDisk-Extreme-Tarjeta-microSDXC-adaptador/dp/B07FCMBLV6/ref=sr_1_7?`
[3] `https://thepihut.com/products/letstrust-tpm-for-raspberry-pi`
[4] `https://es.farnell.com/raspberry-pi/sc0213/rpi-suministro-usb-c-5-1v-3a-blanco/dp/3106941`
[5] `https://es.farnell.com/pro-power/cat6outdoor100m/cable-cat-6-outdoor-use-100m/dp/2580443`

4.9.2 Cost Manteniment

- Cost per transacció: Encara no sabem el cost real dels *tokens* de la Teranyina. Com que aquesta està basada en Polkadot, assumim el cost per transacció de Moonbeam (0,00802177451€) perquè és la *blockchain* més semblant.

  Com sabem que el nombre total de sensors és 64, hem calculat la taula 3 per fer una estimació del cost mensual de publicar les transaccions a la *blockchain* depenent de la freqüència de publicació.

Freqüència	**Cost (€)**
1 min	22.486,64
1 h	374,78
6 h	62,46
12 h	31,23
24 h	15,62

Taula 3: Cost mensual de la publicació a la *blockchain* per diferents freqüències.

  Fent una anàlisi tenint en compte, per una banda, el cost de les transaccions i per altra la seguretat del sistema, hem cregut convenient triar una freqüència de 6 h.

  Una freqüència gaire superior implicaria massa volum de dades sense validar i podria provocar problemes als microcontroladors. En canvi, una freqüència inferior ens permetria tenir les dades validades de manera més assídua, no implicaria una millora en la seguretat, però incrementaria el cost mensual a causa de l'augment del nombre de transaccions.

  Tot i això, també hem de comentar en aquest punt que no és necessari que tots els sensors tinguin la mateixa freqüència de validació, sinó que depenen de la seva importància es podrien fer en intervals diferents. De totes maneres, per simplificar el pressupost, hem tingut en compte que tots els sensors validen les dades cada 6 hores.

  A partir d'aquestes dades podem calcular quin cost energètic té la nostra solució en termes de transaccions a la *blockchain*:

  Tenim 64 sensors que pengen 4 transaccions al dia, fent un total de 256 transaccions per dia. Sabent que de mitjana les *parachains* de Polkadot consumeixen 14 kJ per transacció (veure apartat 4.5.3), el cost energètic diari de les transaccions és de 3,584 MJ. Passant a kWh podem saber que l'impacte energètic de la nostra solució és menor a 1 kWh diari o uns 30 kWh al mes. Aquest impacte es pot compensar usant plaques solars o altres fonts d'energia renovable.

- *Host* de la base de dades: Per fer el càlcul del cost de mantenir una base de dades en funcionament hem mirat el que costaria aquest servei a Amazon Web Services. Per un servidor Amazon EC2 de 1 TB de disc dur el preu mensual seria d'aproximadament 23 €.
- Consum de les Raspberries: Si tenim 11 Raspberries que de mitjana consumeixen 5 W, en total equival a 0,055 kW o 39.6 kWh en un mes. A partir d'aquestes dades podem compensar aquesta potència usant energia renovable (plaques solars, eòlica, etc.)

El cost mensual de mantenir la nostra solució en funcionament seria de: 94,56€.

5 Prova de concepte

Hem volgut implementar una demostració molt realista de com seria l'execució en un entorn de producció. Hem implementat una solució completa des de la recollida de dades fins a la part de validació d'aquestes. Amb aquesta prova de concepte aconseguim, per una banda, demostrar la viabilitat funcional de la proposta i per altra poder fer una estimació del que suposaria implementar la proposta a una planta real.

5.1 Recollida de dades dels sensors

Per fer la recol·lecció de les dades, hem implementat un programa en Python que funciona dins una Raspberry Pi.

Aquest programa, tal com s'ha exposat a la proposta, té tres funcionalitats:

1. Recollir les dades dels sensors: al no disposar de sensors reals, hem creat un generador de dades que segueixen una distribució normal per a simular les que obtindríem dels sensors. Hem dissenyat el programa per interactuar amb aquests sensors ficticis i prendre mesures en funció de la configuració de cada sensor. Així doncs, podem simular un sensor de conductivitat elèctrica que pren mesures cada minut, un sensor d'amoni que mesura cada dos,...

   Hem deixat exemples de com s'hauria d'implementar la recollida de dades amb sensors reals de manera que seria senzill aplicar-ho a l'entorn d'una planta real.

2. Publicar les dades a la base de dades: el programa a mesura que es van prenent les mostres, les va penjant en temps real a la base de dades.

3. Publicar els *hashos* a la *blockchain*: el programa calcula el *hash* de les mostres que ha pres cada sensor a cada període i en calcula el *hash* fent servir l'algoritme SHA-256. Un cop s'ha obtingut el *hash* de les dades de cada sensor, les publica a l'*smart contract*.

El programa l'hem implementat de la manera més modular possible. D'aquesta manera podem fer canvis en qualsevol de les funcionalitats de manera molt senzilla i sense haver de tocar gaire codi. A més, és capaç de detectar i recuperar-se de forces errors comuns, com caigudes de tensió, pèrdua de connectivitat temporal, entre d'altres.

En treballar amb un fitxer de configuració d'entorn, podem canviar de *blockchain*, *smart contract* o base de dades de manera molt fàcil, tan sols caldria canviar una línia d'un fitxer de text. Això ens és útil si volem passar d'un entorn de proves a un entorn real.

5.2 Emmagatzematge de les dades

Volíem una base de dades relacional que acceptés peticions complexes per a fins estadístics com *TimescaleDB*. Tanmateix, a causa de limitacions de temps hem acabat fent servir *MariaDB*.

5.3 Verificació de les dades

Com s'ha explicat a la proposta, la verificació la pot dur a terme qualsevol persona o institució a partir de l'algoritme i els passos explicats. Per tal de poder demostrar un exemple de com es podria fer aquesta verificació, hem implementat un validador en format de pàgina web amb *Nodejs*. Algunes de les llibreries utilitzades són *express*, *mariadb* i *crypto*. El que fa aquest validador web per darrere és: Primer, recull totes les mesures des de la base de dades. Després,

en calcula el *hash* de l'interval al qual pertanyen. Finalment, executa la crida a l'*smart contract* per saber si el *hash* existeix a la *blockchain*. Tot això es mostra d'una manera amena a través de la mateixa web on es poden veure les mesures dels diferents sensors a temps real i com es van validant a través de la *blockchain*. En cas de manipulació de les dades es pot veure també com el validador web ho detecta i ho mostra a través de la interfície.

5.4 Visualització de les dades

L'eina que hem escollit per fer la visualització de les dades és el Redash. El Redash és un *software* de codi obert, que s'hauria d'hostejar en un dels servidors del Consorci d'Aigües o llogar un servidor al núvol, que permet de forma molt simple consultar la base de dades i implementar algunes visualitzacions. Aquestes visualitzacions després es poden agregar en un *dashboard* per a poder veure l'estat de la planta depuradora en temps real.

A més, Redash també permet la configuració d'alertes per quan alguns dels sensors dona resultats fora del rang de seguretat i l'enviament automàtic de correus electrònics quan passa això.

A la prova de concepte de la visualització el que hem fet ha estat simular les dades de dos sensors, el d'amoni i el de conductivitat elèctrica, del dipòsit de capçalera seguint una distribució normal.

Després, hem guardat aquestes dades a la base de dades i hem implementat diferents consultes SQL a Redash per a trobar l'històric dels sensors, els màxims i les medianes entre dues dates. Per acabar, hem dissenyat un *dashboard* que permeti veure aquesta informació junta. A la figura 3 podeu veure el resultat.

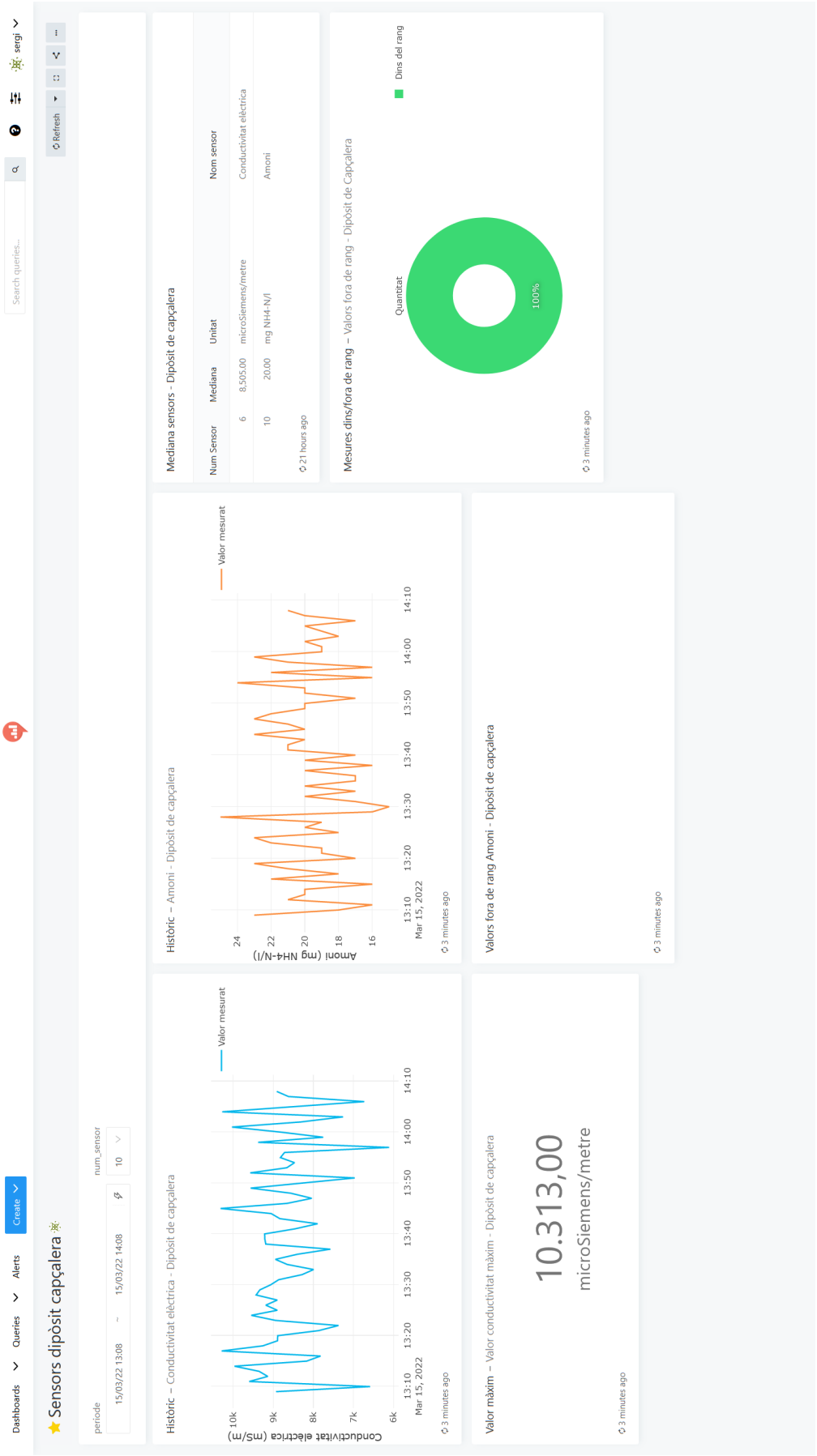

Figura 3: Dashboard implementat per la prova de concepte amb Redash.

6 Conclusions

Primerament, ens ha semblat un treball interessant de fer perquè hem après a aplicar la tecnologia *blockchain* en un nou àmbit. Hem de reconèixer que hem aprofitat l'oportunitat per aprendre més sobre aquesta tecnologia i sobre aplicacions menys conegudes d'aquesta en l'àmbit de la recollida de dades.

En segon lloc, durant l'elaboració de la proposta ens han anat sorgint problemes que hem hagut de solucionar. Un d'aquests era l'optimització de l'espai de les dades a la *blockchain*. Després de fer una pluja d'idees ens va semblar que la idea de penjar només el *hash* de les mesures era l'adient. Un altre problema que vam tenir va ser el grau de seguretat que volíem garantir. Perquè la proposta fos acceptable, que la millora no impliqués un menor rendiment del sistema, vam haver d'assumir que les connexions internes de la planta són segures. També vam estar valorant la possibilitat de preprocessar les mesures dels sensors per corregir errors de la lectura. Però com que l'objectiu és garantir la integritat d'aquestes, vam optar per no fer aquest preprocessament. Cas contrari estaríem trencant amb la idea de tenir dades que no s'han modificat a la *blockchain*.

En tercer lloc, cal dir que la falta d'un equip multidisciplinari ens ha obligat a desenvolupar una proposta centrada en la solució estrictament informàtica. Tanmateix, si l'equip hagués sigut més divers, hauríem pogut elaborar una proposta més polifacètica.

Per últim, la major part del temps l'hem dedicat a desenvolupar una prova de concepte per veure si la proposta era senzilla d'implementar. Hem pogut demostrar que amb eines senzilles com *Python* o *Nodejs* es pot implementar la lògica que interactua amb la *blockchain*. A més a més, la prova ens ha servit per veure si és viable pel que fa a recursos de *hardware* necessaris. En definitiva, hem pogut arribar a desenvolupar una prova de concepte que demostra que la solució proposada és realista i senzilla d'implementar.

7 Treball futur

En la nostra solució, han quedat alguns elements que, tot i ser d'interès, escapen del nostre abast. Seria interessant estudiar la viabilitat i la futura implementació d'aquestes característiques:

- Sistema expert per suggerir tractaments en cas que algun paràmetre surti dels rangs estimats. Aquests rangs haurien de poder ser modificats a través d'una interfície gràfica.
- Sistema d'avisos per SMS o altres plataformes per alertar quan hi hagi algun problema perquè el Redash només permet alertes per correu electrònic
- Sistema d'informes setmanals o mensuals d'estadístiques de les mesures per a la seva publicació automàtica: quantitat d'aigua tractada,...
- Utilització de tècniques de Big Data, com Apache Kafka o Apache Spark, per a realitzar el processament en *streaming* de les dades generades pels sensors.
- Efectivitat dels diferents tractaments, temps resposta, grau de millora,... Necessita moltes dades.
- Utilització d'algorismes de *machine learning* per valorar el cost de sanejar una aigua concreta, depenent de les seves característiques al dipòsit de capçalera. Necessita moltes dades.

Referències

[DigiEconomist, 2022] DigiEconomist (2022). Bitcoin Energy Consumption Index. `https://digiconomist.net/bitcoin-energy-consumption`. [Online; accessed 08-March-2022].

[Platt et al., 2021] Platt, M., Sedlmeir, J., Platt, D., Tasca, P., Xu, J., Vadgama, N., and Ibañez, J. I. (2021). Energy footprint of blockchain consensus mechanisms beyond proof-of-work. *arXiv preprint arXiv:2109.03667*.

[SolanaDevelopers, 2021] SolanaDevelopers (2021). Energy Use Report. `https://solana.com/news/solana-energy-usage-report-november-2021#fn8`. [Online; accessed 08-March-2022].