Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica per il Management

IoT e Smart Irrigation: gestione dei Big Data attraverso un sistema di notifica intelligente

Relatore:
Chiar.mo Prof.
Marco Di Felice

Correlatore:
Prof. Luca Roffia

Presentata da:
Lorenzo Pisanò

Sessione I
2020/2021

"Non ho particolari talenti, sono soltanto appassionatamente curioso"
- Albert Einstein

Indice

Capitolo 1

Introduzione

Ogni essere umano è un'entità a sé stante, eppure abbiamo sempre sentito il bisogno di connetterci gli uni con gli altri.

A partire dalla fine del XIX secolo, abbiamo iniziato a cercare, attraverso il progresso tecnologico, ciò che fino ad allora sembrava irraggiungibile: comunicare in qualsiasi momento, da qualsiasi luogo e con persone molto lontane. Nasce così il World Wide Web (WWW) il 6 agosto 1991, a seguito della prima pubblicazione di un sito web da parte dello studioso inglese Tim Berners-Lee. Successivamente, la naturale evoluzione della tecnologia diventa connettere virtualmente oggetti e luoghi concreti. Proprio da questa necessità, nel 1999, nasce l'Internet delle cose o l'Internet of Things (IoT). 'Oggetti' o 'cose' (Things), che possono comunicare dati ed informazioni diversamente aggregate, determinando le proprie identità digitali. Lo studio della startup 'Statista' [Sta21] dimostra che, nel 2020, il numero degli oggetti connessi ad internet è pari a 8.74 miliardi.

Con questo lavoro di tesi ho potuto approfondire anche un altro tema di grande attualità collegato ad IoT, la 'Smart Irrigation', conosciuta anche come 'Irrigazione di precisione'. Considerando la necessità sempre più evidente di migliorare la gestione della distribuzione irrigua ed energetica nel campo dell'agricoltura e, tenendo presente le indicazione meteoclimatiche e, l'importanza di avere informazioni tempestive ed aggiornate per migliorare le attività in campo, la Smart Irrigation assume un ruolo rilevante nel risparmio idrico ed energentico, evitando sprechi ed usi impropri di queste preziose risorse.

Il software che ho realizzato è stato sviluppato nell'ambito di un programma europeo più vasto, il progetto SWAMP (Smart WAter Management Platform) [SWA21], che ha come obiettivo quello di determinare una svolta decisiva nell'utilizzo moderato e privo di sprechi dell' acqua dolce ad uso irriguo, proponendo un sistema efficiente per la gestione della distribuzione di questo bene in vari contesti. L'area di competenza del progetto fa parte di quella amministrata dal Consorzio di Bonifica dell'Emilia Centrale (CBEC), responsabile delle irrigazioni e del drenaggio d'acqua di un'area di 1200 km2 suddivisi in circa 5400 terreni proprietari [Bon21].

Il software di seguito descritto genera un sistema di acquisizione dati provenienti da alcuni pluviometri dislocati nel comune di Bologna. Successivamente, li elabora e classifica la quantità di pioggia che cade nell'area di studio in 5 differenti livelli di rischio. Queste informazioni vengono poi notificate all'utente attraverso la piattaforma WDA [ARC21a], permettendo di ovviare ad eventi di inondazione e alluvione anche nelle aree adiacenti a quelle classificate 'a rischio'.

Questo elaborato è suddiviso in quattro capitoli: il primo tratta dell'Internet of Things e dell'irrigazione di precisione. Il secondo offre una panoramica dello stato dell'arte del Web Semantico. Successivamente si introduce il progetto SWAMP [SWA21] e la sua architettura. Gli ultimi capitoli sono dedicati esclusivamente alla presentazione dell'applicazione, al suo funzionamento ed ai possibili sviluppi futuri.

Capitolo 2

Stato dell'Arte: IoT and Smart Agriculture

2.1 Internet of Things

2.1.1 Storia della comunicazione mobile

Partendo dal primo dispositivo smartphone, inventato da Martin Cooper che effettuò la prima telefonata da un cellulare il 3 aprile 1973 [Coo21] fino ad arrivare ad oggi, possiamo osservare cambiamenti straordinari.

Nel 1973, Motorola, da leader del mercato elettronico, iniziò a consentire ai suoi utenti di telefonare in mobilità connettendosi alle antenne analogiche, TACS e ETACS (varianti del Total Access Communication System), sparse sul territorio suddiviso in celle.

Successivamente vennero installate le antenne digitali 'GSM' (Global System for Mobile Communications) che consentirono anche l'invio di brevi messaggi (SMS), sviluppandosi velocemente grazie all'integrazione di connessioni sempre più veloci e affidabili alla rete Internet. Le novità rispetto alle funzionalità offerte dal mercato erano molteplici come l'interoperabilità tra reti diverse che facevano capo ad un unico standard internazionale e la comunicazione di tipo digitale.

Con l'introduzione di questa nuova tecnologia, venne introdotta una maggiore velocità di trasmissione grazie alle tecniche di compressione dati, al sistema di autenticazione ottimizzato (attraverso SIM card) e all'introduzione della sicurezza attraverso crittografia. Ad esempio, A5/1 è l'algoritmo che sembrava poter garantire un maggior livello di protezione per i dati che transitavano attraverso le diverse antenne digitali. Ma non tutti i paesi aderirono a questo standard. Venne, dunque, presentata una seconda versione A5/2 distribuita nelle celle di continenti come America ed Asia. Entrambi i sistemi

(A5/1 e A5/2) si rivelarono estremamente vulnerabili, tanto che furono previsti meccanismi automatici di cambio dell'algoritmo in caso di necessità.

Il primo paese a introdurre la tecnologia 3G su scala commerciale è stato il Giappone: nel 2001 circa il 40% delle utenze mondiali erano esclusivamente su reti 3G e la transizione della maggior parte delle utenze da reti 2G a 3G era prevista concludersi verso la fine del 2006. Con questo passaggio, si diede vita ad una nuova generazione tecnologica, dove fu possibile osservare una velocità di trasmissione che passa dai 20-40 kpbs del 2G ai 380-700 kbps presentati con l' Universal Mobile Telecommunications System (3G). Questa evoluzione ha permesso di introdurre la comunicazione di tipo multimediale includendo file come note vocali ed immagini.

La quarta generazione di reti Internet Mobile inizia realmente il 27 Giugno 2011, con la pubblicazione del bando sulla Gazzetta Ufficiale [Gaz11] per l'assegnazione delle licenze agli operatori mobili interessati. Con l'introduzione di questa nuova tecnologia 4G, viene introdotta la possibilità di accedere a contenuti in streaming ad alta definizione attraverso la sola connessione wireless grazie alla velocità teorica approssimativa fino a 150 Mbps.

Nel 2011 viene inoltre pubblicato il nuovo protocollo Internet Ipv6, che sostituisce l'Ipv4, pubblicato dalla IETF nel settembre 1981. Il nuovo protocollo Ipv6 amplia in modo esponenziale i servizi offerti dalla rete, semplificando la gestione degli indirizzi IP. Inoltre, a partire dal 2015 molte delle grandi aziende che iniziano ad offrire servizi Cloud mettono a disposizione piattaforme per sviluppare l'Internet of Things (IoT).

Arriviamo così ai giorni nostri, che vedono la quinta generazione di reti di internet mobile (5G) sempre più reale. La rete 5G sarà infatti caratterizzata da una maggiore velocità (fino ai 1000 Mega), da un basso tempo di latenza e da una migliore continuità e qualità del segnale. Queste caratteristiche fanno pensare che forse si può andare oltre la già abituale comunicazione telefonica e navigazione in Internet per provare a fare qualcosa che possa coinvolgere anche nuovi campi e nuove tipologie di applicazioni (Figura 2.1).

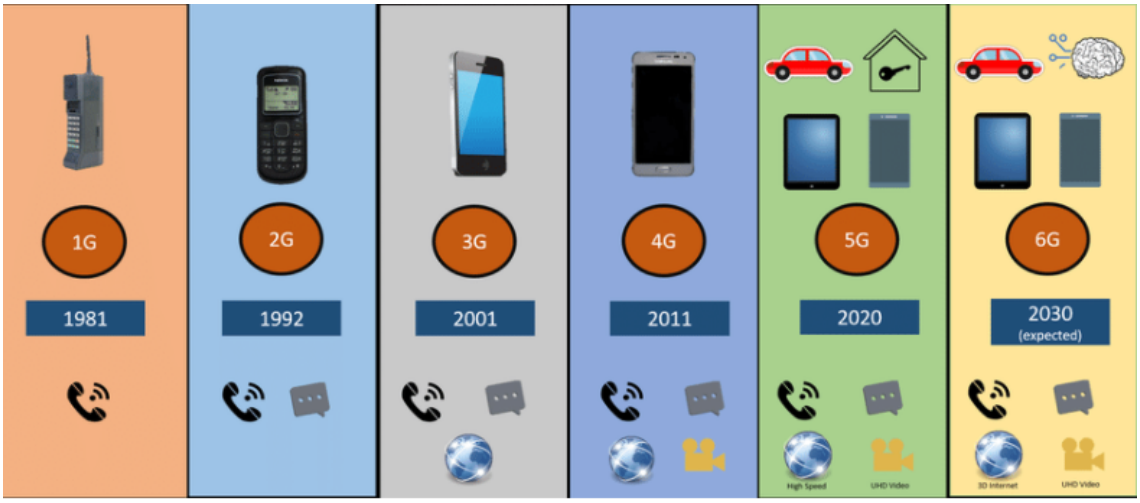

Figura 2.1: Evoluzione della comunicazione mobile dall'1G al 5G, Fonte [RR20]

Per questo motivo la rete 5G è considerata come uno dei presupposti allo sviluppo di quello che oggi viene chiamato 'Internet of Things' (o Internet delle cose o degli oggetti), termine con il quale si vuole indicare la possibilità di estensione della rete Internet anche agli oggetti in grado di supportare le tecnologie necessarie.

2.1.2 Architetture IoT

Per le applicazioni IoT si utilizzano le funzionalità dei microcontrolli. I *microcontrollori* sono circuiti integrati digitali che contengono (nella forma più ridotta) un unico chip e due tipi di memoria, una di tipo non volatile (ROM) ed una volatile (RAM). Inoltre, il circuito possiede un timer ed una porta I/O per comunicazione di tipo seriale.

Questi dispositivi, anche se dotati di risorse limitate, dialogano continuamente tra di loro secondo protocolli ben precisi ed in maniera sincronizzata. I microcontrollori sono in grado di raccogliere dati mediante sensori, di memorizzarli e di inviarli sulla rete. Questo processo viene solitamente effettuato a livello Cloud, dove i dati vengono analizzati, processati ed elaborati, per fornire nuove informazioni utili al cittadino ed ad altre applicazioni secondo gli obiettivi del progettista.

Tutti questi dati, vengono visualizzati dall'utente finale come messaggi push su mobile app, grafici su browser, Qr code e molto altro. L'intero ecosistema di oggetti Hardware/Software che fa parte di un progetto IoT è basato su un'architettura composta da elementi assemblati fra di loro con maestria ed esperienza.

Quando si effettua un'analisi sulla struttura del sistema che si vuole generare, è necessario tener conto di;

- Dispositivi hardware
- Tecnologie di comunicazione Machine-to-Machine
- Servizi Cloud
- Applicazioni software

Dispositivi hardware

I dispositivi hardware possono essere raggruppati in due macrogruppi;

- **A corto raggio:** la comunicazione viene effettuata in rete attraverso bluethoot oppure via cavo. La velocità di comunicazione ed il range del dispositivo viene definito dalla versione del bluethoot che si desidera utilizzare. Per questa scelta, è necessaria un'analisi del trade-off tra efficienza e consumo di batteria. Ad oggi, quest'ultimo fattore è preso molto in considerazione dall'utente finale, che giudica l'applicazione singola come 'molto dispendiosa' in base a quanto questa va ad impattare sull'efficienza e durata del suo dispositivo. E' possibile consultare i dettagli delle differenti versioni di connessione bluetooth attraverso la figura 2.2.

	BLUETOOTH V2.1	BLUETOOTH 4.0 (LE)	BLUETOOTH 5 (LE)
Range	Up to 100 m	Up to 100 m	Up to 400 m
Max range	Around 100 m	Around 100 m	Around 1,000m
Frequency	2.402 – 2.481 GHz	2.402 – 2.481 GHz	2.402 - 2.481 GHz
Max data rate	1- 3 Mbit/s	1 Mbit/s	2 Mbit/s
Application Troughput	0.7-2.1 Mbit/s	Up to 305 kbit/s	Up to 1,360 kbit/s
Topologies	Point-to-point, scatternet	Point-to-point, mesh network	Point-to-point, mesh network
Network Standard	IEEE 802.15.1	IEEE 802.15.1	IEEE 802.15.1

Figura 2.2: Analisi versioni bluetooth

- **A lungo raggio:** la comunicazione viene effettuata tramite Web senza alcuna connessione cavo o bluetooth. Questo è possibile data la presenza di una SIM (Subscriber Identity Module) all'interno dell'apparecchiatura che la identifica in modo univoco. La scheda 'IoT SIM Card' è pensata appositamente per una configurazione remota per dispositivi IoT. Questa permette una connessione a più canali per avere una copertura maggiore ed un accesso lato server per interagire con i dati prodotti dal sensore/attuatore. Inoltre, il traffico dati che viene prodotto a partire dalla IoT SIM Card si appoggia spesso a servizi VPN(come OpenVPN e IPSec VPN), che permettono una comunicazione crittografata e stabile, partendo da un indirizzo IP fisso.

Tecnologie di comunicazione Machine-to-Machine

Il contenuto di questo paragrafo si basa sull'articolo 'Machine-to-machine wireless communication technologies for the Internet of Things: Taxonomy, comparison and open issues' [Fed+18], scritto da Federico Montori, Luca Bedogni, Marco Di Felice e Luciano Bononi dell'Università di Bologna, pubblicato il 23 agosto 2018 nella piattaforma Science Direct [Sci21].

In campo IoT e Smart Irrigation è rara la presenza di sistemi interconessi attraverso la tecnologia wired, vista la necessità di installare sensori sul campo e di trasmettere dati in tempo reale. Per esempio, nel progetto SWAMP, si installano sensori nei campi agricoli che si trovano nella zona amministrata dal Consorzio di Bonifica dell'Emilia Centrale (CBEC) per ricevere informazioni rispetto all'umidità del terreno. Questo viene pensavo per prevedere, in modo più accurato possibile, quando effettuare l'operazione d'irrigazione. E' possibile dunque dedurre che con un approccio di questo tipo, non è pensabile strutturare un'architettura di rete di tipo Wired.

Una tecnologia di comunicazione spesso utilizzata in campo IoT è la Machine-to-Machine (M2M).

Quando si parla di questa tecnologia, si fa riferimento alla connessione ed alla comunicazione di due o più dispositivi che utilizzano e modificano informazioni condivise senza l'utilizzo di un server centralizzato e senza l'intervento umano. Quest'ultima caratteristica, gioca un ruolo essenziale nella gestione delle reti IoT che scambiano informazioni in modo molto frequente.

Per realizzare questo tipo di comunicazione, è possibile utilizzare sia un tipo di connessione wired sia wireless. In contrapposizione, il campo IoT si riferisce spesso ad una comunicazione senza fili.

Tra i forti requisiti che caratterizzano la tecnologia M2M, in questo elaborato di tesi se ne discutono tre in particolare perchè rilevanti nella progettazione dell'applicativo.

- **Basso consumo energico:** i sensori ed attuatori sono spesso alimentati tramite batterie che richiedono di essere caricate per permettere un corretto funzionamento del dispositivo. E' possibile allungare la 'vita' della batteria grazie a sistemi di ottimizzazione del suo utilizzo.

  Di seguito si riporta uno studio 'A low-cost wireless sensor network for long term monitoring of energy performance and sustainability of buildings' scritto da alcuni ricercatori 'dell'Università della Lettonia' [JAI19] che illustra un esempio per il risparmio energetico del microcontrollore ATMega328. I grafici in figura 2.3 e 2.4 mostrano l'ottimizzazione del dispendio energetico del microprocessori attraverso la disattivazione di moduli definiti non necessari.

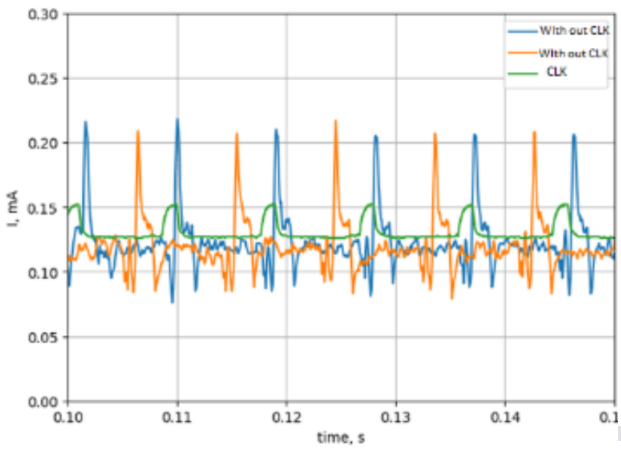

Figura 2.3: Diagramma di corrente a 3,3V di Atmega328P con NRF 24 in modalità sleep prima della disattivazione dei moduli non necessari, Fonte [JAI19]

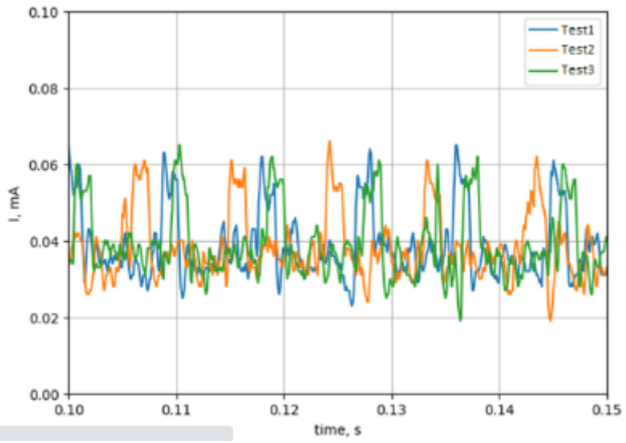

Figura 2.4: Diagramma di corrente a 3,3V di Atmega328P con NRF 24 in modalità sleep dopo della disattivazione dei moduli non necessari, Fonte [JAI19]

Durante lo sviluppo del programma, è stato discussa la necessità di dover ottimizzare la durata della batteria dei sensori sul campo utilizzati dal progetto SWAMP, in modo da garantire un'efficienza dei dati prodotti senza un costante intervento umano.

- **Scalabilità:** il sistema deve garantire scalabilità, per rendere possibile l'estenzione del numero di dispositivi che si uniscono alla rete. Il progetto SWAMP si dimostra facilmente adattabile a questa necessità, con la possibilità di aggiungere sensori

ed attuatori che interagiscono attivamente attraverso la rete internet. E' quindi necessario che si crei un sistema pronto per gestire la dinamicità, evitando di aggiungere condizioni in cui il superamento di un valore statico potrebbe causare un malfunzionamento;

"Scalability also impacts the channel access method, since in dynamic scenarios – i.e. with a non-static number of participants and with dynamically entering and leaving nodes - contention based methods face an increase of collisions, whereas contention-free ones need to deal with a time-consuming reconfiguration"

"La scalabilità ha anche un impatto sul metodo di accesso al canale, poiché in scenari dinamici - cioè con un numero non statico di partecipanti e con nodi che entrano ed escono dinamicamente - i metodi basati sulla contesa affrontano un aumento delle collisioni, mentre quelli senza contesa devono affrontare una riconfigurazione che richiede tempo"

- **Bassa latenza:** la latenza viene definita come il ritardo che si verifica da quando viene fatta la richiesta di una risorsa Web, fino a quando la si ottiene. Questa caratteristica è stata affrontata in fase di sviluppo del programma perchè era necessario valutare la tempistica che intercorreva tra l'invio del dato pluviometrico fino alla ricezione da parte dell'aggregatore. In fase di progettazione è dunque necessario prevedere vari ritardi dovuti all'architettura della rete, strutturando in modo corretto ed utilizzando strumenti che influenzino la latenza;

  "There are physical deployment dependencies such as the link strength between the endpoints and the number of hops in an average communication path as well as the number of nodes in the network. PHY layer mechanisms such as spread spectrum techniques, modulation and coding schemes, frequency and spatial diversity also greatly affect latency"

  "Ci sono dipendenze di distribuzione fisica come la forza di collegamento tra gli endpoint e il numero di salti in un percorso di comunicazione medio, così come il numero di nodi nella rete. Anche i meccanismi del livello PHY, come le tecniche di spread spectrum, gli schemi di modulazione e codifica, la diversità di frequenza e spaziale, influenzano notevolmente la latenza."

Un esempio di tecnologia che utilizza il concetto di M2M è la 'wearable technology'. Con questo termite si intende un insieme di dispositivi indossabili che si connettono allo smarthphone attraverso tecnologie a corto raggio e a bassa larghezza di banda per comunicare con altri apparecchi IoT.

Servizi Cloud

Il Cloud Computing è un termine molto vasto che definisce una collezione di servizi offerti dalla rete internet. La configurazione viene fatta in remoto basandosi su una architettura distribuita.

Il Cloud Computing ha un ruolo molto importante nell'Internet of Things. Uno dei motivi dello stretto collegamento tra il Cloud Computing e l'IoT è che la tecnologia dei dispositivi intelligenti si basa sulla portabilità e sulla semplicità del *microcontrollore*, che è programmato per condividere i dati generati, liberando la limitata memoria a sua disposizione.

Il connubio Cloud-IoT caratterizza quindi la quantità di dati che l'IoT è in grado di elaborare. I servizi offerti attraverso la rete internet mettono a disposizione spazi virtualmente illimitati, ospitando dati che possono essere analizzati al fine di trarre conclusioni rilevanti su utenti, tendenze del mercato e così via. Per questo motivo, si definisce che l'IoT viene sorretto dalla velocità e dalla flessibilità del Cloud.

L'articolo di ricerca 'On the integration of Cloud Computing and internet of things' di alcuni ricercatori dell'Università di Napoli 'Federico II', riporta i vantaggi che entrambe le tecnologie ricevono collaborando l'una con l'altra. [Ale+14]

"Specifically, the Cloud can offer an effective solutionto implement IoT service management and composition as well as applications that exploit the things or the data produced by them. On the other hand, the Cloud can benefit from IoT by extending its scope to deal with real world things in a more distributed and dynamic manner, and for delivering new services in a large number of real life scenarios."

"In particolare, il Cloud può offrire una soluzione efficace per implementare la gestione e la composizione dei servizi IoT e le applicazioni che sfruttano le cose o i dati prodotti da esse. D'altra parte, il Cloud può beneficiare dell'IoT estendendo il suo scopo per trattare con le cose del mondo reale in modo più distribuito e dinamico, e per fornire nuovi servizi in un gran numero di scenari di vita reale."

Negli ultimi anni i servizi Cloud sono stati oggetto di una sempre maggiore richiesta. Anche la maggior parte dei colossi del web, come Google o Aruba, offrono, a condizioni vantaggiose, spazi virtuali utili alla collezione delle informazioni in server fisici collocati in Data Center.

Avere un proprio spazio fisico (Data Center) contenente dati, può essere molto rischioso in diversi ambiti. E' infatti necessario valutare sia la gestione dei dati virtuali,

considerandone la struttura e la necessaria protezione contro possibili attacchi informatici, sia la gestione dell'apparato fisico, in quanto necessita particolare attenzione ai consumi di energia e alla tutela dell'hardware rispetto a danneggiamenti da calamità naturali.

Questi sono solo alcuni esempi delle possibili problematiche che devono essere affrontate nel caso in cui l'analisi della gestione dei dati risulta più conveniente attraverso una memorizzazione su apparato fisico.

Molti sono i vantaggi che il Cloud hosting offre all'utente finale. Uno studio dell'università di Stamford [STA20] pubblicato nel 23 luglio 2020 riporta il valore economico dei servizi Cloud dal 2018 al 2022 (Figura 2.5), dimostrando la crescita dell'interesse di questi servizi negli ultimi anni.

Table 1. Worldwide Public Cloud Service Revenue Forecast (Billions of U.S. Dollars)

	2018	2019	2020	2021	2022
Cloud Business Process Services (BPaaS)	41.7	43.7	46.9	50.2	53.8
Cloud Application Infrastructure Services (PaaS)	26.4	32.2	39.7	48.3	58.0
Cloud Application Services (SaaS)	85.7	99.5	116.0	133.0	151.1
Cloud Management and Security Services	10.5	12.0	13.8	15.7	17.6
Cloud System Infrastructure Services (IaaS)	32.4	40.3	50.0	61.3	74.1
Total Market	**196.7**	**227.8**	**266.4**	**308.5**	**354.6**

BPaaS = business process as a service; IaaS = infrastructure as a service; PaaS = platform as a service; SaaS = software as a service

Figura 2.5: Proiezione del profitto Cloud Hosting dal 2018 al 2022, Fonte [STA20]

Tale concetto è confermato anche dal grafico pubblicato dalla startup Statista (Figura 2.6)[Hol21], che immagina un aumento esponenziale dei big data nei prossimi anni. Dati che richiederanno ambienti di gestione sempre più sicuri e flessibili.

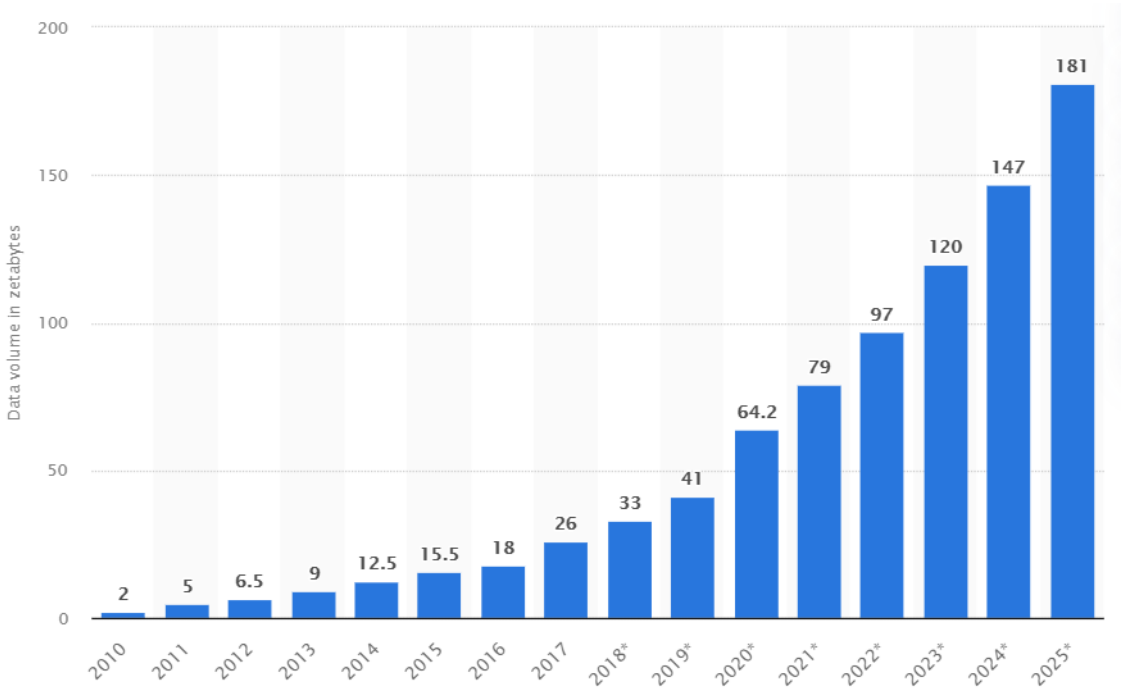

Figura 2.6: Proiezione Statista volume Big Data in crescita dal 2010 al 2025, Fonte [Hol21]

2.2 Smart Irrigation

Il contenuto di questo paragrafo si basa sull'articolo di Nicoletta Boldrini, 'Consumo di acqua eccessivo, come intervenire?' [Bol21]

Con Smart Irrigation si intende l'insieme delle tecniche e tecnologie che costituiscono un sistema integrato ed innovativo per la gestione intelligente delle pratiche di irrigazione.

L'agricoltura è tra le attività umane che da sempre è causa di un elevato consumo di acqua dolce. Si calcola che a livello mondiale, secondo stime Unesco [Une15]), circa il 70 % dell'acqua prelevata dai fiumi, dai laghi e dalle falde sotterranee sia destinato all'irrigazione. E non solo. L'attività agricola è anche l'industria che causa importanti fonti di inquinamento: l'utilizzo di fertilizzanti e pesticidi, ad esempio, produce un forte impatto qualitativo sulle acque, l'agricoltura intensiva è spesso motivo di importanti variazioni degli *habitat* e dell'erosione dei suoli. Alcune pratiche agricole inoltre, portano, a volte ad una modifica della salinizzazione del terreno e delle acque oppure ad impatti ambientali e paesaggistici per la realizzazione delle infrastrutture necessarie al prelievo, al trasporto o alla distribuzione delle acque.

Queste criticità del comparto agricolo si sommano inoltre all'aumento del fabbisogno d'acqua da parte di una popolazione urbana sempre più densa, ed alla maggiore richiesta di acqua da parte di altre industrie (energia, manifattura, etc...).

Questo problema non è nuovo, viene infatti analizzato già nel 1950 dalla FAO (Food and Agriculture Organization of the United Nations) [FAO21], Organizzazione delle Nazioni Unite, che ha tra i suoi obiettivi quello di contribuire ad accrescere i livelli di nutrizione, aumentare la produttività agricola, migliorare la vita delle popolazioni rurali e contribuire alla crescita economica mondiale.

I primi studi comprendono dati rilevati negli anni 1950-1995 [Fao02] ed è possibile trovare informazioni molto interessanti riguardanti stime e proiezioni di dati rispetto al prelievo d'acqua per zona e settore. Questi dati vengono riportati nel report 'Acqua per le Colture, Ogni goccia conta', pubblicato dalla FAO nel 2002 (Figura 2.7).

La Fao sottolinea inoltre l'importanza della distinzione tra acqua prelevata e acqua effettivamente utilizzata, evidenziando lo spreco giornaliero di acqua nei diversi settori industriali (Figura 2.7). Per comprenderne meglio l'importanza, di seguito si trascrive uno dei paragrafi presenti nel report (Capitolo Acqua per usi agricoli) che analizza il ciclo di vita dell'acqua prima del suo utilizzo nei terreni agricoli.

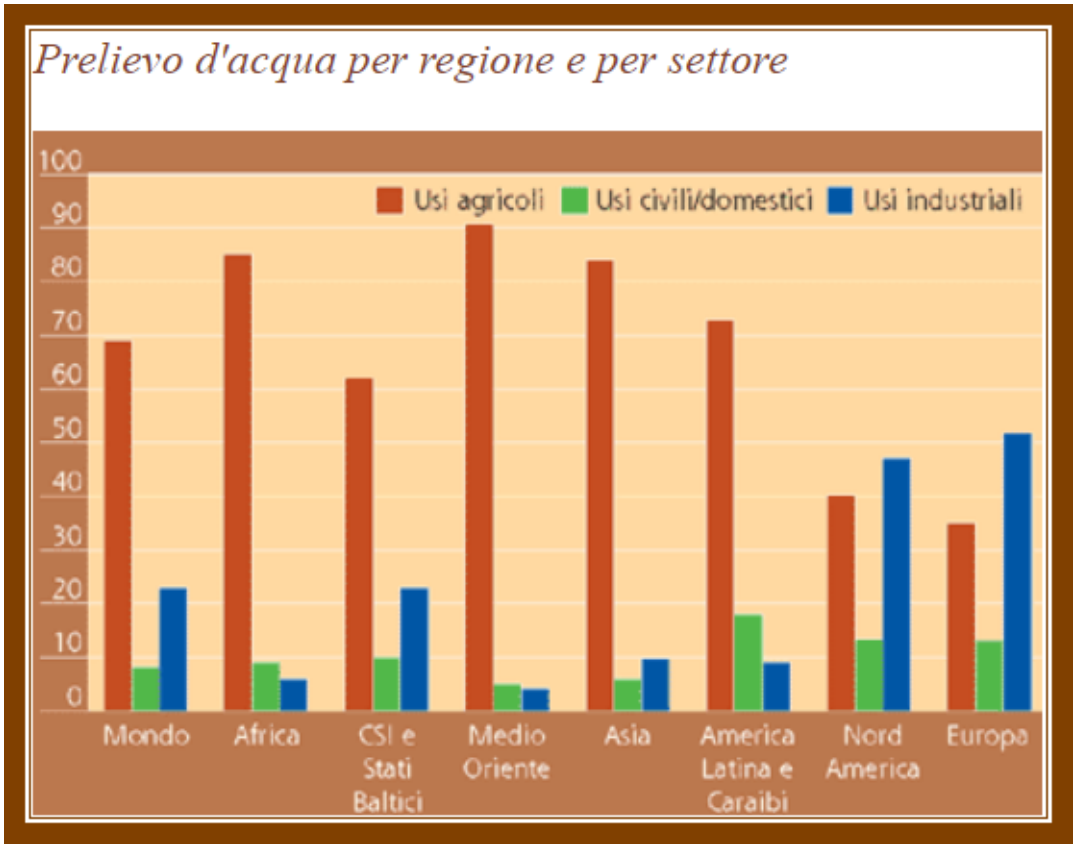

Figura 2.7: Prelievo d'acqua per regione e per settore, Fonte [FAO02]

"È importante distinguere fra l'acqua che è prelevata e l'acqua che è effettivamente consumata. Dei 3600 km3 d'acqua prelevata annualmente, all'incirca la metà è consumata dall'evaporazione e dalla traspirazione delle piante. L'acqua che è stata prelevata ma non consumata, per contro, ruscella nuovamente sulla superficie verso i fiumi o s'infiltra nel terreno e s'immagazzina negli acquiferi. Quest'ultimo tipo d'acqua è generalmente di qualità minore rispetto a quella prelevata. L'irrigazione consuma la maggior parte dell'acqua prelevata (spesso la metà o anche di più) quale risultato dell'evaporazione, dell'inclusione nel raccolto e della traspirazione dalle piante. L'altra metà ricarica la falda o il flusso superficiale o si perde in evaporazione non produttiva."

Inoltre, il documento stimava il prelievo mondiale d'acqua per le diverse tipologie di settore, evidenziando già all'epoca un importante prelievo del settore agricolo (Figura 2.8).

Stima del prelievo mondiale d'acqua (km^3 annui, m^3 pro capite e percentuale del prelievo totale)

	1950	1995
Agricoltura		
prelievo	1100	2500
pro capite	437	436
percentuale del totale	79	69
Industria		
prelievo	200	750
pro capite	79	131
percentuale del totale	14	21
Usi civili/domestici		
prelievo	100	350
pro capite	40	61
percentuale del totale	7	10
Totale		
prelievo	1 400	3 600
pro capite	556	628
percentuale del totale	100	100

Nota: tutti i valori sono arrotondati.

Figura 2.8: Stima prelievo mondiale di acqua 1950-1995, Fonte [FAO02]

La FAO, proprio perchè da anni si occupa di queste problematiche, è il detentore e custode dell'Indicatore OSS (Obiettivo di Sviluppo Sostenibile) 6.4.2. Questo obiettivo misura la pressione delle attività umane sulle fonti naturali di acqua dolce ed è presente nell'Agenda 2030 dove sono riportati gli SDGs (Sustainable Development Goals) [Nat21], 17 obiettivi interconnessi (Figura 2.9), definiti dalle Nazioni Unite come strategia "per ottenere un futuro migliore e più sostenibile per tutti". La Smart Irrigation e l'ottimizzazione dell'utilizzo dell'acqua sono compresi nell'obiettivo numero 6:

"Goal 6. Ensure availability and sustainable management of water and sanitation for all"

Figura 2.9: Illustrazione dei 17 obiettivi delle Nazioni Unite, Agenda 2030

Più in dettaglio, nell'ambito del Goal 6, sono presi in considerazione più indicatori target che dedicano uno spazio alla minimizzazione degli sprechi dell'acqua negli obiettivi delle Nazioni Unite. I target sono riportati nella pagina web relativa alle statistiche pubblicate dall'ONU [ONU21] e che qui riporto per chiarezza.

- **Target 6.4:** By 2030, substantially increase water-use efficiency across all sectors and ensure sustainable withdrawals and supply of freshwater to address water scarcity and substantially reduce the number of people suffering from water scarcity
  - **Indicator 6.4.1:** Change in water-use efficiency over time
  - **Indicator 6.4.2:** Level of water stress: freshwater withdrawal as a proportion of available freshwater resources
- **Target 6.5:** By 2030, implement integrated water resources management at all levels, including through transboundary cooperation as appropriate

- **Indicator 6.5.1:** Degree of integrated water resources management
- **Indicator 6.5.2:** Proportion of transboundary basin area with an operational arrangement for water cooperation

Nel capitolo 3 - SWAMP verrà riportato come il progetto, con la tecnica della Smart Irrigation, contribuisce al raggiungimento di questi obiettivi definiti dall'Organizzazione delle Nazioni Unite.

Capitolo 3

Stato dell'Arte: Web Semantico

3.1 Web Semantico

3.1.1 Introduzione

Durante la fase di analisi e di progettazione del mio elaborato che verrà illustrato nel capitolo 5 "Sistema di notifica autonomo", ho fatto riferimento al libro 'Web Semantico' di Berners-Lee [T B01], che definisce questa tecnologia come:

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

"Il web semantico e' un'estensione del web che ad oggi utilizziamo, dove viene assegnato un significato ben definito alle informazioni, consentendo ai computer ed alle persone di lavorare con un approccio cooperativo."

In effetti il processo si è sviluppato sulla base della 'Torre del Web Semantico' (Figura 3.1), strumento grafico noto a molti sviluppatori. Per sfruttarla al meglio è necessario definire ogni gradino della scala, partendo dalla base.

Il blocco portante di base è costituito dal metalinguaggio *"XML"*(Extensible Markup Language), ovvero un linguaggio marcatore basato sul meccanismo sintattico che consente di definire e controllare il significato degli elementi contenuti in un documento o in un testo.

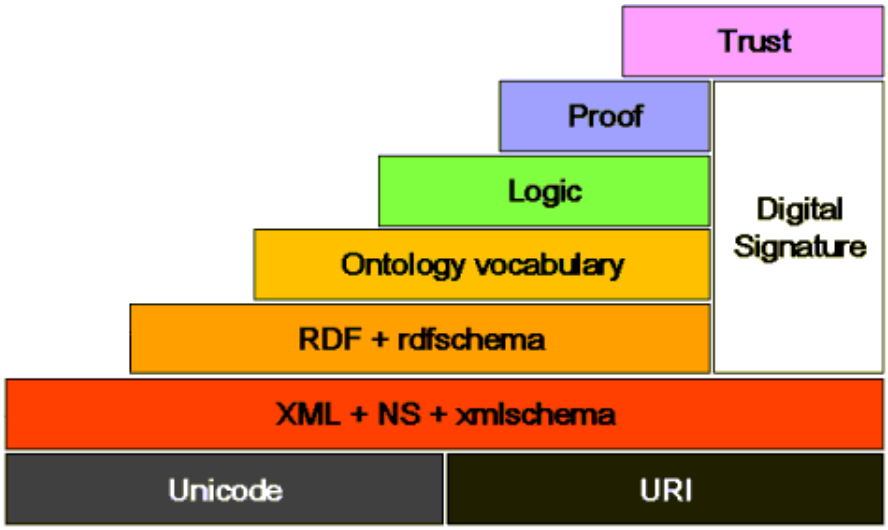

Figura 3.1: La Torre del Web Semantico, Fonte [T B01]

E' necessario notare che XML si basa a sua volta su due grandi blocchi: *"Unicode"* ed *"URI"* (Uniform Resource Identifier). Il primo gruppo, Unicode, è relativo ad un sistema di codifica che assegna un numero univoco ad ogni carattere usato per la scrittura di testi, in maniera indipendente dalla lingua, dalla piattaforma informatica e dal programma utilizzato. Il secondo, URI, invece, fa riferimento al concetto di una sequenza di caratteri che identifica universalmente ed univocamente una risorsa.

Il vero contenuto innovativo, quello che distingue il Web Semantico da XML, e' dato dagli strati superiori della torre, in particolare dal primo strato successivo ad XML, quello di *"RDF"* (Resource Description Framework).

RDF e' un linguaggio di base che per esprimere l'informazione, usa frasi composte da "triple", cioè soggetto, predicato, complemento oggetto (?s ?p ?o). Questo permette di aggregare informazioni provenienti dalle diverse parti del Web, condividendo dati tra molteplici piattaforme e sfruttando al massimo il loro potenziale. Analogamente, se abbiamo una informazione complessa espressa in RDF, possiamo analizzare la loro struttura ed estrarre informazioni in modo rapido e sicuro. Questo ci permette, in presenza di Big Data, di estrarre facilmente informazioni più piccole e trattabili.

Salendo ancora un gradino della Torre, troviamo le *"Ontology vocabulary"* (ontologie), sistemi dettagliati di classificazione. Ogni ontologia definisce un concetto di 'classe' o 'categoria', quindi un insieme di oggetti. Le informazioni contenute danno un dettaglio di dati sempre più preciso.

Prefix	URI
schema	http://schema.org/
qudt	http://qudt.org/schema/qudt#
unit	http://qudt.org/vocab/unit#
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
owl	http://www.w3.org/2002/07/owl#
xsd	http://www.w3.org/2001/XMLSchema#
rdfs	http://www.w3.org/2000/01/rdf-schema#
xs	http://www.w3.org/2001/XMLSchema#
time	http://www.w3.org/2006/time#
sosa	http://www.w3.org/ns/sosa/

Figura 3.2: Ontologie usate dal SEPA

I due livelli successivi sono *"Logic"* (Logica) e *Proof* (Prova). Sono livelli strettamente correlati, che permettono di effettuare ragionamenti anche molto complessi tramite l'uso di 'regole' (rules), l'equivalente di un linguaggio di programmazione ad alto livello, come Java.

L'ultimo gradino della Torre e' rappresentato dal *"Trust"*. Per spiegare questo concetto si può provare a creare un parallelismo tra i Big-Data del Web of Trust e la società civile. L'obiettivo finale è quello del "Web of Trust", cioè di un Web che offra riservatezza e che ispiri sempre più fiducia da parte di chi ne usufruisce. E' però difficile aver fiducia in un gran numero di persone (Big-Data) e questo potrebbe limitare l'utilizzo del Web. Lo scenario che si vuole realizzare è che un utente comunica che ha fiducia in una determinata persona. A sua volta questa persona ha fiducia in altre persone e queste ultime in altre. Tutte queste relazioni di fiducia si aprono come un ventaglio e formano il Web of Trust. Ognuna di queste relazioni avrà un grado di fiducia, associato ad essa. E' da notare che anche la sfiducia è utile come la fiducia. Si supponga che un programma, durante una ricerca, trovi un documento al quale nessuno ha dato esplicita fiducia, ma che nemmeno ha dato esplicita sfiducia. Il programma darà più fiducia, o credibilità, a questo documento piuttosto che ad uno che - per qualche ignota ragione - è stato completamente sfiduciato.

Risalendo la Torre abbiamo verificato come, inserendo nel Web Semantico un'informazione, anche molto articolata e classificabile, se si opera opportunamente, questa potrà essere definita dal programma stesso sicura e affidabile.

3.1.2 SPARQL and Linked Data

SPARQL è un linguaggio di interrogazione (Query) per dati rappresentati tramite RDF, linguaggio standardizzato dal Data Access Working Group, cioè un gruppo di lavoro del consorzio W3C che lo ha reso pubblico ufficialmente il 15 gennaio 2008.

SPARQL è un elemento chiave del "Web Semantico" e consente di estrarre informazioni dalle basi di conoscenza distribuite sul Web. RDF invece, descrive i concetti e le sue relazioni sui dati attraverso l'introduzione di triple (soggetto-predicato-oggetto). Se tali triple hanno degli elementi in comune emerge un grafo di conoscenza.

I Linked Data esprimono la necessità di rappresentare, attraverso gli standard RDF e SPARQL, ogni oggetto reale e virtuale con un URI di riferimento ed inoltre, di collegare questi oggetti tra loro tramite links ad altri URI. Il design utilizza le tre tecnologie standard e fondamentali del Web (HTML, HTTP, URI(s)). Ogni indirizzo dunque, inizia con HTTP e avrà il compito di restituire informazioni in un formato aperto, tra le quali relazioni ad altri oggetti, attraverso altri indirizzi HTTP.

Le specifiche dei Linked Data permettono di fatto l'associazione di concetti che fanno parte anche di Dataset (o Database) differenti. Questo ha portato alla nascita del progetto Linking Open Data, nel quale si sta popolando un grafo in continua espansione, collegando fra loro oggetti provenienti da Dataset differenti, utilizzando triple RDF.

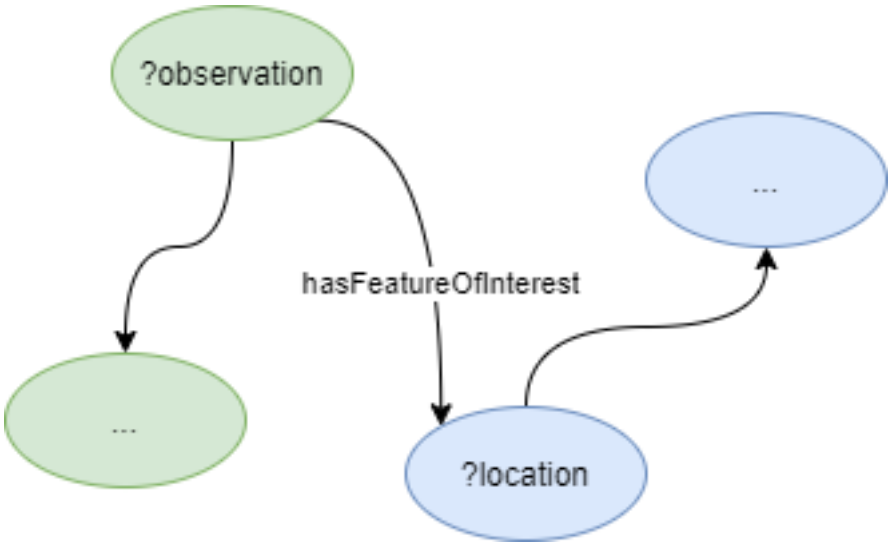

Figura 3.3: Esempio di linked data

E' possibile notare un'applicazione di questo concetto anche nel mio progetto in cui viene dedicato un grafo per le observation:

http://wot.arces.unibo.it/observation

ed un grafo per le features of interest;

http://wot.arces.unibo.it/context

Quando viene inserita una nuova observation, una nuova tripla viene generata e crea una connessione tra le due informazioni, concretizzando il concetto di linked data (Figura 3.3).

3.1.3 Forced Bindings

In SPARQL le operazioni e le sottoscrizioni possono essere modificate a run-time. Per questo motivo vengono usate le Forced Bindings. In generale, il concetto di "binding" viene così definito dalla documentazione ufficiale di W3 [W321] nel paragrafo 1.2.3 - "Result Descriptions" :

A 'binding' is a pair (variable, RDF term). In this result set, there are three variables: x, y and z (shown as column headers). Each solution is shown as one row in the body of the table. Here, there is a single solution, in which variable x is bound to "Alice", variable y is bound to ¡http://example/a¿, and variable z is not bound to an RDF term. Variables are not required to be bound in a solution.

In modo più specifico, le Forced Bindings danno quindi la possibilità allo sviluppatore di modificare il valore di un campo richiesto anche durante la sua esecuzione. Solo alcuni oggetti possono essere specificati con le Forced Bindings;

- URI
- bnode
- literal

Una tipizzazione errata della variabile specificata porta ad una gestione dell'errore che viene restituito.

Per richiedere una Forcer Binding, è necessario generare un oggetto di tipo JSON che contiene tutti i campi che si vogliono modificare. E' buona norma definire un valore default del dato, utilizzando la keyword 'value'. Inoltre viene data la possibilità allo sviluppatore di specificare il tipo di variabile che si sta passando come parametro all'operazione di update utilizzando la keyword 'datatype'.

Durante lo sviluppo del software, ho trovato utile sfruttare questa funzionalità offerta da SPARQL. In figura 3.4 riporto un esempio di Forced Binding:

```
"forcedBindings": {
        "observation": {
                "type": "uri",
                "value": "http://a.uri/ObservationXYZ"
        },
        "comment": {
                "type": "literal",
                "value": "This is an observation"
        },
        "label": {
                "type": "literal",
                "value": "The observation XYZ"
        },
        "location": {
                "type": "uri",
                "value": "http://a.uri/Mars"
        },
        "unit": {
                "type": "uri",
                "value": "unit:DegreeCelsius"
        }
}
```

Figura 3.4: Esempio di Forced Bindings applicato al mio lavoro di tesi

L'esempio riportato in figura 3.4 dimostra che, attraverso il Forced Binding, ho specificato cinque diversi campi che verranno modificati a run-time, quando richiesto dal programma.

Le observation, la location ed il tipo di unità di misura vengono definiti attraverso un URI. Più in dettaglio, il campo "unit" è rappresentato attraverso un'ontologia ben definita; *unit:DegreeCelsius*. Dalla figura 3.2 "Ontologie usate dal SEPA", è possibile osservare che unit è un binding al termine RDF "http://qudt.org/vocab/unit". Il valore default completo del campo "unit" è dunque un URI con questa definizione:

http://qudt.org/vocab/unitDegreeCelsius

I restanti due campi "comment" e "label" gestiscono invece un tipo di dato "literal", quindi una qualsiasi stringa.

Capitolo 4

SWAMP

4.1 Introduzione

SWAMP (Smart WAter Management Platform) [SWA21], è un progetto europeo H2020 a cui partecipa anche l'Università di Bologna, con il Centro di Ricerca Arces [Uni21c] ed i Dipartimenti DISTAL [Uni21b] e DICAM [Uni21a].

Figura 4.1: Logo SWAMP [SWA21]

E' possibile trovare una descrizione del progetto SWAMP nel sito ufficiale della commissione europea [Com17].

4.2 Obiettivi progetto

L'obiettivo principale del progetto è quello di rendere sempre più sostenibili le coltivazioni dei campi agricoli, tramite sistemi intelligenti di rilevamento (per esempio dell'umidità del terreno) che si basano sulla stima del fabbisogno idrico dell'appezzamento. Le coltivazioni, infatti sono tra i principali consumatori di acqua dolce al mondo. L'irrigazione dei terreni ha un impatto notevole sulla sostenibilità ambientale.

SWAMP garantisce un'interoperabilità tra diversi tipi di tecnologie e dispositivi hardware per superare queste sfide. Vengono dunque richiamati i concetti di dispositivi IoT, dispostivi autonomi e analisi avanzata dei dati. Attraverso lo studio delle informazioni raccolte dai sensori, si migliora la qualità dei dati presenti in piattaforma.

SWAMP mira a creare un sistema di irrigazione intelligente ad altissima precisione. L'ottimizzazione dell'irrigazione è legata alla valutazione dello stato dei canali di distribuzione dell'acqua e ai bisogni idrici reali del terreno. Viene quindi studiato il ciclo naturale dell'acqua, tenendo conto delle previsioni meteo e dell'esperienza degli esperti di settore.

Tra gli obiettivi del progetto vi è la rappresentazione digitale delle caratteristiche del terreno e delle colture; l'analisi dei Big-Data attraverso programmi di ottimizzazione suggerisce le irrigazioni necessarie.

4.3 Sistemi piloti: distribuzione Smart dell'acqua

Il progetto SWAMP vede lo sviluppo di quattro progetti pilota, due in Brasile, due in Europa (Italia e Spagna), ognuno dei quali ha un campo applicativo diverso. La zona coinvolta dal progetto italiano è quella amministrata dal Consorzio di Bonifica Emilia Centrale (CBEC) [Con21], responsabile delle irrigazioni e del drenaggio d'acqua di un'area di 1200 km2 [Bon21].

Il progetto pilota opera su due livelli per aumentare l'efficienza del sistema:

- **Livello agricoltore:** migliorare la stima della quantità di acqua necessaria alla coltura, attraverso un'infrastruttura IoT che integra informazioni del terreno, misure sul microclima locale e previsioni meteo.
- **Livello Consorzio:** rendere più efficienti i consumi d'acqua, ottimizzare le richieste di irrigazioni contemporanee e rendere più semplice amministrare la rete di distribuzione dell'acqua.

4.4 Eterogeneità dei dati

I dati messi a disposizione per l'applicativo che farà parte del progetto pilota italiano, sono: le richieste di irrigazione di tutti i campi evidenziati, l'ampiezza delle portate delle paratoie, i livelli dei canali, stato del terreno e osservazioni meteorologiche locali. I dati raccolti sono caricati sul web sotto forma di contenuti semantici. È possibile avere una visione a tutto campo sulla piattaforma web, WDA (Figura 4.2). Nel primo pannello a destra, sono mostrate le quantità di dati raccolti, ossia il numero totale di triple; lo storico che alla data del 10/07/2021, ha raggiunto quota ventiquattro milioni.

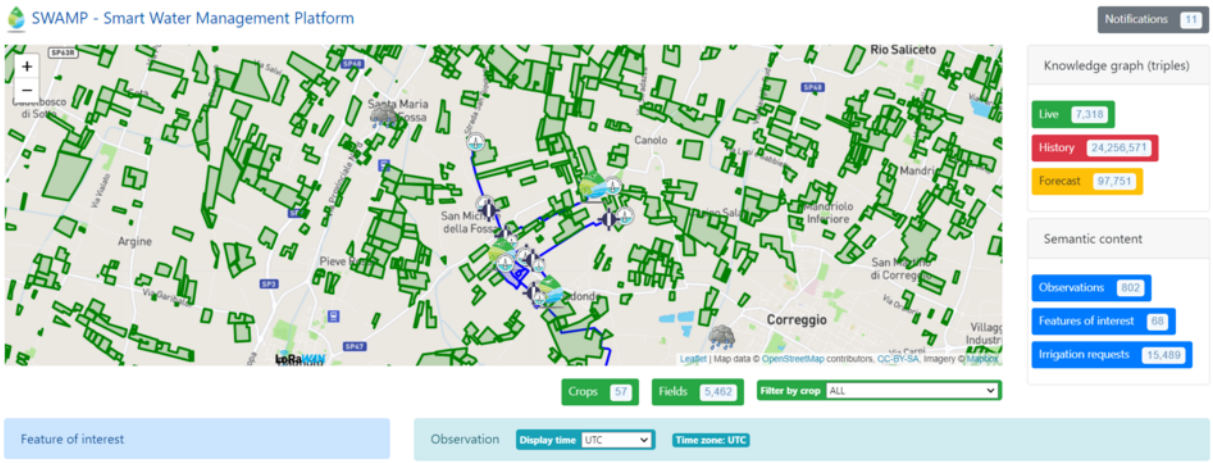

Figura 4.2: Applicazione Web WDA, Fonte [ARC21a]

Riassumendo dunque, SWAMP vuole sviluppare progetti di automatizzazione dei processi di monitoraggio e controllo della rete di distribuzione dell'acqua gestita dal CBEC. Tramite l'implementazione di tecnologie IoT e tecniche di analisi dei dati, intende garantire il miglioramento della rete di distribuzione dell'acqua e sviluppare contemporaneamente processi di tutela e difesa del territorio. La realizzazione di questi obiettivi potrà superare la vita del progetto stesso.

4.5 SEPA

Il SEPA (SPARQL Event Processing Architecture) è un'architettura decentralizzata basata sullo standard SPARQL 1.1 (Update, Query) che promuove e supporta la condivisione di informazioni e di interoperabilità tra molteplici layer del sistema (Figura 4.4). Il SEPA in un articolo [Luc+18] pubblicato nel MDPI Open Access Journal in data 20/04/2018 viene descritto in questo modo:

"This paper presents a decentralized Web-based architecture designed to support the development of distributed, dynamic, context-aware and interoperable services and applications. The architecture enables the detection and notification of changes over the Web of Data by means of a content-based publish-subscribe mechanism where the W3C SPARQL 1.1 Update and Query languages are fully supported and used respectively by publishers and subscribers. The architecture is built on top of the W3C SPARQL 1.1 Protocol and introduces the SPARQL 1.1 Secure Event protocol and the SPARQL 1.1 Subscribe Language as a means for conveying and expressing subscription requests and notifications"

"Questo documento presenta un'architettura decentralizzata basata sul Web progettata per supportare lo sviluppo di servizi e applicazioni distribuiti, dinamici, sensibili al contesto e interoperabili. L'architettura consente il rilevamento e la notifica delle modifiche sul Web dei dati mediante un meccanismo di pubblicazione-sottoscrizione basato sul contenuto in cui i linguaggi W3C SPARQL 1.1 Update e Query sono pienamente supportati e utilizzati rispettivamente da editori e sottoscrittori. L'architettura si basa sul protocollo W3C SPARQL 1.1 e introduce il protocollo SPARQL 1.1 Secure Event e il linguaggio di sottoscrizione SPARQL 1.1 come mezzo per trasmettere ed esprimere richieste e notifiche di sottoscrizione."

4.5.1 Architettura

La descrizione del SEPA specifica anche le caratteristiche di dettaglio come i meccanismi di pubblicazione e di sottoscrizione. Nella figura 4.3 viene presentata l'architettura della tecnologia utilizzata.

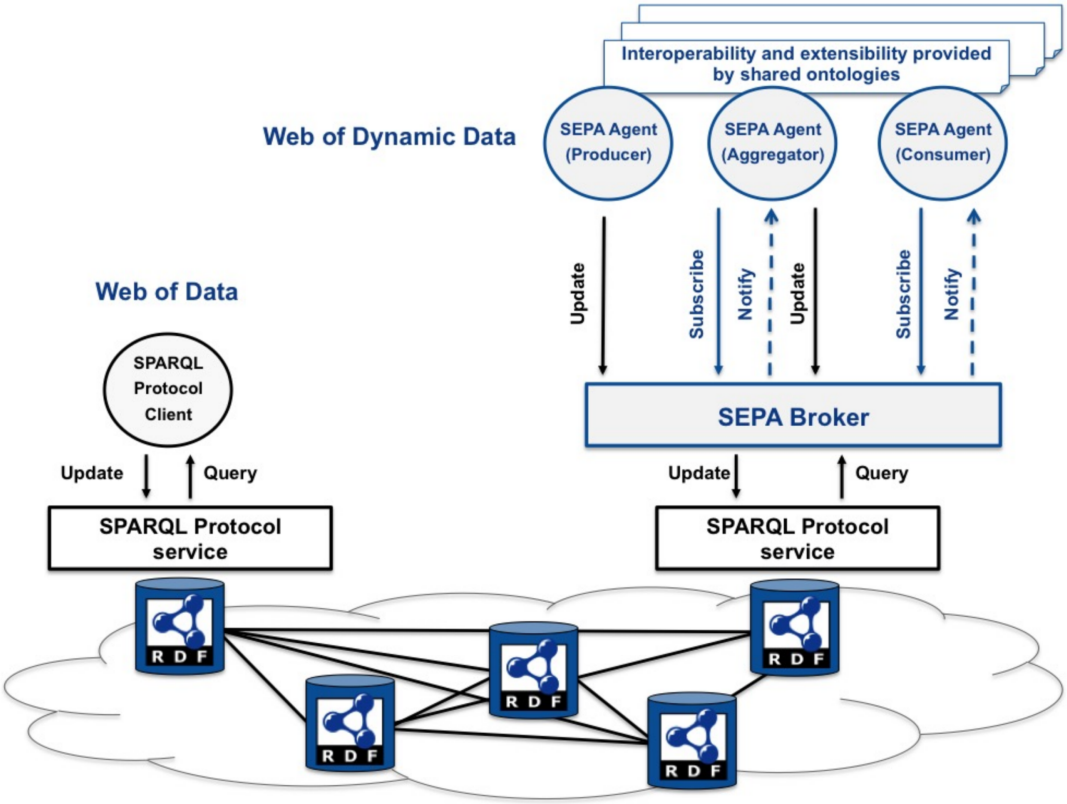

Figura 4.3: architettura SEPA, Fonte [Luc+18]

4.5.2 Entità presenti

Con il SEPA si ha la possibilità di analizzare ed utilizzare tre diverse entità;

- **Producer**: ha il compito di eseguire operazioni di update tramite SPARQL.
- **Consumer**: può effettuare una sottoscrizione data una query. Tramite quest'ultima, il consumer verrà notificato – attraverso notification handler – ogni volta che si modifica un valore restituito dalla query.
- **Aggregator**: questa entità unisce le due precedenti, dando quindi la possibilità di effettuare sia operazioni di update(s), sia operazioni di subscribe(s) (sottoscrizione).

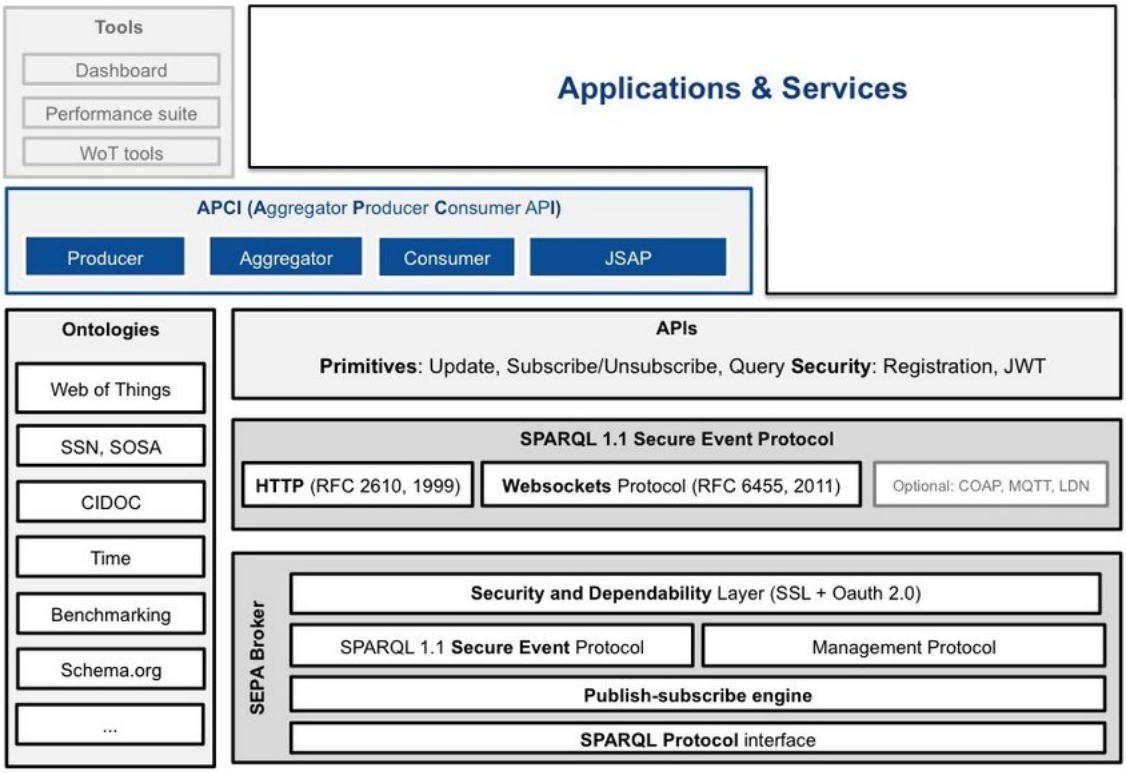

Figura 4.4: APCI - Entità presenti nel SEPA, Fonte [Luc+18]

Ogni entità presente nel sistema comunica con il broker del SEPA (Figura 4.4). Quest'ultimo sfrutta una base di dati Virtuoso (Virtuoso Universal Server).
Virtuoso Universal Server è un ibrido di middleware e motore di database che combina le funzionalità di un tradizionale sistema di gestione di database relazionale, database relazionale di oggetti, database virtuale, RDF, XML, free-text, server di applicazioni web e file server in un unico sistema. Questo si basa sugli HyperLinks, utilizzati come super chiavi del database (URI). Il nucleo di Virtuoso utilizza il linguaggio di SPARQL.

Capitolo 5

Sistema di notifica autonomo

5.1 Il programma

Con il mio progetto di tesi ho creato un sistema di notifica dei dati raccolti dai pluviometri dislocati nel comune di Bologna. L'obiettivo di questo software è quello di prevedere rischi idrogeologici in alcune aree test del comune. Infatti, il software notifica la quantità di pioggia che cade nelle aree di studio, classificandole con 5 differenti livelli di rischio (Capitolo 5.2 - "Analisi").

Queste informazioni, integrate in SWAMP, potranno essere alla base di una migliore comprensione dei processi climatici ed ottimizare l'irrigazione dei campi nei dintorni del comune di Bologna.

Il programma prende in considerazione tutte le entità che compongono il SEPA; produttore, consumatore ed aggregatore. Utilizzando queste figure all'interno del sistema è possibile rendere completamente autonoma l'applicazione. Nel prossimo capitolo illustrerò come, il software, andrà ad interagire con i dati presenti nel grafo su cui si basa il SEPA.

Nella Figura 5.1 riporto lo schema ontologico utilizzato nella progettazione del software. Il grafico è suddiviso nelle due differenti observation che caratterizzano il progetto. Prendendo ad esempio, in considerazione il pluviometro Correggio, il processo si sviluppa con;

- l'observation relativa al pluviometro ed al suo valore (in mm), < http://swampproject.org/observation/cbec/013-R24H >
- l'observation relativa al sistema di notifica, con il suo corrispondente waterLevel, < http://wot.arces.unibo.it/monitor #PluviometroCorreggio >

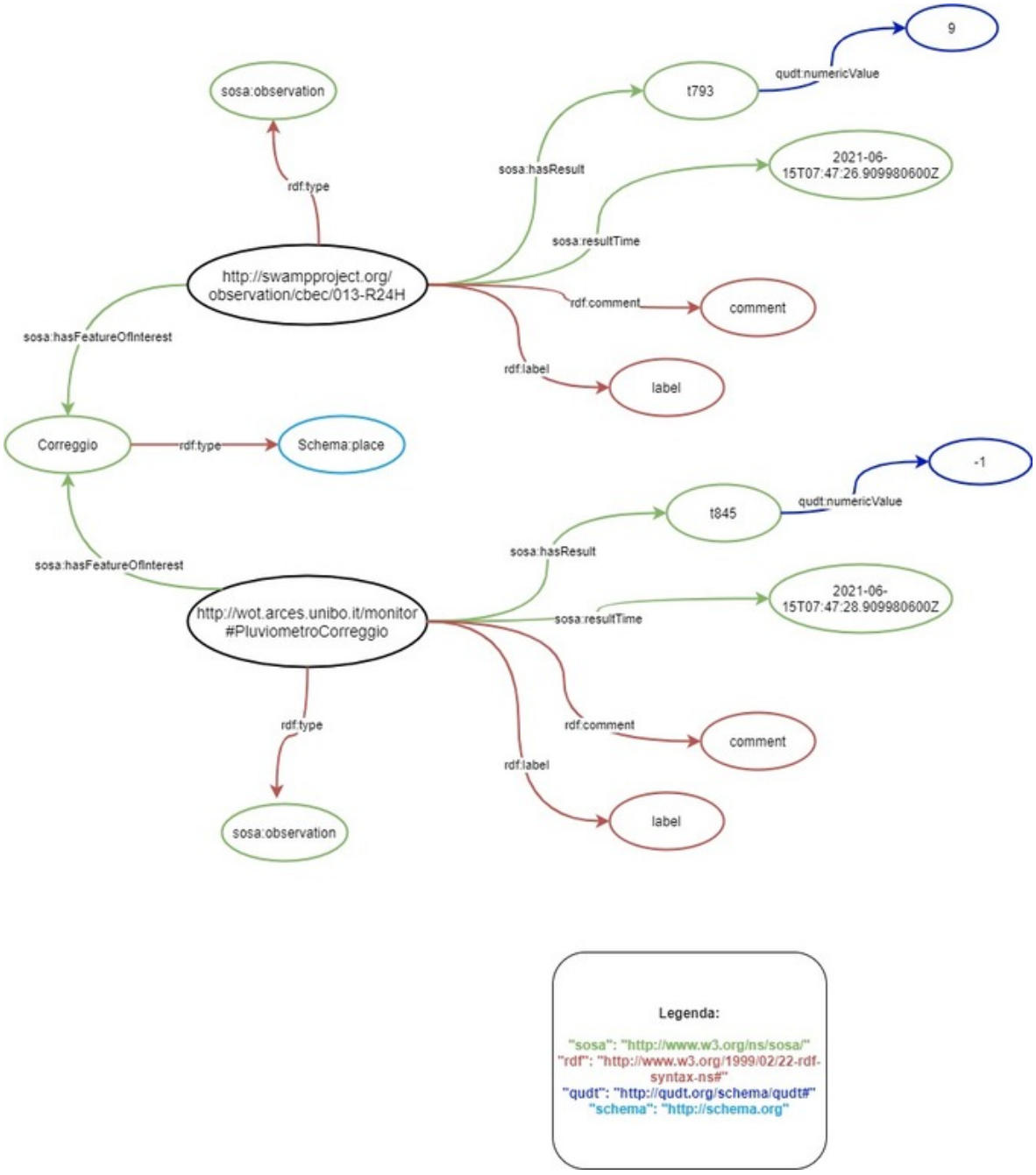

Figura 5.1: Esempio processo ontologico, con legenda utilizzato per il pluviometro di Correggio

Nella figura 5.1 è possibile osservare che il pluviometro di Correggio ha raccolto nell'ultima mezz'ora, 9 mm di acqua. Questo dato fa riferimento alla generazione di dati

casuali illustrati nella figura 5.7.

Inoltre, l'observation relativa al sistema di notifica, segnala un waterLevel -1. Questo dato viene riportato nei prossimi paragrafi e si tratta della gestione di un'eccezione del sistema intelligente per la prima mezz'ora. In questo caso, non si hanno valori di confronto della mezz'ora precedente e non è possibile determinare un range del waterLevel. Nei prossimi paragrafi, questo punto verrà analizzato più in dettaglio.

Il software è pensato per essere completamente autonomo dal sistema che lo ospiterà. Per questo motivo, ho creato delle entità 'produttori' che hanno il compito di uniformare il database che utilizzo. Così facendo si mira all'ottimizzazione del servizio che viene offerto.

Definita l'ontologia utilizzata nel programma, nella Figura 5.2 presento lo schema di comunicazione dell'applicazione.

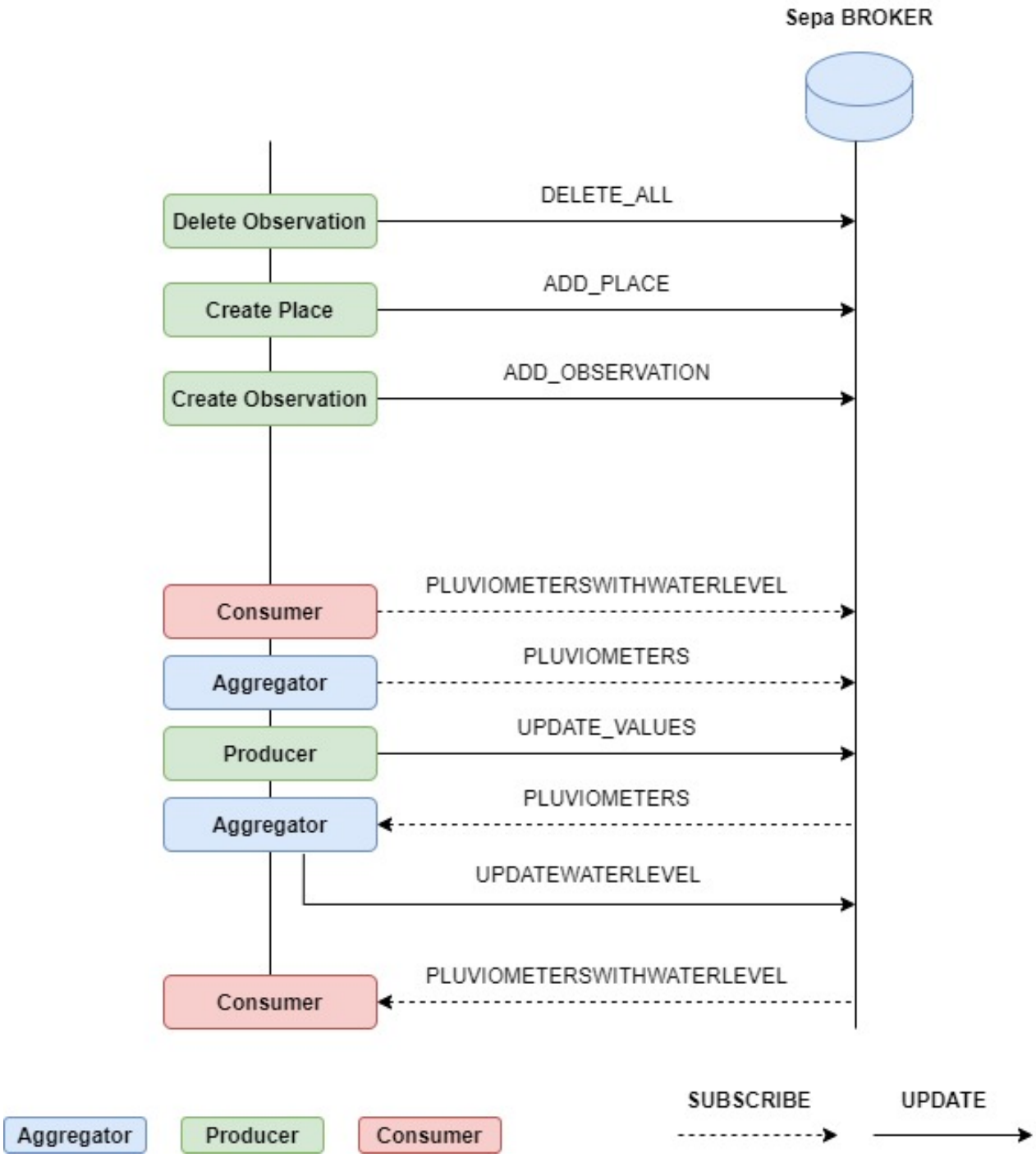

Figura 5.2: Diagramma di comunicazione software

Come evidente dalla lettura della Figura 5.2 il programma è stato suddiviso in due principali fasi:

1. Preparazione Database

   (a) **Delete observation:** Ogni volta che il programma viene mandato in esecuzione, viene creato un produttore che ha il compito di eseguire una query di update per cancellare tutte le triple presenti in tutti i grafi del database a grafo locale (Figura 5.3). **NB:** Questa è un'operazione molto rischiosa, è necessario procedere con molta cautela nella sua esecuzione. In questo caso non viene selezionato nessun grafo, quindi vengono cancellate tutte le triple con scheletro ?soggetto ?predicato ?oggetto presenti in **tutti i grafi** del database.

```
"DELETE_ALL": {
    "sparql": "DELETE WHERE {?s ?p ?o}"
},
```

Figura 5.3: Update cancellazione triple

   (b) **Create place:** Istanziazione di un produttore che crea le features of interest le quali andranno ad collegare i tre pluviometri ai loro valori corrispondenti del sistema di notifica (Figura 5.18, Capitolo 5.3 - "Inserimento Observation").

   Vengono inserite triple nel grafo/context che indicano il nome (literal) e le coordinate (latitudine, longitudine) della feature of interest (literal, literal). La procedura per l'instanziazione di questa nuova observation è consultabile nella figura 5.4.

   (c) **Create observation:** Creazione di triple come raffigurato nella figura 5.4. L'observation verrà poi collegata al suo valore di riferimento del sistema di notifica tramite la feature of interest.

```
"ADD_OBSERVATION": {
    "sparql": "INSERT {GRAPH <http://wot.arces.unibo.it/observation/2021>
                        {?observation rdf:type sosa:Observation ; rdfs:label ?label ;
                         rdfs:comment ?comment ; sosa:hasFeatureOfInterest ?location ;
                          sosa:hasResult _:quantity . _:quantity rdf:type qudt:QuantityValue ;
                           qudt:unit ?unit ; qudt:numericValue 'NaN'}} WHERE {}",
    "forcedBindings": {
        "observation": {
            "type": "uri",
            "value": "http://a.uri/ObservationXYZ"
        },
        "comment": {
            "type": "literal",
            "value": "This is an observation"
        },
        "label": {
            "type": "literal",
            "value": "The observation XYZ"
        },
        "location": {
            "type": "uri",
            "value": "http://a.uri/Mars"
        },
        "unit": {
            "type": "uri",
            "value": "unit:DegreeCelsius"
        }
    }
},
```

Figura 5.4: Istanziazione nuova observation

2. Esecuzione software ed analisi dati

   (a) **Consumer:** Istanziazione consumer che si sottoscrive alla query 'PluviometerWithWaterLevel' (Figura 5.5).

```
"PLUVIOMETERSWITHWATERLEVEL": {
    "sparql": "SELECT ?observation ?timestamp ?waterLevel
               FROM <http://wot.arces.unibo.it/observation/2021>
               WHERE {VALUES ?observation {<http://wot.arces.unibo.it/waterLevelCorreggio>
               <http://wot.arces.unibo.it/waterLevelSantaMaria> <http://wot.arces.unibo.it/waterLevelRotte>}
               ?observation rdf:type sosa:Observation ; sosa:hasResult ?quantity ; sosa:resultTime ?timestamp .
               ?quantity qudt:numericValue ?waterLevel }"
}
```

Figura 5.5: Query livelli pluviometrici

Attraverso la sottoscrizione, si notifica il consumer ad ogni cambiamento del livello d'acqua registrato nei vari pluviometri (in questo caso < http://swamp-project.org/observation/cbec/013-R24H >,
< http://swamp-project.org/observation/cbec/005-R24H >,
< http://swamp-project.org/observation/cbec/003-R24H >).

(b) **Aggregator:** Successivamente, ho preso in considerazione l'entità di aggregator, che effettua una sottoscrizione alla query 'Pluviometer' (Figura 5.6).

```
"PLUVIOMETERS": {
    "sparql": "SELECT ?observation ?value ?timestamp
               FROM <http://swamp-project.org/observation/cbec>
               WHERE { VALUES ?observation {<http://swamp-project.org/observation/cbec/013-R24H>
               <http://swamp-project.org/observation/cbec/005-R24H> <http://swamp-project.org/observation/cbec/003-R24H>}
               ?observation rdf:type sosa:Observation ; sosa:hasResult ?quantity .
               OPTIONAL{ ?observation sosa:resultTime ?timestamp } . ?quantity qudt:numericValue ?value}"
},
```

Figura 5.6: Query valori pluviometri

Attraverso la sottoscrizione, il SEPA si occupa di notificare l'aggregatore ad ogni cambiamento del valore registrato nei diversi pluviometri prima citati.

(c) **Producer:** Quando le due sottoscrizioni vengono effettuate correttamente, viene introdotta la figura del producer. Quest'ultimo ha il compito di generare in modo casuale i valori (range 0-10) tramite l'utilizzo della libreria Math.Rand() di Java, come raffigurato nell'immagine 5.8. Attraverso l'operazione di update 'UpdateValue' viene associato il nuovo dato all'observation presa in considerazione.

```
"UPDATEVALUE": {
"sparql": "WITH <http://swamp-project.org/observation/cbec>
           DELETE { ?quantity qudt:numericValue ?oldValue . ?observation sosa:resultTime ?oldTime }
           INSERT { ?quantity qudt:numericValue ?value . ?observation sosa:resultTime ?timestamp }
           WHERE {BIND(now() AS ?timestamp) . ?observation rdf:type sosa:Observation ; sosa:hasResult ?quantity .
           ?quantity qudt:numericValue ?oldValue . OPTIONAL { ?observation sosa:resultTime ?oldTime}}"
    "forcedBindings": {
        "observation": {
            "type": "uri",
            "value": "http://prova"
        },
        "value": {
            "type": "literal",
            "datatype": "xsd:integer",
            "value": "0"
        }
    },
},
```

Figura 5.7: Update valori pluviometro

Nella Figura 5.7 si riporta l'update che richiede i forced bindings. Le due informazioni richieste sono l'id URI dell'observation e il valore, specificato come intero tramite l'ontologia xml (http://www.w3.org/2001/XMLSchema).

Questi dati vengono inseriti in modo dinamico dal produttore.

Nel file main del progetto viene quindi gestita la generazione casuale di numeri interi.

```
do {
    try {
        producer.sendValue("http://swamp-project.org/observation/cbec/013-R24H", Integer.toString(rand.nextInt(10)));
        producer.sendValue("http://swamp-project.org/observation/cbec/003-R24H", Integer.toString(rand.nextInt(10)));
        producer.sendValue("http://swamp-project.org/observation/cbec/005-R24H", Integer.toString(rand.nextInt(10)));
        System.out.println("\n");
        Thread.sleep(10000);
    } catch (InterruptedException e1) {
        return;
    }
    continue;
} while (true);
```

Figura 5.8: Generazione nuovi valori relativi ai pluviometri

Appena il database è pronto per ospitare nuovi dati dal programma, il produttore invia informazioni relative alle tre observation correlate al valore del pluviometro. Queste operazioni di update vengono fatte all'interno di un ciclo infinito con un intervallo di 10 secondi tra un invio ed un altro.

Ad ogni generazione del dato, viene richiamato il metodo presente nella classe 'ProducerNotifica.java', 'sendValue'. Questa procedura richiede due parametri in ingresso, un'observation (URI) ed un valore (literal), come illustrato nella Figura 5.9.

```java
public boolean sendValue(String valorePluvURI, String value) {
    System.out.println("[PRODUCER] - Sending value: " + value + " to:" + valorePluvURI);

    int retry = 5;

    boolean ret = false;
    while (!ret && retry > 0) {
        try {
            this.setUpdateBindingValue("observation", new RDFTermURI(valorePluvURI));
            this.setUpdateBindingValue("value", new RDFTermLiteral(value));

            ret = update().isUpdateResponse();
        } catch (SEPASecurityException | SEPAProtocolException | SEPAPropertiesException
                | SEPABindingsException e) {
            logger.error(e.getMessage());
            ret = false;
        }
        retry--;
    }

    if (!ret) logger.error("[PRODUCER] UPDATE FAILED");

    return ret;
}
```

Figura 5.9: Invio valori casuali dati producer

L'update viene tentata 5 volte, se in nessuno di questi tentativi l'operazione viene svolta in modo corretto, viene stampato un messaggio di errore.

(d) **Aggregator:** Una volta effettuata correttamente l'operazione di update, l'aggregator verrà notificato della modifica dei valori richiesti dalla query riportata nella figura 5.6. Appena si riceve il nuovo avviso, viene calcolato il nuovo livello di acqua presente nel pluviometro.

Come è possibile vedere dal codice sorgente del software [Pis21], il programma si basa su una struttura base ArrayList < > di tipo Pluviometer. Quest'ultimo è un oggetto creato appositamente per definire un elenco di variabili necessarie che vanno a descrivere e ad identificare il singolo pluviometro (Figura 5.10). Ogni elemento presente nella struttura dati è così organizzato:

```java
private String observation;
private double value;
private String timeStamp;
private double oldValue;
```

Figura 5.10: Struttura oggetto pluviometro

E' possibile dunque distinguere i vari pluviometri presenti nell'arrayList attraverso il campo 'observation', che corrisponde ad un Uniform Resource Identifier (URI) [w321].
Il SEPA mette a disposizione il metodo 'onAddedResult()', che permette all'aggregatore di essere notificato, dopo la sottoscrizione, dagli eventuali cambiamenti. Ad ogni nuovo valore ricevuto è necessario controllare se il pluviometro per il quale è stata inserita una nuova informazione è già esistente nella struttura dati locale (Figura 5.11).

```
for(int i=0; i<pluviometers.size(); i++) {
    if(pluviometers.get(i).getObservation().equals(bindings.getValue("observation")))
        indexToReach = i;
}

if(indexToReach!=-1) {
    pluviometers.get(indexToReach).updateValues(Double.parseDouble(bindings.getValue("value")),
                                                bindings.getValue("timestamp"));
    waterLevel = pluviometers.get(indexToReach).getNewWaterValue();
}else {
    p1 = new Pluviometer(bindings.getValue("observation"),
             Double.parseDouble(bindings.getValue("value")),
             bindings.getValue("timestamp"));
    pluviometers.add(p1);
    waterLevel = pluviometers.get(pluviometers.indexOf(p1)).getNewWaterValue();
}
```

Figura 5.11: Gestione struttura dati ArrayList

Impostando in fase di costruzione della classe il valore della variabile indexToReach pari a -1, è possibile controllare se il valore è pre-esistente. Se questo ha un valore pari a -1, allora è necessario creare un nuovo oggetto Pluviometer ed inserirlo nell'arrayList pluviometers < > ed ottenere un nuovo livello di acqua presente nel pluviometro. Anche Il campo 'oldValue' del nuovo oggetto acquisirà valore -1, per evitare conflitti con i nuovi valori.

In fase di analisi (Capitolo 5.2 - "Analisi") è stato definito che è necessario avere un range di 60 minuti per definire accuratamente la fascia di allarme. Ma, come menzionato precedentemente, viene prevista l'acquisizione delle informazioni con cadenza semi-oraria, di 30 minuti. Per questo motivo, vengono definiti due campi distinti oldValue e value. Il primo, corrisponde alla mezz'ora precedente, il secondo ai 30 minuti correnti alla sua valutazione. Una volta ottenute queste informazioni, è immediato comprendere a pieno la funzione invocata nella figura 5.12, 'getNewWaterLevel'.

```
public int getNewWaterValue() {
    if(oldValue!=0) hourValue = value + oldValue; else return -1;

    if(hourValue<2) return 0;
    else if(hourValue>=2 && hourValue<4) return 1;
    else if(hourValue>=4 && hourValue<6) return 2;
    else if(hourValue>=6 && hourValue<10) return 3;
    else if(hourValue>=10 && hourValue<30) return 4;
    else if(hourValue>=30) return 5;

    return 0;
}
```

Figura 5.12: Generazione nuovo livello pluviometrico

Ricevuto il primo dato, non sarà possibile calcolare correttamente la fascia del livello pluviometrico, per questo motivo è stato assegnato un valore di default -1. Una corretta informazione sarà consultabile dopo la prima ora di applicazione del software.

Prendendo in considerazione il caso in cui il pluviometro sia già registrato nella struttura dati, viene invocato un altro metodo che ha il compito di aggiornare i valori dell'oggetto preso in considerazione (Figura 5.13), conoscendo la sua posizione dell'arrayList attraverso la variabile indexToReach.

```
public void updateValues(double newValue, String newTimeStamp) {
    this.oldValue = value;
    this.value = newValue;
    this.timeStamp = newTimeStamp;
}
```

Figura 5.13: Aggiornamento valori oggetto già esistente nella struttura dati

Appena conclusa l'assegnazione del livello pluviometrico, l'aggregator avrà il compito di effettuare l'update relativa a questa nuova informazione in suo possesso. L'operazione è la seguente (Figura 5.14);

```
"UPDATEWATERLEVEL": {
"sparql": "WITH <http://wot.arces.unibo.it/observation/2021>
           DELETE { ?quantity qudt:numericValue ?oldWaterLevel . ?observation sosa:resultTime ?oldTime }
           INSERT { ?quantity qudt:numericValue ?waterLevel . ?observation sosa:resultTime ?timestamp }
           WHERE {BIND(now() AS ?timestamp) . ?observation rdf:type sosa:Observation ; sosa:hasResult ?quantity .
           OPTIONAL { ?observation sosa:resultTime ?oldTime . ?quantity qudt:numericValue ?oldWaterLevel}}"

    "forcedBindings": {
        "observation": {
            "type": "uri",
            "value": "http://prova"
        },
        "waterLevel": {
            "type": "literal",
            "datatype": "xsd:integer",
            "value": "0"
        }
    },
}
```

Figura 5.14: Update del livello pluviometrico

Tramite questa update, l'aggregator assegna un nuovo valore (literal) ad un' observation (URI). Se questa è esistente, analogamente all'update precedentemente menzionata, il valore viene sostituito dalla nuova informazione generata.

E' necessario sottolineare che il water level può essere assegnato alle observation presenti nel grafo < http://wot.arces.unibo.it/observation/2021 >. Queste triple infatti, come indicato nell'immagine 5.14, contengono il valore di waterLevel con il predicato qudt:numericValue.

Per creare questa stretta relazione tra il valore ed il pluviometro nel grafo utilizziamo la feature of interest. Nella logica del programma invece, sfruttiamo una struttura dati di tipo HashMap (Figura 5.15). Quest'ultima è stata utilizzata per generare un dizionario, quindi mettere in corrispondenza una data chiave con un dato valore. In questo caso specifico, la chiave è l'URI dell'observation relativa al valore espresso dal pluviometro in mm (grafo < http://swamp-project.org/observation/cbec >), ed il valore è l'URI dell'observation relativa al sistema di notifica
(grafo < http://wot.arces.unibo.it/observation/2021 >).

```
map = new HashMap<>();
map.put("http://swamp-project.org/observation/cbec/013-R24H",
        "http://wot.arces.unibo.it/waterLevelCorreggio"); //Correggio
map.put("http://swamp-project.org/observation/cbec/003-R24H",
        "http://wot.arces.unibo.it/waterLevelSantaMaria"); //Santa Maria
map.put("http://swamp-project.org/observation/cbec/005-R24H",
        "http://wot.arces.unibo.it/waterLevelRotte"); //Rotte
```

Figura 5.15: Introduzione HashMap

Grazie all'HashMap è possibile creare questa referenza nella logica del programma. Inoltre, sfruttando la libreria java.util.HashMap, viene utilizzato il metodo put() per inserire un valore nella mappa, ed il valore get() per recuperare il valore data una determinata chiave.

5.2 Analisi

Il pluviometro è lo strumento comunemente utilizzato in meteorologia per misurare la quantità di pioggia caduta in un dato intervallo di tempo (orario, giornaliero, settimanale). L'unità di misura è rappresentata dai millimetri di accumulo (mm), cioè l'altezza pluviometrica a cui corrispondono altrettanti litri d'acqua piovana su una superficie di un metro quadrato. I millimetri di pioggia caduti in un'ora definiscono quella che viene chiamata dai meteorologi, intensità della pioggia, che possiamo suddividere in [PE08];

- Pioggia debole (1 – 2 mm/h)
- Pioggia leggera (2 – 4 mm/h)
- Pioggia moderata (4 – 6 mm/h)
- Pioggia forte (> 6 mm/h)
- Rovescio (> 10 mm/h)
- Nubifragio (> 30 mm/h)

Analizzando queste informazioni, ho associato il range di millimetri raccolti da un pluviometro, ad un valore di tipo numerico, per facilitare futuri studi statistici. Attraverso l' update in SPARQL, i dati vengono così classificati nel grafo observation della piattaforma SWAMP.

- Pioggia debole (1 − 2 mm/h) → 0
- Pioggia leggera (2 – 4 mm/h) → 1
- Pioggia moderata (4 – 6 mm/h) → 2
- Pioggia forte (> 6 mm/h) → 3
- Rovescio (> 10 mm/h) → 4
- Nubifragio (> 30 mm/h) → 5

I dati raccolti provengono da tre pluviometri dislocati in differenti punti del comune di Bologna, così come illustrato nella figura 5.16.

Le informazioni relative ai mm/m^2 del pluviometro vengono inviate all'entità aggregatore del SEPA ogni 30 minuti. Vedremo successivamente come viene gestita l'eccezione

Nome	Nodo Grafo	Latitudine	Longitudine
Pluviometro Santa Maria	nodeID://b14792200	44,8013	10,6913
Rotte (pluviometro)	nodeID://b15167651	44,7510	10,6630
Pluviometro Correggio	nodeID://b14792236	44,7677	10,7636

Figura 5.16: Pluviometri in esame

generata nella prima mezz'ora, non avendo i dati necessari per far riferimento alle informazioni prime riportate (basate su un periodo di 60 minuti).

Per iniziare ad interagire con il SEPA e con il sistema che lo circonda, ho utilizzato il localhost come endpoint per provare ed accertarmi del corretto funzionamento della query. Inoltre, ho utilizzato la dashboard SEPA [ARC20], interfaccia grafica utilizzata per interagire con il database (Virtuoso) tramite query ed update. Viene data anche la possibilità di visualizzare l'ontologia utilizzata (Figura 5.1) ed i dati presenti nella base di dati.

5.3 Inserimento Observation

In fase di analisi ho evidenziato la differenza tra l'observation relativa al valore di mm/h di pioggia caduta nel pluviometro e l'observation correlata al sistema di notifica che si voleva implementare. I due record vengono generati dalla stessa update (ADD_OBSERVATION), ma si distinguono dal grafo a cui fanno riferimento.

L'observation relativa al valore di mm/h di pioggia caduta nel pluviometro si trova nel grafo < http://swamp-project.org/observation/cbec >, mentre l'observation riguardante il valore del livello di acqua generato (Figura 5.12) viene aggiunta nel grafo < http://wot.arces.unibo.it/observation/2021 > .

Come è possibile osservare dalla lettura della figura 5.4 riportata nel capitolo 5.1 - 'Il programma', l'observation di un pluviometro è caratterizzato da;

- Una location (sosa:hasFeatureOfInterest)
- Una relazione di quantità (sosa:hasResult) che ha come valore di riferimento una quantità (qudt:QuantityValue)
- Un'unità di misura (qudt:unit)
- Una quantità (qudt:numeriValue) con valore di default NaN, per evitare inserimenti di dati non veritieri in piattaforma (come 0).
- Un commento, definito da un valore literal
- Una label, definito da un valore literal. Questa tripla è quella che viene stampata nell'interfaccia del progetto SWAMP, la WDA [SWA21].

Viene successivamente riportato un esempio con tre diverse observation;

- < http://swamp-project.org/observation/cbec/013-R24H >
- < http://swamp-project.org/observation/cbec/005-R24H >
- < http://swamp-project.org/observation/cbec/003-R24H >

Per ogni pluviometro vengono, dunque, generati valori casuali. Nel test riportato vengono inseriti; il valore 9 per la prima observation, il valore 1 per la seconda ed il valore 6 per la terza. Il risultato della seguente query (Figura 5.17) è:

SELECT ?observation ?value ?timestamp
FROM < http://swamp-project.org/observation/cbec >
WHERE VALUES ?observation {
< http://swamp-project.org/observation/cbec/013-R24H >
< http://swamp-project.org/observation/cbec/005-R24H >
< http://swamp-project.org/observation/cbec/003-R24H > }
?observation rdf:type sosa:Observation ; sosa:hasResult ?quantity .
OPTIONAL{ ?observation sosa:resultTime ?timestamp } . ?quantity qudt:numericValue ?value

observation	value	timestamp
http://swamp-project.org/observation/cbec/013-R24H	**9^^xsd:integer**	**2021-06-14T11:01:32.463932700Z^^xsd:dateTimeStamp**
http://swamp-project.org/observation/cbec/003-R24H	**1^^xsd:integer**	**2021-06-14T11:01:32.823482500Z^^xsd:dateTimeStamp**
http://swamp-project.org/observation/cbec/005-R24H	**6^^xsd:integer**	**2021-06-14T11:01:33.077736900Z^^xsd:dateTimeStamp**

Figura 5.17: Risultato della query corretta

E' necessario sottolineare che senza la tripla precedentemente menzionata, *?observation sosa:hasResult _:quantity*, il risultato della stessa query sarebbe errato. La stampa visualizzata da dashboard [ARC20] infatti sarebbe l'ultimo dato inserito per quel pluviometro n volte, dove n corrisponde al numero di aggiornamenti affettuati. Questo viene evitato grazie alla connessione tra l'observation ed il singolo risultato, che permettono allo sviluppatore di poter prendere in considerazione solo il valore specificato dalla tripla *?observation sosa:hasResult _:quantity.*

Un'altra update utilizzata è quella relativa alla creazione di una features of interest. Nella figura 5.18 illustro la procedura di update utilizzata per creare una nuova location (feature of interest), che verrà legata alle observation sopra descritte.

Una volta presentate queste due operazioni di update, è più intuitivo comprendere la Figura 3.3, Capitolo 3.1.2 - "SPARQLandLD", che illustra la stretta correlazione presente tra l'observation e la location.

```
"ADD_PLACE": {
    "sparql": "INSERT {GRAPH <http://wot.arces.unibo.it/context>
    {?place rdf:type schema:Place; schema:name ?name ;  schema:GeoCoordinates _:coordinate .
     _:coordinate schema:latitude ?lat ; schema:longitude ?lon}} WHERE {}",

    "forcedBindings": {
        "lat": {
            "datatype": "xsd:decimal",
            "type": "literal",
            "value": "44.489664"
        },
        "lon": {
            "datatype": "xsd:decimal",
            "type": "literal",
            "value": "11.357023"
        },
        "name": {
            "type": "literal",
            "value": "Mars"
        },
        "place": {
            "type": "uri",
            "value": "http://a.uri/Mars"
        }
    },
},
```

Figura 5.18: Inserimento feature of interest

5.3.1 Esempio

Utilizzando l'hashmap riportata, si propone, di seguito, un esempio pratico:

- Producer genera valore 6 per l'observation ⟨http://swampproject.org/ observation/cbec/013-R24H ⟩
- Aggregator viene notificato del cambiamento per l'observation ⟨http://swampproject.org/ observation/cbec/013-R24H ⟩
- Aggregator genera il valore -1 come waterLevel (primo valore, non si ha parametro di confronto)
- Aggregator effettua update per l'observation [map.get(⟨http://swampproject.org/ observation/cbec/013-R24H ⟩)], quindi ⟨http://wot.arces.unibo.it/monitor #PluviometroCorreggio ⟩, vedi figura 4.18.
- Consumer viene notificato della modifica del valore per l'observation ⟨http://wot.arces.unibo.it/monitor #PluviometroCorreggio ⟩
- Producer genera valore 3 per l'observation ⟨http://swampproject.org/ observation/cbec/013-R24H ⟩

- Aggregator viene notificato del cambiamento per l'observation ⟨http://swampproject.org/ observation/cbec/013-R24H ⟩
- Vista la presenza di due valori (confronto di un'intera ora), la somma che viene restituita è 9 (6+3), l'Aggregator genera dunque il dato 3 come waterLevel, presi in considerazione i range presentati nel capitolo 4.1 - Analisi [x⟩6 && x⟨10].
- Aggregator effettua update per [map.get(⟨http://swampproject.org/ observation/cbec/013-R24H ⟩)], quindi ⟨http://wot.arces.unibo.it/monitor #PluviometroCorreggio ⟩, vedi figura 4.18.
- Consumer viene notificato della modifica del valore per l'observation ⟨http://wot.arces.unibo.it/monitor #PluviometroCorreggio ⟩

5.4 Demo

E' possibile visualizzare una Demo del software attraverso questo link:
https://liveunibo-my.sharepoint.com/:v:/g/personal/lorenzo_pisano_studio_unibo_it/EXapdN_x2YFBo1evURaJxWMBDgFNb4ATVMf4vvdhskPtAA?e=dlx52L

NB: Il copia ed incolla tramite OpenLeaf non riporta il carattere '_'

5.5 Integrazione sistema di notifica nel progetto SEPA

Una volta conclusa la fase di testing del sistema autonomo, il prossimo obiettivo è quello di mandare in esecuzione online l'aggregatore che sviluppa il sistema di notifica descritto in modo dettagliato nel capitolo 5.2.

A partire dallo schema delle entità presenti nel sistema, si vuole distinguere quali di queste non sono necessarie dopo aver stabilito la connessione del software al server remoto e quali sono i cambiamenti che devono essere effettuati.

Prima di tutto viene modificato il fsile jsap, cambiando l'host e la porta di comunicazione del software.

```
"host": "localhost",
"oauth": {
    "enable": false
},
"sparql11protocol": {
    "protocol": "http",
    "port": 8000,
    "query": {
        "path": "/query",
        "method": "POST",
        "format": "JSON"
    },
    "update": {
        "path": "/update",
        "method": "POST",
        "format": "JSON"
    }
},
"sparql11seprotocol": {
    "protocol": "ws",
    "reconnect": true,
    "availableProtocols": {
        "ws": {
            "port": 9000,
            "path": "/subscribe"
        },
        "wss": {
            "port": 9443,
            "path": "/secure/subscribe"
        }
    }
},
```

Figura 5.19: Host e porta localhost

```
"host": "mml.arces.unibo.it",
"oauth": {
    "enable": false,
    "register": "https://localhost:8443/oauth/register",
    "tokenRequest": "https://localhost:8443/oauth/token"
},
"sparql11protocol": {
    "protocol": "http",
    "port": 8666,
    "query": {
        "path": "/query",
        "method": "POST",
        "format": "JSON"
    },
    "update": {
        "path": "/update",
        "method": "POST",
        "format": "JSON"
    }
},
"sparql11seprotocol": {
    "reconnect" : true,
    "protocol": "ws",
    "availableProtocols": {
        "ws": {
            "port": 9666,
            "path": "/subscribe"
        },
        "wss": {
            "port": 9443,
            "path": "/secure/subscribe"
        }
    }
},
```

Figura 5.20: Host e porta server remoto (mml)

Come è possibile osservare dalla lettura delle figure 5.19 e 5.20, viene modificato il valore del campo host da "localhost" ad "mml.arces.unibo.it" e tutte le porte di connessione al protocollo sparql11 per le operazioni di query ed update.

Grazie a quest'alterazione del file 'obersvation_pluviometer.jsap', è possibile interfacciare **in modo diretto** il software al server remoto mml. Viene verificato, attraverso dashboard, il collegamento al server mml (Figura 5.21).

Figura 5.21: Connessione a server MML tramite Dashboard

In questo caso la connessione è stata effettuata in modo corretto. La dashboard infatti, si sta interfacciando con il server remoto. Viene effettuato un test per contro prova. Quindi, viene mandata in esecuzione la query 'observation', che richiede, a diversi grafi, informazioni riguardanti observation presenti nel server remoto. La query (Figura 5.22) è così definita;

```
"OBSERVATIONS": {
    "sparql": "SELECT * FROM <http://wot.arces.unibo.it/unit>
                       FROM <http://swamp-project.org/observation/cbec>
                       FROM <http://wot.arces.unibo.it/observation>
                       FROM <http://wot.arces.unibo.it/context>
                       WHERE {?observation rdf:type sosa:Observation ; rdfs:label ?label ;
                       sosa:hasResult ?quantity ; sosa:hasFeatureOfInterest ?location .
                       ?location rdf:type schema:Place ; schema:name ?name ;
                       schema:GeoCoordinates ?coordinate . ?coordinate schema:latitude ?lat ;
                       schema:longitude ?long . ?quantity rdf:type qudt:QuantityValue ;
                       qudt:unit ?unit . OPTIONAL {?quantity qudt:numericValue ?value} .
                       OPTIONAL {?observation sosa:resultTime ?timestamp} .
                       ?location schema:name ?name . OPTIONAL{?unit qudt:symbol ?symbol}}"
},
```

Figura 5.22: Query observation

La query fa parte del gruppo di operazioni di default della connessione al server remoto. Vengono interrogati quattro grafi, ponendo delle condizioni tali da avere una stampa molto dettagliata delle triple che caratterizzano e identificano in modo univoco ogni observation presente nel sistema.

Se la stessa query venisse mandata in esecuzione interfacciandosi con il localhost, 9 observation sarebbero presenti (dopo aver mandato in esecuzione almeno una volta il sistema autonomo);

3 observation delle feature of interest, 3 observation dei pluviometri e 3 observation relativi ai pluviometri per il sistema di notifica.

Quando ci si interfaccia con il server mml, ci si aspetta un valore estremamente maggiore, visti i numeri a 8-9 cifre rispetto alle observation presenti nello storico della WDA di Swamp (figura 4.2 del sottocapitolo 4.4 - "Eterogeneità dei dati").

UPDATE --- QUERY Results: 802 (2481 ms) SUBSCRIBE

observation	label	quantity	location	name	coordinate	lat	long	unit	value	timestamp	symbol
http://swam...	**001-R24H (...**	nodeID://b1...	http://swam...	**Misure**	nodeID://b1...	**44.6987^^x...**	**10.6266^^x...**	http://qudt.or...	**0^^xsd:de...**	**2020-06-26...**	**mm**
http://wot.ar...	**Derivazion...**	nodeID://b2...	swamp:CP2	**Fosdondo ...**	nodeID://b1...	**44.7795262...**	**10.7232298...**	unit:Meter	**0.162286^^...**	**2021-06-15...**	**m^^xsd:st...**
http://wot.ar...	**Allo scaric...**	nodeID://b2...	http://wot.ar...	**Fosdondo ...**	nodeID://b1...	**44.7864^^x...**	**10.7429^^x...**	unit:Meter	**0.635612^^...**	**2021-06-15...**	**m^^xsd:st...**
http://wot.ar...	**A valle sba...**	nodeID://b2...	http://wot.ar...	**Fosdondo**	nodeID://b1...	**44.7801**	**10.7217**	unit:Meter	**0.884101^^...**	**2021-06-15...**	**m^^xsd:st...**
http://wot.ar...	**Precipitati...**	nodeID://b2...	swamp:Ferr...	**Ferrari An...**	nodeID://b9...	**44.790788^...**	**10.736074^...**	unit:Millimeter	**0^^xsd:int...**	**2021-06-14...**	**mm^^xsd:...**

Figura 5.23: Observation mml

In questo modo il server remoto mml restituisce secondo i parametri specificati nel WHERE della query (Figura 5.23) 802 risultati, con un tempo di elaborazione dei dati di 2481 ms (2,4 secondi).

Dopo questi due test, si ha la certezza che il software comunica, correttamente, **in modo diretto** con il server remoto.

A partire dal sistema autonomo precedentemente discusso (Capitolo 5.1), è necessario rimuovere alcune componenti che, una volta preso in considerazione il server remoto, risultano non indispensabili. Osservando la figura che descrive le entità presenti nel sistema (Figura 4.4) osserviamo che la preparazione del database locale, non è più necessaria. Questo perchè, è possibile, attraverso Dashboard, filtrare le observation restituite dal server mml per trovare le feature of interest ed i pluviometri (Figura 5.24).

Quest'ultimi infatti sono già presenti nel server mml, non è dunque necessario aggiungerli durante l'esecuzione del programma. Per trovare questa informazione è stata utilizzata la query 'Pluviometers', illustrata nella Figura 5.6, Capitolo 5.1 "Il programma".

UPDATE --- QUERY Results: 3 (150 ms) SUBSCRIBE

observation	value	timestamp
http://swamp-project.org/observation/cbec/003-R24H	0^^xsd:decimal	2021-06-16T09:58:08.376+0200^^xsd:dateTime
http://swamp-project.org/observation/cbec/005-R24H	0^^xsd:decimal	2021-06-16T09:58:08.376+0200^^xsd:dateTime
http://swamp-project.org/observation/cbec/013-R24H	0^^xsd:decimal	2021-06-16T09:58:08.376+0200^^xsd:dateTime

Figura 5.24: Pluviometri presenti nel server mml

A partire da questo dato, nella finestra 'Explorer' della dashboard è possibile accedere ai dati dell'observation, avendo un chiaro schema delle triple che la caratterizzano (Figura 5.25). Di seguito si riporta la definzione della struttura dell'observation < http://swamp-project.org/observation/cbec/013-R24H > presente nel grafo
< http://swamp-project.org/observation/cbec >.

Predicate	Object	Datatype
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/ns/sosa/Observation	URI
http://www.w3.org/2000/01/rdf-schema#label	013-R24H (Pioggia Cumulata Oggi)	
http://www.w3.org/2000/01/rdf-schema#comment	Retrieved from: https://dati.emiliacentrale.it/rest/misure/	
http://www.w3.org/ns/sosa/resultTime	2021-06-16T10:28:09.089+0200	http://www.w3.org/2001/XMLSchema#dateTime
http://www.w3.org/ns/sosa/hasResult	nodeID://b15435460	BNODE
http://www.w3.org/ns/sosa/hasFeatureOfInterest	http://wot.arces.unibo.it/monitor#PluviometroCorreggio	URI

Figura 5.25: Dettaglio struttura pluviometro Correggio

E' possibile osservare la presenza della feature of interest legata al pluviometro Correggio nell'ultima riga della definizione della sua struttura. La tripla è caratterizzata dall'ontologia 'sosa' (Figura 3.2, Capitolo 3.1.1 - "Web Semantico - Introduzione") e va a specificare l'observation relativa alla location. Viene dunque eliminato anche il secondo produttore presente nello schema di comunicazione del sistema autonomo (Figura 5.2 presente nel capitolo 5.1 - "Il programma"). Quest'ultimo si occupava della generazione delle observation relative alle features of interest attraverso l'operazione di update 'ADD_PLACE' (Figura 5.18, Capitolo 5.3 - "Inserimento observation").

Sempre facendo riferimento alla Figura che descrive le entità presenti nell'applicativo (Figura 5.2, Capitolo 5.1 - "Il programma"), nella sua prima parte, è necessario gestire la generazione delle observation utili a rendere il sistema di notifica effettivo, legandole alle differenti features of interest.

Queste observation però, devono essere create una sola volta, al fine di evitare doppi valori che andrebbero a generare inconsistenza all'interno del database a grafo. Per questo motivo, viene aggiunta nel file main.java una gestione di quest'eccezione tramite parametro di esecuzione. Vengono quindi gestiti due argomenti che possono determinare un comportamento differente nei primi istanti dell'esecuzione del software;

- Se il programmatore specifica un argomento pari a -init, il sistema, alla sua esecuzione, genera le observation necessarie per il sistema di notifica (Figura 5.26).

```
//definition of the new observation for water level
for (int i=0; i < args.length ; i++) {
    if (args[i].equals("-init")) {
        addingObservation(producerPreparation);
        break;
    }
}
```

Figura 5.26: Gestione parametro -init

- Se il programmatore specifica un argomento pari a -consumer, il sistema, alla sua esecuzione, instanzia un consumatore che si sottoscrive alla query 'PLUVIOMETERSWITHWATERLEVELS' (Figura 5.5, Capitolo 5.1 - "Il programma"). Questa scelta viene presa a livello di integrazione del sistema online perchè viene definito non necessario avere un consumatore online che effettua un controllo per la verifica del corretto funzionamento dell'operazione di update 'UPDATEWATERLEVEL' (Figura 5.6, Capitolo 5.1 - "Il programma").

```
//subscription of aggregator and consumer
ConsumerNotifica test = null;
for (int i=0; i < args.length ; i++) {
    if (args[i].equals("-consumer")) {
        test = consumer;
    }
}
```

Figura 5.27: Gestione parametro -consumer

Una volta che si hanno tutte le observation che il sistema di notifica necessita per il suo corretto funzionamento ed il software è in comunicazione con il server remoto, è necessario modulare le operazioni di query e di update.

In fase di analisi è stato definito che le observation relative ai pluviometri ed ai loro valori (mm) di pioggia raccolta sono consultabili attraverso il grafo < http://swamp-project.org/observation/cbec >, mentre le observation contenenti il water level indicante il livello di allarme si trovano nel grafo < http://wot.arces.unibo.it/observation/2021 >.

Con questi dati alla mano, è necessario modificare il grafo che le varie operazioni di update e di query interrogano.

- La query 'PLUVIOMETERS' deve interrogare il grafo < http://swamp-project.org/observation/cbec >, per effettuare un retrieve dei dati rispettivi ai pluviometri.
- La query 'PLUVIOMETERSWITHWATERLEVEL' deve interrogare il grafo < http://wot.arces.unibo.it/observation/2021 > per ottenere i dati rispettivi ai water level associati ai pluviometri.
- L'operazione di update 'UPDATEVALUES' non verrà più presa in considerazione. I dati di pioggia (mm) raccolta da ogni pluviometro vengono ricavati dai sensori presenti sul campo.
- L'operazione di update 'UPDATEWATERLEVELS' deve modificare i dati presenti nel grafo < http://wot.arces.unibo.it/observation/2021 >.

Effettuati questi cambiamenti, il sistema è pronto a comunicare con il server e ad associare in modo corretto ed efficiente il livello di allarme legato ai range descritti in fase di analisi. Questo sistema permetterà, dopo una fase accurata di testing, di prevedere i possibili rischi di inondazione oppure di un forte temporale con dati reali, con l'obiettivo di salvaguardare i raccolti agricoli in alcuni appezzamenti della provincia di Bologna.

Infine, viene ri-definita la figura che descrive le entità presenti nel sistema (Figura 5.2 - Capitolo 5.1 - "Il programma") dopo le modifiche prima specificate.

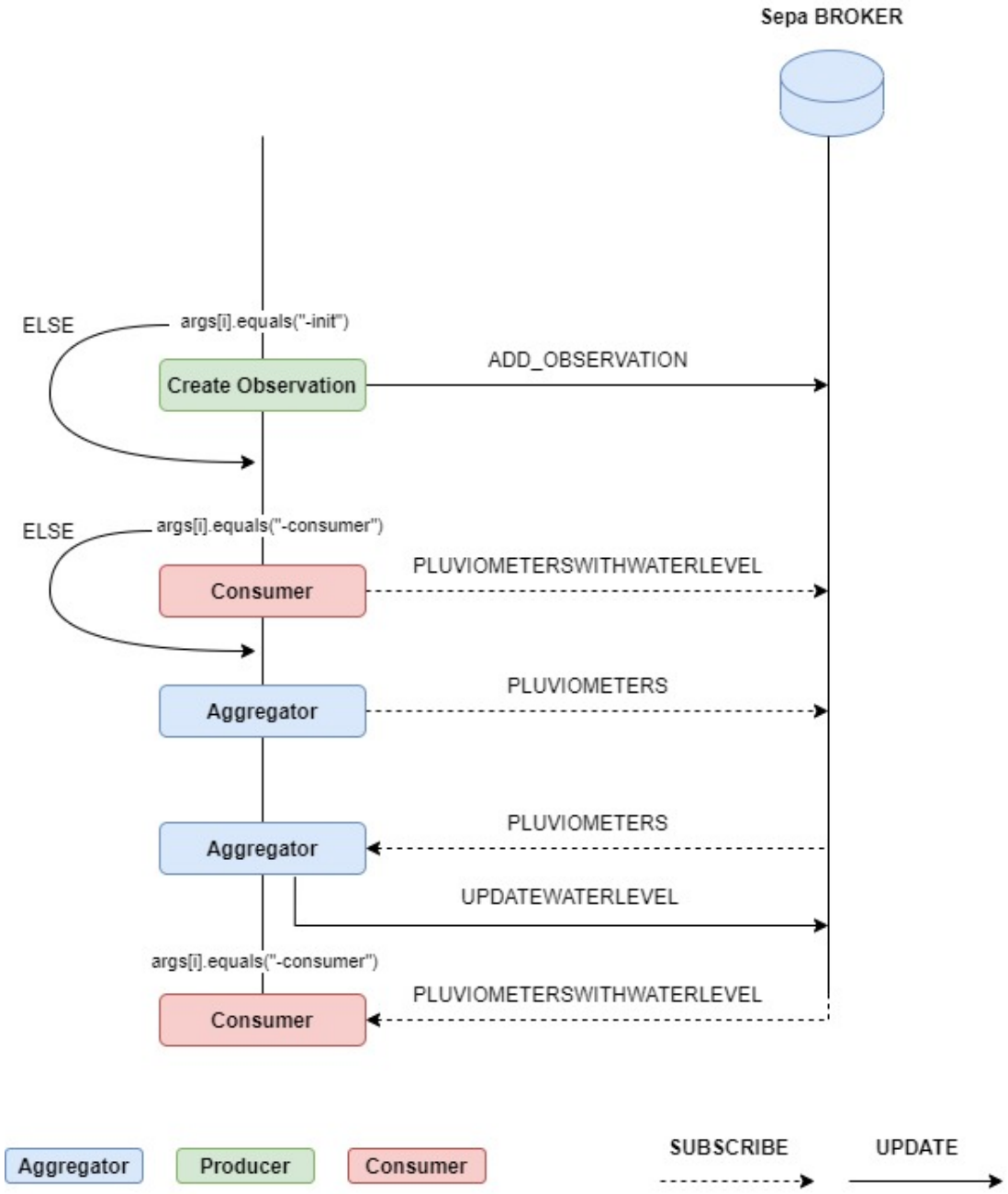

Figura 5.28: Diagramma di comunicazione programma online

Capitolo 6

Conclusioni e Sviluppi futuri

Completata la fase di progettazione e programmazione, è stato avviato, il 05/06/2021, un periodo di testing del software. L'aggregatore che si occupa di gestire il corretto funzionamento del sistema di notifica è attivo nella versione provvisoria della WDA del progetto SWAMP [ARC21b].

Questa piattaforma della WDA è pensata per includere nel sistema anche i risultati e le triple presenti nel grafo < http://wot.arces.unibo.it/observation/2021 >.

Nei primi giorni della sua fase di testing, l'aggregatore ha riscontrato problemi riguardanti un'oscillazione tra i due valori 0 e -1. Dopo una prima analisi del comportamento errato del software, sono state trovate delle observation dei pluviometri in due grafi differenti. L'aggregatore quindi, riceveva valori contrastanti da due triple differenti che specificavano dati incosistenti. L'errore veniva così riportato (esempio pluviometro Santa Maria) nella WDA di SWAMP (Figura 6.1).

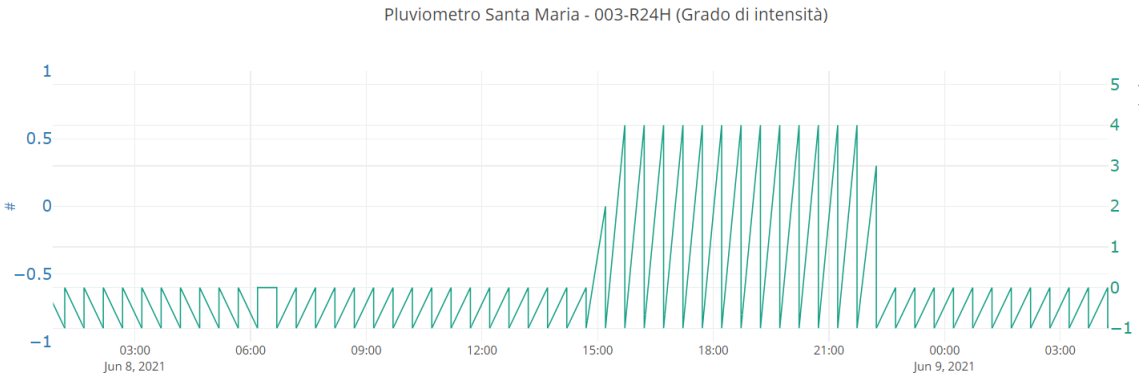

Figura 6.1: Gestione errata del pluviometro Santa Maria

Cancellate le observation aggiuntive e non necessarie, il sistema ha iniziato a funzionare correttamente, identificando le piogge localizzate nelle aree dove sono collocati i tre pluviometri, classificandole per intensità.

Di seguito si riporta l'esempio del pluviometro Correggio, il primo e l'unico che ha registrato un evento piovoso nel periodo che va dal 08/06-02/07. L'attività riportata dal software è relativa alla giornata del 10 giugno 2021. In questa data è stato segnalato un valore di pioggia cumulata dalle 16:20:24 alle 22:22:04 di circa 6.79 mm di acqua piovana ogni mezz'ora, per un totale di circa 88.27 mm. Come è possibile osservare dalla Figura 6.2, la pioggia è caduta in modo costante e concentrata.

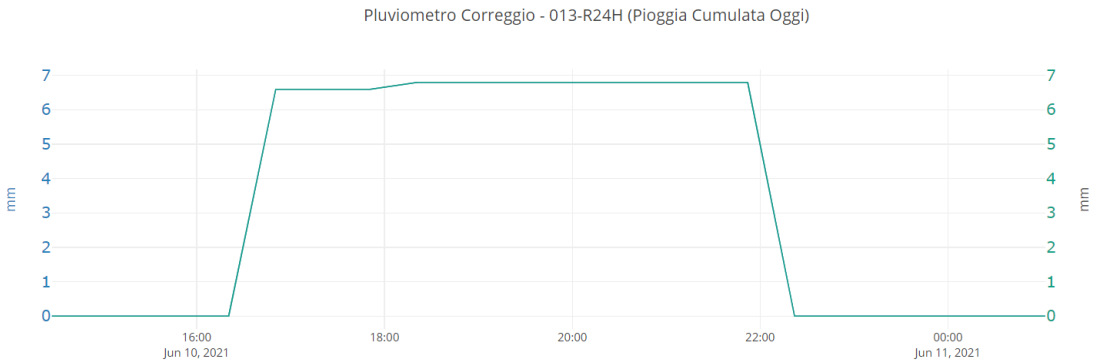

Figura 6.2: Pioggia cumulata dal pluviometro Correggio nella giornata del 10 giugno 2021.

A questi dati l'aggregatore ha associato un grado di allarme (intensità) pari a 4 (Figura 6.3), cioè di un tipo di pioggia forte (Capitolo 5.2 - "Analisi"), dimostrando così la corretta funzionalità del software sviluppato.

Le osservazioni riportate per il pluviometro di Correggio non sono purtroppo state confermate dagli altri pluviometri. La fase di testing portata avanti dal gruppo Arces [Uni21c], permetterà, prima di aggiungerlo in piattaforma WDA originale [ARC21a], di verificare il corretto funzionamento di tutti i sensori.

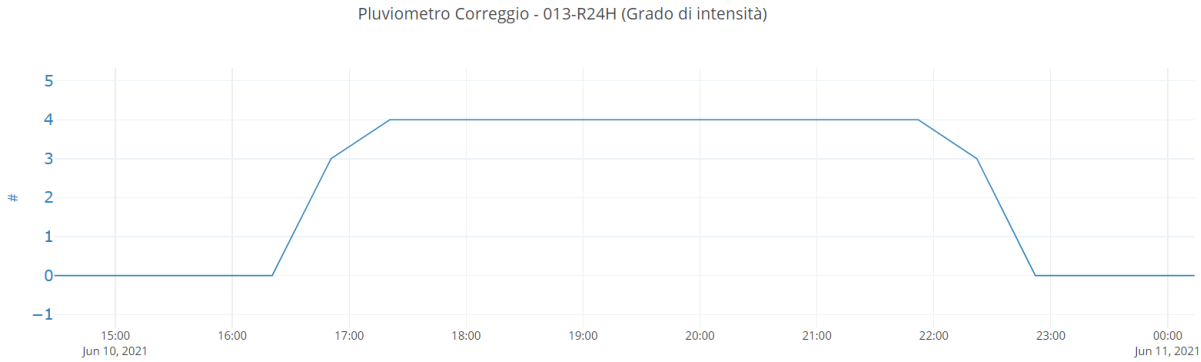

Figura 6.3: Grado di allarme assegnato dal software a seguito dai mm di pioggia registrati dal pluviometro Correggio nella giornata del 10 giugno 2021.

Considerando la scalabilità del progetto SWAMP ed i suoi obiettivi, l'applicativo potrebbe avere numerosi sviluppi futuri. Ad esempio, sarebbe opportuno approfondire i dati ricevuti durante la fase di sperimentazione, contribuendo alla dashboard SWAMP 'SuperSet' [Apa21b], software sviluppato da Apache [Apa21a] per la visualizzazione grafica di dati strutturati ed analizzati attraverso il linguaggio standardizzato per database 'SQL'.

Inoltre, aumentando il numero di pluviometri nell'area del comune di Bologna sarebbe possibile avere dati sempre più precisi e tempistiche più rapide per effettuare operazioni di prevenzione.

Un'altra possibile evoluzione potrebbe essere la trasformazione in notifica delle informazioni, oggi consultabile attraverso piattaforma, coinvolgendo ad esempio, istituzioni pubbliche e private, enti ed associazioni per la tutela del territorio (come vigili del fuoco, protezione civile, etc...).

Un perfezionamento aggiuntivo del software inoltre, potrebbe comprendere l'integrazione dei dati generati dai sensori sul campo (*in-situ*) con i dati da satellite, oppure ottenuti con nuove tecnologie di osservazione della terra, migliorando la qualità delle informazioni, per avere risposte sempre più veloci e puntuali.

Infine, aumentando i settori di impatto, il software potrebbe aver ragione anche in altri campi applicativi, come turismo, trasporti, acquacoltura, etc... L'utilizzo dei dati presenti in piattaforma può contribuire quindi all'ottimizzazione di diversi servizi utili al cittadino.

Bibliografia

[T B01] O. Lassila T. Berners-Lee J. Hendler. *The Semantic Web*. 2001. URL: https://www-sop.inria.fr/acacia/cours/essi2006/Scientific%5C%20American_%5C%20Feature%5C%20Article_%5C%20The%5C%20Semantic%5C%20Web_%5C%20May%5C%202001.pdf.

[FAO02] FAO. *Acqua per le Colture, Ogni goccia conta*. 2002. URL: http://www.fao.org/3/y3918o/y3918o.pdf.

[Fao02] Fao. *ORGANIZZAZIONE PER L'ALIMENTAZIONE E L'AGRICOLTURA DELLE NAZIONI UNITE*. 2002. URL: http://www.fao.org/3/y3918i/y3918i03.htm.

[PE08] Corazzon Paolo e Giuliacci Emanuela. *La meteorologia per tutti*. 2008.

[Gaz11] Ufficiale Gazzetta. *Gazzetta Ufficiale*. 2011. URL: https://www.gazzettaufficiale.it/home.

[Ale+14] Botta Alessio et al. *On the integration of cloud computing and internet of things*. 2014. URL: https://www.researchgate.net/publication/287883657_On_the_integration_of_cloud_computing_and_internet_of_things.

[Une15] Unesco. *L'ACQUA PER UN MONDO SOSTENIBILE, FATTI E CIFRE*. 2015. URL: http://www.unesco.org/new/fileadmin/MULTIMEDIA/HQ/SC/images/WWDR2015Facts_Figures_ITA_web.pdf.

[Com17] European Commission. *Swamp Project approved by European Commission*. 2017. URL: https://cordis.europa.eu/project/id/777112/it.

[Fed+18] Montori Federico et al. *Machine-to-machine wireless communication technologies for the Internet of Things: Taxonomy, comparison and open issues*. 2018. URL: https://www.sciencedirect.com/science/article/pii/S1574119217303668.

[Luc+18] Roffia Luca et al. *Articolo SEPA mdpi*. 2018. URL: https://www.mdpi.com/1999-5903/10/4/36/htm.

[JAI19] Telicko Jevgenijs, Jakovics Andris e Drirkis Ivars. *A low-cost wireless sensor network for long term monitoring of energy performance and sustainability of buildings*. 2019. URL: https://www.researchgate.net/publication/335655214_A_low-cost_wireless_sensor_network_for_long_term_monitoring_of_energy_performance_and_sustainability_of_buildings.

[ARC20] ARCES. *Dashboard SEPA*. 2020. URL: https://github.com/arces-wot/SEPA-Dashboard.

[RR20] Chataut Robin e Akl Robert. *Massive MIMO Systems for 5G and Beyond Networks—Overview, Recent Trends, Challenges, and Future Research Direction*. 2020. URL: https://www.researchgate.net/publication/341332727_Massive_MIMO_Systems_for_5G_and_Beyond_Networks-Overview_Recent_Trends_Challenges_and_Future_Research_Direction.

[STA20] U STAMFORD. *Gartner Forecasts Worldwide Public Cloud Revenue to Grow 6.3% in 2020*. 2020. URL: https://www.gartner.com/en/newsroom/press-releases/2020-07-23-gartner-forecasts-worldwide-public-cloud-revenue-to-grow-6point3-percent-in-2020#:~:text=The%5C%20worldwide%5C%20public%5C%20cloud%5C%20services,increasing%5C%2095.4%5C%25%5C%20to%5C%20%5C%241.2%5C%20billion..

[Apa21a] Apache. *Apache*. 2021. URL: https://www.apache.org/.

[Apa21b] Apache. *Superset*. 2021. URL: https://superset.apache.org/.

[ARC21a] ARCES. *WDA Swamp*. 2021. URL: http://mml.arces.unibo.it/swamp/wda/.

[ARC21b] ARCES. *WDATemp*. 2021. URL: http://mml.arces.unibo.it/swamp/wda/2021/.

[Bol21] N Boldrini. *Consumo di acqua eccessivo, come intervenire?* 2021. URL: https://tech4future.info/consumo-di-acqua-tecnologie-e-applicazioni-possibili/.

[Bon21] Consorzio di Bonifica dell'Emilia Centrale. *Dettagli estenzione consorsio*. 2021. URL: http://www.emiliacentrale.it/cosa-facciamo/statuto/.

[Con21] Consorzio. *Consorzio di Bonifica Emilia Centrale*. 2021. URL: https://www.emiliacentrale.it/.

[Coo21] M Cooper. *How the cell phone changed lives... by its inventor*. 2021. URL: https://www.techradar.com/news/how-the-cell-phone-changed-lives-by-its-inventor.

[FAO21] FAO. *FAO*. 2021. URL: http://www.fao.org/home/en/.

[Hol21] A Holst. *Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2024*. 2021. URL: https://www.statista.com/statistics/871513/worldwide-data-created/.

[Nat21] Unite Nations. *SDG Indicators*. 2021. URL: https://unstats.un.org/sdgs/metadata/.

[ONU21] ONU. *Statistiche Onu SDG*. 2021. URL: https://unstats.un.org/sdgs/metadata/.

[Pis21] Lorenzo Pisanò. *Notifica SEPA Github*. 2021. URL: https://github.com/lorenzo0/NotificaSEPA/.

[Sci21] ScienceDirect. *Sito di Science Direct*. 2021. URL: https://www.sciencedirect.com/.

[Sta21] Statista. *Number of internet of things (IoT) connected devices worldwide in 2018, 2025 and 2030*. 2021. URL: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/.

[SWA21] SWAMP. *Swamp Project Website*. 2021. URL: http://swamp-project.org/.

[Uni21a] Unibo. *DICAM*. 2021. URL: https://dicam.unibo.it/it.

[Uni21b] Unibo. *DISTAL*. 2021. URL: https://distal.unibo.it/it.

[Uni21c] Unibo. *Gruppo ARCES*. 2021. URL: https://centri.unibo.it/arces/en.

[W321] W3. *Documentazione ufficiale W3*. 2021. URL: https://www.w3.org/TR/sparql11-query/.

[w321] w3. *URI*. 2021. URL: https://www.w3.org/Addressing/.

Ringraziamenti

A conclusione di questo elaborato, desidero menzionare tutte le persone, senza le quali questo lavoro di tesi non esisterebbe nemmeno.

In primis, ringrazio l'Università di Bologna, che mi ha accompagnato in questo magnifico percorso triennale e che mi ha dato la possibilità di partire per il progetto Erasmus anche durante questo difficile periodo di pandemia.

Alla mia famiglia, per avermi sempre seguito e motivato a superare le mie paure ed insicurezze. Tutti e cinque, con Coral, siete sempre stati colonne portanti e punti di riferimento fissi sin dal primo giorno. Questa laurea è anche vostra, che avete combattuto e stretto i denti al mio fianco. Vi voglio bene.

Ringrazio infinitamente mamma e papà che mi hanno sempre sostenuto, appoggiando ogni mia decisione, fin dalla scelta del mio percorso di studi. Grazie per esserci sempre stati anche nei momenti più difficili e (molto) prossimi alla consegna della laurea.

Ringrazio i miei due fratelloni, Emanuela e Francesco, per la loro vicinanza materiale ed emotiva. Senza i loro consigli, la loro esperienza ed il loro supporto non sarei mai potuto arrivare fin qui.

Ai miei nonni, Marisa, Antonietta e Donato per l'appoggio che mi avete sempre dato, per la stima che mi avete dimostrato e che mi ha spronato ad andare avanti ed a non arrendermi mai.

Ringrazio Giorgia ed Elisa che, con tanta pazienza hanno condiviso centinaia di ore al bar, in facoltà, vissuto insieme ansie pre-esame e sbronze post-esame.

Ringrazio Enea, Leonardo e Francesco che vicini e lontani non hanno mai smesso di volermi bene e di sostenermi in questi anni così pieni di stimoli.

Ai miei amici e ai colleghi di corso, le persone con cui ho condiviso attimi di gioia e di tristezza, ma che nonostante tutto sono rimasti accanto a me in questi anni di Università. Grazie per essere stati miei complici, ognuno a suo modo, in questo percorso intenso ed entusiasmante, nel bene e nel male.

Un ringraziamento speciale va a Beersa, il gruppo più pazzo ed unito che io conosca, cinque mesi non sono stati abbastanza, pensate quattro righe! Grazie per aver reso il mio erasmus un periodo indimenticabile, pieno di esperienze, viaggi ed infinite avventure.

Un ultimo saluto va a Bologna, la dotta, la rossa e la grassa. La città che ha segnato un periodo della mia vita che non dimenticherò mai. Piazza Maggiore, piazza San Francesco e piazza Verdi, la palestra upClimbing e le scuderie sempre piene ma dove si trova sempre posto, la frase 'andiamo a studiare' che si trasformava in un aperitivo la porterò sempre con me. Spero di rincontrarti presto.