DOCTORAL THESIS

INTEROPERABLE FRAMEWORK AND DYNAMIC WEB PLATFORM FOR INTEGRATED KNOWLEDGE MANAGEMENT IN IOT-ENABLED SMART WATER NETWORKS

authored by

MANDEEP SINGH

supervised by

Prof. Wenyan Wu, Dr. Edlira Vakaj, and Dr. Stamatia Rizou

A thesis submitted in fulfilment of the requirements for the degree of DOCTOR OF PHILOSOPHY in the Faculty of Computing, Engineering and The Built Environment (CEBE)

Birmingham City University

January, 2026

Declaration of Authorship

I, Mandeep Singh, declare that this thesis titled, ‘Interoperable Framework and Dynamic Web Platform for Integrated Knowledge Management in IoT-enabled Smart Water Networks’ and the research work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. Except for such quotations, this thesis is entirely my work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by me jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Date: *2026-01-13*

Signed: *Mandeep Singh*

Abstract

The integration of Internet of Things (IoT) technologies at the physical layer of Smart Water Networks (SWNs), such as smart sensors and valve controllers, enables the establishment of cohesive overlay networks within SWNs. Networked IoT systems support the utilisation of diverse data types, including numerical values, symbols, text, images, and contextualised information, thereby facilitating advanced sensing capabilities that extend beyond the initial coverage areas of SWNs. However, using publicly accessible IoT data poses interoperability challenges, including heterogeneous data representation formats, varying semantic models, and the need to adapt domain-specific standards and ontologies.

To address these challenges, this research first introduces a Data Information Representation Model (DIRM) specifically designed for the water domain. Subsequently, a Data Information Interoperability Model and Methodology (DIIM) is proposed to facilitate both syntactic (structural) and semantic (meaningful) interoperability between applications. Additionally, a Data Information Interoperability Framework (DIIF) is introduced, which leverages Semantic Web (SW) and neural Natural Language Processing (NLP) technologies to support the key steps of DIIM: transformation, alignment, storage, and validation. The transformation step converts semi-structured IoT data into a Resource Description Framework (RDF) graph using a graph converter. The alignment step matches IoT terms used to label measurement values with corresponding terms in other application data models, standards, and ontologies. The storage step records all alignments as pairs of terms in a separate ontology and annotates their references within the IoT graph model. These processes are implemented through the extensible components of DIIF, including the DIIM Semantic Similarity Scoring Tool (DS3T) and the

Semantic Similarity Scoring Ontology (S3O), which support multiple semantic similarity algorithms to address various semantic aspects. For the validation step, DIIF introduces the Data Information Interoperability Ontology (DIIO) as a knowledge base and the Data Information Interoperability Questionnaire (DIIQ) to validate both syntactic and semantic interoperability between applications.

The effectiveness of DIIM and DIIF was demonstrated through implementation and validation in case studies from the water and telecommunications domains. In all validation scenarios, data interoperability was enabled and enhanced by representing data in Knowledge Graphs and by identifying similar terms within domain-specific standards and ontologies, which facilitated rapid adaptation of the required data structures and semantics.

Dedicated to my beloved parents, whose

unconditional love, affection, support, and prayers are my strength in everything I do.

Acknowledgements

The research presented in this thesis was funded by Birmingham City University through the European Union's Horizon 2020 Research and Innovation programme under the Marie Sklodowska-Curie ITN, Horizon 2020 MSCA-EID-IOT4Win project (765921) as part of the scholar's program and supported by many individuals. First and foremost, I would like to praise and thank Waheguru, the Almighty, who has given me infinite grace, knowledge, and opportunity to complete my PhD study and this thesis.

I want to thank my supervisory team, including my first supervisor, Prof. Wenyan Wu, and my second supervisors, Dr Edlira Vakaj and Dr Stamatia Rizou, for their unwavering support throughout my PhD studies. I appreciate my supervisors for their ideas, guidance, and continuous motivation. Without their advice and support, I would not have reached this stage in completing my PhD study. I could not have imagined having a better director and supervisors for my PhD study. I am part of this incredible journey because they believed in me and provided excellent assistance and support in helping me secure a place at Birmingham City University as a PhD student.

I thank my wife, Amanpreet, for her unwavering support and assistance throughout the challenging times and my research journey. She has made a significant contribution to keeping things smooth and flexible during my PhD studies. I could not be more thankful for her kindness, assistance, and motivation during this journey. I am eternally grateful to my Mother for being my strength and inspiration in everything I do, and for being the reason I am at this stage in my life. Also, I would like to thank my children, Pia and Pahul, for their understanding that I need time and peace to complete my PhD studies. No words can truly express how grateful I am for sharing emotional and unforgettable moments with all of them. Last but not least, I would like to thank my best friends for their unwavering support throughout my research journey, which made it enjoyable and unforgettable.

I want to thank SingularLogic and the European Union for the financial support during my research through the Horizon 2020 Marie Skłodowska-Curie Research in projects: IoT4win (765921), Innovation Staff Exchange Programme "Research Collaboration and Mobility for Beyond 5G Future Wireless Networks (RECOMBINE)" under Grant no. 872857 and BG-RRP-2.005 -"Twinning" titled with "Twinning for Excellence in Research in Sustainable Future Communication Networks in the Context of a Green Economy – GREENBEAT".

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Around the world, software providers are building IoT applications by integrating various solutions and systems that enable remote and continuous monitoring and diagnosis of problems, manage maintenance issues, and optimise domain-specific operations using data-driven and knowledge-driven approaches. Organisations' gradual deployment of data-enabled IoT devices, such as smart sensors and actuators, has enabled the development of a cohesive *overlay network* in the IoT landscape (Sensus, 2019). Once applications and IoT devices are networked, they can begin communicating and exchanging information. However, interoperability (the exchange and utilisation of information) cannot succeed without the syntactic (structural) and semantic (meaningful) interoperability of the data/information they share, because interoperability issues arise from heterogeneity in data formats and semantics. Figure 1.1 illustrates an example scenario of an IoT-enabled Smart Water Network (SWN) (Sensus, 2019), in which IoT devices, such as pumps, pipes, sensors, and valves, collect data and send it to SWN applications for monitoring, sensing, analytics, or decision-making purposes. When IoT sends data, it can use any data format, e.g. Comma-separated Values (CSV), JavaScript Object Notation (JSON), or Extensible Markup Language (XML). That data format may not be the expected format of the SWN application. Similarly, to record data, IoT could use a different label, such as $Timestamp$,

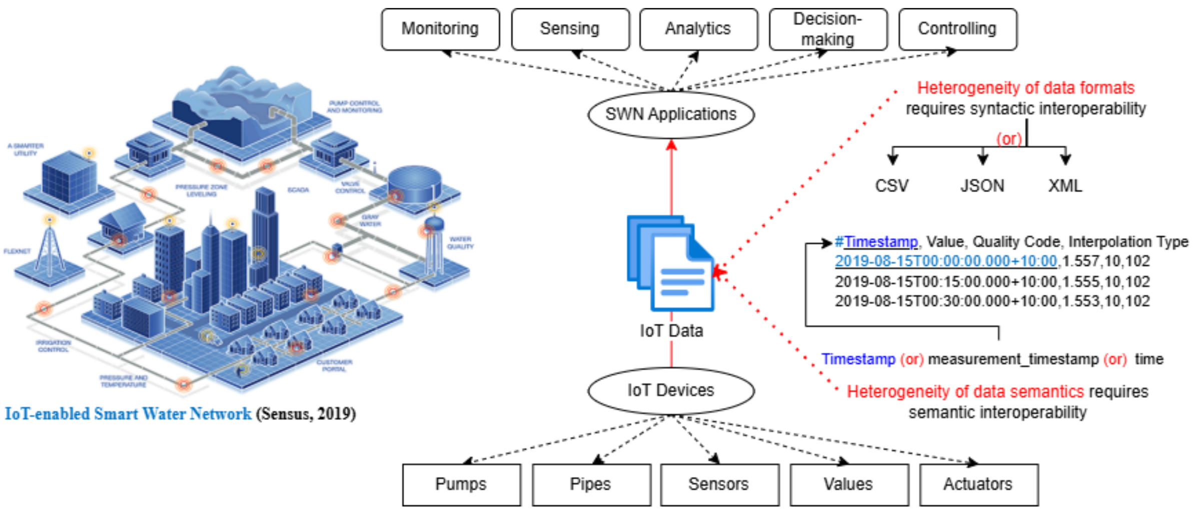

Monitoring
Sensing
Analytics
Decision-making
Controlling
SWN Applications
Heterogeneity of data formats requires syntactic interoperability
(or)
CSV
JSON
XML
#Timestamp, Value, Quality Code, Interpolation Type
2019-08-15T00:00:00.000+10:00,1.557,10,102
2019-08-15T00:15:00.000+10:00,1.555,10,102
2019-08-15T00:30:00.000+10:00,1.553,10,102
IoT Data
Timestamp (or) measurement_timestamp (or) time
Heterogeneity of data semantics requires semantic interoperability
IoT Devices
Pumps
Pipes
Sensors
Values
Actuators
IoT-enabled Smart Water Network (Sensus, 2019)

Figure 1.1: Heterogeneity issues between IoT devices and SWN applications

but the SWN application expects the label as *measurement_timestamp* or *time*. Regardless of heterogeneity issues, for instance, at the point of decision-making, a Decision Support System (DSS) relies on understanding all available data and information from every IoT device and database. Otherwise, it would not be able to provide accurate advice.

The interoperability of IoT-enabled applications remains an active research topic, as noted by Kalatzis et al. (2019). Data interoperability is a significant obstacle to promoting IoT adoption and innovation. A key challenge to achieving semantic data interoperability is aligning heterogeneous data models across diverse implementations. Typically, this process must compute semantic similarity scores among potential synonyms. Despite the existence of algorithms for similarity calculation in the literature, this process requires considerable manual effort from data workers and analysts to align application-specific data models with domain-specific ontologies and standards.

1.2 Problem Statement

Urban water data generated by IoT devices is voluminous and multi-modal, varying in quality, format, and representation. Effective and accurate decision-making and analytics in water management require considering all available data and information from every IoT device at the considered point in real time. Thus, interoperability of integrated data and information among IoT devices and applications of a SWN is necessary to enable intelligent applications and services for the entire water-cycle decision-making process.

Current SWN frameworks fail to address the dynamic alignment of IoT data with the domain ontologies and standards, thereby limiting real-time decision-making. Moreover, an interoperability service is required to address the challenges of expressing, understanding, sharing, and collaborating on data and information across heterogeneous models, supporting diverse decision-making applications by dynamically constructing information models from the same data and information in near real-time.

1.3 Research Aim and Objectives

Interoperability makes the IoT-enabled SWN truly "smart" by enabling seamless, efficient, and coordinated communication and data sharing among all IoT devices and SWN applications, thereby allowing the water system to operate as a unified, intelligent system. Interoperability of IoT data is the crucial requirement for enabling automated, data-driven decision-making for advanced control, e.g. leak detection, pressure optimisation, and quality monitoring, in IoT-enabled SWNs. This thesis aims to design an interoperable framework and dynamic web platform for integrated knowledge management in IoT-enabled Smart Water Networks (SWNs) by utilising Semantic Web (SW) and Natural Language Processing (NLP) technologies.

IoT devices and applications (e.g. analytics and decision-making) of SWNs should seamlessly operate across the network, relying on interoperable data across interrelated domains throughout the entire water cycle. The following objectives of this research work have been established to fulfil the above aim:

1. Identify the interoperability problem of IoT-enabled SWNs and outline a research methodology to address the identified problem.

2. Review literature to identify interoperability-related challenges and research gaps in IoT-enabled SWNs.

3. Develop a model and methodology that enable and enhance the syntactic and semantic interoperability of data collected by IoT devices (e.g. actuators, sensors, and valves) and support data sharing over the Internet.

4. Design and develop a modular framework and web platform by utilising Semantic Web (SW) and Natural Language Processing (NLP) technologies to achieve smart interoperability in IoT-enabled SWNs.

5. Conduct case-study validations of the proposed model, methodology, and framework using predefined interoperability metrics.

1.4 Research Methodology

This research methodology has focused on innovative technologies such as SW, Neural Network (NN), and NLP to develop a novel framework for the interoperability of data and information in IoT-enabled SWNs. SW technologies are selected because they are renowned for achieving interoperability on the Web. NLP is a branch of artificial intelligence focused on enabling computers to understand and interpret documents written in human-readable languages. Currently, the NN-based framework for NLP has reached new standards of quality. It has become the dominant approach for various NLP tasks, including machine translation (MT), machine reading comprehension (MRC), chatbots, and more. Therefore, SW, NN, and NLP technologies became the focus of this research. Figure 1.2 presents the research methodology, from top to bottom, outlining its activities.

1.5 Main Contributions to Knowledge

In this research project, I have focused on innovative technologies such as SW, Machine Learning (ML), NN, and NLP to develop a novel model, methodology, and framework for the interoperability of data and information in IoT-enabled SWNs. During my research, I have investigated the application of these technologies to achieve the aim and objectives of my research.

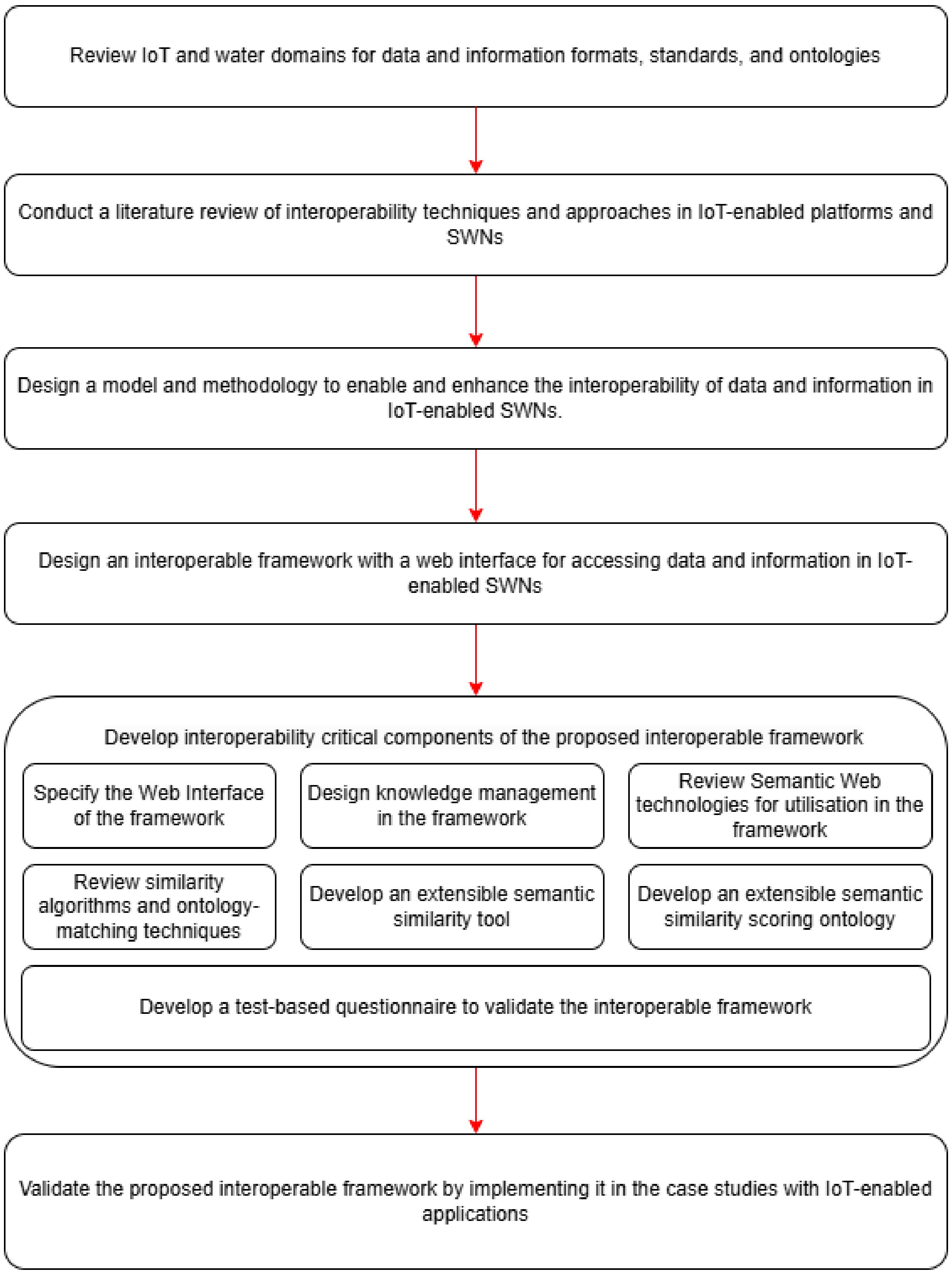

Review IoT and water domains for data and information formats, standards, and ontologies
Conduct a literature review of interoperability techniques and approaches in IoT-enabled platforms and SWNs
Design a model and methodology to enable and enhance the interoperability of data and information in IoT-enabled SWNs.
Design an interoperable framework with a web interface for accessing data and information in IoT-enabled SWNs
Develop interoperability critical components of the proposed interoperable framework
Specify the Web Interface of the framework
Design knowledge management in the framework
Review Semantic Web technologies for utilisation in the framework
Review similarity algorithms and ontology-matching techniques
Develop an extensible semantic similarity tool
Develop an extensible semantic similarity scoring ontology
Develop a test-based questionnaire to validate the interoperable framework
Validate the proposed interoperable framework by implementing it in the case studies with IoT-enabled applications

Figure 1.2: Research methodology

As a result, this thesis presents three key contributions as follows:

1. Conceptual models and methodology as **Data Information Representation Model (DIRM)** and **Data Information Interoperability Model and Methodology (DIIM)**. Data Information Representation Model (DIRM) identifies the layers in which data, information, and knowledge from the water domain are represented. DIIM enables and enhances the interoperability of IoT-enabled applications with a three-step methodology: *(i) transform* data into a graph, *(ii) align* data with other ontologies and standards and *store* alignments, and *(iii) validate* the transformed graph at the syntactic and semantic level.

2. A DIIM-based modular framework and web platform, **Data Information Interoperability Framework (DIIF), for IoT-enabled applications**, with the following interoperability crucial components:

   (a) The **Data Information Interoperability Ontology (DIIO)**, developed with SW technologies, represents the knowledge base that contains domain knowledge defined by domain experts in ontologies and standards. DIIF uses this information to support the enablement and enhancement of the syntactic interoperability of the data collected by IoT for SWNs. This ontology is required to support DIIM's *transform* step.

   (b) **Semantic Similarity Scoring Ontology (S3O)**, developed with SW technologies, for storing different algorithm-based similarities among the terms used in datasets, ontologies, and standards. This ontology is required to support DIIM's *align-and-store* step.

   (c) **DIIM Semantic Similarity Scoring Tool (DS3T)**, with SW, ML, NN, and NLP technologies, utilising various NLP-based algorithms to calculate similarities among the terms used in input datasets, ontologies, and standards, and store the results in S3O.

   (d) A test-driven **Data Information Interoperability Questionnaire (DIIQ)** to validate syntactic and semantic interoperability between IoT-enabled applications, after the application of DIIM and on IoT data. This questionnaire is required to support DIIM's *validate* step.

3. Following **validation showcases with application of DIIM and DIIF in three IoT-enabled application scenarios** from different domains are delivered:

   (a) **IoT-enabled SWN monitoring and management** showcases how DIIM helps to overcome interoperability issues between a SWN system and a monitoring system. Interoperability issues arise from the heterogeneous data formats and standards used by these IoT-enabled systems.

   (b) **Alignment of IoT Data for SWNs** showcases the DIIF 's interoperability enablement and enhancement by aligning IoT data to the other datasets, standards, and ontologies of the water domain. After applying DIIM and DIIF to a set of specified IoT data, standards, and ontologies, the resulting data and information can be used by all applications that support the prescribed standards or ontologies.

   (c) **QoE prediction across IoT-enabled 5G/6G datasets** showcases DIIF's interoperability enablement and enhancement by aligning 5G/6G datasets with other datasets and standards of the Quality of Experience (QoE) domain. This showcase validates the application of DIIM and DIIF beyond SWNs. Thus, the proposed DIIM methodology and DIIF framework can also be applied to any IoT-enabled application domain.

1.6 Publications and Awards

The following conference papers have been published to build on the work conducted in this thesis:

1. **Mandeep Singh**, Moatasim Mahmoud, Rizou Stamatia, Zaharias D. Zaharis, Pavlos I. Lazaridis, Vladimir K. Poulkov, and Wenyan Wu (2024). "Towards 5G/6G Data Harmoni-

sation through NLP and Semantic Web Technologies". In: 2024 Advanced Topics on Measurement and Simulation (ATOMS), pp. 291–294. doi: 10.1109/ATOMS60779.2024.10921618

Award: *This paper was awarded an Excellent Paper Award in the "Oral Papers – Young Scientist" category.*

2. **Mandeep Singh**, Edlira Vakaj, Stamatia Rizou, and Wenyan Wu, 2023. Towards aligning IoT data with domain-specific ontologies through Semantic Web technologies and NLP. SEMANTICS 2023 EU. https://ceur-ws.org/Vol-3510/paper_nlp_1.pdf Keywords: Internet of Things (IoT), Smart Water Network (SWN), Linked Data (LD), ontology, Knowledge Graph (KG), NLP, word2vec, semantic similarity, term alignment

3. **Mandeep Singh**, Wenyan Wu, Stamatia Rizou, and Edlira Vakaj, "Data Information Interoperability Model for IoT-enabled Smart Water Networks," 2022 IEEE 16th International Conference on Semantic Computing (ICSC), Laguna Hills, CA, USA, 2022, pp. 179-186, doi: 10.1109/ICSC52841.2022.00038. keywords: Semantics; Water quality; Ontologies; Syntactics; Data models; Service-oriented architecture; Internet of Things; Data Interoperability; Syntactic harmonization; Semantic Model Generation and Alignment;Smart Water Networks; IoT/WoT; Ontology; Representation Standard; Water Quality Monitoring,

The appendix contains a copy of each paper.

1.7 Thesis Outline

Figure 1.3 shows the structural flow of this thesis's outline, which also presents how it covers the research objectives and questions in layperson's terms. Each chapter includes, among other sections, an overview and summary that provide the reader with a concise understanding of the chapter's content. Each chapter is briefly described as follows:

Chapter 2 presents background concepts and literature on interoperability, the Internet of Things (IoT), and the Smart Water Networks (SWNs). It is concluded by listing interoperability challenges and research gaps in IoT-enabled SWNs.

Chapter 3 proposes the concepts of the Data Information Representation Model (DIRM) and the Data Information Interoperability Model and Methodology (DIIM) for SWNs. DIIM further proposes three steps: transformation, alignment and storage, and validation to enable and enhance the interoperability of IoT data for SWN applications.

Chapter 4 presents the design and implementation of a DIIM-based Data Information Interoperability Framework (DIIF). It also describes how DIIF's crucial components, such as the DIIM Semantic Similarity Scoring Tool (DS3T), the Semantic Similarity Scoring Ontology (S3O), and the Data Information Interoperability Questionnaire (DIIQ) are developed and can be used in data interoperability scenarios.

Chapter 5 demonstrates the application and validity of DIIM and DIIF through different showcases. Showcases are from the water and 5G/6G domains, where the end applications require interoperability of IoT data.

Chapter 6 presents a discussion of the thesis's contributions and a reflection on the research aim and objectives. The chapter concludes by highlighting a few key observations that suggest directions for future research.

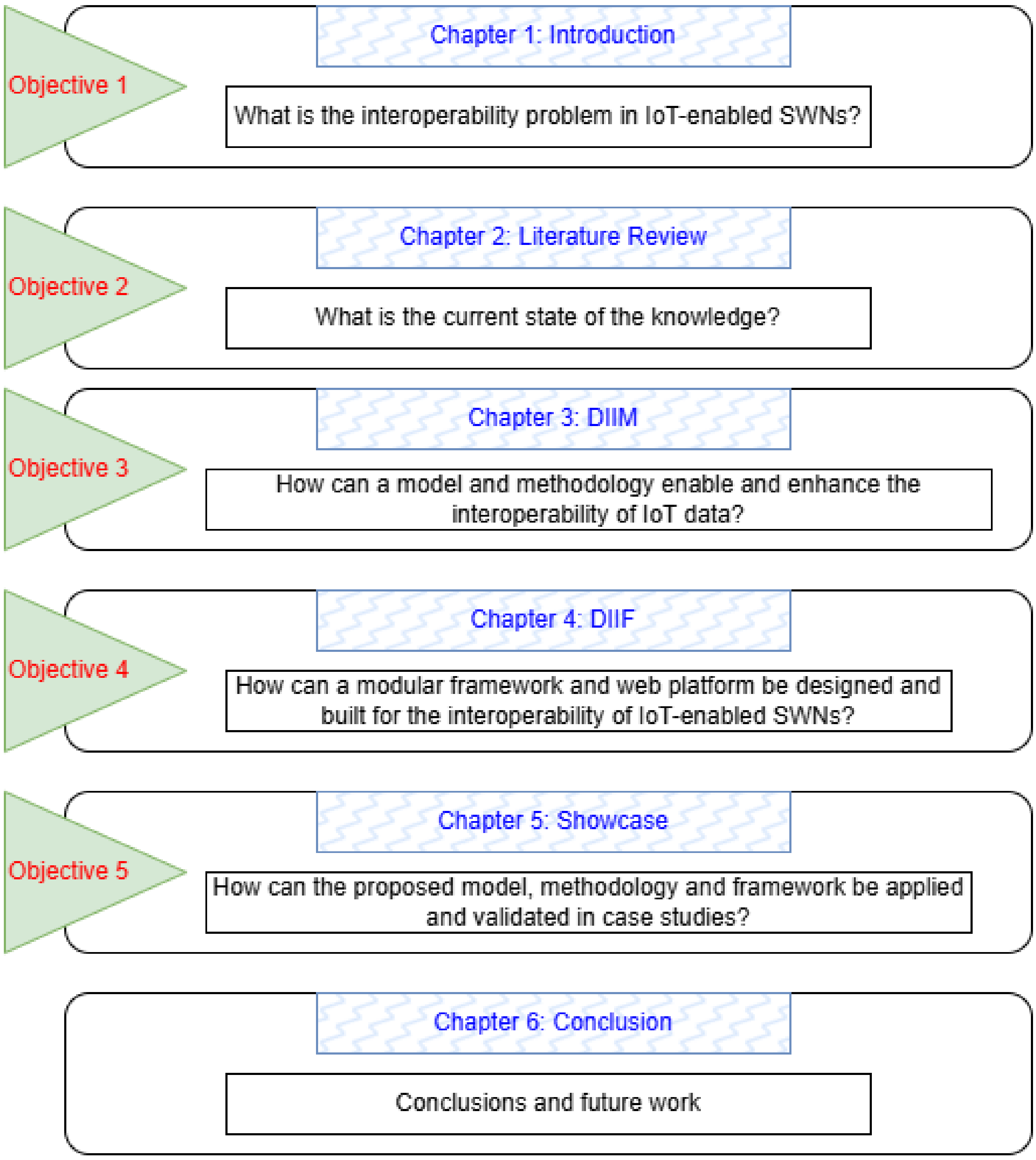

Objective 1
Chapter 1: Introduction
What is the interoperability problem in IoT-enabled SWNs?
Objective 2
Chapter 2: Literature Review
What is the current state of the knowledge?
Objective 3
Chapter 3: DIIM
How can a model and methodology enable and enhance the interoperability of IoT data?
Objective 4
Chapter 4: DIIF
How can a modular framework and web platform be designed and built for the interoperability of IoT-enabled SWNs?
Objective 5
Chapter 5: Showcase
How can the proposed model, methodology and framework be applied and validated in case studies?
Chapter 6: Conclusion
Conclusions and future work

Figure 1.3: The structural flow of this thesis

Chapter 2

Literature Review

The previous chapter introduced the thesis by presenting a problem statement, outlining the research work's aim and objectives, and providing a general explanation of the thesis's contributions. This chapter provides the necessary background and concepts to understand the developed models, methodology, and framework (contributions) presented in Chapters 3 and 4. This chapter also reviews the related work in the IoT and water domains. It concludes the literature review by listing the challenges and research gaps in the IoT-enabled SWNs.

2.1 Overview

This literature review establishes key definitions and concepts related to the Internet of Things (IoT) and the water domain. It examines the architecture of Smart Water Networks (SWNs) and reviews existing SWN solutions within the water sector. SWNs integrate several innovative technologies, including IoT for smart sensors, Data Distribution Service (DDS), Multi-Agent Systems (MAS) for autonomous operations, and Semantic Web technologies for data management and sharing. The enabling technologies are addressed in detail in subsequent sections. The chapter also introduces fundamental terms such as data, information, and knowledge, as well as core concepts like communication and interoperability, as defined in the literature, due to their frequent use by researchers in this field. The most frequently cited interoperability model, the Levels of Conceptual Interoperability Model (LCIM), is discussed and compared

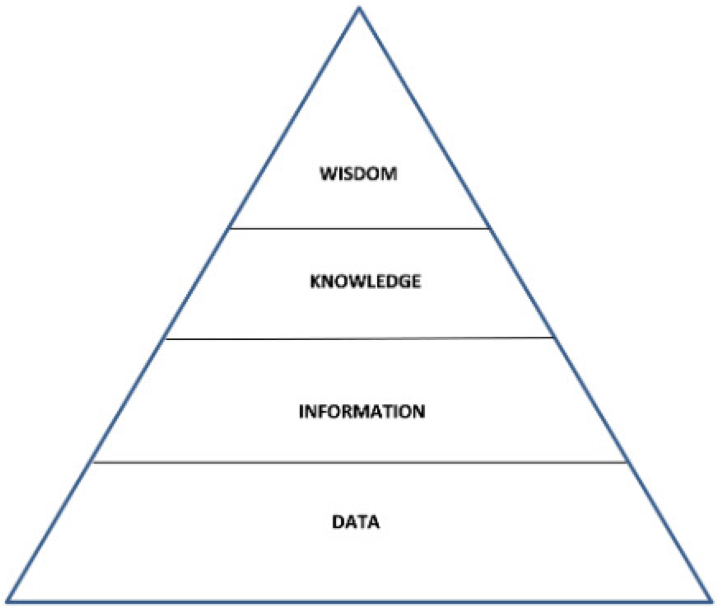


Figure 2.1: DIKW hierarchy (Rowley, 2007)

with the well-known Open Systems Interconnection (OSI) model. The discussion emphasizes the importance of standardized reference models to prevent conflicts in data and information sharing. Numerous researchers highlight the essential role of ontologies in achieving semantic interoperability, particularly within the Semantic Web and across both industry and research applications. This chapter reviews and presents prominent ontologies and research projects from the IoT and water domains. Finally, the chapter lists the challenges and identifies research gaps in IoT-enabled SWNs.

2.2 Knowledge-related Terms, Concepts and Technologies

The following subsections elaborate on key terms, concepts, and technologies related to knowledge that are crucial to this research project.

2.2.1 Knowledge Hierarchy

Figure 2.1 shows the *knowledge hierarchy or knowledge pyramid* (Rowley, 2007), in which he defines *data* as symbols that are properties of observables and *information* as descriptions. The difference between the two is not structural but functional, and information is inferred from

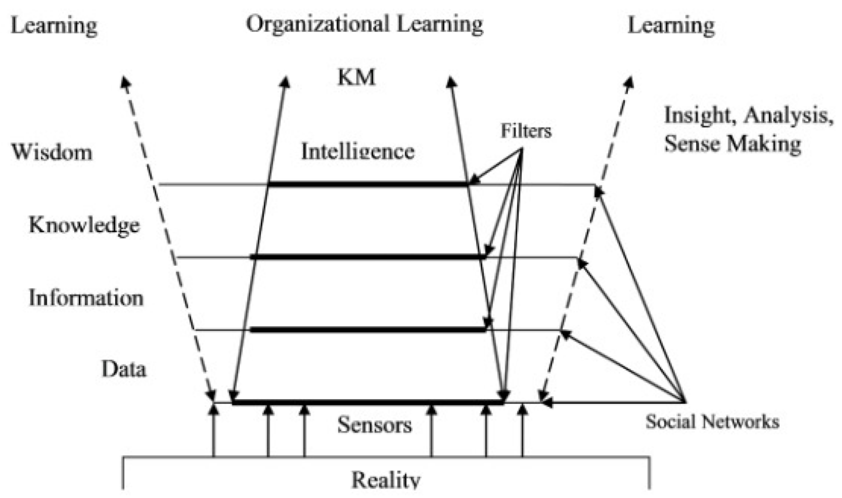


Figure 2.2: The revised Knowledge-KM pyramid (Jennex et al., 2015)

data. *Knowledge* as know-how is acquired from learning, i.e. by instruction or experience, and adaptation, i.e. the correction of the learned under new circumstances. Knowledge acquisition requires understanding an error, why it occurs, and how to correct it. According to Rowley (2007):

1. *Information systems can be automated and generate information out of data.*
2. *That computer-based knowledge systems require higher-order mental faculties; "they do not develop knowledge, but apply knowledge developed by people".*
3. *And that wisdom adds value, endures forever, and will probably never be generated by machines.*

Figure 2.2 shows the revised knowledge-KM pyramids, in which the following terms are proposed for consensus working definitions:

I *Intelligence* refers to specific, actionable knowledge required to make a particular decision in a specific context in a specific organisation.

II *Learning* refers to the acquisition of DIKW that leads to a change in behaviour or expectations within the individual or group engaged in the learning.

Table 2.1: Explanations of DIEK (Dammann, 2019)

Concept	What is it?	How produced?	By whom?	Goal?
Data	Numbers, Symbols, Text, Images, Sound recordings, Unit values	**Collected** from field research, database, measurements in experiments, from individuals, populations	Data Collector	Use as raw data or for information generation Storage, curation, retrieval
Information *is data contextualised*	Data in context	**Contextualisation** by making data useful, and using it for specific tasks	Informatician, informaticist, statistician, data scientist	Use as a source for answering questions Storage, curation, retrieval
Evidence *is information compared*	Useful, contextualised information	**Comparison** with standards, reference values, reference information	Scientist, theoretician, philosopher Interventionist, policymaker	Use for analysis and hypothesis-testing to support claims/hypotheses and decision-making
Knowledge *from evidence*	Evidence-based, (predictive, testable, consistently successful) belief	**Consensus** based on reasoning and discussion		Justification

III *Organisational Learning* refers to learning that leads to quantifiable improvement of activities, increased available knowledge for decision-making, or a sustainable context. An organisation learns if its processing of DIKW changes its potential behaviours.

IV *Social networks* refer to formal or informal, direct or indirect methods used to share DIKW among users.

V *Filters* refer to KM processes that limit access, separate, and capture the Data Information Knowledge Wisdom/Intelligence (DIKW/I) required from that which is not.

Dammann (2019) reviews the knowledge approach (Rowley, 2007) and revises it with the new term *evidence*, as shown in Table 2.1. He overlooks the definition of *wisdom - the addition of value to knowledge that requires judgement* (Rowley, 2007) because he thinks knowledge is more important than wisdom and wisdom requires judgment at all levels of the hierarchy. Further, he believes that knowledge can be defined, in the context of informatics and data science, as a predictive, testable, and consistently successful belief, if there is a causal connection between the facts represented by the data, information, and evidence on the one hand, and our beliefs on the other (Dammann, 2019).

2.2.2 Knowledge Base and Ontologies

According to Murdock et al. (2016), the vision of interoperability requires developing ontologies and enabling sensors, devices, and systems to express their contextual information and data using the designed ontologies. Neches et al. (1991) state that an ontology outlines the essential terms and relationships that form the vocabulary of a specific topic. It also establishes the rules for combining these terms and relationships to expand the vocabulary. The consensual knowledge of a domain, captured by humans in interrelated logical statements, forms the basis for machine-readable ontologies, enabling the sharing and reuse of knowledge between humans and machines. Ontologies often address semantic heterogeneity in data by aligning concepts and terms to address meaning differences and achieve data interoperability (Brodaric et al., 2015).

Another related concept to knowledge and ontology is known as the Knowledge Graph (KG). A KG is a structured representation of data in a graph form by applying graph theory (Tutte, 2001). Ontologies are also based on graph theory. Graph form allows entities (such as sensors, places, and concepts) to be organised as nodes and their relationships as edges, facilitating machine-readable understanding and reasoning. Aggregated machine-readable information, structured as Knowledge Graphs (KGs), serves as the backbone of numerous data science applications, ranging from question answering to recommender systems to predicting drug–target interactions (Hofer et al., 2024).

Ontologies and knowledge graphs relate to the concept of knowledge by providing a formal way to structure, express, and operationalise it. An ontology defines the concepts, relationships, and rules of a domain, essentially the conceptual schema of what counts as knowledge. In contrast, a knowledge graph uses that schema to represent concrete facts and entities as interconnected data. Together, they transform abstract human understanding into machine-interpretable, semantically rich knowledge that can be queried, analysed, and reasoned over. One can generate a KG directly from an ontology, and one can infer an ontology from a KG using additional algorithms, but these are not simple one-to-one conversions. Ontologies and KGs are complementary rather than interchangeable. Where Ontologies define semantics, knowledge graphs operationalise them. This division is precisely why machines can process both, but not in the

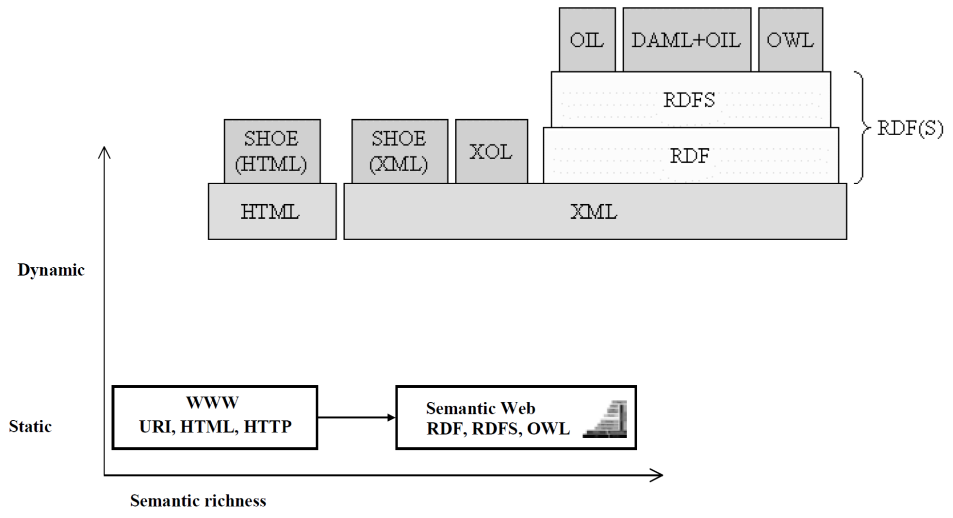


Figure 2.3: Semantic web and semantic web services (Gómez-Pérez, Fernández-López, et al., 2004)

same way or for the same purposes. Ontologies generate new knowledge via formal logical inference. KGs generate new knowledge via graph analytics, embeddings, and statistical learning. Together, they create a powerful environment in which new, reliable, semantically grounded knowledge can be produced.

2.2.3 Semantic Web

The SW is defined as an extension of the current Web in which information is given in a well-defined meaning (i.e. *ontology*), better enabling computers and people to work in cooperation. The data on the Web is defined and linked in such a way (i.e. *annotated*) that it can be used for more effective discovery, automation, integration, and reuse across various applications (Hendler et al., 2002).

Figure 2.3 depicts the World Wide Web Consortium (W3C)'s vision of the Semantic Web by offering semantic web services and semantic web languages that will support people to create data stores on the Web, build vocabularies, and write rules for handling data, hence realising a "Web of linked data". W3C's technology stack provides the following fundamental semantic web standards:

- Web Ontology Language (OWL) and Resource Description Framework Schema (RDFS) to build vocabularies or ontologies, and the Simple Knowledge Organization System (SKOS) for designing knowledge organisation systems. The Rule Interchange Format (RIF) focuses on translating between rule languages and exchanging rules across different systems.

- RDF provides the foundation for publishing and linking of web data.
- SPARQL is a query language to send queries and receive results from an RDF store (*triplestore*), e.g. through HyperText Transfer Protocol (HTTP), Simple Object Access Protocol (SOAP), or Representational State Transfer (REST).

These standards have been implemented as open-source frameworks, e.g. Jena for RDF and SPARQL, by the Apache Software Foundation. Hence, we can use these frameworks to build proof-of-concept prototypes. Linked Data (LD) is a set of web technologies based on the Semantic Web that enables the consolidation of diverse data sources and efficient querying to feed Business Intelligence processes. Linked Open Data (LOD) has LD that anyone can distribute and use freely on the Web. LOD has five main characteristics: *(i) on the Web, (ii) machine-readable data, (iii) non-proprietary format, (iv) uses RDF, and (v) is linked with RDF* (Berners-Lee, 2006).

Ontology concepts in OWL

One can learn domain knowledge from an ontology, which provides a formal, structured representation of the key concepts, relationships, constraints, and semantics that define a domain. By organising knowledge into classes, properties, and axioms, an ontology makes the domain's conceptual structure explicit and machine-interpretable, enabling both humans and automated systems to understand how entities relate, what constraints apply, and how reasoning can derive additional knowledge. In this sense, ontologies not only capture existing domain expertise but also facilitate learning, interpretation, and integration of new information within a consistent conceptual framework.

An ontology is said to have integrity when it is logically consistent, semantically coherent, and structurally well-formed. Ontology integrity is essential because the ontology is structurally well-formed and within the domain. Ontology integrity is crucial because the ontology is the domain knowledge model. If its structure is inconsistent or incoherent, any learning, by humans or machines, becomes unreliable. Ensuring integrity guarantees that the ontology remains a dependable, formally grounded source of domain knowledge.

OWL defines the following concepts to construct an ontology.

- **Classes (Concepts)** are abstract domain concepts, e.g. $owl:Class$:
- **Individuals (Instances)** are concrete members of classes, e.g. $owl:NamedIndividual$.
- **Object Properties** are relations between individuals, e.g. $owl:ObjectProperty$.
- **Datatype Properties** are relations between individuals and literal values, e.g. $owl:DatatypeProperty$.
- **Annotation Properties** are metadata such as labels, comments, or provenance, e.g. $owl:AnnotationProperty$, $rdfs:label$, $rdfs:comment$.
- **Class Axioms** are logical constraints on classes, such as equivalence and disjointness, e.g. $rdfs:subClassOf$, $owl:equivalentClass$, $owl:disjointWith$.
- **Property Axioms** are logical constraints on properties, such as domain, range, and characteristics, e.g. $rdfs:domain$, $rdfs:range$, $owl:FunctionalProperty$, $owl:TransitiveProperty$, $owl:InverseFunctionalProperty$, etc.
- **Individual Axioms** are assertions about individual identity or difference, e.g. $owl:sameAs$, $owl:differentFrom$.
- **Hierarchies (Taxonomies)** are class and property hierarchies, e.g. $rdfs:subClassOf$, $rdfs:subPropertyOf$.
- **Restrictions (Constraints)** are class conditions expressing value, cardinality, and scope constraints, e.g. $owl:Restriction$, $owl:someValuesFrom$, $owl:allValuesFrom$, $owl:hasValue$, $owl:minCardinality$, $owl:maxCardinality$, $owl:cardinality$.

- **OWL restrictions vs rules:** OWL restrictions and rule-based statements serve different but complementary roles and cannot replace one another. Restrictions define class-level constraints within the ontology and support reasoning under the open-world assumption, enabling the inference of class membership and detection of inconsistencies. Rules (e.g. Semantic Web Rule Language (SWRL) rules, rule engines like Drools, and SPARQL rules) by contrast, express conditional logic that can generate new facts, handle multi-variable conditions, and often operate under a closed-world assumption. Because restrictions govern semantic structure while rules provide procedural expressiveness beyond OWL's capabilities, each addresses a distinct aspect of knowledge representation.

Core concepts in RDF

RDF graphs and KGs are closely related but not identical. An RDF graph is a data model, while a KG is a broader concept and system that may use RDF as its underlying representation. RDF graphs define how data is represented, while KGs define how knowledge is modelled, enriched, and used. RDF defines the following core concepts to represent data or information in a graph:

- An RDF **graph** is a set of RDF triples.
- An RDF **triple** consists of three components: **subject** (an Internationalized Resource Identifier (IRI) or a blank node), **predicate** (an IRI), and **object** (an IRI, a literal, or a blank node). It is conventionally written in subject, predicate, and object order.
- **Terms** represent data in an RDF graph. A term can be an IRI, a literal, or a blank node. Therefore, IRIs, literals, and blank nodes are collectively known as RDF terms.
- An **IRI**, developed by the Internet Engineering Task Force (IETF) in Duerst et al. (2005), is an Internet protocol standard that builds on the Uniform Resource Identifier (URI) protocol and allows more Unicode characters.
- **Literals** represent values such as strings, numbers, and dates.

- **Blank** nodes are disjoint from IRIs and literals. They do not identify specific resources but say that something with the given relationships exists without explicitly naming it.

- An RDF **quad** consists of four components and is conventionally written in **subject** (IRI or a blank node), **predicate** (IRI), **object** (IRI, a literal or a blank node), and **graph** (IRI or a blank node) order.

2.3 Interoperability

Both research and industry communities have studied interoperability with mature approaches across diverse domains; as such, the term "Interoperability" is also defined in various ways. In general, the IEEE defines interoperability as *the ability of two or more systems or components to exchange information and use it.* Manso et al. (2009) refer to interoperability as "the working together of diverse autonomous entities to achieve a common goal, requiring the entities to possess the ability to exchange information about system function and domain content." The Information Technology Vocabulary International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 2382:2015(en) (ISO/IEC, n.d.) of fundamental terms defines the interoperability as, *"capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units".* Integration should not be confused with interoperability (Roberts, 2020). Also, according to ISO/IEC (1996), system integration is the progressive assembly of system components into the entire system. Whereas system interoperability refers to the ability of systems or devices within an ecosystem to exchange real-time data among systems without middleware, while interpreting incoming data and presenting it as received, preserving its original context. Unlike system integration, which combines multiple applications and devices into a single system to function as a unified whole, system interoperability requires a common communication language that enables disparate, entirely independent systems to understand and exchange data without added complexity or delay.

Brodaric et al. (2015) define data interoperability as collaboration among data providers

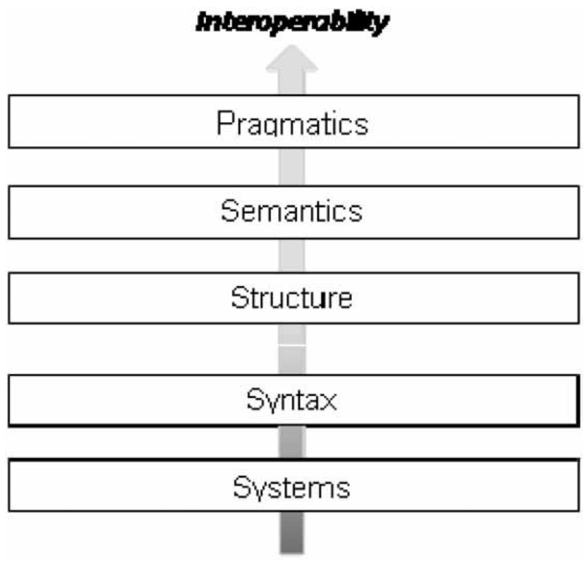


Figure 2.4: Interoperability levels for the message transformation (Brodaric, 2007)

whose goal is to exchange, deliver, or use data by sending messages in a coordinated manner. Such messages must typically be transformed at each interoperability level, either by the sender or by the receiver, into a construct that can be readily consumed and understood by the receiver. This process is often called message alignment.

Levels of Interoperability

By abstracting over the interoperability stacks presented by Manso et al. (2009) and Brodaric (2007), Figure 2.4 illustrates the common levels of data interoperability within a data network or across multiple data networks, from bottom to top.

1. **Systems interoperability** is the ability to overcome the heterogeneity of hardware or software elements required for core functions such as message passing or data manipulation, and it essentially involves platform aspects such as operating systems, transmission protocols, and specific database limitations.

2. **Syntax interoperability** is the ability to overcome differences in abstract or concrete syntaxes of languages that encode a message, including requests for data and the actual data content. The syntax defines a language's alphabet, words, and grammar.

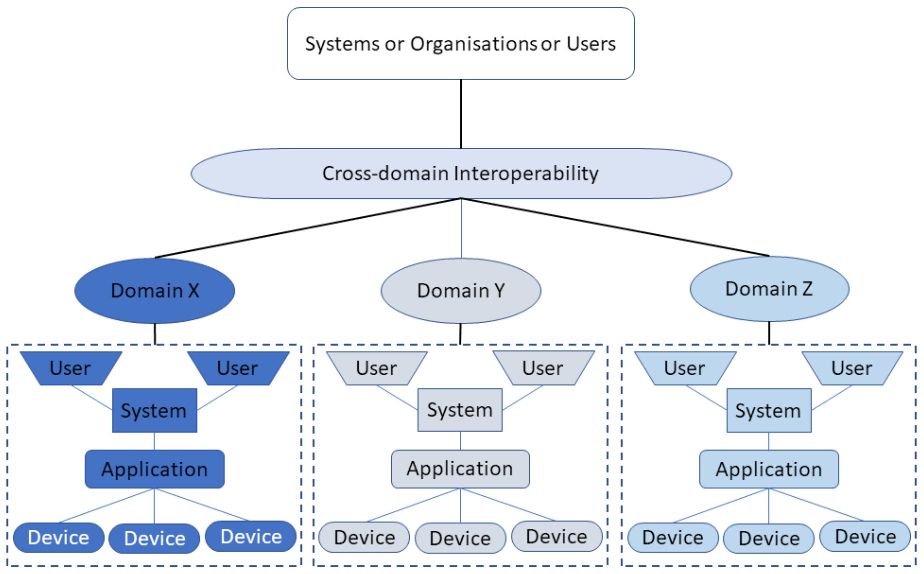


Figure 2.5: Cross-domain interoperability (Naqvi et al., 2020)

3. **Structure interoperability** is the ability to overcome diverse data structures or related web services via the alignment of the associated schemas used in messaging.

4. **Semantic interoperability** is the overcoming of inherent meaning differences (semantic heterogeneity) within components of a message's schema or data content. Inherent meaning is typically represented in digital form, such as vocabularies or ontologies, in a structured manner or as free-form text in an unstructured format.

5. **Pragmatic interoperability** is the ability to overcome contextual factors, e.g. legal, organisational, and economic, to ensure that the message sender and receiver share the exact expectations about the effects of the exchanged messages, and that the context in which this exchange occurs plays a vital role (Tolk et al., 2003).

2.3.1 Cross-domain Interoperability

According to the Network Centric Operations Industry Consortium (NCOIC), cross-domain interoperability refers to the ability of systems and organisations to interact and exchange information (interoperate) across different areas, markets, industries, countries, or communities

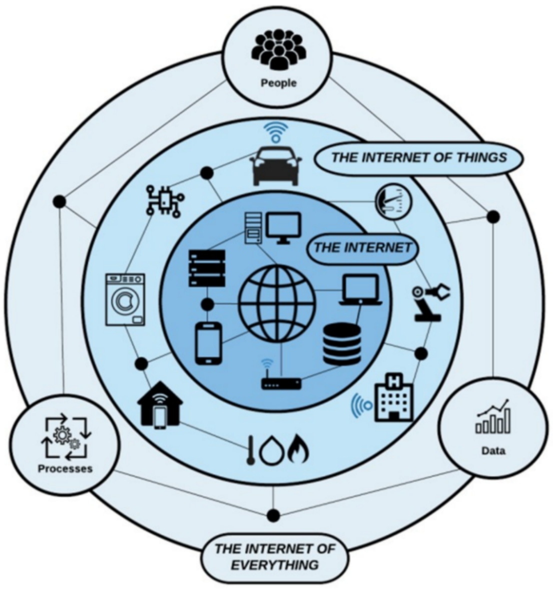


Figure 2.6: Cross-domain data interoperability (Kalatzis et al., 2019)

of interest (domains). As shown in Figure 2.5, cross-domain interoperability enables systems, users, and organisations to seamlessly communicate and conduct activities despite their reliance on different technical environments or frameworks. Figure 2.6 illustrates cross-domain data interoperability as the key enabler for the evolution of the IoT to the Internet of Everything (IoE), hence stating that the next generation of the IoT computing paradigm cannot be realised unless cross-domain data interoperability is facilitated.

2.3.2 Interoperability Models

Figure 2.7 shows one of the most prominent interoperability models for distributed heterogeneous simulation systems. In its original version, LCIM represents five levels of interoperability between two systems, ranging from no interoperability to full interoperability. Level 2 (Aligned Static Data) of LCIM addresses three of the four conflicts (semantic, descriptive, heterogeneous, and structural) that can occur when data from sources is merged. The use of *standardised references, i.e. a common ontology or a common reference model that includes all mappings to standards*, for the information elements in data can avoid conflicts (Tolk et al., 2003).

1. **Semantic conflicts** occur when the concepts of the different local schemata do not match

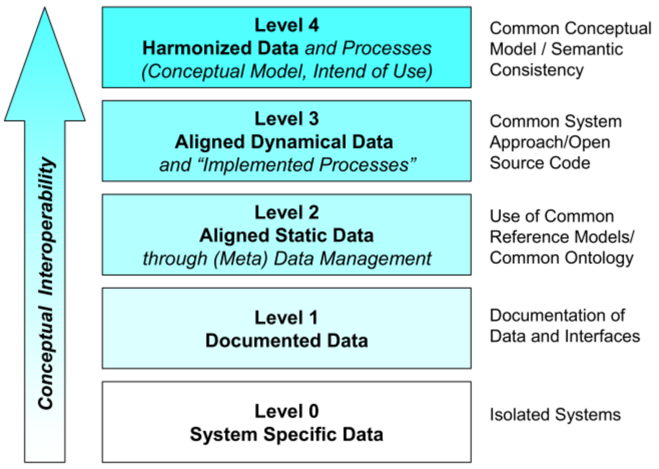

Conceptual Interoperability
Level 4
Harmonized Data and Processes
(Conceptual Model, Intend of Use)
Common Conceptual Model / Semantic Consistency
Level 3
Aligned Dynamical Data
and "Implemented Processes"
Common System Approach/Open Source Code
Level 2
Aligned Static Data
through (Meta) Data Management
Use of Common Reference Models/ Common Ontology
Level 1
Documented Data
Documentation of Data and Interfaces
Level 0
System Specific Data
Isolated Systems

Figure 2.7: Levels of the conceptual interoperability model (Tolk et al., 2003)

exactly but must be aggregated or disaggregated. They may only overlap or be subsets of each other, and so on.

2. **Descriptive conflicts** occur when homonyms, synonyms, different names for the same concept, different attributes or slot values for the same concept, etc, are used. A shared ontology can help avoid ambiguity.

3. **Heterogeneous conflicts** occur when the methodologies being used to describe the concepts differ substantially, e.g. one concept is described in the Unified Modeling Language (UML) (Object Management Group (OMG), 2021), the other in the relational data model description methodology IDEF1X (ISO/IEC/IEEE, 2021).

4. **Structural conflicts** occur when different structures are used to describe the same concept. For example, in one local schema, an attribute is used, while in the other schema, a reference to another concept is used to describe the same part of the view of "reality".

A conceptual interoperability model, as shown in Figure 2.8, comprises consecutive interoperability layers. Achieving interoperability at a higher layer is only possible if the predecessor layer offers interoperability. Since system interoperability involves hardware, software, data, and information, semantic interoperability is crucial, as is the syntax and structure of these components.

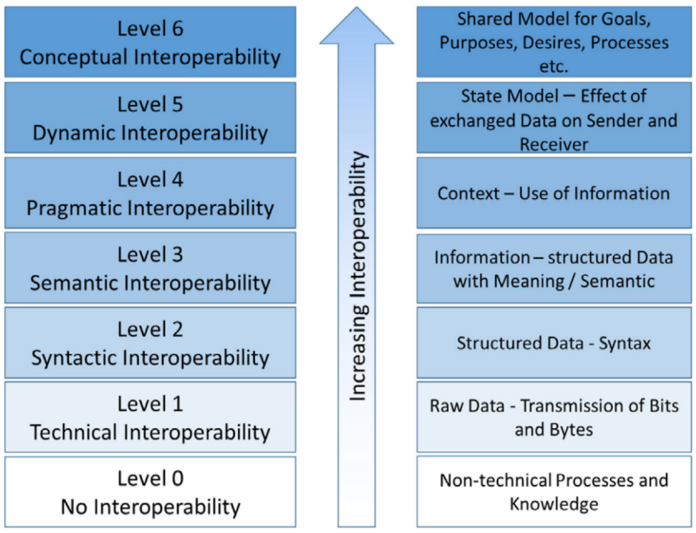

Level 6
Conceptual Interoperability
Level 5
Dynamic Interoperability
Level 4
Pragmatic Interoperability
Level 3
Semantic Interoperability
Level 2
Syntactic Interoperability
Level 1
Technical Interoperability
Level 0
No Interoperability
Increasing Interoperability
Shared Model for Goals, Purposes, Desires, Processes etc.
State Model – Effect of exchanged Data on Sender and Receiver
Context – Use of Information
Information – structured Data with Meaning / Semantic
Structured Data - Syntax
Raw Data - Transmission of Bits and Bytes
Non-technical Processes and Knowledge

Figure 2.8: Interoperability Conceptual Model (Wassermann et al., 2017)

In Figure 2.9, LCIM is presented in a newer version (Turnitsa, 2005) with seven levels, and it is compared with the OSI model (ISO/IEC, 1996). According to Wasserman and Fay, any of the OSI-based communication technologies can enable syntactic and semantic interoperability. However, achieving semantic interoperability at level 3 remains challenging. The Semantic Web (Swartz, 2013) is cited as an example in their statement (Wassermann et al., 2017).

As shown in Figure 2.7, the conceptual interoperability model comprises a series of interoperability layers. Achieving interoperability at a higher layer is only possible if the lower layer provides it. Since system interoperability involves both hardware and software, the syntactic and semantic interoperability of data and information is crucial.

In LCIM, the **Syntactic Interoperability** at level 2 defines the common structure and ensures a shared understanding of symbols for exchanging information, i.e. a common data format is used. At this level, a common protocol is used to structure the data, and the information exchange format is unambiguously defined (Turnitsa and Tolk, 2008).

If a standard information exchange reference model is used at level 3 of LCIM, then **Semantic Interoperability** is reached. At this level, the meaning of the data is shared, and the content of the information exchange requests is unambiguously defined (Turnitsa and Tolk, 2008).

However, Howell, Rezgui, and Thomas Beach (2017) recount the reasons for interoperability failure from Smart Grid (IEEE, 2011) as *(i) lack of machine communication protocols, (ii) lack*

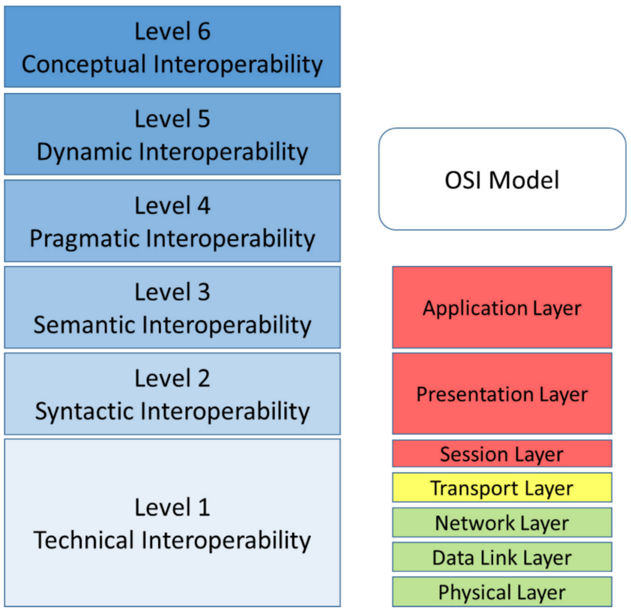


Figure 2.9: Comparison of LCIM with OSI model (Wassermann et al., 2017)

of standard data formats, and (iii) lack of ordinary meaning of exchanged content (Howell, Rezgui, and Thomas Beach, 2017). In IoT-enabled SWNs, some of the existing solutions in this direction are the HyperCat specification (HyperCat, n.d.) that can be used as a standard communication protocol to discover information about IoT assets over the web and WaterML2 (Open Geospatial Consortium (OGC), HDWG, 2014) that can be exploited as a standard data format.

Semantic models address the interoperability issue by creating a shared data format and understanding the domain. These benefits have been acknowledged in the field of semantic web technologies through the W3C's Semantic Web Stack, which highlights the role of ontologies (W3C, 2015). An ontology is a shared, formal domain conceptualisation adopted to integrate data, knowledge, and meaning across people and software components (Howell, Rezgui, Tom Beach, et al., 2016). Howell, Rezgui, and Thomas Beach (2018) present a chronological list of ontologies in the water domain. Unfortunately, there are many ontologies and standards. Still, no common ontology exists in the water domain, unlike the Gene Ontology (GO) (GO Consortium, 2020) ontology, which serves as the standard knowledge base for genes.

Maedche et al. (2000) argue that mapping existing ontologies will be easier than creating a common ontology because a smaller community is involved. They further state that ontologies must be normalised to a uniform representation to eliminate syntactic differences and make semantic differences between the source and target ontologies more apparent (Maedche et al., 2000).

Ouali et al. (2019) also propose Ontology matching as a solution to the problem of semantic heterogeneity in the integration and sharing of information *(contextualised data)*. Ontology matching is the process of establishing mappings between entities which semantically belong to different ontologies. In most ontology mapping approaches, *Ontology matching* addresses semantic heterogeneity in the integration and sharing of information. Ontology mapping employs elementary matching techniques (e.g. string-based, linguistic) to map elements by analysing entities in isolation, disregarding their relationships with other entities (e.g. father/son, brother). Ouali et al. (2019) emphasise that the structural information of an ontology is crucial for ontology mapping, as understanding the hierarchical structure of entities is often necessary to determine an entity's semantics.

2.4 Internet of Things (IoT)

Today, we live in a digitally networked world with more connected things than humans. Device networking began in 1982, when a modified Coke machine at Carnegie Mellon University was connected to a smart appliance that could report its inventory and temperature. According to Business Insider Intelligence (BII), there will be more than 55 billion IoT devices by 2025 (Newman, 2018). The Internet of Things (IoT) enables a network of physical objects connected to the Internet via various sensing equipment (sensors) to exchange information and communicate, thereby enabling intelligent identification, positioning, tracking, monitoring, and management. IoT is an extension and expansion based on the convergence of the Internet, telecommunication, radio, and television networks, and the key to triple-play (*offering data, voice, and video to a subscriber via a single telephone or cable connection*) is to realise the whole Internet Protocol (IP) of the triple play. Therefore, for IoT based on IP, a layered network communication protocol similar to the Internet Transmission Control Protocol/Internet Protocol (TCP/IP) can be used to provide services for various applications at the application layer. In contrast, the protocol allows multiple heterogeneous networks under the IP to run on the optimised network (Sun, 2020).

Communication and Protocols

The ISO has introduced a conceptual model, the OSI model, to achieve interoperability among diverse communication systems using standard communication protocols. The OSI model characterises and standardises the communication functions of a telecommunication or computing system regardless of its underlying internal structure and technology.

A *communication protocol* defines a system of rules, syntax, semantics, synchronisation and error handling to allow two or more communication system entities to transmit information via any variation of a physical quantity. Protocols may be implemented using hardware, software, or a combination of both. In a computer network of interconnected devices and applications, communication protocols can be classified into two main categories: network protocols and data protocols.

Network protocols enable networks of computing devices and applications. Each telecommunication network technology can be classified as wired or wireless, and it has network protocols that generally differentiate it from others in terms of data transmission speed, coverage, limitations, cost, availability, and dedicated applications. Wired networking is realised with technologies such as Integrated Services Digital Network (ISDN), Digital Subscriber Line (DSL), and fibre internet (Fibre to the Home (FTTH)/Fibre to the Premises (FTTP)) for stationary units, offering very high data transfer speeds. Wireless/mobile networking technologies such as Bluetooth (Bluetooth SIG Inc, n.d.), ZigBee, Wi-Fi, Near-field communication (NFC), and cellular 3G/4G/5G are used by mobile computing devices to establish networks and exchange data at variable data transfer rates with specific coverage ranges.

Data protocols enable the sharing of data among networked computing devices and applications at the application layer. In IoT platforms, commonly used data protocols are Message Queuing Telemetry Transport (MQTT), Modbus-RTU/American Standard Code for Information

Interchange (ASCII)/Transmission Control Protocol (TCP), Open Platform Communications Unified Architecture (OPC UA), Simple Network Management Protocol (SNMP), REST/JSON, Advanced Message Queuing Protocol (AMQP), Constrained Application Protocol (CoAP), Extensible Messaging and Presence Protocol (XMPP), SOAP, and Universal Plug and Play (UPnP). A comprehensive survey of IoT messaging protocols, with detailed descriptions of structure and functionality, is presented in (Al-Fuqaha et al., 2015). Data protocols support standard messaging patterns, including *publish/subscribe* and *request/response*, for exchanging data with the network. Data protocols can also be categorised into *(i) data-oriented, (ii) message-oriented, and (iii) resource-oriented* protocols (Meng et al., 2017).

In computer networks, computing devices utilise telecommunication networking technologies to establish a network. Still, there are basic types of communication connections: *(i) A point-to-point* connection allows one device to communicate with another. *(ii) A broadcast/-multicast* connection enables a device to send one message out to the network and have copies of that message delivered to multiple recipients. *(iii) The multi-point* connection allows one device to connect and deliver messages to various devices in parallel.

2.5 Smart Water Networks (SWNs)

There are many definitions of SWN in the literature. Grueau et al. (2019) cite Lee (2008) to define Cyber-physical systems (CPSs) as physical processes interacting with embedded systems in a networked environment, and cite Rasekh et al. (2016) to present SWNs as CPSs formed by the distribution system of pipes together with sensors, actuators, Programmable Logic Controllers (PLCs) and Supervisory control and data acquisition (SCADA) system (Quiñones-Grueiro et al., 2019). According to Zhen-Xing et al. (2015), a SWN is capable of monitoring/sensing with instrumentation, data management, and data analytics for proper actionable information retrieval or extraction, systematic analytics including simulation and optimisation modelling for decision-making, and finally, the automation control for triggering/communicating the instruments in the field. A truly smart water network must be 'smart' at each step to achieve optimal outcomes

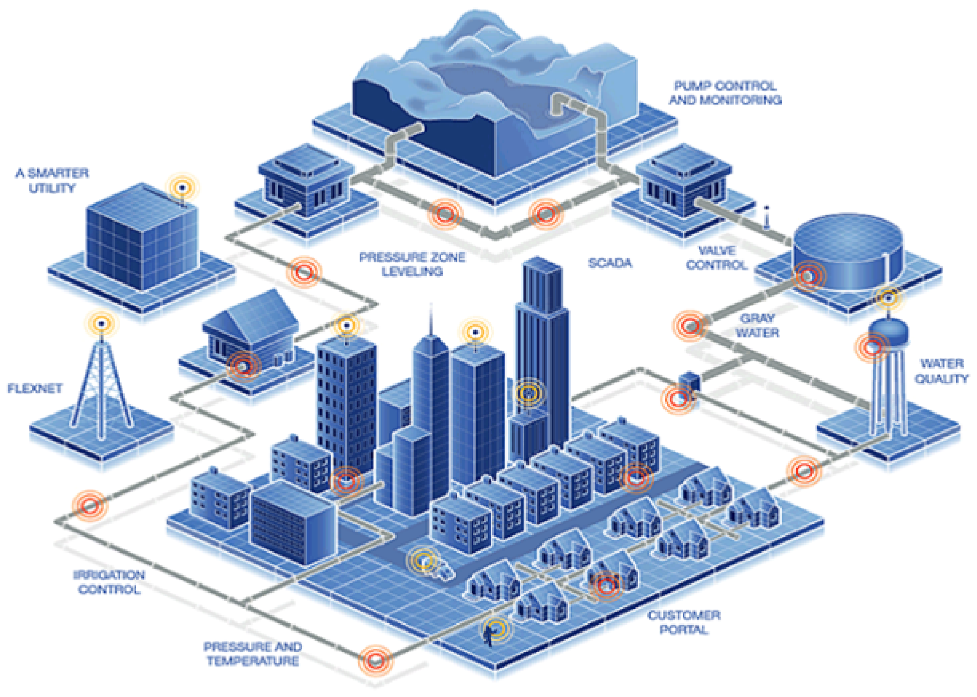


Figure 2.10: An illustration of a SWN (Sensus, 2019)

in water network management and operation (Z. Y. Wu et al., 2015). A progressively smarter SWN means that IoT role definition becomes increasingly automated, less manual, and more accurate at each consecutive step, because the network builds a cumulative knowledge base that it can reuse for future configurations.

In the industry, Sensus (Sensus, 2019) defines a SWN, as shown in Figure 2.10, as a fully integrated set of products, solutions, and systems that enable water utilities to remotely and continuously monitor and diagnose problems, pre-emptively prioritise and manage maintenance issues, and remotely control and optimise all aspects of the water distribution network using data-driven insights (Sensus, 2019).

In the water domain, a widely accepted architecture for smart water is shown in Figure 2.11. It is built from bottom to top of five layers, *(1) The physical layer* is composed of data-less physical elements with mechanical, hydraulic, or chemical functions, e.g. pipes, pumps, valves, and Pressure Reducing Valves (PRVs). *(2) The sensing and control layer* is the interface between the network operator's data systems and the physical layer that enables the connection of the "smarts" of the Smart Water Network to the real, physical network. It comprises electronic

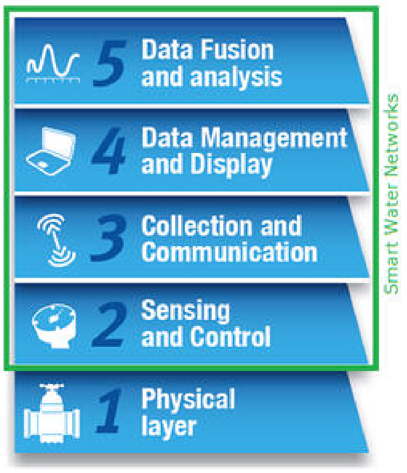

5 Data Fusion and analysis
4 Data Management and Display
3 Collection and Communication
2 Sensing and Control
1 Physical layer
Smart Water Networks

Figure 2.11: SWAN Architecture Layers (SWAN, 2016)

devices and sensors. Sensors measure various parameters (e.g. flow, pressure, water quality, reservoir levels, and water temperature) related to water delivery and distribution. Remote-controlled devices (e.g. remote-controllable pumps, valves, and pressure regulators) enable remote operation of the network. *(3) The collection and communication layer* is the interface between the underlying communications infrastructure and a human operator or other central data systems. It has two primary responsibilities: first, collecting, transmitting, and storing discrete data points; and second, enabling communication (e.g. wired and wireless network technologies) to instruct sensors and actuators on what data to collect or which actions to execute. *(4) The data management and display layer* is the interface between the underlying communications infrastructure and human operators or with other central data systems, e.g. Supervisory control and data acquisition (SCADA). This layer's data collected from various sources may be pre-processed, stored in repositories, transferred, and accessed by Geographic Information System (GIS) or network schematic visualisation tools. This is also responsible for converting human operator commands or instructions from higher-level systems into concrete device settings (e.g. switching several pumps on or off, changing valve states, etc.). *(5) The data fusion and analysis layer* is responsible for integrating raw input data and creating inferred information by applying domain knowledge. The resulting information may be displayed to a human operator, passed on to further analysis within the layer, or trigger automatic action through the data handling layer (or directly via the communications layer). Online hydraulic

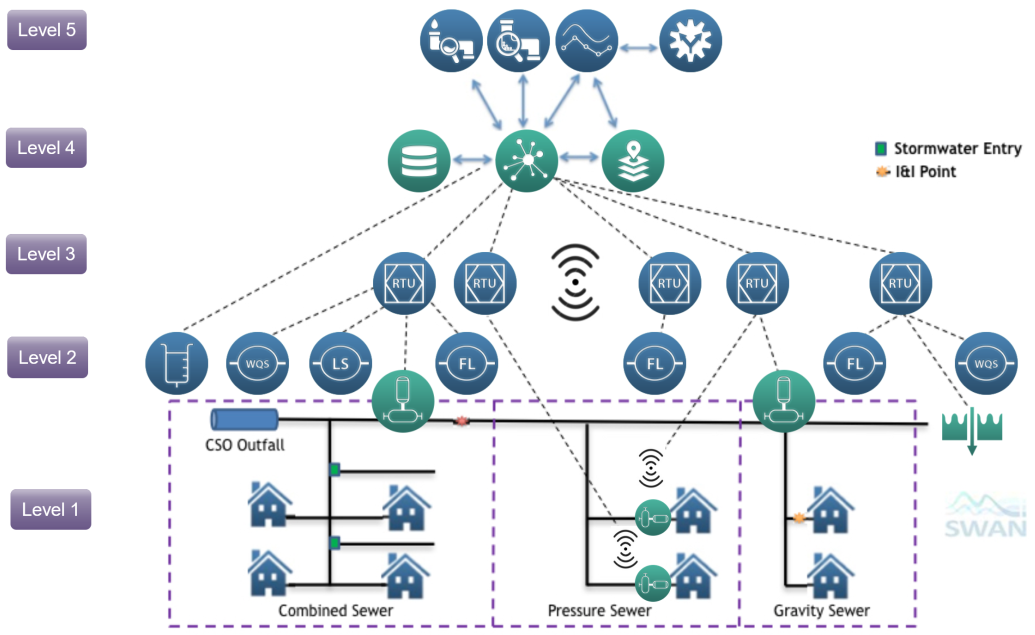


Figure 2.12: SWN architecture for wastewater-network-management (SWAN, n.d.)

modelling systems, network infrastructure monitoring, smart pressure management, smart (non-fixed-feedback) pumping or energy-optimisation systems, and DSSs are present in this layer to build a Smart Water Grid (SWG).

Table 2.2: Challenges and SWN solutions in the water domain (SWAN, n.d.)

Challenge Focus	**SWN Solution**
Leakage	Leakage Detection, Pressure Management, Water Network Management, Customer Metering
Water Quality	Water Quality Monitoring, Water Network Management, Wastewater Network Management
Energy Efficiency	Energy Management, Pressure Management, Wastewater Network Management
Bursts	Leakage Detection, Pressure Management, Water Network Management
Ageing Infrastructure	Energy Management, Pressure Management, Leakage Detection, Water Network Management, Water Quality Monitoring, Wastewater Network Management
Water Scarcity & Drought	Leakage Detection, Pressure Management, Water Network Management, Customer Metering
Apparent Losses	Customer Metering

Table 2.2 lists the significant challenges in the water domain and the possible SWN solutions developed by the SWAN Forum to address them. For example, Figure 2.12 represents a smart wastewater network management solution, in which each level represents its corresponding layer in the SWN architecture of Figure 2.11. At levels 3 and 4 of the wastewater network,

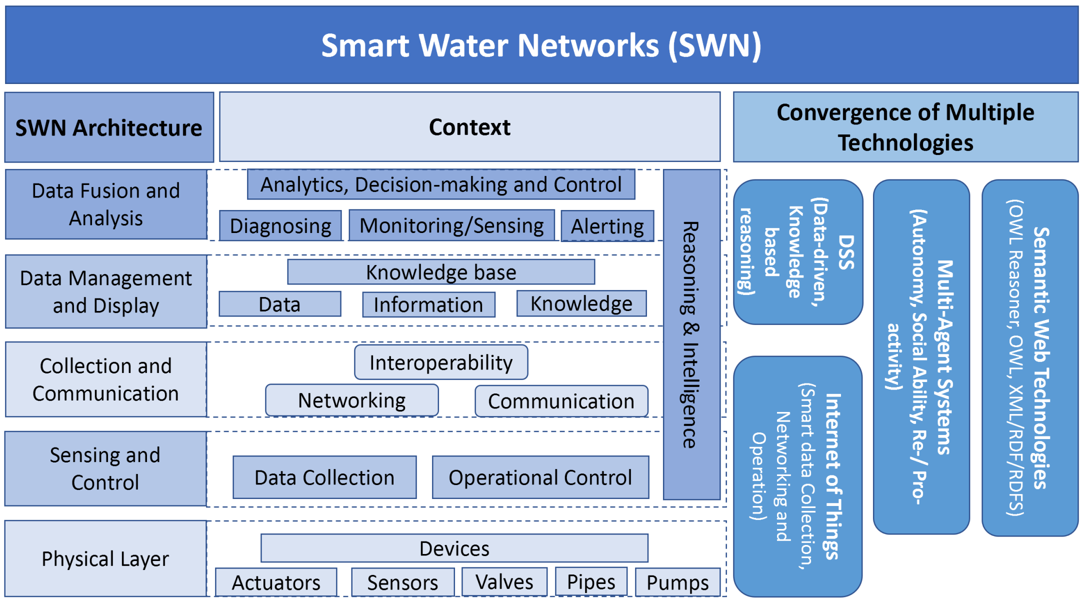

Smart Water Networks (SWN)
SWN Architecture
Context
Convergence of Multiple Technologies
Data Fusion and Analysis
Analytics, Decision-making and Control
Diagnosing
Monitoring/Sensing
Alerting
Data Management and Display
Knowledge base
Data
Information
Knowledge
Collection and Communication
Interoperability
Networking
Communication
Sensing and Control
Data Collection
Operational Control
Physical Layer
Devices
Actuators
Sensors
Valves
Pipes
Pumps
Reasoning & Intelligence
DSS (Data-driven, Knowledge based reasoning)
Internet of Things (Smart data Collection, Networking and Operation)
Multi-Agent Systems (Autonomy, Social Ability, Re-/ Pro-activity)
Semantic Web Technologies (OWL Reasoner, OWL, XML/RDF/RDFS)

Figure 2.13: Relevence of other technologies and concepts in SWN

monitoring and detecting anomalies (e.g. blockages or infiltration) helps operating companies effectively manage sewage flows at level 5 (SWAN, n.d.). At level 5, a DSS is required for wastewater management, as it enables decision-makers to utilise communication technologies, data, documents, knowledge, and/or models to identify and solve problems and make informed decisions (Power et al., 2001).

Figure 2.13 illustrates the convergence of multiple technologies in the SWN, each playing a role in different contexts and at various layers of the SWN architecture. On the left, the layered architecture of SWN is presented, along with the key context (devices, software artefacts) associated with each layer. In the right part of the figure, key technologies are identified and positioned within the layered architecture, indicating their contributions to the different SWN layers. Whilst a traditional water network infrastructure where a SCADA controls assets (e.g. mains, reservoirs, and pumps), IoT devices encapsulate these assets at the physical layer and offer data collection and operational control at the sensing and control layer in an IoT-enabled SWN. Smart IoT devices enable networking and communication with other applications and IoT devices in the collection and communication layer. Since decentralised and distributed control schemes are required to manage and operate the distributed smart devices in a SWN, the Information and Communications Technology (ICT) paradigm of Multi-Agent System (MAS) offers a promising solution for intelligent distributed control, monitoring, and automation. MAS technology can be applied in all layers of a SWN architecture when applications and devices are modelled as agents that can interact, cooperate with other agents, and change their behaviour through Artificial Intelligence (AI)-based strategies to accomplish their goal. In Machine-to-Machine (M2M) communication, smart devices and applications (e.g. decision-making and analytics) require a homogeneous machine-readable data format to cooperate, and this is where semantic web technologies (OWL, reasoners, RDF/RDFS) come in to represent, share and reason the data of a SWN. Thus, they exist in all layers of a SWN architecture. DSS can utilise semantic web technologies to build a knowledge base with ontologies authored in OWL. Collected data and real-world entities of a SWN can be stored as RDF facts. Once data and domain expert knowledge are integrated into a computational model, DSS can apply data-driven and knowledge-based approaches to reason about the collected data and to support or assist decision-making applications and human operators at both the data management and display layer and the data fusion and analysis layer.

2.6 Complexities and Specifics of IoT Application to SWNs

In a generalised IoT system, devices are often heterogeneous, loosely coupled, and designed to operate with a high degree of independence. A smart bulb, a thermostat, and a security system camera may share a network, but their operational dependencies are limited. In contrast, within a SWN, the elements are functionally interdependent and tightly coupled. Their behaviour has direct, system-level implications, e.g. a pressure sensor's readings influence pump operations or leakage detection nodes depend on correlated signals from several upstream devices. These interlinked functions, such as pressure control, flow management, leakage localisation, pump scheduling, and quality monitoring, form a coherent operational chain, rather than isolated device behaviours.

The key difference between general IoT and IoT applications in SWNs is the presence of strict domain-specific operational requirements in SWNs, including high reliability, precise temporal data, and the integration of heterogeneous sensing and control systems that support critical water infrastructure. Unlike general IoT, which typically emphasises flexible sensing and user-oriented services, SWNs demand rigorous consistency across hydraulic, quality, and asset management data streams. This distinction directly influenced the interoperability of Data and Information by requiring more substantial semantic alignment, standardised representations, and mechanisms to capture domain constraints, ensuring that data from diverse subsystems could be meaningfully integrated and correctly interpreted within water-network operations.

SWNs integrate IoT-enabled sensors, actuators, and analytics platforms to monitor, manage, and optimise water distribution, quality, and consumption. However, IoT transforms traditional water systems into cyber-physical networks. Nevertheless, this introduces the following specific technical and operational complexities:

- **Heterogeneous Sensing Infrastructure**: IoT devices in SWNs include flow meters, pressure sensors, turbidity sensors, pH meters, leak detectors, and smart valves, often from different manufacturers and using other communication standards, e.g. LoRa, NB-IoT, ZigBee, etc. This heterogeneity complicates data integration and synchronisation across the network.

- **Real-Time, Distributed Monitoring**: Water systems are geographically dispersed and require continuous monitoring for leaks, bursts, contamination, and usage optimisation. IoT devices must operate in harsh environments (underground, high humidity, electromagnetic interference), demanding robust, energy-efficient communication and fault-tolerant networking.

- **High Data Volume and Velocity**: Large-scale sensor deployment generates high-frequency data streams, e.g. flow rates, pressures, chemical properties, etc. Efficient edge processing and filtering are essential for minimising transmission overhead and facilitating real-time anomaly detection.

- **Spatio-Temporal Complexity**: Water networks have dynamic hydraulic behaviour, i.e. pressure and flow vary over time, with demand and topology. IoT-enabled SWNs must handle spatio-temporal correlations, requiring adaptive models and context-aware decision-making.

- **Interoperability and Data Standardisation**: Water utilities and municipalities often use legacy systems with non-uniform data schemas. IoT integration in SWNs must ensure semantic interoperability between these old systems and modern IoT platforms, which is one of the most complex challenges in IoT-enabled SWNs.
- **Security, Privacy, and Data Integrity**: Compromised IoT devices or manipulated sensor data can lead to wrong control decisions (e.g. misreading leaks or contamination). Secure communication protocols and trustworthy data provenance are vital.

2.7 Interoperability Approaches in IoT and Water Domains

Existing approaches to support semantic interoperability in the relevant IoT projects, e.g. Smart End-to-end Massive IoT Interoperability, Connectivity and Security (SEMIoTICS), (Milis et al., 2017), and Bridging the Interoperability Gap of the IoT (BIGIoT) (Bröring et al., 2017), propose interoperability solutions. These solutions are based on the transitive conversion model for data protocols, e.g. if MQTT can be converted to/from CoAP and CoAP can be converted to/from REST can then be converted to/from MQTT. Closer to our application scenario in the water domain, a similar interoperability approach is adopted in the water-related projects, e.g. WISDOM (Howell, Rezgui, and Thomas Beach, 2017) and Water Enhanced Resource Planning (WatERP) (Anzaldi, W. Wu, et al., 2014), where at first a base ontology, e.g. WISDOM ontology, is aligned with all possible standards and ontologies. Then it is used to convert from one standard/ontology to another. These approaches assume that an ontology of IoT data already exists or has been adopted. However, this assumption may not hold in real-world scenarios, where IoT data may be reused across different domain applications, each using different terms to label its data. IoT data can be reused across different domain applications because many IoT measurements describe fundamental physical, environmental, or behavioural phenomena that are relevant far beyond the domain in which they were initially collected. Therefore, the data is not inherently tied to a single application; instead, its value emerges from how it can be interpreted in multiple contexts. Therefore, these works do not address the problem considered in this research, including the automatic identification of potential synonymous terms among existing ontologies and standards. Furthermore, SEMIoTICS and WISDOM rely on the transitive conversion model to achieve semantic interoperability. However, this will force applications to model their data and information in their ontology.

Table 2.3 uses a simplified OSI model and specifies the applications and protocols that may be utilised in analytics communication of a SWN. The Table also describes the requirements of the corresponding application and protocols. As stated in (SWAN, 2016), no single protocol best suits all the SWN applications and communication infrastructure, and the application layer depends on the data acquisition purpose. Various application protocols are required for specific purposes, as a general application protocol would be too complex to support efficient business processes.

Table 2.3: SWN Analytics Protocols (SWAN, 2016)

Application	**Application Presentation Session layer**	**Transport Network layer**	**Requirements**
SCADA-Server/Analytics	Open Platform Communications Data Access (OPC DA), JSON, RDF, OWL, HTTP/CoAP, RESTful web services, SOAP, Web Services Description Language (WSDL), XML, CSV, Open Database Connectivity (ODBC), OGC, SensorML, WaterML2.0, OpenMI,	IPv6, IPv4, TCP, User Datagram Protocol (UDP)	Web services tend to be proprietary interfaces but can have automatic discovery; secure communication, security, backfill, redundancy, interoperability
Data logger – Server/Analytics	MQTT, HTTP, Lightweight Machine-to-Machine (LWM2M), CoAP Backfilling	IPv6, IPv4, TCP, UDP, Cellular IoT small data	Secure data exchange; fault-tolerant; command transmission (bidirectional communication);
GIS-Analytics	OGC web services, Geography Markup Language (GML), ISO19139	IPv6, IPv4, TCP, UDP	Secure data exchange, usability

Although a middleware approach, using standard protocols and interface description languages, such as CORBA, Distributed Component Object Model (DCOM), and Web Services, can help overcome communication barriers. Still, data format heterogeneity remains an issue in middleware solutions. To solve this issue, various approaches were applied: *(i) software bridges* to achieve one-to-one mapping between different protocols; *(ii) intermediary-based* solutions to achieve N-one-M mapping by using an intermediary protocol between N and M systems that employ various protocols; and *(iii) common abstractions* to enable the interoperability of legacy systems by abstracting their behaviour. DeXMS is introduced as a solution to interconnect heterogeneous Things across middleware boundaries by automating the synthesis of protocol mediators that support interconnection (Bouloukakis et al., 2019). It builds on the Data eXchange (DeX) connector model, which comprehensively abstracts and represents existing and future IoT middleware protocols. Table 2.4 compares the other related frameworks of different approaches, such as SOA to an IoT platform that abstracts Things or their data as services and offers functionalities, e.g. discovery service, the composition of services and access to services. Gateway is a common approach to connect a set of sensors and actuators that interact using Media Access Control (MAC) layer protocols, e.g. Bluetooth, ZigBee, etc., to the Internet. Cloud Computing (CC) enables IoT platforms to store, process, analyse, and retrieve vast amounts of data remotely, reliably and at low cost in a cloud environment. Model-Driven Engineering (MDE) define (i) modelling languages for specifying a system at different levels of abstraction, (ii) model-to-model transformations that translate models into another set of models, typically closer to the final system, and (iii) model-to-text transformations that generate software artefacts, e.g. source code or XML, from models. Finally, as shown in Figure 2.14, DeXMS's approach is to identify common abstract interaction types across the core interaction paradigms (Client/Server, Publish/Subscribe, Data Streaming and Tuple Space[5]) encountered in the IoT, and build a DeX Application Programming Interface (API) connector model that abstracts the underlying heterogeneous IoT protocols of a middleware.

Table 2.4: Comparison of DeXMS with related frameworks (Bouloukakis et al., 2019)

Frameworks	**Supported protocols**	**Direct bridging**	**Software abstractions**	**Constrained devices**	**Mediator synthesis**
SOA[1]	1-3	few	almost	few	no
Gateways[1]	2-4	all	none	some	no
CC[1]	2-4	all	yes	yes	no
MDE[1]	0	none	none	yes	yes
DeXMS	5	yes	yes	yes	yes

Table 2.5 compares the major European Union (EU) funded IoT research projects in terms of

[5]associative memory paradigm for parallel/distributed computing
[1]considered frameworks in (Bouloukakis et al., 2019)

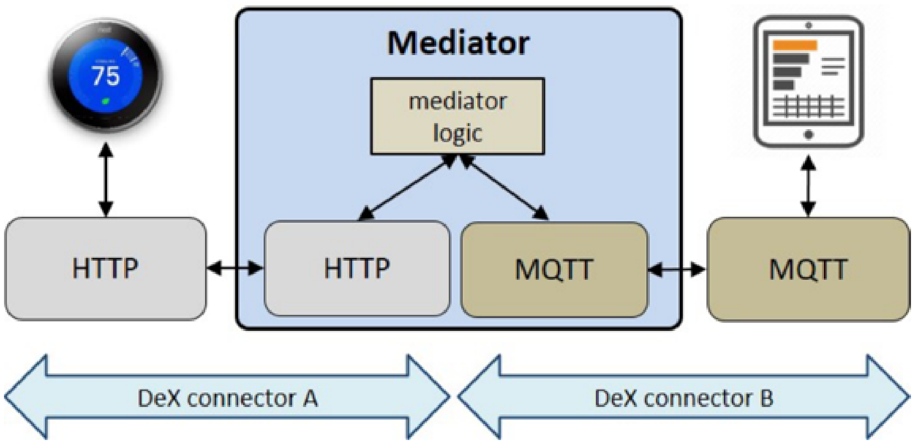


Figure 2.14: SEMIoTICS – DeXMS: Enabling Data eXchange (DeX) via Protocol Mediators (Milis et al., 2017)

interoperability features. Among the IoT projects, SEMIoTICS not only offers interoperability at four levels but also goes two steps further than its competitors. It utilises semi-automatic pattern-driven techniques for cross-domain application operation and interaction.

For IoT ecosystems, BigIoT introduces five interoperability patterns: *(i) cross-platform access, (ii) cross-application domain access, (iii) platform independence, (iv) platform-scale independence, and (v) higher-level service facades*. Although these patterns facilitate the reuse of data and services across platforms within an ecosystem, there is a need for an automatic service search and orchestration (Bröring et al., 2017).

The OpenIoT project provides an open source IoT platform that manages cloud environments for IoT "entities" and resources (such as sensors, actuators and smart devices) and enables the semantic interoperability of IoT services in the cloud. The OpenIoT cloud platform utilises the W3C Semantic Sensor Network (SSN) ontology as a common, standards-based model for the semantic unification of diverse IoT systems. It offers a versatile infrastructure for collecting and semantically annotating data from virtually any sensor. It also exploits the LD concept to link related sensor datasets and provides functionality for dynamically filtering and selecting data streams, as well as handling mobile sensors (Soldatos et al., 2015).

The INTER-IoT aims to design and implement, and experiment with, an open cross-layer

Table 2.5: Comparison of Features in IoT-Platforms (Hatzivasilis et al., 2018)

Feature / IoT-Platform	**SEMIoTICS**	**BigIoT**	**OpenIoT**	**INTER-IoT**
Technological interoperability	Yes	No	No	No
Syntactic interoperability	Yes	No	No	No
Semantic interoperability	Yes	Yes	Yes	Yes
Organizational interoperability	Yes	Yes	Yes	No
Pattern-based modelling	Yes	Yes	No	No
Pattern-based semi-automatic management	Yes	No	No	No

framework and associated methodology to enable voluntary interoperability among heterogeneous IoT platforms. It considers interoperability across all software stack layers across the domains of health and transportation/logistics (Ganzha et al., 2017). In this project, an ontology alignment format, Inter Platform Semantic Mediator (IPSM), is developed to express and perform semantic translations for both complex and straightforward alignments described in RDF format (Szmeja et al., 2018).

In comparison with IoT interoperability approaches, the WatERP framework from the water domain proposes an architecture that harmonises the communication between systems that control, monitor, and manage the water supply distribution chain by using a SOA-MAS approach together with a knowledge base driven by the Water Management Ontology (WMO) (Anzaldi, W. Wu, et al., 2014). This approach integrates and utilises innovative technologies, SOA, web services, MAS, and semantic web languages to handle the interoperability issue of monitoring and decision-making applications within SWNs, via offering a standardised SOA-MAS-based interface and communication interpretation through WMO. Additionally, through the SOA-MAS-based approach, intelligent orchestration of system functionalities within the architecture is achieved, as agents can be conceptualised with the Believe Desire Intention (BDI) (Rao et al., 1998) model to become autonomous and cooperative to achieve their declarative and procedural goals (Winikoff et al., 2002).

The WISDOM project (Zarli et al., 2014) (Cardiff University, n.d.) enables interoperability between things and software in smart water networks through a software platform that utilises ontologies for semantics and web services for web-enabled sensors, thereby integrating business operations across the water value chain. They define a water value chain as the artefacts, agents, and processes involved in delivering potable water to consumers from natural water sources

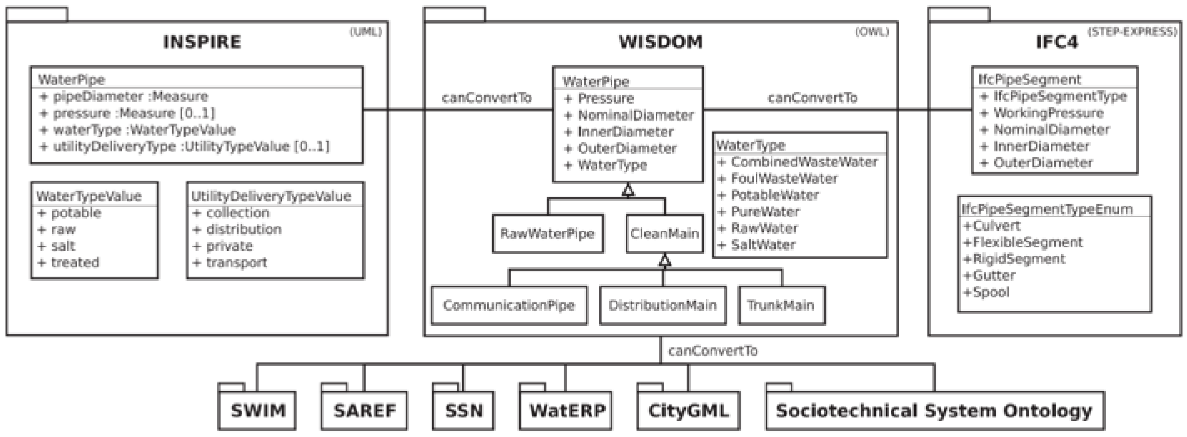


Figure 2.15: Conversion of WISDOM ontology into other standards and ontologies (Howell, Rezgui, and Thomas Beach, 2017)

and safely disposing of foul and run-off wastewater. Their approach to interoperability involves integrating existing data models, which are formalised in different data formats and often utilise heterogeneous domain perspectives. They intersect with existing models and align them with the WISDOM ontology, which serves as a common ontology to support data interoperability across these models. As shown in Figure 2.15, they promote interoperability through semantic web technologies and by performing a schema conversion from a knowledge base of devices instantiated within the WISDOM ontology into another model, e.g. SAREF, Infrastructure for spatial information in Europe (INSPIRE), Industry Foundation Classes 4 (IFC4), Semantic Water Interoperability Model (SWIM), etc. (Howell, Rezgui, and Thomas Beach, 2017).

Interoperability of applications in IoT-enabled SWN remains an issue, as current solutions do not cover all interoperability layers and do not provide appropriate semantic models for the interoperability, management, reasoning, and sharing of heterogeneous, static, and dynamic data. Currently, many frameworks focus on the bottom interoperability layers (technical, syntactic, semantic, and pragmatic). However, the top interoperability layers, which are defined differently in the literature, follows after the pragmatic layers. In the interoperability stack (see Figure 2.8), the syntactic and semantic interoperability layers serve as the bottom-level, providing a foundation for the top-level interoperability layers. As the bottom interoperability

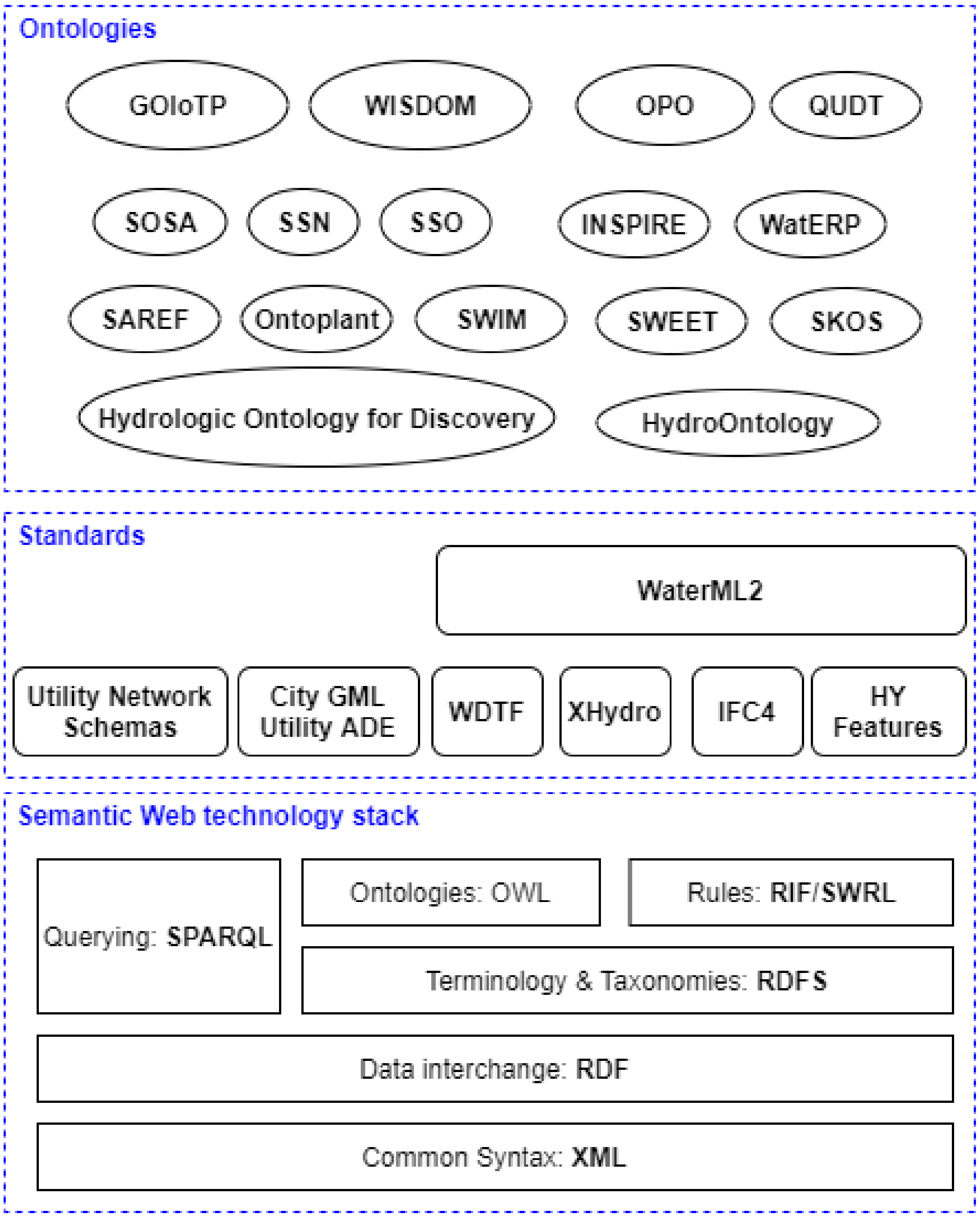


Figure 2.16: An overview of standards and ontologies used in the water domain

layers are stable and consistent, they are technical in nature. The layers above the pragmatic layer vary because they address business, organisational, strategic, and societal factors that differ across domains and research perspectives. Consequently, the literature proposes different names, scopes, and interpretations for these higher-level interoperability layers. This means that pragmatic and organisational layers cannot be fully interoperable if the syntactic and semantic layers do not sufficiently support the current standards and ontologies for the IoT and water domains. Additionally, most interoperability solutions in the water sector are developed using a vertical application approach to smart networking, thereby undermining the potential of cross-domain integration of IoT solutions. In the research IoT projects, e.g. SEMIoTICS and BigIoTb, interoperability solutions are based on a transitive conversion model for data protocols. For example, if MQTT can be converted to/from CoAP and CoAP can be converted to/from REST, then MQTT can be converted to/from REST. A similar interoperability approach is adopted in water-related projects (e.g. WISDOM and WatERP), where, initially, a base ontology (e.g. the WISDOM ontology) is aligned with all relevant standards and ontologies. It is then used to convert from one standard/ontology to another. Semantic web technologies are utilised in all projects to build semantic models with ontologies. Automation and orchestration of services using MAS have been observed in some water-related projects.

2.8 Semantic Modelling in IoT and Water Domains

Table 2.6 is an extended version of (Howell, Rezgui, and Thomas Beach, 2018), which depicts the common ontologies, formats, and standards used in the water domain to conceptualise domain knowledge. Here, we note that there is no widely agreed-upon or standard representation of these ontologies, although they utilise semantic web technologies for knowledge and information sharing. However, there have been attempts to reuse and merge existing standards and ontologies rather than to build from scratch, as listed in Table 2.6. A standard is a formally agreed specification that defines how something should be represented, exchanged, or processed in order to ensure interoperability and consistency. An ontology is a formal, explicit specification of a shared conceptualisation of a domain. It defines what exists in the domain and how concepts relate to one another in a machine-interpretable way. A standard can be transformed into an ontology, but only through deliberate semantic modelling, not mechanical translation.

Ontology development for SWNs should naturally follow established SW standards such as RDF, OWL, SPARQL, and, critically, the W3C SSN/Sensor, Observation, Sample, and Actuator (SOSA) ontologies for modelling sensors, observations, and actuators. These standards enforce a modular, interoperable, and reasoning-ready design, ensuring that SWN concepts such as pumps, valves, pressure nodes, and hydraulic behaviours can be represented consistently and reused across systems. By grounding SWN ontologies in these widely accepted foundations, developers can achieve semantic alignment with broader IoT and smart-city ecosystems, support automated inference, and avoid reinventing vocabularies, which ultimately leads to more scalable and reliable knowledge models.

Open Geospatial Consortium (OGC) Hydrology Domain Working group (HDWG) developed an international standard WaterML 2.0 (WaterML2) (OGC, HDWG, 2014) to harmonise various OGC and ISO standards. It utilises the Observations and Measurements (O&M) data model (ISO/IEC, n.d.) for its core definitions and to enable the delivery and consumption of observation data using systems that conform to the Sensor Observation Service (SOS) standard. Furthermore, it enables integrating water observation data with other domains of environmental science, such as geology and meteorology. Yu et al. (2015) present an architecture for WaterML 2.0 validation that combines document structure and semantic validation, e.g. domain and business concepts in the content.

SAREF (ETSI Technical Committee, 2020) unifies 23 ontologies by supporting alignments in systems with three or more smart appliance ontologies. The SAREF ontology has been adopted by the European Telecommunications Standards Institute (ETSI), thereby giving it significant precedence for reuse.

The SSN ontology has also been widely adopted in the water and IoT domains to describe sensors and their observations, the procedures involved, the features of interest under study, the samples used to collect them, the observed properties, and actuators. SSN adopts a horizontal

Table 2.6: A chronological list of ontologies and standards in the water domain

Name	Description	Purpose	Supported Standards & Ontologies
GOIoTP Ontology	Generic Ontology for IoT Platforms (GOIoTP) (Szmeja, 2018) is developed with OWL in 2018 as part of the INTER-IoT project (Szmeja et al., 2018).	It offers modular data structures for describing entities commonly found in IoT, facilitating interoperability among various IoT artefacts (platforms, devices, services, etc.).	SSN and SOSA
SWIM Ontology	SWIM is developed by Aquamatix (AQUAMATIX, 2017) with OWL in 2016.	It provides a device-level IoT semantic model for the water industry.	HyperCat
WISDOM Ontology	WISDOM (Cardiff University, n.d.) is developed by Cardiff University with OWL in 2015.	It was used for the Cyber-physical and social ontology of the water value chain.	INSPIRE, IFC4, SWIM, SAREF, SSN, WatERP WMO, CityGML, SSO
SAREF Ontology	SAREF (ETSI Technical Committee, 2020) is developed by ETSI in RDF/OWL and serialised in Turtle (Berners-Lee and W3C, 2014b).	It was used as the common denominator for 23 smart-appliance domain models in 2015.	contains 20 sub-ontologies
OntoPlant Ontology	OntoPlant (Sottara, 2014) is developed by (Sottara et al., 2014) in OWL.	It extends the SSN ontology to decouple control logic from equipment choices in wastewater treatment plants in 2014.	SSN
Utility Network Schemas Standard	Utility Network Schemas are developed by EC-INSPIRE (INSPIRE, 2015) in XML.	These were used for the water and sewer network model as part of a large European directive for geospatial data exchange in 2013.	Not verified yet
WatERP WMO Ontology	WatERP WMO are developed by EURECAT-WatERP (Ciancio et al., 2015) with Semantic Markup for Web Services (OWL-S).	It was used as a lightweight ontology of generic concepts for water sensing and management in 2013.	WaterML2.0, HY FEATURES, SWEET
SSN SOSA Ontology	SSN SOSA is developed by OGC W3C with OWL in 2017 (W. OGC, 2017).	They describe sensors and their observations, the involved procedures, the studied features of interest, the samples used to do so, the observed properties, and actuators.	SSN
WaterML 2.0 Standard	WaterML was developed by the OGC using XML in 2012 (OGC, HDWG, 2014).	It is a new data exchange standard in Hydrology to exchange hydro-meteorological observations and measurements. It harmonises several exchange formats for water data with relevant OGC and ISO standards.	Hydrologic, WDTF and standards of XHydro, CSIRO, CUAHSI, USGS, BOM (AU), NOAA (US), KISTERS (DE)
WDTF Standard	Water Data Transfer Format (WDTF) is developed by Australian Bureau of Meteorology with XML in 2013 (Walker et al., 2009).	It was used as a format for transferring flood warning and forecasting data to the governing body. It is the precursor of WaterML2.0.	Not verified yet
CityGML UtilityADE Standard	CityGML UtilityADE was developed by OGC with XML in 2012 (OGC, 2016).	It is a domain extension for modelling utility networks in 3D city models based on topology and component descriptions.	Not verified yet
SSN Ontology	SSN was developed by W3C with OWL in 2012 (Compton et al., 2012).	It describes sensors and sensor networks for use in web applications, regardless of the application domain.	Not verified yet
SWEET Ontology	SWEET was developed by NASA with OWL in 2011 and updated in 2019 (DiGiuseppe et al., 2014).	It is a middle-level ontology for environmental terminology.	Not verified yet
Hydrologic Ontology for Discovery	It was developed by CUAHSI with OWL in 2010 (Tarboton et al., 2010).	It supports the discovery of time-series hydrologic data collected at a fixed point. It is a precursor of WaterML2.	Not verified yet
hydrOntology Ontology	hydrOntology was developed by (Vilches-Blázquez et al., 2015) with OWL in 2009 .	It aims to integrate hydrographical data sources from a town-planning perspective using a top-down methodology.	Not verified yet

and vertical modularisation architecture, incorporating a lightweight yet self-contained core ontology, SOSA, for its elementary classes and properties.

In the WaterWorX solution for water management, SWIM, developed by Aquamatix (AQUAMATIX, 2017), enables interoperability with water-domain applications, such as pumpWorX, sewageWorX, and netWorX. SWIM comprises domain and applied ontologies that define the things and their instance types that exist in an IoT-enabled SWN.

The WISDOM project proposes a semantic model for intelligent water sensing and analytics, utilising a domain ontology created through Izssa's ontology integration approaches. The WISDOM model integrates heterogeneous data sources and various ontologies, thereby necessitating validation. At first, they validate the domain model as an accurate, sufficient, and shared domain conceptualisation by domain experts. They validate the ontology instantiation and deployment as a web service within a cloud-based platform through software testing (Howell, Rezgui, Tom Beach, et al., 2016).

WatERP WMO (Varas, 2014) is constructed to match the supply and demand in the water domain. It aligns with major ontologies, standards, and formats. Additionally, it includes concepts for observations, measurements, actions, and alerts.

In the INTER-IoT (Szmeja et al., 2018) project, GOIoTP is developed as a core ontology and reference metadata model for IoT platforms. It offers modular data structures that describe entities such as device structure, platform, observation, actuation, units, measurements, location, service, and user. Generic Ontology for IoT Platforms Extended (GOIoTPex), also developed in the INTER-IoT project, extends and fills the stub classes/concepts from GOIoTP with more specific classes, properties and individuals.

Both top-level ontologies, GOIoTP and WISDOM, bring complementary concepts and domain knowledge. Thus, to build a semantic model that represents data and information from the IoT and water domains, we will require either a unified ontology, such as combining the GOIoTP and WISDOM ontologies, or the ability to adapt to both ontologies as circumstances require. However, in both cases, an issue of ontology integration arises, and Izssa's ontology integration can solve this approach (Izza, 2009): *(i) Ontology mapping* to establish correspondence rules between concepts of two ontologies. *(ii) Ontology alignment* to bring two or more ontologies into a mutual agreement. *(iii) Ontology transformation* to change the structure of the ontology to make it compliant with another. *(iv) Ontology fusion* to build a new ontology from two or more existing ones.

2.9 Other Interoperability Approaches

Ontology learning, ontology alignment, and semantic similarity techniques jointly underpin semantic interoperability in complex, heterogeneous domains. Ontology learning accelerates the construction and evolution of domain ontologies by automatically extracting concepts and relations from unstructured or semi-structured data, thereby providing a shared conceptual foundation. Semantic similarity methods quantify relatedness between terms or structures, enabling the detection of meaningful correspondences across independently developed models. Ontology alignment operationalises these signals by establishing explicit mappings between ontologies, allowing heterogeneous systems to interpret each other's data and services consistently. Together, these approaches enable scalable, adaptive, and semantically coherent integration across IoT and other multi-domain environments. The following subsections discuss these interoperability approaches.

2.9.1 Ontology learning

Ontology learning is the (semi-)automatic process of extracting ontology components, terms (concepts), taxonomies, non-taxonomic relations, axioms, and constraints, from structured, semi-structured, or unstructured sources (text, KGs, databases, logs). Its goal is to accelerate ontology engineering while ensuring outputs are usable for downstream reasoning and integration.

Ontology learning (ontology extraction, generation, or acquisition) employs techniques such as machine learning, knowledge acquisition, NLP, statistics, and information retrieval to extract concepts and their relations (Park et al., 2010). Combining these techniques, e.g. natural language's standard pattern matching with machine learning, also enables ontology engineers to extract concepts and their relations (Daelemans et al., 2004). Typically, ontologies can be generated from various data types, including textual data, dictionaries, knowledge bases, semi-structured schemas, and relational schemas. Most works on automatic ontology construction have focused on extracting ontologies from text (Tao et al., 2009). Some tools, such as OntoLT (Buitelaar, Olejnik, et al., 2004), initially involve extracting terms from a domain-specific corpus. Then, the extracted terms are clustered to identify potential classes and taxonomies. Relations can also be extracted by computing a statistical measure of connectedness between identified clusters (Park et al., 2010). Asim et al. (2018) present a recent survey of ontology learning techniques and applications.

Although older, a foundational survey (Gómez-Pérez and Manzano-Macho, 2004) provides a structured taxonomy of ontology learning techniques (linguistic, statistical, and machine learning), discusses tools, and compares methods for semi-automated ontology construction. Useful for historical context and layering classical methods with modern Large Language Model (LLM) approaches.

2.9.2 Ontology Matching

Ontology matching is a crucial component in establishing semantic interoperability. One of the main challenges in ontology matching is semantic heterogeneity, i.e. the differences in modelling between the two ontologies to be integrated. The semantics within most ontologies or schemas are typically incomplete because fully specifying all conceptual, relational, and contextual knowledge for a domain is both practically infeasible and computationally undesirable. Domains evolve, perspectives differ, and many forms of knowledge cannot be exhaustively formalised. Therefore, external background knowledge plays a significant role in (semi-)automated ontology and schema matching (Portisch et al., 2022).

Ontology matching is the non-trivial task of finding correspondences between entities of two or more given ontologies or schemas. It is integral to ensuring semantic interoperability. Matching can be performed manually or through an automated matching system. Ontology matching is a problem for Open Data (e.g. matching publicly available domain ontologies or interlinking concepts in the Linked Open Data Cloud (Linked Open Data Cloud, 2007)) and for private companies that need to integrate disparate data stores for transactional or analytical purposes. (Portisch et al., 2022)

Ontology matching has matured considerably, supported by stable heuristics, established toolkits, and systematic benchmarking through initiatives such as the Ontology Alignment Evaluation Initiative (OAEI). However, the literature shows that progress has plateaued: most systems remain overly dependent on lexical similarity, struggle with sparse or noisy metadata, and perform inconsistently on real-world, heterogeneous ontologies. While recent embedding-based and LLM-augmented approaches improve recall and handle deeper semantic variation, they pose challenges for explainability, computational cost, and susceptibility to errors when not constrained. Current evidence suggests that the most effective approach is a hybrid one: combining symbolic heuristics, ontology-aware embeddings, and selective human or LLM assistance, alongside improved evaluation on industrial datasets. Overall, ontology matching remains essential for semantic interoperability, but its practical deployment requires more robust methods, better provenance handling, and lifecycle-oriented alignment processes.

In domains beyond NLP, ontology similarity primarily relies on structural, logical, instance-based, and embedding-driven features rather than on lexical cues. Structural approaches compare class hierarchies and graph topologies, whereas logical similarity assesses shared axioms, properties, and constraints. Instance-based measures assess the overlap or distribution of associated data, which is particularly relevant in IoT or industrial ontologies. Embedding methods capture latent patterns across large, heterogeneous ontologies, enabling scalable similarity computation. Domain-specific knowledge, such as functional roles, measurement units, and operational constraints, further informs similarity assessment. Combining these signals is essential for robust alignment, interoperability, and knowledge integration in technical and industrial applications.

Table 2.7: String similarity algorithms

Name	**Type**
Levenshtein	Edit distance based
Hamming	Edit-distance-based
Jaro-Winkler	Edit-distance-based
Jaccard index	Token-based
Sorensen-Dice	Token-based
Ratcliff-Obershelp	Sequence-based

2.9.3 Similarity Approaches and Algorithms

String-search Matching (SSM) is a lexical-level technique used in NLP pipelines to detect exact or approximate occurrences of predefined textual patterns efficiently. While it does not model linguistic semantics, it plays a critical role in preprocessing, rule-based extraction, and candidate identification.

Within NLP, string similarity algorithms operate at the lowest level of linguistic analysis, focusing on character- or token-level representations rather than syntactic or semantic structures. As such, they are typically employed as preliminary or auxiliary methods rather than stand-alone solutions for language understanding. String similarity algorithms quantify similarity between textual strings based on character- or token-level overlap, edit operations, or shared substrings, without modelling linguistic meaning or context. Table 2.7 lists some of the well-known NLP algorithms for calculating similarity. These algorithms can be categorised into edit-distance-based, token-based, and sequence-based.

Edit-distance-based

These algorithms compute the number of operations, such as insertion, deletion, or substitution of a single character, or transposition of two adjacent characters, required to convert one string into another. As the number of performed operations increases, the similarity between the two strings decreases. Some of the commonly used edit-distance algorithms are as follows:

- The Levenshtein algorithm (Kessler, 1995) is a string edit distance measure that quantifies the distance between the pronunciations of corresponding words in different dialects or closely related languages. It computes the minimum cost of transforming one string of segments into another via insertions, deletions, or substitutions (Beijering et al., 2008).

- The Hamming distance algorithm calculates the distance between two strings of equal length as the number of positions at which the corresponding symbols are different. Therefore, it measures the minimum number of substitutions required to transform one string into the other, or the number of errors needed to do so (Pilar Angeles et al., 2015).
- Jaro-Winkler's algorithm gives high scores to two strings if (1) they contain the same characters, but within a certain distance from one another, and (2) the order of the matching characters is the same. To be exact, the distance of finding a similar character is less than half of the length of the longest string.

Token-based

These algorithms assume the input is a sequence of tokens rather than a complete string. Therefore, they convert strings into token sets (e.g. n-grams) by splitting them with delimiters and identifying similar tokens across the two sets, regardless of token length. The greater the number of common tokens, the greater the similarity between the sets.

Sequence-based

These algorithms attempt to identify the longest common substring between two strings and evaluate equal-length character combinations with equal importance. The greater the number of common sequences, the higher the similarity score.

Embedding-based semantic similarity

Embedding-based semantic similarity approaches represent words, phrases, or texts as dense vectors learned from large corpora, such that semantic relatedness is reflected by geometric proximity in an embedding space. In 2013, Google launched **Word2Vec** as an open-source toolkit for generating word vectors and mathematical representations. In general, a word is represented as a vector, enabling the computer to identify and process it for subsequent tasks efficiently. There are two methods for representing words as vectors: One-Hot Representation and Distributed Representation. For example, "happy", One-Hot is expressed as [0,0,0,1,0,. . .], and Distributed is described as [0.25,1.63,-0.62, 0.76,0.39,. . . .]. When dealing with large amounts of data, using One-Hot encoding can lead to a dimensionality disaster, treating any two words as completely isolated from one another, lacking semantic connections. In contrast, distributed representations reduce the dimensionality of the high-dimensional vocabulary space to a relatively low one. This approach mitigates the curse of dimensionality and reveals relationships among words. Word2Vec utilises the n-gram model, which assumes that a word is only related to the surrounding n words and is not affected by other words in the text. It calculates semantic similarities using word-similarity. Word2Vec primarily uses the Continuous Bag-of-Words (CBOW) and Skip-gram models for training. Additionally, it uses Hierarchical Softmax and negative sampling to improve processing speed (Jin et al., 2018).

String similarity and embedding-based semantic similarity represent complementary approaches to textual comparison. String similarity methods operate at the lexical level, comparing surface forms based on shared characters or tokens, and are computationally efficient and interpretable but insensitive to meaning. In contrast, embedding-based semantic similarity methods encode text into dense vector representations learned from large corpora, thereby capturing semantic relatedness, synonymy, and contextual meaning, but at the cost of higher computational complexity and reduced interpretability. Consequently, many NLP systems integrate both approaches to balance efficiency and semantic coverage.

2.10 Knowledge-Driven Water DSS

In recent years, ICT and innovative technologies, such as IoT and SW, have been applied to SWNs to enable intelligent sensing and smart water management. A Decision Support System (DSS) plays a crucial role in the operation of clean and wastewater networks and the management of assets in a more efficient, sustainable, and reliable manner. Many research projects and initiatives, such as ICT4Water, EIP-water, WISDOM, and WatERP, have been launched under the

European Commission Seventh Framework Programme (EC FP7) to investigate the challenges and impacts of integrating these technologies into SWN. In the Horizon 2020 programme, there are also several projects to address interoperability in IoT and water domains, such as INTER-IoT, Ocean Data Interoperability Platform (ODIP-2), Worldwide Interoperability for SEmantics IoT (Wise-IoT), Bridging the Interoperability Gap of the Internet of Things (BIG IoT), and Smart End-to-end Massive IoT Interoperability, Connectivity and Security (SEMIoTICS).

The SWAN Interoperability Workgroup emphasises the need for pervasive interoperability to integrate and implement innovative technologies in SWNs, given the diverse range of communication protocols used in smart water applications (SWAN, 2016). The primary cause is the retrofitting of new smart applications onto existing/proprietary network management systems, which were designed from an automation/vertically integrated perspective rather than with cross-domain interoperability in mind.

Howell, Rezgui, and Thomas Beach (2018) propose a water knowledge management platform that extends the Internet of Things towards a Semantic Web of Things by leveraging the Semantic Web to address the heterogeneity of web resources. The platform enables data-driven and knowledge-based decision support through programming interfaces and a comprehensive, rule-based ontology that encodes domain concepts, relationships, and operational rules for automated reasoning. The model-driven DSS approach focuses on analysing data stored in databases or warehouses using GIS functionalities, On-Line Analytical Processing (OLAP), or quantitative models to extract patterns. The knowledge-driven DSS approach utilises a knowledge base to reason about a problem to find a solution by using an inference engine (Serrat-Capdevila et al., 2011).

Urbani et al. (2006) propose a generic decision support system framework based on MAS and GIS. Anzaldi, W. Wu, et al. (2014) identify the issue of not combining multiple inference engines in a single tool, instead of handling different water management situations with specific reasoning models or procedures. The HydroLOGIC system was designed in Australia primarily to evaluate the consequences of various irrigation strategies and to explore options for optimising yield and Water Use Efficiency (WUE) at the field level in cotton (Richards et al., 2008). It integrates knowledge and information from other platforms and systems, supporting many standardised ontologies and data formats aligned with OGC® (Anzaldi, Rubion, et al., 2014).

Table 2.8 lists the DSSs used in wastewater treatment plants to support decision-making on quality, operational, design, energy, and sustainability aspects. Mannina et al. (2019) give a comprehensive review of DSS and classify them into four main types: Life Cycle Assessment (LCA), Mathematical Models (MM), Multi-Criteria Decision Making (MCDM), and Intelligent Decision Support Systems (IDSS).

According to Escobar Esteban et al. (2020), the next generation of decision-making software tools in SWNs requires integrating multiple, heterogeneous data sources across different knowledge domains to efficiently and sustainably maintain water reservoirs and supply networks. To address this challenge, they propose utilising LD and SW to harmonise data from various sources, query efficiently, and feed the results into the upper-level Business Intelligence (BI) processes (Escobar Esteban et al., 2020).

Data generated in IoT-enabled SWNs is not only heterogeneous but also of a highly dynamic nature, as mobile smart sensors or IoT devices, particularly those with wireless connectivity, may continuously send data streams while connected or abruptly stop transmission when disconnected from the network. Therefore, to bridge the gap between reasoning over data and data stream processing, stream reasoning is required (Margara et al., 2014). Margara et al. (2014) present models as research areas for representing, processing, and retrieving information in stream reasoning systems. Representation models are time, historical, and uncertainty models. Processing models include query, reasoning, and uncertainty-propagation models. Retrieval models handle large, dynamic, and distributed data. While reviewing the state-of-the-art stream reasoning, Dell'Aglio et al. (2017) conclude that stream reasoners should offer richer query languages, which include a wider set of operators to encode user needs and the engine to evaluate them. Reasoners are now capable of deductive and inductive reasoning techniques. Reasoning

[1]Design
[2]Energy consumption
[3]Operational optimization
[4]Improvement of the effluent Quality
[5]Environmental Sustainability

Table 2.8: List of applied DSSs in WWTPs around the world (Mannina et al., 2019)

DSS Type	**Scope**	**Application Description with References**.
IDSS	O^3	Supervision of a WWTP located in the Barcelona region, Catalonia (Pascual Pañach et al., 2018)
IDSS	D^1	Optimal design of WWTPs to reduce resources and operational costs (Ye et al., 2021)
IDSS	S^5	The IDSS has been applied to the Danube River. Consequently, the WWTPs effluent quality has been optimised through IDSS (Oprea, 2018)
IDSS / MCDM	E^2	Two real conventional activated sludge systems (CAS) WWTPs in Germany and the Netherlands (Torregrossa, Hernández-Sancho, et al., 2017)
IDSS / MM	Q^4	Real WWTP of Tabriz, Iran (Nadiri et al., 2018)
LCA	S^5	Real WWTP located in Copenhagen, Denmark (Yoshida et al., 2014)
LCA	O^3 S^5	Applied to Betanzos and Calafell WWTPs, both located in Spain. (Lorenzo-Toja et al., 2016)
LCA	S^5	Applied to Tarragona WWTP, Spain (Pintilie et al., 2016)
LCA / MCDM	D^1	Applied to two different WWTPs (La Garriga and Granollers), located in Spain (Morera et al., 2015)
LCA / MM	E^2 S^5	Real wastewater infrastructure of Delhi, India (P. Singh et al., 2018)
LCA / MM	O^3	Plant data were generated with the STOAT simulator, which has been set up to replicate the operational conditions of the WWTP of Solingen-Burg, Germany (Torregrossa, Marvuglia, et al., 2018)
MCDM	S^5	Applied for extensive technologies(constructed wetlands and pond systems) and intensive technologies (extended aeration, membrane bioreactor, rotating biological contactor, trickling filter, and sequencing batch reactor) (Molinos-Senante et al., 2014)
MCDM	D^1	Applied to a laboratory scale municipal WWTPs (Bertanza et al., 2015)
MCDM	S^5	Large WWTP, which serves 1,000,000 person equivalents, to enable the exploration of a wide variety of alternatives(Garrido-Baserba et al., 2015)
MCDM	S^5	Two case studies for the application of several scenarios: 1) selection of technology for an upcoming township project in Mumbai, India; 2) lake rejuvenation project in the suburbs of Thane, India (Kalbar et al., 2016)
MCDM	D^1	It presents a conceptual DSS to assess fit-for-purpose wastewater treatment and reuse, and is applied to a hypothetical case study. (Chhipi-Shrestha et al., 2017)
MCDM	D^1	Applied for two extensive technologies (constructed wetlands and pond systems), and five intensive technologies (extended aeration, membrane bioreactor, rotating biological contactor, trickling filter, sequencing batch reactor (Arroyo et al., 2018)
MCDM	O^3	Real WWTP located in Whyalla, South of Australia (Chow et al., 2018)
MCDM	D^1	Real WWTP of Minnesota, United States (Xin et al., 2018)
MM	S^5	Real advanced hybrid WWTP (Kyung et al., 2015)
MM	O^3	Hypothetical structure as the catchment described in ATV A 128 (ATV, 1992) (Saagi et al., 2016)
MM	O^3 Q^4	China's urban WWTPs (Zeng et al., 2017)
MM	O^3	Real WWTP located in the province of Alicante, Spain (Díaz-Madroñero et al., 2018)
MM	S^5	Thirty small WWTPs from Spain were sampled between 2014 and 2016, featuring three different secondary treatment technologies: CAS system, rotating biological contactors (RBC) and trickling filters (TF) (Gémar et al., 2018)
MM	S^5	Real data obtained from WWTP located in the Lake Taihu region, China (Jiang et al., 2018)
MM	Q^4	Seawater obtained from a clean coastal site in Saint John's, Canada (Jing et al., 2018)

frameworks should be scalable and able to integrate and reason over vast amounts of heterogeneous data while meeting time requirements. Additionally, the reasoner must handle issues such as noise (faulty or inaccurate sensor outputs) and heterogeneity. The Internet of Things and Industry 4.0 could serve as the real-world application domains for stream reasoning. They also stress the development of benchmarking and evaluation activities to compare and contrast current solutions (Dell'Aglio et al., 2017).

Table 2.9: Comparative analysis of survey or overview studies in IoT-based water distribution systems (Pagano et al., 2025)

Secondary study	Background of Water Distribution Systems (WDS)	Overview of IoT	Challenges in IoT-based WDS	Overview of IoT for leakage detection	Focus on edge computing for WDS	Massive IoT scenario in WDS	Guidelines for massive IoT-based WDS
(Lalle et al., 2021)	x	✓	✓	✓	x	x	x
(Oberascher et al., 2022)	x	✓	✓	x	x	x	x
(Ismail et al., 2022)	x	x	✓	x	x	x	x
(Yuan et al., 2019)	x	x	x	✓	x	x	x
(Li et al., 2020)	x	x	✓	x	x	x	x
(Abu-Bakar et al., 2021)	x	x	x	x	x	x	x
(Velayudhan et al., 2022)	✓	✓	✓	✓	✓	x	x
(Islam et al., 2022)	x	✓	x	✓	x	x	x
(Pagano et al., 2025)	✓	✓	✓	✓	✓	✓	✓

Pagano et al. (2025) review the integration of IoT and Edge Computing (EC) technologies into WDS, with a focus on large-scale implementations. By analysing 255 studies, they identify key challenges: interoperability, scalability, energy efficiency, network coverage, and reliability. Their paper emphasises the role of edge computing in processing data locally to reduce cloud dependency, improve scalability, and support sustainable smart-city and digital-twin applications. Table 2.9 provides a comparative analysis of studies on IoT-based WDS.

2.11 Interoperability challenges in IoT-enabled SWNs

The literature review has made it clear that interoperability of SWN applications enabled by IoT requires interoperability at the syntactic (data exchange) and semantic (understanding the meaning of exchanged data) layers to overcome the syntactic and semantic heterogeneity across different data sources. For example, Industry 4.0 (Schwab, 2017) cannot yield the potential of interconnected IoT if the data sent and received by IoT cannot be understood and used by applications. Some of the challenges that need to be addressed to achieve interoperability in the water domain are:

1. **No common standard ontology leads to the adaptation of existing ontologies:** In the water domain, numerous domain- and application-specific ontologies exist, but there is no common standard water ontology, as in medicine with the Gene Ontology (GO Consortium, 2020). As discussed in Section 2.2.2, an ontology models only specific parts or aspects of the real world, and the ontology engineer's interests and viewpoint limit the models or ontologies the engineer creates. Another reason is that reusing existing ontologies is not widely practised, as extending or merging them is a complex task in ontology engineering. Therefore, each application typically builds its own application-specific ontology. A consortium of organisations and companies from the public and private sectors is needed to make a common ontology. In this context, applications fail to adapt existing ontologies. Thus, they cannot semantically understand the data shared by other applications without semantic mappings.

2. **Transformation of data in various formats:** SWN applications generally encode/represent data in their preferred format (e.g. CSV or JSON) and also publish data in that format. Therefore, the data encoding format of one application may differ from that of another application that wants to share it. Thus, both applications must be able to translate/serialise others' data formats, and this capability must be implemented and deployed. However, covering many data encoding formats will become challenging as applications interoperate and use different data representation formats.

3. **Adaptation of water standards:** WaterML2 (OGC, HDWG, 2014) is an XML-based standard that was developed by the OGC group (CSIRO, CUAHSI, USGS, BOM, NOAA, KISTERS, and others) to standardise data (hydro-meteorological observations and measurements) exchange in Hydrology. However, as listed in Table 2.6, many standards and data models were developed before the publication of WaterML2 and do not directly support it. Adapting standards such as WaterML2 remains challenging for water data collected and stored in traditional ways using the Relational Database System (RDBS).

4. **Generation of missing semantic models (ontologies):** Semantic interoperability requires understanding data through conceptual knowledge, which is generally represented in ontologies. However, these ontologies are mainly absent from existing databases because ontology development remains a cumbersome, manual task. In addition, these ontologies must be developed manually, considering the schemas of the represented data across different formats, which helps identify the data's structural organisation. Ontologies are semantic models that define domain concepts, relationships, and rules, typically stored in repositories or knowledge bases rather than traditional databases. Beyond manual engineering, ontologies can be developed using ontology learning, KG–driven induction, reuse of existing ontologies, or LLM-assisted bootstrapping. These approaches accelerate development while preserving domain relevance and enabling automated reasoning.

5. **State-of-the-art technologies in the water domain** are missing to achieve interoperability. For example, knowledge graphs can be used to represent data. On the one hand, KGs

can accommodate various aspects of data, including measurement data, data similarity, and links to existing entities.

6. **Data alignment requires different algorithms** to be considered in various situations. A tool can align the same data to a different dataset, ontology, or standard only if it can accommodate multiple matching algorithms and reasoning mechanisms for computing similarity scores under alternative settings.

7. **Further enrichment of openly accessible IoT data with metadata**, e.g. in supported formats, standards, or ontologies, is missing to quickly enable interoperability between machines and humans in a runtime scenario. In most cases, IoT data is not interoperable because it is typically presented in semi-structured formats, such as CSV or JSON, for practical reasons. Therefore, it cannot be further enriched with metadata without editing the data structure or model. However, if IoT data is represented as a graph, it can be easily enriched with metadata without altering the structure of the original data.

8. **Interoperability-specific domain knowledge** is still not widely known and is limited to human domain experts only. For example, which standards or ontologies exist in the IoT and water domains, and which standards or ontologies support each other? This hinders the software applications' ability to adapt to new standards and ontologies at runtime.

2.12 Research gap in IoT-enabled SWNs

The integration of IoT technologies into SWNs has led to highly heterogeneous ecosystems comprising diverse sensors, communication protocols, data models, legacy supervisory systems, and domain-specific standards. The literature consistently identifies interoperability, particularly at the semantic level, as the principal barrier preventing seamless data exchange, cross-domain integration, and coordinated decision-making across SWN applications. Although substantial research has been conducted on IoT interoperability, semantic modelling, and water-domain data representation, several significant gaps remain.

First, the water domain lacks a commonly agreed-upon, unified ontology. Existing ontologies and standards, including SSN, SAREF, SWIM, WISDOM, and WaterML2.0, among others, provide valuable domain perspectives but remain fragmented and unaligned. Unlike mature domains with widely accepted reference ontologies (e.g. the Gene Ontology), the water sector lacks a consolidated conceptualisation. This fragmentation hampers semantic consistency, restricts cross-platform reuse of data, and complicates automated reasoning across SWN applications.

Second, current interoperability frameworks assume the existence or adoption of a base ontology to mediate between domain models. Major IoT-focused projects, such as SEMIoTICS, BIGIoT, and INTER-IoT, as well as water-oriented projects like WISDOM and WatERP, rely on this assumption when implementing transitive data conversion or ontology alignment. In practice, however, IoT data are generated, reused, and exchanged across multiple domains. Each domain employs different terminologies, naming conventions, and modelling assumptions. Consequently, these approaches do not adequately address the challenge of identifying meaningful correspondences or synonymous concepts across heterogeneous standards in real-world deployments.

Third, prevailing interoperability solutions in both IoT and water domains rely heavily on transitive conversion models for protocol translation or ontology transformation (e.g. MQTT → CoAP → REST, or ontology A → core ontology → ontology B). While effective in constrained use cases, these models suffer from poor scalability, a strong dependence on intermediary schemas, and a tendency to force applications into platform-specific representations. This undermines their suitability for dynamic, cross-domain data exchange in large-scale SWNs.

Fourth, existing solutions predominantly adopt vertical, application-specific approaches that support interoperability within isolated water-domain use cases but fail to extend to broader IoT ecosystems. Such siloed architectures prevent meaningful integration across domains such as energy management, urban infrastructure, and environmental monitoring. This is an increasingly important requirement for next-generation, cross-sectoral IoT deployments.

Fifth, despite advances in ontology matching and schema alignment, a notable absence remains of automated or semi-automated methods for discovering synonymy, hierarchy, or structural relationships among concepts across water and IoT ontologies. Most current approaches rely on manual curation or partial matching techniques and do not exploit lexical, structural, or graph-based semantics in a comprehensive or scalable manner. This limits their ability to harmonise heterogeneous data models in complex SWNs.

Sixth, existing semantic models exhibit limited capability to represent and reason over the dynamic, streaming nature of IoT data in SWNs. While static asset descriptions are well captured in many ontologies, real-time sensor data streams are characterised by temporal variation, context changes, and discontinuities. Hence, they lack robust semantic annotation and reasoning support. This constrains the development of intelligent, real-time analytics, anomaly detection, and automated control strategies.

Finally, interoperability frameworks generally do not span the full spectrum of interoperability layers. Although technical, syntactic, and semantic aspects are addressed to varying degrees, pragmatic and organisational interoperability, which govern operational context, decision-making semantics, and cross-stakeholder alignment, remain insufficiently supported. Likewise, the integration of semantic interoperability with MAS and DSS is limited, despite their growing importance in distributed control and knowledge-driven management in SWNs.

Taken together, these gaps demonstrate a pressing need for a comprehensive semantic interoperability framework that can (i) unify heterogeneous IoT and water-domain ontologies, (ii) automatically identify semantic relationships across diverse data sources, (iii) support both static and dynamic data representations, and (iv) enable cross-domain integration and intelligent decision-making within IoT-enabled SWNs. This thesis addresses these gaps by proposing novel models, methods, and a framework to advance interoperability across the whole IoT–water knowledge ecosystem.

2.13 Summary

This chapter provides readers with a deeper understanding of interoperability in IoT-enabled SWNs and offers a review of the literature on IoT and water domains. At first, this chapter introduced the importance of SWN in tackling the water scarcity and water quality crisis, as SWNs are being built to enable remote and continuous monitoring, diagnosis of problems, management of maintenance issues, and optimisation of the water distribution network throughout the entire life-cycle of water. SWNs are evolving from SCADA to the next level as the gradual deployment of IoT devices, such as smart sensors and actuators, introduces an overlay of IoT capabilities, offering new opportunities for intra- and cross-domain applications. Still, it has further fuelled the interoperability issue. In an IoT-enabled SWN, the application interoperability can not be achieved without achieving syntactic and semantic interoperability of the data shared across different communication layers. Thus, the interoperability of applications remains a pressing topic in both industry and academia, and various approaches and solutions have been published recently to address this issue.

This chapter also presented the definitions and architectures of SWNs found in industry and academia. It listed the significant challenges and solutions developed in the water domain. Since a SWN results from the integration of many technologies, such as IoT for smart sensors, MAS for autonomous activity, and semantic web technologies (OWL, RDF, SPARQL, etc.) for managing and sharing data, it describes the characteristics and purpose of these technologies. This part also presented the fundamental concepts of a SWN, such as interoperability, interoperability levels/layers, the DIKW hierarchy, the knowledge base, and ontology, as defined in the literature. The third part presented the literature review. Initially, it listed influential projects from the EC FP7 and Horizon 2020 programmes that address interoperability challenges in the IoT and water domains. Then, the relevant work on knowledge management and DSS in the water domain is presented. Following this, the historical development of standards and ontologies is depicted to support the development of semantic models and the sharing of data on observations and measurements in the IoT and water domains. Thereafter, interoperability and its approaches and solutions from IoT and SWN projects were discussed.

Finally, some identified challenges and gaps during the investigation are highlighted and summarised. Interoperability of applications in IoT-enabled SWN remains an issue, as current solutions do not cover all interoperability layers and do not provide appropriate semantic models for the interoperability, management, reasoning, and sharing of heterogeneous, static, and dynamic data/information.

Chapter 3

Data Information Interoperability Model and Methodology (DIIM)

The previous chapter reviewed the literature on the IoT and water domains, listing their challenges and research gaps that motivate the contributions presented in this thesis. This chapter explains the first significant contribution, the Data Information Interoperability Model and Methodology (DIIM), in detail, and demonstrates how it achieves Objective 3.

3.1 Overview

In this chapter, the first contributions of the research are presented as the Data Information Representation Model (DIRM) and the Data Information Interoperability Model and Methodology (DIIM). First, DIRM as a model is proposed to abstract and represent the concept of representing data and information at the different layers involved in digital transformation after humans and smart devices in the water domain collect data. Based on the DIRM, DIIM as a methodology prescribes how to systematically perform steps to harmonise data and information across the three representation layers. DIIM describes the syntactic and semantic harmonisation process for data and information, focusing solely on the syntactic and semantic interoperability levels, rather than all seven LCIM levels. The core of DIIM is built on the Model-Driven Architecture (MDA) and the SW approach. Thus, a DIIM-based system can utilise the existing SW tools and technologies to syntactically and semantically harmonise data. Related work shows that a semantic model (ontology) is crucial for enabling data interoperability. Therefore, data represented in semi-structured formats are transformed into graph-based representations. Finally, a generic use case for evaluating the DIIM with a proposed parameter setup, including datasets, reference dictionaries, ontologies, and tools, is presented.

3.2 Data Information Representation Model (DIRM) for Smart Water Networks (SWNs)

Figure 3.1 displays the proposed DIRM for SWNs. DIRM describes the bottom-up process of data and information collection and the digital representation of this information in the context of water, because data collection begins at water resources with field data collectors. DIRM refers to the collection and communication layer of the SWN architecture. Data and information collected according to DIRM are subsequently managed at the data management and display layer and finally used by the SWN applications in the data fusion and analysis layer of the SWAN architecture. According to Farias et al. (2016), the water data collection process encompasses both real-world physical objects (e.g. water resources, humans, and smart devices) and conceptual entities (e.g. documentation, reports, measurement values, images, sounds, etc.). The digital transformation involves objects that represent collected data and information, facilitating computer-aided processing for operators or analysts. According to DIRM, data and information become more structured as the digital transformation progresses from digital to ontological representation.

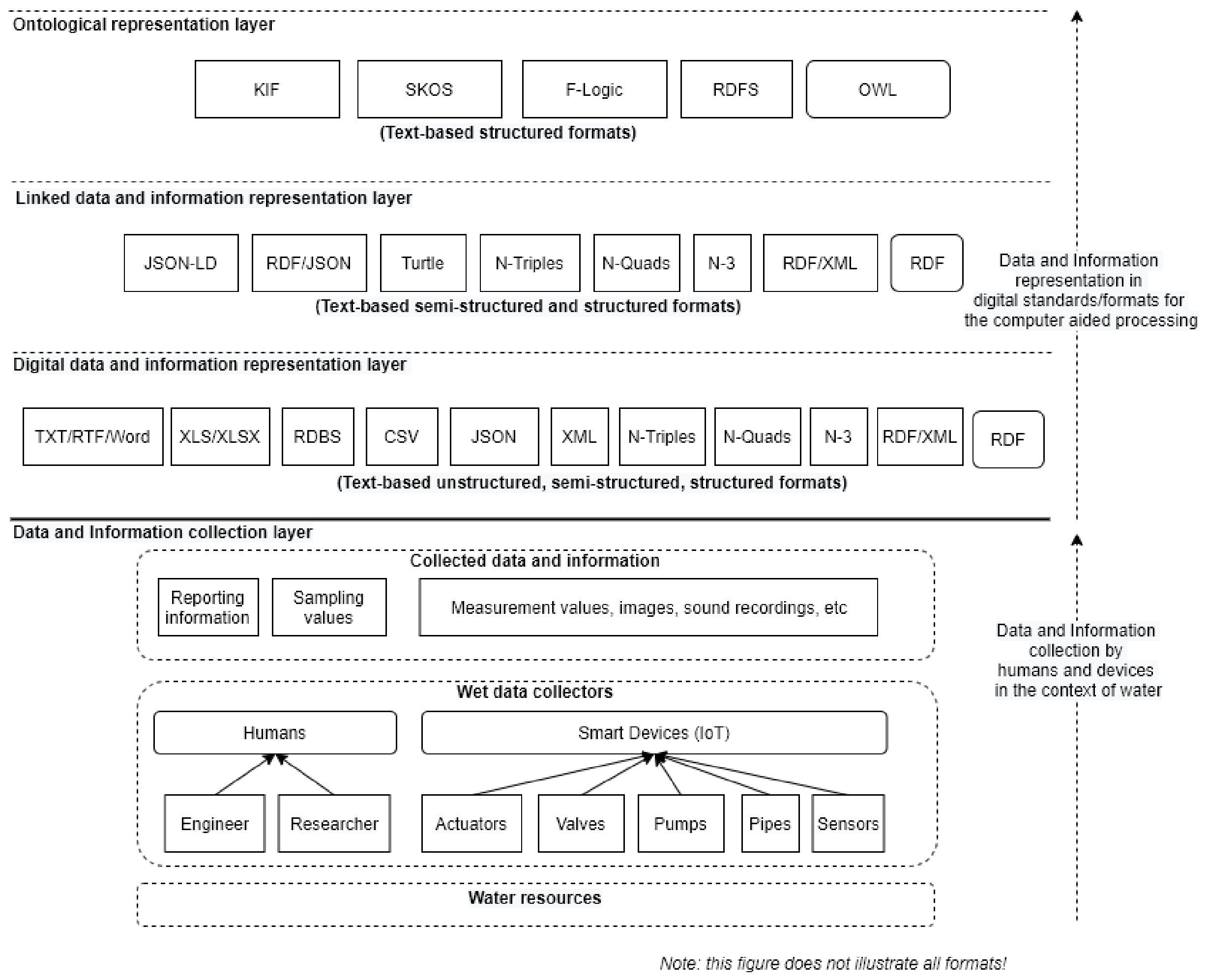

Ontological representation layer
KIF
SKOS
F-Logic
RDFS
OWL
(Text-based structured formats)
Linked data and information representation layer
JSON-LD
RDF/JSON
Turtle
N-Triples
N-Quads
N-3
RDF/XML
RDF
(Text-based semi-structured and structured formats)
Data and Information representation in digital standards/formats for the computer aided processing
Digital data and information representation layer
TXT/RTF/Word
XLS/XLSX
RDBS
CSV
JSON
XML
N-Triples
N-Quads
N-3
RDF/XML
RDF
(Text-based unstructured, semi-structured, structured formats)
Data and Information collection layer
Collected data and information
Reporting information
Sampling values
Measurement values, images, sound recordings, etc
Data and Information collection by humans and devices in the context of water
Wet data collectors
Humans
Smart Devices (IoT)
Engineer
Researcher
Actuators
Valves
Pumps
Pipes
Sensors
Water resources
Note: this figure does not illustrate all formats!

Figure 3.1: Data Information Representation Model (DIRM) for SWNs

The **Data and Information collection layer** comprises water resources, wet data collectors, and collected data and information. A water resource is any water that occurs on Earth, regardless of its state (i.e., vapour, liquid, or solid), and is potentially useful to humans or modern infrastructure. Of these, the most abundant resources are the waters of the oceans, rivers, and lakes; other available water resources include groundwater and deep subsurface waters, as well as glaciers and permanent snowfields (Britannica, 2020). Wet data collectors include humans (e.g. engineers for buildings and the environment, and hydrology researchers), as well as smart sensing and monitoring devices (e.g. sensors, actuators, valves, pumps, pipes, etc.) that record water-related data and information. The data and information collected by wet data collectors can take various forms, including reports, sampling and measurement values, images, and sounds, which are beneficial for humans, ICT systems, and computerised DSS for SWNs. For humans, water resources are essential for survival, economic activity, and environmental sustainability. In ICT systems, water resources are not physically consumed but are represented and processed as data and knowledge. Their usefulness arises from computational representation and analysis. In computerised DSS, water resources play a central role as the domain of decision-making. DSSs use water-related data and models to support informed, timely, and optimal decisions.

The **Digital data and information representation layer** comprises digital objects representing the data and information collected by the wet data collectors. These digital objects could be defined in various platform-specific formats required by the applications to process the data and information. An example is the Windows platform, where Microsoft Word or Excel serve as applications, and data or information is represented as text in Microsoft Word or as tabular values in Microsoft Excel. Additionally, the collected data and information are represented in various formats, including text-based, unstructured, semi-structured, and structured, enabling humans, ICT systems, and computerised DSS for SWNs to read and understand them.

The **Linked data and information representation layer** is where digitalised data and information from different sources are interlinked to be exchanged, reused, and integrated into a human and machine-readable format. Sir Tim Berners-Lee coined the term Linked Data (Berners-Lee, 2006) in the context of the Semantic Web to describe the creation of Linked Data from multiple datasets. Linked Data is preferably represented as a graph because its underlying data model, RDF, encodes information as *subject–predicate–object* triples, which naturally form a labelled directed graph. Graphs make relationships explicit, treat links as first-class elements, and allow flexible, schema-evolving integration of heterogeneous datasets. They also support computerised efficient traversal-based querying, data linking across sources, and semantic reasoning and inference. Consequently, a graph representation aligns both conceptually and technically with the goals of Linked Data. Some of the popular graph-based encoding formats to represent data and information are RDF, JavaScript Object Notation for Linking Data (JSON-LD) (Lanthaler et al., 2012), Turtle (Berners-Lee and W3C, 2014b), N-Triples (Beckett et al., 2014), N-Quads (W3C, 2014a), Notation3 (N3) (Berners-Lee and W3C, 2014a), and RDF/XML (W3C, 2013b).

The **Ontological data and information representation layer** represents structured data and information with additional rich semantics and domain knowledge, enabling convenient linking to other datasets. This layer provides an ontology as a semantic model to facilitate interoperability of data among SWN applications. Representing data and information as an ontology enables reasoning and inference because an ontology provides formal, machine-interpretable semantics that go beyond simple data storage (McGuinness et al., 2004). Because these definitions are expressed in a formal logic–based language (such as OWL, grounded in Deep Learning (DL)), a reasoning engine can automatically derive implicit knowledge from explicitly stated facts. For example, if an ontology states that *River is a subclass of SurfaceWaterBody*, and *SurfaceWaterBody is a subclass of WaterResource*. Then a reasoner can infer that every instance of River is also an instance of WaterResource, even if this is not explicitly asserted in the data. Facts in ontologies are instance-level assertions stored in the ABox (Assertion Box) and represented as

RDF triples; they can be explicitly stated or implicitly inferred. They describe concrete, real-world situations rather than abstract domain knowledge. Some of the widely used languages and formats to represent knowledge and to build ontologies are Knowledge Interchange Format (KIF) (Genesereth et al., 1992), SKOS, (Frame Logic (F-Logic) (Kifer et al., 1989), RDFS (W3C, 2014b), and OWL (W3C, 2013a).

3.3 Data Information Interoperability Model and Methodology (DIIM) for SWNs

Figure 3.2 illustrates how a DIRM-based model and methodology can achieve application-specific (based on the requirements of the given application) interoperability with a set of existing tools and technologies. As described in Section 3.2, data and information collection does not affect interoperability; instead, it involves digitising data and information in different formats across different layers. Thus, the Data and Information source layer of DIRM remains an independent process. DIIM can affect interoperability among the other three representation layers (digital, linked data, and ontological) if applications do not use a common standard representation format and a shared semantic model. At the digital and linked data representation layers, data and information can be represented in formats such as RDF, which can convert them to and from other formats. Similarly, OWL can serve as an interoperable language for the ontological layer.

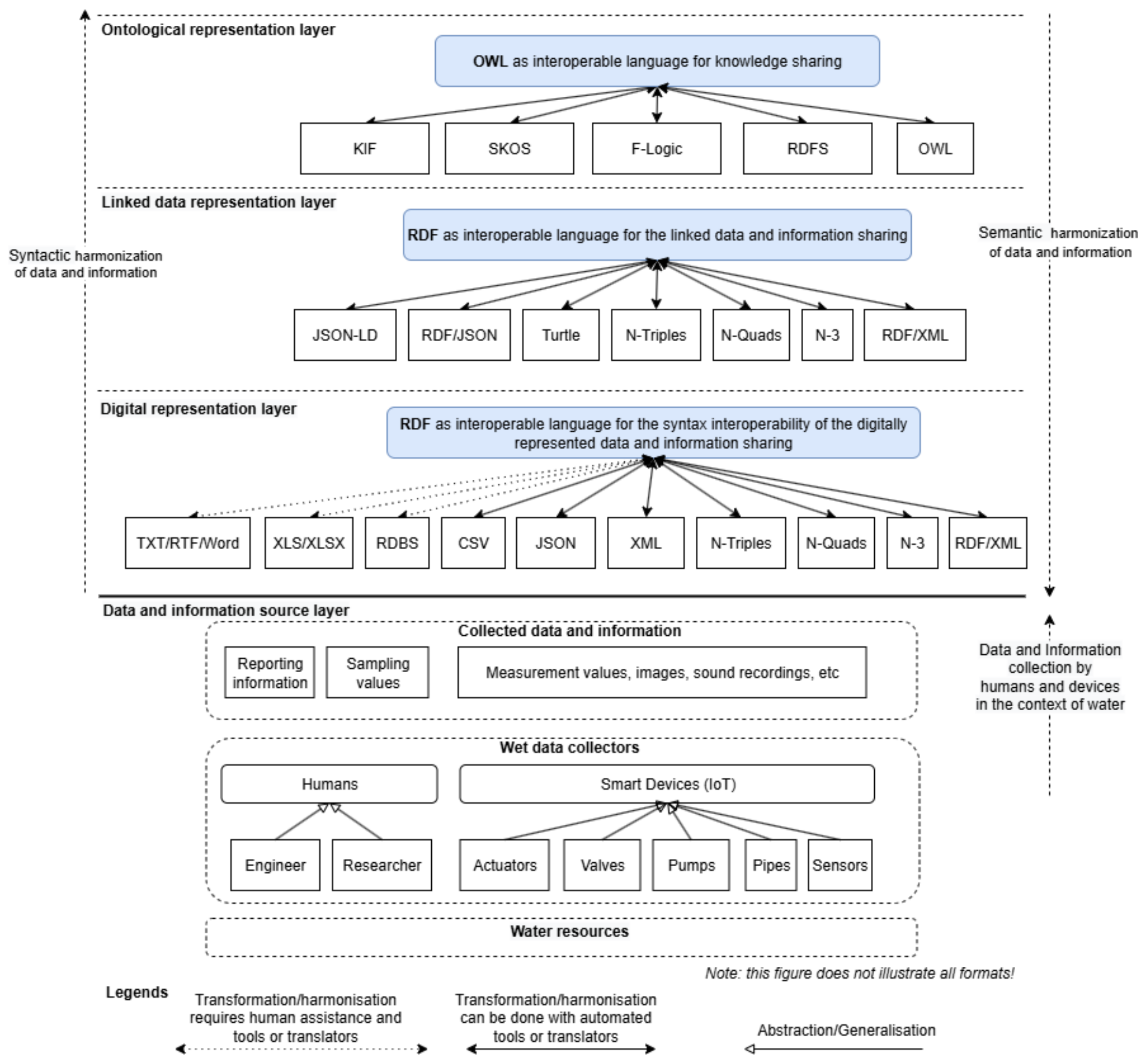

Ontological representation layer
OWL as interoperable language for knowledge sharing
KIF
SKOS
F-Logic
RDFS
OWL
Linked data representation layer
RDF as interoperable language for the linked data and information sharing
JSON-LD
RDF/JSON
Turtle
N-Triples
N-Quads
N-3
RDF/XML
Syntactic harmonization of data and information
Semantic harmonization of data and information
Digital representation layer
RDF as interoperable language for the syntax interoperability of the digitally represented data and information sharing
TXT/RTF/Word
XLS/XLSX
RDBS
CSV
JSON
XML
N-Triples
N-Quads
N-3
RDF/XML
Data and information source layer
Collected data and information
Reporting information
Sampling values
Measurement values, images, sound recordings, etc
Data and Information collection by humans and devices in the context of water
Wet data collectors
Humans
Smart Devices (IoT)
Engineer
Researcher
Actuators
Valves
Pumps
Pipes
Sensors
Water resources
Note: this figure does not illustrate all formats!
Legends
Transformation/harmonisation requires human assistance and tools or translators
Transformation/harmonisation can be done with automated tools or translators
Abstraction/Generalisation

Figure 3.2: Application-specific syntactic and semantic harmonisation of data with DIIM

As illustrated in Figure 3.2, DIRM-based DIIM proposes a methodological approach that utilises a set of existing tools and technologies to provide syntactic and semantic interoperability of data and information for water-domain applications. DIIM defines a systematic procedure for the syntactic translation/conversion of the data from one representation/format to another by applying MDA across the Digital, Linked Data, and Ontological layers. To achieve semantic interoperability, the semantic models (ontologies) are aligned (Euzenat and Shvaiko, 2007) and merged by utilising ontology alignment tools. As in LCIM (Turnitsa and Tolk, 2008), DIIM also requires syntactic harmonisation of data and ontology before the semantic harmonisation process begins.

DIIM does not require syntactic harmonisation when two applications want to share and use each other's data, provided that both applications use the same format to represent their data and the same language to build an ontology. Therefore, no data conversion or ontology translation is required. However, while following the Semantic Web (W3C, 2015) approach, DIIM proposes representing the data at the digital and linked data layers in RDF and building the semantic model (ontology) in OWL. Thus, RDF serves as the syntax and graph model for representing data and information, and OWL adds formal semantics and reasoning capabilities by representing domain knowledge in ontologies. However, according to the MDA-based approach of DIIM, any digital format/language can be used to represent the data and build an ontology, provided that there is a converter/translator for conversion to and from each format/language. Some of these languages are discussed in Sec 2.2.3. Figure 3.3 illustrates the four processing states that enable application-specific interoperability in DIIM. Application-specific interoperability enables applications to exchange data, regardless of differences in data representation formats and semantic models. If two applications, $App1$ and $App2$, want to understand and use each other's datasets, they must go through the four states illustrated in Figure 3.3. If the inputs to DIIM are $D1, O1, D2, O2$, then the output will be an aligned and merged ontology $AMOwF12$ with the definition and assumptions listed in Tables 3.1 and 3.2.

1. **Extraction of ontologies:** The first step is to ensure that ontologies exist for $D1$ and $D2$ and are not empty. If the conditions are not met, then the ontologies will be extracted.

Table 3.1: List of definitions in DIIM

$Def.1$	$Let\ App_1 := Application\ 1;$ // an IoT-enabled Application 1
$Def.2$	$Let\ App_2 := Application\ 2;$ // an IoT-enabled Application 2
$Def.3$	$Let\ D_1 := Dataset\ 1;\ \&\&\ D_1\ != null;$ // dataset of the Application 1
$Def.4$	$Let\ DF_1 := DatasetFormat\ 1;\ \&\&\ DF_1\ != null;$ // format of the Dataset 1
$Def.5$	$Let\ D_2 := Dataset\ 2;\ \&\&\ D_2\ != null;$ // dataset of Application 2
$Def.6$	$Let\ DF_2 := DatasetFormat\ 2;\ \&\&\ DF_2\ != null;$ // format of the Dataset 2
$Def.7$	$Let\ G_1 := Graph\ for\ G1;$ // Graph representation of Dataset 1
$Def.8$	$Let\ GF_1 := GraphFormat\ 1;\ \&\&\ GF_1\ != null;$ // format of the Graph 1
$Def.9$	$Let\ G_2 := Graph\ for\ G2;$ // Graph representation of Dataset 2
$Def.10$	$Let\ GF_2 := GraphFormat\ 2;\ \&\&\ GF_2\ != null;$ // format of the Graph 2
$Def.11$	$Let\ IDF_1 := InterfaceDataFormat\ 1;\ \&\&\ IDF_1\ != null;$ // format of data that is accepted by Application 1's interface
$Def.12$	$Let\ IDF_2 := InterfaceDataFormat\ 2;\ \&\&\ IDF_2\ != null;$ // format of data that is accepted by Application 2's interface
$Def.13$	$Let\ STD_1 := SetofTermsusedinDataset\ 1;\ \&\&\ STD_1\ != null;$ // set of terms used to label the measurement values in a dataset by Application 1
$Def.14$	$Let\ STD_2 := SetofTermsusedinDataset\ 2;\ \&\&\ STD_2\ != null;$ // set of terms used to label the measurement values in a dataset by Application 2
$Def.15$	$Let\ SSTD_1 := SelectedSetofTermsusedinDataset\ 1;\ \&\&\ SSTD_1 \subseteq STD_1;$ // selected (considered in a use case) set of terms used to label the measurement values in a dataset by Application 1
$Def.16$	$Let\ SSTD_2 := SelectedSetofTermsusedinDataset\ 2;\ \&\&\ SSTD_2 \subseteq STD_2;$ // selected (considered in a use case) set of terms used to label the measurement values in a dataset by Application 2

Table 3.2: List of assumptions in DIIM

$Ass.1$	$If\ App_1\ has\ DF_1;\ \&\&\ App_1\ has\ IDF_1;\ => DF_1 == IDF_1;$ // an application's interface accepts the same data format as the format of // the application's dataset
$Ass.2$	$If\ App_2\ has\ DF_2;\ \&\&\ App_2\ has\ IDF_2;\ => DF_2 == IDF_2;$ // an application's interface accepts the same data format as the format of // the application's dataset

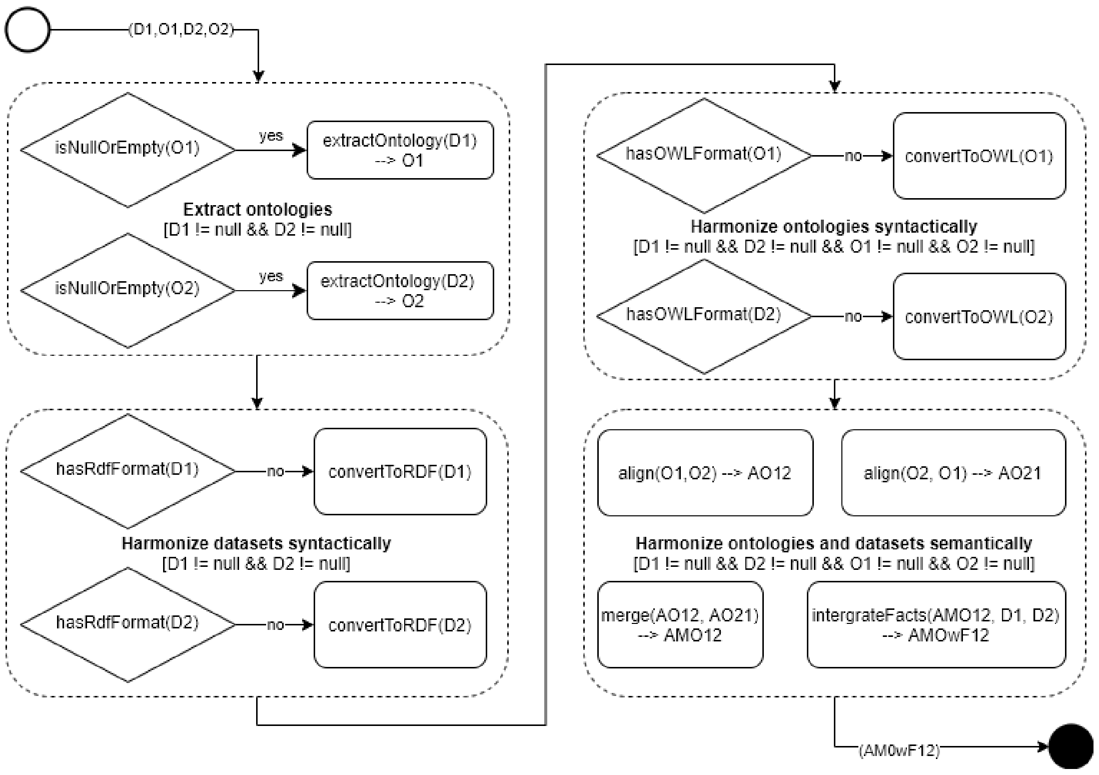

(D1,O1,D2,O2)
isNullOrEmpty(O1)
yes
extractOntology(D1) --> O1
Extract ontologies
[D1 != null && D2 != null]
isNullOrEmpty(O2)
yes
extractOntology(D2) --> O2
hasRdfFormat(D1)
no
convertToRDF(D1)
Harmonize datasets syntactically
[D1 != null && D2 != null]
hasRdfFormat(D2)
no
convertToRDF(D2)
hasOWLFormat(O1)
no
convertToOWL(O1)
Harmonize ontologies syntactically
[D1 != null && D2 != null && O1 != null && O2 != null]
hasOWLFormat(D2)
no
convertToOWL(O2)
align(O1,O2) --> AO12
align(O2, O1) --> AO21
Harmonize ontologies and datasets semantically
[D1 != null && D2 != null && O1 != null && O2 != null]
merge(AO12, AO21) --> AMO12
intergrateFacts(AMO12, D1, D2) --> AMOwF12
(AMOwF12)

Figure 3.3: Application-specific interoperability enablement in DIIM

2. **Harmonisation of datasets syntactically:** The next step is to harmonise the datasets syntactically into RDF; i.e. if they are not in RDF format, convert them to RDF.
3. **Harmonisation of ontologies syntactically:** At this step, ontologies $O1$ and $O2$ are converted to OWL if they are not defined in OWL.
4. **Harmonisation of datasets and ontologies semantically:** In the final stage, both ontologies $O1$ and $O2$ are aligned with each other. The aligned ontologies $AO12$ and $AO21$ have been merged to yield the aligned and merged ontology $AMO12$. The data from two datasets, $D1$ and $D2$, are integrated into $AMO12$. Thus, the final output is an aligned, merged ontology $AMOwF12$ that integrates data as facts from both datasets. The $AMOwF12$ ontology is then fully interoperable for both applications $App1$ and $App2$

3.3.1 Step 1: Transformation for syntactic harmonisation of data

In DIIM, syntactic harmonisation is a bottom-up process that starts at the digital representation layer, progresses to the linked data representation layer, and concludes at the ontological layer.

Table 3.3 describes abbreviations used in Tables 3.4, 3.5, 3.8, and 3.9.

Syntactic harmonisation at the digital representation layer: As shown in Table 3.4, if two different datasets D_1 and D_2 are represented in the same syntactic format, then they can be syntactically interoperable for the applications that can support the represented syntactic format. Suppose two datasets are not in the same format. For example, in Table 3.4, $D1_x$ and $D2_y$ represent two different formats. Therefore, they need a function f_{st} that can transform the format of one dataset into that of another so that the application can understand the datasets syntactically. Furthermore, the syntactic interoperability of the datasets can be achieved for formats that cannot be directly transformed into one another, but through transitive transformations.

Syntactic harmonisation at the linked data representation layer: At the linked data representation layer, two datasets are treated as a single dataset, but some of the data is not locally available; instead, it can be retrieved via URIs (Berners-Lee and W3C, 1994). Suppose applications want to interlink their datasets at this layer. In that case, they require either a common data representation format or a common transformation/conversion to an agreed-upon exchange format. As DIIM explained in Section 3.3, at the digital representation layer, RDF is the preferred language for representing and exchanging the linked datasets. Therefore, it is proposed to convert the dataset to RDF formats. No conversion operation is required if the datasets are already represented in RDF at the digital representation layer.

Syntactic harmonisation at the ontological representation layer: At this layer, the semantic model of applications is represented as ontologies. Applications use these semantic models to construct and represent their data; therefore, as shown in Table 3.5, while exchanging data, both applications must have a common language to represent their ontologies, or they must harmonise (translate/convert) their ontologies to a common agreed-upon language. To harmonise the ontologies, OWL is the preferred language for DIIM, as it is the core standard for the Semantic Web.

Table 3.3: Description of abbreviations used in Tables 3.4, 3.5, 3.8, and 3.9

Abbreviation	**Description**
App	Application
$App1$,$App2$	$App1$ and $App2$ are different applications
D	Dataset
$D1$, $D2$	$D1$ are $D2$ are different datasets
D_{1JSON}	Dataset1 in JSON format
D_{2JSON}	Dataset2 in JSON format
D_{1XML}	Dataset1 in XML format
D_{2XML}	Dataset2 in XML format
D_{1CSV}	Dataset1 in CSV format
D_{2CSV}	Dataset2 in CSV format
D_{1X}	Dataset1 in X format, where X format is unequal to Y format.
D_{2Y}	Dataset2 in Y format, where X format is unequal to Y format.
DF	Data format
$DF1$, $DF2$	$DF1$, $DF2$ are different data formats
f_{st}	Syntactic transformation function. After its application to two datasets or ontologies with different formats, they become syntactically interoperable.
O	Ontology
$O1$, $O2$	$O1$ and $O2$ are different ontologies
O_{1OWL}	Ontology1 in OWL format
O_{2OWL}	Ontology2 in OWL format
$O_{1F-Logic}$	Ontology1 in F-Logic format
$O_{2F-Logic}$	Ontology2 in F-Logic format
O_{1SKOS}	Ontology1 in SKOS format
O_{2SKOS}	Ontology2 in SKOS format
O_{1KIF}	Ontology1 in KIF format
O_{2KIF}	Ontology2 in KIF format
O_{1X}	Ontology1 in X format, where X format is unequal to Y format.
O_{2Y}	Ontology2 in Y format, where X format is unequal to Y format.
$SSTD$	Selected Set of Terms used in a Dataset
$SSTD1$, $SSTD2$	$SSTD1$ and $SSTD2$ are different Selected Set of Terms used in a Dataset (SSTD)
Syn_{intop}	Two datasets or ontologies with different formats are syntactically interoperable.

Table 3.4: Syntactic harmonisation matrix for the data and linked data representation layer

Dataset formats	D_{1RDF}	D_{1JSON}	D_{1XML}	D_{1CSV}	D_{1X}
D_{2RDF}	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$
D_{2JSON}	$f_{st} \rightarrow Syn_{intop}$	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$
D_{2XML}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$
D_{2CSV}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$
D_{2Y}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$

Table 3.5: Syntactic harmonisation matrix for the ontological representation layer

Ontology languages	O_{1OWL}	$O_{1F-Logic}$	O_{1SKOS}	O_{1KIF}	O_{1X}
O_{2OWL}	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$
$O_{2F-Logic}$	$f_{st} \rightarrow Syn_{intop}$	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$
O_{2SKOS}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$
O_{2KIF}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	Syn_{intop}	$f_{st} \rightarrow Syn_{intop}$
O_{2Y}	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$	$f_{st} \rightarrow Syn_{intop}$

3.3.2 Step 2: Alignment and Storage for semantic harmonisation of data

In DIIM, semantic harmonisation is a top-down process. At first, semantic models (ontologies) are harmonised through alignment at the ontological layer. Then, the aligned semantic model is applied to the linked data and digital representation layer through annotations. As stated by Ganzha et al. (2017), syntactic harmonisation across three layers is a prerequisite for initiating semantic harmonisation in DIIM.

Semantic harmonisation at ontological representation layer: At this layer, the similarity correspondences (references between ontologies with similarity scores that are similar to each other according to applied similarity algorithms) between two ontologies are identified through the ontology matching process, which can use various matching techniques, e.g. semantic, syntactic, terminological, structural, and extensional (Euzenat and Shvaiko, 2007). Once correspondences are found, ontologies are aligned using the OWL align construct in both ontologies. Ontology alignment is the process of identifying and establishing semantic correspondences between entities (e.g. classes, properties, individuals) in two or more ontologies. Every ontology has cross-aligned correspondences with entities in other ontologies. The next step is to merge the aligned ontologies and integrate the data on both sides, enabling local understanding and utilisation in both applications. Ontology merging is the process of combining two or more ontologies into a single, unified ontology that represents the knowledge of all source ontologies. Alignment informs and constrains merging, while merging materialises alignment into a consolidated knowledge model. Together, they enable semantic interoperability and integrated reasoning across heterogeneous knowledge sources.

Semantic harmonisation at the linked data representation layer: The semantic harmonisation at this layer can occur either through a reasoning engine (W3C, 2020) that utilises the aligned ontology to semantically understand the entities in linked data, or by directly using the merged ontology, which also incorporates linked data from different interoperable datasets. A reasoning layer is required on top of the linked data because, if linked data is not represented in any ontology format (e.g. RDF or JSON-LD), semantic correspondences cannot be presented or computed in these formats, as they do not support rich semantics and semantic annotations.

Semantic harmonisation at the digital representation layer: To semantically harmonise a dataset with another dataset at the digital representation layer, a reasoner and an aligned ontology are required. As with the linked data layer, a merged ontology could also be used to represent the integrated data in OWL and to add semantic annotations that refer to the aligned entities from the other dataset.

Groß et al. (2016) elaborate that an ontology enables the representation of data in a machine-processable form, ultimately allowing for reasoning, the generation of new knowledge, and the automatic detection of inconsistencies in semantic models. Therefore, DIIM also identifies an ontology's role in interoperability as crucial. Unfortunately, an application or domain ontology is not always available, and datasets are not always freely accessible online. The main reason for the absence of an ontology is that most data are represented at the digital layer of DIRM in non-ontological formats, as the collected data are intended for computation within its system and do not necessarily require an ontology. Additionally, building an ontology is a time-consuming task carried out by domain experts; as a result, it is often set aside with low priority.

3.3.3 Step 3: Validation for application-specific interoperability

Figure 3.3 illustrates the four processing states that enable application-specific interoperability in DIIM. However, it is also essential to first identify the interoperability requirements. If any exist, validate them after applying DIIM. Tables 3.6 and 3.7 provide information on cases where interoperability is needed between two applications. If syntactic or semantic interoperability is required, DIIM enables its interoperability by applying DIIM. Then, reviewing these matrices should list that no interoperability is required, as listed in Tables 3.8 and 3.9. For illustration,

Table 3.6: Syntactic interoperability validation requirement matrix

Application (App) Data format (DF)	**App1** $App1\_DF1$	**App2** $App2\_DF2$
App1 $App1\_DF1$	**No** $(App1\_DF1 = App1\_DF1)$	(i) **Yes**, if $App1\_DF1 \neq App2\_DF2$ (ii) **No**, if $App1\_DF1 = App2\_DF2$
App2 $App2\_DF$	(i) **Yes**, if $App2\_DF2 \neq App1\_DF1$ (ii) **No**, if $App2\_DF2 = App1\_DF1$	**No** $(App2\_DF2 = App2\_DF2)$

Table 3.7: Semantic interoperability validation requirement matrix

Application (App) SSTD	**App1** $App1\_SSTD1$	**App2** $App2\_SSTD2$
App1 $App1\_SSTD1$	**No** $(App1\_SSTD1 = App1\_SSTD1)$	(i) **Yes**, if $App1\_SSTD1 \neq App2\_SSTD2$ (ii) **No**, if $App1\_SSTD1 = App2\_SSTD2$
App2 $App2\_SSTD$	(i) **Yes**, if $App2\_SSTD2 \neq App1\_SSTD1$ (ii) **No**, if $App2\_SSTD2 = App1\_SSTD1$	**No** $(App2\_SSTD2 = App2\_SSTD2)$

in the option (i), we have two different Selected Set of Terms used in a Dataset (SSTD) inputs, $SSTD_1$ and $SSTD_2$. After the application of DIIM, for each term in $SSTD_1$, we can either find the same or a similar term in $SSTD_2$. Therefore, this validation requirement no longer exists and is therefore struck through in these requirements matrices.

3.4 DIIM's Application in IoT-enabled SWNs

Interoperability of Data and Information is key to IoT-enabled SWNs. DIIM follows the MDA approach (Mazón et al., 2008) and offers application-specific interoperability, aiming to achieve syntactic interoperability through the use of semantic web technologies, which provide both the social structure and the technical means to facilitate it (Ushold et al., 2005). In particular, RDF is used for the digital representation of data and interlinking, while OWL serves as a common language for knowledge representation. Figure 3.4 illustrates how different ontologies

Table 3.8: Syntactic interoperability validation requirement matrix after DIIM's application

Application (App) Data format (DF)	**App1** $App1\_DF1$	**App2** $App2\_DF2$
App1 $App1\_DF1$	**No** $(App1\_DF1 = App1\_DF1)$	(i) ~~**Yes**, if $App1\_DF1 \neq App2\_DF2$~~ (ii) **No**, if $App1\_DF1 = App2\_DF2$
App2 $App2\_DF$	(i) ~~**Yes**, if $App2\_DF2 \neq App1\_DF1$~~ (ii) **No**, if $App2\_DF2 = App1\_DF1$	**No** $(App2\_DF2 = App2\_DF2)$

Table 3.9: Semantic interoperability validation requirement matrix after DIIM's application

Application (App) SSTD	**App1** $App1\_SSTD1$	**App2** $App2\_SSTD2$
App1 $App1\_SSTD1$	**No** ($App1\_SSTD1 = App1\_SSTD1$)	*(i)* ~~**Yes**, if $App1\_SSTD1 \neq App2\_SSTD2$~~ *(ii)* **No**, if $App1\_SSTD1 = App2\_SSTD2$
App2 $App2\_SSTD2$	*(i)* ~~**Yes**, if $App2\_SSTD2 \neq App1\_SSTD1$~~ *(ii)* **No**, if $App2\_SSTD2 = App1\_SSTD1$	**No** ($App2\_SSTD2 = App2\_SSTD2$)

and standards interconnect to enable interoperability for data and information exchange in the water domain. The description breakdown of the figure is as follows:

- **Top Layer - Ontology Syntax:** Here, OWL serves as the interoperable syntax format for ontologies. This layer contains multiple ontologies represented by oval shapes, such as GOIoTP, WISDOM, SOSA, SSN, INSPIRE, Syndromic Surveillance Ontology (SSO), SWIM, WatERP, Semantic Web for Earth and Environment Technology (SWEET), SAREF, OntoPlant, Hydrologic Ontology for Discovery, and HydroOntology. These ontologies represent different conceptual frameworks or vocabularies for describing aspects of the water domain. The arrows between them indicate relationships or dependencies, e.g. WISDOM supports several ontologies, such as SSO, SSN, INSPIRE, and WatERP.
- **Middle Layer - Standards:** Rectangular boxes represent standards such as WaterML2, WDTF, IFC4, HY Features, Utility Network Schemas, CityGML Utility ADE, etc. Blue arrows point upwards from these standards to ontologies, showing that the standards are supported by ontologies (semantic layer). WaterML2 serves as a central standard, backed by ontologies such as SWEET and WatERP, and in turn supports several other standards, including WDTF, IFC4 XHydro, and HY Features.
- **Bottom Layer - Data Representation Syntax:** At the base of Figure 3.4, RDF is shown as the interoperable syntax format for digitally represented data/information. XML is shown below as the base syntax, as RDF is built upon XML.

These layers indicate the syntactic interoperability level, i.e. how data is structured and exchanged. Furthermore, it shows that data can be syntactically transformed from one standard to RDF using a standard-specific converter. Many RDF converters for semi-structured data formats are already implemented in the ConverterToRdf tools list (W3C, n.d.).

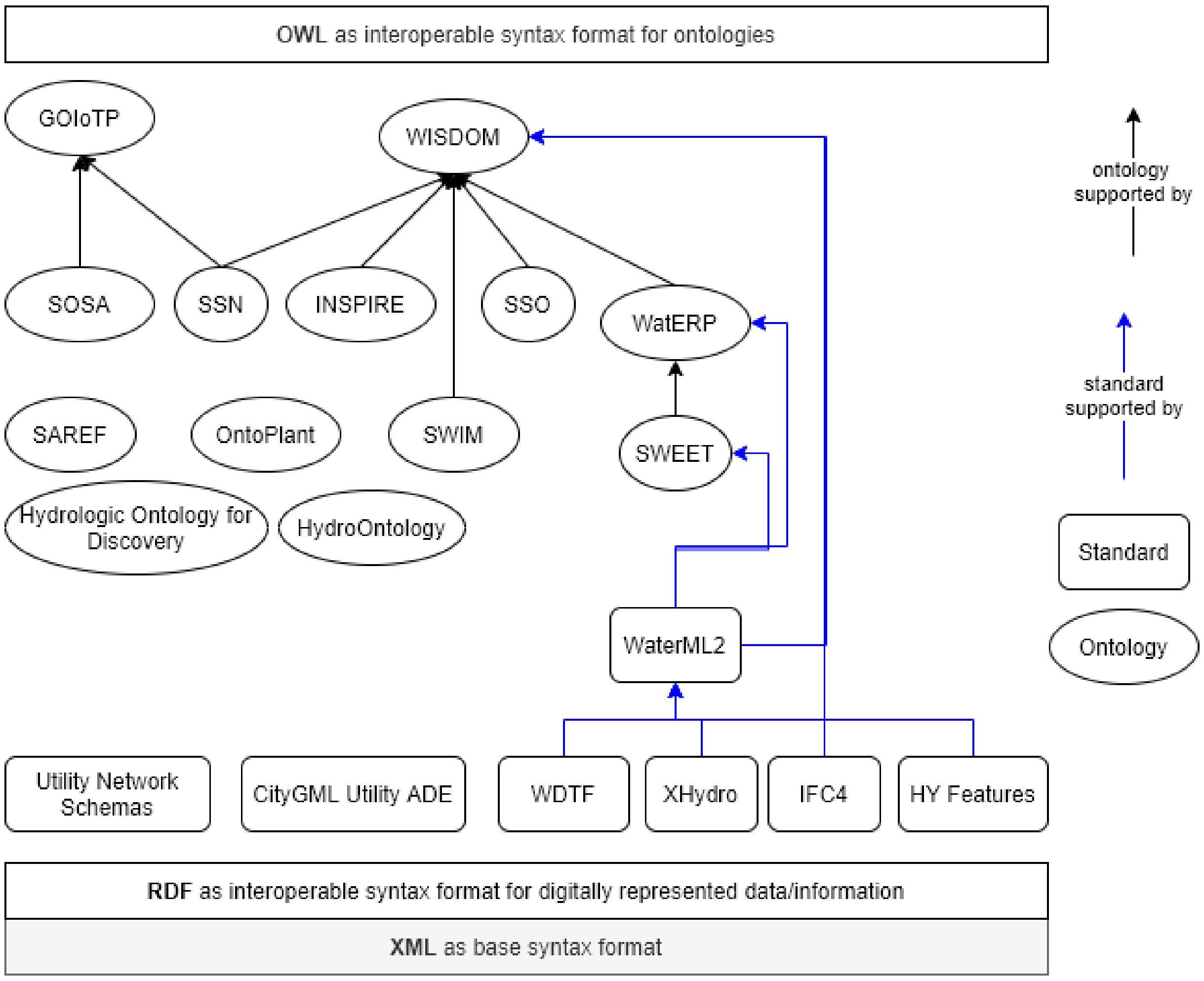


Figure 3.4: Syntactic and semantic interoperability through standards and ontologies in the water domain

In Chapter 2, the reviewed IoT projects (SEMIoTICS, BigIoT, OpenIoT, and INTER-IoT) utilise the transitive conversion model with mediator logic for data protocols to achieve interoperability solutions. For example, DeXMS: $MQTT \leftrightarrow CoAP$ and $CoAP \leftrightarrow REST$, then achieving interoperability of $MQTT \leftrightarrow REST$. Semantic interoperability relies on ontology.

A similar interoperability approach is adopted in water-related projects. Initially, a base ontology, e.g. the WISDOM ontology, is aligned with all relevant standards and ontologies and then used to convert between them. Therefore, an ontology is crucial to achieving semantic interoperability. Semantic web technologies are utilised to develop semantic models using ontologies across all reviewed water-domain projects on semantic interoperability. Figure 3.4 illustrates the well-known standards and ontologies of the water domain. All listed standards are based on markup language XML, and all listed ontologies are defined in OWL. As XML, RDF, and OWL are Semantic Web technologies, RDF is built on XML and OWL is built on RDF (Gómez-Pérez, Fernández-López, et al., 2004). Thus, the proposed languages RDF and OWL for representing data and information in DIRM are well-suited for both syntactic and semantic interoperability. In DIIM, OWL can be used as an interoperable language to overcome the syntactic and semantic heterogeneity of ontologies at the ontological layer, and RDF can be used to overcome the syntactic heterogeneity of data and information represented in any of the listed water domain standards and ontologies.

3.4.1 Semantic model creation methodology from semi-structured datasets

Though many tools and methods can (semi-)automatically extract ontologies from various sources, ontology learning remains a developing field in which each task of the ontology learning layer cake (terms, synonyms, concepts, concept hierarchy, relations, relation hierarchy, axioms, schemata, general axioms) is a vast research area that needs improvement. Each stage is dependent on the results of the previous stage. If one stage produces incorrect information, it will propagate to subsequent stages, ultimately resulting in low-quality ontologies (Asim et al., 2018). Figure 3.5 illustrates the proposed data-driven ontology extraction process for a semi-structured dataset. This figure visually explains the process of generating, aligning, and enriching an

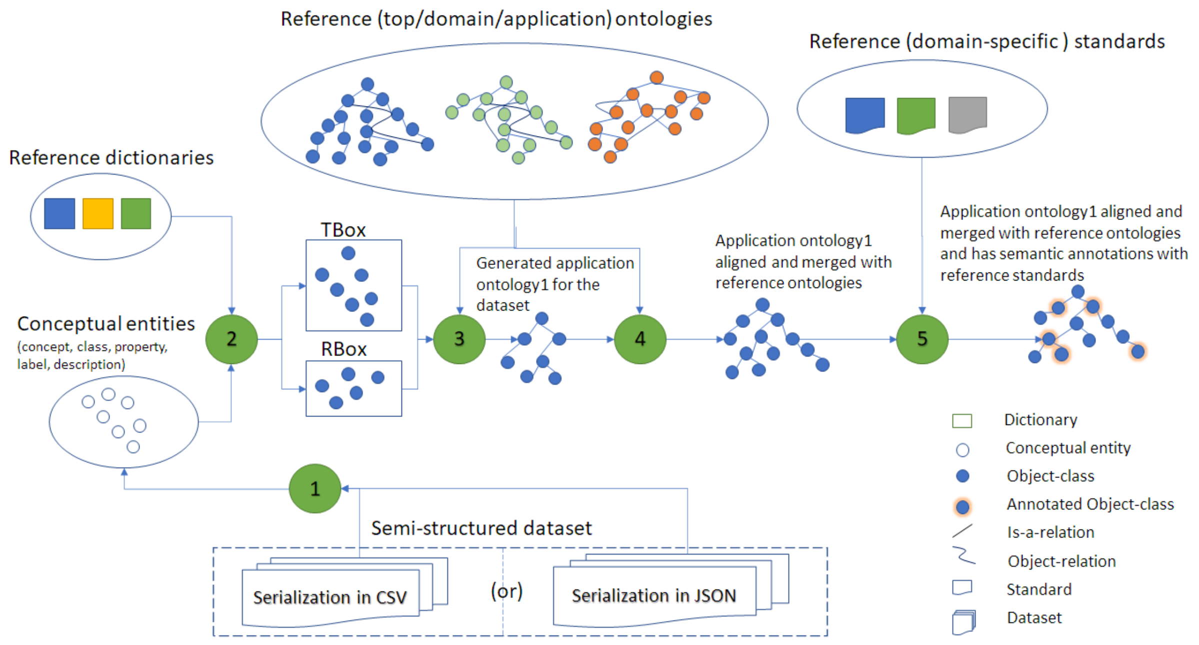

Reference (top/domain/application) ontologies
Reference (domain-specific) standards
Reference dictionaries
TBox
RBox
Conceptual entities
(concept, class, property, label, description)
Generated application ontology1 for the dataset
Application ontology1 aligned and merged with reference ontologies
Application ontology1 aligned and merged with reference ontologies and has semantic annotations with reference standards
1
2
3
4
5
Semi-structured dataset
Serialization in CSV
(or)
Serialization in JSON
Dictionary
Conceptual entity
Object-class
Annotated Object-class
Is-a-relation
Object-relation
Standard
Dataset

Figure 3.5: Ontology extraction from a dataset

ontology from a semi-structured dataset using reference ontologies and standards. The process is a five-step workflow (labelled 1–5 in green circles) that shows how a semi-structured dataset (such as CSV or JSON) evolves into a semantically enriched ontology that aligns with reference ontologies and standards.

This represents raw data that can be processed to generate an ontology.

Step 1 - Input Semi-Structured Dataset: At the bottom left, the process starts with a semi-structured dataset as an input. The first step involves preparing the semi-structured dataset (in this case, datasets are either in CSV or JSON) for extraction by filtering out conceptual entities from the data values. The conceptual entities are the terms used to model or express concepts, classes, properties, relations, labels, descriptions, and other related elements. In CSV, they are located in the dataset's header, and in JSON, they serve as keys. Creating an ontology from CSV headers involves extracting column names as domain concepts, classifying them into ontology entities, formalising them as classes and properties, and organising them into a minimal conceptual structure. Although limited in expressiveness, this approach provides a crucial first step toward semantic integration when only structural metadata is available.

Step 2 - Generation of TBox and RBox: The ontology learning process starts by generating TBox (terminological box, representing the schema or ontology structure) (Farias et al., 2016) and RBox (role box, representing relationships among entities) (Walter et al., 2014) from the given input of the dataset's conceptual entities. Reference dictionaries (represented by small coloured rectangles) help identify or map these entities. Tbox and Rbox are constructed by matching conceptual entities to words in language-specific reference dictionaries, such as WordNet (Princeton University, 2021). In Tbox, conceptual entities (terms) identified as nouns are grouped. In Rbox, relationships among terms are grouped by identifying verbs within conceptual entities. Relationships can be hierarchical (sub or supersumption), compositional (part of, contains, or has), or associational. Together, the TBox specifies what exists in the domain, while the RBox specifies how those concepts are related, enabling reasoning and inference in the ontology.

Step 3 - Creation of the application ontology for the dataset: Once the TBox and RBox are generated, pattern-based matching techniques of NLP are applied to identify concepts and their associated relations in the reference ontology repository. A reference repository contains existing ontologies, which are top-level/upper ontologies (define very general, domain-independent concepts that are common across all domains (Guarino, 1998)), domain-specific-level ontologies (model the concepts, relationships, and constraints of a particular application domain (Guarino, 1998)) and application-level ontologies (designed to model knowledge specific to a particular application, system, or use case, rather than an entire domain (Guarino, 1998)) ontologies from the IoT and water domains. After matching terms between the terms across the TBox, RBox, and reference ontologies, an ontology is generated in the DIIM-proposed language, OWL. This ontology represents the structure, classes, and relations discovered in the data.

Step 4 - Ontology alignment and merging: Figure 3.6 illustrates the alignment process for integrating the application ontology with domain-specific and upper ontologies. At this step, the goal is to align the newly generated ontology with reference ontologies to enhance interoperability. The more aligned terms the newly generated ontology has, the

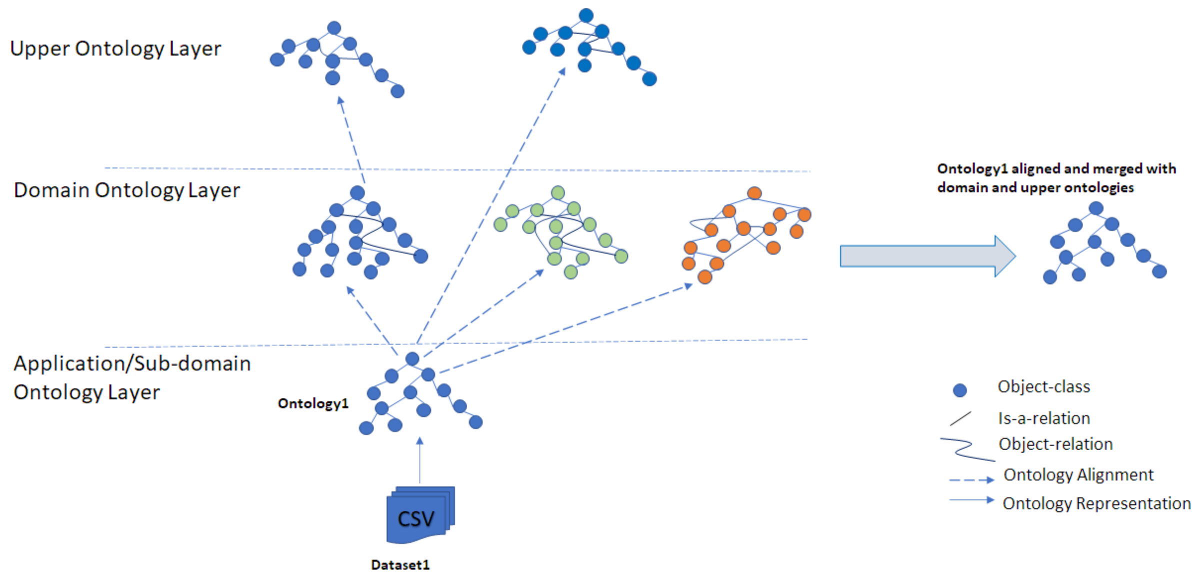


Figure 3.6: Ontology alignment with upper and domain ontologies

more likely it is to find matching terms during the semantic matching of two ontologies. Ontology alignment of the newly generated ontology begins with the top-level ontologies, then proceeds to align domain-specific ontologies, and finally aligns all application/task ontologies. All found alignments are recorded with OWL align constructs. After the alignment process, all ontologies are merged into the newly generated ontology with an n-ary merge algorithm, e.g. (Babalou et al., 2020). The final output of this step is an ontology that retains an alignment reference to other ontologies and integrates with them in the ontology repository. In short, the ontology generated in step 3 is aligned and merged with reference ontologies, which are categorised into three groups: top-level ontologies (general concepts), Domain ontologies (broad, domain-specific concepts), and Application ontologies (context-specific ontologies). This alignment adds semantic depth and ensures consistency with existing knowledge models.

Step 5 - Semantic annotation with standards: The aligned ontology is further merged with reference (domain-specific) standards. This task creates a semantically enriched ontology that complies with domain standards and has semantic annotations connecting it to those standards. The resulting ontology contains annotated object classes (indicated by blue circles with orange outlines).

In summary, Figure 3.5 illustrates a pipeline for transforming raw or semi-structured data into a semantically rich ontology, thereby ensuring both syntactic and semantic interoperability with existing knowledge bases and standards. Thus, DIIM will enable an application to make its data understandable to applications that support the reference ontologies and standards used in steps 4 and 5 for alignment.

3.4.2 Use case scenario

A possible use case for DIIM is to share and understand two datasets from data lakes located in different regions. Modern data-driven systems increasingly rely on multiple data lakes developed independently, evolving autonomously, and adopting heterogeneous conceptual models. These data lakes often employ distinct schemas and vocabularies, making cross-dataset interoperability and integrated analytics difficult. This use case addresses the problem of semantic heterogeneity across distributed data lakes by leveraging a layered ontology-based approach to align and merge datasets into a unified semantic representation. The proposed use case demonstrates how ontology alignment across applications, domains, and upper-ontology layers can be used to integrate heterogeneous data lakes semantically, preserving conceptual consistency and enabling unified access. In this use case, Figure 3.7 shows a DIIM's align-and-merge process to facilitate interoperability between lake data from two different sources.

Use case description: The use case begins with the ingestion of heterogeneous datasets into separate data lakes. Each dataset is semantically annotated using a corresponding local ontology that captures application-specific concepts and relationships.

Local ontologies are then aligned with relevant domain ontologies, which provide a shared semantic vocabulary for a particular knowledge area. These domain ontologies are further aligned through one or more upper ontologies, ensuring conceptual coherence at an abstract level. Ontology alignment techniques are applied to identify equivalence, subsumption (is-a), and object-property correspondences across ontologies.

Once alignments are established, the system performs ontology merging, consolidating aligned concepts into a single, integrated lake ontology. This merged ontology preserves

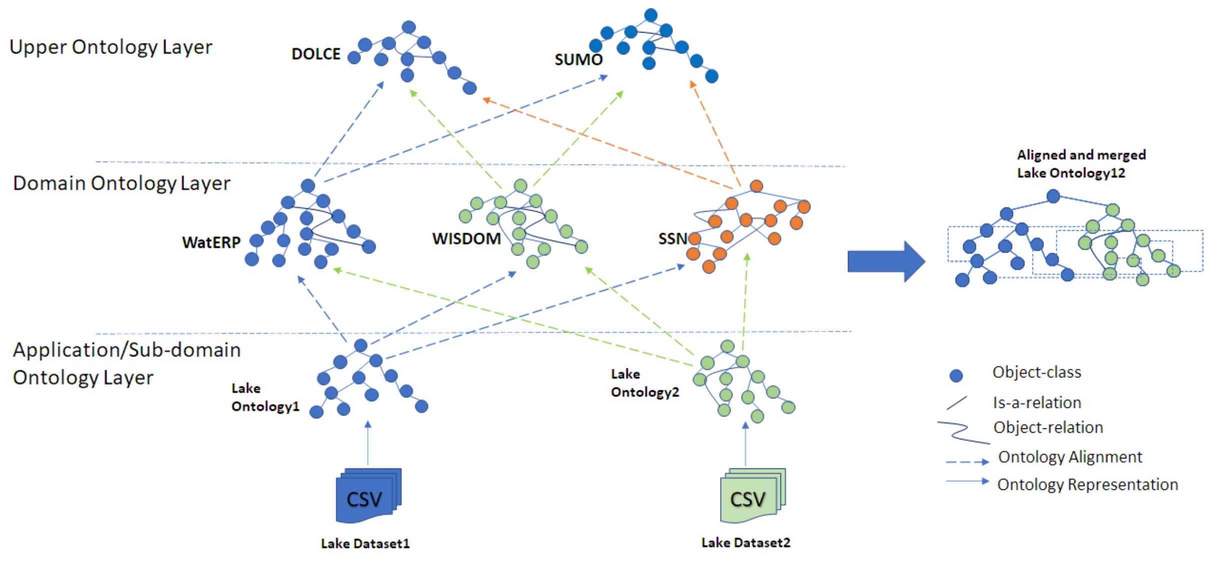


Figure 3.7: Alignment and merging of the two different lake datasets

semantic relationships and enables consistent interpretation of data across previously isolated data lakes. The resulting aligned and merged ontology serves as a unified semantic layer, supporting cross-lake querying, reasoning, and advanced analytics.

Actors:

1. Ontology Engineer: Designs and maintains local, domain, and upper ontology mappings.
2. Data Engineer: Ingests datasets and associates them with local ontologies.
3. Semantic Integration System: Performs ontology alignment, merging, and reasoning.
4. End User / Analytical Application: Consumes integrated data via unified queries and analytics

Preconditions:

1. Input datasets are available in structured or semi-structured formats.
2. Local ontologies exist or can be generated for each dataset.
3. Relevant domain and upper ontologies are accessible.
4. Ontology alignment mechanisms are available (manual, semi-automatic, or automatic).

Postconditions:

1. A unified, aligned lake ontology is produced.
2. Semantic relationships across datasets are explicitly represented.
3. Integrated datasets can be queried and reasoned over using a common semantic model.

4. Traceability between original datasets and merged ontology concepts is preserved.

Functional Requirements:

1. The system shall support ingestion of heterogeneous datasets into multiple data lakes.
2. The system shall allow modelling of datasets using local application or sub-domain ontologies.
3. The system shall integrate external domain ontologies and upper ontologies.
4. The system shall support ontology alignment, including class equivalence, subsumption, and object-property mappings.
5. The system shall merge aligned ontologies into a single coherent ontology.
6. The system shall maintain traceability between source ontologies and merged concepts.
7. The system shall enable unified querying across merged datasets.

Non-Functional Requirements:

1. Interoperability: The system shall support syntactic and semantic interoperability among data lakes and comply with Semantic Web standards (e.g., RDF, OWL, SPARQL).
2. Scalability: Ability to handle large ontologies and high-volume data lakes.
3. Extensibility: Support for incremental integration of new datasets and ontologies.
4. Consistency: Logical coherence of the merged ontology.
5. Maintainability: Support for the evolution of ontologies and alignments.
6. Explainability: Transparent representation of alignment and merging decisions.

Input: For the given use case, a possible set of input parameters, such as datasets, dictionaries, domain ontologies, upper/top-level ontologies, domain standards, and tools for learning, aligning, and merging ontologies, is listed in Table 3.10.

Output: The expected output of the DIIM will be a lake $ontology12$ derived from the given aligned and merged ontologies, lake $ontology1$ and lake $ontology2$. If $ontology1$ or $ontology2$ does not exist, they are generated from the input datasets.

Table 3.10: A set of input parameters for the evaluation of DIIM in the water domain

Input	**Source**
Datasets Datasets	Dataset1 (Gibson, 2016) in CSV format without ontology Dataset2 (Environment Agency GOV UK, 2016) in CSV format without ontology
Reference dictionaries	WordNet (Princeton University, 2021)
Reference domain ontologies	WISDOM SWIM SSN
Reference upper/top-level ontologies	DOLCE-UL (Guarino and Gangemi, 2017) SUMO (IEEE, 2004) COSMO (Cassidy, 2020)
Reference standards	WaterML2.0 (OGC, HDWG, 2014)
Ontology learning tools	Text2Onto (Cimiano et al., 2005) OntoLT (Buitelaar and Sintek, 2004) OntoBuilder (Gal et al., 2004) DODDLE-OWL (Morita et al., 2006)
Ontology alignment & merge tools	CoMerger (Babalou et al., 2020) PROMPT (Noy and Musen, 2000)

3.4.3 Comparison of DIIM with LCIM and OSI

Figure 3.8 describes DIIM from a model and methodology perspective. DIIM, as a model in terms of computer science, is a software system that takes three domain-specific inputs, IoT data, standards and ontologies, and generates an output of IoT KG with annotations that refer to its alignments to input standards and ontologies. DIIM, as a methodology in computer science, is the behaviour of a software system that implements it. DIIM's first step, *transformation*, is to convert input IoT data into a KG that is represented in RDF. RDF converters can convert it to any required data format, enabling syntactic interoperability at level 2 of LCIM (Wassermann et al., 2017). The second step, *alignment and storage*, aligns the IoT data with standards and ontologies and stores the alignment information as annotations in the IoT KG. When IoT data is required by the applications that support these standards or ontologies, semantic interoperability is enabled at LCIM's level 3 through annotations. In DIIM's third step, *validation*, the generated IoT KG with annotations is validated for syntactic and semantic interoperability through an

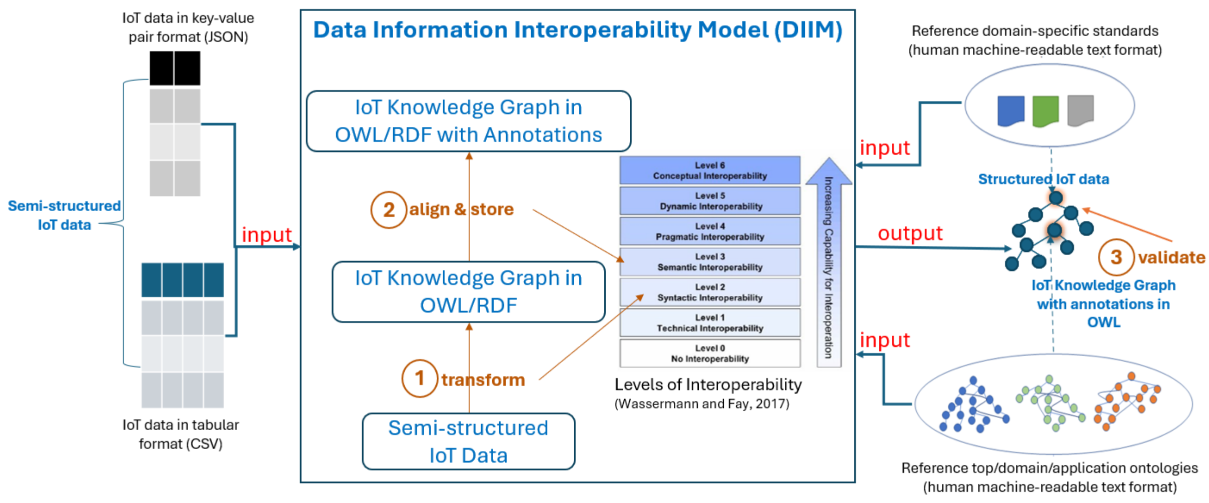

IoT data in key-value pair format (JSON)
Semi-structured IoT data
IoT data in tabular format (CSV)
input
Data Information Interoperability Model (DIIM)
IoT Knowledge Graph in OWL/RDF with Annotations
2 align & store
IoT Knowledge Graph in OWL/RDF
1 transform
Semi-structured IoT Data
Level 6 Conceptual Interoperability
Level 5 Dynamic Interoperability
Level 4 Pragmatic Interoperability
Level 3 Semantic Interoperability
Level 2 Syntactic Interoperability
Level 1 Technical Interoperability
Level 0 No Interoperability
Increasing Capability for Interoperation
Levels of Interoperability
(Wassermann and Fay, 2017)
input
output
input
Reference domain-specific standards (human machine-readable text format)
Structured IoT data
3 validate
IoT Knowledge Graph with annotations in OWL
Reference top/domain/application ontologies (human machine-readable text format)

Figure 3.8: An application scenario for DIIM's steps

application's requirement-based test.

DIIM is a conceptual model, just like LCIM and OSI. It focuses on the water domain while maintaining the DIRM as a fundamental approach to differentiate how data, information, and knowledge are collected and digitalised. Although this demonstrates the power of DIIM, enabling and advancing data and information interoperability for the applications in the water domain, it can be applied in any application domain where data interoperability is required.

Figure 3.9 visually compares three interoperability frameworks, LCIM, OSI, and DIIM, by aligning their corresponding layers to illustrate how interoperability concepts map across them. The figure is divided into three vertical sections: LCIM on the left (blue column), the OSI model in the middle (multicoloured stack), and DIIM on the right (blue gradient area). Horizontal dashed lines connect related layers across the three models, illustrating their conceptual relationships. In the OSI model, the presentation layer is responsible for presenting (i.e. encoding, encrypting, and compressing) data to the application layer. The presentation layer is also responsible for sending data from the application layer to another connected application through the session layer. According to Wasserman and Fay, LCIM's syntactic interoperability level maps to the presentation layer, and its semantic interoperability level maps to the application layer, in DIIM, syntactic interoperability extends to the application layer, as an application is responsible for converting data from one format to another if the

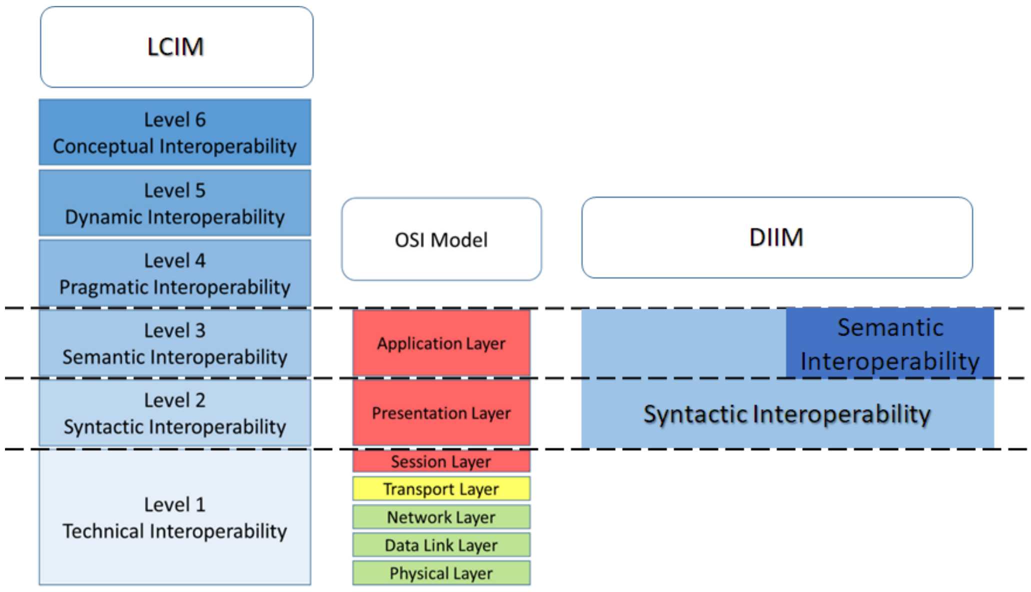


Figure 3.9: Comparison of DIIM with LCIM and OSI

format/standard of received/sent is not in the mutually agreed standard format. Semantic interoperability happens at the application layer in LCIM and DIIM because applications ensure semantic interoperability by applying a standardised meaning (semantic) to the sent/received data. In summary, DIIM bridges the gap between data exchange (syntax) and data understanding (semantics), aligning with parts of LCIM and OSI that address the meaning of communication rather than pure transmission.

3.5 Summary

In this chapter, two conceptual models, the Data Information Representation Model (DIRM) and the Data Information Interoperability Model and Methodology (DIIM) for Smart Water Networks (SWNs), are proposed. DIRM describes the bottom-up data and information collection process, as well as the digital representation of the collected data and information, in the context of water. It introduces four layers to represent data, information, and knowledge in the water domain: (i) data and information collection, (ii) digital data and information representation, (iii) linked data and information representation, and (iv) ontological representation. Based on the DIRM, DIIM describes how syntactic and semantic interoperability can be achieved in the top three layers of DIRM. DIIM comprises three steps: transformation, alignment and storage, and validation, enabling interoperability between two domain-specific applications. It also provides insights into the semantic modelling of the water domain and into how DIIM can be applied in IoT-enabled SWNs by adopting the MDA and SW approaches. In particular, OWL is proposed to address syntactic and semantic heterogeneity at the ontological layer, and OWL and RDF are proposed to address syntactic heterogeneity in data and information represented across various water domain standards and ontologies. This chapter also provides a methodology for creating a semantic model from semi-structured datasets that do not utilise an ontology or a standard. Finally, it also describes the application of DIIM in a potential use case in which two different lake datasets are aligned and merged to enable interoperability.

Chapter 4

Data Information Interoperability Framework (DIIF)

The previous chapter proposed the Data Information Representation Model (DIRM) and Data Information Interoperability Model and Methodology (DIIM) to enable and enhance the interoperability of IoT data for SWNs. This chapter presents the second significant contribution: a DIIM-based modular platform, the Data Information Interoperability Framework (DIIF), and details the essential components required to achieve Objective 4.

4.1 Overview

Conceptual models and methodologies alone are insufficient without a framework that enables practical implementation. In this context, the Data Information Interoperability Framework (DIIF) constitutes the second contribution of this thesis. The objective is to design and develop a modular framework and web platform that utilises Semantic Web (SW) and Natural Language Processing (NLP) technologies to achieve advanced interoperability in IoT-enabled SWNs. The chapter first discusses in detail the loosely coupled modular architecture of DIIF, followed by an explanation of DIIF's three-layered integrated data, information, and knowledge management concept. The subsequent section introduces DIIF's Service-Oriented Architecture (SOA), which enables the framework to operate as a web-based mediator platform. This architecture allows

SWNs and IoT applications to subscribe to and publish IoT data through DIIF's web services. Additionally, DIIF leverages the Semantic Web (SW) and Model-Driven Architecture (MDA) approaches to provide both syntactic and semantic interoperability of integrated IoT data and information for SWN applications. The following section introduces DIIF's ontologies, DIIO and S3O, which extend the framework's knowledge capabilities. DIIO serves as DIIF's knowledge base, containing information on existing standards, ontologies, and technologies in the IoT, water management, and Semantic Web domains, as well as details about registered IoT data. S3O augments DIIO by providing a semantic layer that identifies similarities among IoT data, standards, and ontologies. It supports multiple similarity calculation algorithms and their respective scores, thereby facilitating the alignment of IoT data with specific standards and ontologies. The next section provides a detailed description of DS3T, which is based on the DIIM concept. DS3T employs various Natural Language Processing (NLP) and Machine Learning Neural Network (ML NN) algorithms to compute semantic similarity between terms in datasets, standards, and ontologies. It processes IoT datasets, domain-specific standards, and ontologies as input and generates an S3O output containing algorithm-based semantic-similarity scores for terms. The subsequent section explains the implementation of DIIM's three steps within DIIF. The final section summarises the research undertaken to design and implement DIIF.

4.2 Data Information Interoperability Framework (DIIF) Architecture

The proposed DIIF Framework is built on Semantic Web technologies. It uses RDF as a basis for representing data at the Digital Data Information Representation Layer (DDIRL), the Linked Data Information Representation Layer (LDIRL), and the Ontological Representation Layer (ORL) of DIRM. From a technical perspective, Figure 4.1 illustrates the key components and their classification, which are designed as web services for the loosely coupled DIIM architecture. This means that each software component of DIIF exposes its functionality or data over a network, typically the Web, enabling access in a standardised, machine-interpretable way. Each web service should provide the technical interface description in WSDL (W3C, 2007) and a semantic, ontology-based description in OWL-S (W3C, 2004). Thus, DIIF enables different applications, possibly implemented in various technologies, to communicate and exchange data automatically. Figure 4.1 illustrates how each component is conceptualised as a modular entity that encapsulates functionality and communicates with other components through the SOA (Krafzig et al., 2005).

- **IoT components** are responsible for IoT data-related tasks. *Publication/Subscription Manager* provides a service for publishing or subscribing to IoT data. Each data subscription or publication creates a profile in the KB semantic model. *IoT Locator* scans the Thing Description Directory (TDD) (W3C, 2023) for IoT and returns a list of IoT that match the subscription. The IoT data collector collects data from the publisher's endpoint and delivers it to the KB. The IoT data pusher retrieves processed data from the KB and pushes it to the subscriber's endpoint (an End User/Analytical Application that has subscribed to IoT data).
- The **Data Analysis and Task management components** are responsible for IoT data analysis and generating tasks required to process the collected IoT data. *Data Analyser and Flagger* analyse the collected IoT data and create a publisher profile with a semantic model. Additionally, it flags every syntactic and semantic difference between the publisher's and the subscriber's semantic models. The *Task Generator* creates one or more tasks for each flag to achieve syntactic and semantic harmonisation of the collected data. The *Task Executor* executes generated tasks. The *Data Validator* validates the processed data and, upon failure, reschedules the *Data Analyser and Flagger*.
- **Syntactic components** harmonise the IoT data syntactically to an application standard. *Domain-specific Standard Adaptor* translates the collected data into a domain-specific standard. *Data Format Converter* serialises/deserialises the collected data from OWL into a platform or application-specific data format.
- **KB** incorporates knowledge of the DIIM methodology into the DIIM ontology, writ-

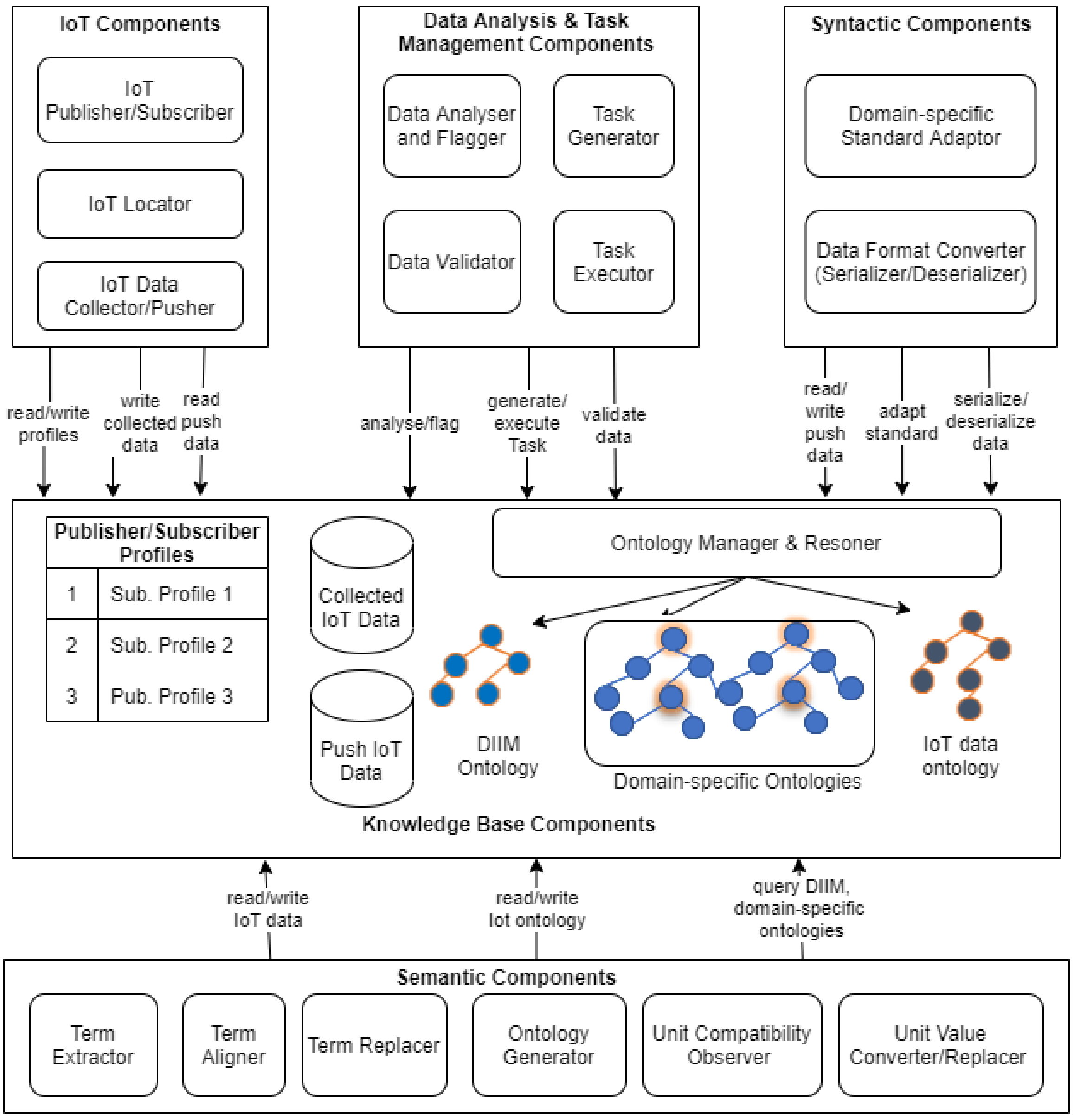


Figure 4.1: SOA-based architecture of DIIF components

ten in OWL. The core of the ontology contains domain-specific knowledge for a SWN application, including domain ontologies, standards, serialisation formats, measurement properties, and their units. After importing them, the *IoT Data Ontology* stores the collected IoT data in OWL/RDF. The *Ontology Manager* manages all ontologies in the KB. The *Publisher and Subscriber profiles* are also described and populated in KB using OWL. The *Term Aligner and Tagger* utilise existing alignment tools, e.g., ALIN, MapOnto, and Yam++, to align the IoT ontology (derived from IoT data) with other ontologies in the KB and to annotate its terms with the aligned terms. An *Ontology Reasoner*, such as Hermit or Pellet, reasons over the KB and answers queries. KB also includes a list of available OWL/RDF translators, serialisers, and deserialisers that can transform data between formats.

- **Semantic components** harmonise IoT data to domain-specific and application ontologies. *Term Extractor* extracts terms from the collected data. *Term Aligner* aligns the extracted terms with those of domain-specific ontologies and the subscriber's semantic model. *Ontology Generator* uses RDF conversion tools (W3C, n.d.) to generate an ontology of the collected IoT data and annotate its terms with aligned terms. The *Term Replacer* replaces the terms for the IoT data that is subscribed to. The *Unit Compatibility Observer* sets a flag if the IoT measurement unit differs from the subscriber's unit. The *Unit Valuer Converter and Replacer* replaces collected data values using the conversion unit formula if a unit conversion flag is present.

4.3 Integrated Knowledge Management

While following the DIRM and DIIM approaches, DIIF builds its integrated knowledge management concept in three layers, as shown in Figure 4.2. This figure, titled "Data, Information and Knowledge Management in DIIF", illustrates a layered architecture that represents how data, information, and knowledge are structured and managed within the DIIF. It displays three hierarchical layers: Access Layer, Source Layer, and Linked Layer, each serving distinct roles in processing and linking data to knowledge.

1. The **Access layer** forms the foundation of the architecture and grants applications access to data, information, and knowledge stored in DIIF. It represents the entry point for data collection and access within DIIF. This layer enables users or systems to access raw data and initiate its flow upward through the framework. While it's not subdivided, it conceptually connects external data sources to the internal framework.
2. The **Sourced layer** is the data foundation of DIIF. It integrates multiple data sources, each contributing distinct information. It retains the imported IoT data, standards, and ontologies in their original formats. Therefore, it contains three main components: (i) IoT Data represents data collected from IoT devices such as sensors, smart meters, and environmental monitors. (ii) Standards include standardised data models or data exchange formats that ensure interoperability. (iii) Ontologies provide structured vocabularies or conceptual models that describe relationships and semantics within the data. Together, these elements create a rich data ecosystem that combines raw sensor data, formalised standards, and semantic structures.
3. The **Linked layer** represents the stage of knowledge integration and enrichment. It manages the KGs (IoT data transformed into KGs, KGs containing the semantic similarity scores of terms used in IoT data, standards, and ontologies, and KGs containing alignment information) as an integrated knowledge base, utilising SW's linked-data concept. It includes three progressively enriched types of knowledge graphs, each building upon the previous one: (i) Transformed Knowledge Graphs, i.e. raw data from the source layer is transformed into structured graph-based representations, linking data entities and relationships. (ii) Knowledge Graphs with Semantic Similarity Scores. These graphs are enhanced with semantic similarity metrics, enabling the system to quantify the closeness of relationships among entities or concepts. (iii) Knowledge Graphs with Alignment Information. The final, most refined layer. It includes alignment information that links equivalent or related entities across datasets, ontologies, or domains, thereby enabling true semantic interoperability.

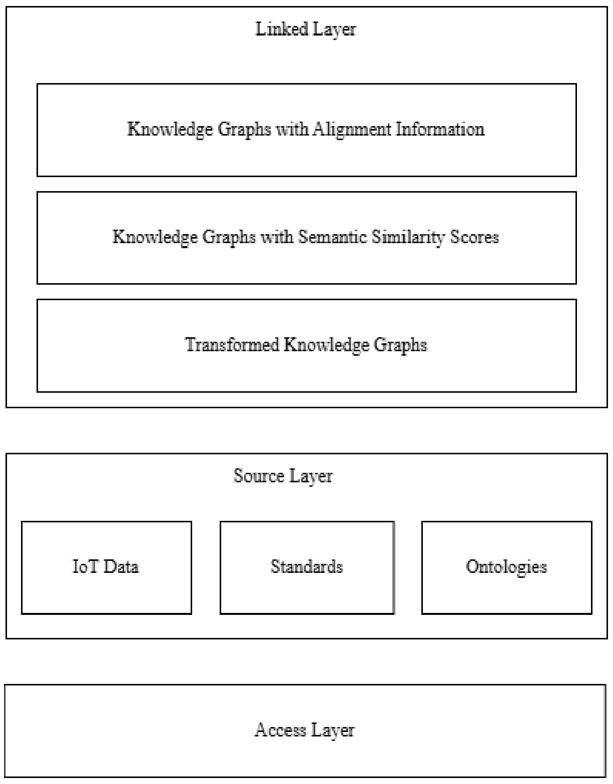

Linked Layer
Knowledge Graphs with Alignment Information
Knowledge Graphs with Semantic Similarity Scores
Transformed Knowledge Graphs
Source Layer
IoT Data
Standards
Ontologies
Access Layer

Figure 4.2: Data, Information, and Knowledge Management in DIIF

4.4 DIIF's Operational Activity as a Web Platform

Figure 4.3 illustrates DIIF's operational steps as a web platform that utilises existing tools and technologies to provide syntactic and semantic interoperability for data and information to IoT-enabled applications. The activity diagram illustrates the procedure for enabling syntactic and semantic interoperability between ad-hoc IoT (a collection of IoT devices that self-organise dynamically without relying on a fixed infrastructure or preconfigured network topology) and IoT-enabled applications. DIIF's procedural steps are as follows.

1. **Property subscription:** An IoT-enabled application subscribes to receive data of a particular property, e.g. temperature. The DIIF subscription interface specifies the parameters as follows:

   (i) **mandatory**{*subscriberEndpoint, communicationProtocol, serializationFormat, propertyName, measurementUnit*}

   (ii) **optional**{*latitude, longitude, fromDate, toDate, standardName, ontologyURI, applicationDomain*}

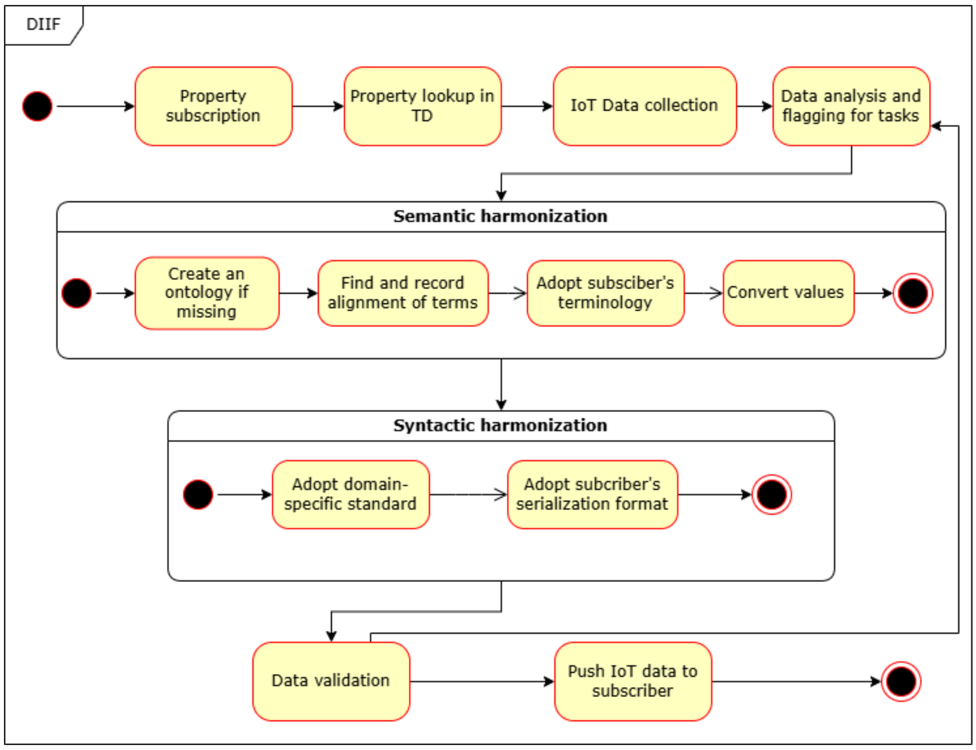

DIIF
Property subscription
Property lookup in TD
IoT Data collection
Data analysis and flagging for tasks
Semantic harmonization
Create an ontology if missing
Find and record alignment of terms
Adopt subsciber's terminology
Convert values
Syntactic harmonization
Adopt domain-specific standard
Adopt subcriber's serialization format
Data validation
Push IoT data to subscriber

Figure 4.3: DIIF's operational activity diagram as a web platform for SWN applications

2. **Property lookup in TDD:** DIIF keeps on scanning the TDD unless a match to the subscription is found.
3. **IoT data collection:** Once a subscription's match is found, DIIF starts collecting the available IoT data and metadata. IoT measurement data is stored in an ontology by modelling each measurement as a semantically rich observation instance that links sensors, observed properties, features of interest, values, units, and time. This step enables interoperability, reasoning, and intelligent decision support beyond raw data storage.
4. **Data analysis and task flagging:** In the next step, data is analysed and flagged based on the subscription. For each flag, a task is created, scheduled, and executed.
5. **Semantic harmonisation:** If IoT data do not refer to an ontology, an ontology is created for the terms used in the collected data. These terms are then aligned to the domain-specific and subscriber ontologies. All found alignments are stored as annotations in the IoT ontology. If the IoT measurement units do not match, the property values are converted and stored in the IoT ontology.
6. **Syntactic harmonisation:** The data in the IoT ontology is transformed into the domain-

specific standard and serialisation format required by the subscriber.

7. **Data validation and push:** Newly harmonised data is validated according to the subscription profile. If validation fails, DIIF switches to Step *Data analysis and flagging for tasks*; otherwise, data is pushed to the subscriber.

4.5 Ontologies in DIIF

All ontologies for the Data Information Interoperability Framework (DIIF) are created using the open-source ontology editor and framework Protégé (Noy, Crubézy, et al., 2003) for building intelligent systems. The ontologies were developed by following recognised ontology engineering methodologies and guidelines proposed by Noy and McGuinness (2001). The ontology development process followed the following key steps:

1. Requirements Specification: Define the scope and competency questions (what queries the ontology should support).
2. Conceptualisation: Identify core concepts, relationships, and constraints from domain analysis.
3. Knowledge Acquisition: Collect relevant domain knowledge from literature, datasets, and existing standards.
4. Integration and Reuse: Incorporate or align existing ontologies to avoid redundancy.
5. Implementation: Encode the ontology using OWL (Web Ontology Language) for machine readability.
6. Evaluation and Refinement: Validate using reasoning tools (e.g. Protégé and HermiT) and expert feedback.

4.5.1 Role of Ontologies in This Research

In this research, ontologies serve as structured, hierarchical classifications that define and organise entities, their attributes, and their relationships within IoT-enabled SWNs. They underpin the semantic modelling approach by enabling:

- consistent representation of heterogeneous IoT and water-domain data
- cross-domain interoperability
- automated reasoning and ontology matching
- identification of synonymous or related concepts across standards
- building blocks for a unified ontology (combining water-domain and IoT-domain concepts)

Table 4.1 provides a clear explanation of terms used in this research for ontology development.

4.5.2 Overview of Ontologies in DIIF

This section introduces the following ontologies developed in the Data Information Interoperability Framework (DIIF):

1. **Data Information Interoperability Ontology (DIIO)** serves as the knowledge base of the DIIF. DIIO should contain domain knowledge regarding existing standards, ontologies, and technologies in the IoT, water, and semantic web domains. All existing standards and ontologies discussed in Chapter 2 have been modelled in the DIIO. Furthermore, it should include information on which standards and ontologies are mutually supportive. It should also record information about the IoT data upon registration in the framework.
2. **Semantic Similarity Scoring Ontology (S3O)** extends DIIO. It provides a semantic layer for storing information about the similarity among IoT data, standards, and ontologies, as well as among the terms used in them. S3O should accommodate all possible similarity-computation algorithms and their corresponding scores, thereby enabling alignment of IoT data with domain-specific standards and ontologies. S3O should be able to answer the queries, e.g. which ontologies or standards are similar to each other, which terms exist in which ontologies or standards, or which terms are semantically similar or identical to which terms and which NLP algorithms were used to calculate the similarity score between documents (ontologies and standards described in digital documents) and terms.

The ontologies within the DIIF were developed to provide a formal, semantic foundation for interoperability among heterogeneous datasets, standards, and IoT-driven data streams from the water and IoT domains. SWN applications can use these ontologies to enable and enhance the

Table 4.1: Explanation of terms used in the ontology development

Term	**General Meaning in Semantic Modelling**	**Specific Meaning in This Research with an Example in Context**
Class	A high-level category representing a type of entity (e.g. Sensor, Pipe, Observation).	Represents the core building blocks of SWN knowledge—physical assets, IoT devices, data structures, events, actions, etc. Classes define the domain's essential concepts, which are used for interoperability. Examples: Sensor, Valve, FlowMeasurement, PressureObservation, Actuator.
Subclass	A more specific type of a class, inheriting its properties.	Allows modelling of specialised water or IoT entities without redefining everything. Helps align concepts across standards by finding fine-grained similarities. Examples: PressureSensor subclass of Sensor; PRVActuator subclass of Actuator.
Concept	A general term for any identifiable unit of meaning in an ontology.	A concept may refer to any IoT or water-domain element that needs semantic alignment. Used as the atomic unit in ontology matching and automatic synonym detection. Examples: LeakEvent, WaterQualityParameter, HydraulicZone.
Property / Attribute	A characteristic that describes a class or concept (e.g. hasValue and hasUnit). Used to unify how different systems represent the same data fields.	Properties help detect structural heterogeneity between standards and support semantic interoperability. Examples: hasPressureValue, occurredAt, observedBy, usesProtocol.
Object Property	A relationship between two entities.	Models relationships within the SWN (e.g. sensor observes a phenomenon and actuator controls a device). Crucial for establishing semantic equivalence between ontologies. Examples: observes(Sensor → Property), controls(Actuator → Pipe).
Data Property	A link between an entity and a literal value.	Standardises numeric and textual values from heterogeneous IoT data streams to enable unified reasoning. Examples: hasValue = 3.4, hasUnit = 'bar', timestamp = '2024-05-17T...'.
Instance / Individual	A concrete example of a class. Represents actual sensors, devices, measurements, and events coming from real SWNs.	Forms the dynamic, real-time knowledge graph used for reasoning. Examples: Sensor_1234, FlowMeasurement_2023-04-12.
Domain / Sub-domain	A thematic grouping of related concepts.	Used to merge IoT and water domains: e.g. combining GOIoTP's IoT concepts with WISDOM's water concepts. Examples: IoT domain (protocols, device types) + Water domain (hydraulics, assets, quality).
Alignment	Mapping concepts across different ontologies.	Core mechanism for achieving interoperability: identifies equivalences, synonyms, partial matches, and shared structure across water and IoT standards. Examples: Mapping PressureSensor (WISDOM) ↔ Sensor:Pressure (SAREF).
Annotations	annotation properties are used to attach human-readable or metadata information to ontology elements.	Their purpose is documentation, labelling, provenance, and alignment with external vocabularies. For example, *rdfs:label "Sensor"@en*.

interoperability of IoT data.

These ontologies were implemented in OWL 2 using Protégé and serialised in RDF for interoperability with external systems. KGs generated from the ontology serve as the backbone for the Linked Layer in DIIF, enabling semantic enrichment, alignment, and reasoning across diverse datasets.

These ontologies were validated through a combination of automated reasoning and expert review. Consistency checks ensured logical coherence, while competency questions verified that the ontology met all information retrieval and interoperability requirements. These ontologies are included in the appendix and can also be accessed at the DIIF's GitLab Uniform Resource Locator (URL) (M. Singh, 2025).

Ultimately, the ontologies facilitate both syntactic and semantic interoperability within DIIF, transforming raw IoT data into semantically enriched knowledge graphs that support intelligent querying, alignment, and decision-making across interconnected systems. These ontologies contain knowledge derived from the literature review conducted in Chapter 2 and the current implementation state of DS3T in DIIF. Therefore, these ontologies require management, given the state-of-the-art progress in the IoT and water domains.

4.5.3 Data Information Interoperability Ontology (DIIO)

This section introduces the Data Information Interoperability Ontology (DIIO) as a knowledge base for the DIIF, supporting the interoperability of IoT data and information. To demonstrate the practical value of DIIO, it was initially developed based on the literature review of technologies, standards, and ontologies. In particular, all information from Table 2.6 and Figure 3.4 on the ontologies and standards of the IoT and water domain is realised in DIIO to examine and support the syntactic and semantic interoperability of IoT data. Figure 4.4 illustrates how DIIO is currently built with initial knowledge from the SW, IoT, and water domains.

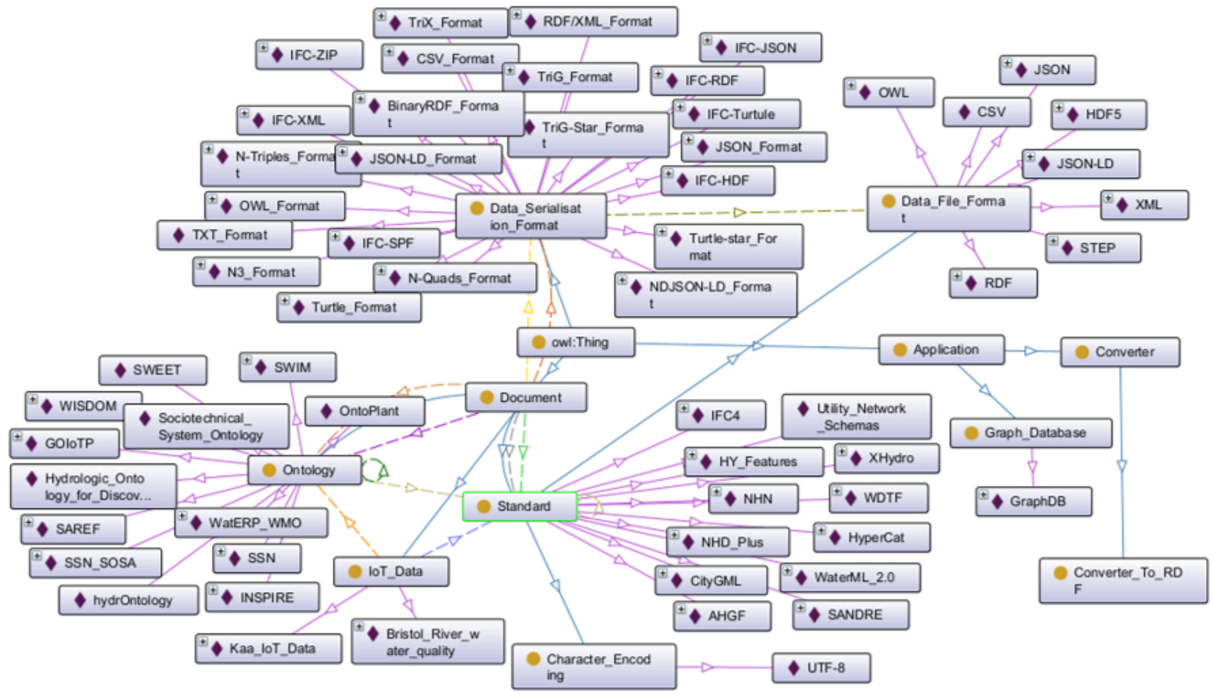


Figure 4.4: DIIO illustration with classes, instances, and relationships among them

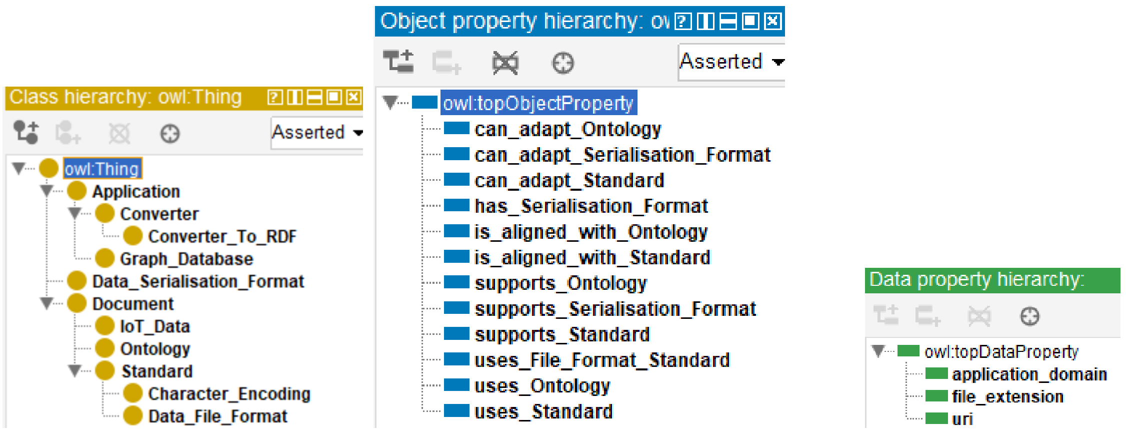
DIIO Conceptualisation

Figure 4.5 displays DIIO's conceptualisation by listing its classes, object properties and data properties in a tree view. The Class Entities view shows that all modelled classes have the root class $owl : Thing$, a general class without any instances. To understand each entity, DIIO describes it with the $rdf : comment$ annotation. DIIO also defines the domain and range of each data and object property. DIIO is included in the appendix and can also be accessed via DIIF's Gitlab URL (M. Singh, 2025).

DIIO Logical Rules

DIIO uses SWRL (Horrocks et al., 2004) rules to generate new knowledge through an inference engine such as Pellet (Sirin et al., 2007). The current version of DIIO contains SWRL rules on the following topics:

1. This rule states that if there are two ontologies $o1$ and $o2$ and $o1$ supports $o2$, then $o1$ can be converted into $o2$. As $can\_adapt\_Ontology$ is a symmetric and transitive relationship, $o2$ can also be converted into $o1$ and other ontologies supported by $o1$.


Class hierarchy: owl:Thing

Asserted

- owl:Thing
  - Application
    - Converter
      - Converter_To_RDF
    - Graph_Database
  - Data_Serialisation_Format
  - Document
    - IoT_Data
    - Ontology
    - Standard
      - Character_Encoding
      - Data_File_Format

Object property hierarchy: ow

Asserted

- owl:topObjectProperty
  - can_adapt_Ontology
  - can_adapt_Serialisation_Format
  - can_adapt_Standard
  - has_Serialisation_Format
  - is_aligned_with_Ontology
  - is_aligned_with_Standard
  - supports_Ontology
  - supports_Serialisation_Format
  - supports_Standard
  - uses_File_Format_Standard
  - uses_Ontology
  - uses_Standard

Data property hierarchy:

- owl:topDataProperty
  - application_domain
  - file_extension
  - uri

(a) Class entities (b) Object properties (c) Data properties

Figure 4.5: Conceptualisation of DIIO

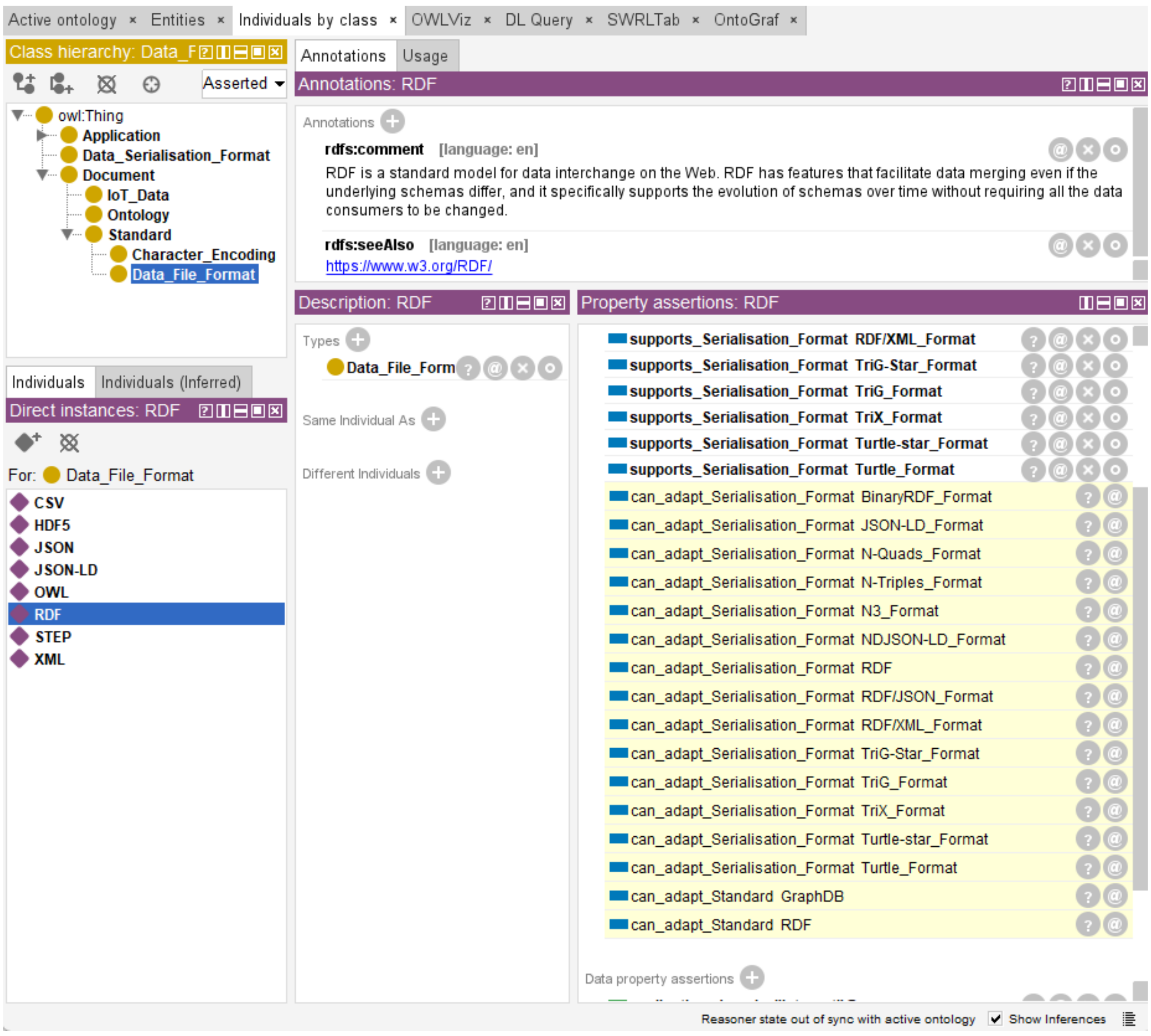
$$
\begin{aligned}
&diio:Ontology(?o1) \land diio:Ontology(?o2)\\
&\land\, diio:supports\_Ontology(?o1,?o2)\\
&\rightarrow\ diio:can\_adapt\_Ontology(?o1,?o2)
\end{aligned}
$$

In simple terms, whenever one ontology supports another, the supporting ontology can adapt the supported one. For example, if the WISDOM ontology supports the SSN ontology, then data and information expressed in WISDOM can be converted to SSN and vice versa. The rule does not perform the adaptation itself; it only infers this capability as new knowledge in the ontology.

2. This rule states that if there are ontology o and standard s, and o supports s, then o can be converted into s. As *can_adapt_Standard* is a symmetric relationship, s can also be converted into o and other standards and ontologies supported by $o1$.

$$
\begin{aligned}
&diio:Ontology(?o) \land diio:Standard(?s)\\
&\land\, diio:supports\_Standard(?o,?s) \rightarrow diio:can\_adapt\_Standard(?o,?s)
\end{aligned}
$$

In simple terms, if an ontology supports a standard, then the ontology can adapt that standard. For example, if the WISDOM ontology supports the WaterML2 standard, then data and information expressed in WISDOM can be adapted to it, and vice versa. As with the previous rule, this rule does not carry out any adaptation itself; it only infers a new



owl:Thing
- Application
- Data_Serialisation_Format
- Document
  - IoT_Data
  - Ontology
  - Standard
    - Character_Encoding
    - Data_File_Format

For: Data_File_Format

CSV
HDF5
JSON
JSON-LD
OWL
RDF
STEP
XML

Annotations

rdfs:comment [language: en]
RDF is a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed.

rdfs:seeAlso [language: en]
https://www.w3.org/RDF/

Types: Data_File_Form

supports_Serialisation_Format RDF/XML_Format
supports_Serialisation_Format TriG-Star_Format
supports_Serialisation_Format TriG_Format
supports_Serialisation_Format TriX_Format
supports_Serialisation_Format Turtle-star_Format
supports_Serialisation_Format Turtle_Format
can_adapt_Serialisation_Format BinaryRDF_Format
can_adapt_Serialisation_Format JSON-LD_Format
can_adapt_Serialisation_Format N-Quads_Format
can_adapt_Serialisation_Format N-Triples_Format
can_adapt_Serialisation_Format N3_Format
can_adapt_Serialisation_Format NDJSON-LD_Format
can_adapt_Serialisation_Format RDF
can_adapt_Serialisation_Format RDF/JSON_Format
can_adapt_Serialisation_Format RDF/XML_Format
can_adapt_Serialisation_Format TriG-Star_Format
can_adapt_Serialisation_Format TriG_Format
can_adapt_Serialisation_Format TriX_Format
can_adapt_Serialisation_Format Turtle-star_Format
can_adapt_Serialisation_Format Turtle_Format
can_adapt_Standard GraphDB
can_adapt_Standard RDF

Reasoner state out of sync with active ontology

Figure 4.6: List of formats that the RDF file format can adapt through inference rules

semantic relationship that represents an adaptation capability.

3. This rule states that if a data file format dff supports the serialisation format ssf, then dff can adopt the serialisation format ssf. Figure 4.6 shows how many serialisation formats the RDF file format standard can adopt through this rule.

$$diio : Data\_File\_Format(?dff) \wedge diio :$$
$$supports\_Serialisation_{F}ormat(?dff, ?ssf)$$
$$\rightarrow\ diio : can\_adapt\_Serialisation\_Format(?dff, ?ssf)$$

In simple terms, if a data file format supports a serialisation format, it can adapt to it. As with the previous rules, the purpose of this rule is not to perform any technical conversion, but to infer a semantic capability for reasoning about data interoperability and format compatibility. After applying this rule and inferring, Figure 4.6 lists all file formats that can be adapted to the RDF file format. In the figure, the inferred serialisation formats are listed in yellow.

DIIO Queries to support the competency questions on interoperability

DIIO provides domain knowledge on standards, their serialisation formats, extensions and supported standards. A list of useful SPARQL queries to address key DIIO competency questions that support the syntactic interoperability of IoT data in SWNs is as follows:

1. Figure 4.7 shows a SPARQL query to retrieve a list of registered standards and ontologies, along with their application domains.
2. Query for the list of IoT data registered in DIIF.

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX diio: <http://www.co-ode.org/ontologies/diio#>
SELECT ?iot ?dat_form WHERE { ?iot rdf:type diio:IoT_Data. ?iot diio:has_Serialisation_Format ?dat_form. }
```

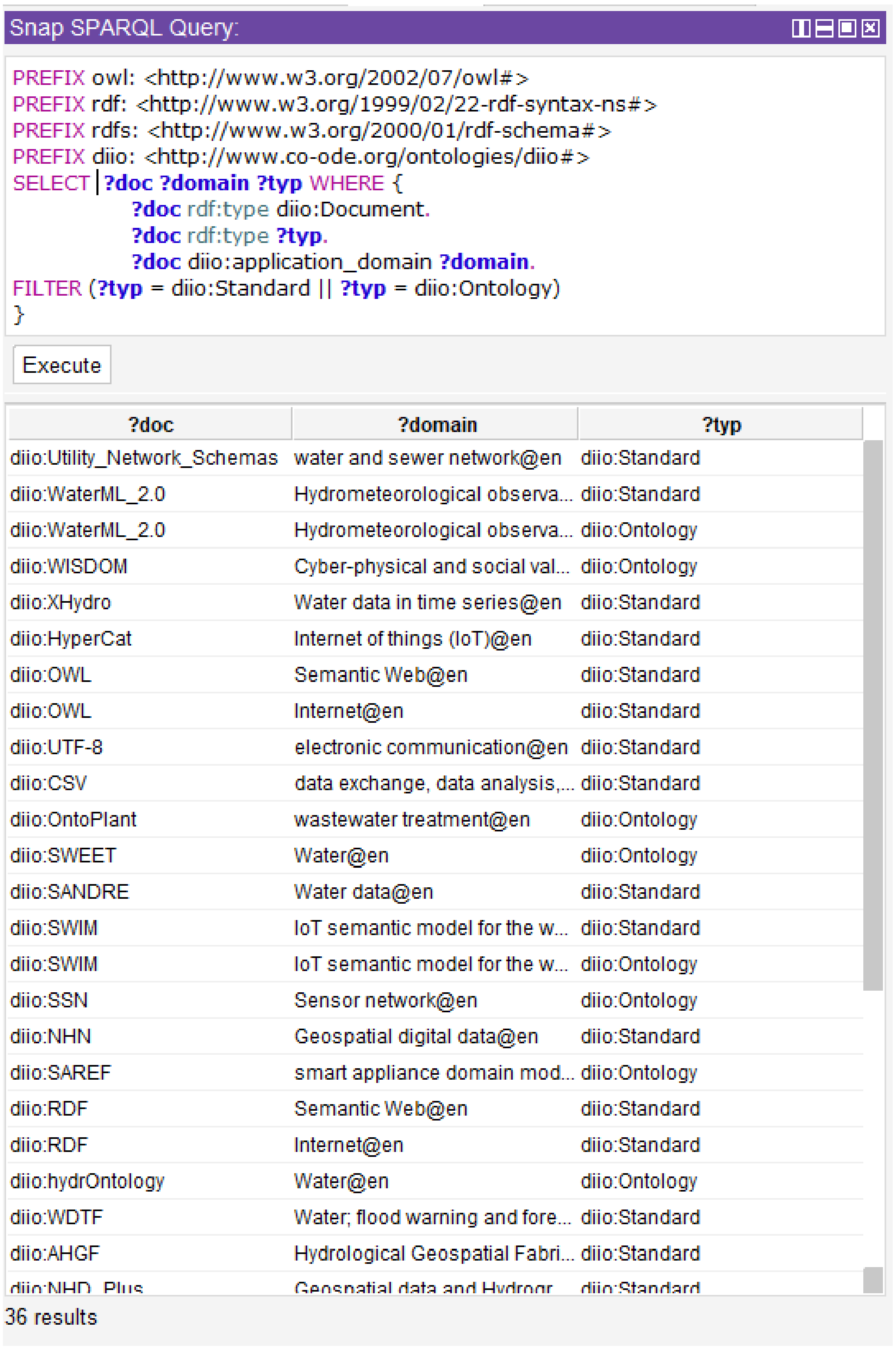
3. Query for a list of serialisation formats to serialise data in a file, and what their file extensions are.

```
... prefix statements ...
SELECT ?dat_form ?ext WHERE { ?iot rdf:type diio:IoT_Data. ?dat_form diio:file_extension ?ext. }
```

4. Query for a list of standards for data formats to store data in a file.

```
... prefix statements ...
SELECT ?dat_file_form WHERE { ?dat_file_form rdf:type diio:Data_File_Format. }
```

5. Query for a list of standards for data encoding in a file.

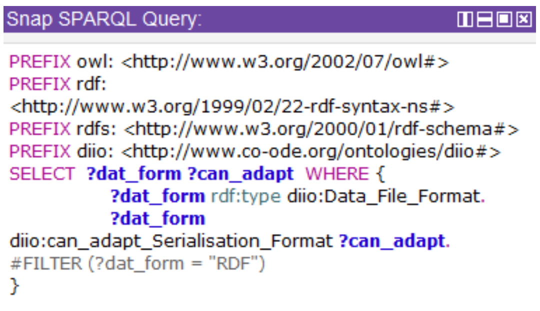


```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX diio: <http://www.co-ode.org/ontologies/diio#>
SELECT ?doc ?domain ?typ WHERE {
          ?doc rdf:type diio:Document.
          ?doc rdf:type ?typ.
          ?doc diio:application_domain ?domain.
FILTER (?typ = diio:Standard || ?typ = diio:Ontology)
}
```

?doc	?domain	?typ
diio:Utility_Network_Schemas	water and sewer network@en	diio:Standard
diio:WaterML_2.0	Hydrometeorological observa...	diio:Standard
diio:WaterML_2.0	Hydrometeorological observa...	diio:Ontology
diio:WISDOM	Cyber-physical and social val...	diio:Ontology
diio:XHydro	Water data in time series@en	diio:Standard
diio:HyperCat	Internet of things (IoT)@en	diio:Standard
diio:OWL	Semantic Web@en	diio:Standard
diio:OWL	Internet@en	diio:Standard
diio:UTF-8	electronic communication@en	diio:Standard
diio:CSV	data exchange, data analysis,...	diio:Standard
diio:OntoPlant	wastewater treatment@en	diio:Ontology
diio:SWEET	Water@en	diio:Ontology
diio:SANDRE	Water data@en	diio:Standard
diio:SWIM	IoT semantic model for the w...	diio:Standard
diio:SWIM	IoT semantic model for the w...	diio:Ontology
diio:SSN	Sensor network@en	diio:Ontology
diio:NHN	Geospatial digital data@en	diio:Standard
diio:SAREF	smart appliance domain mod...	diio:Ontology
diio:RDF	Semantic Web@en	diio:Standard
diio:RDF	Internet@en	diio:Standard
diio:hydrOntology	Water@en	diio:Ontology
diio:WDTF	Water; flood warning and fore...	diio:Standard
diio:AHGF	Hydrological Geospatial Fabri...	diio:Standard
diio:NHD_Plus	Geospatial data and Hydrogr	diio:Standard

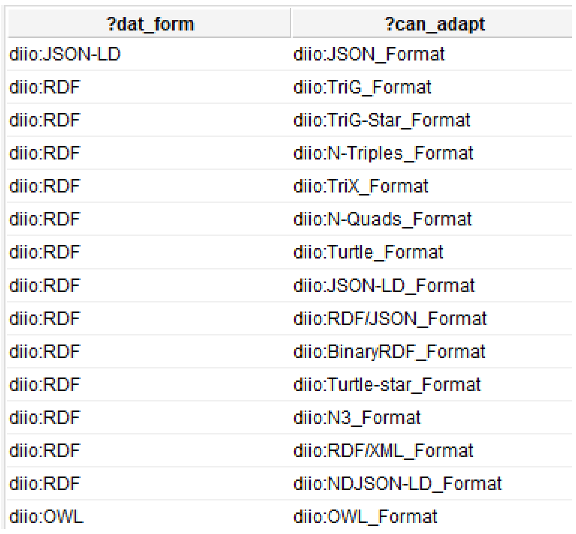
36 results

Figure 4.7: Query standards and ontologies in DIIO



```
Snap SPARQL Query:
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf:
<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX diio: <http://www.co-ode.org/ontologies/diio#>
SELECT  ?dat_form ?can_adapt  WHERE {
             ?dat_form rdf:type diio:Data_File_Format.
             ?dat_form
diio:can_adapt_Serialisation_Format ?can_adapt.
#FILTER (?dat_form = "RDF")
}
```

(a) Query in SPARQL



?dat_form	?can_adapt
diio:JSON-LD	diio:JSON_Format
diio:RDF	diio:TriG_Format
diio:RDF	diio:TriG-Star_Format
diio:RDF	diio:N-Triples_Format
diio:RDF	diio:TriX_Format
diio:RDF	diio:N-Quads_Format
diio:RDF	diio:Turtle_Format
diio:RDF	diio:JSON-LD_Format
diio:RDF	diio:RDF/JSON_Format
diio:RDF	diio:BinaryRDF_Format
diio:RDF	diio:Turtle-star_Format
diio:RDF	diio:N3_Format
diio:RDF	diio:RDF/XML_Format
diio:RDF	diio:NDJSON-LD_Format
diio:OWL	diio:OWL_Format

(b) Results

Figure 4.8: Query serialisation formats that can be adopted by which data formats in DIIO

> *... prefix statements ...*
>
> SELECT ?char_encode_form WHERE { ?char_encode_form rdf:type diio:Character_Encoding. }

6. Query for a list of applications used to store graph-based data.

> *... prefix statements ...*
>
> SELECT ?app WHERE { ?app rdf:type diio:Graph_Database. }

7. Query the list of converter applications that convert data from one format into another.

> *... prefix statements ...*
>
> SELECT ?conv WHERE { ?conv rdf:type diio:Converter. }

8. Query which ontology or standard is used in the sourced IoT data?

> *... prefix statements ...*
>
> SELECT ?iot ?uses_stand ?uses_onto WHERE { ?iot rdf:type diio:IoT_Data. ?iot diio:uses_Ontology ?uses_onto. ?iot diio:uses_Standard ?uses_stand. }

9. Query to find which serialisation formats can be adopted by which data file format standard. Figure 4.8 displays such a query and its results.

4.5.4 Semantic Similarity Scoring Ontology (S3O)

This section describes the proposed Semantic Similarity Scoring Ontology (S3O) (M. Singh, 2023), which stores terms used in IoT data and ontologies, along with the similarity scores computed by various algorithms. Additionally, it stores the directly calculated Similarity between any IoT data and an ontology. S3O covers all competency questions to support similarity calculation for aligning the IoT terms with domain-specific ontologies.

S3O competency questions to support similarity calculation

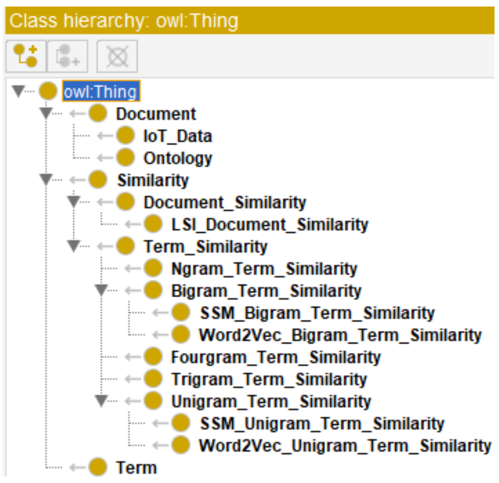
The list of competency questions supported by S3O is as follows:

1. Which ontologies, IoT datasets, and standards exist in DIIF's KB?
2. Which ontologies, IoT datasets, and standards are similar to each other?
3. Which terms are used in which ontologies, IoT datasets, and standards?
4. Which similarity algorithms exist in DIIF's KB?
5. Which terms are similar to each other according to which similarity algorithms?

S3O Conceptualisation

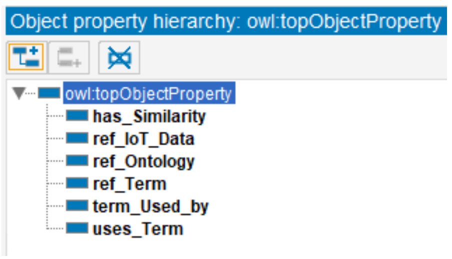
Figure 4.9 displays the conceptualisation of the S3O ontology that was developed in Protégé (Noy, Crubézy, et al., 2003). S3O ontology starts with an abstract class *Thing*. It has two data properties: *serialization_format*, which represents the data representation format, and *URI*, which serves as the identifier; its subclasses inherit both. Its direct subclasses are *Document*, *Term*, and *Similarity*. The *Term* class represents a word or phrase used to describe a thing or express a concept used in any IoT data or an ontology. For example, *Sensor* is modelled as an instance of the *Term* class. *Document* class represents an object of the text sequence type. S3O models *IoT_data* and *Ontology* as document objects for NLP. For example, SSN ontology is modelled as an instance of the *Ontology* class. And any *IoT_data* that contains *metadata* and *measurement data* accessible from any IoT device is modelled as an instance of the *IoT_data* class. Now the textual representation of the SSN ontology and IoT data, provided as instances, can be accessed by NLP algorithms for similarity computation. The *Ontology* class contains the terms, relations, and properties used to describe the data, information, and knowledge of a specific domain application. The *Similarity* abstract class abstracts over all similarity algorithms. For example, currently, NLP algorithms, *SSM_Similarity*, and *Word2Vec_Similarity* are defined as subclasses of the *Similarity* abstract class. *SSM_Similarity* represents SSM-based similarity, in which a SSM algorithm computes term similarity. *Word2Vec_Similarity* is word-to-vector-based similarity, where the Word2Vector (Word2Vec) algorithm is applied to calculate the similarity of documents and terms. Similarly, other similarity calculation algorithms can be defined as subclasses of the *Similarity* abstract class. It holds the *similarity_value* data property, which stores the similarity score of documents or terms as determined by the algorithm. S3O introduces *Ngram* classes to store the Similarity of combined terms, e.g. *temperature sensor* (bigrams) that appear as sequences of words in IoT data and ontologies. S3O has introduced an object property, *has_Similarity*, for storing information about similarity-based relationships among *IoT_Data*, *Ontology*, and *Term* classes. *ref_IoT_Data*, *ref_Ontology*, and *ref_Term* are object properties for references. *term_Used_by* and *uses_Term* are inverse object properties for storing information about when IoT data or ontologies use a *Term*.

S3O describes each entity using the $rdf : comment$ annotation to improve understanding. The current version of S3O has been added to the appendix and can also be accessed at the DIIF's Gitlab URL (M. Singh, 2025).



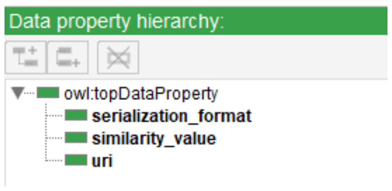
```
Class hierarchy: owl:Thing
▼ owl:Thing
  ▼ Document
      IoT_Data
      Ontology
  ▼ Similarity
    ▼ Document_Similarity
        LSI_Document_Similarity
    ▼ Term_Similarity
        Ngram_Term_Similarity
      ▼ Bigram_Term_Similarity
          SSM_Bigram_Term_Similarity
          Word2Vec_Bigram_Term_Similarity
        Fourgram_Term_Similarity
        Trigram_Term_Similarity
      ▼ Unigram_Term_Similarity
          SSM_Unigram_Term_Similarity
          Word2Vec_Unigram_Term_Similarity
    Term
```

(a) Class entities



```
Object property hierarchy: owl:topObjectProperty
▼ owl:topObjectProperty
    has_Similarity
    ref_IoT_Data
    ref_Ontology
    ref_Term
    term_Used_by
    uses_Term
```

(b) Object properties



```
Data property hierarchy:
▼ owl:topDataProperty
    serialization_format
    similarity_value
    uri
```

(c) Data properties

Figure 4.9: Conceptualisation of S3O

4.6 DIIM Semantic Similarity Scoring Tool (DS3T)

DIIM Semantic Similarity Scoring Tool (DS3T) is built on the conceptual Data Information Interoperability Model (DIIM) (M. Singh, W. Wu, et al., 2022). As elaborated in Section 3.3, DIIM takes any IoT dataset, domain-specific standards, and ontologies as input and generates an annotated IoT KG as output. DIIM's transformation step (see subsection 3.3.1) converts the semi-structured IoT dataset into an IoT KG. DIIM's alignment step (see subsection 3.3.2) aligns the terms/labels (used for data/values) in the IoT dataset with the terms/words used in ontologies to describe concepts, relations, instances, and axioms. DIIM's storage step (see subsection 3.3.2) stores the alignments in the IoT KG as annotations on the respective terms, linking them to related ontologies.

4.6.1 Implementation of DS3T

The implementation of DIIM Semantic Similarity Scoring Tool (DS3T) in Data Information Interoperability Framework (DIIF) begins with functionalities, such as term extraction, ontology generation, similarity calculation, term alignment, and annotation from IoT data and domain-specific ontologies, as they are the core functionalities of the framework to enable interoperability of IoT data. In this context, DS3T is implemented in the interpreted high-level programming language *Python* (Foundation, 2001), as renowned NLP technologies are available as executable libraries. DS3T's code was developed in *Visual Studio Code* (Microsoft, 2015) and managed in GitLab (GitLab, 2014) under the repository URL (M. Singh, 2025). It is assumed that DS3T will be further developed to meet use-case-specific requirements. However, a list of implemented features and functionalities in the current DS3T version is as follows:

- **Extraction of terms from DIIM input sources**, e.g. JSON (IoT data) and OWL (domain-specific knowledge in ontologies) formatted files. Using libraries that implement NLP algorithms, each extracted term is cleaned, and its synonyms and lemmas are identified.
- **Creation of DIIM Datasets and Similarity Matrices objects** that store the processed information on input data and ontologies. The specifications of DIIM_Dataset and

Table 4.2: DIIM's dataset object for storage of input data

DIIM_Dataset		
Attribute	**Data Type**	**Description**
id	int	DIIM's unique identification number
name	str	Name of the dataset
source_filepath	str	File path of the sourced dataset as input to DIIM
size	int	Size of the sourced dataset file
type	str	Type of the data format, e.g. JSON or OWL
terms_sourced	list[str]	List of sourced terms after extraction
terms_sourced_count	int	Count of sourced terms after extraction
terms_cleaned	list[str]	List of cleaned sourced terms after extraction
terms_cleaned_count	int	Count of cleaned sourced terms after extraction
synonyms	dict(str,list[str])	For each term in the dataset, there is a list of synonyms found by Wordnet (Princeton University, 2021) Natural Language Toolkit (NLTK) (Bird et al., 2022)
synonyms_count	int	Count of synonyms
lemmas	dict(str,list[str])	For each synonym, there is a list of lemmas found by Word-net NLTK
lemmas_count	int	Count of lemmas

DIIM_Similarity are described in Tables 4.2 and 4.4, respectively.

- **Serialisation and deserialisation of the information** generated by DIIM about extracted terms and their similarities. This helps reduce the resources the Central Processing Unit (CPU) uses to analyse and compute similarity when the program is restarted.
- **Visualisation of DIIM similarity matrices** with MatPlotLib (Matplotlib Development Team, 2003).
- **Logging the activity of the DIIM** in the *diim.log* file via the Python logging library.

Processing of input datasets and ontologies

DIIM processes the input datasets and ontologies in the following steps and keeps the relevant information in DIIM storage objects:

- *Extraction of terms:* DIIM extracts all keys from a JSON file. If the data is nested within sub-elements, it is flattened into a simple dictionary object. For OWL files, RDFLIB (RDFLib Team, 2009) is used to extract terms. Only lexical terms are stored for further processing. Extracted terms are called *terms_sourced* because they are extracted from the sourced input.
- *Cleaning of terms:* All extracted terms are cleaned to obtain lexical terms made of letters.

Table 4.3: DIIM's term extraction statistics

Input Source	Format	Extracted Terms	Cleaned Terms	Wordnet Synonyms	Wordnet Lemmas
Reservoir IoT data	JSON	9	11	11	58
COSMO	OWL	27853	27796	27796	32994
DOLCE Ultra Lite (DUL)	OWL	193	192	192	481
GO	OWL	44	57	57	171
SSN	OWL	111	110	110	186
SWIM	OWL	68	67	67	936
WISDOM	OWL	678	675	675	831
SWIM	OWL	68	67	67	936
Water Quality Ontology	OWL	100	115	115	147

These terms are called *terms_cleaned*. This step requires further pattern recognition to filter and split compound words, e.g. AbstractionPoint, FoulPumpingStation, hasParticipant, etc.

- *Synonyms and lemmas:* Wordnet NLTK is used to find synonyms for each cleaned term, and all found synonyms are stored in a Python dictionary object, where cleaned terms are keys, and a list of synonyms is the value. Similarly, all lemmas are queried and stored in a Python dictionary object for each synonym. Here, synonyms are keys, and a list of lemmas is the value.

Table 4.3 presents the statistics on the term extraction by DS3T. For example, DS3T extracts from Reservoir IoT data in JSON format following terms: *(auto~temperature, mac, model, Name, serial, Description, latitude, longitude, timestamp, value, values)*. DS3T extracted 9 terms, then cleaned the list, yielding 11 cleaned terms. Wordnet NLTK has found 11 synonyms and 58 lemmas for cleaned terms.

Calculating the term similarity

The current implementation of DIIM uses Python libraries to compute term similarity between input datasets and the ontologies by applying SSM and Word2Vec (see Section 2.9.3). Table 4.3 provides the statistics on term extraction from input datasets and ontologies. The similarity calculation of all terms with each other takes more than 10 hours to complete. Table 4.4 presents the definition of the DIIM_Similarity object, which stores the relevant information for the similarity calculation.

Table 4.4: DIIM's similarity object to store similarity calculation

DIIM_Similarity		
Attribute	**Datatype**	**Description**
id	int	DIIM's unique identification number
algorithm_info	str	Information on the used similarity calculation algorithm, e.g. Spacy or Levenshtein
dataset1_name	str	Name of the dataset/ontology 1
dataset1_cleanedterms	list[str]	List of cleaned terms from first dataset
dataset2_name	str	Name of the dataset/ontology 2
dataset2_cleanedterms	list[str]	List of cleaned terms from second dataset
similarity_matrix	dict(str,list[str])	A dictionary-shaped matrix to store the similarity values between dataset 1 and 2 terms. The terms from dataset 1 are saved in the first column (0th index) under 'token'. The terms from dataset 2 are saved as columns at index 1.
filePath	str	Path of the file that serialises the similarity Matrix stored in this object

4.6.2 DS3T's executional routine with Word2Vec and SSM algorithms

All ontologies and standards must be reviewed, regardless of the alignment process (manual or computer-assisted). The number of ontologies and standards to be reviewed in a computer-assisted alignment can be high when all possible alignments are searched across a massive repository. Here, NLP-assisted alignment of IoT data with domain-specific applications could be beneficial, aiding semantic communication. In a computer-assisted alignment process, various algorithms are used to measure term similarity; indeed, new algorithms will be developed as cases become more specific. Across multiple use cases, it is necessary to consider different algorithmic results. Therefore, a system must store information about the applied algorithms, their calculated similarity scores, and the terms with their reference relations. An ontology has an advantage over a database schema because entities in an ontology can be linked directly to their original ontologies, and reasoning can be based on similarity scores computed by different algorithms. In this context, DS3T's approach includes the following steps:

Step 1: Build a corpus from IoT data and ontologies: The first step is to create corpora of IoT data and domain-specific ontologies to enable NLP to compute term similarity. Algorithm 1 explains the process of building a corpus from a given URI. A NLP corpus (a text-only version of IoT data and ontological information, prepared specifically for language-based analysis) is the textual representation of IoT datasets and ontologies without any digits or special characters. It is constructed by removing digits and special characters while preserving words (text) in

IoT datasets and ontologies as they occur. Table 4.5 lists the parameters and functions used in defined algorithms. URIs represent Inputs and outputs. Algorithm 1 is described as follows:

Purpose: The goal of this algorithm is to build a clean, tokenised NLP corpus from a collection of text files located at a given URI (e.g. a directory or web resource). Essentially, it reads all text files, cleans and preprocesses the text, and stores the resulting tokenised documents in a list-like structure called a corpus.

Inputs and Outputs: Input: $dataDir$, a URI or directory containing raw text files. Output: corpora, a structured collection (list) of processed and tokenised text data, ready for NLP tasks (like training models, text mining, etc.).

Line-by-Line Description:

- Line 2 Initialise the corpus list: Create an empty list of corpora to store processed text data from each file.
- Iterate each file (Lines 3–11): Loop through all files found in $dataDir$.
- Line 4 Read the file: Use $readFile(file)$ to load the contents of each text file into memory as a string, called $fileString$.
- Line 5 Convert to lowercase: Standardise text by converting all characters to lowercase. Example: "Hello World!" $\rightarrow$ "hello world!"
- Line 6 Remove numeric characters: Eliminate all digits and numbers that might not contribute meaningfully to language analysis.
- Line 7 Remove special characters: Strip punctuation and symbols (e.g. @, #, !, ?, etc.) to retain only alphabetic content.
- Line 8 Remove stop-words: Filter out common words that add little semantic value, such as "the," "is," "in," "and," etc.
- Line 9 Tokenise the text: Split the cleaned string into tokens (usually words or meaningful units). Example: "water temperature sensor measurement" $\rightarrow$ ["water", "temperature", "sensor", measurement].
- Line 10 Add tokens to corpus: Append the resulting list of tokens to the corpus list.
- Line 12 Return the complete corpus: After processing all files, return the compiled corpus,

Table 4.5: Parameters and functions used in pseudo-algorithms for steps 1-5

Name	**Description**
$dataDir$	The URI of the directory or repository where IoT_{data} or $Ontos_{data}$ are held in Unicode Transformation Format – 8-bit (UTF-8) format.
IoT_{data}	Semi-structured IoT data presented in human-machine readable UTF-8 text format and serialized in JSON or CSV file.
$Ontos_{data}$	Ontologies described in human-machine readable UTF-8 text format and serialized in a text, XML, HyperText Markup Language (HTML), RDF, or OWL file.
$corpora$	It is a collection of corpora that is processed after reading the resource from a given URI of IoT_{data} or $Ontos_{data}$.
IoT_{corpus}	Corpus of IoT data presented in human-machine readable UTF-8 text file.
$Ontos_{corpus}$	Corpus of ontology presented in human-machine readable UTF-8 text file.
f_{corpus}	This function takes IoT_{data} or $Onto_{data}$ as input and transforms the input to a IoT_{corpus} or $Onto_{corpus}$ that is suitable for NLP operations.

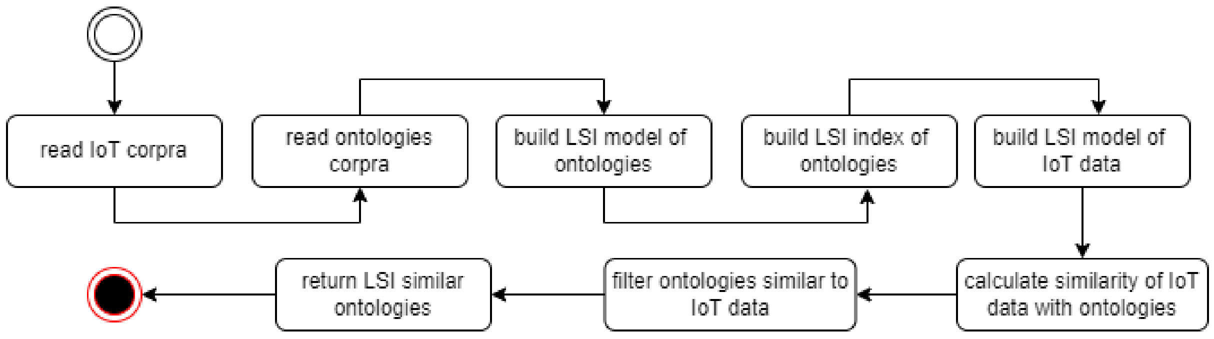

read IoT corpra
read ontologies corpra
build LSI model of ontologies
build LSI index of ontologies
build LSI model of IoT data
calculate similarity of IoT data with ontologies
filter ontologies similar to IoT data
return LSI similar ontologies

Figure 4.10: Build LSI from IoT data or ontologies and calculate their similarity score

which is a list of tokenised, cleaned documents.

Step 2: Finding potential ontologies for alignment: In this step, DS3T shortlists the ontologies that resemble the given IoT data to reduce computational time. Because there may be many ontologies for processing, some may be aligned and others may not. This step can be skipped if the number of given ontologies is minimal. Figure 4.10 shows DS3T's activity diagram for calculating the similarity score in this step. In the following algorithm 2, DS3T uses LSI to identify term-similarity-based relationships between the given IoT dataset and ontologies. However, any other similarity algorithm could be applied to shortlist the relevant ontologies. The Gensim library (Řehůřek, 2009) creates an LSI model for each ontology and indexes these models. Finally, the index is compared with the LSI model of IoT data to calculate their similarity/relatedness. Algorithm 2 is described as follows:

Purpose: This algorithm constructs Latent Semantic Indexing (LSI) models from two

Algorithm 1 Build an NLP corpus from a given URI

```
Input $dataDir$ /* URI */
Output $corpora$ /* URI */
 1: procedure $f_{corpus}$
 2:     $corpora \leftarrow List()$
 3:     for each $file \in dataDir$ do
 4:         $fileString \leftarrow readFile(file)$
 5:         $lowerString \leftarrow lowerCase(fileString)$
 6:         $cleanedString \leftarrow removeNumeric(lowerString)$
 7:         $cleanedString \leftarrow removeSpecialChars(cleanedString)$
 8:         $cleanedString \leftarrow removeStopwords(cleanedString)$
 9:         $corpus \leftarrow tokenize(cleanedString)$
10:         $corpora.add(corpus)$
11:     end for
12:     return $corpora$
13: end procedure
```

text-based corpora: one derived from IoT (Internet of Things) data and the other from ontologies. Then calculates their semantic similarity. In simpler terms, it determines the degree of similarity between IoT textual data and ontology concepts by comparing their latent semantic representations.

Inputs and Outputs Input: pathOfIoTCorpus: file path or URI to the preprocessed IoT text corpus. pathOfOntoCorpus: file path or URI to the ontology text corpus. Output: pathOfLsiSimilarOntos: file path or URI where the results (similar ontologies) are saved.

Line-by-Line Description:

- Line 1 Initialise Procedure: Defines the function that performs LSI-based similarity calculation between IoT and ontology corpora.
- Line 2 Read Corpora: Load both text corpora into memory. iotCorpus: textual descriptions from IoT data sources. ontoCorpus: textual data from ontology definitions or metadata.
- Line 3 Build Ontology LSIModel: Apply Latent Semantic Indexing (LSI) to the ontology corpus. LSI reduces the dimensionality of the textual data and captures semantic relationships between words and concepts.
- Line 4 Build Ontology LSI Index: Create an index (a searchable data structure) from the

ontology LSI model. This index enables fast similarity computation against new input data (in this case, IoT data).

- Line 5 Build an IoT LSI model: Construct a similar LSI representation for IoT data. Each IoT document is projected into the same latent semantic space as the ontology model for comparison.
- Line 6 Compute Similarity: Measure the semantic similarity between the IoT corpus and the ontology corpus using their LSI vectors. Typically, cosine similarity is used to quantify how "close" two document vectors are in semantic space. High score (close to 1): IoT and ontology documents are semantically related. Low score (close to 0): They are dissimilar.
- Line 7 Filter Similar Ontologies: Select only those ontology entries whose similarity scores exceed a predefined threshold (e.g. 0.7). This filters out weak or irrelevant matches, leaving only semantically similar ontologies.
- Line 8 Save Results: Store the filtered list of similar ontologies (and possibly their similarity scores) to a file. Return the file path to the results.
- Line 9 Return Output: Output the file path or URI containing the LSI-based similarity results.

Algorithm 2 Build LSI from IoT data or ontologies and calculate their similarity score

```
Input path of $IoT_{corpus}$ and $Onto_{corpus}$
Output $pathOfLsiSimilarOntos$
 1: procedure $f_{lsiSimilarity}$
 2:     $iotCorpus \leftarrow readCorpus(pathOfIoTCorpus)$
 3:     $ontoCorpus \leftarrow readCorpus(pathOfOntoCorpus)$
 4:     $ontoLSIModel \leftarrow buildLSIModel(ontoCorpus)$
 5:     $ontoLSIIndex \leftarrow buildLSIIndex(ontoLSIModel)$
 6:     $iotLSIModel \leftarrow buildLSIModel(iotCorpus)$
 7:     $iotOntoLSISimilarity \leftarrow calculateSimilarity(ontoLSIIndex, iotLSIModel)$
 8:     $lsiSimilarOntos \leftarrow filterOntologies(iotOntoLSISimilarity, threshold)$
 9:     $pathOfLsiSimilarOntos \leftarrow writeFile(lsiSimilarOntos)$
10:     return $pathOfLsiSimilarOntos$.
11: end procedure
```

Step 3: Build dictionaries and Word2Vec models of IoT data and ontologies: As shown in Algorithm 3, DS3T first builds dictionaries (a list of used terms) and Word2Vec models (a representation of used terms as vectors) from the given IoT corpus and ontology corpora (a list of corpora). Then, DS3T trains Word2Vec models of ontologies with the given IoT corpus. Algorithm 3 is described as follows:

Purpose: The algorithm constructs Word2Vec models and word dictionaries for both IoT data corpora and Ontology corpora. It then combines and trains these models to form a joint Word2Vec model across the two domains, thereby capturing shared semantic relationships. The resulting models enable cross-domain semantic alignment, thereby linking IoT terms to ontology concepts using learned vector representations.

Input and Output: Input $iotCorpora$: preprocessed and tokenised IoT text $corpus$, $ontosCorpora$: preprocessed ontology text corpus. Each corpus is a list of tokenised documents. Output $iotW2vModels$: Word2Vec models trained on IoT data, $iotDicts$: word–index dictionaries for IoT corpora, $ontosW2vModels$: Word2Vec models trained on ontology data, $ontosDicts$: corresponding dictionaries for ontology data, and $iotOntosW2vModels$: combined IoT–ontology Word2Vec models capturing cross-domain semantics.

Line-by-Line Description:

- Line 1 Start the Procedure: This defines the function that builds all the required Word2Vec models and dictionaries.
- Line 2 Build a Word2Vec model for ontologies: It trains a Word2Vec model using the ontology corpus. The result is a vector space in which each ontology term (e.g., sensor, temperature, and measurement) is represented by a dense vector. These embeddings capture semantic relationships between ontology terms, for example, vector("sensor") ≈ vector("device") and vector("data") ≈ vector("information").
- Line 3 Build Word2Vec Model for IoT Data: Similarly, it trains a Word2Vec model using IoT-related text data. It learns contextual meaning of IoT terms, for example, vector("sensor") ≈ vector("temperature") and vector("device") ≈ vector("controller"). We now have two separate semantic spaces: one for ontology concepts and one for the IoT domain language.
- Lines 4–5 Build Dictionaries: Each dictionary maps words to their corresponding indices or embedding vectors. This helps standardise word access and lookups during model

comparison or retraining.

- Line 6 Initialise Combined Model List: A list to store all combined Word2Vec models for IoT ontologies.
- Lines 7–14 Combine IoT and Ontology Models: There are two loops: the outer and the inner. The outer loop iterates over each IoT dataset (useful when multiple IoT sources are available). For every ontology model, the inner loop performs retraining or fine-tuning using the IoT corpus. This merges the conceptual understanding of ontology with the semantics of real-world IoT data. Analytically, this adjusts word vectors to reflect cross-domain semantic alignment. For example, if "sensor" frequently co-occurs with "temperature" in IoT data, and "sensor" is linked to "device" in the ontology, the retrained model now semantically connects all three terms. This creates a shared embedding space bridging the IoT and ontology terminologies. After processing all ontology models, it stores all IoT ontology models generated from this iteration.
- Line 15 End Procedure: The algorithm prepares all Word2Vec models, dictionaries, and hybrid models for later use.

Algorithm 3 Build Word2Vec models and dictionaries of IoT data and ontologies

Input $iotCorpora$ and $ontosCorpora$

Output $iotW2vModels$, $iotDicts$, $ontosW2vModels$, $ontosDicts$, and $iotOntosW2vModels$

```
1:  procedure $f_{w2v}$
2:      $ontosW2vModels \leftarrow$ buildW2VModel(ontosCorpora)
3:      $iotW2vModels \leftarrow buildW2VModel(iotCorpora)$
4:      $ontosDicts \leftarrow buildDict(ontosW2VModels)$
5:      $iotDicts \leftarrow buildDict(iotW2VModels)$
6:      $allIotOntosW2vModels \leftarrow List()$
7:      for each $iotCorpus \in iotCorpora$ do
8:          $iotOntosW2vModels \leftarrow List()$
9:          for each $ontoW2vModel \in ontoW2vModels$ do
10:             $iotOntoW2vModel \leftarrow train(ontoW2vModel, iotCorpus)$
11:             $iotOntosW2vModels.add(ontoWIotW2vModel)$
12:         end for
13:         $allIotOntosW2vModels.add(iotOntosW2vModels)$
14:     end for
15: end procedure
```

Step 4: Calculate algorithm-based similarity score of terms in IoT data and ontologies:

In this step, DS3T calculates the algorithm-based similarity of each term in IoT data with terms used in the given ontologies. In Algorithm 4, DS3T uses Word2Vec similarity and String-search Matching (SSM) algorithms to demonstrate the similarity calculation procedure. Hence, other similarity calculation algorithms can be incorporated into the procedure to obtain more preferred results. As outputs, the $iotOntoW2vSimilarityList$ and $iotOntoSsmSimiarity$ lists contain all terms from the IoT data and ontologies, along with their calculated Word2Vec and SSM similarity scores. Algorithm 4 is described as follows:

Purpose: Conceptually, the algorithm performs a hybrid semantic matching. The SSM component captures explicit ontology-level semantics (e.g. hierarchical or conceptual closeness) via string matching. The Word2Vec component captures implicit semantic similarity from vector-space representations of words in context. Combining these two enables more robust alignment of IoT ontology terms, improving interoperability and semantic understanding across heterogeneous IoT systems.

Input and Output: Inputs: $iotW2vModels$ Word2Vec models trained on IoT data, $iotDicts$ Dictionaries of terms extracted from IoT data, $ontosW2vModels$ Word2Vec models trained on ontology terms, $ontosDicts$ Dictionaries of ontology terms, and $iotOntosW2vModels$ Joint or aligned Word2Vec models for IoT–ontology mapping. Outputs: $iotOntoW2vSimilarityList$ Word2Vec-based term similarity scores between IoT and ontology terms, and $iotOntoSsmSimilarity$ Semantic similarity scores (from ontology structure or semantic distance).

Line-by-Line Description:

- Step 1 Start and initialisation of the Procedure (Line 1-3): This defines the function that requires two lists, iotOntoW2vSimilarityList and iotOntoSsmSimilarityList, based on the given similarity algorithms, e.g. SSM and Word2Vec. These lists will store similarity scores for each pair of IoT and ontology terms.
- Step 2 SSM-Based Similarity Calculation (Lines 4–16): For each IoT dictionary ($iotDict$) in the collection $iotDicts$, iterate each IoT term ($iotTerm$). For each ontology dictionary ($ontoDict$) in $ontosDicts$, iterate through each ontology term ($ontoTerm$). Compute a semantic similarity between the IoT term and the ontology term. Store each pairwise

similarity score in a temporary list for that IoT term ($iotOntoSsmTermSimilarity$). After processing all ontology terms for one IoT term, append the list of scores to $iotOntoSsmSimilarityList$.

- Step 3 Word2Vec-Based Similarity Calculation (Lines 17–27): For each IoT dictionary and IoT term, initialise a list to store the similarity of this term with ontology embeddings. For each joint IoT–ontology Word2Vec model in $iotOntosW2vModels$, compute the cosine similarity between the IoT term vector and ontology term vectors in that model. Store the resulting similarities for that IoT term and add them to $iotOntoW2vSimilarityList$.
- Step 4 Output: Finally, the algorithm returns two key structures, $iotOntoW2vSimilarityList$, Word2Vec-based similarities and $iotOntoSsmSimilarityList$, semantic-based similarities. These lists can then be merged or used separately, depending on the application, e.g. for ontology mapping, semantic search, knowledge graph enrichment, or automatic alignment between IoT vocabularies and ontologies.

Step 5: Build an ontology and store algorithm-based similarity scores: In the final step, DS3T builds an ontology that stores information about the similarity of IoT data to the ontology. It also stores the algorithm-based similarity score between two terms in IoT data and ontologies. Algorithm 5 describes the procedure for populating the ontology with facts and creating similarity relationships among entities. S3O contains all terms related to IoT data and ontologies, along with their calculated Word2Vec and SSM similarity scores. It also includes the LSI similarity score for the given IoT data relative to the ontologies. Algorithm 5 is described as follows:

Purpose: This algorithm's purpose is to integrate similarity scores (calculated by Algorithm 4) into an existing ontology, resulting in an enriched ontology model called S3O.

Inputs and Outputs: Inputs: $IoTcorpus$ Path to the IoT corpus or dictionary source, $Ontocorpus$ Path to the ontology corpus containing ontology terms and relationships, and S3O The ontology model to be enriched or updated with similarity data. Output: S3O updated (enriched) ontology that includes IoT-ontology term relations and similarity scores.

Line-by-Line Description:

Algorithm 4 Calculate algorithm-based similarity score of terms in IoT data and ontologies

Input $iotW2vModels$, $iotDicts$, $ontosW2vModels$, $ontosDicts$, and $iotOntosW2vModels$
Output $iotOntoW2vSimilarityList$ and $iotOntoSsmSimiarity$

```
1:  procedure $f_{sim}$
2:      $iotOntoW2vSimilarityList \leftarrow List()$
3:      $iotOntoSsmSimiarityList \leftarrow List()$
4:  /* SSM-based similarity calculation*/
5:      for each $iotDict \in iotDicts$ do
6:          for each $iotTerm \in iotDict$ do
7:              for each $ontoDict \in ontoDicts$ do
8:                  $iotOntoSsmTermSimiarity \leftarrow List()$
9:                  for each $ontoTerm \in ontoDict$ do
10:                     $iotOntoSsmSimilarity \leftarrow calculateSsm(iotTerm, ontoTerm)$
11:                     $iotOntoSsmTermSimiarity.add(iotOntoSsmSimilarity)$
12:                 end for
13:                 $iotOntoSsmSimiarityList.add(iotOntoSsmTermSimiarity)$
14:             end for
15:         end for
16:     end for
17: /* Word2Vec-based similarity calculation*/
18:     for each $iotDict \in iotDicts$ do
19:         for each $iotTerm \in iotDict$ do
20:             $iotOntoW2vTermSimiarity \leftarrow List()$
21:             for each $iotOntosW2vModel \in iotOntosW2vModels$ do
22:                 $iotOntoW2vSimilarity \leftarrow calculateW2vSimilarity(iotTerm, iotOntosW2vModel)$
23:                 $iotOntoW2vTermSimiarity.add(iotOntoW2vSimilarity)$
24:             end for
25:             $iotOntoW2vSimilarityList.add(iotOntoW2vTermSimiarity)$
26:         end for
27:     end for
28: end procedure
```

- Step 1 Initialise Procedure (Line 1–2): S3O is the main procedure. This loads the S3O ontology's structure, such as classes, instances, and relationships, serving as the base into which new similarity information will be inserted.
- Step 2 Create Term Relations (Line 3): This step links IoT terms and ontology terms inside the ontology model. Each IoT term from $iotDictList$ and each ontology term from $ontosDictList$ is represented as a concept or instance in the ontology. Relationships such as $hasIoTTerm$, $mapsToOntologyTerm$, or $isSimilarTo$ might be created. The aim is to explicitly model all IoT and ontology terms as nodes/entities within the ontology graph.
- Step 3 Create Similarity Relations (Lines 4–6): The algorithm now integrates the different similarity scores computed earlier (in Algorithm 4). The aim is to enrich the ontology with multiple semantic layers of similarity: Contextual (Word2Vec), Latent Semantic Conceptual (LSI), and Hierarchical/Structural (SSM). Each relation provides a different perspective on how an IoT term aligns with ontology terms.
- Step 4 Write the Updated Ontology (Line 7): The final ontology (S3O) is written to a file or database. This file contains all IoT and ontology terms, along with their similarity-based connections. The aim is to persist the enriched ontology so it can be reused by IoT systems, semantic reasoners, or alignment tools.
- Step 5 Return the Updated Path (Line 8): Returns the file path where the enriched ontology has been stored. So other modules or services can load it for semantic queries or reasoning.

Algorithm 5 Store the algorithm-based similarity scores in an ontology

Input path of IoT_{corpus}, $Onto_{corpus}$, and S3O
Output S3O

```
1: procedure $f_{s3o}$
2:     s3o ← loadS3OSchema()
3:     s3o ← createTermRelations(s3o,iotDictList,ontosDictList)
4:     s3o ← createSimilarityRelations(s3o,iotOntoW2vSimilarityList)
5:     s3o ← createSimilarityRelations(iotOntoLSISimilarity)
6:     s3o ← createSimilarityRelations(s3o,iotOntoSsmSimiarity)
7:     pathOfS3o ← writeFile(s3o)
8:     return pathOfS3o.
9: end procedure
```

4.7 Data Information Interoperability Questionnaire (DIIQ)

This section presents the proposed test-based Data Information Interoperability Questionnaire (DIIQ) (M. Singh, 2024) to validate the interoperability between two IoT-enabled applications. The Data Information Interoperability Questionnaire (DIIQ) was developed as a structured, test-based instrument for evaluating the interoperability of two IoT-enabled applications at both the syntactic and semantic levels. Interoperability assessment in IoT environments requires not only technical validation, such as verifying data formats, aligning vocabulary, and ensuring unit consistency, but also an informed interpretation of test results. The Data Information Interoperability Questionnaire (DIIQ) formalises this assessment process by guiding respondents through a set of targeted questions derived from the DIIM's interoperability requirements and its implementation within the DIIF. When DIIM and DIIF are utilised in use cases, DIIQ helps to validate the achieved interoperability. Therefore, DIIQ is essential for attaining Objective 5.

Disclaimer: This Questionnaire is used only for research purposes to validate the interoperability of IoT-enabled applications at the syntactic and semantic levels. It does not collect personal information.

The current version of the DIIQ comprises three sections, each containing questions based on test cases for syntactic and semantic interoperability. However, the Questionnaire could be extended to meet the interoperability requirements of the application scenario. The Questionnaire consists of the following three sections, each serving a distinct purpose within the overall validation workflow:

1. The **Personal Questionnaire** assesses the respondent's domain knowledge, technical expertise, and familiarity with relevant concepts, including data formats, IoT terminology, ontology alignment, and NLP similarity measures. This ensures that the subsequent interoperability assessments are performed by an adequately qualified individual, thereby improving the reliability of the evaluation. The higher a respondent's score, the greater the respondent's credibility.
2. The **Syntactic Interoperability Questionnaire** focuses on verifying whether the two datasets under consideration can exchange information at the structural level. Respondents

conduct a set of syntactic test cases, such as verifying data format equivalence, validating data representation correctness, and confirming the preservation of measurement values, and then answer the corresponding questions. The outcomes enable the classification of syntactic interoperability as full, partial, or none.

3. The **Semantic Interoperability Questionnaire** evaluates whether the meaning of data is preserved across applications. This section is grounded in the semantic interoperability requirement matrix and requires respondents to confirm the alignment of domain terms and measurement units across the two information sources. For each term in the selected set, the Questionnaire assesses whether a corresponding aligned term and unit exist in the other application. As with the syntactic section, results are categorised as full, partial, or no interoperability.

The **results calculation** can yield full (all test questions are answered with "Yes"), partial (at least two are answered, one with "Yes" and the others with "No"), or no (all test questions are answered with "No") interoperability. The expected completion time depends on the respondent's domain knowledge and ability to run the test cases. A respondent with sufficient domain knowledge and skills, e.g. a PhD student like me who holds a Master's degree in computer science and has domain knowledge in IoT and SWNs, should be able to complete the Questionnaire within a week. A copy of DIIQ is added to the appendix.

Collectively, the DIIQ provides a rigorous, repeatable, and methodologically transparent approach for evaluating the interoperability outcomes produced through DIIM and DIIF. While the current version is designed for human administration, its structure is compatible with future automation, particularly in contexts where software agents can systematically execute interoperability testing. The DIIQ therefore serves both as a practical assessment tool and as a formal methodological component in research addressing syntactic and semantic interoperability in IoT-based information systems.

4.7.1 DIIQ Personal

Currently, we assume a human respondent will complete the Questionnaire. Therefore, the personal questionnaire section is essential. This Questionnaire does not involve any tests and could be completed by a human respondent in five to ten minutes. The following essential questions are raised in the personal Questionnaire to a respondent:

1. What is the highest academic title you currently hold?
2. How would you rate your proficiency in the English language?
3. How familiar are you with computer science?
4. How familiar are you with the data representation formats, such as CSV, XML, JSON, RDF, etc?
5. How familiar are you with terms used in the water domain, such as pH, temperature, turbidity, Fahrenheit, Celsius, do, etc.?
6. How familiar are you with terms used in the IoT domain, such as sensor, timestamp, measurement, location, longitude, latitude, etc.?
7. How familiar are you with the NLP concepts, such as terms/words, unigrams, bigrams, similarity/relatedness, semantics, syntactics, etc.?
8. How familiar are you with the word similarity scoring algorithms, such as Word to Vector, Levenshtein, String Syntax Matching, etc.?
9. How familiar are you with the process of word mapping?
10. How familiar are you with the process of mapping/aligning datasets?
11. How familiar are you with the process of mapping/aligning ontologies?
12. Please rate the similarity of the following two terms to label the measurement values of a sensor: Sensor, s
13. Please rate the similarity of the following two terms to label measurement values of a sensor: Sensor, sen
14. Please rate the similarity of the following two terms to label measurement values of a temperature sensor: Temperature, temp
15. Please rate the similarity of the following two terms to label measurement values of a

sensor: Dissolved Oxygen, DO

16. Please rate the similarity of the following two terms to label measurement values of a sensor: Water volume, WaterVolume
17. Please rate the similarity of the following two terms to label measurement values of a sensor: DissolvedOxygen, Dissolved Oxygen
18. Please rate the similarity of the following two terms to label measurement values of a sensor: ph, power of hydrogen
19. Please rate the similarity of the following two terms to label data values: ph, potential of hydrogen
20. Do you have access to all resources related to the Questionnaire?

A respondent can access the DIIQ Personal Questionnaire online (M. Singh, 2024).

4.7.2 DIIQ Syntactic

Before a human respondent attempts this Questionnaire, adequate skill, knowledge, and access to all questionnaire-related material must be confirmed through the Personal Questionnaire. The following essential questions are raised in the Syntactic Interoperability Questionnaire for a respondent:

1. Is the data format of both datasets the same?

   Note: According to the syntactic interoperability requirement matrix presented in Table 3.6 for two applications, if the answer is ***yes****, we do not need any operation as the syntactic interoperability of the data format is given and the Questionnaire is completed. If the answer is* ***no****, DIIM's Transform step is required. The transform step will convert the data into a KG and then convert it to the required format. By repeating the question, the answer will be* ***yes****.*
2. Is the data format valid in which the source data is represented?
3. Are the measurement values the same as in the source data?

Obtaining a **yes** to all questions confirms full syntactic interoperability.

4.7.3 DIIQ Semantic

To build a Semantic Questionnaire, it is essential to consider the semantic interoperability requirement matrix presented in Table 3.7 for two applications. If semantic interoperability is required, DIIM and DIIF must be applied before attempting the Semantic Questionnaire. Additionally, the selected set of terms (based on a given use case) used to label the measurement values must be confirmed when semantic interoperability is required. For each term t in a selected set of terms in $App1$, a term t' in $App2$ must exist and be aligned. Additionally, the units of the measurement values must also be aligned, and the measurement values must be converted to the aligned unit of the other application. Therefore, the following two questions must be repeated for each term in the selected set.

1. Is there an aligned term t in the other application?
2. Is there a measurement unit aligned to the considered term t in the other application?

By answering **yes** to all questions, the full validation of semantic interoperability is confirmed.

4.8 Realisation of DIIM Steps in DIIF

Figure 4.11 illustrates how DIIM's three steps are implemented in DIIF. Each DIIM step comprises a set of sub-stages. DIIM steps are executed in the order: *Transformation*, *Alignment and Storage*, and *Validation*. Blue lines denote existing stages, and red lines denote transitions between stages. Black-filled circles with a single red circle represent a start. Black-filled circles with double red circles represent the end.

4.8.1 DIIM Step 1: Transformation

In DIIM's Transformation step, IoT data is transformed into a KG by utilising a graph converter. There are many converter tools for converting data from one format to another. In particular, W3C maintains a list of tools that can convert data from various formats into RDF graphs (W3C, n.d.). Using RDF to represent KG offers several advantages, e.g. datasets can be automatically merged by combining their triples or quads, annotations can be added to the KG to provide

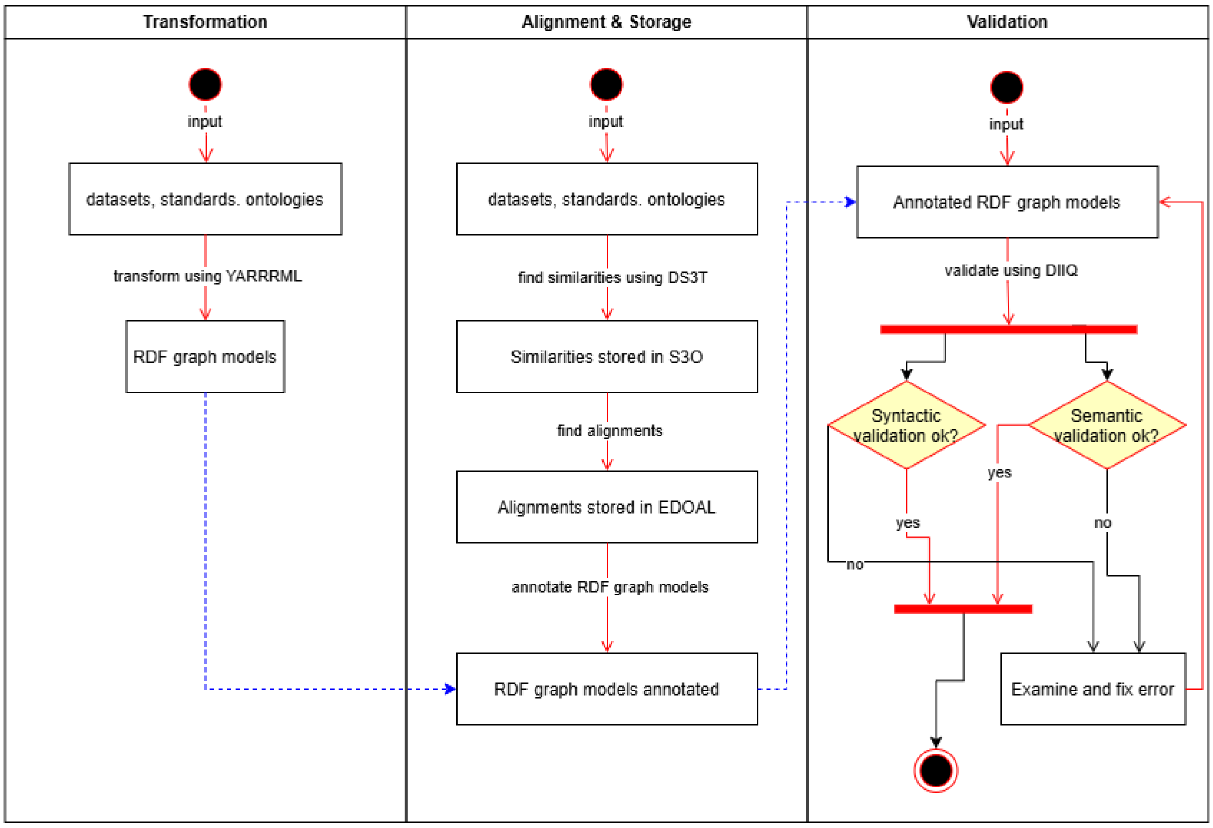

Transformation
Alignment & Storage
Validation
input
datasets, standards. ontologies
transform using YARRRML
RDF graph models
input
datasets, standards. ontologies
find similarities using DS3T
Similarities stored in S3O
find alignments
Alignments stored in EDOAL
annotate RDF graph models
RDF graph models annotated
input
Annotated RDF graph models
validate using DIIQ
Syntactic validation ok?
Semantic validation ok?
yes
no
yes
no
Examine and fix error

Figure 4.11: Implementation of DIIM steps in DIIF

additional context, and common RDF vocabularies can be utilised for semantic description to improve integration. Once different KGs are merged, RDF allows for expressive queries over the unified data.

There are also multiple ways to convert data, including scripts, declarative mapping languages, and languages that perform query translation rather than data translation (e.g. R2RML, a language for expressing customised mappings from relational databases to RDF datasets). Regardless of the approach, data conversion involves mapping the source data into a set of RDF statements. As data is converted and serialised into RDF statements via mappings, these statements can be serialised into various formats. Therefore, YARRRML (Heyvaert et al., 2018) defines mappings that transform semi-structured data into RDF KGs.

4.8.2 DIIM Step 2: Alignment and Storage

To realise DIIM's Alignment and Storage step, DIIF utilises three key tools, DS3T, S3O and EDOAL (Euzenat, David, et al., n.d.). DS3T processes the inputs by applying NLP and ML NN algorithms, such as word to vector and String-search Matching (SSM), to calculate similarities between terms used in input IoT data, standards, and ontologies. DS3T stores the algorithm-based semantic similarity scores for all terms in S3O. Once alignment is confirmed for the given use case, selected alignments are stored in an EDOAL KG. In the final step, IoT KG is annotated with the references to alignments in the EDOAL KG. This interlinking creates integrated KGs that not only contain IoT data but also their semantic relationships to domain-specific standards and ontologies.

4.8.3 DIIM Step 3: Validation

DIIF utilises DIIQ to realise the DIIM's Validation step. As validation is a use case-specific task, it is essential to determine the syntactic and semantic interoperability requirements relevant to the use case based on Tables 3.6 and 3.7. If both syntactic and semantic validation are confirmed, the endpoint of the DIIM methodology is reached. Otherwise, all relevant KGs are examined, the error is fixed, and the validation step is repeated.

4.9 Summary

Despite the existing work discussed in Chapter 2, there is a lack of tools to support the interoperability of IoT data at runtime. This chapter presented a holistic, interoperable framework and dynamic web platform, DIIF, built on the concepts of DIRM and DIIM to address the challenges listed in Section 2.11. DIIF provides a web interface for an IoT-enabled SWN application to discover, subscribe to, and publish data from not only live IoT devices but also legacy systems via URIs. It utilises SW and MDA approaches to enable and enhance the interoperability of IoT data. DIIF uses RDF to represent IoT data as KGs and converts semi-structured IoT data from various formats into KGs using RDF converters. While applying the LD concept, the

KGs are treated as an integrated body of knowledge. DIIF's ontologies DIIO and S3O extend knowledge in the domains of interoperability and ontology matching. DIIO demonstrates how interoperability-relevant knowledge from the SW, IoT, and water domains can be expressed and accessed via an ontology, while minimising coding and the involvement of domain experts. S3O showed how semantic similarity scores calculated by various algorithms can be managed in a KG to support different semantic similarity scenarios.

DS3T is very useful for aligning IoT data with existing ontologies and data models, which are essential for supporting and promoting semantic interoperability across the heterogeneous IoT landscape. DS3T does not support any specific algorithm for computing the semantic similarity score of terms. Still, it allows any algorithm to be implemented and (re)computes similarity scores for the same terms.

DIIF also provides DIIQ to validate syntactic and semantic interoperability between two applications. This component further differentiates it from other interoperability frameworks.

Although topics such as security, scalability, and backup are essential for a web platform, they were kept outside the research scope, assuming that when a DIIF-based application is deployed on a cloud platform, these topics will be addressed before operational activity. The Graphical User Interface (GUI) was out of scope in the DIIF architecture design because a DIIF-based application can be built headless, run in the background, or controlled by other programs. Therefore, a GUI that serves as both the application's front- and back-ends was not critical and could be developed separately later. While the front-end GUI is responsible for the user interface, the back-end GUI facilitates the management of content data and processes. Both can communicate with DIIF through its web interface and present the content to the user.

Chapter 5

Validation Showcases

The previous chapter outlined the DIIM-based Data Information Interoperability Framework (DIIF), which utilises Service-Oriented Architecture (SOA) and Semantic Web (SW) technologies to ensure the syntactic and semantic interoperability of IoT data. This chapter presents the third contribution, comprising validation studies that verify the effectiveness of DIIM and DIIF in achieving Objective 5.

5.1 Overview

This section provides an overview of the three validation showcases where DIIM and DIIF were applied and validated. These showcases illustrate how DIIF was utilised in an existing IoT-enabled SWN system, as well as how it enabled interoperability between two IoT-enabled applications in the water and 5G/6G domains. Each of these showcases has a published paper in its respective domain.

The first showcase, **IoT-enabled SWN monitoring and management**, is from the water domain. It validates DIIF's ability to enable interoperability between two distinct water-quality monitoring systems. These systems differ in that they have installed sensors from other manufacturers, which use different serialisation formats and do not share a common semantic model. Despite these differences, the application of DIIM enables syntactic and semantic interoperability between these applications (M. Singh, W. Wu, et al., 2022).

The second showcase, **Alignment of IoT data for SWNs**, is also from the water domain. It validates that DIIF supports the alignment of sensor data with water and IoT domain ontologies, thereby increasing semantic interoperability between them. This showcase presents the novel S3O and DS3T to accommodate various semantic similarity calculation algorithms, which are required for aligning in different situations (M. Singh, Vakaj, et al., 2023).

The third showcase, **QoE prediction on IoT 5G/6G datasets**, is from the telecommunications domain, where systems utilise several mechanisms to collect data from 5G/6G-enabled IoT devices. Therefore, 5G/6G data must be interoperable with monitoring, prediction, and decision-support systems. However, 5G/6G data are typically mapped to local data models for local applications, which poses challenges for their use in cross-domain or heterogeneous applications due to interoperability limitations. This validation study evaluates the ability of DIIM and DIIF to support and enhance the interoperability of 5G/6G data through NLP and SW technologies, thereby achieving 5G/6G data harmonisation (M. Singh, Mahmoud, et al., 2024).

5.2 Showcase IoT-enabled SWN monitoring and management

To validate the effectiveness of DIIM and DIIF, M. Singh, W. Wu, et al. (2022) published a reference implementation for the proposed work described in Chapters 3 and 4. This validation study evaluates a representative scenario for IoT data characterising a SWN application.

5.2.1 A Case study on IoT-enabled SWN monitoring and management

Figure 5.1 illustrates a potential interoperability scenario for smart sensing in the water domain, where a network of IoT (it is about connecting physical devices, e.g. smart sensors, to networks) or Web of Things (WoT) (it is about using Web standards to access, describe, and uniformly integrate those devices) is constructed through the interconnection of several smart devices to support the pervasive and ubiquitous functionality of a Water Quality Monitoring System (WQMS). On the one hand, we recognise cloud-based water-quality monitoring of the reservoir using S_1, S_2, and S_3 sensors for temperature, pH, and dissolved oxygen, respectively. On the

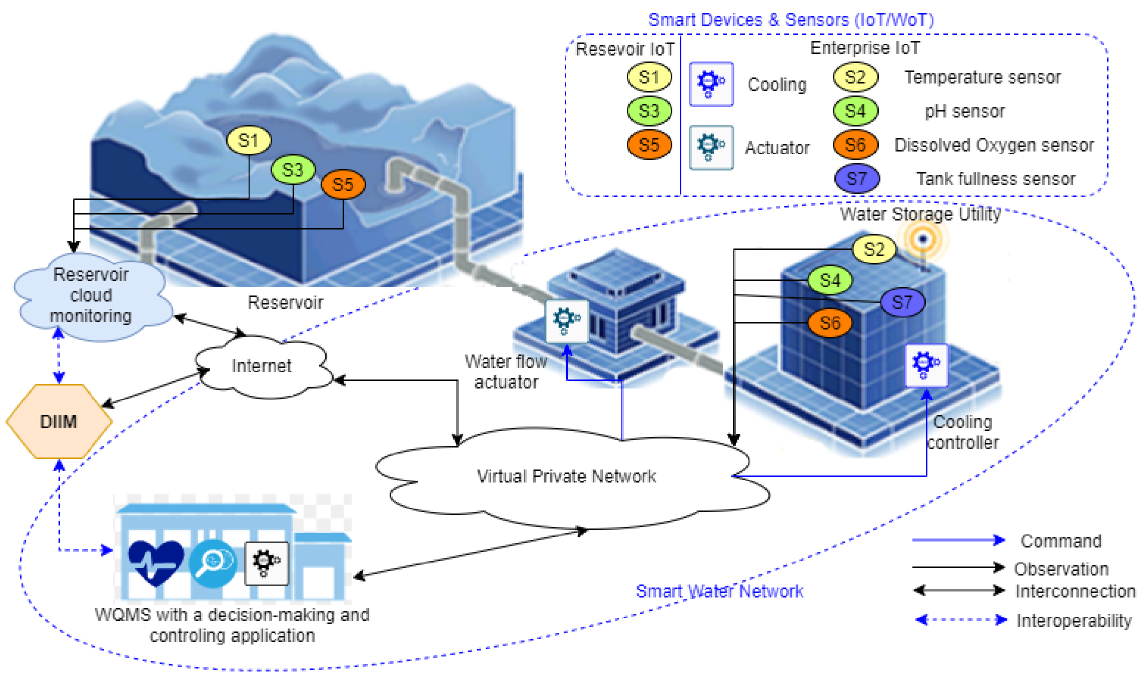


Figure 5.1: Interoperability motivation scenario in IoT-enabled SWN monitoring and management

other hand, we consider an enterprise that uses lake water as a source of drinking water to produce bottled water. The enterprise has established a SWN by interconnecting its IoT/WoT and WQMS within a Virtual Private Network (VPN) to monitor and manage the quality of stored water. The smart sensors (S_2, S_4, S_6, and S_7) monitor temperature, pH, dissolved oxygen, and tank fullness, and send the data to the enterprise WQMS for centralised remote monitoring and management. The decision-making application of WQMS bases decisions on enterprise logic (a formalised set of rules, constraints, and decision-making principles that govern how an enterprise makes decisions and coordinates its processes, data, and systems). It issues commands for the smart devices (cooling controller and actuator) to keep the stored lake water under ideal conditions, e.g. water temperature in a range of 2-15 °C, pH value in 6-9 logarithmic units, dissolved oxygen concentration in 80-120 mg/L, and tank fullness in 700000 and 900000 m^3. Table 5.1 lists the rules computed by the decision-making system when sensors send the observation data to the WQMS. Consequently, the controller dispatches operational commands to operate the cooling and actuator utilities.

Table 5.1: Enterprise logic rules to manage the stored water

Property	**Rule**	**Utility**	**Command**
Temperature	if $S_2 value > 15$	water cooling	on
Temperature	if $S_2 value <= 2$	water cooling	off
Water volume	if $S_7 value <= 700000$	actuator	open
Water volume	if $S_7 value >= 900000$	actuator	close

If the water temperature exceeds 15 °C, the cooling system at the water storage utility must be used to cool the water. This operation will increase electricity consumption and production costs. Alternatively, as shown in Table 5.2, the stored water can be cooled by opening the actuator if the reservoir water temperature is below 15 °C and there is still storage capacity in the water tank. However, the decision-making application of WQMS must understand and evaluate the reservoir sensor data to make real-time decisions. Therefore, the syntactic and semantic interoperability (Noura et al., 2019) of the data collected by the IoT must be enabled regardless of the data serialisation format and lexical label name of the observed data.

Table 5.2: Extended logic rule to manage water temperature

Property	**Rule**	**Utility**	**Command**

Temperature	if $S_2 value > 15$ AND
	if $S_1 value < 15$ AND
	if $S_7 value < 80000$
		actuator	open

Table 5.3 lists the seven modelled sensors as an indicative scenario of interoperability between the enterprise WQMS and reservoir cloud monitoring. The S_1 and S_3 sensors are from a Chinese manufacturer. They serialise data in JSON, using the terms *temp* and *ph* to label the observed data. The S_2 and S_5 sensors are from an American manufacturer and use the CSV format to serialise data. They label the monitored data with the terms *Temperature* and *DO*. The S_4, S_6, and S_7 sensors are from a European manufacturer, and they use the XML data format to serialise the data and label the observed data with terms *pH* and *DissolvedOxygen*. The enterprise's WQMS decision-making system follows the Semantic Web approach. Therefore, it expects the data to be well-defined in RDF/XML format and uses terms defined by the Australian Government Linked Data Working Group for marine water-quality observations within a water-quality ontology (Australian Government Linked Data Working Group, 2016). The EWQM ontology (re)uses the concepts (terms) from the SKOS data model and units of measurement

Table 5.3: Representation of the modelled sensors

Sensor	**Sensor observation property**	**Serialization format**	**Label name of observed property**	**Unit of observed property**
(IoT)	***(Data context)***	***(Data syntax)***	***(Data semantic)***	
S_1	Temperature	JSON	temp	°F
S_2	Temperature	RDF/XML	Temperature	°C
S_3	pH	JSON	ph	
S_4	pH	RDF/XML	pH	logarithmic units
S_5	Dissolved Oxygen	JSON	DO	ppm
S_6	Dissolved Oxygen	RDF/XML	DissolvedOxygen	mg/L
S_7	Water volume	RDF/XML	WaterVolume	m^3

from the Quantities, Units, Dimensions, Data Types (QUDT) ontology. In contrast to enterprise applications, none of the IoT applications for reservoirs refer to any ontology or data-model standard. However, their data is publicly accessible from the cloud in JSON and CSV formats. This situation poses challenges of syntactic and semantic heterogeneity that the enterprise must address if it wants to utilise the data from reservoir sensors in its WQMS.

5.2.2 Application of Data Information Interoperability Model and Methodology (DIIM)

The following section describes the DIIM application procedure, enabling interoperability in the previously outlined case study of a water-bottling enterprise and a water reservoir. Table 5.4 displays the indicative setup for the given case study's DIIM.

Table 5.4: DIIM's indicative parameter setup

Parameter	**Input**
application domain	water quality monitoring
IoT data subscriber	enterprise WQMS
IoT data publisher	reservoir cloud monitoring application
domain-specific ontologies	SAREF, Geo, Time, QUDT, GeoRSS
application-specific ontologies	DIIM, IoT data ontology, EWQM
subscriber/publisher profiles	WQMS-profile, reservoir cloud-profile
domain-specific standard	WaterML2
IoT collect data	data collected from publisher
Parameter	**Output**
IoT push data	syntactically and semantically harmonized IoT data for subscriber
IoT ontology	IoT ontology with collected data and annotations as references to the terms of other ontologies

Based on the setup, DIIM will execute the following operational activities:

1. **Property subscription**: DIIM's operational activity begins when the enterprise WQMS subscribes to the water quality property (Temperature in °C) at a particular location, in RDF/XML format, at a specified endpoint. DIIM creates a subscriber profile of the subscription requested by the WQMS.
2. **TDD lookup**: DIIM uses its DIIM ontology for water-quality descriptions to semantically match the TDD for the subscribed water-quality property in the IoT cloud. The lookup services keep searching until an IoT is found. Then, DIIM creates a publisher profile of the matched IoT and links it to the subscriber profile that matches the semantic search.
3. **Data collection**: The data collector starts collecting the meta and measurement data based on the publisher profile. An exemplary input to DIIM, consisting of IoT data collected in JSON format is shown below. The fetched data is stored in the collected IoT database.

### DIIM Input: a snippet of IoT data in JSON format ###

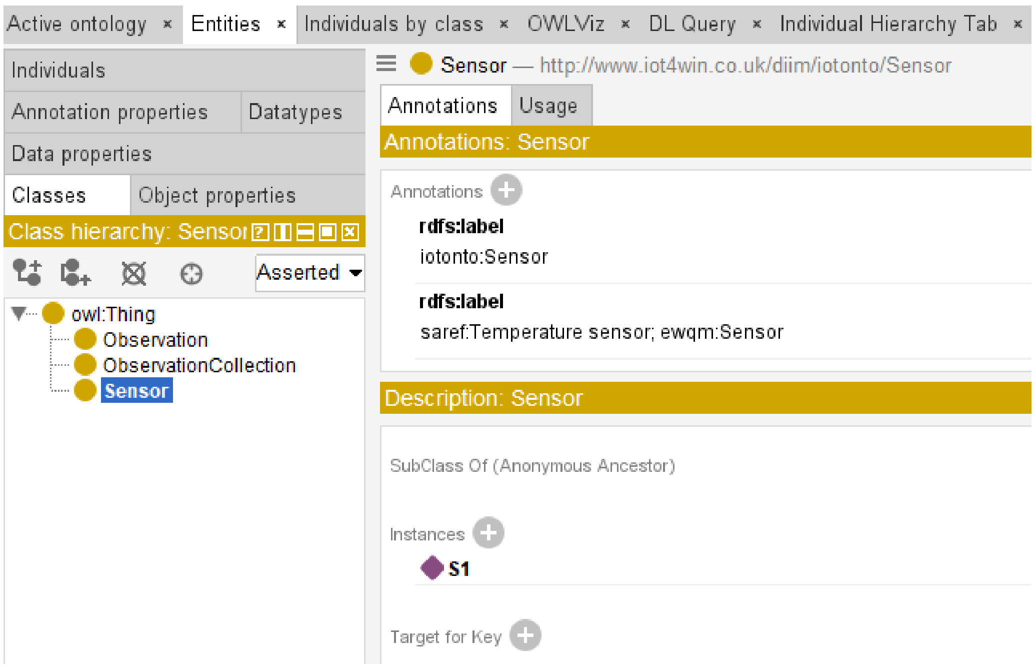
Meta data: "Name":"S1","Description":"The sensor measures water temperature in Fahrenheit", "serial":"00-14-22-01-23-45", "model":"BFG9000", "mac":"50:8c:b1:77:e8:e6", "latitude":51.75543,"longitude":-1.03248

Measurement data: [{"4baa-a2ff-8741efad4e63": {"temp":[
{"timestamp":"2021-08-09T17:01:28.796Z","values":{"value":20}},
{"timestamp":"2021-08-09T17:01:38.792Z","values":{"value":24}},
...]}}]

4. **Data analysis**: The collected data is analysed, and flags are set in the next step. Since the profiles of WQMS (subscriber) and reservoir (publisher) do not match, DIIM sets flags for serialisation message content format RDF/XML, property name (term) harmonisation, and property value conversion.
5. **Semantic harmonisation**: Since a flag for the property name and value conversion is set, the collected data objects are relabelled and converted according to the subscriber profile. As shown in Table 5.5, DIIM will map the terms of the reservoir and WQMS to DIIM, WQMS ontology, and domain-specific ontologies. DIIM will create an IoT ontology for the reservoir and annotate its terms with terms from other ontologies. DIIM converts the temperature value from Fahrenheit to Celsius and stores it in the IoT ontology.

Table 5.5: Ontological alignment of terms among reservoir IoT, DIIM KB, and enterprise WQMS

	Terms of		
Reservoir	**DIIM Ontologies**	**WQMS**	**EWQM Ontology**
S_1	saref:Temperature sensor	S_2	ewqm:Sensor
temp	saref:Temperature	Temperature	qudt:water_temperature
f	saref:Temperature unit	C	qudt:unit
water	saref:Water	Water	ewqm:object
latitude	geo:latitude	Location	georss:point
longitude	geo:longitude	Location	georss:point
timestamp	time:dateTimeStamp	dateTime	ewqm:dateTime
values,value	saref:value	value	ewqm:value
Name	rdfs:Literal	name	ewqm:name
Description	rdfs:Comment	description	ewqm:description



Sensor — http://www.iot4win.co.uk/diim/iotonto/Sensor

Annotations Usage

Annotations: Sensor

Annotations

rdfs:label
iotonto:Sensor

rdfs:label
saref:Temperature sensor; ewqm:Sensor

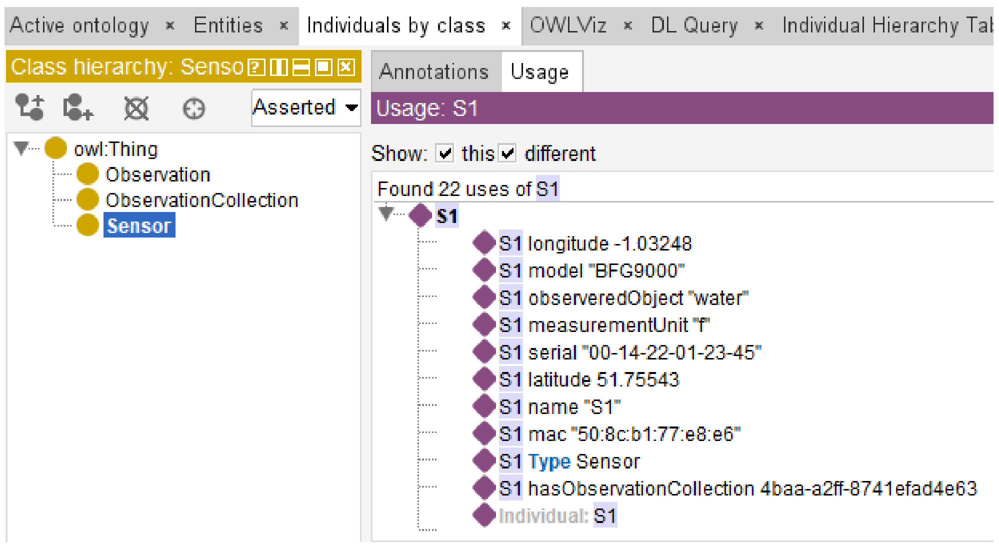
Description: Sensor

SubClass Of (Anonymous Ancestor)

Instances
S1

Target for Key

Figure 5.2: Sensor with annotated names from SAREF and EWQM ontologies



Usage: S1

owl:Thing
Observation
ObservationCollection
Sensor

Show: ✓ this ✓ different

Found 22 uses of S1

```
S1
    S1 longitude -1.03248
    S1 model "BFG9000"
    S1 observeredObject "water"
    S1 measurementUnit "f"
    S1 serial "00-14-22-01-23-45"
    S1 latitude 51.75543
    S1 name "S1"
    S1 mac "50:8c:b1:77:e8:e6"
    S1 Type Sensor
    S1 hasObservationCollection 4baa-a2ff-8741efad4e63
    Individual: S1
```

Figure 5.3: Sensor instance details

By applying DIIM (see Chapter 3), an IoT ontology is generated and populated with collected data, and annotated with references to terms from other domain-specific ontologies, as shown in Figures 5.2 and 5.3. An RDF version of the IoT ontology is as follows:

```
<!—- DIIM Output: IoT ontology with examples of data and annotations —->
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:XMLS="http://www.w3.org/2001/XMLSchema#"
xmlns:iotonto="http://www.iot4win.co.uk/diim/iotonto/"
xmlns:saref="http://uri.etsi.org/m2m/saref#"
xmlns:ewqm="http://waterenterprise.com/wqm/ewqm#"
xmlns:georss="http://www.georss.org/georss/"
xmlns:qudt="http://qudt.org/2.1/schema/qudt">
<owl:Class rdf:about="iotonto:#Observation"/>
<owl:Class rdf:about="iotonto:#Sensor"/>
```

```
<owl:ObjectProperty rdf:about="iotonto:#hasObservation">
<rdfs:subPropertyOf rdf:resource="owl:#topObjectProperty"/>
<rdfs:domain rdf:resource="iotonto:#Sensor"/>
<rdfs:range rdf:resource="iotonto:#Observation"/>
</owl:ObjectProperty>
```

```
<iotonto:Sensor rdfs:label="saref:Temperature sensor; ewqm:Sensor">
<iotonto:name rdf:datatype="XMLS:string">S1</iotonto:name>
<iotonto:description rdf:datatype="XMLSchema:string">The sensor measures water temperature
in Fahrenheit</iotonto:description>
<iotonto:latitude rdf:datatype="XMLS:float">51.75543</iotonto:latitude>
<iotonto:longitude rdf:datatype="XMLS:float">-1.03248</iotonto:longitude>
<iotonto:mac rdf:datatype="XMLS:string">50:8c:b1:77:e8:e6</iotonto:mac>
<iotonto:serial rdf:datatype="XMLS:string">00-14-22-01-23-45</iotonto:serial>
<iotonto:model rdf:datatype="XMLS:string">BFG9000</iotonto:model>
<iotonto:measurementUnit rdf:datatype="XMLS:string">f</iotonto:measurementUnit>
<iotonto:observeredObject rdf:datatype="XMLS:string">water</iotonto:observeredObject>
</iotonto:Sensor>
```

```
<iotonto:ObservationCollection>
<iotonto:id rdf:datatype="XMLS:string">4baa-a2ff-8741efad4e63</iotonto:id>
<iotonto:property rdf:datatype="XMLS:string">temp</iotonto:property>
<iotonto:hasObservation rdf:parseType="Collection">
<iotonto:Observation> <iotonto:timestamp
rdf:datatype="XMLS:string">2021-08-09T17:01:28.796Z</iotonto:timestamp>
<iotonto:values> <rdf:Seq> <rdf:li>20</rdf:li> </rdf:Seq> </iotonto:values>
</iotonto:Observation>
</iotonto:hasObservation> </iotonto:ObservationCollection>
```

Table 5.6: Interoperability enablement and enhancement of the reservoir's monitoring data

DIIM applied	Format	Convert to Formats	Ontology	Adapt Ontologies
no	JSON	none	none	none
yes	RDF	RDF/XML[6] RDF/OWL, binary RDF, RDF/JSON, JSON-LD, N-Quads, N-Triples, N3, ND JSON-LD, TriG-Star, TriG, TriX, Turtle, Turtle-star, GraphDB, etc.[7]	reservoir ontology	WQMS ontology[6] SAREF, Geospatial ontologies, Time ontology, etc.[7]

```
<owl:Axiom>
<owl:annotatedSource rdf:resource="iotonto:#4baa-a2ff-8741efad4e63"/>
<owl:annotatedProperty rdf:resource="iotonto:#property"/>
<owl:annotatedTarget rdf:datatype="XMLS:string">temp</owl:annotatedTarget>
<rdfs:label rdf:datatype="XMLS:string">qudt:water_temperature</rdfs:label>
<rdfs:label rdf:datatype="XMLS:string">saref:Temperature</rdfs:label>
</owl:Axiom>
</rdf:RDF>
```

6. **Syntactic harmonisation**: The WaterML2 time-series standard is first adopted for the push IoT data. Then an OWL translator serialises the data in RDF/XML.
7. **Data push**: After validation, the Data distributor pushes syntactically and semantically correct and harmonised publisher data in the WaterML2 time-series standard, using the EWQM ontology's semantics, to the subscriber endpoint.

5.2.3 Validation of interoperability enablement and enhancement

In this showcase, an enterprise's WQMS wants to use the reservoir's monitoring data. However, as shown in Table 5.3, the WQMS's data interface requires data in RDF/XML, whereas the reservoir's monitoring data is in JSON. All enablement and enhancement cases are discussed as follows:

[6]interoperability enabled
[7]interoperability enhanced

1. **Syntactic enablement**: After the DIIM application, the reservoir's monitoring data is transformed into RDF format. The reservoir's monitoring data can now be directly converted to RDF/XML, as the RDF data format standard supports both RDF/XML. Therefore, the syntactic enablement of the reservoir's monitoring data for the enterprise's WQMS is achieved. Thus, the syntactic interoperability requirement is validated.
2. **Semantic enablement**: After the DIIM application, the reservoir's data model is aligned with the semantic model of WQMS. The reservoir's monitoring data can now be aligned with the WQMS's ontology, as terms used to label measurements can be interchanged. Therefore, the semantic enablement of the reservoir's monitoring data for the enterprise's WQMS is achieved. Thus, the semantic interoperability requirement is validated.
3. **Syntactic enhancement**: After DIIM is applied, the reservoir's monitoring data is transformed into RDF format. As shown in Table 5.6, RDF data can be directly converted to formats supported by RDF. Various RDF converters are also available for converting data formats. This validates improved syntactic interoperability of the reservoir's data.
4. **Semantic enhancement**: After application of DIIM, the reservoir's data model is not only aligned with the semantic model of WQMS but also with other ontologies. As shown in Table 5.6, the reservoir's data can also adapt to different ontologies. This validates enhanced semantic interoperability.

5.2.4 Showcase summary

In summary, DIIM's steps to enable interoperability in the showcase are:

(i) DIIM takes subscription parameters from enterprise WQMS and IoT data from the reservoir cloud as inputs.

(ii) DIIM analyses the collected data and transforms it into an OWL/RDF-expressed semantic model.

(iii) Ontology alignment tools, e.g. OntoAligner (Giglou et al., 2025), GraphMatcher (Efeoglu, 2024), BERTMap (He et al., 2022), and DS3T (M. Singh, Vakaj, et al., 2023), align the newly generated semantic model with domain-specific ontologies and the ontology of the enterprise WQMS. DIIM records all found matches in the semantic model as annotations.

(iv) Finally, the OWL/RDF translator adheres to the WaterML2 standard for time-series data and transforms the semantic model into the required format for the enterprise WQMS. Then DIIM uses the terms from the enterprise WQMS's ontology to relabel the data. Since DIIM has also aligned the semantic model of the reservoir IoT data with domain-specific ontologies, the reservoir IoT data becomes interoperable with all applications that support these ontologies.

This validation confirms that the proposed method addresses the syntactic and semantic interoperability challenges of IoT-enabled SWNs. *DIIM's syntactic interoperability approach* addresses serialisation format issues during the parsing of the IoT data by data-consuming applications by applying data format translation. Additionally, DIIM adopts domain-specific standards, e.g. WaterML2, to represent water-related data in time series before delivering it to the consumer application. *DIIM's semantic interoperability approach* harmonises the semantic models of an IoT system and a data-consuming application by aligning their ontologies. Suppose an IoT application neither uses an existing ontology nor builds an ontology for its data. In that case, DIIM creates a semantic model in OWL from available IoT data, aligns its terms with domain-specific ontologies, and (re)annotates the IoT semantic model. With this method, DIIM enables interoperability between IoT and a SWN application and further enhances interoperability by adopting domain-specific ontologies and standards. Because, after alignment of IoT's semantic model to domain-specific ontologies, the IoT data becomes interoperable for all those applications that use these domain-specific ontologies. In the motivation scenario, DIIM acts as a mediator in the water domain, enabling data-based interoperability between an IoT platform and a SWN application. However, any other discipline can use this approach to enable interoperability, where the utilisation of IoT data is beneficial.

5.3 Alignment of IoT Data with Domain-specific Ontologies

As stated in Section 2.11, data collected by IoT in the water domain are available in CSV format. Since the CSV format does not directly support annotation or linking mechanisms, it cannot be linked directly to existing ontologies or standards in the water domain. Adapting an existing ontology is a cumbersome, manual task due to its semantic complexity, interdependencies, limited automation, and the need for expert judgment. Even minor adaptations require careful analysis to preserve meaning and consistency, making the process time-consuming and labour-intensive. Furthermore, many applications use different ontologies and standards in the water domain. Therefore, the data requires presentation in various formats and simultaneous adaptation to other ontologies and standards. This validation study examines how DIIF supports alignment of IoT data with domain-specific ontologies through Semantic Web and NLP technologies. The research presented in this showcase was published in the SEMANTICS 2023 EU conference (M. Singh, Vakaj, et al., 2023).

5.3.1 Implementation of DIIF

The code for implementing the DIIF was developed in Visual Studio Code (Microsoft, 2015). For the implementation of the approach, Python (Foundation, 2001) and many Python-based NLP libraries, e.g. Gensim (Řehůřek, 2009) for Word2Vector (Word2Vec) and Latent Semantic Index (LSI), import Matplotlib (Matplotlib Development Team, 2003) for visualisation, Pandas (NumFOCUS Inc., 2020) for data storage and retrieval, and RDFLib (RDFLib Team, 2009) for processing S3O, were utilised. Protégé (Noy, Crubézy, et al., 2003), an ontology development environment tool, is used to author and examine S3O ontology facts on term similarity, written in the Turtle RDF serialisation format. Pellet (Sirin et al., 2007) reasoner is used to reason the S3O. The Snap SPARQL Protocol and RDF Query Language (SPARQL) Query plugin (Horridge et al., 2016) for Protégé is used to query the S3O facts.

The showcase application takes two inputs: IoT data and domain-specific ontologies for the water and IoT domains. In particular, the datasets considered are related to water quality. The

Table 5.7: Input: IoT data

Dataset Name	Format	License	Topic	Summary
Bristol River water quality	CSV	Open Government Licence	Water Quality	River quality monitoring data (chemical, physical and bacteriological parameters tested) from 1994. The laboratory analysis & reporting can take up to a couple of months to be available (Bristol City Council, 2010).
Kaa IoT Data	JSON	Private	Water temperature	The data holds the values of a temperature sensor that was simulated locally to send data to the KAA IoT Cloud platform (M. Singh, 2021).

showcase application processed IoT data serialised in CSV or JSON formats. More information on the characteristics of the IoT dataset is available in Table 5.7. Table 5.8 holds the information on the ontologies used as input. Input ontologies are from the upper, water, biological, or water domain. The showcase application processed these ontologies from serialisation formats, e.g. text, XML, HTML, RDF, or OWL.

S3O schema was developed in Protégé and exported as an RDF file. RDFlib (RDFLib Team, 2009) was used to generate a graph by loading the S3O schema and populating it with facts and information computed by the showcase program, including terms from IoT and ontologies, their relations, and algorithm-based similarity scores.

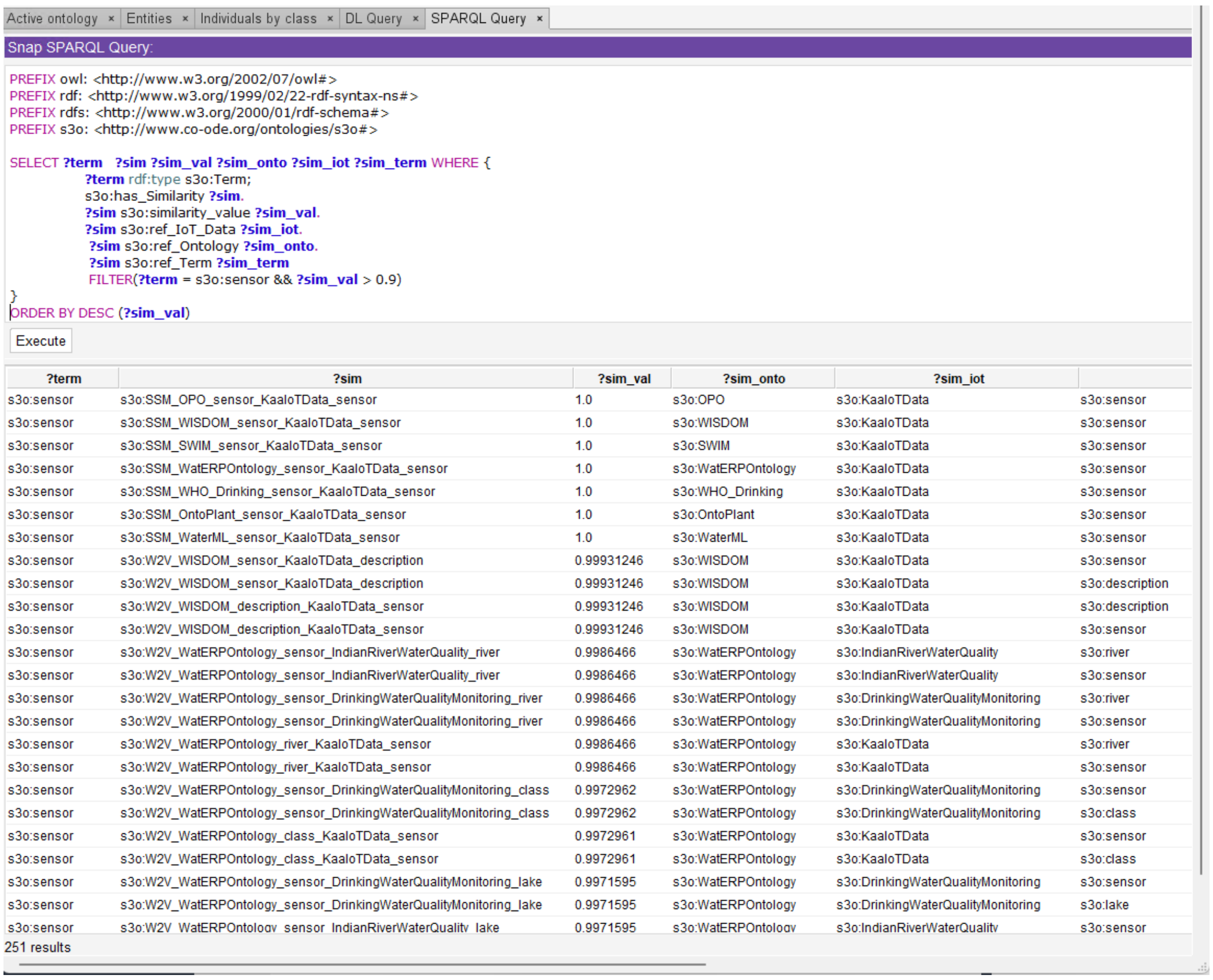
DS3T uses descriptive-naming-pattern $< algorithmname > \_ < ontology - name > \_ < ontologyterm - name > \_ < IoT - name > \_ < IoT - term - name >$ for the similarity class instances. For example, in the $SSM\_SensorML\_counts\_BristolWaterQuality\_units$ instance, the SSM algorithm is used to calculate the similarity between $counts$ from $SensorML$ ontology and $units$ from $BristolWaterQuality$ data.

As primary output, the showcase application generates the S3O in RDF turtle format. Figure 5.4 displays the results of a SPARQL query to S3O. The query searches for terms similar to "sensor" within domain-specific standards and ontologies provided as input to DS3T.

Figure 5.5 visualises a part of the generated S3O. It illustrates various relationships among entities in S3O. In particular, the $W3v\_Term\_Similarity$ class has an instance named as $SensorML\_counts\_BristolWaterQuality\_units$ that contains a Word2Vec-based similarity score between the terms $counts$ and $units$. Term $counts$ belongs to the $SensorML$ ontology and $units$ the $BristolWaterQuality$ dataset.

Table 5.8: Input: domain-specific ontologies and standards

Name	**Domain**	**Summary**
COSMO	upper	COSMO (Cassidy, 2020) is a foundation ontology that allows it to represent all the basic ('primitive') ontology elements of an application.
DOLCE	upper	DOLCE (Guarino and Gangemi, 2017) is a linguistic and cognitive engineering descriptive ontology.
GOIoTP	IoT	GOIoTP (Szmeja, 2018) is developed as part of the INTER-IoT project; it offers modular data structures for the description of entities most commonly appearing in IoT in the context of interoperating various IoT artefacts (platforms, devices, services, etc).
INSPIRE	upper	INSPIRE (INSPIRE, 2015) represents a set of concepts within a domain and the relationships between those concepts.
OntoPlant	water	Sottara et al. (2014) have extended the SSN ontology to decouple control logic from equipment choices in wastewater treatment plants.
SAREF	IoT	SAREF (ETSI Technical Committee, 2020) is a shared consensus model that facilitates the matching of existing assets in the smart applications domain.
SensorML	IoT	Sensor Model Language (SensorML) (OGC, 2019) provides a robust and semantically-tied means of defining processes and processing components associated with the measurement and post-measurement transformation of observations.
SSN	IoT	SSN (W. OGC, 2017) ontology describes sensors and sensor networks for use in web applications, independent of any application domain.
SOSA	IoT	SOSA (W3C, 2016) can be used directly for lightweight applications or provide the basis for additional specialisation and axiomatisation in vertical and horizontal extensions.
SWIM	water, IoT	SWIM (Reynolds, 2013) is developed by Aquamatix for the Device-level IoT semantic model for the water industry.
WaterML	water	WaterML2 (OGC, HDWG, 2014) is a new data exchange standard in Hydrology to exchange many hydro-meteorological observations and measurements. It harmonises several exchange formats for water data with relevant OGC and ISO standards.
WatERP Ontology	water	It is developed by EURECAT-WatERP (Anzaldi, W. Wu, et al., 2014). It is a lightweight ontology of generic concepts for water sensing and management.
WHO Drinking standard	water	WHO standard guidelines (World Health Organization (WHO), 2021) to maintain the relevance, quality, and integrity of the Guidelines for drinking-water quality (GDWQ) whilst ensuring their continuing development in response to new or newly appreciated information and challenges.
WISDOM	water	WISDOM (Howell, Rezgui, and Thomas Beach, 2017) developed by Cardiff University for the cyber-physical and social ontology of the water value chain.



```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX s3o: <http://www.co-ode.org/ontologies/s3o#>

SELECT ?term   ?sim ?sim_val ?sim_onto ?sim_iot ?sim_term WHERE {
           ?term rdf:type s3o:Term;
           s3o:has_Similarity ?sim.
           ?sim s3o:similarity_value ?sim_val.
           ?sim s3o:ref_IoT_Data ?sim_iot.
            ?sim s3o:ref_Ontology ?sim_onto.
            ?sim s3o:ref_Term ?sim_term
            FILTER(?term = s3o:sensor && ?sim_val > 0.9)
}
ORDER BY DESC (?sim_val)
```

?term	?sim	?sim_val	?sim_onto	?sim_iot	
s3o:sensor	s3o:SSM_OPO_sensor_KaaloTData_sensor	1.0	s3o:OPO	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WISDOM_sensor_KaaloTData_sensor	1.0	s3o:WISDOM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_SWIM_sensor_KaaloTData_sensor	1.0	s3o:SWIM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WatERPOntology_sensor_KaaloTData_sensor	1.0	s3o:WatERPOntology	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WHO_Drinking_sensor_KaaloTData_sensor	1.0	s3o:WHO_Drinking	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_OntoPlant_sensor_KaaloTData_sensor	1.0	s3o:OntoPlant	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WaterML_sensor_KaaloTData_sensor	1.0	s3o:WaterML	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WISDOM_sensor_KaaloTData_description	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WISDOM_sensor_KaaloTData_description	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:description
s3o:sensor	s3o:W2V_WISDOM_description_KaaloTData_sensor	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:description
s3o:sensor	s3o:W2V_WISDOM_description_KaaloTData_sensor	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_IndianRiverWaterQuality_river	0.9986466	s3o:WatERPOntology	s3o:IndianRiverWaterQuality	s3o:river
s3o:sensor	s3o:W2V_WatERPOntology_sensor_IndianRiverWaterQuality_river	0.9986466	s3o:WatERPOntology	s3o:IndianRiverWaterQuality	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_river	0.9986466	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:river
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_river	0.9986466	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_river_KaaloTData_sensor	0.9986466	s3o:WatERPOntology	s3o:KaaloTData	s3o:river
s3o:sensor	s3o:W2V_WatERPOntology_river_KaaloTData_sensor	0.9986466	s3o:WatERPOntology	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_class	0.9972962	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_class	0.9972962	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:class
s3o:sensor	s3o:W2V_WatERPOntology_class_KaaloTData_sensor	0.9972961	s3o:WatERPOntology	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_class_KaaloTData_sensor	0.9972961	s3o:WatERPOntology	s3o:KaaloTData	s3o:class
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_lake	0.9971595	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_lake	0.9971595	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:lake
s3o:sensor	s3o:W2V_WatERPOntology_sensor_IndianRiverWaterQuality_lake	0.9971595	s3o:WatERPOntology	s3o:IndianRiverWaterQuality	s3o:sensor

251 results

Figure 5.4: Querying term "sensor" in S3O with SPARQL

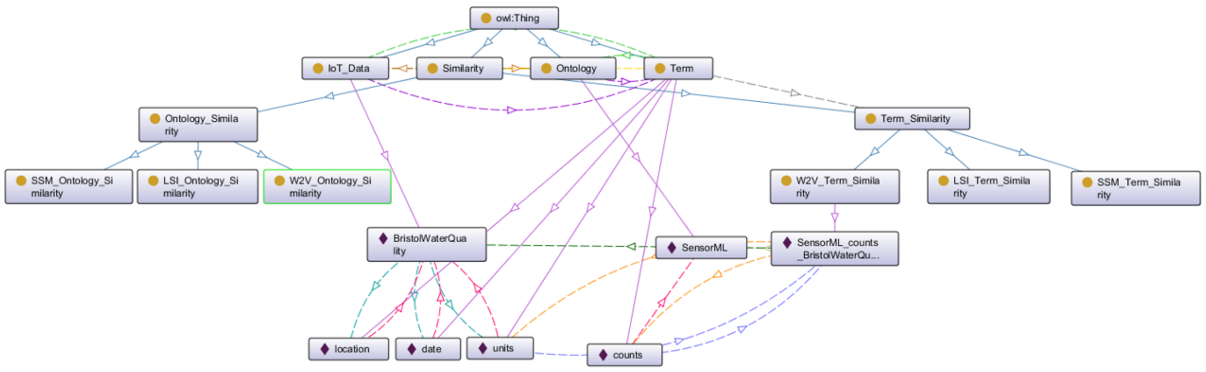


Figure 5.5: Querying term "sensor" in S3O with SPARQL

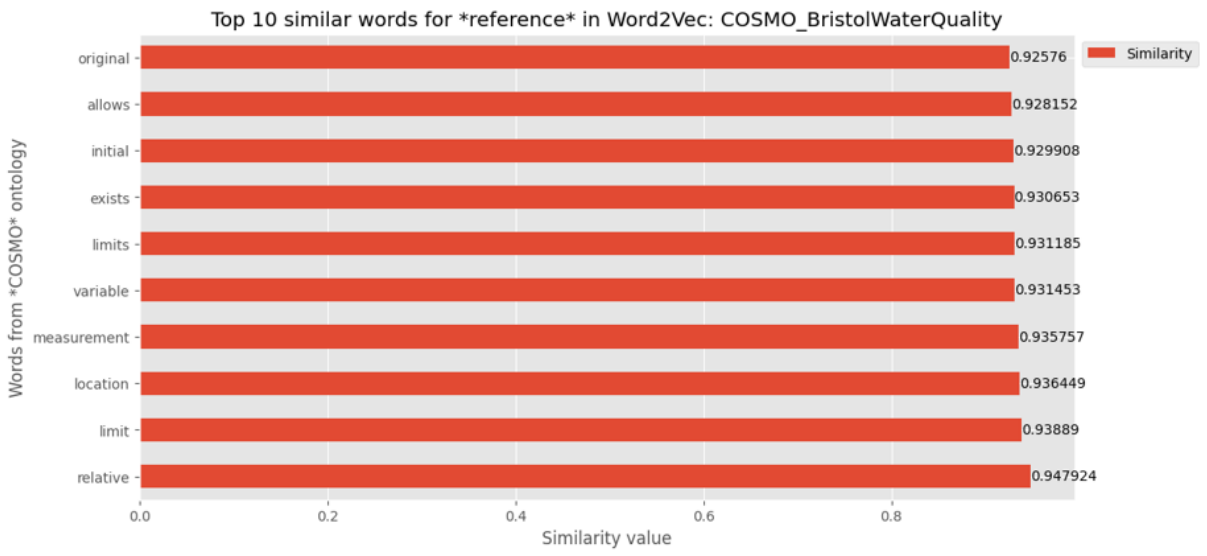


Figure 5.6: Word-to-vector based top 10 similar words for "reference" in COSMO

Other outputs of the showcase application are pickle files and bar charts. Pickle files persist the calculated information for the compared terms, the applied algorithms, and their semantic similarity scores. Persistent files are particularly advantageous for efficient use of computing resources. Bar charts provide, for each IoT term, a visualisation of the top 10 terms with the highest similarity scores, as determined by the Word2Vec algorithm. Bar charts help humans to identify similar terms quickly. Figure 5.6 provides validation evidence that the computed similarity rankings support alignment. It displays the top 10 similar terms to "reference" in the Bristol water quality dataset. The Word2Vec algorithm has identified 10 similar terms in the upper ontology COSMO, thereby aligning the Bristol water quality dataset with COSMO.

5.3.2 Showcase summary

This validation study evaluates the proposed *novel methodology based on Semantic Web technologies (OWL, KG, RDF, and LD) and NLP (LSI, Word2Vec, and N-Gram similarity) to discover related ontologies and align IoT data terms with these ontologies*. Furthermore, this work contributes to the development of a new ontology, the Semantic Similarity Scoring Ontology (S3O). The proposed S3O stores term-similarity scores computed by evaluating the applied algorithms. This ontology can be easily extended to include the evaluation results of other algorithms. This way, it does not promote any specific algorithm for aligning terms. Instead, it can incorporate all alignment algorithms that may become relevant at some point. Therefore, it stores the similarity scores of all alignment algorithms in S3O, and an ontology engineer can query these scores, explore the linked terms across different ontologies, and decide on the final alignment/mapping. This validation confirms the effectiveness of the approach in an IoT-enabled smart water application. However, the proposed solution is extensible to incorporate new ontologies for alignment and to consider newly developed term alignment algorithms.

5.4 IoT-enabled 5G/6G Datasets for QoE Prediction

IoT-enabled 5G/6G application scenarios are those scenarios in which devices and applications use a 5G/6G infrastructure to communicate and share information. Figure 5.7 shows some of these scenarios: (i) a live camera stream viewed on a tablet with Augmented Reality (AR), (ii) viewing a media stream in a Virtual Reality (VR) headset, or (iii) monitoring and analysing data sent by IoT for prediction and decision-making. Data collected in 5G/6G application scenarios often serves a specific application purpose, leading to heterogeneous data models when these applications do not use domain-specific ontologies or standards. For example, data collected by the application in the first scenario, viewing camera stream with augmented reality, cannot be used directly in the third scenario by the data analysis and decision-making application if these applications use different data models. Hence, diverse 5G/6G data must be harmonised with the end application to ensure interoperability before other applications use them. This poses a significant challenge to interoperability when integrating 5G/6G data into another pre-established system. To achieve data model interoperability, the syntactic (structural) and semantic (meaningful) interoperability of the shared data/information must be ensured.

In this context, the harmonising tasks are cumbersome and challenging for an application engineer during the integration of 5G/6G applications, as they require a manual review of the relevant application data models and domain-specific ontologies or standards to align with 5G/6G data. Additionally, before aligning each term used in the 5G/6G data models with the

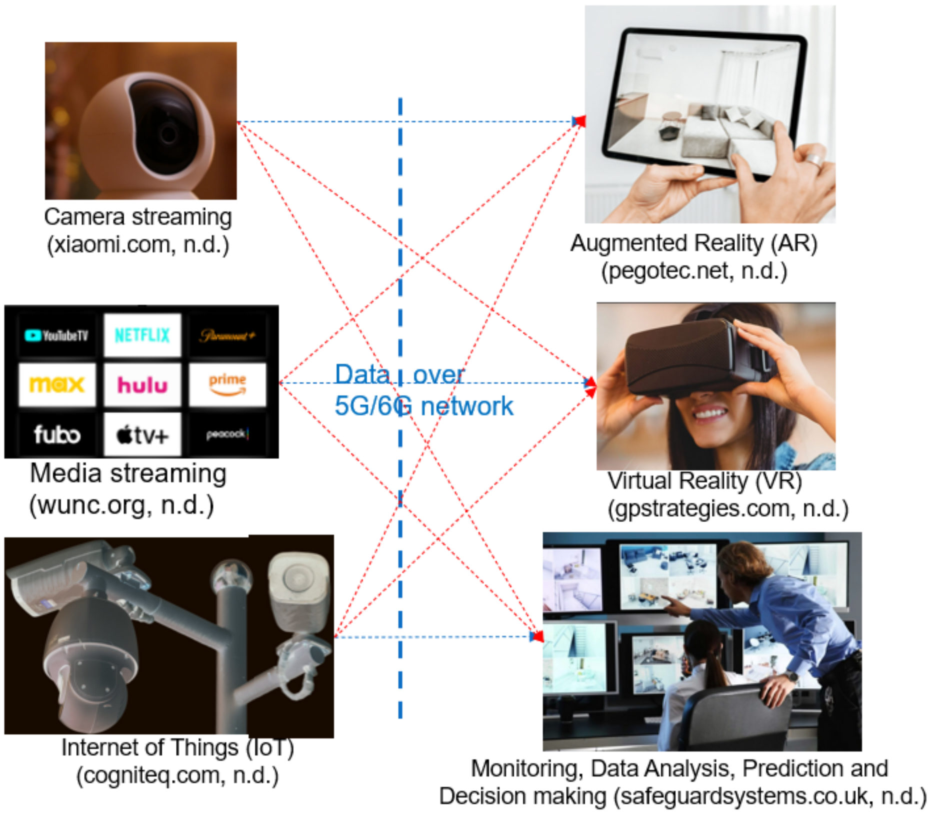


Figure 5.7: IoT-enabled 5G/6G application scenarios

concepts (terms) defined in the domain-specific ontologies or standards, all related terms in the given ontologies must be considered according to their similarity or relatedness.

This validation study examines QoE in 5G/6G applications from different angles. We first define objective and subjective metrics for measuring QoE in 5G/6G-enabled applications. Next, we motivate the data interoperability problem in the context of QoE data alignment and present an approach to address it, along with preliminary results. Finally, as part of the case study, ML algorithms were utilised to develop a QoE prediction model. Our results validate the potential to achieve interoperability among 5G/6G datasets for QoE prediction applications through semantic technologies and ML. We presented the work done in this showcase in an IEEE ATOMS conference (M. Singh, Mahmoud, et al., 2024).

5.4.1 A Case Study on QoE prediction across IoT-enabled 5G/6G datasets

QoE prediction models are particularly used to help improve content delivery services by informing resource allocation decisions that enhance the user experience. Multiple features can be exploited in video delivery services to predict user satisfaction with the videos. This includes information about the video content, network parameters, and display specifications. Combining QoE data from different sources can be beneficial in various contexts, including QoE prediction. However, other video services may adopt different terminologies in their QoE frameworks. This section validates the feasibility of applying QoE prediction across two different QoE datasets. We use the $LIVE\_NFLX$ dataset (Bampis et al., n.d.) for fitting QoE models and apply the models for predicting the subjective QoE scores in Aleksandrlvchenko's QoE-Assessment dataset (Ivchenko, n.d.). We use two features to predict the final MOS. Namely, the mean Peak Signal-to-Noise Ratio (PSNR) and the number of stalling events. We selected these two metrics because they are strongly related to users' perceptions of video content. Intuitively, higher PSNR values and fewer stalling events should improve QoE. The two datasets use different labels for these features. To harmonise the data from the two datasets, we manually map the terms from both datasets that correspond to these features. We also match and normalise the subjective scores from each dataset to their respective scoring ranges, enabling a unified data scale.

Three regression methods were used in experiments: Linear Regression (LR), Kernel Ridge Regression (KRR), and Support Vector Regression (SVR). We first fit each of the three models on the $LIVE\_NFLX$ dataset and evaluate their performance on a test set of the same dataset. We then utilise the same models, fitted using only the $LIVE\_NFLX$ dataset, to predict the QoE scores derived from $Aleksandrlvchenko$'s dataset. Figure 5.8 illustrates the original (true) data points and the fitted prediction models. Normalised MOS are shown for both datasets, obtained from pairs of stalling events and PSNR values. In Figure 5.8, the MOS scores for the test set of the $LIVE\_NFLX$ dataset are shown, along with the predictions from each model. The unseen data points of $Aleksandrlvchenko$'s dataset are depicted in Figure 5.8, along with the MOS values from the trained models. The resulting Mean Absolute Error (MAE) and the Mean Squared Error (MSE) values are summarised in Table 5.9. The results show that QoE prediction

Table 5.9: QoE prediction error over the two datasets

MODEL	LIVE_NFLX		AleksandrIvchenko	
	MAE	MSE	MAE	MSE
LR	0.1142	0.0214	0.1138	0.0211
KRR	0.1115	0.0213	0.1240	0.0243
SVR	0.1213	0.0239	0.1099	0.0186

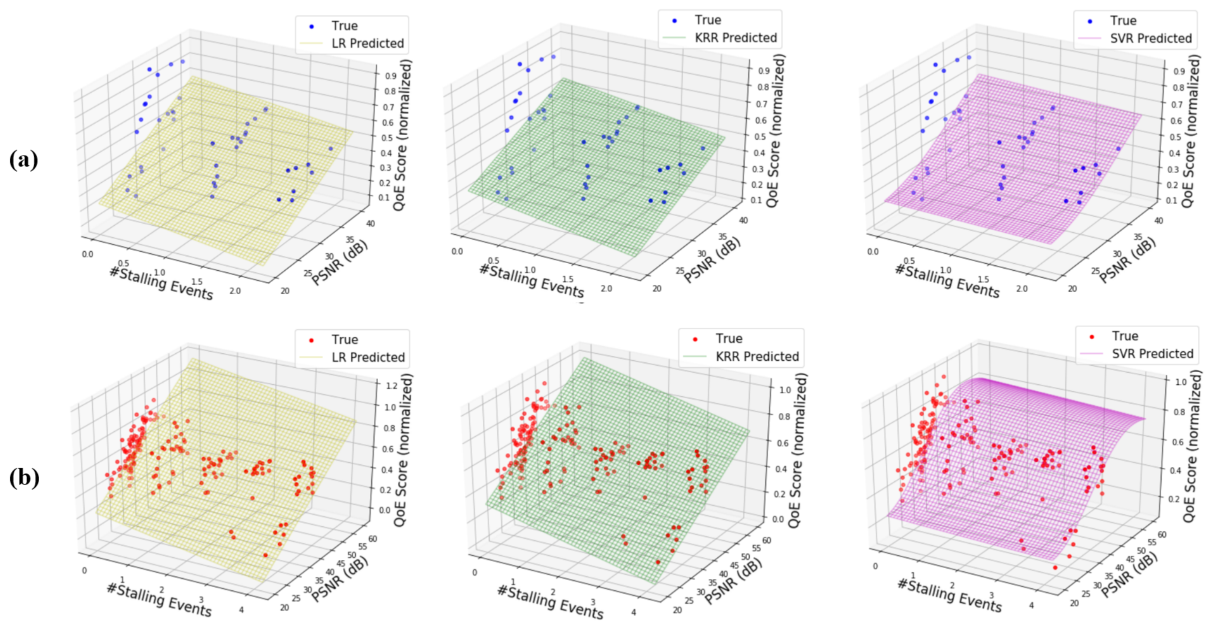


Figure 5.8: QoE prediction results obtained from three different prediction models. (a) predicted MOS scores over the *LIVE_NFLX* dataset. (b) predicted MOS scores over *AleksandrIvchenko* dataset

models can be utilised across different datasets. All three models yielded reasonable predictions on the unseen dataset. This held even though *Aleksandrlvchenko*'s dataset contains data points with wider ranges for both PSNR and stalling events. This highlights the ability of the QoE models to extrapolate over broader input ranges.

Requirement: QoE data harmonisation

5G/6G refers to the next generation of wireless communication technology. One of the hot research areas for 6G technology is edge AI, which involves processing data at or near the source rather than transmitting it back to a centralised data centre for analysis. This could significantly improve areas such as autonomous vehicles, healthcare, and manufacturing. However, this requires harmonising 5G/6G data across different applications and data formats, or labels, to

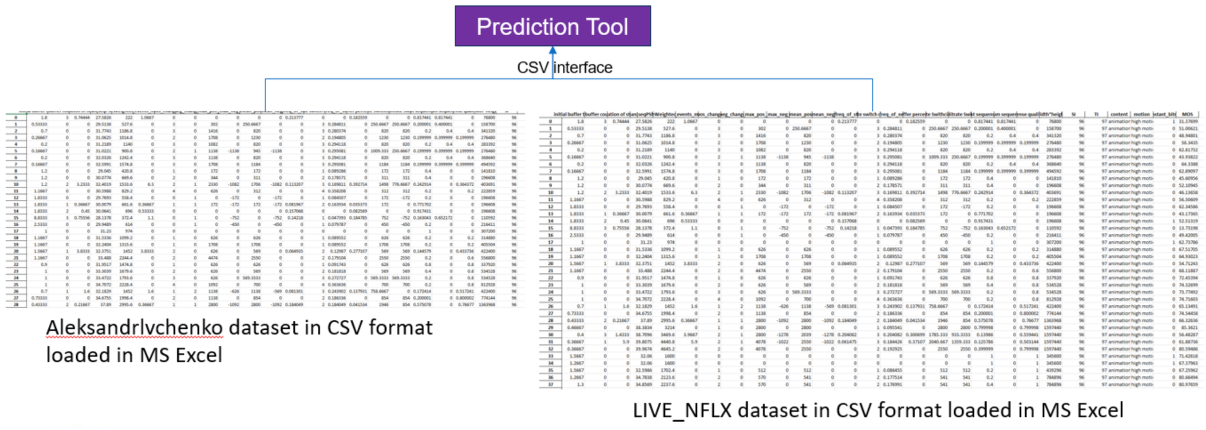

Prediction Tool
CSV interface
Aleksandrlvchenko dataset in CSV format loaded in MS Excel
LIVE_NFLX dataset in CSV format loaded in MS Excel

Figure 5.9: Data interface of the prediction tool

record measurement data. In the data integration process, the following two harmonisation scenarios arise:

1. **Data-model to data-model harmonisation**: When two applications with different data models want to use each other's data, their data models must be harmonised so they can treat their datasets as a single source. The case study presented in the previous section is based on this scenario.
2. **Data-model to standard/ontology harmonisation**: An application's data-model must align with domain-specific standards and ontologies so that other domain-specific applications can use its datasets as a single source. This scenario enhances the interoperability of a data model.

In both scenarios, we must overcome syntactic (differences in data representation formats) and semantic (differences in data model terminology for the same data) interoperability issues in 5G/6G data. Therefore, the given datasets, standards, or ontologies must be manually inspected before they can be harmonised into a single data source. Harmonising data can become a cumbersome task when done manually. However, NLP and SW technologies can support data harmonisation.

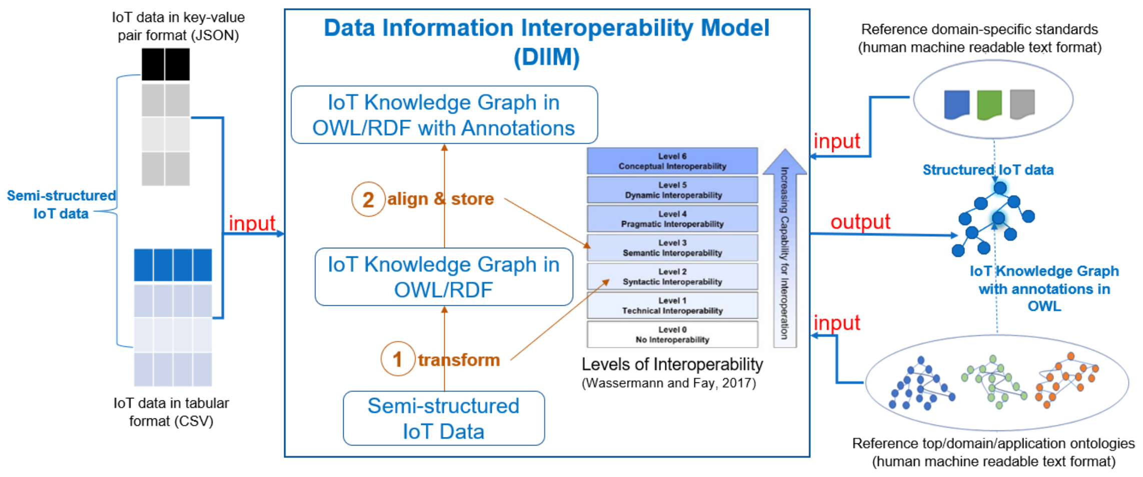

IoT data in key-value pair format (JSON)
Semi-structured IoT data
IoT data in tabular format (CSV)
input
Data Information Interoperability Model (DIIM)
IoT Knowledge Graph in OWL/RDF with Annotations
2 align & store
IoT Knowledge Graph in OWL/RDF
1 transform
Semi-structured IoT Data
Level 6 Conceptual Interoperability
Level 5 Dynamic Interoperability
Level 4 Pragmatic Interoperability
Level 3 Semantic Interoperability
Level 2 Syntactic Interoperability
Level 1 Technical Interoperability
Level 0 No Interoperability
Increasing Capability for Interoperation
Levels of Interoperability
(Wassermann and Fay, 2017)
input
output
input
Reference domain-specific standards (human machine readable text format)
Structured IoT data
IoT Knowledge Graph with annotations in OWL
Reference top/domain/application ontologies (human machine readable text format)

Figure 5.10: Data Information Interoperability Model (DIIM) (M. Singh, W. Wu, et al., 2022)

Table 5.10: Input datasets and standards

Name	Source format	Type	Reference
LIVE_NFLX	CSV	dataset	(Bampis et al., n.d.)
AleksandrIvchenko	CSV	dataset	(Ivchenko, n.d.)
DashReStreamer	CSV	dataset	(Hodzic, Cosovic, et al., n.d.)
IEEEIQA	PDF converted to Text (TXT)	standard	(Wang et al., 2004)
QoEAnalysis	PDF converted to TXT	standard	(Hodzic and al., 2022)

5.4.2 Implementation of DIIF

M. Singh, W. Wu, et al. (2022) introduce a DIIM to enable and enhance the interoperability of the IoT data for SWN applications. To achieve syntactic interoperability in Figure 5.10, the given data is converted to RDF using an RDF converter (W3C, n.d.). It can then be transformed into an application-specific format. To achieve semantic interoperability between two applications, similar terms in their data models are linked via annotations in the RDF graphs that represent their data. To address the semantic issue, we follow the DIIM that takes datasets and domain-specific ontologies or standards as input and generates an annotated KG as output. DS3T (M. Singh, Vakaj, et al., 2023) compares the similarity between terms used in datasets and ontologies/standards. DS3T stores the found similarity results in S3O (M. Singh, Vakaj, et al., 2023), with references to the respective terms and the originating documents (datasets or standards/ontologies) that use them.

We need to harmonise the datasets before feeding them to the prediction tool as a single source that uses the data model known to the prediction tool's interface. To enable harmonisation (interoperability) of input datasets shown in 5.10, the requirements are as follows:

1. **Syntactic requirement**: As shown in Figure 5.9, the prediction tool can only read CSV datasets. Therefore, all input datasets must be represented in CSV format.
2. **Semantic requirement**: In the case study, we start with the AleksandrIvchenko dataset and consider terms *seqpsnr* and *stalling*. Therefore, we must identify terms in other datasets and standards that are similar to or contain measurement values to align with the terms of AleksandrIvchenko's dataset.

5.4.3 DIIM Steps

In this case study, we do not have any operational activity for DIIF's web services because the prediction application has no web interface, and datasets are manually loaded into the application's interface. Therefore, we start directly with the DIIM steps.

Transformation

According to the case study requirements, we do not need to harmonise the $Aleksandrlvchenko$ and $LIVENetflix$ datasets because their source format is CSV, and the prediction tool can process the input data in CSV format. However, we still perform DIIM's transform step to prepare them for its align-and-store step by converting the CSV dataset using the RDF Mapping Language (RML) to an RDF graph model. In Figure 5.11, the transform output is shown as a KG because we add some contextual information, such as the dataset's name, source, and domain, to the generated graph. Furthermore, transforming datasets into RDF graph models will enable interoperability with scenarios that require data formats other than CSV.

Alignment and Storage

For DIIM's *Align and Store* step, we set up with the DS3T, which takes two types of input as listed in Table 5.10, datasets ($Aleksandrlvchenko$, $DashReStreamer$, $LIVENetflix$)

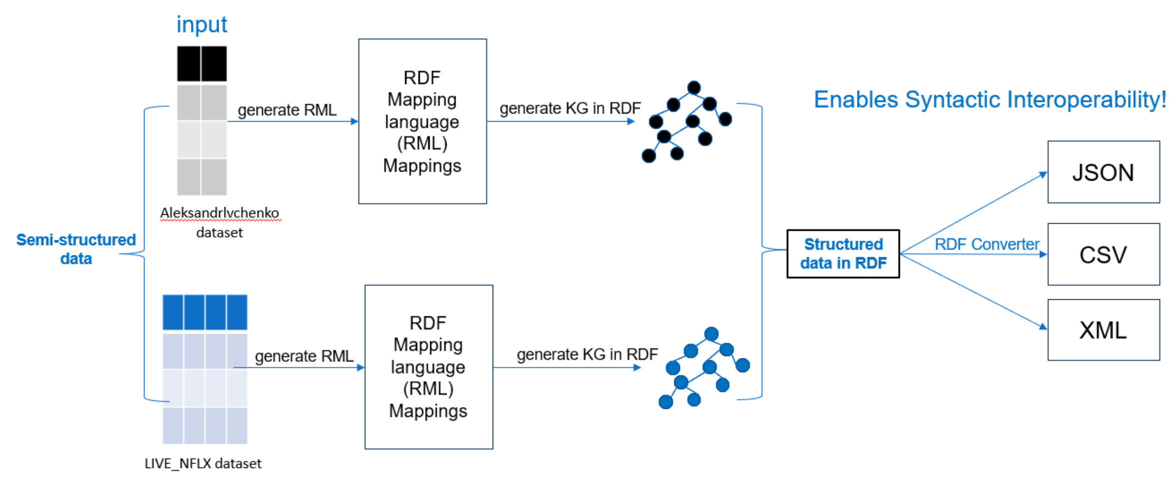


Figure 5.11: Transform step in the showcase

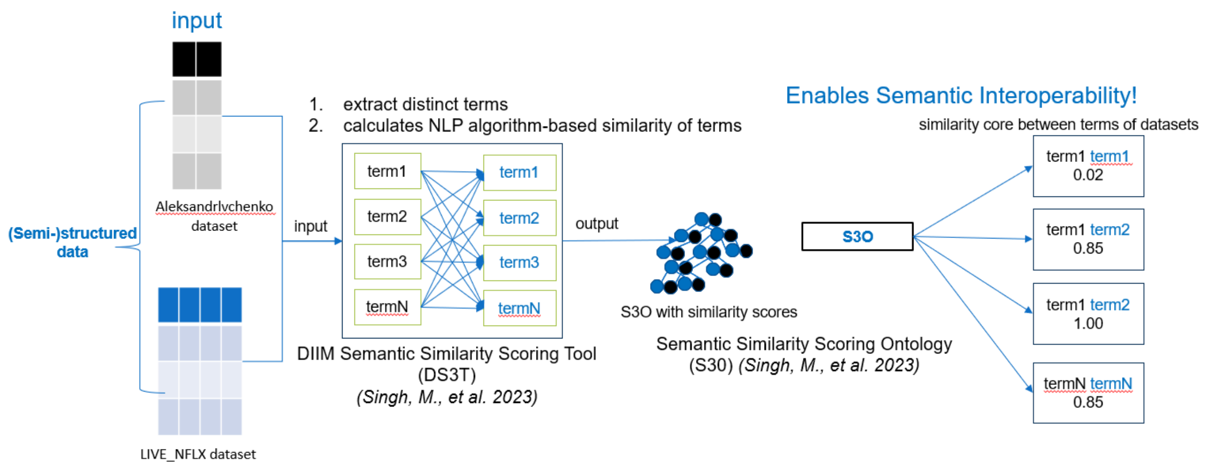

input
1. extract distinct terms
2. calculates NLP algorithm-based similarity of terms
Enables Semantic Interoperability!
similarity core between terms of datasets
(Semi-)structured data
Aleksandrlvchenko dataset
input
term1
term2
term3
termN
term1
term2
term3
termN
output
S3O
term1 term1 0.02
term1 term2 0.85
term1 term2 1.00
termN termN 0.85
S3O with similarity scores
DIIM Semantic Similarity Scoring Tool (DS3T) (Singh, M., et al. 2023)
Semantic Similarity Scoring Ontology (S30) (Singh, M., et al. 2023)
LIVE_NFLX dataset

Figure 5.12: Algin and Store step in the showcase

and standards ($QoEAnalysis$ and $IEEEIQA$), and generates an S3O with facts on similarity between terms used in the given inputs.

As shown in Figure 5.12, DS3T first extracts distinct terms from all input datasets and standards. Figure 5.13 compares the number of extracted distinct terms from the inputs. Since manually extracting distinct terms from domain-specific standards and datasets is challenging for humans, DS3T is more feasible.

In the second step, DS3T calculates the similarity between two unigram terms using the SSM algorithm implemented in *difflib* (Python Software Foundation, 2001) and stores the results in S3O. To examine the similarity calculation results, we can load the S3O into an ontology editor,

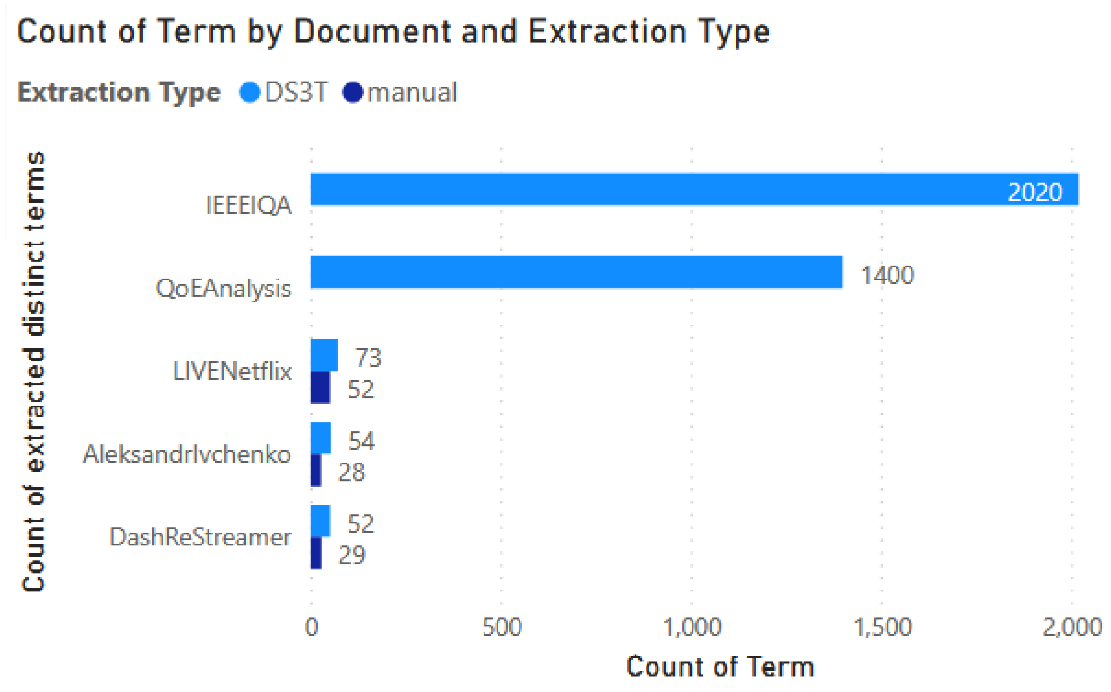


Figure 5.13: Count of distinct terms extracted manually and by DS3T from input datasets and standards

Table 5.11: SSM-based selected alignment of terms among datasets and standards

AleksandrIvchenko	**LIVENetflix**	**DashReStreamer**	**IEEEIQA**	**QoEAnalysis**
seqpsnr →	psnr (0.73)	psnr (0.73)	psnr (0.73)	psnr (0.73)
stalling →	-	stall (0.77)	-	stall (0.77)

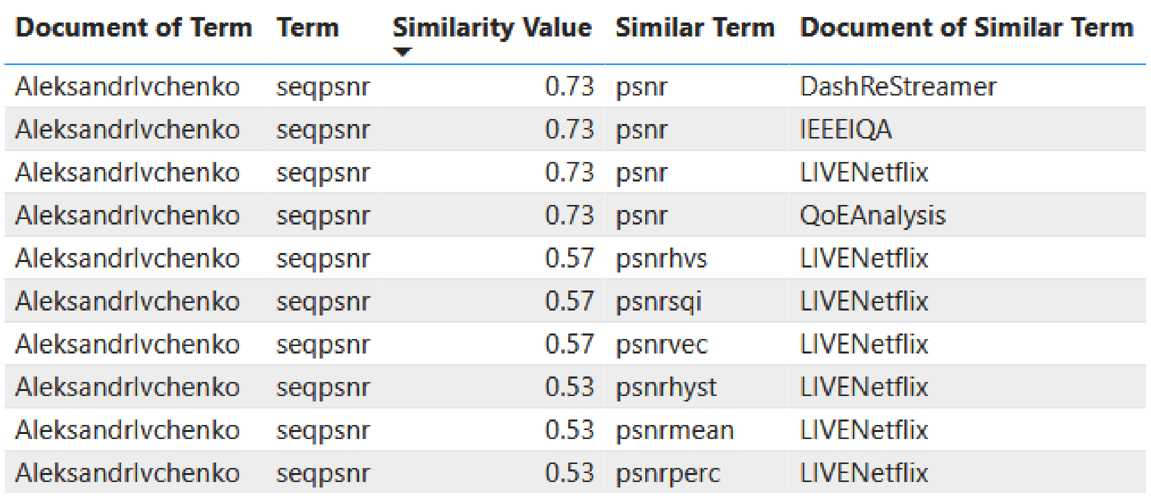
e.g. Protege (Noy, Crubézy, et al., 2003), and query it using SPARQL. DS3T can also export similarity results in CSV format, which can be imported into a more user-friendly tool, such as *Excel* or *PowerBI* (Microsoft, n.d.).

Figure 5.14 shows SSM-based similar terms found in datasets and standards for $seqpsnr$ used in the $Aleksandrlvchenko$ dataset. Figure 5.15 shows SSM-based similar terms found in datasets and standards for *stalling* used in the $Aleksandrlvchenko$ dataset.

After examining the DS3T results in S3O, we select to align the terms *eqpsnr* and *stalling* as shown in Table 5.11. The SSM algorithm does not find any suitable match for *stalling* in the LIVENetflix dataset because *ns* is used for labelling the measurement value.

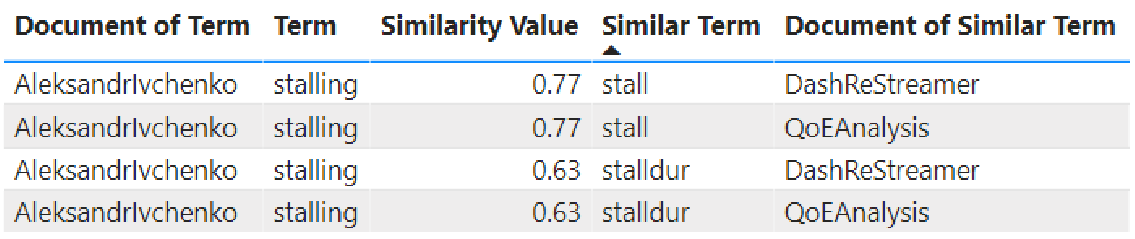
Based on Table 5.11, we present the alignment of terms between Aleksandrlvchenko and the datasets and domain-specific standards as follows in an EDOAL ontology 5.12.

We annotate the term $seqpsnr$ in the $Aleksandrlvchenko$ graph with links to the alignment



Document of Term	Term	Similarity Value	Similar Term	Document of Similar Term
AleksandrIvchenko	seqpsnr	0.73	psnr	DashReStreamer
AleksandrIvchenko	seqpsnr	0.73	psnr	IEEEIQA
AleksandrIvchenko	seqpsnr	0.73	psnr	LIVENetflix
AleksandrIvchenko	seqpsnr	0.73	psnr	QoEAnalysis
AleksandrIvchenko	seqpsnr	0.57	psnrhvs	LIVENetflix
AleksandrIvchenko	seqpsnr	0.57	psnrsqi	LIVENetflix
AleksandrIvchenko	seqpsnr	0.57	psnrvec	LIVENetflix
AleksandrIvchenko	seqpsnr	0.53	psnrhyst	LIVENetflix
AleksandrIvchenko	seqpsnr	0.53	psnrmean	LIVENetflix
AleksandrIvchenko	seqpsnr	0.53	psnrperc	LIVENetflix

Figure 5.14: SSM-based similar terms found by DS3T for term *seqpsnr* in datasets and domain-specific standards

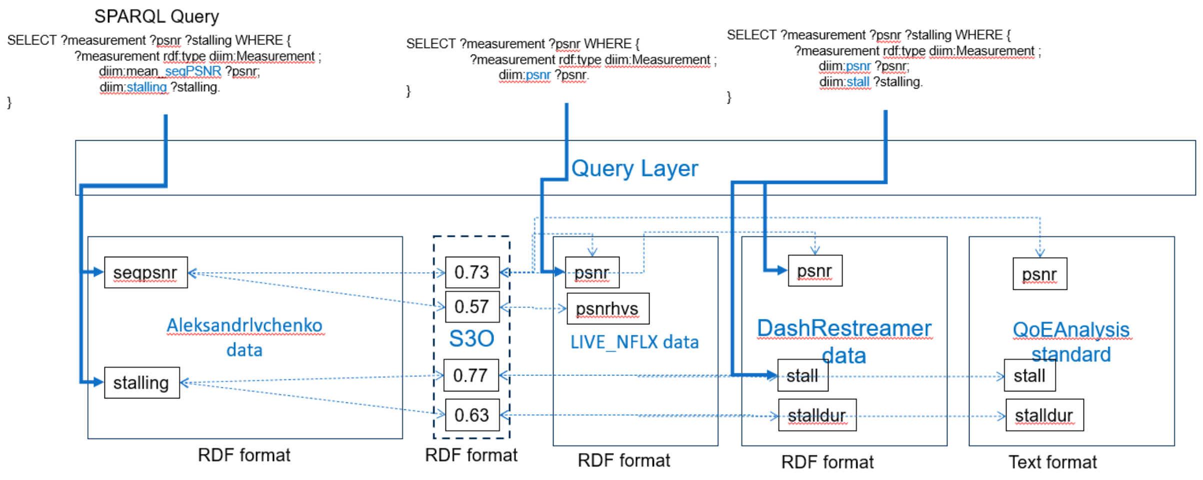


Document of Term	Term	Similarity Value	Similar Term	Document of Similar Term
AleksandrIvchenko	stalling	0.77	stall	DashReStreamer
AleksandrIvchenko	stalling	0.77	stall	QoEAnalysis
AleksandrIvchenko	stalling	0.63	stalldur	DashReStreamer
AleksandrIvchenko	stalling	0.63	stalldur	QoEAnalysis

Figure 5.15: SSM-based similar terms found by DS3T for term *stalling* in datasets and domain-specific standards

Table 5.12: EDOAL-based alignment representation of selected aligned terms between Aleksandrlvchenko and other datasets and domain-specific standards

```
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and LIVENetflix —->
...
<align:map>
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_LIVENetflix_psnr" />
   <align:seqpsnr><edoal:Class rdf:about=".../Aleksandrlvchenko#seqpsnr" /></align:seqpsnr>
   <align:psnr><edoal:Class rdf:about=".../LIVENetflix#psnr" /></align:psnr>
   <align:relation>=</align:relation>
   <align:measure rdf:datatype="&xsd;float">0.73</align:measure>
  </align:Cell>
</align:map>
...
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and DashReStreamer —->
...
<align:map>
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_DashReStreamer_psnr"/>
   <align:seqpsnr><edoal:Class rdf:about=".../Aleksandrlvchenko#seqpsnr" /></align:seqpsnr>
   <align:psnr><edoal:Class rdf:about=".../DashReStreamer#psnr" /></align:psnr>
   <align:relation>=</align:relation>
   <align:measure rdf:datatype="&xsd;float">0.73</align:measure>
  </align:Cell>
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_stalling_DashReStreamer_stall"/>
   <align:seqpsnr><edoal:Class rdf:about=".../Aleksandrlvchenko#stalling" /></align:stalling>
   <align:psnr><edoal:Class rdf:about=".../DashReStreamer#stall" /></align:stall>
   <align:relation>=</align:relation>
   <align:measure rdf:datatype="&xsd;float">0.77</align:measure>
  </align:Cell>
</align:map>
...
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and IEEEIQA —->
...
<align:map>
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_IEEEIQA_psnr"/>
   <align:seqpsnr><edoal:Class rdf:about=".../Aleksandrlvchenko#seqpsnr" /></align:seqpsnr>
   <align:psnr><edoal:Class rdf:about=".../IEEEIQA#psnr" /></align:psnr>
   <align:relation>=</align:relation>
   <align:measure rdf:datatype="&xsd;float">0.73</align:measure>
  </align:Cell>
</align:map>
...
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and QoEAnalysis —->
...
<align:map>
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_QoEAnalysis_psnr"/>
   <align:seqpsnr><edoal:Class rdf:about=".../Aleksandrlvchenko#seqpsnr" /></align:seqpsnr>
   <align:psnr><edoal:Class rdf:about=".../QoEAnalysis#psnr" /></align:psnr>
   <align:relation>=</align:relation>
   <align:measure rdf:datatype="&xsd;float">0.73</align:measure>
  </align:Cell>
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_stalling_QoEAnalysis_stall"/>
   <align:seqpsnr><edoal:Class rdf:about=".../Aleksandrlvchenko#stalling" /></align:stalling>
   <align:psnr><edoal:Class rdf:about=".../QoEAnalysis#stall" /></align:stall>
   <align:relation>=</align:relation>
   <align:measure rdf:datatype="&xsd;float">0.77</align:measure>
  </align:Cell>
</align:map>
...
```



```
SELECT ?measurement ?psnr ?stalling WHERE {
        ?measurement rdf:type diim:Measurement ;
          diim:mean_seqPSNR ?psnr;
          diim:stalling ?stalling.
}
```

```
SELECT ?measurement ?psnr WHERE {
        ?measurement rdf:type diim:Measurement ;
          diim:psnr ?psnr.
}
```

```
SELECT ?measurement ?psnr ?stalling WHERE {
        ?measurement rdf:type diim:Measurement ;
          diim:psnr ?psnr;
          diim:stall ?stalling.
}
```

Figure 5.16: Query datasets as a single source in the showcase

Table 5.13: Alignment-based annotation of Aleksandrlvchenko graph with references to the terms from other datasets and domain-specific standards

```
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and LIVENetflix —->
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_LIVENetflix_psnr" />
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and DashReStreamer —->
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_DashReStreamer_psnr"/>
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_stalling_DashReStreamer_stall"/>
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and IEEEIQA —->
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_IEEEIQA_psnr"/>
<!—- EDOAL-based alignment representation between Aleksandrlvchenko and QoEAnalysis —->
  <align:Cell rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_seqpsnr_QoEAnalysis_psnr"/>
  <a rdf:about=".../Aleksandrlvchenko#SSM_Aleksandrlvchenko_stalling_QoEAnalysis_stall"/>
...
```

terms in the $LIVENetflix$ graph, and also annotate the $LIVENetflix$ graph towards the $Aleksandrlvchenko$ graph. As shown in Figure 5.16, a federated SPARQL query using both terms $seqpsnr$ from $Aleksandrlvchenko$ and $psnr$ from $LIVENetflix$ can extract data from these graphs as a single source. Therefore, they have been harmonised into a single, integrated, linked data source. In this way, we have enabled semantic interoperability between both datasets.

Similarly, we annotate terms $seqpsnr$, $stalling$ and $stall$ with reference links to and from the input datasets ($Aleksandrlvchenko$, $psnr$ $LIVENetflix$, and $DashReStreamer$) and domain-specific standards ($IEEIQA$ and $QoEAnalysis$), to enhance their interoperability in the QoE domain. Now, for all input datasets, we have enhanced their interoperability, as they are not only semantically harmonised with one another but also aligned with standards.

Table 5.14: Interoperability requirement matrix in the showcase

Name	Data format (DF)	Syntactic requirement of $QoE - App$	Selected Terms	Semantic requirement of $QoE - App$
$AleksandrIvchenko$	CSV	No, CSV = CSV	seqpsnr, stalling	No, (seqpsnr, stalling) = (seqpsnr, stalling)
$LIVENetflix$	CSV	No, CSV = CSV	psnr	Yes (psnr) ≠ (seqpsnr, stalling)
$DashReStreamer$	CSV	No, CSV = CSV	psnr, stall	Yes (psnr, stall) ≠ (seqpsnr, stalling)

Validation

As shown in Table 5.14, we do not have no syntactic requirements. Because all datasets are in CSV format, the QoE prediction application's interface accepts CSV data. However, we have a semantic requirement to use the $LIVENetflix$ and $DashReStreamer$ datasets. The prediction application searches for data stored under the terms $(seqpsnr, stalling)$, which are not present in the $LIVENetflix$ and $DashReStreamer$ datasets. Therefore, DIIM steps are applied to enable semantic interoperability.

After DIIM's application, all datasets represented in KGs are aligned, and we query them as a single source. We will obtain all answers with $true$ responses by completing the DIIQ questionnaire. Thus, full interoperability is enabled in the showcase.

DIIM also enhanced the syntactic and semantic interoperability of the input datasets. At DIIM's *transformation* step, we make them syntactically interoperable across many formats by converting them from CSV to RDF using RML (Ben De Meester et al., 2022), i.e. turning them into knowledge graphs. These knowledge graphs can be converted into an application-specific format using an RDF converter if the application doesn't support the CSV format. We also aligned the datasets with other domain-specific datasets and standards during DIIM's *alignment and storage* step. This enables them to adapt and use these standards in supporting applications. Hence, it enhances the semantic interoperability of the input datasets.

5.4.4 Showcase summary

We identified the harmonisation issues that arise when predicting 5G/6G data sourced from different applications, which may use different representation formats or label terms to denote the same data values in their data models. To address these issues, we proposed the DIIM approach and validated DIIM's application and evaluation by semantically harmonising the given datasets with the standards from the QoE domain. We found similar terms in datasets and standards using the SSM algorithm in DS3T. By storing 5G/6G data in separate knowledge graphs and interlinking their terms by similarity, we can query data across graphs as a single dataset. In the validation study, we confirmed the benefit of QoE prediction across datasets, although they do not originate from the same application. In the future, we will investigate other NLP algorithms for term alignment.

Chapter 6

Conclusions

The preceding chapter presented the validation showcases from the IoT, water, and 5G/6G domains. In these showcases, the application of DIIM and DIIF demonstrated how interoperability issues, such as data harmonisation at the syntactic and semantic levels, are addressed by leveraging SW and NLP technologies. This chapter concludes the research presented in this thesis by providing explicit conclusions for each of the research objectives outlined in Section 1.3, a summary of the research contributions, and a discussion of its merits, limitations, and future work directions.

6.1 Research Contributions

This thesis advances the field of data and information interoperability for IoT-enabled systems by providing a coherent set of theoretical, methodological, and practical contributions.

From a **theoretical** perspective, the research introduces two foundational conceptual models, the DIRM and DIIM. DIRM establishes a structured representation of data, information, and knowledge across domain layers, thereby clarifying how these elements manifest within the water domain. DIIM further contributes theoretically by formalising the process of achieving interoperability through a three-step procedure: graph transformation, ontology, and standard-based alignment with persistent storage, along with syntactic and semantic validation.

The **methodological** contributions are embodied in the operationalisation of DIIM through the development of the DIIF. DIIF comprises a suite of interoperable components that collectively enable the systematic application of DIIM: the DIIO for representing domain knowledge; the S3O for capturing similarity scores derived from heterogeneous algorithms; the DS3T, which integrates Semantic Web technologies, machine learning, neural networks, and natural language processing to compute and store semantic similarities; and the DIIQ, which provides a test-driven mechanism for validating syntactic and semantic interoperability. Together, these elements establish a unified methodological pipeline that supports reproducible, scalable, and interoperable processes.

The **practical** contributions are demonstrated through three validation showcases conducted across distinct IoT-enabled domains. The first, focusing on SWN monitoring and management, evidences the ability of DIIM to resolve interoperability challenges arising from heterogeneous data formats and standards. The second, concerning the alignment of IoT data for water systems, demonstrates how DIIF facilitates cross-application data use by aligning IoT datasets with established ontologies and standards. The third, centred on QoE prediction using 5G/6G datasets, confirms the applicability of DIIM and DIIF beyond the water domain, thereby underscoring their generalisability across diverse IoT-enabled contexts. These showcases verify both the operational feasibility and domain-agnostic utility of the proposed models, methodologies, and tools.

In sum, this thesis presents an integrated body of theoretical insights, methodological innovations, and practical validations that collectively advance the understanding and realisation of data and information interoperability within IoT-enabled environments. The contributions presented herein provide a foundation for future research and development aimed at enhancing interoperability across increasingly complex and heterogeneous IoT ecosystems for SWNs.

6.2 Research Conclusions

This section summarises, chapter by chapter, how the research successfully fulfilled the objectives set in Section 1.3.

1. Chapter 1 introduced the concept of IoT-enabled SWNs and the interoperability problem faced by SWN applications when they seek to use data collected by IoT from the physical layer of a water network. It explains why interoperability (the exchange and utilisation of IoT data) is crucial for the success of an IoT-enabled SWN and remains an active research topic. Objective 1 was completed in this chapter by identifying the interoperability problem in IoT-enabled SWNs and presenting an outline of the research methodology to address it.
2. Chapter 2 covered the necessary background and concepts to comprehend the developed models, methodology, and framework (contributions) presented in chapters 3 and 4. This chapter also reviewed the related work on interoperability approaches in the IoT and water domains. To achieve Objective 2, the literature review concluded by listing the challenges and research gaps in the IoT-enabled SWNs.
3. Chapter 3 proposed the conceptual Data Information Interoperability Model and Methodology (DIIM) to enable and enhance the interoperability of IoT data at the syntactic and semantic levels, thereby achieving Objective 3. Initially, the Data Information Representation Model (DIRM) is conceptualised to capture how data are collected and represented in SWNs. With the assumption that DIRM is used in SWNs to represent data and information, the concept of DIIM is proposed with three steps: *(i) transformation, (ii) alignment and storage, (iii) validation* to enable and enhance the interoperability of IoT data. The first step is to transform any semi-structured data into a knowledge graph. In the second step, the input data is aligned with domain-specific standards and ontologies, and alignment information is stored in the knowledge graph (created in the first step) as annotations. The generated knowledge graph with annotations is validated for syntactic and semantic interoperability in the validation step. In DIIM, the model is seen as a system that takes inputs and generates outputs. The methodology is the system's behaviour that enables interoperability in three steps.
4. Chapter 4 proposed a DIIM-based modular framework and web platform, Data Information Interoperability Framework (DIIF), to achieve smart data and information interoperabil-

ity among IoT-enabled SWN applications.DIIF is designed and developed with crucial interoperability components, Data Information Interoperability Ontology (DIIO), Semantic Similarity Scoring Ontology (S3O), DIIM Semantic Similarity Scoring Tool (DS3T), and Data Information Interoperability Questionnaire (DIIQ) to accomplish Objective 4. Ontologies DIIO and S3O are described in OWL, and their rules are expressed in SWRL. Where DIIO, S3O, and DS3T support enabling and enhancing the interoperability of IoT data at the syntactic and semantic levels, DIIO and Data Information Interoperability Questionnaire (DIIQ) help validate the syntactic and semantic interoperability of IoT data for SWN applications.

DIIO is developed as the KB of DIIF, containing the information on IoT data, standards, and ontologies across the IoT, water, and SW domains. DIIO can be queried to determine which data formats can be converted to other formats to enable and enhance syntactic interoperability. DIIO can also indicate which standards or ontologies support which others. This information is used to adopt an ontology or standard for IoT data when it uses a particular standard or ontology. Adopting other ontologies and standards for IoT data enables and enhances semantic interoperability for SWN applications.

S3O is designed to accommodate all possible algorithms and tools for computing the similarity of terms used in IoT data, standards, and ontologies, as well as the similarity among them. S3O extends DIIO by adding an abstract semantic-similarity layer to the alignment process, enabling consideration of algorithm- or tool-based semantic-similarity scores across diverse situations.

When IoT data is registered to DIIF, and if the data is not in a graph format such as RDF, it is transformed into KGs using RML to realise the DIIM *transformation* step.

DS3T is developed as an extensible semantic similarity scoring tool that utilises various NLP and NN algorithms and technologies to compute term similarity across IoT data, standards, and ontologies. It uses S3O, i.e. authored with SW technologies, RDF/OWL, to store the similarity scores as a graph that is linked with the newly generated KGs of IoT data and DIIO. After analysing similarity scores in S3O and confirming alignments, the alignments are stored in the EDOAL graph and the linked to the respective terms in IoT KGs. DIIF orchestrates various tools and technologies, such as DS3T, S3O, NLP, NN, EDOAL, KGs, EDOAL to realise DIIM's *alignment and storage* step.

A requirement-driven, test-based DIIQ is developed to validate interoperability between two IoT-enabled applications. DIIQ consists of three questionnaires: personal, syntactic, and semantic. The personal questionnaire validates the technical ability of the person conducting the tests for syntactic and semantic interoperability and completes the syntactic and semantic questionnaires. DIIQ classifies the interoperability as complete (all test results are true), partial (at least one test result is false), or no interoperability (all test results are false). An application's data interoperability is enhanced when it meets the interoperability requirements of two or more applications.

DIIF is designed as a modular framework, meaning that each component can be used independently and replaced with any other component that performs better. DIIF serves as a web platform that provides services for IoT and SWN applications. DIIF's web services interface allows applications to register and subscribe to IoT data. DIIF acts as a mediator between IoT and SWN applications, pulling data from IoT and pushing it to subscriber applications by converting it to the required format and adhering to the subscriber's standards or ontology. DIIF's Data Information Knowledge Management (DIKM) module is designed to align with SW technologies for storing and managing data, information, and knowledge related to IoT, as well as with domain-specific standards, ontologies, and applications. The DIKM module consists of three layers: an access layer for accessing data, a source layer for storing source data, and a linked layer for storing and linking graphs.

5. Chapter 5 presented three validation showcases of implementing the proposed DIIM and DIIF in different application domains to complete Objective 5.

The first showcase, *IoT-enabled SWN monitoring and management*, validated the novel DIIM and DIIF by mediating between two different monitoring systems that lack a common ontology and use different IoT devices to monitor and manage their water. DIIM overcomes the syntactic heterogeneity that arises when IoT devices are from different manufacturers, which use different standards to produce IoT, by transforming the semi-structured IoT data from the reservoir monitoring system into an RDF graph and converting it to meet the water bottling application's decision-making requirements. DIIM addresses the semantic heterogeneity between the two systems by aligning their semantic models. The alignment references are stored in the IoT's semantic model as annotations, which are used to adopt other applications' terminology for translation when those applications query the IoT data. The second showcase, *Alignment of IoT data with domain-specific ontologies*, validated the novel DIIM and DIIF using SW technologies (OWL, KG, RDF, and LD) and NLP (LSI, Word2Vec, and N-gram similarity) to discover related standards or ontologies and align IoT data terms with these standards or ontologies from the water domain. Further, it contributed to the S3O and DS3T components of DIIF. S3O can store similarity scores for terms, IoT data, standards, and ontologies, derived using various algorithms, without being bound to a specific algorithm or technology. DS3T takes input from IoT data, standards, and ontologies, and generates a S3O-based KG that contains N-gram-based similarity scores calculated by the applied algorithms.

The third showcase, *IoT-enabled 5G/6G datasets for QoE prediction*, validated the novel DIIM and DIIF in IoT-enabled 5G/6G application domains, confirming that the proposed model, methodology, and framework are not bound to the SWNs. They are domain-independent and can address the syntactic and semantic interoperability issues. DIIF stores the 5G/6G data in separate knowledge graphs and interlinks their terms based on similarity to build a single-sourced dataset. Thus, syntactic and semantic harmonisation issues are resolved by creating single-sourced data from multiple graphs, enabling IoT data to be queried as if it had originated from a single application.

6.3 Merits and Limitations

The following sections briefly list the merits and limitations of the Data Information Interoperability Model and Methodology (DIIM) and the Data Information Interoperability Framework (DIIF).

Merits of DIIM and DIIF against other frameworks

- The proposed model and methodology (DIIM) and the framework (DIIF) for IoT-enabled SWNs are domain independent, i.e. they could be used for any domain that digitalises its data and faces interoperability issues at the syntactic and semantic layers.
- DIIF utilises standardised SW technologies, such as RDF/OWL, to represent KGs, and the RDF converter to convert data into KGs and transform KGs into the required data format. If a new data format emerges, the W3C will provide guidelines for developing an RDF converter for that format.
- For alignment of data with domain-specific standards and ontologies, DIIF's component DS3T utilises state-of-the-art technologies, such as NLP-based algorithms SSM (Edit-distance-based) and word to vector (ML NN-based), to calculate similarities among terms in input datasets, ontologies, and standards, and to store the results in an RDF-based S3O.
- DIIF has a modular design, i.e. each component is an independent, interchangeable module with well-defined responsibilities. Thus, not only does DIIF provide the flexibility to use other better-established tools or technologies, but each component, such as DIIO, S3O, and DS3T, can be used in other frameworks or applications without applying the DIIF to the entire system.
- DIIF's web interface allows it to act as a mediator between two running applications or systems. It can also import legacy data and make it available for analysis and decision-making applications.
- During the semantic alignment of data originating from different applications, DIIF also considers converting measurement values to the target application's measurement units when they differ.

- DIIF manages data, information, and knowledge in interlinked KGs. If sourced IoT data are not represented in RDF/OWL, they are transformed into KGs using an RDF converter. As sourced IoT data is annotated and linked to existing standards, ontologies, and KGs. Keeping IoT data in interlinked federated KGs provides more context and can be treated as a single source. This enables querying and reasoning about data within a broader context.
- Semantic similarity, due to its divergent nature, can vary in different situations, and various tools or technologies can calculate the similarity between two terms, IoT datasets, ontologies, or standards differently. Diversity in the calculation of semantic similarity is well supported in S3O and DS3T, which are designed to be extensible.
- DIIM and DIIF provide IoT that can adapt to all possible standards and ontologies simultaneously, without fixating on a single standard or ontology. Thus making it future-proof.

Limitations of the DIIM and DIIF's Components

- DIIM and DIIF enable and enhance interoperability only at the syntactic and semantic levels of the interoperability model (see Section 2.8). DIIM and DIIF do not address other interoperability levels.
- Although some crucial interoperability components, such as DIIO, S3O, and DS3T, were developed for DIIF, they are not yet fully implemented. Therefore, further implementation is required to utilise it as a mediator software for web applications.
- Semantic similarity calculation in DIIF is limited by the algorithms (SSM and Word2Vec) implemented in DS3T. Other algorithms or technologies may require further implementation to meet the requirements.
- Despite S3O and DS3T providing all possible semantic similarities between two terms, the selection and confirmation of the final alignment are best left to domain experts, who know the application domain better.
- Although DIIM and DIIF focus on aligning semi-structured IoT data with other datasets, ontologies, and standards, they can also align structured data with ontologies and each other. However, the currently implemented solution relies on N-gram term similarity,

which assumes a key-value relationship between an entity label and its values. Therefore, the current implementation cannot calculate the semantic similarity of an entity defined as a complex data type.

- The recorded domain knowledge in DIIO and S3O ontologies requires periodic review and maintenance by domain experts. Otherwise, it could provide outdated or incorrect results to queries.
- DIIF will require enormous computing power to process many datasets, ontologies, and standards. Therefore, it faces scalability and performance challenges similar to those encountered when loading, storing, and updating large datasets, commonly referred to as Big Data.
- DIIF, as a web platform, could face security issues, e.g. data theft and denial-of-service attacks, as the significance of IoT data and the number of applications for data pull and push increase. Such security issues are not yet addressed in DIIF.

6.4 Future Work

This thesis's proposed model, method, and framework pave the way for further advances in research that should be theoretically investigated and empirically evaluated. Here are a few potential ideas for future work to consider:

- Further development of DIIO to manage more domain-specific knowledge from various domains, e.g. rather explicitly in programming as software code or relying on manual intervention of domain experts at runtime.
- The developed DIIF can be further developed as a Multi-Agent System (MAS) that utilises the Believe Desire Intention (BDI) model (Rao et al., 1998) to execute and orchestrate DIIF processing tasks, e.g. health, agriculture, finance, energy, etc., service discovery, translation, tagging, and validation, automatically and efficiently at runtime.
- In the future, DIIM and DIIF can also cover interoperability levels, such as pragmatic, dynamic, and conceptual, as the preceding levels, syntactic and semantic, are fundamental to them.

- S3O and DS3T could be further developed and implemented with various other algorithms to support the complex alignment of data, standards, and ontologies.
- Efficiently creating, loading, updating, and querying interlinked KGs in which domain-specific IoT data, standards, and ontologies are stored.
- Integrating DIIF with other open source projects and initiatives, e.g. Fiware4water (Fiware4Water, n.d.) and Data Spaces (Publications Office of the European Union, n.d.).

In conclusion, integrating IoT into SWNs introduces new sensing, control, and automation capabilities. However, it also brings complex heterogeneity, data management, and interoperability challenges. This research has directly addressed these issues by structuring data semantics and exchange, bridging legacy and modern systems, supporting cross-domain data fusion, and reproducible research. Thus, enabling scalable, intelligent, and collaborative water network management for IoT-enabled SWNs. In particular, the thesis advances interoperability for IoT-enabled SWNs by introducing an innovative Data Information Interoperability Framework (DIIF) based on DIIM. This framework enhances the syntactic and semantic interoperability of IoT data by treating it as an annotated RDF/OWL graph, with annotations aligned with various standards and ontologies. DIIF utilises SW, NN, ML, and NLP technologies to convert datasets into KGs, which can be converted into a specific data format and adopt domain-specific standards or ontologies as required. DIIF, as a web platform, can easily integrate into new and existing legacy systems through its web services interface.

Acronyms

AI Artificial Intelligence. 34, 155

AMQP Advanced Message Queuing Protocol. 29

API Application Programming Interface. 38

AR Augmented Reality. 152

ASCII American Standard Code for Information Interchange. 28

ATOMS Advanced Topics on Measurement and Simulation. xviii, 153, 270

BCU Birmingham City University. 207

BDI Believe Desire Intention. 40

BI Business Intelligence. 54

BIGIoT Bridging the Interoperability Gap of the IoT. 36, 61

BII Business Insider Intelligence. 27

CBOW Continuous Bag-of-Words. 52

CC Cloud Computing. 38

CoAP Constrained Application Protocol. 29, 36, 37, 43, 61

COSMO COmmon Semantic MOdel. xvii, 115, 149, 151

CPSs Cyber-physical systems. 29

CPU Central Processing Unit. 114

CSV Comma-separated Values. 1, 37, 59, 60, 76, 83, 89, 118, 138, 139, 147, 148, 157, 158, 160, 164

DCOM Distributed Component Object Model. 37

DDIRL Digital Data Information Representation Layer. 94

DDS Data Distribution Service. 11

DeX Data eXchange. 38

DeXMS Data eXchange Mediator Synthesizer. xix, 38

DIEK Data Information Evidence Knowledge. xix, 14

DIIF Data Information Interoperability Framework. xvi, xvii, 6, 7, 9, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 104, 105, 108, 111, 112, 127, 128, 131, 132, 133, 134, 135, 136, 147, 158, 167, 168, 170, 171, 172, 173, 174, 175, 176, 209, 255, 261

DIIM Data Information Interoperability Model. xvi, xx, 6, 7, 72, 74, 75, 77, 78, 82, 84, 88, 89, 90, 91, 94, 95, 113, 133, 135, 139, 140, 141, 142, 145, 146, 157, 164, 165, 167, 170, 171, 172, 174

DIIM Data Information Interoperability Model and Methodology. xvi, xvii, xix, xx, 6, 7, 9, 65, 66, 70, 72, 73, 75, 78, 79, 80, 86, 89, 90, 91, 92, 93, 94, 97, 113, 114, 115, 116, 127, 128, 130, 131, 132, 133, 135, 136, 144, 145, 146, 157, 158, 164, 167, 168, 169, 171, 172, 173, 174, 175, 176

DIIO Data Information Interoperability Ontology. xvi, xvii, 94, 102, 104, 105, 106, 108, 109, 110, 134, 168, 170, 173, 174, 175

DIIQ Data Information Interoperability Questionnaire. 127, 128, 130, 133, 134, 164, 168, 171

DIKM Data Information Knowledge Management. 171

DIKW Data Information Knowledge Wisdom. xv, 12, 13, 14, 63

DIKW/I Data Information Knowledge Wisdom/Intelligence. 14

DIRM Data Information Representation Model. 6, 65, 66, 70, 72, 78, 82, 90, 91, 92, 94, 97, 133, 167, 169

DL Deep Learning. 69

DS3T DIIM Semantic Similarity Scoring Tool. xviii, 94, 104, 113, 115, 116, 118, 120, 121, 123, 124, 133, 134, 136, 145, 148, 157, 158, 159, 160, 161, 165, 168, 170, 171, 172, 173, 174, 176

DSL Digital Subscriber Line. 28

DSS Decision Support System. xix, 32, 33, 34, 53, 54, 55, 62, 63, 68

DUL DOLCE Ultra Lite. 115

EC Edge Computing. 58

EC FP7 European Commission Seventh Framework Programme. 53, 63

EDOAL Expressive and Declarative Ontology Alignment Language. xx, 133, 160, 162, 163, 171

ETSI European Telecommunications Standards Institute. 44

EU European Union. 38

EWQM Enterprise Water Quality Monitoring. xvii, 138, 139, 141, 144

F-Logic (Frame Logic. 70, 76, 77

FTTH Fibre to the Home. 28

FTTP Fibre to the Premises. 28

GIS Geographic Information System. 31, 37, 53

GML Geography Markup Language. 37

GO Gene Ontology. 26, 58, 115, 190

GOIoTP Generic Ontology for IoT Platforms. 45, 46, 80, 103, 149

GOIoTPex Generic Ontology for IoT Platforms Extended. 46

GUI Graphical User Interface. 134

HDWG Hydrology Domain Working group. 44

HTML HyperText Markup Language. 118, 148

HTTP HyperText Transfer Protocol. 17, 37

ICT Information and Communications Technology. 33, 52, 68

IDSS Intelligent Decision Support Systems. 54, 55

IEC International Electrotechnical Commission. 20

IEEE Institute of Electrical and Electronics Engineers. xviii, 20, 25, 89, 153, 192, 270

IETF Internet Engineering Task Force. 19

IFC4 Industry Foundation Classes 4. 41, 45, 80

INSPIRE Infrastructure for spatial information in Europe. 41, 45, 80

IoE Internet of Everything. 23

IoT Internet of Things. xv, xvii, xix, xx, 1, 2, 3, 4, 6, 7, 9, 11, 23, 26, 27, 28, 29, 30, 33, 34, 35, 36, 37, 38, 39, 40, 41, 43, 44, 45, 46, 47, 52, 53, 54, 57, 58, 60, 61, 62, 63, 64, 65, 73, 79, 82, 84, 89, 92, 93, 95, 97, 98, 99, 100, 101, 102, 103, 104, 108, 110, 111, 112, 113, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 131, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 144, 145, 146, 147, 148, 149, 151, 152, 157, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176

IP Internet Protocol. 27, 28, 37

IPSM Inter Platform Semantic Mediator. 40

IRI Internationalized Resource Identifier. 19, 20

ISDN Integrated Services Digital Network. 28

ISO International Organization for Standardization. 20, 28, 37, 44, 45, 149

JSON JavaScript Object Notation. 1, 29, 37, 59, 60, 76, 83, 113, 114, 115, 118, 138, 139, 140, 144, 148

JSON-LD JavaScript Object Notation for Linking Data. 69, 78

KB Knowledge Base. xx, 95, 97, 111, 141, 170

KG Knowledge Graph. 15, 19, 59, 89, 113, 130, 131, 133, 134, 151, 157, 158, 172, 174

KGs Knowledge Graphs. 15, 16, 19, 47, 59, 98, 104, 132, 133, 134, 164, 170, 171, 173, 174, 176

KIF Knowledge Interchange Format. 70, 76, 77

KRR Kernel Ridge Regression. 154

LCA Life Cycle Assessment. 54, 55

LCIM Levels of Conceptual Interoperability Model. xv, xvi, 23, 25, 26, 65, 72, 89, 90, 91

LD Linked Data. 17, 39, 54, 133, 144, 151, 172

LDIRL Linked Data Information Representation Layer. 94

LLM Large Language Model. 48, 49, 59

LOD Linked Open Data. 17

LR Linear Regression. 154

LSI Latent Semantic Index. xvii, 118, 119, 120, 124, 126, 151, 172

LWM2M Lightweight Machine-to-Machine. 37

M2M Machine-to-Machine. 34

MAC Media Access Control. 38

MAE Mean Absolute Error. 154

MAS Multi-Agent System. 33, 34, 40, 43, 53, 62, 63

MCDM Multi-Criteria Decision Making. 54, 55

MDA Model-Driven Architecture. 65, 72, 79, 92, 133

MDE Model-Driven Engineering. 38

ML Machine Learning. 4, 6, 133, 153, 173, 176

MM Mathematical Models. 54, 55

MOS Mean Opinion Score. xvii, 154, 155

MQTT Message Queuing Telemetry Transport. 28, 36, 37, 43, 61

MSE Mean Squared Error. 154

N3 Notation3. 69

NCOIC Network Centric Operations Industry Consortium. 22

NFC Near-field communication. 28

NLP Natural Language Processing. 4, 6, 47, 49, 50, 52, 84, 102, 111, 112, 113, 116, 117, 118, 119, 127, 133, 136, 147, 151, 156, 165, 167, 170, 171, 172, 173, 176

NLTK Natural Language Toolkit. 114, 115

NN Neural Network. 4, 6, 133, 170, 171, 173, 176

OAEI Ontology Alignment Evaluation Initiative. 49

ODBC Open Database Connectivity. 37

OGC Open Geospatial Consortium. 26, 37, 44, 45, 54, 59, 89, 149, 197

OLAP On-Line Analytical Processing. 53

OMG Object Management Group. 24, 197

OPC DA Open Platform Communications Data Access. 37

OPC UA Open Platform Communications Unified Architecture. 29

ORL Ontological Representation Layer. 94

OSI Open Systems Interconnection. xv, xvi, 25, 26, 28, 37, 90, 91

OWL Web Ontology Language. 16, 17, 18, 19, 34, 37, 44, 45, 63, 69, 70, 72, 74, 75, 76, 77, 78, 79, 80, 82, 84, 85, 88, 92, 95, 97, 104, 114, 115, 118, 144, 145, 146, 148, 151, 170, 172, 173, 174, 176

OWL-S Semantic Markup for Web Services. 45, 95

PDF Portable Document Format. 157

PLCs Programmable Logic Controllers. 29

PRVs Pressure Reducing Valves. 30

PSNR Peak Signal-to-Noise Ratio. 154, 155

QoE Quality of Experience. xvii, xx, 136, 153, 154, 155, 163, 164, 165, 168, 172

QUDT Quantities, Units, Dimensions, Data Types. 139

RDBS Relational Database System. 59

RDF Resource Description Framework. xvii, 17, 19, 20, 34, 37, 40, 44, 45, 63, 69, 70, 72, 74, 75, 76, 78, 79, 80, 82, 88, 89, 92, 94, 97, 104, 107, 108, 118, 131, 132, 133, 138, 139, 140, 142, 144, 145, 146, 147, 148, 151, 157, 158, 164, 170, 172, 173, 174, 176

RDFS Resource Description Framework Schema. 16, 34, 70

REST Representational State Transfer. 17, 29, 36, 37, 43, 61

RIF Rule Interchange Format. 17

RML RDF Mapping Language. 158, 164, 170

S3O Semantic Similarity Scoring Ontology. xvii, 6, 94, 102, 111, 112, 124, 126, 133, 134, 136, 147, 148, 150, 151, 152, 157, 159, 160, 168, 170, 171, 172, 173, 174, 175, 176

SAREF Smart Appliances REFerence. xvii, 41, 44, 45, 61, 80, 103, 139, 141, 144, 149

SCADA Supervisory control and data acquisition. 29, 33, 37, 63

SEMIoTICS Smart End-to-end Massive IoT Interoperability, Connectivity and Security. 36, 37, 61

SensorML Sensor Model Language. 149

SKOS Simple Knowledge Organization System. 17, 70, 76, 77, 138

SNMP Simple Network Management Protocol. 29

SOA Service-Oriented Architecture. xvi, 38, 40, 95, 96, 135

SOAP Simple Object Access Protocol. 17, 29, 37

SOS Sensor Observation Service. 44

SOSA Sensor, Observation, Sample, and Actuator. 44, 45, 46, 80, 149

SPARQL SPARQL Protocol and RDF Query Language. xvii, 17, 19, 44, 63, 88, 108, 110, 148, 150, 160, 163

SSM String-search Matching. xviii, xx, 112, 115, 123, 124, 125, 126, 148, 159, 160, 161, 165, 173, 174

SSN Semantic Sensor Network. 39, 44, 45, 61, 80, 89, 106, 111, 115, 149

SSO Syndromic Surveillance Ontology. 80

SSTD Selected Set of Terms used in a Dataset. 79, 80

SVR Support Vector Regression. 154

SW Semantic Web. 4, 6, 16, 44, 52, 54, 65, 92, 98, 104, 133, 134, 135, 136, 156, 167, 170, 171, 172, 173, 176

SWAN Smart Water Networks Forum. xv, xix, 31, 32, 33, 37, 53, 66, 201

SWEET Semantic Web for Earth and Environment Technology. 80

SWG Smart Water Grid. 32

SWIM Semantic Water Interoperability Model. 41, 45, 46, 61, 80, 89, 115, 149

SWN Smart Water Network. xv, xvi, xvii, xix, 1, 2, 3, 7, 9, 29, 30, 32, 33, 34, 37, 40, 41, 44, 46, 52, 53, 54, 58, 59, 60, 61, 63, 64, 66, 69, 97, 100, 102, 103, 133, 135, 136, 137, 146, 157, 168, 169, 170, 171

SWNs Smart Water Networks. xvi, 3, 4, 6, 7, 9, 11, 26, 29, 34, 35, 36, 44, 53, 54, 60, 61, 62, 63, 66, 67, 68, 79, 92, 93, 94, 101, 103, 108, 128, 136, 146, 168, 169, 172, 173, 176

SWRL Semantic Web Rule Language. 19, 105, 170

TCP Transmission Control Protocol. 29, 37

TCP/IP Transmission Control Protocol/Internet Protocol. 28

TDD Thing Description Directory. 95, 100, 140

TXT Text. 157

UDP User Datagram Protocol. 37

UML Unified Modeling Language. 24

UPnP Universal Plug and Play. 29

URI Uniform Resource Identifier. 19, 75, 111, 116, 117, 118, 119, 133

URL Uniform Resource Locator. 104, 105, 112, 113

UTF-8 Unicode Transformation Format – 8-bit. 118

VPN Virtual Private Network. 137

VR Virtual Reality. 152

W3C World Wide Web Consortium. 16, 26, 39, 44, 45, 131, 173

WatERP Water Enhanced Resource Planning. 36, 40, 43, 45, 46, 52, 61, 80

WDS Water Distribution Systems. 57, 58

WDTF Water Data Transfer Format. 45, 80

WHO World Health Organization. 149, 204

WISDOM Water analytics and Intelligent Sensing for Demand Optimised Management. xvi, 36, 37, 40, 41, 43, 45, 46, 52, 61, 80, 82, 89, 103, 106, 115, 149

WMO Water Management Ontology. 40, 45, 46

Word2Vec Word2Vector. 112, 115, 120, 121, 122, 123, 124, 125, 126, 148, 151, 174

WoT Web of Things. 136, 137

WQMS Water Quality Monitoring System. xx, 136, 137, 138, 139, 140, 141, 144, 145, 146

WSDL Web Services Description Language. 37, 95

WUE Water Use Efficiency. 53

WWTP Wastewater Treatment Plant. xix, 55

XML Extensible Markup Language. 1, 37, 38, 45, 59, 69, 76, 80, 82, 118, 138, 139, 140, 144, 145, 148

XMPP Extensible Messaging and Presence Protocol. 29

References

Abu-Bakar, Halidu, Leon Williams, and Stephen Henry Hallett (2021). "A review of household water demand management and consumption measurement". In: *Journal of Cleaner Production* 292, p. 125872.

Anzaldi, Gabriel, E. Rubion, A. Corchero, R. Sanfeliu, X. Domingo, J. Pijuan, and F. Tersa (2014). "Towards an enhanced knowledge-based Decision Support System (DSS) for integrated water resource management (IWRM)". In: *Procedia Engineering* 89, pp. 1097–1104. ISSN: 18777058. DOI: 10.1016/j.proeng.2014.11.230. URL: http://dx.doi.org/10.1016/j.proeng.2014.11.230.

Anzaldi, Gabriel, Wenyan Wu, Andreas Abecker, Edgar Rubión, Aitor Corchero, Ambreen Hussain, and Michael Quenzer (2014). "Integration of water supply distribution systems by using interoperable standards to make effective decisions". In.

AQUAMATIX (2017). *Aquamatix Smart Water Systems*. accessed 22 Jan. 2025. URL: http://www.aquamatix.net/.

Arroyo, Paz and María Molinos-Senante (2018). "Selecting appropriate wastewater treatment technologies using a choosing-by-advantages approach". In: *Science of the Total Environment* 625, pp. 819–827.

Asim, Muhammad Nabeel, Muhammad Wasim, Muhammad Usman Ghani Khan, Waqar Mahmood, and Hafiza Mahnoor Abbasi (2018). "A survey of ontology learning techniques and applications". In: *Database : the journal of biological databases and curation* 2018. ISSN: 1758-0463. DOI: 10.1093/database/bay101. URL: https://europepmc.org/articles/PMC6173224.

Australian Government Linked Data Working Group (2016). *Water quality data archive*. accessed 6 July 2021. URL: http://environment.data.gov.au/.

Babalou, Samira, Elena Grygorova, and Birgitta König-Ries (2020). “CoMerger: a customizable online tool for building a consistent quality-assured merged ontology”. In: *In ESWC, Poster and Demo Track June*. accessed 22 Jan. 2021. URL: https://github.com/fusion-jena/CoMerger.

Bampis, C. G., Zhi Li, Ioannis Katsavounidis, Te-Yuan Huang, Chaitanya Ekanadham, and Alan C. Bovik (n.d.). *Towards Perceptually Optimized End-to-end Adaptive Video Streaming*. Accessed 17 May 2024. URL: https://live.ece.utexas.edu/research/LIVE_NFLX_II/live_nflx_plus.html.

Beckett, David and W3C (2014). *Semantic Web: N-Triples*. accessed 9 Mar. 2022. URL: https://www.w3.org/TR/n-triples/#:~:text=N-Triples/.

Beijering, Karin, Charlotte Gooskens, and Wilbert Heeringa (2008). “Predicting intelligibility and perceived linguistic distance by means of the Levenshtein algorithm”. In: *Linguistics in the Netherlands* 25.1, pp. 13–24.

Ben De Meester, Pieter Heyvaert and Delva Thomas (2022). *RDF Mapping Language (RML)*. Accessed 6 Jan. 2023. URL: https://rml.io/specs/rml/.

Berners-Lee, Tim (2006). *Linked Data*. URL: https://www.w3.org/DesignIssues/LinkedData.html.

Berners-Lee, Tim and W3C (1994). *World Wide Web Consortium (W3C): URI*. accessed 9 Mar. 2022. URL: https://www.w3.org/Addressing/URL/uri-spec.html.

— (2014a). *Semantic Web: N3*. accessed 9 Mar. 2022. URL: https://www.w3.org/TeamSubmission/n3/.

— (2014b). *Semantic Web: Turtle*. accessed 9 Mar. 2022. URL: https://www.w3.org/TR/turtle/.

Bertanza, Giorgio, Matteo Canato, Giuseppe Laera, and Maria Concetta Tomei (2015). “Methodology for technical and economic assessment of advanced routes for sludge processing and disposal”. In: *Environmental Science and Pollution Research* 22.10, pp. 7190–7202.

Bird, Steven and Liling Tan (2022). *Natural Language Toolkit*. accessed 12 Jan. 2022. URL: https://www.nltk.org/index.html.

Bluetooth SIG Inc (n.d.). *Bluetooth*. Accessed 31 Jan. 2025. URL: https://www.bluetooth.com/.

Bouloukakis, Georgios, Nikolaos Georgantas, Patient Ntumba, and Valérie Issarny (2019). “Automated synthesis of mediators for middleware-layer protocol interoperability in the IoT”. In: *Future Generation Computer Systems* 101, pp. 1271–1294. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2019.05.064. URL: http://www.sciencedirect.com/science/article/pii/S0167739X18323586.

Bristol City Council (2010). *Bristol river water quality*. accessed: 26/09/2022. URL: https://www.data.gov.uk/dataset/dd6658fc-7d1a-4ab2-9ea4-6aa936d21608/bristol-river-water-quality.

Britannica (2020). *Britannica: Water resource*. accessed 20 Jan. 2022. URL: https://www.britannica.com/science/water-resource.

Brodaric, Boyan (2007). “Geo-Pragmatics for the Geospatial Semantic Web”. In: vol. 11. 3, pp. 453–477.

Brodaric, Boyan, Nate Booth, Eric Boisvert, and Jessica Lucido (2015). “Groundwater data network interoperability”. In: *Journal of Hydroinformatics* 18.2, pp. 210–225. ISSN: 1464-7141. DOI: 10.2166/hydro.2015.242. eprint: https://iwaponline.com/jh/article-pdf/18/2/210/389069/jh0180210.pdf. URL: https://doi.org/10.2166/hydro.2015.242.

Bröring, A., S. Schmid, C. Schindhelm, A. Khelil, S. Käbisch, D. Kramer, D. Le Phuoc, J. Mitic, D. Anicic, and E. Teniente (2017). “Enabling IoT Ecosystems through Platform Interoperability”. In: *IEEE Software* 34.1, pp. 54–61.

Buitelaar, Paul, Daniel Olejnik, and Michael Sintek (2004). “A protégé plug-in for ontology extraction from text based on linguistic analysis”. In: *European Semantic Web Symposium*. Springer, pp. 31–44.

Buitelaar, Paul and Michael Sintek (2004). “Ontolt version 1.0: Middleware for ontology extraction from text”. In: *Proc. of the Demo Session at the International Semantic Web Conference*.

Cardiff University (n.d.). *Water Analytics and Intelligent Sensing for Demand Optimised Management (WISDOM)*. accessed 10 Jan. 2025. URL: https://www.cardiff.ac.uk/water-research-institute/research/activities/wisdom-project.

Cassidy, Patrick (2020). *COSMO*. accessed 23 Jan 2021. URL: http://micra.com/COSMO/.

Chhipi-Shrestha, Gyan, Kasun Hewage, and Rehan Sadiq (2017). “Fit-for-purpose wastewater treatment: Conceptualization to development of decision support tool (I)”. In: *Science of the Total Environment* 607, pp. 600–612.

Chow, Christopher WK, Jixue Liu, Jiuyong Li, Nick Swain, Katherine Reid, and Christopher P Saint (2018). “Development of smart data analytics tools to support wastewater treatment plant operation”. In: *Chemometrics and Intelligent Laboratory Systems* 177, pp. 140–150.

Ciancio, Julia, Catherine J Chomat, Jorge Helmbrecht, and Gabriel Anzaldi (2015). “WatERP: Water Enhanced Resources Planning (ERP)" Where water supply meets demand"”. In: *Natural Resources-Sustainable Targets, Technologies, Lifestyles and Governance*, p. 356.

Cimiano, Philipp and Johanna Völker (2005). “text2onto”. In: *International conference on application of natural language to information systems*. Springer, pp. 227–238.

Compton, Michael, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, et al. (2012). “The SSN ontology of the W3C semantic sensor network incubator group”. In: *Journal of Web Semantics* 17, pp. 25–32.

Daelemans, W and ML Reinberger (2004). “Shallow text understanding for ontology content evaluation”. In: *IEEE Intelligent Systems* 19.4, pp. 74–81.

Dammann, O. (2019). “Data, Information, Evidence, and Knowledge:: A Proposal for Health Informatics and Data Science”. In: URL: https://doi.org/10.5210/ojphi.v10i3.9631.

Dell’Aglio, Daniele, Emanuele Della Valle, Frank van Harmelen, and Abraham Bernstein (2017). “Stream reasoning: A survey and outlook”. In: *Data Science* 1. DOI: 10.3233/DS-170006.

Díaz-Madroñero, Manuel, Modesto Pérez-Sánchez, José R Satorre-Aznar, Josefa Mula, and P Amparo López-Jiménez (2018). “Analysis of a wastewater treatment plant using fuzzy goal programming as a management tool: A case study”. In: *Journal of cleaner production* 180, pp. 20–33.

DiGiuseppe, Nicholas, Line C Pouchard, and Natalya Fridman Noy (2014). “SWEET ontology coverage for earth system sciences”. In: *Earth Science Informatics* 7.4, pp. 249–264.

Duerst, Martin and Michel Suignard (2005). *Rfc 3987: Internationalized resource identifiers (iris)*.

Efeoglu, Sefika (2024). “Graphmatcher: A graph representation learning approach for ontology matching”. In: *arXiv preprint arXiv:2404.14450*.

Environment Agency GOV UK (2016). *Water quality data archive*. accessed 24 Jan. 2021. URL: https://environment.data.gov.uk/water-quality.

Escobar Esteban, María Pilar, María del Mar Roldán-García, Jesús Peral Cortés, Gustavo Candela Romero, and José García-Nieto (2020). *An Ontology-Based Framework for Publishing and Exploiting Linked Open Data: A Use Case on Water Resources Management*.

ETSI Technical Committee (2020). *Smart Applications REFerence Ontology, and extensions (SAREF)*. accessed 10 Dec. 2022. URL: https://saref.etsi.org/.

Euzenat, Jérôme, Jérôme David, and Manuel Atencia (n.d.). *EDOAL: Expressive and Declarative Ontology Alignment Language*. Accessed 18 Apr. 2025. URL: http://ns.inria.org/edoal/1.0/.

Euzenat, Jérôme and Pavel Shvaiko (2007). *Ontology matching*. Springer.

Farias, Tarcisio Mendes de, Ana Roxin, and Christophe Nicolle (2016). “SWRL rule-selection methodology for ontology interoperability”. In: *Data & Knowledge Engineering* 105, pp. 53–72.

Fiware4Water (n.d.). *Fiware4Water*. accessed 11 Apr. 2025. URL: https://www.fiware4water.eu/.

Foundation, Python Software (2001). *Python™*. accessed 7 Feb 2022. URL: https://www.python.org/.

Al-Fuqaha, A., M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash (2015). “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications”. In: *IEEE Communications Surveys Tutorials* 17.4, pp. 2347–2376.

Gal, Avigdor, Giovanni Modica, and Hasan Jamil (2004). “Ontobuilder: Fully automatic extraction and consolidation of ontologies from web sources”. In: *Proceedings. 20th International Conference on Data Engineering*. IEEE, p. 853.

Ganzha, Maria, Marcin Paprzycki, Wiesław Pawłowski, Paweł Szmeja, and Katarzyna Wasielewska (2017). “Semantic interoperability in the Internet of Things: An overview from the INTER-IoT perspective”. In: *Journal of Network and Computer Applications* 81, pp. 111–124. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2016.08.007. URL: http://www.sciencedirect.com/science/article/pii/S1084804516301618.

Garrido-Baserba, MMSPG, M Molinos-Senante, José María Abelleira-Pereira, LA Fdez-Güelfo, M Poch, and F Hernández-Sancho (2015). “Selecting sewage sludge treatment alternatives in modern wastewater treatment plants using environmental decision support systems”. In: *Journal of Cleaner Production* 107, pp. 410–419.

Gémar, Germán, Trinidad Gómez, María Molinos-Senante, Rafael Caballero, and Ramón Sala-Garrido (2018). “Assessing changes in eco-productivity of wastewater treatment plants: The role of costs, pollutant removal efficiency, and greenhouse gas emissions”. In: *Environmental Impact Assessment Review* 69, pp. 24–31.

Genesereth, Michael R and Richard E Fikes (1992). *Knowledge interchange format-version 3.0: Reference manual*.

Gibson, J.A.E (2016). *Modern day limnology and palaeolimnology of lakes in the Framnes Mountains and Stillwell Hills*. accessed 24 Jan. 2021. DOI: doi:10.4225/15/574BC23AC70FA. URL: https://data.aad.gov.au/metadata/records/ASAC_2509.

Giglou, Hamed Babaei, Jennifer D’Souza, Oliver Karras, and Sören Auer (2025). “Ontoaligner: A comprehensive modular and robust python toolkit for ontology alignment”. In: *arXiv preprint arXiv:2503.21902*.

GitLab (2014). *GitLab*. accessed 7 Feb 2022. URL: https://about.gitlab.com/.

GO Consortium (2020). *Gene Ontology*. accessed 9 Mar. 2022. URL: http://geneontology.org/.

Gómez-Pérez, Asunción, Mariano Fernández-López, and Oscar Corcho (2004). *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer.

Gómez-Pérez, Asunción and David Manzano-Macho (2004). “An overview of methods and tools for ontology learning from texts”. In: *The knowledge engineering review* 19.3, pp. 187–212.

Groß, Anika, Cédric Pruski, and Erhard Rahm (2016). “Evolution of biomedical ontologies and mappings: overview of recent approaches”. In: *Computational and structural biotechnology journal* 14, pp. 333–340.

Grueau, Cédric, André Antunes, Bruno Ferreira, Miguel Gonçalves, and Nelson Carriço (2019). “Towards an integrated platform for decision support in water utility management”. In.

Guarino, Nicola (1998). *Formal ontology in information systems: Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy*. Vol. 46. IOS press.

Guarino, Nicola and Aldo Gangemi (2017). *DOLCE-Ultralite*. accessed 23 Jan 2021. URL: www.loa.istc.cnr.it/dolce/overview.html.

Hatzivasilis, George, Ioannis Askoxylakis, George Alexandris, Darko Anicic, Arne Bröring, Vivek Kulkarni, Konstantinos Fysarakis, and George Spanoudakis (2018). "The Interoperability of Things: Interoperable solutions as an enabler for IoT and Web 3.0". In: *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE, pp. 1–7.

He, Yuan, Jiaoyan Chen, Denvar Antonyrajah, and Ian Horrocks (2022). "BERTMap: a BERT-based ontology alignment system". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 5, pp. 5684–5691.

Hendler, James, Tim Berners-Lee, and Eric Miller (2002). "Integrating Applications on the Semantic Web". In: *Journal of the Institute of Electrical Engineers of Japan* 122(10), pp. 676–680. URL: https://www.w3.org/2002/07/swint.html.

Heyvaert, Pieter, Ben De Meester, Anastasia Dimou, and Ruben Verborgh (2018). "Declarative Rules for Linked Data Generation at your Fingertips!" In: *Proceedings of the 15th ESWC: Posters and Demos*.

Hodzic, Kerim and et al. (2022). "Realistic video sequences for subjective QoE analysis". In: *Proceedings of the 13th ACM Multimedia Systems Conference*. MMSys '22. Athlone, Ireland: Association for Computing Machinery, pp. 246–251. ISBN: 9781450392839.

Hodzic, Kerim, Mirsad Cosovic, Sasa Mrdovic, Jason J. Quinlan, and Darijo Raca (n.d.). *DashReStreamer*. Accessed 17 May 2024. URL: https://github.com/khodzic2/DashReStreamer/tree/master.

Hofer, Marvin, Daniel Obraczka, Alieh Saeedi, Hanna Köpcke, and Erhard Rahm (2024). "Construction of Knowledge Graphs: Current State and Challenges". In: *Information* 15.8. ISSN: 2078-2489. DOI: 10.3390/info15080509. URL: https://www.mdpi.com/2078-2489/15/8/509.

Horridge, Matthew and Mark Musen (2016). "Snap-SPARQL: a java framework for working with SPARQL and OWL". In: *Ontology Engineering: 12th International Experiences and*

Directions Workshop on OWL, OWLED 2015, co-located with ISWC 2015, Bethlehem, PA, USA, October 9-10, 2015, Revised Selected Papers 12. Springer, pp. 154–165.

Horrocks, Ian, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean (2004). *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. Accessed 18 Apr. 2025. URL: https://www.w3.org/submissions/SWRL/.

Howell, Shaun, Yacine Rezgui, and Thomas Beach (2017). "Integrating building and urban semantics to empower smart water solutions". In: *Automation in Construction* 81, pp. 434–448.

— (2018). "Water utility decision support through the semantic web of things". In: *Environmental Modelling & Software* 102, pp. 94–114.

Howell, Shaun, Yacine Rezgui, Tom Beach, Wanqing Zhao, Julia Terlet, and Haijiang Li (2016). "Smart Water System Interoperability: Integrating Data and Analytics for Demand Optimized Management through Semantics". In: *ICCCBE*, pp. 1–9.

HyperCat (n.d.). *HyperCat*. accessed 9 Mar. 2022. URL: https://hypercatiot.github.io/.

IEEE (2004). *Suggested Upper Merged Ontology (SUMO)*. accessed 23 Jan 2021. URL: www.ontologyportal.org.

— (2011). "IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads". In: *IEEE Std 2030-2011*, pp. 1–126.

INSPIRE, EU (2015). *INSPIRE Ontology*. accessed 10 Dec. 2022. URL: https://inspire.ec.europa.eu/glossary/Ontology.

Islam, Mohammed Rezwanul, Sami Azam, Bharanidharan Shanmugam, and Deepika Mathur (2022). "A review on current technologies and future direction of water leakage detection in water distribution network". In: *IEEE Access* 10, pp. 107177–107201.

Ismail, Shereen, Diana W Dawoud, Nadhem Ismail, Ronald Marsh, and Ali S Alshami (2022). "IoT-based water management systems: survey and future research direction". In: *IEEE Access* 10, pp. 35942–35952.

ISO/IEC (1996). "ISO/IEC 7498-1: 1994 information technology–open systems interconnection–basic reference model: The basic model". In: *International Standard ISOIEC* 74981, p. 59.

— (n.d.). *Information technology — Vocabulary*. Accessed 31 Jan. 2025. URL: https://www.iso.org/obp/ui/#iso:std:iso-iec:2382:ed-1:v2:en.

ISO/IEC/IEEE (2021). *Syntax and Semantics for IDEF1X97 (IDEFobject)*. URL: https://www.iso.org/standard/60614.html.

Ivchenko, A. V. (n.d.). *QoE-Assesment*. Accessed 17 May 2024. URL: https://github.com/AleksandrIvchenko/QoE-assesment.

Izza, S (2009). "Integration of industrial information systems: from syntactic to semantic integration approaches". In: *Enterprise Information Systems* 3.1, pp. 1–57. DOI: 10.1080/17517570802521163. eprint: https://doi.org/10.1080/17517570802521163. URL: https://doi.org/10.1080/17517570802521163.

Jennex, Murray and Summer Bartczak (2015). "A Revised Knowledge Pyramid". In: *International Journal of Knowledge Management* 9, pp. 19–30. DOI: 10.4018/ijkm.2013070102.

Jiang, Yu, Ariel Dinar, and Petra Hellegers (2018). "Economics of social trade-off: balancing wastewater treatment cost and ecosystem damage". In: *Journal of environmental management* 211, pp. 42–52.

Jin, Xiaolin, Shuwu Zhang, and Jie Liu (2018). "Word Semantic Similarity Calculation Based on Word2vec". In: *2018 International Conference on Control, Automation and Information Sciences (ICCAIS)*, pp. 12–16. DOI: 10.1109/ICCAIS.2018.8570612.

Jing, Liang, Bing Chen, Baiyu Zhang, and Xudong Ye (2018). "Modeling marine oily wastewater treatment by a probabilistic agent-based approach". In: *Marine Pollution Bulletin* 127, pp. 217–224.

Kalatzis, Nikos, George Routis, Yiorgos Marinellis, Marios Avgeris, Ioanna Roussaki, Symeon Papavassiliou, and Miltiades Anagnostou (2019). "Semantic interoperability for IoT platforms in support of decision making: An experiment on early wildfire detection". In: *Sensors (Switzerland)* 19.3. ISSN: 14248220. DOI: 10.3390/s19030528.

Kalbar, Pradip P, Subhankar Karmakar, and Shyam R Asolekar (2016). "Life cycle-based decision support tool for selection of wastewater treatment alternatives". In: *Journal of Cleaner Production* 117, pp. 64–72.

Kessler, Brett (1995). “Computational dialectology in irish gaelic”. In: *arXiv preprint cmp-lg/9503002*.

Kifer, Michael and Georg Lausen (1989). “F-Logic: A Higher-Order Language for Reasoning about Objects, Inheritance, and Scheme”. In: *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*. SIGMOD ’89. Portland, Oregon, USA: Association for Computing Machinery, pp. 134–146. ISBN: 0897913175. DOI: 10.1145/67544.66939. URL: https://doi.org/10.1145/67544.66939.

Krafzig, Dirk, Karl Banke, and Dirk Slama (2005). *Enterprise SOA: service-oriented architecture best practices*. Prentice Hall Professional.

Kyung, Daeseung, Minsun Kim, Jin Chang, and Woojin Lee (2015). “Estimation of greenhouse gas emissions from a hybrid wastewater treatment plant”. In: *Journal of Cleaner Production* 95, pp. 117–123.

Lalle, Yandja, Mohamed Fourati, Lamia Chaari Fourati, and João Paulo Barraca (2021). “Communication technologies for Smart Water Grid applications: Overview, opportunities, and research directions”. In: *Computer Networks* 190, p. 107940.

Lanthaler, Markus and Christian Gütl (2012). “On using JSON-LD to create evolvable RESTful services”. In: *Proceedings of the Third International Workshop on RESTful Design*, pp. 25–32.

Lee, Edward A (2008). “Cyber physical systems: Design challenges”. In: *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*. IEEE, pp. 363–369.

Li, Jiada, Xiafei Yang, and Robert Sitzenfrei (2020). “Rethinking the framework of smart water system: A review”. In: *Water* 12.2, p. 412.

Linked Open Data Cloud (2007). *The Linked Open Data Cloud*. accessed 22 Jan. 2021. URL: https://lod-cloud.net/.

Lorenzo-Toja, Yago, Carolina Alfonsín, María José Amores, Xavier Aldea, Desirée Marin, María Teresa Moreira, and Gumersindo Feijoo (2016). “Beyond the conventional life cycle inventory in wastewater treatment plants”. In: *Science of the Total Environment* 553, pp. 71–82.

Maedche, Alexander and Steffen Staab (2000). "Semi-automatic engineering of ontologies from text". In: *Proceedings of the 12th international conference on software engineering and knowledge engineering*. Citeseer, pp. 231–239.

Mannina, Giorgio, Taise Ferreira Rebouças, Alida Cosenza, Miquel Sànchez-Marrè, and Karina Gibert (2019). "Decision support systems (DSS) for wastewater treatment plants – A review of the state of the art". In: *Bioresource Technology* 290, p. 121814. ISSN: 0960-8524. DOI: https://doi.org/10.1016/j.biortech.2019.121814. URL: http://www.sciencedirect.com/science/article/pii/S0960852419310442.

Manso, Miguel-Ángel, Monica Wachowicz, and Miguel-Ángel Bernabé (2009). "Towards an integrated model of interoperability for spatial data infrastructures". In: vol. 13. 1. Wiley Online Library, pp. 43–67.

Margara, Alessandro, Jacopo Urbani, Frank [van Harmelen], and Henri Bal (2014). "Streaming the Web: Reasoning over dynamic data". In: *Journal of Web Semantics* 25, pp. 24–44. ISSN: 1570-8268. DOI: https://doi.org/10.1016/j.websem.2014.02.001. URL: http://www.sciencedirect.com/science/article/pii/S1570826814000067.

Matplotlib Development Team (2003). *Matplotlib: Visualization with Python*. accessed 8 Feb 2022. URL: https://matplotlib.org/.

Mazón, Jose-Norberto and Juan Trujillo (2008). "An MDA approach for the development of data warehouses". In: *Decision Support Systems* 45.1. Data Warehousing and OLAP, pp. 41–58. ISSN: 0167-9236. DOI: https://doi.org/10.1016/j.dss.2006.12.003. URL: https://www.sciencedirect.com/science/article/pii/S0167923606002077.

McGuinness, Deborah L and Paulo Pinheiro Da Silva (2004). "Explaining answers from the semantic web: The inference web approach". In: *Journal of Web Semantics* 1.4, pp. 397–413.

Meng, Z., Z. Wu, C. Muvianto, and J. Gray (2017). "A Data-Oriented M2M Messaging Mechanism for Industrial IoT Applications". In: *IEEE Internet of Things Journal* 4.1, pp. 236–246.

Microsoft (2015). *Visual Code Studio*. accessed 7 Feb 2022. URL: https://code.visualstudio.com/.

Microsoft (n.d.). *Power BI*. Accessed 10 June 2024. URL: https://www.microsoft.com/en-us/power-platform/products/power-bi.

Milis, Georgios M, Christos G Panayiotou, and Marios M Polycarpou (2017). "SEMIoTICS: Semantically enhanced IoT-enabled intelligent control systems". In: *IEEE Internet of Things Journal* 6.1, pp. 1257–1266.

Molinos-Senante, María, Trinidad Gómez, Manel Garrido-Baserba, Rafael Caballero, and Ramón Sala-Garrido (2014). "Assessing the sustainability of small wastewater treatment systems: A composite indicator approach". In: *Science of the total environment* 497, pp. 607–617.

Morera, Serni, Joaquim Comas, Manel Poch, and Lluís Corominas (2015). "Connection of neighboring wastewater treatment plants: economic and environmental assessment". In: *Journal of Cleaner Production* 90, pp. 34–42.

Morita, Takeshi, Naoki Fukuta, Noriaki Izumi, and Takahira Yamaguchi (2006). "DODDLE-OWL: a domain ontology construction tool with OWL". In: *Asian Semantic Web Conference*. Springer, pp. 537–551.

Murdock, Paul and Louay Bassbouss (2016). "Semantic Interoperability for the Web of Things Semantic Interoperability for the Web of Things 1 Insight Centre for Data Analytics". In: August. DOI: 10.13140/RG.2.2.25758.13122.

Nadiri, Ata Allah, Sima Shokri, Frank T-C Tsai, and Asghar Asghari Moghaddam (2018). "Prediction of effluent quality parameters of a wastewater treatment plant using a supervised committee fuzzy logic model". In: *Journal of cleaner production* 180, pp. 539–549.

Naqvi, Muhammad Raza, Muhammad Waseem Iqbal, Syed Khuram Shahzad, Iqra Tariq, Marium Malik, Faseeha Ehsan, Natash Ali Mian, and Nadia Tabassum (2020). "A concurrence study on interoperability issues in iot and decision making based model on data and services being used during inter-operability". In: *Lahore Garrison University Research Journal of Computer Science and Information Technology* 4.4, pp. 73–85.

Neches, Robert, Richard E Fikes, Tim Finin, Thomas Gruber, Ramesh Patil, Ted Senator, and William R Swartout (1991). "Enabling technology for knowledge sharing". In: *AI magazine* 12.3, pp. 36–36.

Newman, Peter (2018). *IoT Report: How Internet of Things Technology Is Now Reaching Mainstream Companies and Consumers*.

Noura, Mahda, Mohammed Atiquzzaman, and Martin Gaedke (2019). "Interoperability in internet of things: Taxonomies and open challenges". In: *Mobile Networks and Applications* 24.3, pp. 796–809.

Noy, Natalya Fridman, Monica Crubézy, Ray W Fergerson, Holger Knublauch, Samson W Tu, Jennifer Vendetti, and Mark A Musen (2003). "Protégé-2000: an open-source ontology-development and knowledge-acquisition environment: AMIA 2003 Open Source Expo". In: *AMIA Annual Symposium Proceedings*. Vol. 2003, p. 953.

Noy, Natalya Fridman and Deborah L McGuinness (2001). *Ontology development 101: A guide to creating your first ontology*.

Noy, Natalya Fridman and Mark A Musen (2000). "Algorithm and tool for automated ontology merging and alignment". In: *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00). Available as SMI technical report SMI-2000-0831*. Vol. 115. sn.

NumFOCUS Inc. (2020). *Pandas: Visualization with Python*. accessed 17 Dec 2022. URL: https://pandas.pydata.org//.

Oberascher, Martin, Wolfgang Rauch, and Robert Sitzenfrei (2022). "Towards a smart water city: A comprehensive review of applications, data requirements, and communication technologies for integrated management". In: *Sustainable Cities and Society* 76, p. 103442.

OGC (2016). accessed 24 Jan. 2024. URL: https://www.ogc.org/standards/citygml/.

— (2019). *Sensor Model Language (SensorML)*. accessed 1 Dec. 2022. URL: https://www.ogc.org/standards/sensorml.

OGC, W3C (2017). *Semantic Sensor Network Ontology*. accessed 9 Mar. 2022. URL: https://www.w3.org/TR/vocab-ssn/.

OGC, HDWG (2014). *WaterML2*. accessed 23 Jan.2021. URL: http://waterml2.org/.

OMG (2021). *Unified Modeling Language*. URL: https://www.omg.org/spec/UML/.

Oprea, Mihaela (2018). "A knowledge modelling framework for intelligent environmental decision support systems and its application to some environmental problems". In: *Environmental Modelling & Software* 110, pp. 72–94.

Ouali, Imene, Faiza Ghozzi, Raouia Taktak, and Mohamed Saifeddine Hadj Sassi (2019). "Ontology alignment using stable matching". In: *Procedia Computer Science* 159, pp. 746–755.

Pagano, Antonino, Domenico Garlisi, Ilenia Tinnirello, Fabrizio Giuliano, Giovanni Garbo, Mariana Falco, and Francesca Cuomo (2025). "A survey on massive IoT for water distribution systems: Challenges, simulation tools, and guidelines for large-scale deployment". In: *Ad Hoc Networks* 168, p. 103714. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2024.103714. URL: https://www.sciencedirect.com/science/article/pii/S1570870524003251.

Park, Jinsoo, Wonchin Cho, and Sangkyu Rho (2010). "Evaluating ontology extraction tools using a comprehensive evaluation framework". In: *Data & Knowledge Engineering* 69.10, pp. 1043–1061. ISSN: 0169-023X. DOI: https://doi.org/10.1016/j.datak.2010.07.002. URL: http://www.sciencedirect.com/science/article/pii/S0169023X10000844.

Pascual Pañach, Josep, Miquel Àngel Cugueró Escofet, Pere Aguiló Martos, and Miquel Sànchez-Marrè (2018). "An interoperable workflow-based framework for the automation of building intelligent process control systems". In: *9th International Congress on Environmental Modelling and Software: Ft. Collins, Colorado, USA, June 2018*. International Environmental Modelling and Software Society (iEMSs), pp. 1–8.

Pilar Angeles, Maria del and A Espino Gamez (2015). "Comparison of methods hamming distance, jaro, and monge-elkan". In: *DBKDA 2015*, p. 73.

Pintilie, Loredana, Carmen M Torres, Carmen Teodosiu, and Francesc Castells (2016). "Urban wastewater reclamation for industrial reuse: An LCA case study". In: *Journal of cleaner production* 139, pp. 1–14.

Portisch, Jan, Michael Hladik, and Heiko Paulheim (2022). "Background knowledge in ontology matching: A survey". In: *Semantic Web* Preprint, pp. 1–55.

Power, Daniel and Shashidhar Kaparthi (2001). "Building Web-based decision support systems". In: *Course Technology European Journal of Operational Research International Journal of Production Research Journal of Decision Systems* 11.

Princeton University (2021). *WordNet: A Lexical Database for English*. accessed 22 Jan. 2021. URL: https://wordnet.princeton.edu/publications.

Publications Office of the European Union (n.d.). *European data*. accessed 11 Apr. 2025. URL: data.europa.eu.

Python Software Foundation (2001). *Python difflib*. Accessed 8 June 2024. URL: https://docs.python.org/3/library/difflib.html.

Quiñones-Grueiro, Marcos, Alberto Prieto-Moreno, Cristina Verde, and Orestes Llanes-Santiago (2019). "Decision Support System for Cyber Attack Diagnosis in Smart Water Networks". In: *IFAC-PapersOnLine* 51.34. 2nd IFAC Conference on Cyber-Physical and Human Systems CPHS 2018, pp. 329–334. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2019.01.024. URL: http://www.sciencedirect.com/science/article/pii/S2405896319300266.

Rao, Anand S. and MICHAEL P. Georggeff (1998). "Decision Procedures for BDI Logics". In: *Journal of Logic and Computation* 8.3, pp. 293–343. ISSN: 0955-792X. DOI: 10.1093/logcom/8.3.293. eprint: https://academic.oup.com/logcom/article-pdf/8/3/293/9438541/8-3-293.pdf. URL: https://doi.org/10.1093/logcom/8.3.293.

Rasekh, Amin, Amin Hassanzadeh, Shaan Mulchandani, Shimon Modi, and M Katherine Banks (2016). *Smart water networks and cyber security*.

RDFLib Team (2009). *RDFLib*. accessed 8 Feb 2022. URL: https://rdflib.readthedocs.io/en/stable/.

Řehůřek, Radim (2009). *Gensim*. accessed 2 Mar 2022. URL: https://radimrehurek.com/gensim/index.html.

Reynolds, L (2013). "The SWIM concept: an open interoperable data standard". In: *IET Conference Proceedings*. The Institution of Engineering & Technology.

Roberts, Bobby (2020). *Integration Vs Interoperability: What'S The Difference?* URL: https://blog.sisfirst.com/integration-v-interoperability-what-is-the-difference.

Rowley, Jennifer (2007). "The wisdom hierarchy: representations of the DIKW hierarchy". In: *Journal of Information Science* 33.2, pp. 163–180. DOI: 10.1177/0165551506070706. eprint: https://doi.org/10.1177/0165551506070706. URL: https://doi.org/10.1177/016555150607070.

Saagi, Ramesh, Xavier Flores-Alsina, Guangtao Fu, David Butler, Krist V Gernaey, and Ulf Jeppsson (2016). "Catchment & sewer network simulation model to benchmark control strategies within urban wastewater systems”. In: *Environmental Modelling & Software* 78, pp. 16–30.

Schwab, Klaus (2017). *The fourth industrial revolution*. Currency.

Sensus (2019). *Water 20/20: Bringing Smart Water Networks Into Focus*. accessed: 10 Feb. 2025. URL: https://smartwaternetworks.com/.

Serrat-Capdevila, Aleix, Juan B Valdes, and Hoshin V Gupta (2011). “Decision support systems in water resources planning and management: stakeholder participation and the sustainable path to science-based decision making”. In: *Efficient decision support systems—Practice and challenges from current to future*, pp. 423–440.

Singh, Mandeep (2021). *Kaa Cloud IoT Sensor Data*. accessed 9 Jun. 2023. URL: https://www.kaaiot.com/.

— (2023). *Semantic Similarity Scoring Ontology*. accessed 9 Jan. 2023. URL: https://github.com/mxrandhawa/s3o.

— (2024). *Data and Information Interoperability Questionnaire (DIIQ) Personal*. accessed: 27 Dec. 2024. URL: https://forms.office.com/e/zJ15GTEE3Y.

— (2025). *Data Information Interoperability Framework*. accessed 9 Mar. 2025. URL: https://gitlab.com/phd30/DIIM/.

Singh, Mandeep, Moatasim Mahmoud, Rizou Stamatia, Zaharias D. Zaharis, Pavlos I. Lazaridis, Vladimir K. Poulkov, and Wenyan Wu (2024). “Towards 5G/6G Data Harmonization through NLP and Semantic Web Technologies”. In: *2024 Advanced Topics on Measurement and Simulation (ATOMS)*, pp. 291–294. DOI: 10.1109/ATOMS60779.2024.10921618.

Singh, Mandeep, Edlira Vakaj, Stamatia Rizou, and Wenyan Wu (2023). “Towards aligning IoT data with domain-specific ontologies through Semantic Web technologies and NLP”. In: *SEMANTICS 2023 EU*.

Singh, Mandeep, Wenyan Wu, Stamatia Rizou, and Edlira Vakaj (2022). “Data information interoperability model for IoT-enabled smart water networks”. In: *2022 IEEE 16th International Conference on Semantic Computing (ICSC)*. IEEE, pp. 179–186. DOI: 10.1109/ICSC52841.2022.00038.

Singh, Pratima and Arun Kansal (2018). “Energy and GHG accounting for wastewater infrastructure”. In: *Resources, Conservation and Recycling* 128, pp. 499–507.

Sirin, Evren, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz (2007). “Pellet: A practical owl-dl reasoner”. In: *Journal of Web Semantics* 5.2, pp. 51–53.

Soldatos, John, Nikos Kefalakis, Manfred Hauswirth, Martin Serrano, Jean-Paul Calbimonte, Mehdi Riahi, Karl Aberer, Prem Prakash Jayaraman, Arkady Zaslavsky, Ivana Podnar Žarko, et al. (2015). “Openiot: Open source internet-of-things in the cloud”. In: *Interoperability and open-source solutions for the internet of things*. Springer, pp. 13–25.

Sottara, Davide (2014). *OntoPlant-Measures ontology*. accessed: 10 Feb. 2025. URL: https://github.com/WWTP/OntoPlant-Measures.

Sottara, Davide, Jean Claude Coreale, Thierry Spetebroot, Dalila Pulcini, Daniele Giunchi, Fabrizio Paolucci, and Luca Luccarini (2014). *An ontology-based approach for the instrumentation, control and automation infrastructure of a WWTP*. URL: https://www.academia.edu/download/53892589/An_ontology-based_approach_for_the_instr20170718-3232-1pm8ts1.pdf.

Sun, Chenghao (2020). “Research on investment decision-making model from the perspective of “Internet of Things + Big data””. In: *Future Generation Computer Systems* 107, pp. 286–292. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2020.02.003. URL: http://www.sciencedirect.com/science/article/pii/S0167739X19321387.

SWAN (2016). *Communication in Smart Water Networks*. accessed: 20 Mar. 2022. URL: https://swan-forum.com/publications/.

— (n.d.). *WASTEWATER NETWORK MANAGEMENT*. accessed: 10 Feb. 2022. URL: https://www.swan-tool.com/wastewater-network-management.

Swartz, Aaron (2013). “Aaron Swartz’s A Programmable Web: An Unfinished Work”. In: *Synthesis Lectures on the Semantic Web: Theory and Technology* 3.2, pp. 1–64.

Szmeja, Paweł (2018). *Generic Ontology for IoT Platforms*. URL: https://inter-iot.github.io/ontology/.

Szmeja, Paweł, Maria Ganzha, Marcin Paprzycki, Wiesław Pawłowski, and Katarzyna Wasielewska (2018). “Declarative ontology alignment format for semantic translation”. In: *2018 3rd Inter-*

national Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU). IEEE, pp. 1–6.

Tao, Cui and David W. Embley (2009). “Automatic hidden-web table interpretation, conceptualization, and semantic annotation”. In: *Data & Knowledge Engineering* 68.7. Special Issue: 26th International Conference on Conceptual Modeling (ER 2007) – Six selected and extended papers, pp. 683–703. ISSN: 0169-023X. DOI: https://doi.org/10.1016/j.datak.2009.02.010. URL: http://www.sciencedirect.com/science/article/pii/S0169023X09000172.

Tarboton, David G, David Maidment, Ilya Zaslavsky, Daniel P Ames, Jon Goodall, and Jeffery S Horsburgh (2010). “Cuahsi hydrologic information system 2010 status report”. In: *Consortium of Universities for the Advancement of Hydrologic Science, Inc* 34.

Tolk, Andreas and James Muguira (2003). “The Levels of Conceptual Interoperability Model”. In: *Fall Simulation Interoperability Workshop* September, pp. 1–9.

Torregrossa, Dario, F Hernández-Sancho, J Hansen, A Cornelissen, T Popov, and G Schutz (2017). “Energy saving in wastewater treatment plants: A plant-generic cooperative decision support system”. In: *Journal of cleaner production* 167, pp. 601–609.

Torregrossa, Dario, Antonino Marvuglia, and Ulrich Leopold (2018). “A novel methodology based on LCA+ DEA to detect eco-efficiency shifts in wastewater treatment plants”. In: *Ecological indicators* 94, pp. 7–15.

Turnitsa, Charles (2005). “Extending the levels of conceptual interoperability model”. In: *Proceedings IEEE summer computer simulation conference, IEEE CS Press*.

Turnitsa, Charles and Andreas Tolk (2008). “Knowledge representation and the dimensions of a multi-model relationship”. In: *2008 Winter Simulation Conference*. IEEE, pp. 1148–1156.

Tutte, William Thomas (2001). *Graph theory*. Vol. 21. Cambridge University Press.

Urbani, Dominique and Marielle Delhom (2006). “Water Management Using a New Hybrid Multi-Agents System - Geographic Information System Decision Support System Framework”. In: pp. 314–319. DOI: 10.1109/ISEIMA.2006.344967.

Ushold, Mike and Christopher Menzel (2005). *Semantic Integration & Interoperability Using RDF and OWL*. URL: https://www.w3.org/2001/sw/BestPractices/OEP/SemInt/.

Varas, Gabriel Anzaldi (2014). “Water Enhanced Resource Planning “Where water supply meets demand””. In: URL: https://cordis.europa.eu/docs/projects/cnect/3/318603/080/deliverables/001-D13GenericOntologyforwatersupplydistributionchainv13.pdf.

Velayudhan, Nibi Kulangara, Preeja Pradeep, Sethuraman N Rao, Aryadevi Remanidevi Devidas, and Maneesha Vinodini Ramesh (2022). “IoT-enabled water distribution systems—A comparative technological review”. In: *IEEE Access* 10, pp. 101042–101070.

Vilches-Blázquez, LM, JA Ramos, FJ López-Pellicer, O Corcho, and J Nogueras-Iso (2015). *hydrOntology a global ontology of the hydrographical domain*.

W3C (2004). *OWL-S: Semantic Markup for Web Services*. accessed 9 Mar. 2024. URL: https://www.w3.org/submissions/OWL-S/.

— (2007). *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language*. accessed 9 Mar. 2024. URL: https://www.w3.org/TR/2007/REC-wsdl20-20070626/.

— (2013a). *Semantic Web: OWL*. accessed 9 Mar. 2022. URL: https://www.w3.org/OWL/.

— (2013b). *Semantic Web: RDF/XML*. accessed 9 Mar. 2022. URL: https://www.w3.org/TR/rdf-syntax-grammar/.

— (2014a). *Semantic Web: N-Quads*. accessed 9 Mar. 2022. URL: https://www.w3.org/TR/n-quads/.

— (2014b). *Semantic Web: RDFS*. accessed 9 Mar. 2022. URL: https://www.w3.org/TR/rdf-schema/.

— (2015). *Semantic Web*. accessed 9 Mar. 2022. URL: http://www.w3.org/standards/semanticweb/.

— (2016). *Sensor-Observation-Sampling-Actuator ontology*. accessed 19 August 2022. URL: https://www.w3.org/2015/spatial/wiki/SOSA_Ontology.

— (2020). *World Wide Web Consortium (W3C): OWL Implementations*. accessed 11 Jan 2021. URL: https://www.w3.org/2001/sw/wiki/OWL/Implementations.

— (2023). *Web of Things (WoT) Discovery*. accessed 11 Jan 2025. URL: https://www.w3.org/TR/wot-discovery/.

— (n.d.). *ConverterToRdf*. accessed 20 Oct 2021. URL: https://www.w3.org/wiki/ConverterToRdf.

Walker, G, P Taylor, S Cox, P Sheahan, R Anderssen, R Braddock, and L Newham (2009). "Water Data Transfer Format (WDTF): Guiding principles, technical challenges and the future". In: *Proc. 18th World IMACS Congress and MODSIM09 Int. Congress on Modelling and Simulation*, pp. 4381–4387.

Walter, Tobias, Fernando Silva Parreiras, and Steffen Staab (2014). "An ontology-based framework for domain-specific modeling". In: *Software & Systems Modeling* 13.1, pp. 83–108.

Wang, Zhou and et al. (2004). "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4, pp. 600–612.

Wassermann, E. and A. Fay (2017). "Interoperability rules for heterogenous multi-agent systems: Levels of conceptual interoperability model applied for multi-agent systems". In: *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, pp. 89–95. DOI: 10.1109/INDIN.2017.8104752.

WHO (2021). *Guidelines for drinking-water quality (GDWQ)*. accessed 10 Jan. 2022. URL: https://www.who.int/publications-detail-redirect/9789240045064.

Winikoff, M, L Padgham, J Harland, and J Thangarajah (2002). "Declarative and procedural goals in intelligent agent systems". In: *International Conference on Principles of Knowledge Representation and Reasoning 22-25 April 2005*.

Wu, Zheng Yi, Mahmoud El-Maghraby, and Sudipta Pathak (2015). "Applications of deep learning for smart water networks". In: *Procedia Engineering* 119, pp. 479–485.

Xin, Chunhua, Min M Addy, Jinyu Zhao, Yanling Cheng, Yiwei Ma, Shiyu Liu, Dongyan Mu, Yuhuan Liu, Paul Chen, and Roger Ruan (2018). "Waste-to-biofuel integrated system and its comprehensive techno-economic assessment in wastewater treatment plants". In: *Bioresource Technology* 250, pp. 523–531.

Ye, Xudong, Bing Chen, Rune Storesund, and Baiyu Zhang (2021). "System control and optimization in wastewater treatment: a particle swarm optimization (PSO) approach". In: *Soft computing techniques in solid waste and wastewater management*. Elsevier, pp. 393–407.

Yoshida, Hiroko, Julie Clavreul, Charlotte Scheutz, and Thomas H Christensen (2014). "Influence of data collection schemes on the Life Cycle Assessment of a municipal wastewater treatment plant". In: *Water research* 56, pp. 292–303.

Yu, Jonathan, Peter Taylor, Simon J.D. Cox, and Gavin Walker (2015). “Validating observation data in WaterML 2.0”. In: *Computers & Geosciences* 82, pp. 98–110. ISSN: 0098-3004. DOI: https://doi.org/10.1016/j.cageo.2015.06.001. URL: http://www.sciencedirect.com/science/article/pii/S0098300415001326.

Yuan, Zhiguo, Gustaf Olsson, Rachel Cardell-Oliver, Kim van Schagen, Angela Marchi, Ana Deletic, Christian Urich, Wolfgang Rauch, Yanchen Liu, and Guangming Jiang (2019). “Sweating the assets–the role of instrumentation, control and automation in urban water systems”. In: *Water research* 155, pp. 381–402.

Zarli, Alain, Yacine Rezgui, Daniela Belziti, and Elenia Duce (2014). “Water analytics and intelligent sensing for demand optimised management: the wisdom vision and approach”. In: *Procedia Engineering* 89, pp. 1050–1057.

Zeng, Siyu, Xing Chen, Xin Dong, and Yi Liu (2017). “Efficiency assessment of urban wastewater treatment plants in China: Considering greenhouse gas emissions”. In: *Resources, Conservation and Recycling* 120, pp. 157–165.

Zhen-Xing, Wu and Tian Xing-Yan (2015). “Research of Ontology Merging Based on Concept Similarity”. English. In: pp. 831–834.

Appendix A

Ethical Considerations

Since ethical approval is essential to a research project at Birmingham City University (BCU), I have applied for it and declared all relevant information in the application form. Throughout the duration of my research, I have adhered to all rules and policies established by BCU for researchers. My research study involves reading, writing, and simulating with computers, so I spent much time sitting on computers and watching computer screens. This posed risks to my health and mental well-being. To avoid this risk, I took regular breaks, changed my place of reading and writing, and found a balance between work and life.

Ethical, health, and safety concerns were considered before planning and carrying out activities such as visiting project collaborators and supervisors, travelling, and attending conferences, meetings, workshops, and work sites.

In my research, I have used data and information generated by humans and machines. I have always complied with data protection laws and sought the creator's consent to address copyright issues.

Appendix B

Data Information Interoperability Ontology (DIIO)

All DIIF Ph.d.-relevant data and source code are available under the URL https://gitlab.com/phd30/DIIM.

source-code

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.co-ode.org/ontologies/diio"
   xml:base="http://www.co-ode.org/ontologies/diio"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xml="http://www.w3.org/XML/1998/namespace"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:diio="http://www.co-ode.org/ontologies/diio#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:swrla="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#">
  <owl:Ontology rdf:about="http://www.co-ode.org/ontologies/diio">
    <owl:versionIRI rdf:resource="http://www.co-ode.org/ontologies/diio/0.1.0"/>
    <rdfs:comment xml:lang="en">Data and Information Interoperability Ontology DIIO</rdfs:comment>
  </owl:Ontology>
```

```
<!--
///////////////////////////////////////////////////////////////////////////////////////
//
// Annotation properties
//
///////////////////////////////////////////////////////////////////////////////////////
 -->




<!-- http://swrl.stanford.edu/ontologies/3.3/swrla.owl#isRuleEnabled -->

<owl:AnnotationProperty rdf:about="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#isRuleEnabled"/>



<!--
///////////////////////////////////////////////////////////////////////////////////////
//
// Object Properties
//
///////////////////////////////////////////////////////////////////////////////////////
 -->




<!-- http://www.co-ode.org/ontologies/diio#can_adapt_Ontology -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#can_adapt_Ontology">
```

```
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#SymmetricProperty"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#can_adapt_Serialisation_Format -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#can_adapt_Serialisation_Format">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#SymmetricProperty"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#can_adapt_Standard -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#can_adapt_Standard">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#SymmetricProperty"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#has_Serialisation_Format -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#has_Serialisation_Format">
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
</owl:ObjectProperty>
```

```
<!-- http://www.co-ode.org/ontologies/diio#is_aligned_with_Ontology -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#is_aligned_with_Ontology">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#SymmetricProperty"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
  <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
  <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#is_aligned_with_Standard -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#is_aligned_with_Standard">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#SymmetricProperty"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
  <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
  <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#supports_Ontology -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#supports_Ontology">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
  <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
  <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
</owl:ObjectProperty>
```

```
<!-- http://www.co-ode.org/ontologies/diio#supports_Serialisation_Format -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#supports_Serialisation_Format">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#supports_Standard -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#supports_Standard">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#uses_File_Format_Standard -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#uses_File_Format_Standard">
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#uses_Ontology -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#uses_Ontology">
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#IoT_Data"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
</owl:ObjectProperty>



<!-- http://www.co-ode.org/ontologies/diio#uses_Standard -->

<owl:ObjectProperty rdf:about="http://www.co-ode.org/ontologies/diio#uses_Standard">
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#IoT_Data"/>
    <rdfs:range rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
</owl:ObjectProperty>



<!--
///////////////////////////////////////////////////////////////////////////////////////
//
// Data properties
//
///////////////////////////////////////////////////////////////////////////////////////
 -->




<!-- http://www.co-ode.org/ontologies/diio#application_domain -->

<owl:DatatypeProperty rdf:about="http://www.co-ode.org/ontologies/diio#application_domain">
    <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
    <rdfs:comment xml:lang="en">An application domain is the area of reality that a document is designed
    ↪   and composed for, such as a water management or monintoring systems.</rdfs:comment>
</owl:DatatypeProperty>
```

```
<!-- http://www.co-ode.org/ontologies/diio#file_extension -->

<owl:DatatypeProperty rdf:about="http://www.co-ode.org/ontologies/diio#file_extension">
  <rdfs:domain rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</owl:DatatypeProperty>



<!-- http://www.co-ode.org/ontologies/diio#uri -->

<owl:DatatypeProperty rdf:about="http://www.co-ode.org/ontologies/diio#uri">
  <rdfs:domain rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#anyURI"/>
  <rdfs:comment xml:lang="en">A Uniform Resource Identifier URI is a string that identifies resources on
  ↪  the Semantic Web. URIs are used to name and address documents, objects, concepts, and
  ↪  more.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/cooluris/</rdfs:seeAlso>
</owl:DatatypeProperty>



<!--
///////////////////////////////////////////////////////////////////////////////////////
//
// Classes
//
///////////////////////////////////////////////////////////////////////////////////////
 -->
```

```
<!-- http://www.co-ode.org/ontologies/diio#Application -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Application">
  <rdfs:comment xml:lang="en">In computer science, an application is a software program that performs a
  ↪  specific task for a user.</rdfs:comment>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Character_Encoding -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Character_Encoding">
  <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
  <rdfs:comment>Character encoding is the process of assigning numbers to graphical characters,
  ↪  especially the written characters of human language, allowing them to be stored, transmitted, and
  ↪  transformed using computers.</rdfs:comment>
  <rdfs:seeAlso>https://www.w3.org/International/articles/definitions-characters/</rdfs:seeAlso>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Converter -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Converter">
  <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Application"/>
  <rdfs:comment xml:lang="en">A converter is a device or tool that converts one form of signal, data, or
  ↪  energy into another</rdfs:comment>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Converter_To_RDF -->
```

```
<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Converter_To_RDF">
    <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Converter"/>
    <rdfs:comment xml:lang="en">A Converter to RDF is a tool which converts application data from an
    ↪   application-specific format into RDF for use with RDF tools and integration with other data.
    ↪   Converters may be part of a one-time migration effort, or part of a running system which provides a
    ↪   semantic web view of a given application.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/wiki/ConverterToRdf</rdfs:seeAlso>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Data_File_Format -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Data_File_Format">
    <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
    <rdfs:comment xml:lang="en">Compute systems store data in binary form, called a file. However, the
    ↪   actual raw data within a file are stored in a strutured way. A file format standard contains a complete
    ↪   definition of both data structure and content.</rdfs:comment>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format">
    <rdfs:comment xml:lang="en">Data serialization is the process of writing the state of an object to a
    ↪   stream, and is the process of rebuilding the stream back into an object.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://dl.acm.org/doi/10.1145/944579.944589</rdfs:seeAlso>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Document -->
```

```
<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Document">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:comment xml:lang="en">In computer science, a &quot;document&quot; refers to a digital file
  ↪ containing primarily textual information, including its structure and formatting like fonts, colors, and
  ↪ images, essentially an electronic version of a traditional paper document that can be created, stored,
  ↪ and accessed on a computer system using word processing applications.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.computerhope.com/jargon/d/document.htm</rdfs:seeAlso>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Graph_Database -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Graph_Database">
  <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Application"/>
  <rdfs:comment xml:lang="en">A graph database application in computer science is a software program
  ↪ that stores, queries, and modifies data in a graph structure. Graph databases are used to model and
  ↪ store data that has complex relationships, such as those in social networks.</rdfs:comment>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#IoT_Data -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#IoT_Data">
  <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
  <rdfs:comment xml:lang="en">The Internet of Things IoT refers to a network of physical devices,
  ↪ vehicles, appliances, and other physical objects that are embedded with sensors, software, and
  ↪ network connectivity, allowing them to collect and share data.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.ibm.com/think/topics/internet-of-things</rdfs:seeAlso>
</owl:Class>
```

```
<!-- http://www.co-ode.org/ontologies/diio#Ontology -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Ontology">
   <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
   <rdfs:comment xml:lang="en">The W3C defines an ontology as a formal specification of knowledge, or
   ↪  conceptualization, that uses the Web Ontology Language OWL. Ontologies are used to describe the
   ↪  structure of data, including classes, properties, and relationships</rdfs:comment>
</owl:Class>



<!-- http://www.co-ode.org/ontologies/diio#Standard -->

<owl:Class rdf:about="http://www.co-ode.org/ontologies/diio#Standard">
   <rdfs:subClassOf rdf:resource="http://www.co-ode.org/ontologies/diio#Document"/>
   <rdfs:comment xml:lang="en">A computer science standard is a set of guidelines that allow different
   ↪  hardware and software to work together. Standards help ensure compatibility, interoperability, and
   ↪  safety</rdfs:comment>
</owl:Class>



<!--
///////////////////////////////////////////////////////////////////////////////////////
//
// Individuals
//
///////////////////////////////////////////////////////////////////////////////////////
 -->



<!-- http://www.co-ode.org/ontologies/diio#AHGF -->
```

```
    <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#AHGF">
        <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
        <diio:application_domain xml:lang="en">Hydrological Geospatial Fabric</diio:application_domain>
        <rdfs:comment xml:lang="en">Australian Hydrological Geospatial Fabric AHGF
Australian Hydrological Geospatial Fabric Geofabric Product Guide</rdfs:comment>
        <rdfs:seeAlso
        ↪   xml:lang="en">http://www.bom.gov.au/water/geofabric/documents/v3_0/ahgf_productguide_V3_0_release.pdf</rdfs:seeA
    </owl:NamedIndividual>



    <!-- http://www.co-ode.org/ontologies/diio#BinaryRDF_Format -->

    <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#BinaryRDF_Format">
        <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
        <diio:file_extension xml:lang="en">.brf</diio:file_extension>
        <rdfs:comment xml:lang="en">Binary RDF Representation for Publication and Exchange HDT. RDF HDT
        ↪   Header-Dictionary-Triples is a binary format for publishing and exchanging RDF data at large scale.
        ↪   RDF HDT represents RDF in a compact manner, natively supporting splitting huge RDF graphs into
        ↪   several chunks. It is designed to allow high compression rates.</rdfs:comment>
        <rdfs:seeAlso
        ↪   xml:lang="en">https://www.w3.org/submissions/2011/SUBM-HDT-20110330/</rdfs:seeAlso>
    </owl:NamedIndividual>



    <!-- http://www.co-ode.org/ontologies/diio#Bristol_River_water_quality -->

    <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#Bristol_River_water_quality">
        <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#IoT_Data"/>
        <diio:has_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#CSV_Format"/>
        <diio:application_domain xml:lang="en">water quality monitoring</diio:application_domain>
    </owl:NamedIndividual>
```

```
<!-- http://www.co-ode.org/ontologies/diio#CSV -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#CSV">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
  <diio:application_domain xml:lang="en">data exchange, data analysis, data migration and data
  ↪  storage/backup</diio:application_domain>
  <rdfs:comment xml:lang="en">Comma-separated values CSV is a text file format that uses commas to
  ↪  separate values, and newlines to separate records. A CSV file stores tabular data numbers and text
  ↪  in plain text, where each line of the file typically represents one data record. Each record consists of
  ↪  the same number of fields, and these are separated by commas in the CSV file. If the field delimiter
  ↪  itself may appear within a field, fields can be surrounded with quotation marks. RFC 4180 proposes
  ↪  a specification for the CSV format;</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://datatracker.ietf.org/doc/html/rfc4180</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#CSV_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#CSV_Format">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <diio:file_extension>.csv</diio:file_extension>
  <rdfs:comment xml:lang="en">Comma-separated values CSV is a text file format that uses commas to
  ↪  separate values, and newlines to separate records.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://en.wikipedia.org/wiki/Comma-separated_values</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#CityGML -->
```

```
<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#CityGML">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#XML_Format"/>
   <diio:application_domain xml:lang="en">Utility networks in 3D city models</diio:application_domain>
   <rdfs:comment xml:lang="en">The CityGML 3.0 Conceptual Model Standard describes a common
   ↪  semantic information model for the representation of 3D urban objects. The primary function of the
   ↪  model is to define the human interpretation of modelled data objects as well as their geometric
   ↪  representation and relationships.</rdfs:comment>
   <rdfs:seeAlso xml:lang="en">https://www.ogc.org/publications/standard/citygml/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#GOIoTP -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#GOIoTP">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
   <diio:supports_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#SSN"/>
   <diio:supports_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#SSN_SOSA"/>
   <diio:application_domain xml:lang="en">IoT artefacts platforms, devices, services,
   ↪  etc</diio:application_domain>
   <rdfs:comment xml:lang="en">GOIoTP Paweł Szmeja, 2018 is developed with OWL in 2018 as part of
   ↪  the INTER-IoT project Szmeja et al., 2018; it offers modular data structures for the description of
   ↪  entities most commonly appearing in IoT in the context of interoperating various IoT artefacts
   ↪  platforms, devices, services, etc.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#GraphDB -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#GraphDB">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Graph_Database"/>
   <diio:supports_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#RDF"/>
```

```
    <rdfs:comment xml:lang="en">Ontotext GraphDB is a highly efficient, scalable and robust graph
    ↪   database with RDF and SPARQL support. With excellent enterprise features, integration with
    ↪   external search applications, compatibility with industry standards, and both community and
    ↪   commercial support, GraphDB is the preferred database choice of both small independent
    ↪   developers and big enterprises.

GraphDB supports multiple RDF formats for importing or exporting data. All RDF formats have at least one file
↪   extension and MIME type that identify the format. Where multiple file extensions or MIME types are
↪   available, the preferred file extension or MIME type is listed first.</rdfs:comment>
    <rdfs:seeAlso
    ↪   xml:lang="en">https://graphdb.ontotext.com/documentation/10.8/index.html</rdfs:seeAlso>
  </owl:NamedIndividual>



  <!-- http://www.co-ode.org/ontologies/diio#HDF5 -->

  <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#HDF5">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
    <rdfs:comment xml:lang="en">ISO/TS 10303-26:2011 specifies a binary representation of
    ↪   EXPRESS-driven data using the Hierarchical Data Format Version 5 HDF5.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.iso.org/standard/50029.html</rdfs:seeAlso>
  </owl:NamedIndividual>



  <!-- http://www.co-ode.org/ontologies/diio#HY_Features -->

  <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#HY_Features">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
    <diio:can_adapt_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#INSPIRE"/>
    <diio:can_adapt_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#AHGF"/>
    <diio:can_adapt_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#NHD_Plus"/>
    <diio:can_adapt_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#NHN"/>
```

```
        <diio:can_adapt_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#SANDRE"/>
        <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#XML_Format"/>
        <diio:supports_Serialisation_Format
        ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#RDF/XML_Format"/>
        <diio:application_domain xml:lang="en">Surface hydrologic features</diio:application_domain>
        <rdfs:comment xml:lang="en">The OGC Surface Hydrology Features HY_Features standard defines a
        ↪   common conceptual information model for identification of specific hydrologic features independent
        ↪   of their geometric representation and scale. The model describes types of surface hydrologic
        ↪   features by defining fundamental relationships among various components of the hydrosphere. This
        ↪   includes relationships such as hierarchies of catchments, segmentation of rivers and lakes, and the
        ↪   hydrologically determined topological connectivity of features such as catchments and waterbodies.
        ↪   The standard also defines normative requirements for HY_Features implementation schemas and
        ↪   mappings to meet in order to be conformant with the conceptual model.</rdfs:comment>
        <rdfs:seeAlso xml:lang="en">https://docs.ogc.org/is/14-111r6/14-111r6.html</rdfs:seeAlso>
    </owl:NamedIndividual>



    <!-- http://www.co-ode.org/ontologies/diio#Hydrologic_Ontology_for_Discovery -->

    <owl:NamedIndividual
    ↪   rdf:about="http://www.co-ode.org/ontologies/diio#Hydrologic_Ontology_for_Discovery">
        <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
        <diio:application_domain xml:lang="en">Water</diio:application_domain>
        <rdfs:comment xml:lang="en">It was developed by Consortium for the Advancement of
Hydrological Sciences Inc. CUAHSI with OWL in 2010 to
support the discovery of time-series hydrologic data collected
at a fixed point. It is a precursor of WaterML2.</rdfs:comment>
    </owl:NamedIndividual>



    <!-- http://www.co-ode.org/ontologies/diio#HyperCat -->
```

```
<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#HyperCat">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
  <diio:supports_Serialisation_Format
  ↪  rdf:resource="http://www.co-ode.org/ontologies/diio#JSON_Format"/>
  <diio:application_domain xml:lang="en">Internet of things IoT</diio:application_domain>
  <rdfs:comment xml:lang="en">Hypercat is an open, JSON-based standard for the Internet of Things IoT
  ↪  that enables devices to share information. It&apos;s designed to help devices understand each
  ↪  other&apos;s data, regardless of location, manufacturer, or format</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://iot.ieee.org/articles-publications/newsletter/january-2016/hypercat-
  ↪  resource-discovery-on-the-internet-of-things.html</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#IFC-HDF -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC-HDF">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#HDF5"/>
  <diio:file_extension xml:lang="en">.ifcHDF</diio:file_extension>
  <rdfs:comment xml:lang="en">ifcHDF uses HDF and is based on the ISO 10303-26 standard for STEP
  ↪  data representation in HDF</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#IFC-JSON -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC-JSON">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#JSON"/>
  <diio:file_extension xml:lang="en">.ifcJSON</diio:file_extension>
  <rdfs:comment xml:lang="en">ifcJSON uses JSON, a modern format often used by web
  ↪  applications</rdfs:comment>
```

```
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#IFC-RDF -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC-RDF">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
   <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#RDF"/>
   <diio:file_extension xml:lang="en">.ifcOWL</diio:file_extension>
   <rdfs:comment xml:lang="en">IFC-RDF is a XML-based semantic data format that uses RDF and is
   ↪  expressed in the ifcOWL ontology.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#IFC-SPF -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC-SPF">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
   <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#STEP"/>
   <diio:file_extension xml:lang="en">.ifc</diio:file_extension>
   <rdfs:comment xml:lang="en">IFC-SPF is a text format defined by ISO 10303-21 &quot;STEP-File&quot;,
   ↪  where each line typically consists of a single object record, and having file extension
   ↪  &quot;.ifc&quot;. This is the most widely used IFC format, having the advantage of compact size yet
   ↪  readable text.</rdfs:comment>
   <rdfs:seeAlso xml:lang="en">https://en.wikipedia.org/wiki/Industry_Foundation_Classes</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#IFC-Turtule -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC-Turtule">
```

```
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#RDF"/>
  <diio:file_extension xml:lang="en">.ifcOWL</diio:file_extension>
  <rdfs:comment xml:lang="en">IFC-Turtle Terse RDF Triple Language is a textual semantic data format
  ↪   that uses RDF and is expressed in the ifcOWL ontology.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#IFC-XML -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC-XML">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#XML"/>
  <diio:file_extension xml:lang="en">.ifcXML</diio:file_extension>
  <rdfs:comment xml:lang="en">IFC-XML is an XML format defined by ISO 10303-28
  ↪   &quot;STEP-XML&quot;, having file extension &quot;.ifcXML&quot;. This format is suitable for
  ↪   interoperability with XML tools and exchanging partial building models. Due to the large size of
  ↪   typical building models, this format is less common in practice.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#IFC-ZIP -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC-ZIP">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <diio:file_extension xml:lang="en">.ifcZIP</diio:file_extension>
  <rdfs:comment xml:lang="en">IFC-ZIP is a ZIP compressed format consisting of an embedded IFC-SPF
  ↪   file or IFC-XML file and having file extension &quot;.ifcZIP&quot;.</rdfs:comment>
</owl:NamedIndividual>
```

```
<!-- http://www.co-ode.org/ontologies/diio#IFC4 -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#IFC4">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#IFC-HDF"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#IFC-JSON"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#IFC-RDF"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#IFC-SPF"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#IFC-Turtule"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#IFC-XML"/>
   <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#IFC-ZIP"/>
   <diio:application_domain xml:lang="en">Digital model of a building or other facility such as a bridge,
   ↪  highway, tunnel and so on</diio:application_domain>
   <rdfs:comment xml:lang="en">Industry Foundation Classes IFC are the open and neutral data format for
   ↪  openBIM. They are the international standard for building information modelling used for sharing and
   ↪  exchanging construction and facility management data across different software
   ↪  applications.</rdfs:comment>
   <rdfs:seeAlso xml:lang="en">https://www.designingbuildings.co.uk/wiki/IFC4</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#INSPIRE -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#INSPIRE">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
   <rdfs:comment xml:lang="en">Infrastructure for Spatial Information in Europe INSPIRE</rdfs:comment>
   <rdfs:seeAlso xml:lang="en">https://inspire.ec.europa.eu/glossary/Ontology</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#JSON -->
```

```
<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#JSON">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
  <diio:has_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#JSON_Format"/>
  <diio:application_domain xml:lang="en">Internet</diio:application_domain>
  <rdfs:comment xml:lang="en">JSON JavaScript Object Notation is a lightweight data-interchange format.
  ↪  It is easy for humans to read and write. It is easy for machines to parse and
  ↪  generate.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.json.org/json-en.html</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#JSON-LD -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#JSON-LD">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
  <diio:has_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#JSON-LD_Format"/>
  <diio:supports_Serialisation_Format
  ↪  rdf:resource="http://www.co-ode.org/ontologies/diio#JSON_Format"/>
  <rdfs:comment xml:lang="en">JSON-LD is a lightweight Linked Data format. It is easy for humans to
  ↪  read and write. It is based on the already successful JSON format and provides a way to help JSON
  ↪  data interoperate at Web-scale. JSON-LD is an ideal data format for programming environments,
  ↪  REST Web services, and unstructured databases such as Apache CouchDB and
  ↪  MongoDB.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://json-ld.org/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#JSON-LD_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#JSON-LD_Format">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
  <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#JSON-LD"/>
```

```
    <diio:file_extension xml:lang="en">.jsonld</diio:file_extension>
    <rdfs:comment xml:lang="en">JSON-LD 1.1, a JSON-based format to serialize Linked
    ↪  Data.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/json-ld/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#JSON_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#JSON_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#JSON"/>
    <diio:file_extension>.json</diio:file_extension>
    <rdfs:comment xml:lang="en">JSON JavaScript Object Notation is a lightweight data-interchange
    ↪  format.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.json.org/json-en.html</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#Kaa_IoT_Data -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#Kaa_IoT_Data">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#IoT_Data"/>
    <diio:has_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#JSON_Format"/>
    <diio:application_domain xml:lang="en">Water temperature monitoring</diio:application_domain>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#N-Quads_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#N-Quads_Format">
```

```
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#RDF"/>
    <diio:file_extension xml:lang="en">.nq</diio:file_extension>
    <rdfs:comment xml:lang="en">N-Quads is a line-based, plain text format for encoding an RDF
    ↪  dataset.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/n-quads/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#N-Triples_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#N-Triples_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#RDF"/>
    <diio:file_extension>.nt</diio:file_extension>
    <rdfs:comment xml:lang="en">N-Triples is a line-based, plain text format for encoding an RDF
    ↪  graph.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/n-triples/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#N3_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#N3_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#RDF"/>
    <diio:file_extension>.n3</diio:file_extension>
    <rdfs:comment xml:lang="en">N3 is a compact and readable alternative to RDF&apos;s XML syntax, but
    ↪  also is extended to allow greater expressiveness. It has subsets, one of which is RDF 1.0 equivalent,
    ↪  and one of which is RDF plus a form of RDF rules.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/TeamSubmission/n3/</rdfs:seeAlso>
</owl:NamedIndividual>
```

```
  <!-- http://www.co-ode.org/ontologies/diio#NDJSON-LD_Format -->

  <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#NDJSON-LD_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#JSON-LD"/>
    <diio:file_extension xml:lang="en">.jsonl</diio:file_extension>
    <diio:file_extension xml:lang="en">.ndjson</diio:file_extension>
    <diio:file_extension xml:lang="en">.ndjsonld</diio:file_extension>
    <rdfs:comment xml:lang="en">Newline-delimited JSON NDJSON is a data format for structured data that
    ↪   defines the structure of JSON data using lines as separators</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://github.com/json-ld/ndjson-ld</rdfs:seeAlso>
  </owl:NamedIndividual>



  <!-- http://www.co-ode.org/ontologies/diio#NHD_Plus -->

  <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#NHD_Plus">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
    <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#XML_Format"/>
    <diio:application_domain xml:lang="en">Geospatial data and Hydrography</diio:application_domain>
    <rdfs:comment xml:lang="en">USGS National Hydrography Dataset Plus NHD Plus
NHDPlus is an integrated suite of application-ready geospatial data products, incorporating many of the best
↪   features of the National Hydrography Dataset NHD, the National Elevation Dataset NED, and the National
↪   Watershed Boundary Dataset WBD. NHDPlus, based on the medium resolution NHD 1:100,000-scale,
↪   includes the stream network and improved linear networking,
feature naming, and “value added attributes” VAA. NHDPlus also includes elevation-derived
catchments</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.epa.gov/system/files/documents/2023-
    ↪   04/NHDPlusV2_User_Guide.pdf</rdfs:seeAlso>
  </owl:NamedIndividual>
```

```
<!-- http://www.co-ode.org/ontologies/diio#NHN -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#NHN">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
  <diio:application_domain xml:lang="en">Geospatial digital data</diio:application_domain>
  <rdfs:comment xml:lang="en">The Canadian National Hydro Network NHN focuses on providing a
  ↪  quality geometric description and a set of basic attributes describing Canada&apos;s inland surface
  ↪  waters. It provides geospatial digital data compliant with the NHN Standard such as lakes,
  ↪  reservoirs, watercourses rivers and streams, canals, islands, drainage linear network, toponyms or
  ↪  geographical names, constructions and obstacles related to surface waters, etc. The best available
  ↪  federal and provincial data are used for its production, which is done jointly by the federal and
  ↪  interested provincial and territorial partners. The NHN is created from existing data at the 1:50 000
  ↪  scale or better. The NHN data have a great potential for analysis, cartographic representation and
  ↪  display and will serve as base data in many applications. The NHN Work Unit Limits were created
  ↪  based on Water Survey of Canada Sub-Sub-Drainage Area.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://open.canada.ca/data/en/dataset/a4b190fe-e090-4e6d-881e-
  ↪  b87956c07977</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#OWL -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#OWL">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
  <diio:supports_Serialisation_Format
  ↪  rdf:resource="http://www.co-ode.org/ontologies/diio#OWL_Format"/>
  <diio:application_domain xml:lang="en">Internet</diio:application_domain>
  <diio:application_domain xml:lang="en">Semantic Web</diio:application_domain>
```

```
    <rdfs:comment xml:lang="en">OWL is a file format that stands for Web Ontology Language. It&apos;s a
    ↪  standard way to represent information about resources and their relationships. OWL files can be
    ↪  used to create visual representations, make inferences, and query knowledge.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/OWL/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#OWL_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#OWL_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:uses_File_Format_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#OWL"/>
    <diio:file_extension>.owl</diio:file_extension>
    <rdfs:comment xml:lang="en">The W3C Web Ontology Language OWL is a Semantic Web language
    ↪  designed to represent rich and complex knowledge about things, groups of things, and relations
    ↪  between things.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/OWL/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#OntoPlant -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#OntoPlant">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
    <diio:application_domain xml:lang="en">wastewater treatment</diio:application_domain>
    <rdfs:comment xml:lang="en">OntoPlant Sottara, 2014 is developed by Sottara et al., 2014 in OWL. It
    ↪  extends the SSN ontology to decouple control logic from equipment choices in wastewater treatment
    ↪  plants in 2014.</rdfs:comment>
</owl:NamedIndividual>
```

```
<!-- http://www.co-ode.org/ontologies/diio#RDF -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#RDF">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
  <diio:has_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#RDF/XML_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#BinaryRDF_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#JSON-LD_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#N-Quads_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#N-Triples_Format"/>
  <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#N3_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#NDJSON-LD_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#TriG-Star_Format"/>
  <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#TriG_Format"/>
  <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#TriX_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#Turtle-star_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#Turtle_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#RDF/JSON_Format"/>
  <diio:supports_Serialisation_Format
  ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#RDF/XML_Format"/>
  <diio:application_domain xml:lang="en">Internet</diio:application_domain>
  <diio:application_domain xml:lang="en">Semantic Web</diio:application_domain>
  <rdfs:comment xml:lang="en">RDF is a standard model for data interchange on the Web. RDF has
  ↪   features that facilitate data merging even if the underlying schemas differ, and it specifically supports
  ↪   the evolution of schemas over time without requiring all the data consumers to be
  ↪   changed.</rdfs:comment>
```

```
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/RDF/</rdfs:seeAlso>
  </owl:NamedIndividual>



  <!-- http://www.co-ode.org/ontologies/diio#SANDRE -->

  <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#SANDRE">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
    <diio:application_domain xml:lang="en">Water data</diio:application_domain>
    <rdfs:comment xml:lang="en">French National Service for Water Data and Reference-dataset
    ↪  Management SANDRE
The Sandre has for mission to build and make available the water reference data      <rdfs:seeAlso
↪  xml:lang="en">https://www.sandre.eaufrance.fr/sandre-core-part-french-water-information-system-
↪  sie?lang=en</rdfs:seeAlso>
  </owl:NamedIndividual>



  <!-- http://www.co-ode.org/ontologies/diio#SAREF -->

  <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#SAREF">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
    <diio:application_domain xml:lang="en">smart appliance domain models</diio:application_domain>
    <rdfs:comment xml:lang="en">SAREF STF-578, 2020 is developed by ETSI etsi in RDF/OWL and
    ↪  serialized in Turtle Berners-Lee, W3C, Prud'hommeaux, et al., 2014b as a Common denominator of
    ↪  23 smart appliance domain models in 2015</rdfs:comment>
  </owl:NamedIndividual>



  <!-- http://www.co-ode.org/ontologies/diio#SSN -->

  <owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#SSN">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
  <diio:application_domain xml:lang="en">Sensor network</diio:application_domain>
  <rdfs:comment xml:lang="en">SSN Semantic Sensor Network was developed by W3C with OWL in
  ↪  2012. It describes sensors and sensor networks for use in web applications, independent of any
  ↪  application domain.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#SSN_SOSA -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#SSN_SOSA">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
  <diio:application_domain xml:lang="en">Sensors and their observations</diio:application_domain>
  <rdfs:comment xml:lang="en">SSN SOSA is developed by OGC W3C with OWL in 2017. They describe
  ↪  sensors and their observations, the involved procedures, the studied features of interest, the
  ↪  samples used to do so, the observed properties, and actuators.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#STEP -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#STEP">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
  <rdfs:comment xml:lang="en">ISO 10303 is an ISO standard for the computer-interpretable
  ↪  representation and exchange of product manufacturing information. It is an ASCII-based
  ↪  format.!!!FIX ME!!!Its official title is: Automation systems and integration — Product data
  ↪  representation and exchange. It is known informally as &quot;STEP&quot;, which stands for
  ↪  &quot;Standard for the Exchange of Product model data&quot;. ISO 10303 can represent 3D objects
  ↪  in Computer-aided design CAD and related information.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.iso.org/standard/72237.html</rdfs:seeAlso>
</owl:NamedIndividual>
```

```
<!-- http://www.co-ode.org/ontologies/diio#SWEET -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#SWEET">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
  <diio:application_domain xml:lang="en">Water</diio:application_domain>
  <rdfs:comment xml:lang="en">Semantic Web for Earth and Environment Technology SWEET was
  ↪  developed by NASA with OWL in 2011 and updated in 2019. It is a middle-level ontology for
  ↪  environmental terminology.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#SWIM -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#SWIM">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
  <diio:supports_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#HyperCat"/>
  <diio:application_domain xml:lang="en">IoT semantic model for the water
  ↪  industry</diio:application_domain>
  <rdfs:comment xml:lang="en">SWIM is developed by Aquamatix AQUAMATIX, 2017 with OWL in 2016.
  ↪  It provides a device-level IoT semantic model for the water industry.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#Sociotechnical_System_Ontology -->

<owl:NamedIndividual
↪  rdf:about="http://www.co-ode.org/ontologies/diio#Sociotechnical_System_Ontology">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
```

```
    <diio:application_domain xml:lang="en">human interaction with technology, e.g. healthcare systems,
    ↪   manufacturing processes, infrastructure projects, organizational change management, education
    ↪   technology, transportation networks, social media platforms, environmental sustainability initiatives,
    ↪   disaster management, and complex software systems</diio:application_domain>
    <rdfs:comment xml:lang="en">Socio-technical systems ontology is a way of thinking about systems that
    ↪   considers both the social and technical aspects of a system. The goal is to improve the design and
    ↪   performance of a system by optimizing the social and technical subsystems.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#TXT_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#TXT_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:file_extension>.txt</diio:file_extension>
    <rdfs:comment xml:lang="en">A text file sometimes spelled textfile; an old alternative name is flat file is a
    ↪   kind of computer file that is structured as a sequence of lines of electronic text. A text file exists
    ↪   stored as data within a computer file system.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://en.wikipedia.org/wiki/Text_file</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#TriG-Star_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#TriG-Star_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:file_extension xml:lang="en">.trigs</diio:file_extension>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#TriG_Format -->
```

```
<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#TriG_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:file_extension>.trig</diio:file_extension>
    <rdfs:comment xml:lang="en">RDF 1.2 TriG shares triple terms with [RDF12-TURTLE] as a fourth kind of
    ↪  RDF term which can be used as the object of another triple, making it possible to make statements
    ↪  about other statements. RDF 1.2 TriG also adds shares directional language-tagged strings with
    ↪  [RDF12-TURTLE].</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/rdf12-trig/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#TriX_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#TriX_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:file_extension xml:lang="en">.trix</diio:file_extension>
    <diio:file_extension xml:lang="en">.xml</diio:file_extension>
    <rdfs:comment xml:lang="en">TriX Triples in XML is a serialization for named graphs. TriX aims to
    ↪  provide a highly normalized, consistent XML representation for RDF graphs, allowing the effective
    ↪  use of generic XML tools such as XSLT, XQuery, etc.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/2004/03/trix/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#Turtle-star_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#Turtle-star_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:file_extension>.ttls</diio:file_extension>
    <rdfs:comment xml:lang="en">RDF-star extends RDF with a convenient way to make statements about
    ↪  other statements</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://w3c.github.io/rdf-star/cg-spec/editors_draft.html</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#Turtle_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#Turtle_Format">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
    <diio:file_extension>.ttl</diio:file_extension>
    <rdfs:comment xml:lang="en">A textual syntax for RDF called Terse RDF Triple Language Turtle that
    ↪  allows an RDF graph to be completely written in a compact and natural text form, with abbreviations
    ↪  for common usage patterns and datatypes.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/turtle/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#UTF-8 -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#UTF-8">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Character_Encoding"/>
    <diio:application_domain xml:lang="en">electronic communication</diio:application_domain>
    <rdfs:comment xml:lang="en">UTF-8 is a character encoding standard used for electronic
    ↪  communication. Defined by the Unicode Standard, the name is derived from Unicode Transformation
    ↪  Format – 8-bit. Almost every webpage is stored in UTF-8.</rdfs:comment>
    <rdfs:seeAlso xml:lang="en">https://www.unicode.org/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#Utility_Network_Schemas -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#Utility_Network_Schemas">
```

```
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
    <diio:application_domain xml:lang="en">water and sewer network</diio:application_domain>
    <rdfs:comment xml:lang="en">Utility Network Schemas are developed by EC-INSPIRE Maintenance and
    ↪  Implementation, 2015 in XML for the water and sewer network model as part of a large European
    ↪  directive for geospatial data exchange in 2013.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#WDTF -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#WDTF">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
    <diio:application_domain xml:lang="en">Water; flood warning and forecasting</diio:application_domain>
    <rdfs:comment xml:lang="en">Water Data Transfer Format WDTF is developed by Australian Bureau of
    ↪  Meteorology with XML in 2013 as a format for transferring flood warning and forecasting data to the
    ↪  governing body. It is the precursor of WaterML2.0.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#WISDOM -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#WISDOM">
    <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
    <diio:can_adapt_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#INSPIRE"/>
    <diio:can_adapt_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#SAREF"/>
    <diio:can_adapt_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#SSN"/>
    <diio:can_adapt_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#SWIM"/>
    <diio:can_adapt_Ontology rdf:resource="http://www.co-ode.org/ontologies/diio#WatERP_WMO"/>
    <diio:application_domain xml:lang="en">Cyber-physical and social value chain of
    ↪  water</diio:application_domain>
    <rdfs:comment xml:lang="en">WISDOM Cardiff, 2014 is developed by Cardiff University with OWL in
    ↪  2015 for Cyber-physical and social ontology of the water value chain.</rdfs:comment>
```

```
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#WatERP_WMO -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#WatERP_WMO">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
  <diio:application_domain xml:lang="en">water sensing and management</diio:application_domain>
  <rdfs:comment xml:lang="en">WatERP WMO are developed by EURECAT-WatERP Ciancio et al., 2015
  ↪   with Semantic Markup for Web Services OWL-S as a lightweight ontology of generic concepts for
  ↪   water sensing and management in 2013.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#WaterML_2.0 -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#WaterML_2.0">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
  <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#XML_Format"/>
  <diio:supports_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#HY_Features"/>
  <diio:supports_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#IFC4"/>
  <diio:supports_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#WDTF"/>
  <diio:supports_Standard rdf:resource="http://www.co-ode.org/ontologies/diio#XHydro"/>
  <diio:application_domain xml:lang="en">Hydrometeorological observations and
  ↪   measurements</diio:application_domain>
  <rdfs:comment xml:lang="en">WaterML is developed by OGC with XML in 2012. It is a new data
  ↪   exchange standard in Hydrology to exchange hydrometeorological observations and measurements.
  ↪   It harmonizes several exchange formats for water data with relevant OGC and ISO
  ↪   standards.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.ogc.org/publications/standard/waterml/</rdfs:seeAlso>
</owl:NamedIndividual>
```

```
<!-- http://www.co-ode.org/ontologies/diio#XHydro -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#XHydro">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
  <diio:supports_Serialisation_Format rdf:resource="http://www.co-ode.org/ontologies/diio#XML_Format"/>
  <diio:application_domain xml:lang="en">Water data in time series</diio:application_domain>
  <rdfs:comment xml:lang="en">XHydro is an XML format for inter-departmental and cost-efficient
  ↪  time-series data exchange.</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.xhydro.org/index_en.html</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#XML -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#XML">
  <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
  <diio:application_domain xml:lang="en">Internet</diio:application_domain>
  <diio:application_domain xml:lang="en">Semantic Web</diio:application_domain>
  <rdfs:comment xml:lang="en">The Extensible Markup Language XML is a subset of SGML. Its goal is to
  ↪  enable generic SGML to be served, received, and processed on the Web in the way that is now
  ↪  possible with HTML. XML has been designed for ease of implementation and for interoperability with
  ↪  both SGML and HTML. XML is a way to store, share, and exchange data between systems.
  ↪  It&apos;s a markup language that uses tags to define the structure and meaning of data. XML is
  ↪  similar to HTML, but XML doesn&apos;t have predefined tags</rdfs:comment>
  <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/xml/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#XML_Format -->
```

```
<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#XML_Format">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
   <diio:file_extension>.xml</diio:file_extension>
   <rdfs:comment xml:lang="en">Extensible Markup Language XML is a simple, very flexible text format
   ↪  derived from SGML ISO 8879.</rdfs:comment>
   <rdfs:seeAlso xml:lang="en">https://www.w3.org/XML/</rdfs:seeAlso>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#hydrOntology -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#hydrOntology">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
   <diio:application_domain xml:lang="en">Water</diio:application_domain>
   <rdfs:comment xml:lang="en">hydrOntology was developed by Vilches-Blázquez et al., 2015 with OWL
   ↪  in 2009. It aims to integrate hydrographical data sources: town planning perspective and top-down
   ↪  methodology.</rdfs:comment>
</owl:NamedIndividual>



<!-- http://www.co-ode.org/ontologies/diio#RDF/JSON_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#RDF/JSON_Format">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
   <diio:file_extension xml:lang="en">.rj</diio:file_extension>
   <rdfs:comment xml:lang="en">RDF/JSON allows an RDF graph to be completely written in a form
   ↪  compatible with the JavaScript Object Notation JSON [RFC4627] and alternative to the one
   ↪  recommended in JSON-LD [JSON-LD].</rdfs:comment>
   <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/rdf-json/</rdfs:seeAlso>
</owl:NamedIndividual>
```

```
<!-- http://www.co-ode.org/ontologies/diio#RDF/XML_Format -->

<owl:NamedIndividual rdf:about="http://www.co-ode.org/ontologies/diio#RDF/XML_Format">
   <rdf:type rdf:resource="http://www.co-ode.org/ontologies/diio#Data_Serialisation_Format"/>
   <diio:file_extension>.owl</diio:file_extension>
   <diio:file_extension>.rdf</diio:file_extension>
   <diio:file_extension>.rdfs</diio:file_extension>
   <diio:file_extension>.xml</diio:file_extension>
   <rdfs:comment xml:lang="en">RDF/XML is a syntax, defined by the W3C, to express i.e. serialize an
   ↪  RDF graph as an XML document.</rdfs:comment>
   <rdfs:seeAlso xml:lang="en">https://www.w3.org/TR/rdf-syntax-grammar/</rdfs:seeAlso>
</owl:NamedIndividual>



<!--
///////////////////////////////////////////////////////////////////////////////////////
//
// Rules
//
///////////////////////////////////////////////////////////////////////////////////////
 -->

<rdf:Description rdf:about="http://www.co-ode.org/ontologies/diio#dff">
   <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.co-ode.org/ontologies/diio#ssf">
   <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.co-ode.org/ontologies/diio#o">
   <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.co-ode.org/ontologies/diio#s">
```

```
  <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.co-ode.org/ontologies/diio#o1">
  <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.co-ode.org/ontologies/diio#o2">
  <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
</rdf:Description>
<rdf:Description>
  <swrla:isRuleEnabled
  ↪  rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">true</swrla:isRuleEnabled>
  <rdfs:comment></rdfs:comment>
  <rdfs:label>S1</rdfs:label>
  <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Imp"/>
  <swrl:body>
    <rdf:Description>
      <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
      <rdf:first>
        <rdf:Description>
          <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#ClassAtom"/>
          <swrl:classPredicate rdf:resource="http://www.co-ode.org/ontologies/diio#Data_File_Format"/>
          <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#dff"/>
        </rdf:Description>
      </rdf:first>
      <rdf:rest>
        <rdf:Description>
          <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
          <rdf:first>
            <rdf:Description>
              <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#IndividualPropertyAtom"/>
              <swrl:propertyPredicate rdf:resource="http://www.co-
              ↪  ode.org/ontologies/diio#supports_Serialisation_Format"/>
              <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#dff"/>
              <swrl:argument2 rdf:resource="http://www.co-ode.org/ontologies/diio#ssf"/>
```

```
                  </rdf:Description>
                </rdf:first>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              </rdf:Description>
            </rdf:rest>
          </rdf:Description>
        </swrl:body>
        <swrl:head>
          <rdf:Description>
            <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
            <rdf:first>
              <rdf:Description>
                <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#IndividualPropertyAtom"/>
                <swrl:propertyPredicate
                ↪  rdf:resource="http://www.co-ode.org/ontologies/diio#can_adapt_Serialisation_Format"/>
                <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#dff"/>
                <swrl:argument2 rdf:resource="http://www.co-ode.org/ontologies/diio#ssf"/>
              </rdf:Description>
            </rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          </rdf:Description>
        </swrl:head>
      </rdf:Description>
      <rdf:Description>
        <swrla:isRuleEnabled
        ↪  rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">true</swrla:isRuleEnabled>
        <rdfs:comment></rdfs:comment>
        <rdfs:label>S3</rdfs:label>
        <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Imp"/>
        <swrl:body>
          <rdf:Description>
            <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
            <rdf:first>
              <rdf:Description>
          <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#ClassAtom"/>
          <swrl:classPredicate rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
          <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#o1"/>
        </rdf:Description>
      </rdf:first>
      <rdf:rest>
        <rdf:Description>
          <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
          <rdf:first>
            <rdf:Description>
              <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#ClassAtom"/>
              <swrl:classPredicate rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
              <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#o2"/>
            </rdf:Description>
          </rdf:first>
          <rdf:rest>
            <rdf:Description>
              <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
              <rdf:first>
                <rdf:Description>
                  <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#IndividualPropertyAtom"/>
                  <swrl:propertyPredicate
                  ↪  rdf:resource="http://www.co-ode.org/ontologies/diio#supports_Ontology"/>
                  <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#o1"/>
                  <swrl:argument2 rdf:resource="http://www.co-ode.org/ontologies/diio#o2"/>
                </rdf:Description>
              </rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            </rdf:Description>
          </rdf:rest>
        </rdf:Description>
      </rdf:rest>
    </rdf:Description>
  </swrl:body>
```

```
    <swrl:head>
      <rdf:Description>
        <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
        <rdf:first>
          <rdf:Description>
            <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#IndividualPropertyAtom"/>
            <swrl:propertyPredicate
            ↪  rdf:resource="http://www.co-ode.org/ontologies/diio#can_adapt_Ontology"/>
            <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#o1"/>
            <swrl:argument2 rdf:resource="http://www.co-ode.org/ontologies/diio#o2"/>
          </rdf:Description>
        </rdf:first>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      </rdf:Description>
    </swrl:head>
  </rdf:Description>
  <rdf:Description>
    <swrla:isRuleEnabled
    ↪  rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">true</swrla:isRuleEnabled>
    <rdfs:comment></rdfs:comment>
    <rdfs:label>S2</rdfs:label>
    <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#Imp"/>
    <swrl:body>
      <rdf:Description>
        <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
        <rdf:first>
          <rdf:Description>
            <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#ClassAtom"/>
            <swrl:classPredicate rdf:resource="http://www.co-ode.org/ontologies/diio#Ontology"/>
            <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#o"/>
          </rdf:Description>
        </rdf:first>
        <rdf:rest>
          <rdf:Description>
```

```
                <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
                <rdf:first>
                  <rdf:Description>
                    <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#ClassAtom"/>
                    <swrl:classPredicate rdf:resource="http://www.co-ode.org/ontologies/diio#Standard"/>
                    <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#s"/>
                  </rdf:Description>
                </rdf:first>
                <rdf:rest>
                  <rdf:Description>
                    <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
                    <rdf:first>
                      <rdf:Description>
                        <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#IndividualPropertyAtom"/>
                        <swrl:propertyPredicate
                        ↪   rdf:resource="http://www.co-ode.org/ontologies/diio#supports_Standard"/>
                        <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#o"/>
                        <swrl:argument2 rdf:resource="http://www.co-ode.org/ontologies/diio#s"/>
                      </rdf:Description>
                    </rdf:first>
                    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                  </rdf:Description>
                </rdf:rest>
              </rdf:Description>
            </rdf:rest>
          </rdf:Description>
        </swrl:body>
        <swrl:head>
          <rdf:Description>
            <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#AtomList"/>
            <rdf:first>
              <rdf:Description>
                <rdf:type rdf:resource="http://www.w3.org/2003/11/swrl#IndividualPropertyAtom"/>
```

```
                    <swrl:propertyPredicate
                    ↪ rdf:resource="http://www.co-ode.org/ontologies/diio#can_adapt_Standard"/>
                    <swrl:argument1 rdf:resource="http://www.co-ode.org/ontologies/diio#o"/>
                    <swrl:argument2 rdf:resource="http://www.co-ode.org/ontologies/diio#s"/>
                </rdf:Description>
            </rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </rdf:Description>
    </swrl:head>
</rdf:Description>
</rdf:RDF>



<!-- Generated by the OWL API version 4.5.29.2024-05-13T12:11:03Z https://github.com/owlcs/owlapi -->
```

source-code

```
Data_File_Format?dff ^ supports_Serialisation_Format?dff,?ssf ->
↪ is_avialable_in_Serialisation_Format?dff, ?ssf

Data_File_Format?dff ^ supports_Serialisation_Format?dff,?ssf ->
↪ can_be_converted_into_Serialisation_Format?dff, ?ssf

Data_File_Format?dff ^ supports_Serialisation_Format?dff,?ssf ->
↪ has_Serialisation_Format?dff, ?ssf

Standard?s ^ Ontology?o ^ supports_Standard?o, ?s ->
↪ can_be_converted_into_Standard?o,?s

Ontology?o1 ^ Ontology?o1 ^ supports_Ontology?o1, ?o2 ->
↪ can_be_converted_into_Ontology?o1,?o2
```

```
alined --> can_be_converted_into_Ontology
```

Appendix C

Semantic Similarity Scoring Ontology (S3O)

All DIIF Ph.d.-relevant data and source code are available under the URL https://gitlab.com/phd30/DIIM.

source-code

```
@prefix : <http://www.w3.org/2002/07/owl#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix s3o: <http://www.co-ode.org/ontologies/s3o#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://www.w3.org/2002/07/owl#> .

[ rdf:type owl:Ontology
 ] .

#################################################################
#    Object Properties
#################################################################
```

```
###  http://www.co-ode.org/ontologies/s3o#has_Similarity

s3o:has_Similarity rdf:type owl:ObjectProperty ;

                   rdfs:domain s3o:Document ,

                               s3o:Term ;

                   rdfs:range s3o:Similarity .




###  http://www.co-ode.org/ontologies/s3o#ref_IoT_Data

s3o:ref_IoT_Data rdf:type owl:ObjectProperty ;

                 rdfs:domain s3o:Similarity ;

                 rdfs:range s3o:IoT_Data .




###  http://www.co-ode.org/ontologies/s3o#ref_Ontology

s3o:ref_Ontology rdf:type owl:ObjectProperty ;

                 rdfs:domain s3o:Similarity ;

                 rdfs:range s3o:Ontology .




###  http://www.co-ode.org/ontologies/s3o#ref_Term

s3o:ref_Term rdf:type owl:ObjectProperty ;

             rdfs:domain s3o:Similarity ;

             rdfs:range s3o:Term .




###  http://www.co-ode.org/ontologies/s3o#term_Used_by

s3o:term_Used_by rdf:type owl:ObjectProperty ;

                 owl:inverseOf s3o:uses_Term ;

                 rdfs:domain s3o:Term ;

                 rdfs:range s3o:Document ,

                            s3o:IoT_Data ,

                            s3o:Ontology .
```

```
###  http://www.co-ode.org/ontologies/s3o#uses_Term
s3o:uses_Term rdf:type owl:ObjectProperty ;
              rdfs:domain s3o:Document ,
                          s3o:IoT_Data ,
                          s3o:Ontology ;
              rdfs:range s3o:Term .



#################################################################
#    Data properties
#################################################################

###  http://www.co-ode.org/ontologies/s3o#serialization_format
s3o:serialization_format rdf:type owl:DatatypeProperty ;
                         rdfs:domain owl:Thing .



###  http://www.co-ode.org/ontologies/s3o#similarity_value
s3o:similarity_value rdf:type owl:DatatypeProperty ;
                     rdfs:domain s3o:Similarity .



###  http://www.co-ode.org/ontologies/s3o#uri
s3o:uri rdf:type owl:DatatypeProperty ;
        rdfs:domain owl:Thing .



#################################################################
#    Classes
#################################################################

###  http://www.co-ode.org/ontologies/s3o#Bigram_Term_Similarity
s3o:Bigram_Term_Similarity rdf:type owl:Class ;
                           rdfs:subClassOf s3o:Term_Similarity .
```

```
###  http://www.co-ode.org/ontologies/s3o#Document

s3o:Document rdf:type owl:Class ;

             rdfs:subClassOf owl:Thing .



###  http://www.co-ode.org/ontologies/s3o#Document_Similarity

s3o:Document_Similarity rdf:type owl:Class ;

                        rdfs:subClassOf s3o:Similarity .



###  http://www.co-ode.org/ontologies/s3o#Fourgram_Term_Similarity

s3o:Fourgram_Term_Similarity rdf:type owl:Class ;

                             rdfs:subClassOf s3o:Term_Similarity .



###  http://www.co-ode.org/ontologies/s3o#IoT_Data

s3o:IoT_Data rdf:type owl:Class ;

             rdfs:subClassOf s3o:Document .



###  http://www.co-ode.org/ontologies/s3o#LSI_Document_Similarity

s3o:LSI_Document_Similarity rdf:type owl:Class ;

                            rdfs:subClassOf s3o:Document_Similarity .



###  http://www.co-ode.org/ontologies/s3o#Ngram_Term_Similarity

s3o:Ngram_Term_Similarity rdf:type owl:Class ;

                          rdfs:subClassOf s3o:Term_Similarity .



###  http://www.co-ode.org/ontologies/s3o#Ontology

s3o:Ontology rdf:type owl:Class ;
```

```
             rdfs:subClassOf s3o:Document .



###  http://www.co-ode.org/ontologies/s3o#SSM_Bigram_Term_Similarity

s3o:SSM_Bigram_Term_Similarity rdf:type owl:Class ;

                               rdfs:subClassOf s3o:Bigram_Term_Similarity .



###  http://www.co-ode.org/ontologies/s3o#SSM_Unigram_Term_Similarity

s3o:SSM_Unigram_Term_Similarity rdf:type owl:Class ;

                                rdfs:subClassOf s3o:Unigram_Term_Similarity .



###  http://www.co-ode.org/ontologies/s3o#Similarity

s3o:Similarity rdf:type owl:Class ;

               rdfs:subClassOf owl:Thing .



###  http://www.co-ode.org/ontologies/s3o#Term

s3o:Term rdf:type owl:Class ;

         rdfs:subClassOf owl:Thing .



###  http://www.co-ode.org/ontologies/s3o#Term_Similarity

s3o:Term_Similarity rdf:type owl:Class ;

                    rdfs:subClassOf s3o:Similarity .



###  http://www.co-ode.org/ontologies/s3o#Trigram_Term_Similarity

s3o:Trigram_Term_Similarity rdf:type owl:Class ;

                            rdfs:subClassOf s3o:Term_Similarity .



###  http://www.co-ode.org/ontologies/s3o#Unigram_Term_Similarity
```

```
s3o:Unigram_Term_Similarity rdf:type owl:Class ;
                            rdfs:subClassOf s3o:Term_Similarity .


###  http://www.co-ode.org/ontologies/s3o#Word2Vec_Bigram_Term_Similarity
s3o:Word2Vec_Bigram_Term_Similarity rdf:type owl:Class ;
                                    rdfs:subClassOf s3o:Bigram_Term_Similarity .


###  http://www.co-ode.org/ontologies/s3o#Word2Vec_Unigram_Term_Similarity
s3o:Word2Vec_Unigram_Term_Similarity rdf:type owl:Class ;
                                     rdfs:subClassOf s3o:Unigram_Term_Similarity .


###  Generated by the OWL API version 4.5.9.2019-02-01T07:24:44Z
↪  https://github.com/owlcs/owlapi
```

Appendix D

DIIM Semantic Similarity Scoring Tool (DS3T)

All DIIF Ph.d.-relevant data and source code are available under the URL https://gitlab.com/phd30/DIIM.

Appendix E

Data Information Interoperability Questionnaire (DIIQ)

Data and Information Interoperability Questionnaire (DIIQ) Personal

Expected completion time: The expected completion time 10 - 15 min.
Submit deadline: The respondent with sufficient domain knowledge and skills should be able to complete the questionnaire within a week of receiving the invitation.
Contact Info: Mandeep Singh mandeep.singh11@mail.bcu.ac.uk
Disclaimer: This questionnaire is used only for research purposes to validate the interoperability of IoT-enabled applications at syntactic and semantic levels. It does not collect personal information.

1. What is the highest academic title you currently hold? *

- ◯ None
- ◯ Bechealor
- ◯ Masters
- ◯ Doctor
- ◯ Professor

2. How would you rate your proficiency in the English language? *

- ◯ No Proficiency
- ◯ Elementary Proficiency
- ◯ Limited Working Proficiency
- ◯ Professional Working Proficiency
- ◯ Full Professional Proficiency
- ◯ Native / Bilingual Proficiency

3. How familiar are you with computer science? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

4. How familiar are you with the data representation formats, such as CSV, XML, JSON, RDF, etc..? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

5. How familiar are you with terms used in the water domain, such as pH, temperature, turbidity, Fahrenheit, Celsius, do, etc.? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

6. How familiar are you with terms used in the IoT domain, such as sensor, timestamp, measurement, location, longitude, latitude, etc.? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

7. How familiar are you with the NLP concepts, such as terms/words, unigrams, bigrams, similarity/relatedness, semantics, syntactic, etc.? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

8. How familiar are you with the word similarity scoring algorithms, such as Word to Vector, Levenstein, String Syntax Matching, etc.? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

9. How familiar are you with the process of word mapping? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

10. How familiar are you with the process of mapping/aligning datasets? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

11. How familiar are you with the process of mapping/aligning ontologies? *

- ◯ Not at all familar
- ◯ Slightly familiar
- ◯ Somewhat familiar
- ◯ Moderately familiar
- ◯ Extremely familiar

12. Please rate the similarity of the following two terms to label data values: Sensor, s *

- ◯ not similar
- ◯ somewhat similar
- ◯ similar
- ◯ very much similar
- ◯ identical

13. Please rate the similarity of the following two terms to label measurement values of a sensor: Sensor, sen *

- ◯ not similar
- ◯ somewhat similar
- ◯ similar
- ◯ very much similar
- ◯ identical

14. Please rate the similarity of the following two terms to label measurement values of a temperature sensor: Temperature, temp *

- ◯ not similar
- ◯ somewhat similar
- ◯ similar
- ◯ very much similar
- ◯ identical

15. Please rate the similarity of the following two terms to label measurement values of a sensor: Dissolved Oxygen, DO *

- ◯ not similar
- ◯ somewhat similar
- ◯ similar
- ◯ very much similar
- ◯ identical

16. Please rate the similarity of the following two terms to label measurement values of a sensor: Water volume, WaterVolume *

- ◯ not similar
- ◯ somewhat similar
- ◯ similar
- ◯ very much similar
- ◯ identical

17. Please rate the similarity of the following two terms to label measurement values of a sensor: DissolvedOxygen, Dissolved Oxygen *

- not similar
- somewhat similar
- similar
- very much similar
- identical

18. Please rate the similarity of the following two terms to label measurement values of a sensor: ph, power of hydrogen *

- not similar
- somewhat similar
- similar
- very much similar
- identical

19. Please rate the similarity of the following two terms to label data values: ph, potential of hydrogen *

- not similar
- somewhat similar
- similar
- very much similar
- identical

20. Do you have access to all resources related to the questionnaire? *

- Yes
- No

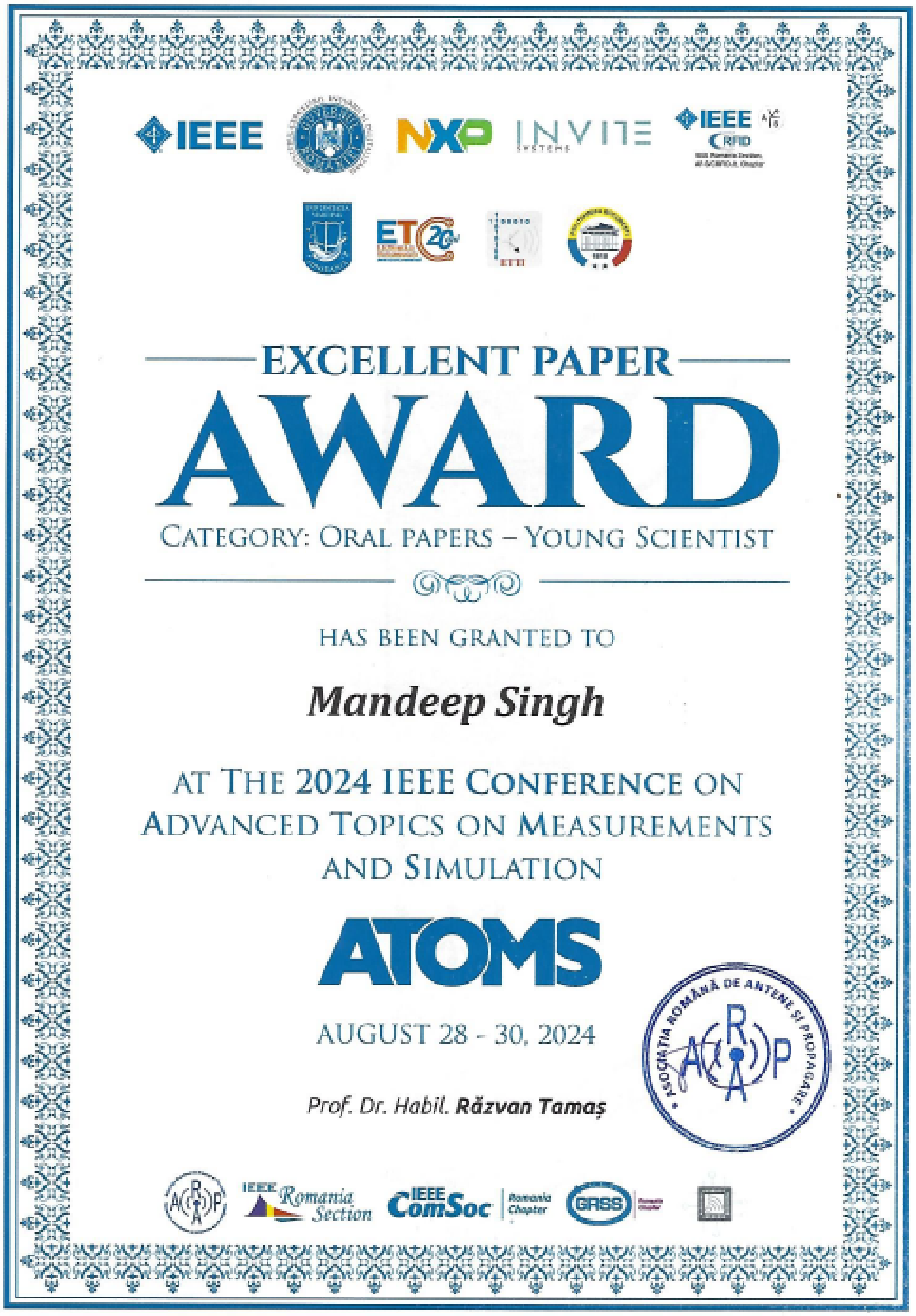
Appendix F

Publications and Awards

F.1 IEEE ATOMS 2024

ingh, Mandeep, Moatasim Mahmoud, Rizou Stamatia, Zaharias D. Zaharis, Pavlos I. Lazaridis, Vladimir K. Poulkov, and Wenyan Wu (2024). "Towards 5G/6G Data Harmonization through NLP and Semantic Web Technologies". In: 2024 Advanced Topics on Measurement and Simulation (ATOMS), pp. 291–294. doi: 10.1109/ATOMS60779.2024.10921618

Online link to the publication site



EXCELLENT PAPER

AWARD

CATEGORY: ORAL PAPERS – YOUNG SCIENTIST

HAS BEEN GRANTED TO

Mandeep Singh

AT THE 2024 IEEE CONFERENCE ON ADVANCED TOPICS ON MEASUREMENTS AND SIMULATION

ATOMS

AUGUST 28 - 30, 2024

Prof. Dr. Habil. ***Răzvan Tamaș***

ASOCIAȚIA ROMÂNĂ DE ANTENE ȘI PROPAGARE

Figure F.1: Excellent Paper Award at IEEE ATOMS 2024

Towards 5G/6G Data Harmonization through NLP and Semantic Web Technologies

1st Mandeep Singh
R&D, SingularLogic
15561 Cholargos, Greece
Birmingham City University
B5 5JU Birmingham, UK
0000-0001-6381-1140

2nd Moatasim Mahmoud
R&D, SingularLogic
15561 Cholargos, Greece
Aristotle University of Thessaloniki
54124 Thessaloniki, Greece
0000-0002-8319-6912

3rd Rizou Stamatia
R&D, SingularLogic
15561 Cholargos, Greece
0000-0002-3683-060X

4th Zaharias D. Zaharis
Aristotle University of Thessaloniki
54124 Thessaloniki, Greece
0000-0002-4548-282X

5th Pavlos I. Lazaridis
University of Huddersfield
HD1 3DH Huddersfield, U.K
0000-0001-5091-2567

6th Vladimir K. Poulkov
Technical University of Sofia
1000 Sofia, Bulgaria
0000-0003-3226-5639

7th Wenyan Wu
Birmingham City University
B5 5JU Birmingham, UK
0000-0002-4823-3685

***Abstract*—Telecommunication systems utilize several mechanisms to collect data from 5G/6G-enabled IoT. In the 5G/6G community, various AI techniques and tools are applied to 5G/6G data to monitor, predict, and make decisions. Therefore, 5G/6G data must be interoperable for monitoring, prediction, and decision support systems. However, 5G/6G data are typically mapped in local data models for local applications, which poses challenges to using them in different or cross-domain applications due to a lack of interoperability issues. In this paper, we propose an approach to support and enhance the interoperability of 5G/6G data through NLP and Semantic Web technologies to achieve 5G/6G data harmonization.**

***Index Terms*—5G/6G data, QoE, interoperability, data harmonization, knowledge graph.**

I. Introduction

Data collected in 5G/6G application scenarios often serve a particular application-specific purpose, which leads to heterogeneous data models if these applications don't use domain-specific ontologies or standards. Hence, diverse 5G/6G data models must be aligned with the end application to make them interoperable before other applications use them. This poses a significant challenge to interoperability when integrating 5G/6G data into a pre-established system. Interoperability of 5G/6G data can only be successful by aligning the syntactic (structure) and semantic (meaning) interoperability of the data/information they share. In this line, the alignment tasks are cumbersome and challenging for an application engineer during the integration of 5G/6G applications since it requires a manual review of the relevant data models of applications and domain-specific ontologies or standards that he could potentially align with the 5G/6G data. Additionally, before aligning each term used in the 5G/6G data with the concepts (terms) defined in the domain-specific ontologies or standards, all related terms in the given ontologies must be considered according to their similarity or relatedness.

This paper discusses Quality of Experience (QoE) in 5G/6G applications from different angles. Initially, we provide relevant definitions of objective and subjective metrics to measure the QoE in 5G/6G-enabled applications. Next, we motivate the data interoperability problem in the context of QoE data alignment and provide an approach to address this problem along with preliminary results. Finally, as part of the case study, we utilize Machine Learning (ML) algorithms to develop a QoE prediction model. Our work showcases the potential of using semantic technologies and machine learning techniques in the context of QoE prediction.

II. QoE Literature Review

QoE provides quantified subjective and objective methodologies for depicting the users' satisfaction with a specific service or application. QoE prediction estimates subjective QoE scores based on various network measurements and service specifications. In this context, ML and Deep Learning (DL) have been investigated to predict QoE levels in communication networks for multiple applications. [1] considers ML-based prediction of users' QoE in Software-defined Networking (SDN). First, a subjective evaluation based on Degradation Category Ratings (DCR) is conducted. The collected data is then used to train and assess the performance of four ML solutions: Decision Tree (DT), neural network, K-nearest Neighbours (KNN), and Random Forest (RF). In this work, both full reference parameters and application metrics have been utilized for QoE prediction. In [4], a Convolutional Neural Network (CNN) has been utilized for continuous QoE prediction in video streaming applications. This work shows CNNs can provide accurate QoE estimates while maintaining low computational complexity. To improve QoE in edge-enabled Internet of Things (IoT) networks, the authors in [8] developed an improved QoE model and proposed a novel Deep Reinforcement Learning (DRL) solution. The adopted QoE model uses Quality of Service (QoS) information derived from the computation offloading processes.

QoE aims to quantify human satisfaction and perception of a service. However, depending on their application, people's experience services can be subject to varying parameters. Thus, researchers studying QoE have pointed out the importance of considering the unique nature and demands of different sets of applications. For instance, the IoT connects many devices to the Internet. These devices are developed to perform different tasks and can be used in various applications. IoT devices are typically designed to operate with little

TABLE I
QoE PREDICTION ERROR OVER THE TWO DATASETS

MODEL	LIVE_NFLX		AleksandrIvchenko	
	MAE	MSE	MAE	MSE
LR	0.1142	0.0214	0.1138	0.0211
KRR	0.1115	0.0213	0.1240	0.0243
SVR	0.1213	0.0239	0.1099	0.0186

to no human involvement, which creates a challenge when assessing the application QoE. Because of this, researchers started investigating QoE aspects in emerging 5G/6G-enabled IoT applications. The authors in [11] conducted an experimental study of QoE parameters for smart-wearable devices. They included 40 human subjects in a free-living environment with five different wearable devices and used the results for QoE modelling. [15] provided a QoE measurement framework for IoT applications and tested their methodology in Jakarta Smart City. This work targeted six public services, including garbage trucks, fire and rescue services, and city bus transportation.

III. A Case Study: QoE Prediction Across Datasets

QoE prediction models are particularly utilized to help improve content delivery services by making resource allocation decisions to enhance the user experience. Multiple features can be exploited in video delivery services to predict user satisfaction with the videos. This includes information about the video content, network parameters, and display specifications. Combining QoE data from different sources can be beneficial in various contexts, one of which is QoE prediction. However, different video services may adopt different terminologies in their QoE frameworks. This section tests the feasibility of applying QoE prediction across two different QoE datasets. We use the $LIVE\_NFLX$ dataset [2] for fitting QoE models and apply the models for predicting the subjective QoE scores in Aleksandrlvchenko's QoE-Assessment dataset [12]. We use two features to predict the final Mean Opinion Score (MOS). Namely, the mean Peak Signal-to-Noise Ratio (PSNR) and the number of stalling events. We selected these two metrics as they strongly relate to the users' perception of video content. Intuitively, higher PSNR values and fewer stalling events should improve QoE. The two datasets use different labels to refer to these features. To harmonize the data from the two datasets, we manually map the terms from both datasets that correspond to these features. We also match and normalize the subjective scores from each dataset according to their scoring ranges to work with unified data scales.

We utilize three regression methods in our experiments: Linear Regression (LR), Kernel Ridge Regression (KRR), and Support Vector Regression (SVR). We first fit each of the three models using the $LIVE\_NFLX$ dataset and assess their performance over a test portion of the same dataset. We then utilize the same models, fitted using only the $LIVE\_NFLX$ dataset, to predict the QoE scores derived from $Aleksandrlvchenko$'s dataset. Figure 1 illustrates the original (true) data points and the fitted prediction models. Normalized MOS are shown for both datasets, obtained from pairs of stalling events and PSNR values. In Figure 1, the MOS scores are shown for the test part of the $LIVE\_NFLX$ dataset, along with the predictions from each model. The unseen data points of $Aleksandrlvchenko$'s dataset are depicted in Figure 1, along with the MOS values from the trained models. The resulting mean absolute error Mean Absolute Error (MAE) and the Mean Squared Error (MSE) values are summarized in Table I. The results show that QoE prediction models can be utilized across different datasets. All three models provided decent predictions when applied to the unseen dataset. This held even though $Aleksandrlvchenko$'s dataset contains data points with larger input ranges for both PSNR and stalling events. This highlights the ability of the QoE models to extrapolate for larger input ranges.

IV. QoE data harmonization

5G/6G refers to the next generation of wireless communication technology. One of the hot research areas for 6G technology is in conjunction with edge Artificial Intelligence (AI), which involves processing data at or near the source rather than transmitting it back to a centralized data centre for analysis. This could significantly improve areas such as autonomous vehicles, healthcare, and manufacturing. However, this requires harmonizing 5G/6G data in different applications with different data formats or labels to record measurement data. In the data integration process, the following two harmonization scenarios arise:

1) Data model to data model: When two different applications with different data models want to use each other's data, their data models must be harmonized to use their datasets as a single source. The case study presented in the previous section is based on this scenario.
2) Data model to standard/ontology: An application's data model must adapt domain-specific standards and ontologies so that all other domain-specific applications can use its datasets as a single source. This scenario enhances the interoperability of a data model.

In both scenarios, we must overcome syntactic (different data representation formats) and semantic (using different terms in data models to label the same data) interoperability issues of the 5G/6G data. Therefore, someone must manually inspect the given datasets, standards, or ontologies before they can be harmonized as a single data source. Harmonizing data can become a cumbersome task when done manually. However, Natural Language Processing (NLP) and Semantic Web (SW) technologies can support the data harmonization process.

A. Approach

In [13], Singh et al. introduce a Data and Information Interoperability Model (DIIM) to enable and enhance the interoperability of the IoT data for Smart Water Network (SWN) applications. To achieve syntactic interoperability in Figure 2, given data is converted into Resource Description Framework (RDF) format by using a RDF-converter [16], then it can be transformed into an application-specific format. To achieve semantic interoperability between two applications, similar terms used in the data models of given applications are linked via annotations in the RDF graphs, representing the data of these applications. To address the semantic issue, we follow the DIIM approach that takes

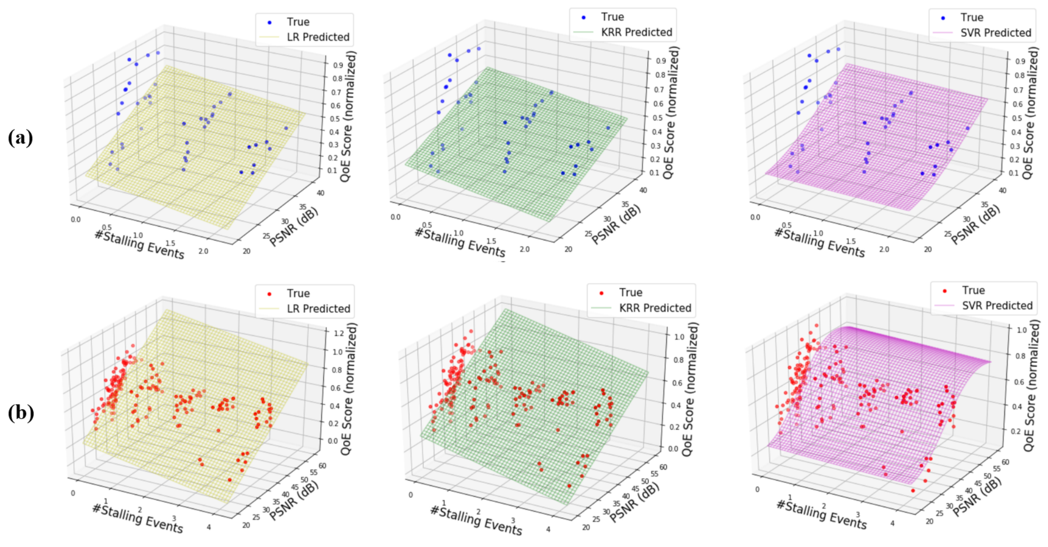


Fig. 1. QoE prediction results obtained from three different prediction models. (a) predicted MOS scores over the $LIVE\_NFLX$ dataset. (b) predicted MOS scores over $AleksandrIvchenko$ dataset

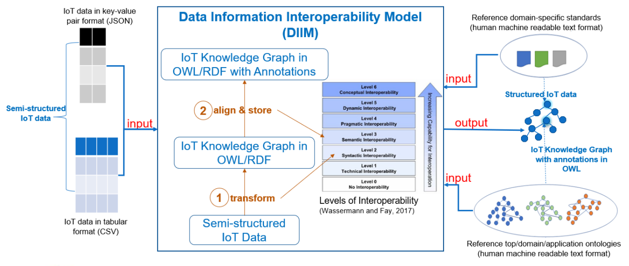

IoT data in key-value pair format (JSON)
Semi-structured IoT data
IoT data in tabular format (CSV)
input
Data Information Interoperability Model (DIIM)
IoT Knowledge Graph in OWL/RDF with Annotations
2 align & store
IoT Knowledge Graph in OWL/RDF
1 transform
Semi-structured IoT Data
Levels of Interoperability (Wassermann and Fay, 2017)
Increasing Capability for Interoperation
input
output
input
Reference domain-specific standards (human machine readable text format)
Structured IoT data
IoT Knowledge Graph with annotations in OWL
Reference top/domain/application ontologies (human machine readable text format)

Fig. 2. Data and Information Interoperability Model (DIIM)

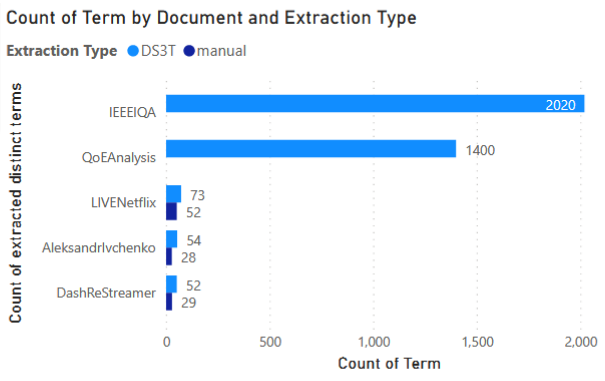


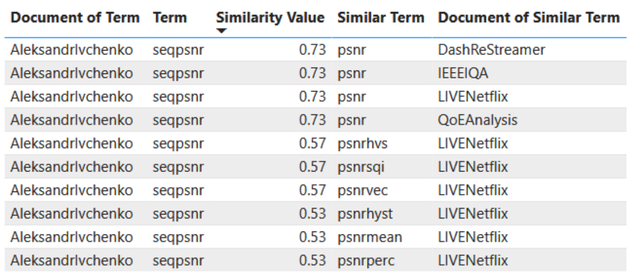
Fig. 3. Count of distinct terms extracted manually and by DS3T from input datasets and standards

datasets and domain-specific ontologies or standards as input and generates an annotated Knowledge Graph as output. DIIM Semantic Similarity Scoring Tool (DS3T) [14] compares the similarity between terms used in datasets and ontologies/standards. DS3T stores the found similarity results in a Semantic Similarity Score Ontology (S3O) [14] with reference links to the respective terms and their originating documents (datasets or standards/ontologies), which use these terms.

B. QoE 5G/6G Showcase and Evaluation

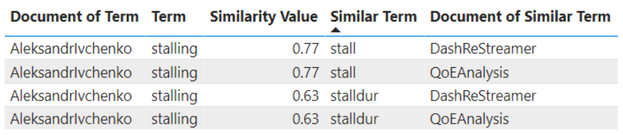
In the QoE 5G/6G showcase, we want to harmonize $Aleksandrlvchenko$ and $LIVENetflix$ datasets so that they can be used in the above-mentioned case study on QoE prediction without manually integrating the datasets. At DIIM's *transform* step, we make them syntactically interoperable by transforming them from Comma-separated Values (CSV) into RDF format with RDF Mapping Language (RML) [3], i.e. converting them into knowledge graphs. Now, these knowledge graphs can be converted into an application-specific format by a RDF-converter if an application doesn't support CSV format. For DIIM's *align & store* step, we set up with the DS3T, which takes two inputs, datasets ($Aleksandrlvchenko$ [12], $DashReStreamer$ [6], and $LIVENetflix$ [2]) and standards ($QoEAnalysis$ [7] and $IEEEIQA$ [17]), and generates a S3O with facts on similarity between terms used in given inputs. DS3T calculates the similarity between two unigram terms by applying String-search Matching (SSM) algorithm implemented by *difflib* [5] and stores the results in S3O. To examine the similarity calculation results, we can load the S3O in an ontology editor, e.g., Protege [10], and query it with SPARQL Protocol and RDF Query Language (SPARQL). DS3T can also export similarity results in CSV format, which can be imported into a more user-friendly tool, like *Excel* or *PowerBI* [9]. Figure 3 compares the number of extracted distinct terms from the inputs. Since manually extracting distinct terms from domain-specific standards and datasets can become challenging for humans, but DS3T proves more feasible.

Figure 4 shows SSM-based similar terms found in datasets and standards for $seqpsnr$ used in $Aleksandrlvchenko$ dataset. Figure 5 shows SSM-based similar terms found in datasets and standards for $stalling$ used in the $Aleksandrlvchenko$ dataset. We annotate the terms $seqpsnr$ and $stalling$ in the $Aleksandrlvchenko$ graph with links to similar terms in the $LIVENetflix$ graph and also annotate the $LIVENetflix$ graph towards $Aleksandrlvchenko$ graph. With a federated SPARQL query using both terms $seqpsnr$ from $Aleksandrlvchenko$ and



Document of Term	Term	Similarity Value	Similar Term	Document of Similar Term
Aleksandrlvchenko	seqpsnr	0.73	psnr	DashReStreamer
Aleksandrlvchenko	seqpsnr	0.73	psnr	IEEEIQA
Aleksandrlvchenko	seqpsnr	0.73	psnr	LIVENetflix
Aleksandrlvchenko	seqpsnr	0.73	psnr	QoEAnalysis
Aleksandrlvchenko	seqpsnr	0.57	psnrhvs	LIVENetflix
Aleksandrlvchenko	seqpsnr	0.57	psnrsqi	LIVENetflix
Aleksandrlvchenko	seqpsnr	0.57	psnrvec	LIVENetflix
Aleksandrlvchenko	seqpsnr	0.53	psnrhyst	LIVENetflix
Aleksandrlvchenko	seqpsnr	0.53	psnrmean	LIVENetflix
Aleksandrlvchenko	seqpsnr	0.53	psnrperc	LIVENetflix

Fig. 4. SSM-based similar terms found by DS3T for term *seqpsnr* in datasets and domain-specific standards



Document of Term	Term	Similarity Value	Similar Term	Document of Similar Term
Aleksandrlvchenko	stalling	0.77	stall	DashReStreamer
Aleksandrlvchenko	stalling	0.77	stall	QoEAnalysis
Aleksandrlvchenko	stalling	0.63	stalldur	DashReStreamer
Aleksandrlvchenko	stalling	0.63	stalldur	QoEAnalysis

Fig. 5. SSM-based similar terms found by DS3T for term *stalling* in datasets and domain-specific standards

$psnr$ $LIVENetflix$, we can extract data from these graphs as a single source. Therefore, they have become harmonized and a single integrated linked data source. Similarly, we annotate $Aleksandrlvchenko$ and $LIVENetflix$ with links to other datasets and domain-specific standards, e.g., $DashReStreamer$, $IEEIQA$, and $QoEAnalysis$, to enhance their interoperability in the QoE domain.

V. Conclusion

We introduced the harmonizing issues that arise when predicting 5G/6G data sourced from different applications, which don't necessarily use the same representation format or label terms differently to label the same data values in their data models. To address these issues, we proposed the DIIM approach and showcased DIIM's application and evaluation of datasets and standards from the QoE domain. We found similar terms in datasets and standards using the SSM algorithm in DS3T. By storing the 5G/6G data in separate knowledge graphs and interlinking their terms based on similarity, we can query the data from any graph and regard it as single-sourced data. In the case study, we have presented the benefit of QoE prediction across datasets, although they don't originate from the same application. In future, we will investigate other NLP algorithms for term alignment.

Acknowledgment

This research was supported by the European Union through the Horizon 2020 Marie Skłodowska-Curie Research and Innovation Staff Exchange Programme "Research Collaboration and Mobility for Beyond 5G Future Wireless Networks (RECOMBINE)" under Grant no. 872857. This work is financed by the European UnionNextGenerationEU through the National Recovery and Resilience Plan of the Republic of Bulgaria, BG-RRP-2.005 -"Twinning" with Project No. BG-RRP-2.005-002 titled "Twinning for Excellence in Research in Sustainable Future Communication Networks in the Context of a Green Economy – GREENBEAT".

References

[1] Abar, T., et al.: Machine learning based qoe prediction in sdn networks. In: 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC). pp. 1395–1400 (2017). https://doi.org/10.1109/IWCMC.2017.7986488

[2] Bampis, C.G., et al.: Towards perceptually optimized end-to-end adaptive video streaming, https://live.ece.utexas.edu/research/LIVE_NFLX_II/live_nflx_plus.html, accessed 17 May 2024

[3] Dimou, A., et. al.: Rml tools (2013), https://rml.io/, accessed 8 June 2024

[4] Duc, T.N., et al.: Convolutional neural networks for continuous qoe prediction in video streaming services. IEEE Access **8** (2020)

[5] Foundation, P.S.: Python difflib (2001), https://docs.python.org/3/library/difflib.html, accessed 8 June 2024

[6] Hodzic., K., et al.: Dashrestreamer, https://github.com/khodzic2/DashReStreamer/tree/master, accessed 17 May 2024

[7] Hodzic, K., et al.: Realistic video sequences for subjective qoe analysis. In: Proceedings of the 13th ACM Multimedia Systems Conference. p. 246–251. MMSys '22, Association for Computing Machinery (2022)

[8] Lu, H., et el.: Edge qoe: Computation offloading with deep reinforcement learning for internet of things. IEEE Internet of Things Journal **7**(10) (2020)

[9] Microsoft: Power bi, https://www.microsoft.com/en-us/power-platform/products/power-bi, accessed 10 June 2024

[10] Noy, N.F., et al.: Protégé-2000: an open-source ontology-development and knowledge-acquisition environment. In: AMIA... Annual Symposium proceedings. AMIA Symposium. vol. 2003, pp. 953–953. American Medical Informatics Association (2003)

[11] Pal, D., et al.: A quantitative approach for evaluating the quality of experience of smart-wearables from the quality of data and quality of information: An end user perspective. IEEE Access **7**, 64266–64278 (2019). https://doi.org/10.1109/ACCESS.2019.2917061

[12] Qoe-assesment, https://github.com/AleksandrIvchenko/QoE-assesment, accessed 17 May 2024

[13] Singh, M., , et al.: Data information interoperability model for iot-enabled smart water networks. In: 2022 IEEE 16th International Conference on Semantic Computing (ICSC). pp. 179–186. IEEE (2022)

[14] Singh, M., et al.: Towards aligning iot data with domain-specific ontologies through semantic web technologies and nlp. SEMANTICS 2023 EU (2023)

[15] Suryanegara, M., et al.: A 5-step framework for measuring the quality of experience (qoe) of internet of things (iot) services. IEEE Access **7**, 175779–175792 (2019). https://doi.org/10.1109/ACCESS.2019.2957341

[16] W3C: Convertertordf, https://www.w3.org/wiki/ConverterToRdf, accessed 10 June 2024

[17] Wang, Z., et al.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004)

F.2 SEMANTICS NLP4KG 2023

Singh, M., Vakaj, E., Rizou, S. and Wu, W., 2023. Towards aligning IoT data with domain-specific ontologies through Semantic Web technologies and NLP. SEMANTICS 2023 EU

Online link to the publication site

Towards aligning IoT data with domain-specific ontologies through Semantic Web technologies and NLP

Mandeep Singh[1,2], Edlira Vakaj[2], Stamatia Rizou[1] and Wenyan Wu[2]

[1]*SingularLogic, Achaias 3, Kifisia 145 64, Greece*
[2]*Birmingham City University, 15 Bartholomew Row, Birmingham B55JU, UK*

Abstract
Internet of Things (IoT) data has the potential to be utilized in many domain-specific applications to enable smart sensing in areas that were not initially covered during the conceptualization phase of these applications. Typically, data collected in IoT scenarios serve a specific purpose and follow heterogeneous data models and domain-specific ontologies. Therefore, IoT data could not easily be integrated into domain-specific applications, as it requires ontology alignment of diverse data models with the end application. This poses a big challenge to semantic interoperability during the integration of IoT data into a pre-established system. In this line, the alignment process is cumbersome and challenging for an ontology engineer, since it requires a manual review of the relevant ontologies that could be aligned with the IoT data. Additionally, before aligning each term used in the IoT data with the concepts defined in the domain-specific ontologies, all similar/related terms in the given ontologies must be considered. In this paper, we propose a solution that supports the alignment process by utilizing semantic web technologies and Natural Language Processing (NLP). Our novel solution proposes an NLP-based term alignment with a similarity score that supports identifying the relevant terms used in IoT data and ontologies and stores the similarity scores among terms based on different similarity algorithms. We showcase our solution by aligning IoT sensor data with the water and IoT domain ontologies.

Keywords
Internet of Things (IoT), Smart Water Network (SWN), Linked Data (LD), ontology, Knowledge Graph (KG), NLP, word2vec, semantic similarity, term alignment

1. Introduction

Around the world, software providers are building Internet of Things (IoT) applications by integrating various solutions and systems that enable remote and continuous monitoring and diagnosis of problems, manage maintenance issues and optimize domain-specific problems by utilizing data-driven and knowledge-driven approaches. The gradual deployment of data-enabled IoT devices, such as smart sensors and actuators, by organizations has offered an

SEMANTICS 2023 EU: 19th International Conference on Semantic Systems, September 20-22, 2023, Leipzig, Germany
msingh@singularlogic.eu;mandeep.singh11@mail.bcu.ac.uk;mandeep.singh@bcu.ac.uk (M. Singh); edlira.vakaj@bcu.ac.uk (E. Vakaj); srizou@singularlogic.eu (S. Rizou); wenyan.wu@bcu.ac.uk (W. Wu)
https://singularlogic.eu (M. Singh); https://www.bcu.ac.uk/computing/about-us/our-staff/edlira-vakaj (E. Vakaj); https://singularlogic.eu (S. Rizou); https://www.bcu.ac.uk/engineering/about-us/our-staff/wenyan-wu (W. Wu)
0000-0001-6381-1140 (M. Singh); 0000-0002-0712-0959 (E. Vakaj); 0000-0002-3683-060X (S. Rizou); 0000-0002-4823-3685 (W. Wu)

opportunity to build a cohesive 'overlay network' in the IoT landscape. Once applications and IoT devices are networked, they can start communicating and exchanging information. However, their interoperability (exchange and making use of information) can not be successful without the syntactic (structure) and semantic (meaning) interoperability of the data/information they share. For instance, at the point of decision-making, a Decision Support System (DSS) relies on the understanding of every bit of data/information that is available from every single IoT device and database, otherwise, it would not be able to advise correctly.

Interoperability of IoT-enabled applications is still a subject of research as cited in [1], one of the main obstacles towards the promotion of IoT adoption and innovation is data interoperability. A key challenge to achieving semantic data interoperability is the alignment of heterogeneous data models among the diverse implementations. Typically, this process needs the calculation of the so-called semantic similarity scores among potential synonymous terms. Despite the existence of similarity calculation algorithms in the literature, this process requires considerable manual work from data workers and analysts to align the application-specific data models to domain-specific ontologies and standards. As a response to this challenge, in this paper, we present our approach that relies on the so-called Semantic Similarity Scoring Ontology (S3O), which automatically identifies the pairs of potential synonymous terms between a given IoT data and existing ontologies and standards and stores their similarity scores to make them available for future reference and reuse. Our approach is expected to significantly improve the efficiency of the cumbersome IoT data alignment process, providing a framework that could be extended to incorporate several ontologies, standards, and similarity score algorithms. To validate our proposed approach, we present a reference implementation of the S3O and will validate its application in an IoT-enabled smart water application.

The rest of the paper is organized as follows. Section 2 discusses the related works in ontology-based semantic interoperability and Natural Language Processing (NLP)-based semantic similarity. Section 3 defines the research questions and proposes our novel approach to address these questions. In section 3.3, the proposed Semantic Similarity Scoring Ontology (S3O) is described. In section 4, we present a showcase of our approach by applying in the water domain for IoT-enabled Smart Water Networks (SWNs).

2. Related Work

Existing approaches to support semantic interoperability in the relevant IoT projects are Smart End-to-end Massive IoT Interoperability, Connectivity and Security (SEMIoTICS) [2] and Bridging the Interoperability Gap of the IoT (BIGIoT) [3]. They propose interoperability solutions that are based on the transitive conversion model for data protocols, e.g., if Message Queuing Telemetry Transport (MQTT) can be converted to/from Constrained Application Protocol (CoAP) and CoAP can be converted to/from Representational State Transfer (REST) than MQTT can be converted to/from REST. Closer to our application scenario in the water domain, a similar interoperability approach is adopted in the water-related projects e.g., Water analytics and Intelligent Sensing for Demand Optimised Management (WISDOM) [4] and Water Enhanced Resource Planning (WatERP) [5], where at first a base ontology (e.g., WISDOM ontology) is aligned with all possible standards and ontologies then it is used to convert from one standard/ontology to another. Overall, these approaches assume that an ontology of IoT data already exists or has already been adopted. However, this assumption may not hold in real-world scenarios, where IoT data may be re-used in different domain applications, each one using different terms to label their data. Therefore, these works do not address the problem considered in this paper, which also includes the automatic identification of potential synonymous terms among existing ontologies and standards.

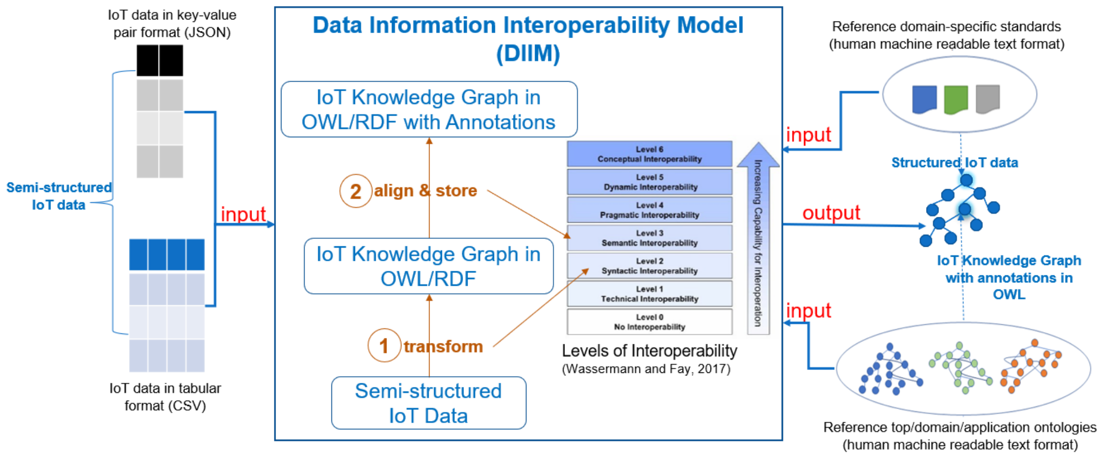
Other related works focus on the alignment of similar terms among different dictionaries. In this context, the alignment process of a dictionary or an ontology aims to align the terms used in data models of different applications to achieve semantic interoperability, i.e., find semantic similarity/relatedness of two terms that originate from different ontologies or data models. Since the initiative of Semantic Web (SW), interest in developing and using ontologies for semantic interoperability has grown. This has also led to research approaches for ontology alignment in recent decades. In [6], a survey and comparison of most of the ontology-based similarity/relatedness measures is presented. It also proposes a feature-based similarity measure based on taxonomical features of an ontology to calculate semantic similarity. In [7], a semantic similarity measure based on information distance for ontology alignment is presented. A recently published paper[8] summarizes and compares ontology matching solutions that use the same type of information, and analyses the challenges in different types of information. The list of similarity calculation algorithms is growing with time, as there is no single algorithm to find the perfect semantic match of terms we will need to consider and reason on the similarity score of all possible algorithms during the alignment process. In this context, less attention has been given to a solution that could assist in managing and reasoning the similarity of all computed similarity-based algorithms.

Our work differs in this respect since it introduces an ontology that encompasses different similarity algorithms and provides an abstraction to store pairs of similar terms and their scores, respectively. It enables an ontology engineer to create a linked Knowledge Graph (KG) from multiple domains, such as environment, healthcare, finance, and government while aligning IoT data from different sources with domain-specific ontologies through NLP.

3. Research Questions & Proposed Approach

3.1. Research Questions

Despite the existing work discussed in Section 2, it is evident that there is a lack of automated tools to support the semantic interoperability of IoT data. In this paper, we argue that a holistic approach for the alignment of IoT data to existing ontologies and data models is necessary to support and promote semantic interoperability across the heterogeneous IoT landscape. This holistic approach should provide automated processes for the calculation of the so-called *similarity score* between different terms (i.e., data labels) and should subsequently create a persistent model, expressed through an ontology that will store the relation between the terms under comparison in the associated metadata (including similarity scores, scoring algorithm etc.). The resulting similarity scoring ontology could then be queried to retrieve similarity scores and support fully automated or semi-automated processes for aligning different semantic models. To this end, the research questions addressed in this paper are summarized below:



IoT data in key-value pair format (JSON)
Semi-structured IoT data
IoT data in tabular format (CSV)
input
Data Information Interoperability Model (DIIM)
IoT Knowledge Graph in OWL/RDF with Annotations
2 align & store
IoT Knowledge Graph in OWL/RDF
1 transform
Semi-structured IoT Data
Level 6 Conceptual Interoperability
Level 5 Dynamic Interoperability
Level 4 Pragmatic Interoperability
Level 3 Semantic Interoperability
Level 2 Syntactic Interoperability
Level 1 Technical Interoperability
Level 0 No Interoperability
Increasing Capability for Interoperation
Levels of Interoperability
(Wassermann and Fay, 2017)
Reference domain-specific standards (human machine readable text format)
input
Structured IoT data
output
IoT Knowledge Graph with annotations in OWL
input
Reference top/domain/application ontologies (human machine readable text format)

Figure 1: Data and Information Interoperability Model (DIIM)

1. Which terms belonging to existing ontologies or data models can characterize a given entity, e.g., object or attribute name, in an IoT dataset?

2. Given a list of relevant terms identified, which is the semantic similarity score (potentially calculated with different algorithms) between two terms that may potentially refer to the same entity?

3. Given a term associated with an entity, which are the terms that show a semantic similarity score above a given threshold?

4. How the semantic similarity score between two terms could be retrieved efficiently to support recurring queries and avoid score re-calculation?

3.2. Proposed Approach

To answer the above research questions, our approach builds on the conceptual Data and Information Interoperability Model (DIIM) introduced in [9]. As shown in figure 1, Data and Information Interoperability Model (DIIM) takes any IoT dataset and domain-specific ontologies as input and generates an annotated IoT KG as output. DIIM transforms the semi-structured IoT dataset into an IoT KG. It aligns the terms/words (labels to name data/values) of IoT the dataset with the terms/words used in ontologies to describe the concepts, relations, instances, and axioms. Found alignments are stored in the IoT KG as annotations of the respective terms to link the terms to related ontologies. All ontologies must be reviewed regardless of the alignment process nature (manual or computer-assisted). In a computer-assisted alignment, the number of ontologies to be reviewed could become high when all possible alignments are searched in a big repository of ontologies. Here, NLP-assisted alignment of the IoT data with domain-specific applications could be beneficial and aid the semantic communication process. In a computer-assisted alignment process, one finds various alignment algorithms to find term-similarity and

Table 1
Parameters and functions used in algorithms

Name	Description
$dataDir$	The URI of the directory or repository where IoT_{data} or $Ontos_{data}$ are held in Unicode Transformation Format – 8-bit (UTF-8) format.
IoT_{data}	Semi-structured IoT data presented in human-machine readable UTF-8 text format and serialized in JavaScript Object Notation (JSON) or Comma-separated Values (CSV) file.
$Ontos_{data}$	Ontologies described in human-machine readable UTF-8 text format and serialized in a text, Extensible Markup Language (XML), HyperText Markup Language (HTML), Resource Description Framework (RDF), or Web Ontology Language (OWL) file.
$corpora$	It is a collection of human-machine readable UTF-8 files that are processed after reading the resource from a given URI of IoT_{data} or $Ontos_{data}$.
IoT_{corpus}	Corpus of IoT data presented in human-machine readable UTF-8 text file.
$Ontos_{corpus}$	Corpus of ontology presented in human-machine readable UTF-8 text file.
f_{corpus}	This function takes IoT_{data} or $Onto_{data}$ as input and transforms the input to a IoT_{corpus} or $Onto_{corpus}$ that is suitable for NLP operations.

surely new algorithms will be developed as algorithms are case-specific, and we may need to consider different algorithm results in different cases. Therefore, we have a construct that can hold information about the applied algorithms, their calculated similarity score and terms with their reference relation. We propose an ontology instead of a database schema because we can link the terms directly to the original ontologies and reason on similarity calculated by the different algorithms. In this line, our approach includes the following steps:

Step 1 Build corpus from IoT data and ontologies: The first step is to create corpora of IoT data and domain-specific ontologies, so that NLP could be applied to align the terms. Algorithm 1 explains the process of building a corpus from a given Uniform Resource Identifier (URI). A NLP corpus is the textual representation of IoT datasets and ontologies without any digits and special characters. It is built while eliminating digits and special characters and keeping words (text) in the IoT datasets and ontologies as they occur. Table 1 lists the parameters and functions used in defined algorithms. Inputs and outputs are represented through URI

Step 2 Finding potential ontologies for alignment: In this step, we shortlist the ontologies that resemble the given IoT data to save computation time. Because there could be many ontologies for processing, some of them could be aligned and some of them not. If the number of given ontologies is significantly low, then this step could be skipped. In the following algorithm 2, we use Latent Semantic Index (LSI) to find the term-similarity-based relationship between the given IoT dataset and ontologies. However, any other kind of similarity algorithm could be applied to shortlist the relevant ontologies. Gensim library [10] creates a LSI model of each ontology and indexes these models. Finally, the index is compared with the LSI model of IoT data to calculate their similarity/relatedness. The threshold to filter ontologies is an optional parameter. If it is not provided in the input, 1.0 as 100% similarity becomes the default value.

Step 3 Build dictionaries and Word2Vector (Word2Vec) models of IoT data and ontologies: As shown in the algorithm 3, we first build dictionaries (list of used terms) and Word2Vec models (representation of used terms as vectors) from given IoT corpus and ontology corpora

Algorithm 1 Build NLP corpus from given URI

Input $dataDir$ /* URI */
Output $corpora$ /* URI */

```
1:  procedure $f_{corpus}$
2:      $corpora \leftarrow List()$
3:      for each $file \in dataDir$ do
4:          fileString ← readFile(file)
5:          lowerString ← lowerCase(fileString)
6:          cleanedString ← removeNumeric(lowerString)
7:          cleanedString ← removeSpecialChars(cleanedString)
8:          cleanedString ← removeStopwords(cleanedString)
9:          $corpus \leftarrow tokenize(cleanedString)$
10:         $corpora.add(corpus)$
11:     end for
12:     return $corpora$
13: end procedure
```

Algorithm 2 Build LSI from IoT data or ontologies and calculate their similarity score

Input path of IoT_{corpus}, $Onto_{corpus}$ and $threshold$ /* optional parameter*/
Output *pathOfLsiSimilarOntos*

```
1:  procedure $f_{lsiSimilarity}$
2:      iotCorpus ← readCorpus(pathOfIoTCorpus)
3:      ontoCorpus ← readCorpus(pathOfOntoCorpus)
4:      ontoLSIModel ← buildLSIModel(ontoCorpus)
5:      ontoLSIIndex ← buildLSIIndex(ontoLSIModel)
6:      iotLSIModel ← buildLSIModel(iotCorpus)
7:      iotOntoLSISimilarity ← calculateSimilarity(ontoLSIIndex,iotLSIModel)
8:      lsiSimilarOntos ← filterOntologies(iotOntoLSISimilarity,threshold)
9:      pathOfLsiSimilarOntos ← writeFile(lsiSimilarOntos)
10:     return pathOfLsiSimilarOntos.
11: end procedure
```

(list of corpus). Then, we train Word2Vec models of ontologies with given IoT corpus.

Step 4 Calculate algorithm-based similarity score of terms in IoT data and ontologies: In this step, we want to calculate the algorithm-based similarity of each term in IoT data with terms used in given ontologies. In algorithm 4 we use Word2Vec similarity and String-search Matching (SSM) algorithms to demonstrate the similarity calculation procedure. Hence, any other similarity calculation algorithms of choice can be added to the procedure to have preferred results. As output, $iotOntoW2vSimilarityList$ and $iotOntoSsmSimiarity$ lists have all terms of IoT data and ontologies, and their calculated Word2Vec and String-search Matching (SSM) similarity score.

Step 5 Build an ontology and store algorithm-based similarity score: In the final step,

Algorithm 3 Build Word2Vec models and dictionaries of IoT data and ontologies

Input $iotCorpora$ and $ontosCorpora$

Output $iotW2vModels$, $iotDicts$, $ontosW2vModels$, $ontosDicts$, $iotOntosW2vModels$, and $allIotOntosW2vModels$

```
 1: procedure $f_{w2v}$
 2:     $ontosW2vModels \leftarrow$ buildW2VModel(ontosCorpora)
 3:     iotW2vModels $\leftarrow buildW2VModel(iotCorpora)$
 4:     $ontosDicts \leftarrow buildDict(ontosW2VModels)$
 5:     $iotDicts \leftarrow buildDict(iotW2VModels)$
 6:     $allIotOntosW2vModels \leftarrow List()$
 7:     for each $iotCorpus \in iotCorpora$ do
 8:         $iotOntosW2vModels \leftarrow List()$
 9:         for each $ontoW2vModel \in ontoW2vModels$ do
10:             $iotOntoW2vModel \leftarrow train(ontoW2vModel, iotCorpus)$
11:             $iotOntosW2vModels.add(ontoWIotW2vModel)$
12:         end for
13:         $allIotOntosW2vModels.add(iotOntosW2vModels)$
14:     end for
15:     return $iotW2vModels$, $iotDicts$, $ontosW2vModels$, $ontosDicts$,
        $iotOntosW2vModels$, $allIotOntosW2vModels$
16: end procedure
```

we build an ontology that stores the information on the similarity of IoT data with an ontology. It also stores the applied algorithm-based similarity score as a relation between two terms that are used in IoT data and ontology. The following algorithm 5 depicts the procedure to populate the ontology with facts and create similarity relationships among entities. S3O contains all terms of IoT data and ontologies, and their calculated Word2Vec and SSM similarity score. Additionally, it also contains the LSI similarity score of given IoT data in relation to ontologies.

3.3. Semantic Similarity Scoring Ontology

In this section, we describe the proposed Semantic Similarity Scoring Ontology (S3O) [11] that stores the terms used in IoT data and ontologies, and stores the similarity score based on the applied various algorithms. Additionally, it stores the directly calculated similarity between any IoT data and an ontology. S3O covers all research questions for aligning the IoT terms with domain-specific ontologies. When S3O is loaded and populated with re facts it will hold information to answer the question from section 3.1.

Figure 2 displays the S3O ontology that was developed in Protégé [12]. S3O ontology starts with an abstract class *Thing*. It has two data properties, *serialization_format* to represent the data representation format and *URI* for identification, that are inherited by its subclasses. Its direct subclasses are *Document*, *Term*, and *Similarity*. *Term* class represents a word or phrase used to describe a thing or express a concept used in any IoT data or an ontology. *Document* class represents an object of the text sequence type. In our approach, we identify *IoT_data* and

Algorithm 4 Calculate algorithm-based similarity score of terms in IoT data and ontologies

Input $iotDicts$, $ontosDicts$ $iotOntosW2vModels$

Output $iotOntoW2vSimilarityList$ and $iotOntoSsmSimiarityLlist$

```
1:  procedure $f_{sim}$
2:      $iotOntoW2vSimilarityList \leftarrow List()$
3:      $iotOntoSsmSimiarityList \leftarrow List()$
4:  /* SSM-based similarity calculation*/
5:      for each $iotDict \in iotDicts$ do
6:          for each $iotTerm \in iotDict$ do
7:              for each $ontoDict \in ontoDicts$ do
8:                  $iotOntoSsmTermSimiarity \leftarrow List()$
9:                  for each $ontoTerm \in ontoDict$ do
10:                     $iotOntoSsmSimilarity \leftarrow calculateSsm(iotTerm, ontoTerm)$
11:                     $iotOntoSsmTermSimiarity.add(iotOntoSsmSimilarity)$
12:                 end for
13:                 $iotOntoSsmSimiarityList.add(iotOntoSsmTermSimiarity)$
14:             end for
15:         end for
16:     end for
17: /* Word2Vec-based similarity calculation*/
18:     for each $iotDict \in iotDicts$ do
19:         for each $iotTerm \in iotDict$ do
20:             $iotOntoW2vTermSimiarity \leftarrow List()$
21:             for each $iotOntosW2vModel \in iotOntosW2vModels$ do
22:                 $iotOntoW2vSimilarity \leftarrow$
23: $calculateW2vSimilarity(iotTerm, iotOntosW2vModel)$
24:                 $iotOntoW2vTermSimiarity.add(iotOntoW2vSimilarity)$
25:             end for
26:             $iotOntoW2vSimilarityList.add(iotOntoW2vTermSimiarity)$
27:         end for
28:     end for
29:     return $iotOntoW2vSimilarityList$, $iotOntoSsmSimiarityList$
30: end procedure
```

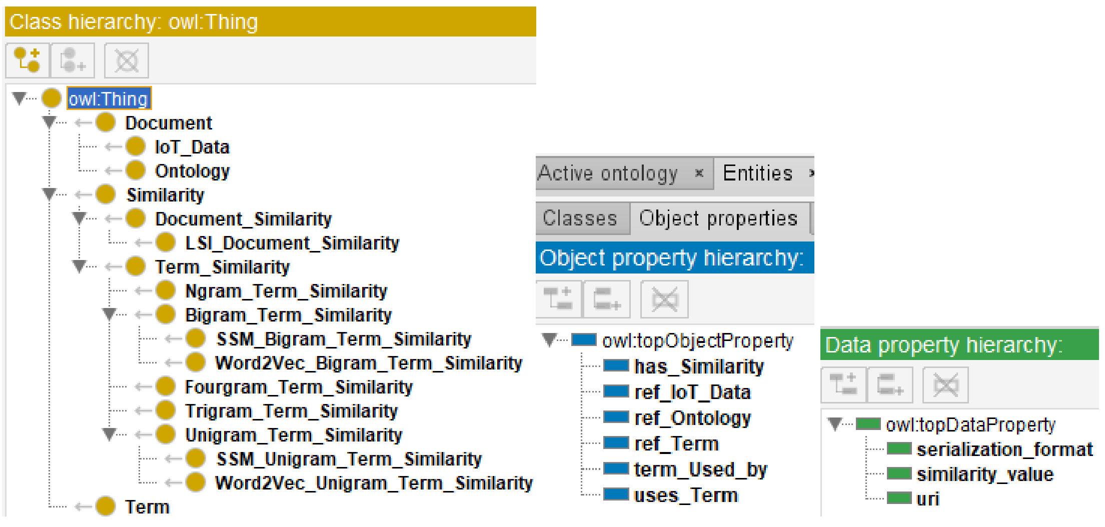
Ontology as document objects for NLP. *IoT_data* contains *metadata* and *measurement data* that can be accessed from any IoT device. The *Ontology* class contains the terms, relations, and properties used to describe the data, information, and knowledge of a specific domain application. The *Similarity* abstract class abstracts over all similarity algorithms, e.g., *SSM_Similarity Word2Vec_Similarity* that are applied to calculate the similarity of documents and terms. It holds *similarity_value* data property to store the similarity score of the documents or terms. We introduce *Ngram* classes to store the similarity of combined terms, e.g., *temperature sensor* (bigrams) that appear as sequences of words in IoT data and ontologies. We have introduced

Algorithm 5 Store the algorithm-based similarity score in an ontology

Input $iotDicts$, $ontosDicts$, $iotOntoW2vSimilarityList$, $iotOntoLSISimilarity$, $iotOntoSsmSimiarity$, and S3O

Output S3O

```
1: procedure $f_{s3o}$
2:     s3o ← loadS3OSchema()
3:     s3o ← createTermRelations(s3o,iotDicts,ontosDicts)
4:     s3o ← createSimilarityRelations(s3o,iotOntoW2vSimilarityList)
5:     s3o ← createSimilarityRelations(iotOntoLSISimilarity)
6:     s3o ← createSimilarityRelations(s3o,iotOntoSsmSimiarity)
7:     pathOfS3o ← writeFile(s3o)
8:     return pathOfS3o.
9: end procedure
```



(a) Class entities **(b)** Object properties **(c)** Data properties

Figure 2: Schema of S3O

object property *has_Similarity* to store the information on similarity-based relationships among *IoT_Data*, *Ontology*, and *Term* classes. *ref_IoT_Data*, *ref_Ontology*, and *ref_Term* are object properties for references. *term_Used_by* and *uses_Term* are inverse object properties to store the information when the term is used by IoT-data or ontologies.

4. Showcase implementation for Smart Water Networks (SWN) applications

To showcase the validity of our approach, we have created a reference implementation [13] of the proposed approach described in Section 3.2 and tested this in a specific scenario for IoT data characterizing a Smart Water Network (SWN) application.

Implementation Setup: The solution was developed in Visual Code Studio[14] Integrated Development Environment (IDE). For the implementation of the approach, Python [15] and many Python-based NLP libraries, e.g. Gensim [10] for Word2Vector (Word2Vec) and Latent Semantic Index (LSI), import Matplotlib[16] for visualization, Pandas[17] for data storage and retrieval, RDFLib [18] for processing S3O, were utilized. Protégé [12], an ontology development environment tool, is used to author and examine S3O ontology facts on term similarity written in Turtle RDF serialization format. Pellet[19] reasoner is used to reason the S3O. Snap SPARQL Protocol and RDF Query Language (SPARQL) Query plugin [20] for Protégé are used to query the S3O facts.

Showcase characteristics: The showcase application takes two inputs, IoT data and domain-specific ontologies of the water and IoT domain. In particular, we consider publicly accessible data sets, related to water quality data. The showcase application processed the IoT data serialized in formats CSV or JSON. More information about the IoT data set characteristics can be found in Table 2. Table 3 holds the information on the ontologies used as input. Input ontologies are from the upper, water, biological, or water domain. The showcase application processed these ontologies from serialization formats, e.g., text, XML, HTML, RDF, or OWL. S3O schema was developed in Protégé and exported as an RDF file. RDFlib [18] was used to generate a graph by loading S3O schema and populating it with the facts/information that is computed by the showcase program on the terms of IoT and ontologies with their relations and algorithm-based similarity score. We use descriptive-naming-pattern $< algorithmname >\_< ontology-name >\_< ontologyterm-name >\_< IoT-name >\_< IoT-term-name >$ for the similarity class instances. For example, in $SSM\_SensorML\_counts\_BristolWaterQuality\_units$ instance, SSM algorithm is used to calculate the similarity of $counts$ from $SensorML$ ontology and $units$ from $BristolWaterQuality$ data.

Table 2
Input: IoT data

File Name	Bristol River water quality.csv [21]	Kaa IoT Data [22]
Format	CSV	JSON
License	Open Government Licence	Public access
Topic	Water Quality	Water temperature
Summary	River quality monitoring data (chemical, physical and bacteriological parameters tested) from 1994.	The data holds the values of a temperature sensor that was simulated locally to send data to the KaaIoT cloud.
Extracted words	access, allison, ammonium, annes, apr, ashton, aug, avenue, avonmouth, badocks, boiling, bottom, bright, briscoe, briscoes, … [155]	auto, bfg, description, fahrenheit, latitude, longitude, mac, measures, model, name, sensor, serial, temperature, timestamp, value, values, water, [17]

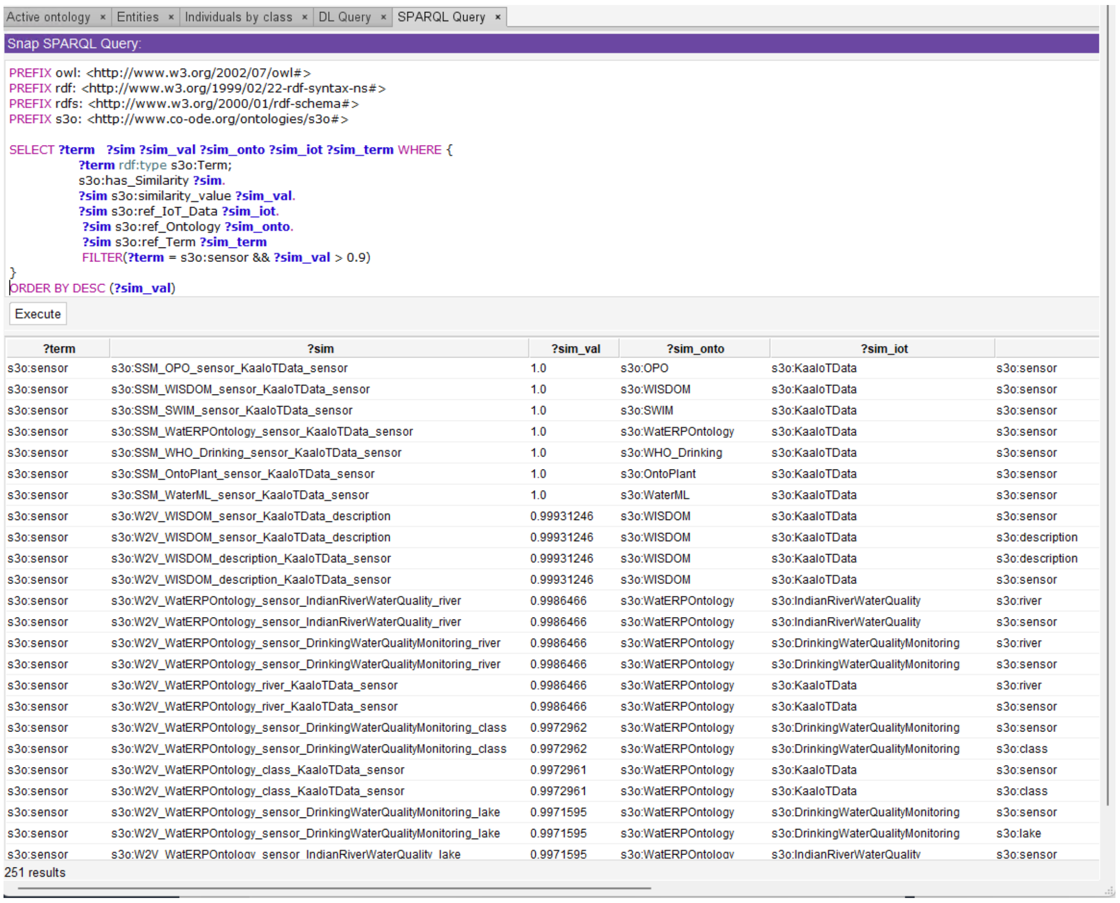
As primary output, the showcase application generates the S3O in RDF turtle format. Other outputs of the showcase application are persisting all calculated information in pickle files and

Table 3
Input: domain-specific ontologies and standards

Name, Domain	**Description**
COSMO[23], upper	It is a foundation ontology that allows it to represent all the basic ('primitive') ontology elements of an application.
DOLCE[24], upper	It is a descriptive ontology for linguistic and cognitive engineering.
GO[25], biological	It provides the foundation for computational analysis of large-scale molecular biology and genetics experiments in biomedical research.
GOIoTP[26], IoT	GOIoTP is developed as part of the INTER-IoT project; it offers modular data structures for the description of entities most commonly appearing in IoT in the context of interoperating various IoT artefacts (platforms, devices, services, etc).
INSPIRE[27], upper	Representation of a set of concepts within a domain and the relationships between those concepts
OntoPlant[28], water	Sottara et al. have extended the SSN ontology to decouple control logic from equipment choices in wastewater treatment plants.
OPO[29], water	It is an Observational Process Ontology for water quality monitoring.
SAREF[30], IoT	It is a shared consensus model that facilitates the matching of existing assets in the smart applications domain.
SensorML[31], IoT	It provides a robust and semantically-tied means of defining processes and processing components associated with the measurement and post-measurement transformation of observations.
SSN[32], IoT	This ontology describes sensors and sensor networks, for use in web applications, independent of any application domain.
SOSA[33], IoT	It can be used directly for lightweight applications, or provide the basis for additional specialization and axiomatization in vertical and horizontal extensions.
SWIM[34], water, IoT	It is developed by Aquamatix for the Device-level IoT semantic model for the water industry
WaterML[35], water	WaterML2 is a new data exchange standard in Hydrology to exchange many kinds of hydro-meteorological observations and measurements. It harmonizes a number of exchange formats for water data with relevant OGC and ISO standards.
WatERPOntology[5], water	It is developed by EURECAT-WatERP. It is a lightweight ontology of generic concepts for water sensing and management.
WHO_Drinking[36], water	WHO standard guidelines to maintain the relevance, quality and integrity of the Guidelines for drinking-water quality (GDWQ), whilst ensuring their continuing development in response to new, or newly-appreciated, information and challenges.
WISDOM[4], water	It is developed by Cardiff University for the cyber-physical and social ontology of the water value chain.

creating bar charts of the top 10 Word2Vec-based on similar terms of each IoT-term.

Querying similarity in S3O: Figure 3 shows an output of querying the S3O with facts [11] for the term "sensor". In particular, a SPARQL query [11] is executed in the Snap SPARQL [20], Query tab of Protégé provides results on similar terms to term "sensor" as shown in the figure 3. *Protégé*[12], an ontology development environment tool, is used to load S3O facts on term



```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX s3o: <http://www.co-ode.org/ontologies/s3o#>

SELECT ?term  ?sim ?sim_val ?sim_onto ?sim_iot ?sim_term WHERE {
            ?term rdf:type s3o:Term;
            s3o:has_Similarity ?sim.
            ?sim s3o:similarity_value ?sim_val.
            ?sim s3o:ref_IoT_Data ?sim_iot.
             ?sim s3o:ref_Ontology ?sim_onto.
             ?sim s3o:ref_Term ?sim_term
             FILTER(?term = s3o:sensor && ?sim_val > 0.9)
}
ORDER BY DESC (?sim_val)
```

?term	?sim	?sim_val	?sim_onto	?sim_iot	
s3o:sensor	s3o:SSM_OPO_sensor_KaaloTData_sensor	1.0	s3o:OPO	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WISDOM_sensor_KaaloTData_sensor	1.0	s3o:WISDOM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_SWIM_sensor_KaaloTData_sensor	1.0	s3o:SWIM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WatERPOntology_sensor_KaaloTData_sensor	1.0	s3o:WatERPOntology	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WHO_Drinking_sensor_KaaloTData_sensor	1.0	s3o:WHO_Drinking	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_OntoPlant_sensor_KaaloTData_sensor	1.0	s3o:OntoPlant	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:SSM_WaterML_sensor_KaaloTData_sensor	1.0	s3o:WaterML	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WISDOM_sensor_KaaloTData_description	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WISDOM_sensor_KaaloTData_description	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:description
s3o:sensor	s3o:W2V_WISDOM_description_KaaloTData_sensor	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:description
s3o:sensor	s3o:W2V_WISDOM_description_KaaloTData_sensor	0.99931246	s3o:WISDOM	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_IndianRiverWaterQuality_river	0.9986466	s3o:WatERPOntology	s3o:IndianRiverWaterQuality	s3o:river
s3o:sensor	s3o:W2V_WatERPOntology_sensor_IndianRiverWaterQuality_river	0.9986466	s3o:WatERPOntology	s3o:IndianRiverWaterQuality	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_river	0.9986466	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:river
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_river	0.9986466	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_river_KaaloTData_sensor	0.9986466	s3o:WatERPOntology	s3o:KaaloTData	s3o:river
s3o:sensor	s3o:W2V_WatERPOntology_river_KaaloTData_sensor	0.9986466	s3o:WatERPOntology	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_class	0.9972962	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_class	0.9972962	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:class
s3o:sensor	s3o:W2V_WatERPOntology_class_KaaloTData_sensor	0.9972961	s3o:WatERPOntology	s3o:KaaloTData	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_class_KaaloTData_sensor	0.9972961	s3o:WatERPOntology	s3o:KaaloTData	s3o:class
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_lake	0.9971595	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:sensor
s3o:sensor	s3o:W2V_WatERPOntology_sensor_DrinkingWaterQualityMonitoring_lake	0.9971595	s3o:WatERPOntology	s3o:DrinkingWaterQualityMonitoring	s3o:lake
s3o:sensor	s3o:W2V_WatERPOntology_sensor_IndianRiverWaterQuality_lake	0.9971595	s3o:WatERPOntology	s3o:IndianRiverWaterQuality	s3o:sensor

251 results

Figure 3: Find terms similar to the term “sensor” in ontologies and IoT datasets

similarity written in turtle RDF serialization format. *pellet*[19] reasoner is used to reason the S3O. *Snap SPARQL Protocol and RDF Query Language (SPARQL) Query* plugin for *Protégé* is used to query the S3O facts.

Discussion: In this section, we have described the reference implementation of our approach presented in Subsection 3.2 and have demonstrated its applicability considering public IoT data sets and relevant domain-specific ontologies in the water domain. More specifically, our approach uses NLP as the core method to define and calculate the semantic similarity scores.

We start with term/word alignment because in description logic words are used to describe and to label/name values in datasets and entities in ontologies. Therefore, we can also do structural alignment based on similar/related words when we could deduce structural alignment through N-grams, e.g., *’water has ph’* can be deduced to align with the statement *’water contains ph’* if *’has’* and *’contains’* can be aligned. While following the Data and Information Interoperability Model (DIIM) approach, we want to annotate the terms in IoT KG with similar words found in different domain-specific ontologies and standards. To achieve this, we first try to find similar words with this approach and annotate them when the similarity score is higher than the given threshold. At the current stage, only a similarity score with a value of 1 is automatically accepted for auto-annotation and all other similar terms with lower scores are suggested for a human review, which poses a challenge to the manual effort in the alignment process. However, the proposed approach to use NLP-based techniques and accommodate different algorithms in combination with S3O significantly ease the alignment process.

The experiments showed that our approach performs well and manages to create the S3O and subsequently calculate and store the similarity scores for the identified terms in the IoT data sets. We propose S3O instead of database schema because we want to use import-feature to link the terms directly to the original ontologies and IoT data converted into KGs and reason on similarity calculated by the different algorithms as federated KG as whole. S3O covers the current requirements and is subject to extension for new requirements.

Further implementation and experiments are planned to measure the performance of our solution (in terms of time) as well as in terms of precision (i.e., compare the output of our solution with respect to the identification of similar terms using a manual process in small-scale scenarios). Overall, we consider that our solution is an initial step towards systematizing and automating the process of semantic interoperability in the heterogeneous IoT landscape.

5. Conclusion

In this paper, we propose a *novel methodology based on Semantic Web technologies (OWL, KG, RDF, and Linked Data (LD)) and NLP (LSI, Word2Vec and Ngram similarity) to discover related ontologies and align terms of IoT data with these ontologies.* Further, our work contributes to developing a new ontology, the Semantic Similarity Scoring Ontology (S3O). The proposed S3O holds a similarity score of terms based on the similarity evaluation of the applied algorithms. This ontology can be easily extended to include the evaluation results of other algorithms. This way, we do not support a specific algorithm to align terms, rather believe that all alignment algorithms could become relevant at a certain point. Therefore, we store the similarity scores of all alignment algorithms in S3O and an ontology engineer can query the similarity scores explored the linked terms from different ontologies and decide to do the final alignment/mapping. We have showcased the validity of our approach in an IoT-enabled smart water application, however, the proposed solution is extensible in terms of adding new ontologies for alignment and considering newly developed term-alignment algorithms. In our future work, we plan to extend the implementation of the showcase by adding more NLP-based similarity algorithms to support the alignment of the IoT data with the ontologies of the cross-domain applications.

References

[1] N. Kalatzis, G. Routis, Y. Marinellis, M. Avgeris, I. Roussaki, S. Papavassiliou, M. Anagnostou, Semantic interoperability for IoT platforms in support of decision making: An experiment on early wildfire detection, Sensors (Switzerland) 19 (2019). doi:10.3390/s19030528.

[2] G. M. Milis, C. G. Panayiotou, M. M. Polycarpou, Semiotics: Semantically enhanced iot-enabled intelligent control systems, IEEE Internet of Things Journal 6 (2019) 1257–1266.

[3] A. Bröring, S. Schmid, C. Schindhelm, A. Khelil, S. Käbisch, D. Kramer, D. Le Phuoc, J. Mitic, D. Anicic, E. Teniente, Enabling iot ecosystems through platform interoperability, IEEE Software 34 (2017) 54–61.

[4] S. Howell, Y. Rezgui, T. Beach, Automation in Construction Integrating building and urban semantics to empower smart water solutions, Automation in Construction 81 (2017) 434–448. doi:10.1016/j.autcon.2017.02.004.

[5] G. Anzaldi Varas, W. Wu, A. Abecker, E. Rubión Soler, A. Corchero, A. Hussain, M. QUENZER, Integration of water supply distribution systems by using interoperable standards to make effective decisions, 2014.

[6] D. Sánchez, M. Batet, D. Isern, A. Valls, Ontology-based semantic similarity: A new feature-based approach, Expert Systems with Applications 39 (2012) 7718–7728. URL: https://www.sciencedirect.com/science/article/pii/S0957417412000954. doi:https://doi.org/10.1016/j.eswa.2012.01.082.

[7] Y. Jiang, X. Wang, H.-T. Zheng, A semantic similarity measure based on information distance for ontology alignment, Information Sciences 278 (2014) 76–87. URL: https://www.sciencedirect.com/science/article/pii/S0020025514003053. doi:https://doi.org/10.1016/j.ins.2014.03.021.

[8] X. Liu, Q. Tong, X. Liu, Z. Qin, Ontology matching: State of the art, future challenges, and thinking based on utilized information, IEEE Access 9 (2021) 91235–91243. doi:10.1109/ACCESS.2021.3057081.

[9] M. Singh, W. Wu, S. Rizou, E. Vakaj, Data information interoperability model for iot-enabled smart water networks, in: 2022 IEEE 16th International Conference on Semantic Computing (ICSC), 2022, pp. 179–186. doi:10.1109/ICSC52841.2022.00038.

[10] R. Řehůřek, Gensim, 2009. URL: https://radimrehurek.com/gensim/index.html, accessed 2 Mar 2022.

[11] M. Singh, Semantic similarity scoring ontology, 2023. URL: https://github.com/mxrandhawa/s3o, accessed: 04/05/2023.

[12] N. F. Noy, M. Crubézy, R. W. Fergerson, H. Knublauch, S. W. Tu, J. Vendetti, M. A. Musen, Protégé-2000: an open-source ontology-development and knowledge-acquisition environment., in: AMIA... Annual Symposium proceedings. AMIA Symposium, volume 2003, American Medical Informatics Association, 2003, pp. 953–953.

[13] M. Singh, Iot showcase implementation with s3o ontology, 2023. URL: https://github.com/mxrandhawa/iotshowcase, accessed: 08/05/2023.

[14] Microsoft©2022, Visual code studio, 2015. URL: https://code.visualstudio.com/, accessed 7 Feb 2022.

[15] P. S. Foundation, Python™, 2001. URL: https://www.python.org/, accessed 7 Feb 2022.

[16] Matplotlib Development team, Matplotlib: Visualization with python, 2003. URL: https://matplotlib.org/, accessed 8 Feb 2022.

[17] NumFOCUS, Inc., Pandas: Visualization with python, 2020. URL: https://pandas.pydata.org//, accessed 17 Dec 2022.

[18] RDFLib Team, Rdflib, 2009. URL: https://rdflib.readthedocs.io/en/stable/, accessed 8 Feb 2022.

[19] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, Y. Katz, Pellet: A practical owl-dl reasoner, Journal of Web Semantics 5 (2007) 51–53.

[20] M. Horridge, M. Musen, Snap-sparql: a java framework for working with sparql and owl, in: Ontology Engineering: 12th International Experiences and Directions Workshop on OWL, OWLED 2015, co-located with ISWC 2015, Bethlehem, PA, USA, October 9-10, 2015, Revised Selected Papers 12, Springer, 2016, pp. 154–165.

[21] B. C. Council, Bristol river water quality, 2010. URL: https://www.data.gov.uk/dataset/dd6658fc-7d1a-4ab2-9ea4-6aa936d21608/bristol-river-water-quality, accessed: 26/09/2022.

[22] M. Singh, Kaa cloud iot sensor data, 2021. URL: https://www.kaaiot.com/, accessed: 27/12/2021.

[23] P. Cassidy, Cosmo, 2020. URL: http://micra.com/COSMO/, accessed 23 Jan 2021.

[24] N. Guarino, A. Gangemi, Dolce-ultralite, 2017. URL: www.loa.istc.cnr.it/dolce/overview.html, accessed 23 Jan 2021.

[25] G. Consortium, Gene ontology, 2020. URL: http://geneontology.org/.

[26] P. A. o. S. Paweł Szmeja, Systems Research Institute, Generic ontology for iot platforms, 2018. URL: https://inter-iot.github.io/ontology/.

[27] I. Maintenance, Implementation, Inspire ontology, 2015. URL: https://inspire.ec.europa.eu/glossary/Ontology, accessed 10 Dec. 2022.

[28] D. Sottara, J. C. Coreale, T. Spetebroot, D. Pulcini, D. Giunchi, F. Paolucci, L. Luccarini, An ontology-based approach for the instrumentation, control and automation infrastructure of a wwtp, 2014. URL: https://www.academia.edu/download/53892589/An_ontology-based_approach_for_the_instr20170718-3232-1pm8ts1.pdf.

[29] X. Wang, H. Wei, N. Chen, X. He, Z. Tian, An observational process ontology-based modeling approach for water quality monitoring, Water 12 (2020) 715. doi:10.3390/w12030715.

[30] E. STF-578, Smart applications reference ontology, and extensions, 2020. URL: https://saref.etsi.org/, accessed 10 Dec. 2022.

[31] OGC, Sensor model language (sensorml), 2019. URL: https://www.ogc.org/standards/sensorml, accessed 1 Dec. 2022.

[32] W. OGC, Semantic sensor network ontology, 2017. URL: https://www.w3.org/TR/vocab-ssn/, accessed 9 March 2022.

[33] W3C, Sensor-observation-sampling-actuator ontology, 2016. URL: https://www.w3.org/2015/spatial/wiki/SOSA_Ontology, accessed 19 August 2022.

[34] L. Reynolds, The swim concept: an open interoperable data standard, in: IET Conference Proceedings, The Institution of Engineering & Technology, 2013.

[35] H. D. W. g. OGC, Waterml2, 2014. URL: http://waterml2.org/, accessed 23 Jan 2021.

[36] W. H. Organization, Guidelines for drinking-water quality (gdwq), 2021. URL: https://www.who.int/publications-detail-redirect/9789240045064, accessed 10 Jan. 2022.

F.3 IEEE ICSC 2022

M. Singh, W. Wu, S. Rizou and E. Vakaj, "Data Information Interoperability Model for IoT-enabled Smart Water Networks," 2022 IEEE 16th International Conference on Semantic Computing (ICSC), Laguna Hills, CA, USA, 2022, pp. 179-186, doi: 10.1109/ICSC52841.2022.00038. keywords: Semantics;Water quality;Ontologies;Syntactics;Data models;Service-oriented architecture;Internet of Things;Data Interoperability;Syntactic harmonization;Semantic Model Generation and Alignment;Smart Water Networks;IoT/WoT;Ontology;Representation Standard;Water Quality Monitoring

Online link to the publication site

Data Information Interoperability Model for IoT-enabled Smart Water Networks

Mandeep Singh
Birmingham City University
SingularLogic
msingh@singularlogic.eu

Wenyan Wu
Birmingham City University
Birmingham, B47XG, UK
wenyan.wu@bcu.ac.uk

Stamatia Rizou
SingularLogic
Kifissia, 14564, Greece
srizou@singularlogic.eu

Edlira Vakaj
Birmingham City University
Birmingham, B47XG, UK
edlira.vakaj@bcu.ac.uk

***Abstract*—Syntactic and semantic interoperability is a fundamental requirement for the success of the Internet of Things (IoT)-enabled Smart Water Networks (SWNs). Still, whilst consuming publicly accessible IoT data, the syntactic and semantic representation of the collected data poses challenges for the success of pervasive and ubiquitous sensing in the water domain. Challenges include the heterogeneity of data representation formats, semantic models, and the adoption of domain-specific standards and ontologies. These challenges emphasise the requirement for enhanced interoperability in SWNs. To address this, we propose a Data and Information Interoperability Model (DIIM) by combining the Semantic Web technologies, widely known for overcoming interoperability issues, and Model-driven architecture (MDA) approach. DIIM facilitates syntactic interoperability by serialization conversion and adoption of domain-specific standards as well as semantic interoperability of metadata by aligning the semantic models of IoT and Smart Water Network (SWN) applications. Furthermore, it automatically creates an ontology as a semantic model if it is missing and adds references to existing domain-specific ontologies as annotation in their models. We evaluate DIIM methodology by applying it to a real-world use case of IoT-enabled applications for water quality monitoring.**

***Index Terms*—Data Interoperability, Syntactic harmonization, Semantic Model Generation and Alignment, Smart Water Networks, IoT/WoT, Ontology, Representation Standard, Water Quality Monitoring**

I. Introduction

Due to global water scarcity and water quality crisis, each year, billions of dollars are being invested in integrated water resource management to meet the water demand with the supply of affordable, sustainable, and pure water to consumers [1]. Water utilities build Smart Water Networks (SWNs) by integrating various solutions and systems that enable remote and continuous monitoring and diagnosis of problems, manage maintenance issues and optimise the water distribution network by utilising data-driven methods.

The deployment of data-enabled Internet of Things (IoT)/Web of Things (WoT) at the physical layer of a SWN, e.g. smart sensors, valve controllers, and cooling units by water utilities have offered an opportunity to build a cohesive *overlay network* in SWN. Networked IoT offer utilisation of the IoT/WoT data (as numbers, symbols, text, images, sound recordings, unit values, etc.) and information (contextualized data) to enable smart sensing beyond the initial coverage areas of a SWN. However, the SWN applications, e.g. leakage detection, water distribution, water quality management, and customer metering, must interoperate with the IoT/WoT before they can run data-driven information analysis and make decisions or operate appropriately in real-time at the top layer of a SWN.

IoT-enabled SWN applications have data/information interoperability if data providers (IoT) and data consumers (SWN applications) can exchange, deliver, or use data through sending messages in a coordinated way. Such messages must typically be transformed at each interoperability level [2], either by the sender or receiver, to a construct that can be readily consumed and thus understood by the receiver; this process is often referred to as message alignment [3]. Wasserman and Fay compare The Levels of Conceptual Interoperability Model (LCIM) [2] with the Open Systems Interconnection (OSI) model [4] and state that any of OSI based communication technologies can enable syntactic and semantic interoperability, though achieving the semantic interoperability at level 3 remains a major challenge in the Semantic Web [5].

Furthermore, diversity in data syntax formats, e.g. Comma-separated Values (CSV), JavaScript Object Notation (JSON), Extensible Markup Language (XML), and Resource Description Framework (RDF), to serialize IoT data before it can be sent over an IoT data protocol, e.g. Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP), and Hypertext Transfer Protocol (HTTP), leaves IoT software developers with a tough decision on interoperability, i.e., support all possible formats that costs a lot of resources or focus on one or two formats. Similar is the situation for the developers of the SWN applications as they must also implement all possible interfaces to deserialize the received IoT data. In [6], Howell et al. recount the reason of interoperability failure from smart grid [7] as *(i) lack of machine communication protocols, (ii) lack of common data formats, and (iii) lack of the common meaning of exchanged content.* In IoT-enabled SWNs, some of the existing solutions use MQTT or CoAP as a common communication protocol and WaterML2 as a common data format. However, semantic interoperability aspects are not sufficiently addressed by these standards. Thus, IoT-enabled applications require semantic models to understand the content of the exchanged data.

Both industry and academia have acknowledged the benefits of semantic models in the field of semantic web technologies through the World Wide Web Consortium (W3C) 'semantic web stack', which shows ontologies playing a critical role [8]. Groß et al. elaborate that an ontology enables a representation of the data machine-processable, ultimately allowing reasoning, generation of new knowledge, and automatic detection of inconsistencies in the semantic models [9]. However, regarding data semantics, a similar challenge to adopt domain-specific standards and formats remains for the developers of the IoT/WoT and SWN applications, as there are so many ontologies in the IoT and the water domain. Additionally, limited knowledge of existing and appropriate domain-specific ontologies and the cumbersome task of developing or adopting an ontology result in either no application ontology or referencing a single ontology, which limits the interoperability potential.

Hence, a solution to share and use IoT/WoT data should be based on the data interoperability and without necessarily tightly coupling of IoT/WoT with the interfaces of SWN applications at the time of development. The interoperability of IoT-enabled applications is still a subject of research, as cited in [10]; *data interoperability is one of the main obstacles to promote IoT adoption and innovation.* This paper proposes a domain-specific Data and Information Interoperability Model (DIIM) for IoT-enabled applications by utilising the Semantic Web technologies and Model-driven approach. To enable interoperability between IoT applications, DIIM can automatically build a semantic model of given IoT data, adopt domain-specific standards, align to domain-specific ontologies, and transform data into various data serialization formats according to an application's requirements.

We organise the rest of the paper as follows. Section II presents related work on interoperability and semantic modelling in the IoT and water domain. In Section III, we list the identified interoperability challenges of IoT-enabled SWNs. Section IV elaborates the DIIM methodology and its application to a case study of IoT-enabled water quality monitoring. Section V concludes the work presented in this paper.

II. Related Work

This section reviews the related work on interoperability in IoT and water domain projects. It also discusses the semantic modelling in the water domain.

A. Interoperability in SWNs and IoT applications

Hatzivasilis et al. compare the major European Union (EU) funded IoT research projects, BigIoT, OpenIoT, INTER-IoT, and SEMIoTICS in terms of interoperability features. Among the IoT projects, SEMIoTICS not only offers interoperability at four levels but goes 2 steps ahead of its competitors. It utilises semi-automatic pattern-driven techniques for the cross-domain operation and interaction of applications [11].

For IoT ecosystems, BigIoT introduces five interoperability patterns: *(i) cross-platform access, (ii) cross-application domain access, (iii) platform independence, (iv) platform-scale independence, and (v) higher-level service facades.* Although these patterns help to reuse data and services from different platforms of an ecosystem, there is a need for automatic search and orchestration of services [12].

In comparison with IoT interoperability approaches, the Water Enhanced Resource Planning (WatERP) framework from the water domain proposes an architecture that harmonizes the communication between systems. These systems control, monitor, and manage the water supply distribution chain by using a Service Oriented Architecture (SOA)-Multi-Agent System (MAS) approach together with a knowledge base driven by the Water Management Ontology (WMO) [13]. This approach integrates and utilises innovative technologies, SOA, web services, MAS, and semantic web languages to handle the interoperability issue of monitoring and decision-making applications within SWNs. The framework also offers a standardized SOA-MAS-based interface and communication interpretation through WMO. Additionally, through SOA-MAS-based approach, intelligent orchestration of system functionalities within the architecture is achieved, as agents can be conceptualized with Believe Desire Intention (BDI) [14] model to become autonomous and cooperative to achieve their declarative and procedural goals [15].

The Water analytics and Intelligent Sensing for Demand Optimised Management (WISDOM) project enables the interoperability of things and software in smart water networks through a software platform that utilises ontology for semantics and web services for web-enabled sensors to integrate business operations across the water value chain. They define a *water value chain* as the artefacts, agents, and processes involved in delivering potable water to consumers from natural water sources and safely disposing of foul and runoff wastewater. Their interoperability approach integrates existing data models, which are formalized in different data formats and use heterogeneous domain perspectives. They intersect existing models and align them with the WISDOM ontology that is used as a common ontology to support the data interoperability across the existing models. They promote interoperability through semantic web technologies and by performing a schema conversion from a knowledge base of devices instantiated within the WISDOM ontology into another model, e.g. Smart Appliances REFerence (SAREF), Infrastructure for spatial information in Europe (INSPIRE), Industry Foundation Classes 4 (IFC4), Semantic Water Interoperability Model (SWIM) [6].

B. Semantic modelling in the water domain

A chronological list of semantic models in the water domain is presented by Howell [16]. There are many existing ontologies and standards. However, there is no standardized common ontology in the water domain as compared to the Gene Ontology (GO) [17] that represents the knowledge base of genes. Maedche and Staab argue that mapping existing ontologies will be easier than creating a common ontology because a smaller community is involved in the process. Their further argumentation is, ontologies must be normalized to a uniform representation to eliminate syntactic differences and make semantic differences between the source and the target ontology more apparent [18].

Figure 1 depicts the common ontologies, formats, and standards that are being used in the water domain to conceptualize the domain knowledge. All listed standards are based on markup language XML, and all listed ontologies are defined in Web Ontology Language (OWL). As XML, RDF, and OWL are Semantic Web technologies, and RDF is built on XML and OWL is built on RDF [19], we can use OWL as an interoperable language to overcome the syntactic and semantic heterogeneity of data and information that is represented in any of the listed water domain standards and ontologies. However, if IoT data is represented in another standard than Semantic Web technologies and domain-specific ontology is referred to, we require translation and mapping to achieve interoperability. Finally, we can conclude that a common standardized ontology does not exist in the water and IoT domain. However, there have been attempts to recycle and merge existing standards and ontologies rather than build something from scratch.

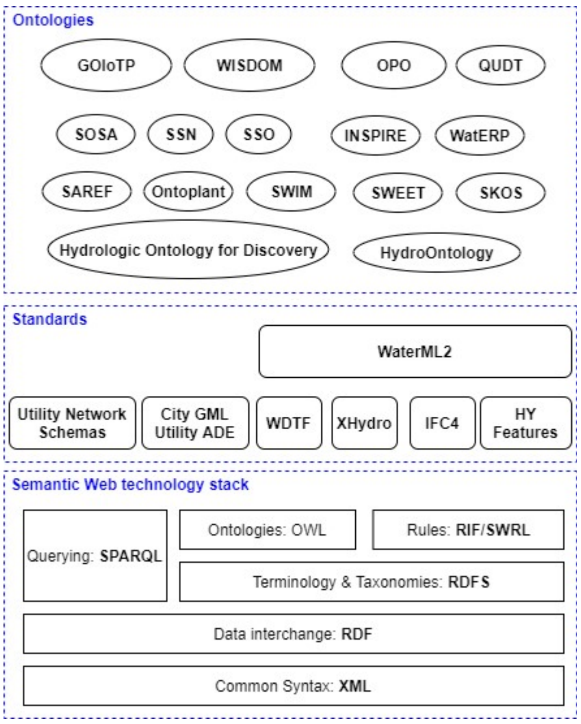


Fig. 1. Semantic Web technologies based standards and ontologies in the IoT and the water domain

WISDOM project proposes a semantic model for intelligent water sensing and analytics through a domain ontology created using Izssa's ontology integration approaches [20]. The WISDOM model integrates heterogeneous data sources and various ontologies; thus, it identifies the necessity of validation. At first, validate the domain model as an accurate, sufficient, and shared conceptualization of the domain by domain experts, then validate the ontology instantiation and deployment as a web service within a cloud-based platform through software testing [21].

In the INTER-IoT project, Generic Ontology for IoT Platforms (GOIoTP) is developed as a core ontology and reference meta-data model for IoT platforms. It offers modular data structures for describing entities like device structure, platform, observation, actuation, units, measurements, location, service, and user. Generic Ontology for IoT Platforms Extended (GOIoTPex), also developed in the INTER-IoT project, extends and fills the stub classes/concepts from GOIoTP with more specific classes, properties and individuals.

Both top-level ontologies, GOIoTP and WISDOM, bring complementary concepts and domain knowledge. Therefore, an IoT-enabled SWN will require such ontologies to build a semantic model representing data and information of IoT and water domain. Hence, an issue of ontology integration arises, Izssa's ontology integration approaches [20] can solve: *(i) Ontology mapping* to establish correspondence rules between concepts of two ontologies. *(ii) Ontology alignment* to bring two or more ontologies into a mutual agreement. *(iii) Ontology transformation* to change the structure of the ontology to make it compliant with another. *(iv) Ontology fusion* to build a new ontology from two or more existing ones.

III. Interoperability challenges of IoT-enabled SWNs

While reviewing the related work, it becomes clear that interoperability is still a hot topic. The IoT-enabled applications require interoperability at syntactic (data exchange) and semantic (understanding the meaning of the exchanged data) layers to overcome interoperability issues. Most of the interoperability solutions for IoT-enabled applications are developed with the vertical application approach for smart networking and undermining the potential brought through the cross-domain integration of IoT-solutions in the water domain. For example, Industry 4.0 [22] cannot yield the potential of interconnected IoT if the data sent and received by IoT cannot be understood and used by consumer applications. Some of the challenges that need to be addressed to achieve IoT-data interoperability in the water domain are:

- **Transformation of data in various representation and encoding formats:** SWN application developers generally encode or represent data in their favourite data format, e.g. XML or JSON, and also publish data in their encoded format, e.g. UTF-8 or Latin1. Therefore, the data encoding format of one application could be different from the data encoding format of another application that wants to share data. So, one of the applications must have the translator/(de)serialization ability for each other's data representation and encoding format, and this must be implemented and deployed. However, when many applications want to interoperate and have different data representation and encoding formats, it will become challenging.
- **No standardized domain-specific ontology:** In the water domain, there are too many domain-specific and application-specific ontologies and no common standard water ontology. One reason is an ontology models only

a specific aspect of the real world based on the ontology engineer aim, therefore the ontology is limited by the interest and viewpoint of an ontology engineer. Another reason is that the reuse of existing ontologies is not widely practised because extending existing or merging an ontology with own ontology is a complex ontology engineering task. Therefore, each application tends to build its application-specific ontology. Additionally, to build a common SWN ontology, a consortium of organizations and companies from the public and private sectors is required. In this context, applications fail to adopt existing ontologies; thus, they cannot semantically understand the data shared by other applications without semantic mappings.

- **Adoption of the water domain-specific standards:** WaterML2 [23] is an XML-based standard that is developed by Open Geospatial Consortium (OGC) group (CSIRO, CUAHSI, USGS, BOM, NOAA, KISTERS, and others) to standardize time series data (hydro-meteorological observations and measurements) exchange in Hydrology. However, data modelled for IoT is not always in WaterML standard, as most of the IoT or SWN application developers do not know at development time which of standards they need to support ad-hoc utilization of the data.
- **Generation of missing ontology and vocabulary:** Semantic interoperability requires an understanding of the data through conceptual knowledge, generally represented through ontologies or vocabulary. However, these ontologies are mostly missing for the existing databases because ontology development remains a cumbersome manual task. In addition, developers must develop these ontologies manually while considering the schema of the represented data in different formats since schema helps to identify the structural organization of data.

These challenges demand a framework that can generate ontology from existing data while reusing the existing ontologies and adopting the existing standards, and facilitate syntactic and semantic interoperability of data and information between IoT and SWN applications.

IV. Data Information Interoperability Model

In this section, the first subsection outlines the Data and Information Interoperability Model (DIIM) that uses Model-driven architecture (MDA) approach and Semantic Web technologies to ensure syntactic and semantic interoperability of the IoT data. The second subsection describes a case study on IoT-enabled water quality monitoring. The third subsection demonstrates the application of DIIM to the case study.

A. DIIM architecture and methodology

From a technical point of view, figure 2 reveals the key components and their classification that are designed as web services for the loosely coupled DIIM architecture.

- **IoT components:** *Publication/Subscription Manager* offers the service to publish or subscribe IoT data. Every data subscription or publication creates a profile with the semantic model in the Knowledge Base (KB). *IoT Locator* scans in Thing Description (TD) of IoT and returns the list of IoT that matches the subscription. IoT data collector collects the data publisher's endpoint and delivers to KB. IoT data pusher retrieves the processed data from KB and pushes it to the subscriber's endpoint.
- **Data analysis and Task management components:** *Data Analyser and Flagger* components analyse the collected IoT data and create a publisher profile with a semantic model. Additionally, it sets a flag for every syntactic and semantic difference between the publisher's and subscriber's semantic model. *Task generator* creates tasks for every flag to achieve syntactic and semantic harmonization of the collected data. The *Task Executor* executes generated tasks. *Data Validator* validates the processed data and reschedules *Data Analyser and Flagger* on validation failure.
- **Semantic components:** *Term Extractor* extracts terms from the collected data. *Term Aligner* aligns the extracted terms to the terms of domain-specific ontologies and the subscriber's semantic model. *Ontology Generator* uses RDF conversion tools [24] to generate an ontology of the collected IoT data and annotate its terms with aligned terms. *Term Replacer* replaces the terms for the push IoT data. *Unit Compatibility Observer* sets a flag if the measurement unit of IoT differs from the subscriber's unit. *Unit Valuer Converter and Replacer* replaces the collected data values according to the conversion unit formula if a unit conversion flag exists.
- **Syntactic components:** *Domain-specific Standard Adaptor* translates the collected data into a domain-

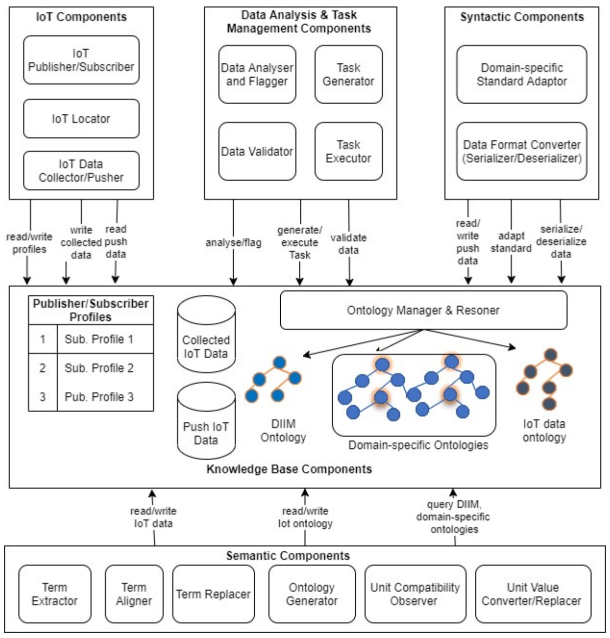
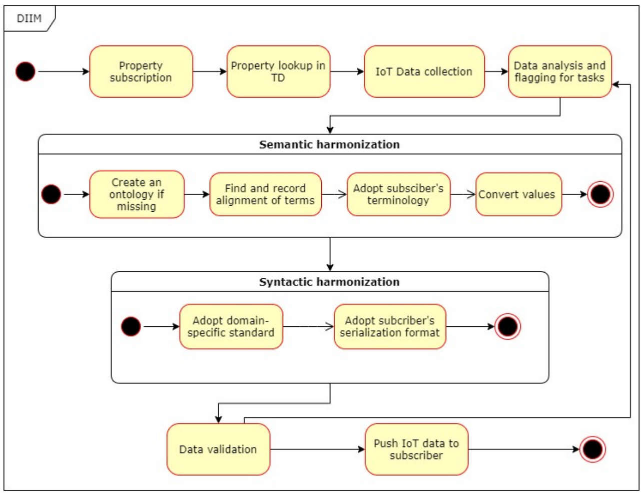


Fig. 2. Key components of DIIM

specific standard. *Data Format Converter* serialize/de-serialize the collected data from OWL into another platform/application-specific data format.

- **KB:** It incorporates the knowledge of DIIM methodology in DIIM ontology written in OWL. The core part of the ontology contains domain-specific knowledge of a SWN-application, such as domain-specific ontologies, standards, serialization formats, measurement properties and their units. *IoT Data Ontology* holds the collected IoT data in OWL. *Ontology Manager* manages all ontologies in KB. *Publisher and Subscriber profiles* are also described and populated in OWL. *Term Aligner and Tagger* utilizes existing alignment tools, e.g. ALIN, MapOnto, and Yam++, to find alignment between IoT data ontology and other ontologies in the KB and annotates its terms with the aligned terms. *Ontology Reasoner*, such as Hermit or Pellet, reasons about the facts in the KB and answers the queries. KB also contains a list of available OWL/RDF translators, serializers and deserializers that can transform data from one format to other.

Figure 3 illustrates DIIM's key methodological steps that utilise a set of existing tools and technologies to offer syntactical and semantic interoperability of data and information to the IoT-enabled applications. The activity diagram highlights the procedure of the syntactic and semantic interoperability enablement between IoT/WoT and IoT-enabled applications. DIIM's procedural steps are as follows.



DIIM
Property subscription
Property lookup in TD
IoT Data collection
Data analysis and flagging for tasks
Semantic harmonization
Create an ontology if missing
Find and record alignment of terms
Adopt subsciber's terminology
Convert values
Syntactic harmonization
Adopt domain-specific standard
Adopt subcriber's serialization format
Data validation
Push IoT data to subscriber

Fig. 3. DIIM's activity states

1) **Property subscription:** An IoT-enabled application subscribes its interest to receive data of a particular property, e.g. temperature. DIIM subscription interface specifies parameters as follows:

> **mandatory**{*subscriberEndpoint, communicationProtocol, serializationFormat, propertyName, measurementUnit*}
> **optional**{*latitude, longitude, fromDate, toDate, standardName, ontologyURI, applicationDomain*}

2) **Property lookup in TD:** DIIM keeps on scanning the TD unless a match to the subscription is found.
3) **IoT data collection:** Once DIIM finds a subscription's match, it collects IoT's available data and metadata and stores it as IoT collected data.
4) **Data analysis and flagging for tasks:** DIIM analyzes Data and flags it according to the subscription as the next step. For each flag, a task is created, scheduled and executed by DIIM.
5) **Semantic harmonization:** If IoT data do not refer to an ontology, DIIM create an ontology for the terms of the collected data. DIIM aligns these terms to the domain-specific and subscriber ontologies. All found alignments are stored as annotations in the IoT ontology. If the measurement unit of IoT does not match to subscription, the property values are converted and stored in the IoT ontology.
6) **Syntactic harmonization:** DIIM queries the data in the IoT ontology and transforms it in the domain-specific standard and serialization format that the subscriber requires. Finally, it stores harmonized data in the push IoT data.
7) **Data validation and push:** Newly harmonized data is validated according to the subscription profile. If validation fails, DIIM switches to Step *Data analysis and flagging for tasks*; otherwise, data is pushed to the subscriber.

B. A case study on IoT-enabled water quality monitoring

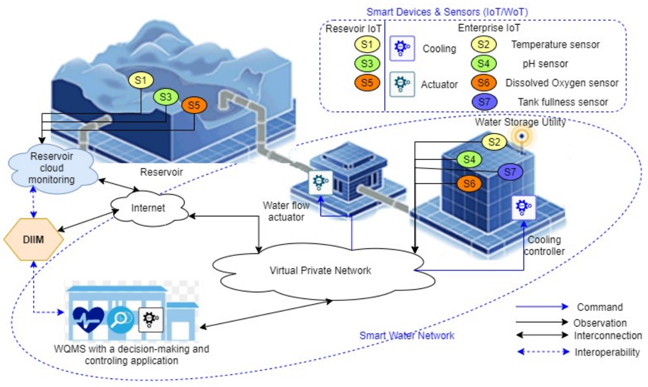


Fig. 4. A case study for interoperability of IoT-enabled WQMSs

Figure 4 illustrates a potential interoperability scenario for smart sensing in the water domain. Here, a network of IoT/WoT (smart sensors and devices) is constructed through the interconnection of several smart devices to support the pervasive and ubiquitous functionality of a WQMS. On one side, we see cloud-based water quality monitoring of the reservoir through S_1, S_2, and S_3 sensors for temperature, pH, and dissolved oxygen, respectively. On the other side, we see an enterprise that uses lake water as a drinking water resource to produce drinking water bottles. The enterprise has built up a SWN by interconnecting the enterprise IoT/WoT and WQMS in a Virtual Private Network (VPN) to monitor and manage the quality of the stored water. The smart sensors (S_2, S_4, S_6, and S_7) observe temperature, pH, dissolved oxygen and tank fullness and send the data to the enterprise WQMS for centralized remote monitoring and management. The decision-making application of WQMS makes decisions based on enterprise logic. It issues commands for the smart devices (cooling controller and actuator) to keep the stored lake water under ideal conditions, e.g., water temperature in a range of 2-15 °C, pH value in 6-9 logarithmic units, dissolved oxygen concentration in 80-120 mg/L and tank fullness in 700000 and 900000 m^3. Table I lists the rules computed by the decision-making system when sensors send the observation data to the WQMS. Consequently, the controller dispatches operational commands to operate the cooling and actuator utilities.

TABLE I
ENTERPRISE LOGIC RULES TO MANAGE THE STORED WATER

Property	Rule	Utility	Command
Temperature	if S_2value > 15	water cooling	on
Temperature	if S_2value <= 2	water cooling	off
Water volume	if S_7value <= 700000	actuator	open
Water volume	if S_7value >= 900000	actuator	close

Considering the case when the temperature of the fetched water exceeds 15 °C, the cooling system at the water storage utility must cool down the water. This operation will result in electricity consumption and raise the production cost. Alternatively, as shown in Table II, the stored water can also be cooled by opening the actuator if the reservoir water temperature is less than 15 °C and there is still storage capacity in the water tank. However, the decision-making application of WQMS must understand and evaluate the reservoir sensor data to make real-time decisions. Therefore, the syntactic and semantic interoperability [25] of the data collected by the IoT must be enabled regardless of the data serialization format and lexical label name of the observed data.

TABLE II
EXTENDED LOGIC RULE TO MANAGE WATER TEMPERATURE

Property	Rule	Utility	Command

Temperature	if S_2value > 15 AND
	if S_1value < 15 AND
	if S_7value < 80000
		actuator	open

Table III displays the seven modelled sensors as an indicative interoperability scenario of the enterprise WQMS with reservoir cloud monitoring. S_1, S_3 and S_5 sensors are from a Chinese manufacturer, and they use JSON data format to serialize and use the terms *temp*, *ph* and *DO* to label their observed data. S_2 sensor is from an American manufacturer, and it uses RDF data format to serialize and uses term *Temperature* to label its observation. S_4, S_6, and S_7 sensors are from a European manufacturer, and they use XML data format to serialize and use the terms *pH* and *DissolvedOxygen* to label the observed data. The decision-making system of the enterprise WQMS follows the Semantic Web approach. Therefore, it expects the data to be well defined in RDF/XML format and uses terms that the Australian Government Linked Data Working Group defines for the marine water quality observations in a water quality ontology [26]. Additionally, the Enterprise Water Quality Monitoring (EWQM) ontology (re)uses the concepts (terms) from the Simple Knowledge Organization System (SKOS) data model and units of measurement from Quantities, Units, Dimensions, Data Types (QUDT) ontology. In contrast to the enterprise application, none of the IoT for reservoir refers to any ontology or data model standard. Although their data is publicly accessible from the cloud in JSON and CSV formats. This situation poses challenges of syntactic and semantic heterogeneity that the enterprise must address if it wants to use the data of reservoir sensors in its WQMS.

TABLE III
REPRESENTATION OF THE MODELLED SENSORS

Sensor	Sensor observation property	Serialization format	Label name of observed property	Unit of observed property
(IoT)	***(Data context)***	***(Data syntax)***	***(Data semantic)***	
S_1	Temperature	JSON	temp	°F
S_2	Temperature	RDF/XML	Temperature	°C
S_3	pH	JSON	ph	
S_4	pH	RDF/XML	pH	logarithmic units
S_5	Dissolved Oxygen	JSON	DO	ppm
S_6	Dissolved Oxygen	RDF/XML	DissolvedOxygen	mg/L
S_7	Water volume	RDF/XML	WaterVolume	m^3

C. DIIM's application in a case study

We describe the DIIM application procedure while enabling interoperability in the previously outlined case study of a water bottling enterprise and a water reservoir. Table IV displays an indicative setup of DIIM for the given case study.

TABLE IV
DIIM'S INDICATIVE PARAMETER SETUP

Parameter	Input
application domain	water quality monitoring
IoT data subscriber	enterprise WQMS
IoT data publisher	reservoir cloud monitoring application
domain-specific ontologies	SAREF, Geo, Time, QUDT, GeoRSS
application-specific ontologies	DIIM, IoT data ontology, EWQM
subscriber/publisher profiles	WQMS-profile, reservoir cloud-profile
domain-specific standard	WaterML2
IoT collect data	data collected from publisher
Parameter	**Output**
IoT push data	syntactically and semantically harmonized IoT data for subscriber
IoT ontology	IoT ontology with collected data and annotations as references to the terms of other ontologies

Based on the setup, DIIM will execute the following operational activities:

1) **Property subscription**: DIIM's operational activity starts when enterprise WQMS subscribes for the water

quality property (Temperature in °C) at a particular location and in the RDF/XML format at a specified endpoint. DIIM creates a subscriber profile of the subscription requested by the WQMS.

2) **TD lookup**: DIIM uses its DIIM ontology for water quality description to semantically match the TD for the subscribed water quality property in the IoT cloud. The lookup services keep on searching unless an IoT is found. Then, DIIM creates a publisher profile of the matched IoT and links it to the subscriber profile that matches the semantic search.
3) **Data collection**: The data collector starts collecting the meta and measurement data based on the publisher profile. An exemplary input to DIIM as collected IoT data in JSON format is revealed below. The fetched data is stored in collected IoT database.

```
### DIIM Input: a snippet of IoT data in JSON format ###
Meta data: "Name":"S1","Description":"The sensor measures water
temperature in Fahrenheit", "serial":"00-14-22-01-23-45",
"model":"BFG9000", "mac":"50:8c:b1:77:e8:e6",
"latitude":51.75543,"longitude":-1.03248
Measurement data: [{"4baa-a2ff-8741efad4e63": {"temp":[
{"timestamp":"2021-08-09T17:01:28.796Z","values":{"value":20}},
{"timestamp":"2021-08-09T17:01:38.792Z","values":{"value":24}},
... ]}}]
```

4) **Data analysis**: The collected data is analysed, and flags are set in the next step. Since the profiles of WQMS (subscriber) and reservoir (publisher) do not match, DIIM sets flags for serialization message content format RDF/XML, property name (term) harmonization and property value conversion.
5) **Semantic harmonization**: Since a flag for the property name and value conversion is set, the data objects of the collected data are relabeled, and its value is converted according to the subscriber profile. As shown in Table V, DIIM will map the terms of reservoir and WQMS to DIIM, WQMS ontology, and domain-specific ontologies. DIIM will create an IoT ontology for the reservoir and annotate its terms with the matched terms of other ontologies. DIIM will convert the temperature value from Fahrenheit to Celsius and store it in the IoT ontology.

TABLE V
ONTOLOGICAL ALIGNMENT OF TERMS AMONG RESERVOIR IOT, DIIM KB, AND ENTERPRISE WQMS

	Terms of		
Reservoir	**DIIM Ontologies**	**WQMS**	**WQMS Ontology**
S_1	saref:Temperature sensor	S_2	ewqm:Sensor
temp	saref:Temperature	Temperature	qudt:water_temperature
f	saref:Temperature unit	C	qudt:unit
water	saref:Water	Water	ewqm:object
latitude	geo:latitude	Location	georss:point
longitude	geo:longitude	Location	georss:point
timestamp	time:dateTimeStamp	dateTime	ewqm:dateTime
values,value	saref:value	value	ewqm:value
Name	rdfs:Literal	name	ewqm:name
Description	rdfs:Comment	description	ewqm:description

The DIIM generated IoT ontology populated with collected data and annotated with references to the terms of other domain-specific ontologies is as follows:

```
<!—- DIIM Output: IoT ontology with examples of data and annotations —->
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:XMLS="http://www.w3.org/2001/XMLSchema#"
xmlns:iotonto="http://www.iot4win.co.uk/diim/iotonto/"
xmlns:saref="http://uri.etsi.org/m2m/saref#"
xmlns:ewqm="http://waterenterprise.com/wqm/ewqm#"
xmlns:georss="http://www.georss.org/georss/"
xmlns:qudt="http://qudt.org/2.1/schema/qudt">
<owl:Class rdf:about="iotonto:#Observation"/>
<owl:Class rdf:about="iotonto:#Sensor"/>
<owl:ObjectProperty rdf:about="iotonto:#hasObservation">
<rdfs:subPropertyOf rdf:resource="owl:#topObjectProperty"/>
<rdfs:domain rdf:resource="iotonto:#Sensor"/>
<rdfs:range rdf:resource="iotonto:#Observation"/>
</owl:ObjectProperty>
<iotonto:Sensor rdfs:label="saref:Temperature sensor; ewqm:Sensor">
<iotonto:name rdf:datatype="XMLS:string">S1</iotonto:name>
<iotonto:description rdf:datatype="XMLSchema:string">The sensor measures
water temperature in Fahrenheit</iotonto:description>
<iotonto:latitude rdf:datatype="XMLS:float">51.75543</iotonto:latitude>
<iotonto:longitude rdf:datatype="XMLS:float">-1.03248</iotonto:longitude>
<iotonto:mac rdf:datatype="XMLS:string">50:8c:b1:77:e8:e6</iotonto:mac>
<iotonto:serial rdf:datatype="XMLS:string">00-14-22-01-23-45</iotonto:serial>
<iotonto:model rdf:datatype="XMLS:string">BFG9000</iotonto:model>
<iotonto:measurementUnit
rdf:datatype="XMLS:string">f</iotonto:measurementUnit>
<iotonto:observeredObject
rdf:datatype="XMLS:string">water</iotonto:observeredObject>
</iotonto:Sensor>
<iotonto:ObservationCollection>
<iotonto:id rdf:datatype="XMLS:string">4baa-a2ff-8741efad4e63</iotonto:id>
<iotonto:property rdf:datatype="XMLS:string">temp</iotonto:property>
<iotonto:hasObservation rdf:parseType="Collection">
<iotonto:Observation> <iotonto:timestamp
rdf:datatype="XMLS:string">2021-08-09T17:01:28.796Z</iotonto:timestamp>
<iotonto:values> <rdf:Seq> <rdf:li>20</rdf:li> </rdf:Seq> </iotonto:values>
</iotonto:Observation>
</iotonto:hasObservation> </iotonto:ObservationCollection>
<owl:Axiom>
<owl:annotatedSource rdf:resource="iotonto:#4baa-a2ff-8741efad4e63"/>
<owl:annotatedProperty rdf:resource="iotonto:#property"/>
<owl:annotatedTarget rdf:datatype="XMLS:string">temp</owl:annotatedTarget>
<rdfs:label rdf:datatype="XMLS:string">qudt:water_temperature</rdfs:label>
<rdfs:label rdf:datatype="XMLS:string">saref:Temperature</rdfs:label>
</owl:Axiom> </rdf:RDF>
```

6) **Syntactic harmonization**: The WaterML2 time-series standard is first adopted for the push IoT data. Then, an OWL translator serializes the data in RDF/XML format.
7) **Data push**: After validation, the Data distributor pushes the syntactically and semantically harmonized publisher data in the WaterML2 time-series standard and semantics of EWQM ontology to subscriber endpoint.

In summary, DIIM's steps to enable interoperability in the use-case are: (i) DIIM takes subscription parameters from enterprise WQMS and IoT data from reservoir cloud as inputs. (ii) DIIM analyses the collected data and transforms the collected data in an OWL/RDF expressed semantic model. (iii) Ontology alignment tools align the newly generated semantic model with the domain-specific ontologies and the ontology of enterprise WQMS. DIIM records all found matches in the semantic model as annotations. (iv) Finally, OWL/RDF translator adopts the WaterML2 standard for time-series data and transforms the semantic model in enterprise WQMS's required format. Then DIIM uses the terms from enterprise WQMS's ontology to relabel the data. Since DIIM has also aligned the semantic model of the reservoir IoT data to domain-specific ontologies, therefore, the reservoir IoT data also becomes interoperable to all those applications which support these domain-specific ontologies.

V. Conclusion

This work presents a novel method to address the syntactic and semantic interoperability challenges of IoT-enabled SWNs. *DIIM's syntactic interoperability approach* overcomes the serialization format issues during the parsing of the IoT data by data-consuming applications by applying MDA methods for data format translation. Additionally, DIIM adopts domain-specific standards, e.g. WaterML2, to represent water-related data in time-series before delivering it to the consumer application. *DIIM's semantic interoperability approach* harmonizes the semantic models of IoT and a data-consuming application by aligning their ontologies to each other. Suppose an IoT application neither uses an existing ontology nor builds one ontology for its data. In that case, DIIM creates a semantic model in OWL from the available IoT data while finding the alignment of its terms to the terms of the domain-specific ontologies and (re)-annotate the IoT semantic model. With this method, DIIM enables interoperability between IoT and a SWN application and enhances the interoperability to the next level by adopting domain-specific ontologies and standards. Because, after alignment of IoT's semantic model to domain-specific ontologies, the IoT data becomes interoperable for all those applications that use these domain-specific ontologies. In the motivation scenario, DIIM acts as a mediator in the water domain while enabling the data-based interoperability between an IoT platform and a SWN application. However, any other discipline can use this approach to enable interoperability, where utilization of IoT data is beneficial.

Acknowledgment

European Union's Horizon 2020 supports this research and innovation programme under the Marie Skłodowska-Curie Innovative Training Networks (ITN) IoT4Win - Internet of Things for Smart Water Innovative Network (765921).

References

[1] Sensus, "Water 20/20: Bringing smart water networks into focus," 2019. [Online]. Available: https://smartwaternetworks.com/

[2] C. Turnitsa, "Extending the levels of conceptual interoperability model," in *Proceedings IEEE summer computer simulation conference, IEEE CS Press*, 2005.

[3] B. Brodaric, N. Booth, E. Boisvert, and J. Lucido, "Groundwater data network interoperability," *Journal of Hydroinformatics*, vol. 18, no. 2, pp. 210–225, 10 2015. [Online]. Available: https://doi.org/10.2166/hydro.2015.242

[4] I. Standardization, "Iso/iec 7498-1: 1994 information technology–open systems interconnection–basic reference model: The basic model," *International Standard ISOIEC*, vol. 74981, p. 59, 1996.

[5] E. Wassermann and A. Fay, "Interoperability rules for heterogenous multi-agent systems: Levels of conceptual interoperability model applied for multi-agent systems," in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, 2017, pp. 89–95.

[6] S. Howell, Y. Rezgui, and T. Beach, "Automation in Construction Integrating building and urban semantics to empower smart water solutions," *Automation in Construction*, vol. 81, pp. 434–448, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.autcon.2017.02.004

[7] IEEE, "Ieee guide for smart grid interoperability of energy technology and information technology operation with the electric power system (eps), end-use applications, and loads," *IEEE Std 2030-2011*, pp. 1–126, 2011.

[8] W3C, "Semantic web," 2015. [Online]. Available: http://www.w3.org/standards/semanticweb/

[9] A. Groß, C. Pruski, and E. Rahm, "Evolution of biomedical ontologies and mappings: overview of recent approaches," *Computational and structural biotechnology journal*, vol. 14, pp. 333–340, 2016.

[10] N. Kalatzis, G. Routis, Y. Marinellis, M. Avgeris, I. Roussaki, S. Papavassiliou, and M. Anagnostou, "Semantic interoperability for IoT platforms in support of decision making: An experiment on early wildfire detection," *Sensors (Switzerland)*, vol. 19, no. 3, 2 2019.

[11] G. Hatzivasilis, I. Askoxylakis, G. Alexandris, D. Anicic, A. Bröring, V. Kulkarni, K. Fysarakis, and G. Spanoudakis, "The interoperability of things: Interoperable solutions as an enabler for iot and web 3.0," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2018, pp. 1–7.

[12] A. Bröring, S. Schmid, C. Schindhelm, A. Khelil, S. Käbisch, D. Kramer, D. Le Phuoc, J. Mitic, D. Anicic, and E. Teniente, "Enabling iot ecosystems through platform interoperability," *IEEE Software*, vol. 34, no. 1, pp. 54–61, 2017.

[13] G. Anzaldi Varas, W. Wu, A. Abecker, E. RUBIÓN, A. Corchero, A. Hussain, and M. QUENZER, "Integration of water supply distribution systems by using interoperable standards to make effective decisions," in *11th International Conference on Hydroinformatics, HIC 2014*, 08 2014.

[14] A. S. Rao and M. P. Georggeff, "Decision Procedures for BDI Logics," *Journal of Logic and Computation*, vol. 8, no. 3, pp. 293–343, 06 1998. [Online]. Available: https://doi.org/10.1093/logcom/8.3.293

[15] M. Winikoff, L. Padgham, J. Harland, and J. Thangarajah, "Declarative and procedural goals in intelligent agent systems," *International Conference on Principles of Knowledge Representation and Reasoning 22-25 April 2005*, 2002.

[16] S. Howell, Y. Rezgui, and T. Beach, "Water utility decision support through the semantic web of things," *Environmental Modelling & Software*, vol. 102, pp. 94 – 114, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364815216307551

[17] G. Consortium, "Gene ontology," 01 2020. [Online]. Available: http://geneontology.org/

[18] A. Maedche and S. Staab, "Semi-automatic engineering of ontologies from text," in *Proceedings of the 12th international conference on software engineering and knowledge engineering*. Citeseer, 2000, pp. 231–239.

[19] A. Gómez-Pérez, M. Fernandez-Lopez, and O. Corcho, *Ontological Engineering*. Springer-Verlag London, 2004.

[20] S. Izza, "Integration of industrial information systems: from syntactic to semantic integration approaches," *Enterprise Information Systems*, vol. 3, no. 1, pp. 1–57, 2009. [Online]. Available: https://doi.org/10.1080/17517570802521163

[21] S. K. Howell, Y. Rezgui, T. Beach, W. Zhao, J. Terlet, and H. Li, "Smart water system interoperability: Integrating data and analytics for demand optimized management through semantics," *ICCCBE*, pp. 1–9, 2016.

[22] K. Schwab, *The fourth industrial revolution*. Currency, 2017.

[23] H. D. W. g. OGC, "Waterml2," 2014, accessed 23 Jan 2021. [Online]. Available: http://waterml2.org/

[24] W3C, "Convertertordf," accessed 20 Oct 2021. [Online]. Available: https://www.w3.org/wiki/ConverterToRdf

[25] M. Noura, M. Atiquzzaman, and M. Gaedke, "Interoperability in internet of things: Taxonomies and open challenges," *Mobile Networks and Applications*, vol. 24, no. 3, pp. 796–809, 2019.

[26] A. G. L. D. W. Group, "Water quality data archive," 2016, accessed 6 July 2021. [Online]. Available: http://environment.data.gov.au/