Environmental Modelling & Software 102 (2018) 94–114

Contents lists available at ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

# Water utility decision support through the semantic web of things

Shaun Howell*, Yacine Rezgui, Thomas Beach

*BRE Institute of Sustainable Engineering, Cardiff University, 52 The Parade, Cardiff, CF24 3AA, UK*

ARTICLE INFO

*Article history:*
Received 11 October 2016
Received in revised form
21 December 2017
Accepted 7 January 2018

*Keywords:*
Water management
Decision support tool
Interoperability
Big data
Ontology
Semantic web
Internet of things
Smart water networks

ABSTRACT

Urban environments are urgently required to become smarter. However, building advanced applications on the Internet of Things requires seamless interoperability. This paper proposes a water knowledge management platform which extends the Internet of Things towards a Semantic Web of Things, by leveraging the semantic web to address the heterogeneity of web resources. Proof of concept is demonstrated through a decision support tool which leverages both the data-driven and knowledge-based programming interfaces of the platform.

The solution is grounded in a comprehensive ontology and rule base developed with industry experts. This is instantiated from GIS, sensor, and EPANET data for a Welsh pilot. The web service provides discoverability, context, and meaning for the sensor readings stored in a scalable database. An interface displays sensor data and fault inference notifications, leveraging the complementary nature of serving coherent lower and higher-order knowledge.



## 1. Introduction

In order to tackle sustainability and economic challenges through ICT, urban environments, including the water sector, are undergoing a transformation towards smart systems through the use of web-enabled sensors, analytics software, and decision support tools. Smart water networks have been noted to promote efficacy, efficiency, and resilience of water infrastructure (CTRL+SWAN, n.d.; Mutchek and Williams, 2014; Thompson and Kadiyala, 2014). However, as with fields such as smart grids and smart cities, the application of ICT in the water value chain is restricted due to an inability to share data and knowledge, and hence interoperate, across the people and software components involved (EIP Water, n.d.). This has limited the impact of advanced applications such as optimisation engines, artificial intelligence, and semantic inference. Network operators need modern decision support tools which empower them to make optimal decisions based on extensive data sources and relevant insights, and this interoperability challenge is increasingly pertinent (Curry et al., 2014).

In smart grids, this has been stated by IEEE to occur due to three main issues: lack of machine communication protocols, lack of common data formats and lack of common meaning of exchanged content (IEEE Standards Committee et al., 2011). In the 'smart water' domain, the same core issues have restricted the utility and hence prevalence of ICT penetration. Notably, a recent report from the ICT4Water cluster of EC FP7 projects highlighted the need for standardised models to address the issue of interoperability in the smart water domain (Vamvakeridou-Lyroudia et al., 2015) and specifically indicated the importance of ontologies as a means to maintain semantic clarity and integrate knowledge. This leads to a clear precedent in the smart water domain to develop common communication protocols, data models and semantic vocabularies.

The Internet of Things is addressing the need for interoperable communication protocols, and much progress has been made in the past 5 years, towards enabling device discoverability and message exchange. However, beyond the requirement for applications to receive data, they must be able to consume and utilise it correctly with confidence, which requires a thorough understanding of its context, meaning, and provenance. This requires a robust approach towards semantic interoperability, and achieving this goes beyond the presently observed Internet of Things, through a convergence with the semantic web. This has been termed the Semantic Web of Things (SWoT), with the key difference being a focus on application-layer interoperability, as opposed to protocol-layer interoperability (Calbimonte et al., 2014). SWoT therefore

* Corresponding author.
*E-mail addresses:* shaunkhowell@gmail.com (S. Howell), RezguiY@cardiff.ac.uk (Y. Rezgui), BeachTH@cardiff.ac.uk (T. Beach).

https://doi.org/10.1016/j.envsoft.2018.01.006
1364-8152/

promises to support advanced applications such as artificial intelligence, where data semantics must be explicitly machine interpretable in order for automated machine-to-machine communication to be sufficient for reliable data utilisation. These explicit semantic statements are typically collated into semantic models, which create a shared understanding of the domain and a shared method of representing data and their meaning. Within this remit, ontologies are the most expressive option, but are also the most complex, in terms of human comprehension and computational complexity. Critically, ontology based models also allow the use of inference to produce new knowledge about a system beyond what has been explicitly stated.

This paper proposes that fostering a Semantic Web of Things will unlock vast potential in the smart water domain, by robustly addressing the interoperability challenge which has been noted as responsible for 40% of the value of IoT systems (Manyika et al., 2015). The paper therefore presents a knowledge-based data and inference platform, and an example decision support tool, which demonstrate the value of the approach. The knowledge management platform couples a scalable time-series database with a semantic knowledge base, empowered by SWRL rules and an accompanying inference engine. This is grounded in a comprehensive domain ontology which describes not only the sensor data and metadata, but also the systemic context of the data, based on a socio-technical system model of the water value chain. This model federates and extends typical GIS schemas, product descriptions, and expert knowledge as an OWL ontology. The ontology is then instantiated through a semi-automatic process, supporting the integration of legacy systems. A key novelty of the approach is the semantic rule base which integrates expert domain knowledge and use case specific heuristic rules with the knowledge-based solution. This is accomplished with real world data from both clean and wastewater networks for a pilot site in Wales, within the context of an EC FP7 research project, entitled 'Water analytics and Intelligent Sensing for Demand Optimised Management' (WISDOM).

The value of the hybrid semantic approach is demonstrated through a decision support tool which integrates a semantic rule engine with a comprehensive set of use-case based rules, the platform, and a graphical user interface. The use case of 'alert impact prediction' was chosen, whereby a problem such as a blockage occurs in the network, and the decision maker needs to evaluate the impact of the incident on the network and its end-users, within the utility's business context. The application therefore uses semantic inference to predict the affected network components, and allows the user to interrogate the issue by viewing dynamic and static data about the network and problem. This requires an integrated presentation of predicted, static, and dynamic data, in a manner suitable to the expert decision making process. By highlighting the value of leveraging both the knowledge-based, and data-driven programming interfaces of the platform, the decision support tool serves as evidence towards the value of the Semantic Web of Things.

The paper therefore addresses the following primary research question: does a decision support system grounded in IoT and semantic web provide added value over rule based systems currently used in the sector and those observed in literature?

The overarching methodology adopted is that of action research, whereby the researcher works with the stakeholders of a target system to both solve an immediate challenge and generate knowledge from the process and outputs. This involves an iterative learning approach of defining the problem, specifying a proposed solution, building and testing a proposed solution, then reflecting and learning from the iteration. Such an approach allows agility through frequent adaptation, and promotes high-quality outputs through transparency and regular expert review. As such, this paper couples the specification, development, and testing of each of the various components investigated towards answering the stated question.

The rest of the paper proceeds by discussing background concepts and related work in section 2, then section 3 provides an overview of the proposed solution, and section 4 presents the knowledge management platform in detail. The benefit of the approach is demonstrated through an example semantic inference use case built on the platform in section 5, and section 6 presents a decision support tool which leverages the coherent lower and higher-order knowledge served by the platform's programming interface. Section 7 then discusses the work conducted and offers a perspective on the future of knowledge management in smart water.

## 2. Background

### 2.1. Smart water and utility decision support tools

The application of ICT and cybernetics principles to the water sector has grown significantly in recent years through the notion of smart water networks (Cahn, 2013; Mutchek and Williams, 2014; Sensus, 2012). These aim to use intelligent sensing (Allen et al., 2013), optimisation (Zhao et al., 2016), and decision support (Schenk, 2010) to operate clean and waste water networks and assets in a more efficient, sustainable and reliable manner. A cluster of European Commission Seventh Framework Programme (EC FP7) research projects, ICT4Water, has been formed to investigate various aspects of this proposition (Vamvakeridou-Lyroudia et al., 2015), and the European Innovation Platform for water (EIP-water) has launched an action group for water monitoring for decision support (EIP Water, n.d.). The smart water networks forum (SWAN) is serving as a nucleus for this trend, and has proposed a framework for smart water networks (Cahn, 2013) consisting of several layers: physical, sensing and control, collection and communication, data management and display, and data fusion and analysis.

One key impact scenario identified by SWAN is intelligent pressure management to reduce leakage and energy consumption whilst improving network resilience. Recent work has demonstrated a 12.5% leakage reduction through intelligent pressure reduction based on an EPANET model (Babel et al., 2009), and this was conferred by another work (Creaco et al., 2016) which minimised energy consumption whilst optimising network pressures. Another work utilised a cloud-based machine learning approach to leakage management (Mounce et al., 2015), and a platform has been developed which aims to integrate ICT with water networks to promote reliable and resilient resource management, whilst reducing energy consumption (Lee et al., 2015), based on a cyber-physical approach. However, implementing these smart water solutions in practice requires pervasive interoperability, as highlighted by the recent SWAN report on communication in smart water (Hauser et al., 2016).

### 2.2. Supporting smart systems through the internet of things

The role of IoT in smart water has been increasingly noted (Wong and Kerkez, 2016; Robles et al., 2015), resulting in various reference architectures and platforms. It is noteworthy that (Robles et al., 2015) highlights the role of semantics and knowledge-oriented interoperability, although the authors only offer a model of a system's ICT components, as opposed to the underlying socio-technical system, although the paper does stress the value of adding semantics to the work presented.

In the broader smart city field, many examples exist of IoT platforms aiming to coordinate data management (ALMANAC,

2015a,b; Kolozali et al., 2014; Lea and Blackstock, 2014; RERUM, n.d.). The CityPulse project (CityPulse, 2014) for example, emphasises scalable IoT stream processing, and includes semantic tagging of streams, but this is based only on a simple ontology describing the domain of data and event streams, rather than contextualising the data through a model of the target socio-technical system. The ALMANAC project (ALMANAC, 2015a,b) proposed a service oriented architecture for the collection and analysis of near real time information, and again boasted semantic interoperability. The ALMANAC platform went beyond the semantic modelling conducted in CityPulse to include domain concepts, such as their ontology for water applications (ALMANAC, 2015a,b), but this only described 6 types of object, so again lacks true domain contextualisation. The RERUM project proposed an IoT framework, but emphasised its security and privacy aspects (RERUM, n.d.); this again utilised a semantic model, but it only described the cyber-physical nature of IoT systems, and not the underlying socio-technical system, similar to CityPulse. Finally, the recent Hypercat standard specifies a lightweight file format and API for discovering and accessing IoT resources (BSI, 2016).

### 2.3. Beyond the internet of things to a semantic web of things

The common emphasis on semantics in the most recent IoT platforms is noteworthy, but fails to deliver the higher-order interoperability potential which the semantic web offers, due to the lack of semantic models for the water domain. The nature of IoT and semantic web convergence has recently been driven by the W3C Web of Things Working Group, which is developing a suite of standards to overcome silos in the industry, including a 'Thing Description' model: a vocabulary for describing Things, with a default serialization of JSON-LD (W3C, 2017). This demonstrates a clear appetite in the industry for development in this direction of convergence. One long-term ambition of such work is to closely integrate time-series data (which a great deal of IoT data is), with RDF data in a homogeneous manner. However, this represents a significant ongoing research challenge in its own right, whereas significant value can be derived in the short-term from using these technologies in a loosely-coupled manner, which allows mature tools to be leveraged alongside each other without fundamental changes, which is a beneficial compromise.

The value of the convergence of the semantic web and IoT has been noted by many authors (Gyrard et al., 2014; Jara et al., 2014; Pfisterer et al., 2011; Sahlmann and Schwotzer, 2015), although they express different perspectives on this new field. These works share an emphasis on powerful interoperability through ontologies and open models. Jara et al. (2014) present their survey and vision towards a SWoT, highlighting interoperability at a greater level of abstraction as an evolution of IoT, through high-level modelling of real world entities. Gyrad et al (Gyrard et al., 2014) also present their vision of SWoT, but instead emphasise it as an evolution, through semantic interoperability across domains, and again through greater domain knowledge modelling. The earlier work of Pfisterer et al. (2011) also proposed SWoT with an emphasis on modelling real world entities, but framed the work as an evolution of semantic sensor networks. Finally (Sahlmann and Schwotzer, 2015), adopts a novel stance on SWoT, whereby web-enabled things exchange micro-ontologies, as an extension of MQTT and CoAP. This proposes an IoT revolution through fundamental change of its enabling technologies, rather than integrating a higher-order knowledge layer above existing approaches.

One example which acknowledged the role of higher-order knowledge management in the water domain is (Stewart et al., 2010), where a knowledge-based system was developed for the web which enabled consumption knowledge to be elicited from smart metering data. Also highly significant, the WatERP project proposed an agent-oriented ICT platform to enable supply and demand matching in water networks, and used a domain ontology alongside a data warehouse to manage the solution's data (WatERP, 2013), although the ontology is still relatively simple compared to those utilised in other domains such as energy and building information modelling.

As well as promoting semantic interoperability, inference represents a key advantage of ontological modelling; that new knowledge can be created from explicit knowledge. That is to say that beyond what is manually or automatically instantiated of a domain model, it is possible to infer new knowledge based on the existing statements made in the domain model. This relies on the 'open world assumption' which ontological modelling utilises. This assumes that anything which is not stated may be true, as opposed to conventional object-oriented modelling which assumes that anything which is not stated is not true. The role and mechanisms of semantic inference have been well explained recently (Shu et al., 2016) in the context of data validation. Based on such a semantic inference capability, an inference engine can utilise various methods to infer further truths from the explicitly stated truths.

### 2.4. Semantics in smart cities and smart water

As a recent challenge, ontological models in the smart water field are sparse, although some relevant examples are observed. Firstly, in the broader smart city field, the ISO/IEC Joint Technical Committee's report on smart cities (ISO/IEC JTC1, 2014) highlighted the need for ontologies. IBM developed the SCRIBE smart city ontology 5 years ago (Uceda-Sosa et al., 2011), commenting on a lack of available ontologies, and stable OWL tools. Their ontology paved the way for formal descriptions of city services, events, metadata, and abstractions. A smart city ontology termed 'Knowledge Model 4 City' was developed in (Bellini et al., 2014), with a focus on public transport and mobility, and also included a mapping to sensor concepts. However, this simple ontology seemed to only facilitate the query of public transport data by SPARQL. The SEMANCO project developed a large smart city ontology in OWL DL-Lite for the purpose of data integration (Nemirovski et al., 2013), resulting in 592 classes. The SEMANCO ontology appears to be intended for the exchange of static data in the planning phase of urban areas though, given the lack of sensor concepts and dynamic data provision. It could therefore contribute to an upper ontology which links ontologies in each sector, as operational data and semantics are closely coupled to their target system and industry. The CityGML standard (Gröger and Plümer, 2012) formalizes concepts and relationships relevant to geospatial knowledge in cities, and some semantics as to the nature of objects and spaces in cities, but in an insufficient manner for interoperability of operational smart city data across verticals. Finally, BSI:PAS 182 (BSI, 2014) proposes a high level smart city ontology, which serves as an important step, albeit as a 'lowest common denominator' approach, which hence captures little semantic depth.

In the smart water field, high-order semantic interoperability is only being recognised as a challenge very recently, in part caused by the growth of IoT and smart water networks. Attention is increasingly being paid to this challenge though, with an ongoing cluster of EC research projects, ICT4Water, highlighting the importance of semantic modelling (Vamvakeridou-Lyroudia et al., 2015). The most relevant existing water ontology is that developed in the WatERP project (WatERP, 2013), which models water balance concepts from the clean water network at a high level. Another very relevant model is the semantic water interoperability model

(SWIM) (Reynolds, 2014), which formalizes a description of water sector devices such as sensors, pumps, reservoirs and valves. As well as this, Waternomics, another ICT4Water project, has developed a linked data model (Curry et al., 2014), and a waste water treatment plant ontology was observed in the literature (Sottara et al., 2014). Also, several examples of water ontologies have been identified in the literature which were not specifically for the purpose described. Most notably are the works related to the CUAHSI Hydrologic Information System (CUAHSI, 2008; Huang et al., 2011; Horsburgh et al., 2009), SWEET (SWEET, n.d.) and HydrOntology (Vilches-Blazquez et al., 2009). These ontologies, and several others, are compared in Table 1. From Table 1, it is clear that whilst significant semantic modelling has been conducted, this is mainly aimed at the field of earth sciences rather than considering man-made water infrastructure artefacts, and is hence not sufficient for the challenge described. The INSPIRE water and wastewater network data specifications (INSPIRE Thematic Working Group Utility and Government Services, 2013) are very relevant to the smart water domain, but they only formalise simple concepts and relationships, hence only including 68 named entities in the UML formalisation available, with the other INSPIRE 'thematic areas' being less relevant from a utility network perspective. The WatERP ontology is very relevant and has significant overlap with the scope of the current work, although it is not comprehensive enough for the identified challenge.

It is evident that little effort to date has focused on the challenge identified; most efforts have been targeted at the earth science domain rather than the man-made water value chain, which only has a small number of limited ontologies. Semantic modelling in this specific domain has been raised as a critical issue (Vamvakeridou-Lyroudia et al., 2015), and is widely acknowledged as such in the neighbouring fields of smart grids and smart cities. There is a significant gap in the field of capturing in-depth knowledge regarding the technological, network, social, sensory and ICT artefacts involved in water management decisions in a water value chain. Also, there is a significant gap in leveraging the value of semantic inference, and of demonstrating the value of the Semantic Web of Things through powerful interoperable platforms and applications.

## 3. Overview and system architecture

This section presents the ICT platform developed, from the sensors and actuators to browser-based interfaces for utility experts and domestic end-users. The requirements engineering process and outputs at the system level are discussed first, then the overall architecture is discussed. Then, a description of the software implementation of the knowledge management components is offered, and finally a focus on the inference and rule engine integrated into the knowledge management service.

### 3.1. Requirements elicitation and the need for semantics

A comprehensive requirements engineering methodology was conducted for the overall software solution, which was then decomposed for the elicitation and refinement of individual component requirements. This followed a typical iterative analysis and design approach alongside industry experts, which is now described briefly before the outcomes pertinent to this paper are discussed.

The first stage of the process was to gather knowledge about the domain, target systems, and intended value proposition of the overall software solution. Following this, formal modelling was conducted of the business processes involved in the target system, and scenarios for the use of ICT within these were developed. Next, software requirements were produced for the overall software solution, following use case specifications and sequence diagrams. These requirements were then iterated alongside domain experts, and the previously developed scenarios, in order to ensure a comprehensive set of requirements was produced. This specification therefore bounded the scope of the software development task, by defining what it should be able to solve, and what is out of scope. This allowed a system architecture to be curated, and the requirements were then decomposed into separate requirements for each component. For the knowledge management service, this was then refined further alongside the development of knowledge modelling requirements and an observed need for domain expert engagement and buy-in regarding the produced ontologies.

The requirement engineering process produced outputs at each

**Table 1**
Summary of relevant semantic models.

| Acronym/name | Description | Owner | # Entities | Date |
|---|---|---|---|---|
| SWIM | Device level IoT semantic model for the water industry. | Aquamatix | 41 | 2016 |
| WISDOM | Cyber-physical and social ontology of the water value chain. | Cardiff University | 492 | 2016 |
| SAREF | 'Common denominator" of 23 smart appliance domain models. | ETSI | 154 | 2015 |
| OntoPlant | Extends the SSN ontology to decouple control logic from equipment choices in waste water treatment plants | Sottara et al. | 301 | 2014 |
| WaterML2 | Common format for hydrological time series data exchange. | OGC | 131 | 2014 |
| Utility Network Schemas | Water and sewer network model; part of a large European directive for geospatial data exchange. | EC-INSPIRE | 68 | 2013 |
| WatERP | Lightweight ontology of generic concepts for water sensing and management. | EURECAT | 25 classes | 2013 |
| WDTF | Format for transferring flood warning and forecasting data to the governing body. Precursor to WaterML2. | Australian Bureau of Meteorology | 337 | 2013 |
| CityGML UtilityADE | Domain extension for modelling utility networks in 3D city models, based on topology and component descriptions. | OGC | 317 | 2012 |
| SSN Ontology | Describes sensors and sensor networks, for use in web applications, independent of any application domain. | W3C | 80 | 2012 |
| SWEET | Middle-level ontology for environmental terminology. | NASA | 6000 | 2011 |
| Hydrologic Ontology for Discovery | Supports the discovery of time-series hydrologic data collected at a fixed point. Precursor to WaterML2. | CUAHSI | 4098 | 2010 |
| HydrOntology | Aims to integrate hydrographical data sources: town planning perspective, top down methodology. | Vilches-Blzquez et al. | 250 | 2009 |

stage, including business process diagrams, software use cases, sequence diagrams, scenario descriptions, meta-requirements, software requirements, competency questions, non-functional ontology and process requirements and a semi-automated extraction of domain vocabulary. Following this extensive process, several observations led to the decision to pursue a system which represented a convergence of IoT, semantic web, artificial intelligence, and rule-based systems. Critically, the need for an ontological grounding for the solution emerged from the following requirements, for which existing approaches were deemed unsuitable:

- The knowledge modelling component should aim to be application agnostic; it should have value outside of the initially intended system.
- The supported data and underlying domain perspectives should be easily extensible and adaptable.
- It should be possible to modify the underlying data structure easily without taking the service offline.
- It should be possible to query over multiple datasets simultaneously.
- It must be possible to integrate multiple ICT systems which exhibit semantic heterogeneity.
- It must be simple to integrate a rule-based approach with the knowledge base.

The classes of domain problems which are directly within the scope of the software are expressed through the scenarios which were a key outcome of the requirements engineering process. Whilst these have extensive descriptions beyond the scope of this paper, they can be broadly understood from the knowledge management modules perspective as follows:

- Integrating domestic smart meter and user engagement data with supply-side data to provide analytics to both sides
- Integrating the visualisation and API access for utility GIS data, telemetry data, asset data and metadata
- Supporting the discovery and use of sensor data for predictive analytics and optimisation, including pumping optimisation, CSO spill prediction, and leakage detection
- Providing knowledge-based analytics into the behavior of the system, using static data and timeseries values at predefined key times and levels of aggregation (e.g. previous single timestep, average of last 10 timesteps, max of last 50 timesteps).

As well as understanding which problems the software is designed to solve, it is relevant to state potential capabilities and problems which are out of scope:

- Direct SPARQL access to all timeseries data
- Generic stream-based RDF reasoning
- Complex hydraulic flow modelling
- Fully automated integration with legacy systems or data

Further, several problems were not directly considered, but the system could be useful in tackling them with little additional work:

- Complex pressure management scheme experiments will not be conducted
- Business strategy level analytics will not be considered, as the focus is on operational decision making
- Maintenance prediction and prioritisation will not be considered
- Remote actuation will not be directly experimented

In a broad sense, the knowledge management platform is capable of solving problems primarily related to the integration of knowledge sources, knowledge-based reasoning (similar to expert reasoning), and the discovery and contextualisation of data and Things. The system improves the access to, and utilisation of, timeseries sensor data by external analytics, but does not itself provide analytics features over full streams of high-velocity timeseries data nor use traditional hydraulic models.

### 3.2. System architecture

The ICT platform utilised a service-oriented architecture to coordinate data and knowledge management in a scalable manner above a communication interoperability layer, to expose a comprehensive range of information to the business services layer. This is shown in Fig. 1 below, which also illustrates the role of the governance service across the core platform services.

### 3.3. Knowledge management software

The knowledge management service included a semantic web service, a time series database, a resource catalogue API, and a Drools rule engine, as well as a RESTful API for querying and updating the knowledge base. The functional architecture of the knowledge management service is shown in Fig. 2 below.

The first noteworthy component of the knowledge management service was the triple store and SPARQL endpoint, which stored static knowledge about the socio-technical system, and contextualised references to the dynamic data stored in the time series database. This was an instantiation of the domain ontology described in section 4. The triple store was updated by subscribing to messages from the event bus, such that it always represented the latest state of the water value chain. The ontology was stored in a persistent triple store which used the Apache Jena libraries for query and update functions. The server exposed a simple RESTful API and a standard SPARQL endpoint, where the former allowed simpler interactions with the web service by client applications, to reduce future development barriers. In order to retrieve static data, applications query the triple store directly. In order to discover dynamic data, or to gain context for known data, applications could query the ontology via the SPARQL endpoint. For applications to retrieve dynamic data, they utilised the time series database portion of the API, which adopted a coherent structure and naming convention.

The time series database utilised the KairosDB libraries (KairosDB, n.d.), which are built on the Apache Cassandra database solution. This allows fault tolerant and scalable data storage and retrieval, and is hence better suited for handling high volumes of raw, well structured, data than a triple store.

Sensor data integration had three aspects: centralisation, pre-processing and storage, and integrated discovery and metadata description. Centralisation of the sensor data involved wireless sensors publishing observations to an Apache Kafka event bus, either directly or via a gateway at some sites. Data streams were subscribed to directly by some real-time applications, and also by the internal knowledge management module, where the observation timestamps and values were then sent to the kairosDB time series service, and also the ontology service. The Kairos service then undertook basic pre-processing before storing observations against an internal sensor ID, and the ontology service updated its 'latest observation value' accordingly. The APIs of these services could then be used across all of the sensors in an integrated way. Finally, the sensor metadata and semantic data was integrated within the ontology service, which was linked with the kairosDB time series data through the internal sensor ID, and cataloguing of the sensors

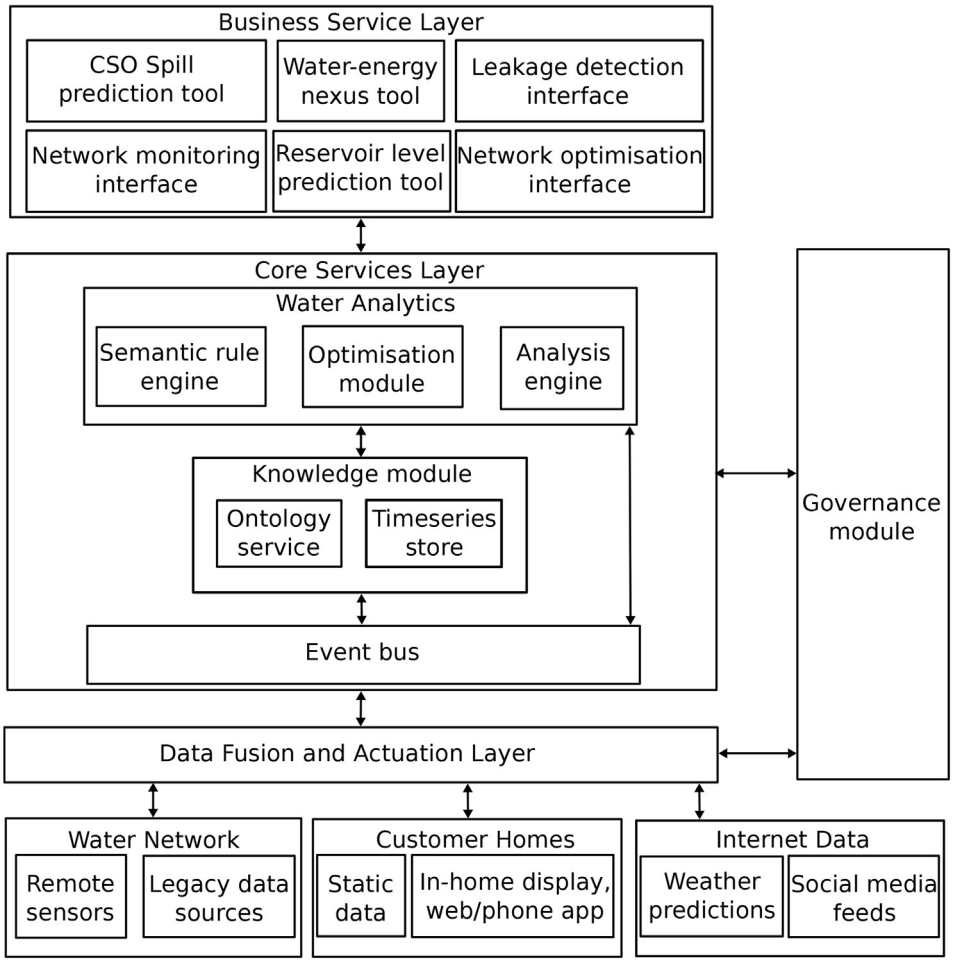


**Fig. 1.** Functional architecture of the proposed ICT solution.

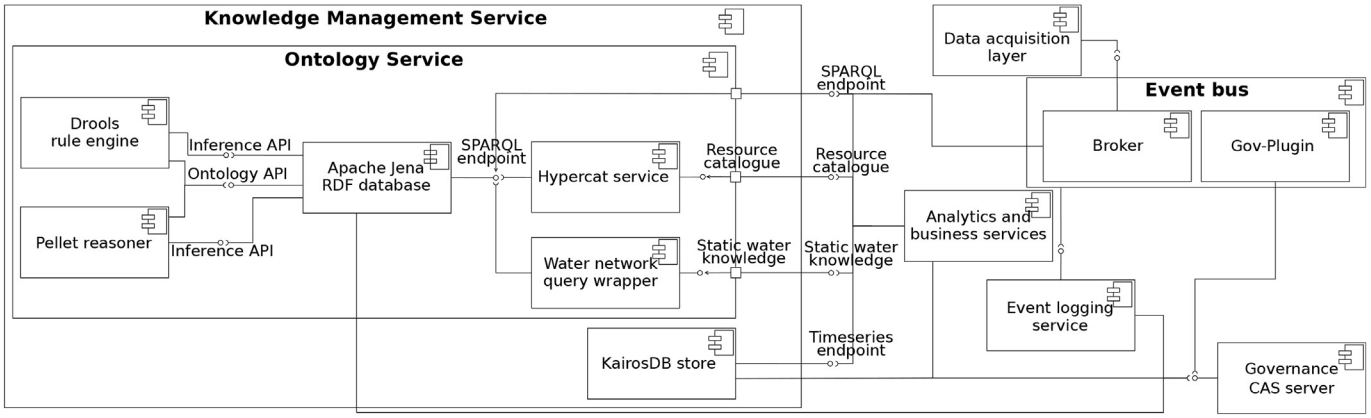


**Fig. 2.** Knowledge management UML component diagram.

was achieved through the Hypercat endpoint.

The inference solution integrated the Pellet reasoner for simpler rules and OWL-based inference, with a separate Drools engine for managing more complex SWRL rules. This approach was chosen as Drools was found to offer better performance and reliability during testing. Therefore, at each timestep, the Drools engine re-evaluated the triple store against the rules, and updated the triple store accordingly with any new inferred knowledge. The use of Drools within an ontology-driven application has been demonstrated previously (Sottara et al., 2012) within a waste water treatment plant, albeit not in an IoT platform setting.

From an architectural perspective, the knowledge inference capabilities of the developed architecture depend on a number of components, including the ontology and rule system. There are two main approaches for integrating both components through either a tight or loose coupling mechanism. While the latter (loose coupling) provides a two-way knowledge flow between the ontology and rule components, the former (tight coupling)

supports a passive knowledge query and as such makes use of deductive rules. The ontology and rule engine have been tightly coupled as:

- The ontology involves established artefacts that are known and conceptualized once, following the requirements capture and user scenario definition stage, by domain experts, and then refined and maintained in an iterative and incremental way.
- The ontology maintenance process is managed separately from the rule system as the ontology is developed as part of a wider framework, namely the built environment domain.
- All terms in the rules are drawn from the developed ontology.
- This approach is supported by an established and open-source rule and ontology integration development environment (i.e. Protégé).

Description Logic Program (DLP) and SWRL are two well-known tightly coupled integration models. DLP is limited in that it relies on the intersection of DL and LP leading to a limited knowledge representation. Conversely, SWRL relies mainly on OWL-DL with additional Datalog rules from LP to address property chaining limitations in OWL-DL. Moreover, variables used in each rules head should also feature in their body (through DL-safe) to ensure the decidable reasoning of the rule engine. The availability of SWRL rule and ontology integration development in the popular Protégé environment has motivated the use of SWRL.

### 3.4. Resource discoverability and interoperability signposting

To promote automated and simplified manual discoverability of water network virtual resources, and to leverage recent developments in the IoT field, the knowledge-based solution was integrated with the recommendations of the Hypercat standard (BSI, 2016), due to its recent community adoption and standardisation. This mandated the modelling and alignment of the simple Hypercat metamodel as an OWL ontology with the water value chain ontology produced. It was also chosen to support the Hypercat RESTful API through a simple wrapper around the JENA ARQ query engine utilised as shown in Fig. 2. This wrapper converted Hypercat queries into SPARQL SELECT queries, so as to retrieve resource descriptions from the knowledge base via the Apache Jena ARQ library. After executing the query, the wrapper then converted the resulting object into a Hypercat compliant catalogue object.

Provisioning for the retrieval of resource information via the Hypercat approach allowed for simpler discoverability, owing to the accessible nature of the Hypercat specification. This also supported greater reusability of the developed solution, through applications developed for the Hypercat ecosystem. By querying the Hypercat API of the knowledge-management service, a catalogue was returned describing all of the active sensors on the network and the service's endpoints, and this was updated automatically as entities were added or removed.

The incorporated API provided an alternative, simpler mechanism by which to retrieve information about sensors installed in the target systems than the SPARQL endpoint. Specifically, the Hypercat-enabled data included the location, type, URI, API type, latest reading value, latest reading time, unit of measure, name of the sensor, and a textual description. Also, a pilot-site agnostic upper catalogue was exposed which provided discoverability of the available knowledge bases and accompanying pilot sites. As the sensor readings were hosted on a KairosDB server, this 'signposting' would allow a developer to integrate the available data into an application with far less difficulty. For a full semantic description of the sensor, sensing method, and physical contextualisation of the data, the developer could then query the full knowledge base via the SPARQL endpoint if desired. Whilst the sensors were the only entities which were Hypercat-enabled, it would have been possible to Hypercat-enable the entire knowledge-base, but this was not required for the use cases considered likely.

### 3.5. Semantic inference and rules engine

The inference and rule engine was closely coupled with the knowledge management software, due to their related functionality, and the need for the inference to maintain pace with updates to the knowledge base. This relied on a combination of the native Apache Jena inference engine, the Pellet engine, and the Drools rule engine; the latter of which handled the triggering of business rules.

Apache Jena natively supports 4 types of reasoner based on the architecture: transitive reasoner, RDFS rule reasoner, OWL/Lite reasoners, and generic rule reasoner (Apache Jena n.d.). These increase in inference capability from the transitive reasoner to the generic rule reasoner, although even the most capable of the Jena reasoners typically achieves less inference than the Pellet reasoner, due to Jena being RDF based and Pellet considering the entire conjunctive query (Complexible, n.d.).

As the Jena API integrates the Pellet inference engine (Complexible, n.d.) with little effort, and the Pellet engine is well-regarded for capability and speed, the Pellet reasoning engine was chosen to exploit the maximum potential from the OWL axioms. Further, Pellet relaxes OWL-DL restrictions on the OWL-Full features, and allows the majority of SWRL built-in atoms. This meant that Pellet could reason over rules which included maths features and numerical comparisons, which have been included in the developed SWRL rule set. However, during testing this inference capability was found to be slower, and less reliable, than the use of a separate Drools engine. The Drools rule engine was incorporated into the platform to augment the inference capabilities of the Pellet reasoner, which achieved acceptable inference completeness and speeds over native OWL axioms, but not over heuristic business rules.

The Drools rule engine is specifically designed for business rules, and uses forward-chaining and backward-chaining to infer new knowledge from existing data, with rules stated in either the Drools native language or a decision table. Drools reasoning is optimised for when x, then y business rules using their PHREAK algorithm; an evolution of the Rete algorithm, whereas Pellet is optimised for knowledge inference using tableaux algorithms over description logic axioms. Therefore, the Drools engine is better able to handle a large number of rules, with faster and more reliable firing of complex rules than Pellet. Also, it is important to note that SWRL limits the kind of rules which can be expressed using built-in atoms, as SWRL only natively handles comparisons and simple maths, string, date and URI functions. Even moderately complex rules can become difficult for humans to read and write, whereas the Drools Rule Language is comparatively simple to write, and yet is more expressive. Drools also allows more complex pattern matching within the conditional part of a rule and more complete functionality in the consequence part, including update, insert, and delete functions. SWRL cannot be used to delete or insert new named entities, modify properties used in the antecedent, count instances or test negated atoms, due to the monotonic nature of SWRL. Whilst the rules were formalised in SWRL and converted to the Drools rule language to experiment with a complete semantic web approach, expressing them in the native Drools language, or a Drools Domain Specific Language (DSL) is an area of further research.

### 3.6. API overview and illustrative usage

The knowledge management system offered a variety of APIs to support the retrieval of knowledge by tools of varying complexity, as shown in Table 2. This could integrate both knowledge-based applications and data-driven applications, and could be used at various lifecycle stages. A generic use of these endpoints would be to firstly discover resources, then retrieve data based on the nature of the client application. A generic process diagram for using these APIs is illustrated in Fig. 3.

As shown in Fig. 3, it would be typical for applications to first query the Hypercat endpoint, to determine the online resources. If the application has already determined the available site knowledge bases and endpoints, it may directly query the site's Hypercat endpoint to then discover the online sensors at that site. At that point, the usage of the system would vary depending on the nature of the client application; a simple dashboard for example may only query the simplified API to determine some common static knowledge about the network and the latest sensor readings. Alternatively, a more complex application may query the knowledge base directly via SPARQL to retrieve detailed static data. An application could also query the timeseries endpoint to retrieve dynamic sensor data, and then query the SPARQL endpoint to retrieve semantic context for the dynamic data, to facilitate its correct use in the client application. An example which highlights the value of these varied endpoints is provided in section 6.

## 4. A knowledge-based approach to water management

This section presents the process and artefacts related to the knowledge-based approach adopted, which includes the development methodology, the domain ontology and its validation, and the instantiation of the ontology from legacy data.

### 4.1. Requirements and ontology engineering

The ontology service's requirements and curation process primarily used the well-regarded NeOn methodology (Gómez-Pérez et al., 2008) for manual curation from existing semantic resources. Minor adaptations were made to this approach, due to changes in the technological landscape in the past 9 years since the NeOn methodology was published, such as the growth of the Internet of Things, which has fundamentally shifted the way the internet is used. Also, a key requirement was for the ontology to have value outside of the target system as a benchmark in the field. This involved balancing the knowledge engineering objectives prioritised by NeOn with the software engineering objectives of the overall IoT project, and the softer requirements from the domain experts of fostering ownership and human intelligibility.

In order to scope the ontology, a literature review was conducted of the semantic resources in the field, to give context to the model. Secondly, the software requirements were decomposed further into competency questions, conceptually orchestrated through the project's scenarios. These were then formed as a set of formal SPARQL queries which the ontology was required to answer, and which were adapted as the project matured and the role of the ontology service became clearer. The initial intent of the ontology was to capture the physical, cyber, and social concepts relevant to network-level operational management in water utility systems; to unify the knowledge management of geospatial data, domain specific concepts, and IoT metadata. The boundaries of the ontologys scope were also clarified beyond the initial intent by stating those concepts which it would not focus on: natural artefacts, electricity consumption, non-domestic consumers, the internal operation of water assets, and financial aspects. An indicative excerpt of the competency questions is now provided:

- How much water is domicile X currently consuming?
- What material is pipe X made from?
- How frequently does sensor X report observations?
- Which are the output pipes of asset X?
- How much water is currently in reservoir X?

- Where is asset X located?

- What is smart valve Xs current state?
- What does sensor X measure, in which units, and what was its latest observation?
- Who is the designated manager for asset X?

Following scope formalisation, existing models were manually selected and reworked into a base ontology, which was extended

**Table 2**
Implemented knowledge management API.

| Endpoint | Description | Parameters | #Response type |
|---|---|---|---|
| GET/cat | Retrieves a catalogue of knowledge bases on the server | None | JSON |
| GET/{site name}/cat | Retrieves a catalogue of resources available at a site | None | JSON |
| /{site name}/sparql | Standard SPARQL endpoint | query | JSON |
| GET/{site name}/water/ … | Convenience endpoint for common water queries | query | JSON |
| GET/{site name}/data | KairosDB endpoint for timeseries data | query | JSON |

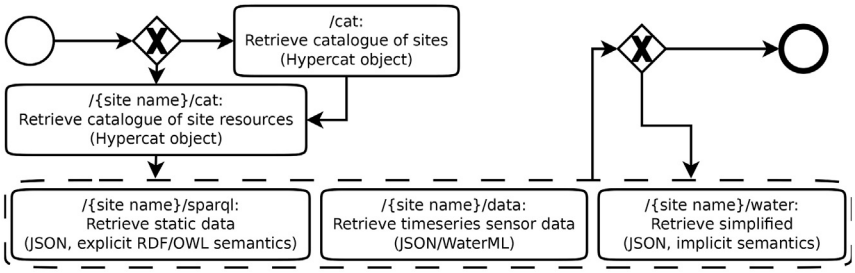


**Fig. 3.** Generalized process of using knowledge base APIs.

with domain and project specific concepts and T-Box axioms, to meet the competency questions derived for each of the project's impact scenarios. This included the reuse of a typical GIS schema and the extension of a socio-technical system model (Dam, 2009). The resulting artefact was then collaboratively iterated with water industry experts towards a consensus until the candidate ontology was deemed an accurate and sufficient description of the water management domain. Following this, further alignments were sought with external semantic resources to foster interoperability.

### 4.2. Water value chain ontology

The water value chain ontology modelled the concepts, relationships, properties, datatypes, restrictions, and logic, in water networks from abstraction to discharge, using OWL constructs. This was grounded in a view of the network as 3 main interlinked systems: physical, social, and cyber. The ontology therefore captured domain knowledge from these 3 perspectives, and across each of these 3 perspectives.

The physical water network model was an extension of a typical GIS schema from a water utility, and so primarily described the physical network assets and their properties. This included objects such as pipes, fittings, manholes, pumps, and treatment plants, and properties such as pipe diameters, fitting locations, and manhole types. The GIS schema was extended through abstraction into a metamodel of socio-technical systems, and was also extended with descriptions of domestic properties, natural water bodies, and physical phenomena and fluid dynamics. The model fundamentally viewed the physical network as a network of nodes and arcs, with each node potentially also being a super-node containing a subsystem of nodes and arcs, as in the case of a treatment plant.

The social water network model described the organisations and people involved in the water value chain, such as utility companies, regulators, and local governments, as well as different functional managers and persons within these organisations. Similar to the physical model, this adopted a 2 level of detail approach, whereby organisations were viewed as supersystems of people, and their relationships and processes. This modelling decision was taken as it was recognised that organisational-level modelling would be sufficient in many cases, but specifying the person responsible for a specific asset may be required for certain applications. Socio-technical relationships were also modelled, such as ownership, management responsibility, and regulations.

The cyber water network model described the sensors, actuators, data, and software, in the ICT system which supports the network's management. This was accomplished as an extension of the W3C SSN ontology (Compton et al., 2012), although observations were not stored in RDF; the ontology instead described where these observations were stored in the timeseries database. This allowed the observations to be contextualised but also stored in a scalable manner. Typical metadata was described, such as sensing frequency and accuracy, but further systemic context was also captured regarding the meaning of the data. Specifically, the nature of the sensed phenomenon and the physical entities being observed were provided with classes and slots for description in the ontology. Cyber-physical relationships were therefore described to formalise relationships between sensors, data, time series database fields, network assets, and fluid dynamics properties. This is illustrated in Fig. 4.

The ontology's grounding in the SSN ontology was facilitated by an adoption of the same upper ontology, the DOLCE+DnS Ultralite (DUL) upper ontology (Gangemi, n.d.). This integrates the DOLCE Lite-Plus library with the Descriptions and Situations ontology, and simplifies it for practical use. Therefore, the developed ontology uses similar design patterns to those of the DUL ontology, and hence focuses on the properties (SSN:property) and roles of the designed artefacts (DUL:DesignedArtifact) which participate in processes (DUL:Process) in the water network, as well as the agents (DUL:SocialAgent) which control these artefacts, and especially the observations (SSN:observation, subclass of DUL:Situation) made by sensors, and their resulting information objects (DUL:InformationObject). The majority of the ontology is hence an extension of these classes with subclasses and additional properties, although concepts related to network topology (i.e. graph theory) and water consumption behavior were also added.

### 4.3. Ontology validation

Following the development of the ontology & accompanying software, the requirements were tested against, to evaluate the ontology and the overall solution's acceptability against initial intentions. This involved a number of specific workshops with domain and software experts.

The initial, automated check of the ontology's consistency through the built in Protégé reasoner was consistently passed, such that the ontology does not contain contradictory statements. The competency questions were answered by the ontology after being posed as SPARQL queries. The domain expert validation was conducted separately with the domain expert partners through one day workshops, and in both cases the ontology's modelling choices were broadly validated, the majority of the detailed modelling choices were validated and corroborated between workshops, and some revisions and extensions were suggested. An additional workshop with the WISDOM partners and external industrial experts was then also conducted, which served to validate that the changes made were sufficient and hence that the ontology was sufficient. The organisation types of these companies are shown in Fig. 5.

The ontology was therefore considered by a wide range of stakeholders in the water value chain, most of whom had little bias towards the project. This offered a broad view on the ontology and hence tested its extent, as well as its detail in areas of the water value chain which the WISDOM partners were not experts in. Consensus was reached that the ontology represented a shared and sufficient conceptualization of the domain by this group, which was a significant milestone in its validation. Some of the comments from the SIG expert validation session were:

- The ontology addresses the problem of interacting between tools (GIS, SAP, customer data)
- Include alarms as well as sensors
- ‘Governing body’ is also called ‘regulator’
- Include ‘water testing company’

These comments were all used to revise the ontology. The majority of the comments however were advisory or generic, such as regarding possible future work, rather than required changes in the scope currently addressed. Examples of these comments were:

- The work could be considered as a type of enterprise service bus
- An ontology is also called a taxonomy
- Sensors could also be ‘social sensors’, which report numbers of tweets etc
- Collaboration relationships exist between utilities which share a water resource

Table 3 below presents an example outcome of the competency question testing, showing how the deployment answers the questions when formalised as SPARQL queries, where the queries were answered in circa 15 ms.

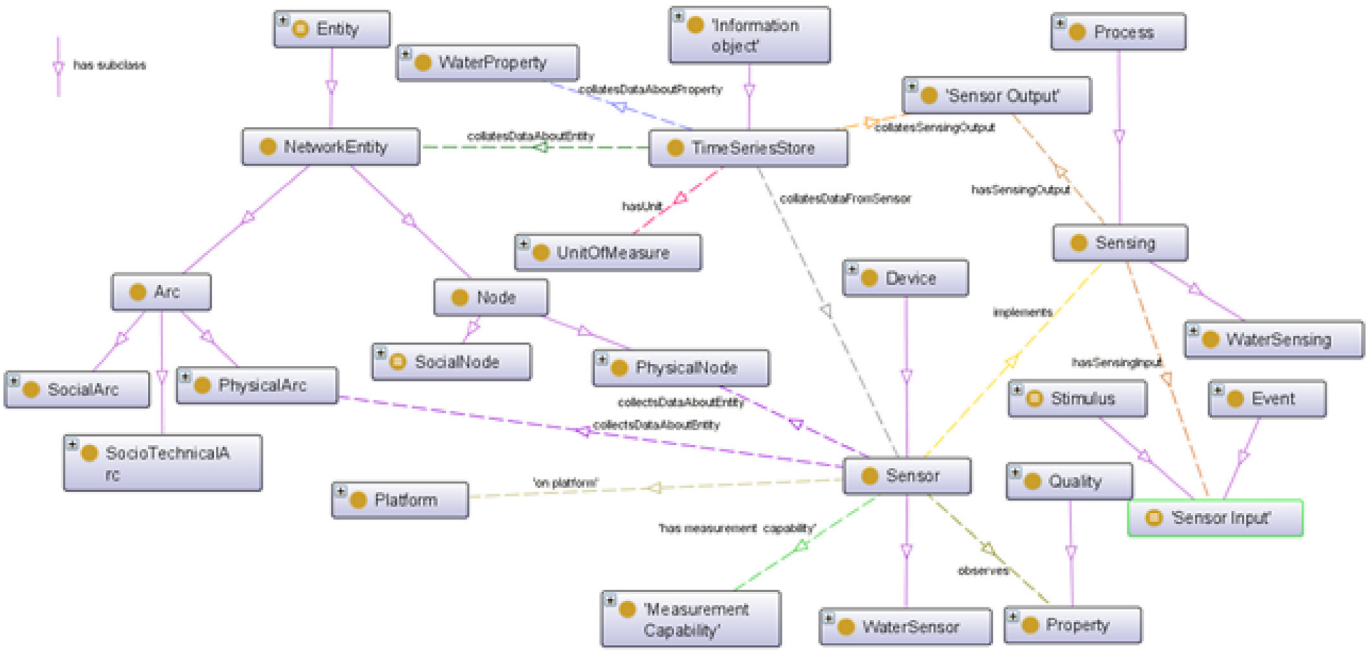


**Fig. 4.** Key cyber-physical relationships in the water value chain ontology (inverses not shown).

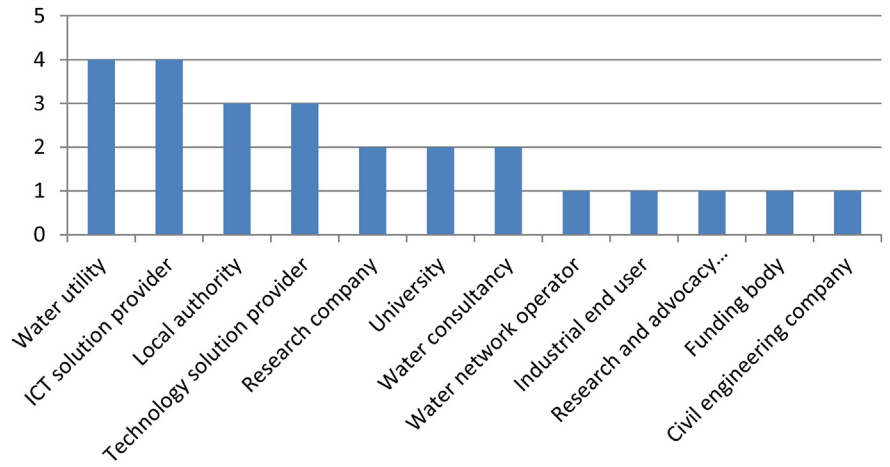


**Fig. 5.** Organisations involved in the validation process.

**Table 3**
Example competency question (prefix statements omitted).

| *Natural language question:* |
|---|
| What is sensor E2000's current reading? |
| *SPARQL query:* |
| SELECT ?reading<br>WHERE {<br>wis:E2000 rdf:type wis:LevelSensor.<br>wis:E2000 wis:hasLatestOutput ?output.<br>?output dul:hasDataValue ?reading } |
| *Output (CSV format):* |
| reading<br>2.00 |

### 4.4. Instantiation through reuse of GIS and EPANET data

Following from the domain ontology development, the knowledge base development involved the instantiation of this domain model at each pilot site. This section presents an overview of the process and outputs of this work, which resulted in pilot site knowledge bases for both clean and waste water networks at Welsh pilot sites. This work adopts a novel methodology of converting GIS and EPANET data to RDF data through a Python script. Each knowledge base used the same domain ontology, but was itself separate to the other knowledge bases, as their integration was not beneficial to the targeted use cases. Each knowledge base was produced by reusing existing data from utility company partners, and then extended manually.

The knowledge bases were instantiated primarily by reusing GIS, sensor, and EPANET data provided by industrial partners, through a Python script. This was then enriched manually with further sensor and social entity descriptions. Each pilot sites Abox was kept separate, but could be trivially merged with the Tbox if required, or kept separate until storage in the triple store. After pre-processing, the Python script used RDF libraries (RDFlib, 2014) and CSV libraries to parse, build and output relevant files, by iterating

over each line of the input file and creating named entities, data properties, and object properties as appropriate. The main reused GIS objects were system valves, meters, mains, control valves, hydrants, asset sensors, conduits, nodes, pumps, sensors and subcatchments as summarised in Table 4 below. In total, these represented 6 MB of data and resulted in 100 k RDF A-box triples, including 10 k named entities.

This model transformation converted the data from simple lists of values with implicit semantics, to an RDF graph based on the developed model, as shown in Fig. 6, with the benefits of explicit semantics, high levels of expressiveness, and unique IDs. Where appropriate, the properties themselves are objects with their own sets of object and data properties; therefore, the meaning is far more explicit and could be reused by other software with greater confidence in its meaning, hence requiring less investment to ensure successful interoperability.

Expressiveness and extensibility are significant benefits of the semantic modelling approach. For example, the 'hasMaterial' property is an object property which connects a pipe to a material, the material can then be described by properties such as surface roughness, for hydraulic modelling, or fracture toughness, for earthquake resilience simulation. These examples show respectively that the approach allows greater value to be derived from the initial data by formally describing it in a machine interpretable manner, and allows extensibility beyond its initial purpose with little effort. Further, semantic inference over the RDF form of the data allows greater value to be derived from the original data. For example the 'goesToIpid' is a datatype property which connects a pipe to an integer, but an SWRL rule was used to infer the knowledge that, given that the integer is the ID of another pipe, the latter is downstream of the former pipe.

## 5. Example use case: fault impact inference

To demonstrate the value of the SWoT approach, and especially the knowledge-based services, an example use case is presented which highlights the value of a platform which exposes dynamic data, higher-order knowledge, and inference-based knowledge. This forms the back-end to the GUI presented in section 6. The use case itself is now described, before presenting the supporting technology for each required human-machine interaction towards the desired decision support.

### 5.1. Use case description and requirements

The proposed use case assumed that a fault had occurred within the water value chain, such as a pipe blockage. The proposed stages of decision making towards addressing this are: i) identification, ii) assessment, iii) impact mitigation, and iv) resolution. Identification is the process of acknowledging the issue and notifying the appropriate person. Assessment is the observation of the issue and appropriate data, and applying expert knowledge to decide on the following actions. Impact mitigation is the minimisation of the effect of the issue on the performance indicators of the organisation through immediate action. Resolution is the restoring of normal functioning of the system through maintenance or a revised system design. Finally, an additional stage of learning was identified, regarding a long term feedback loop so as to reduce the occurrence and impact of similar or related faults in the future, but this was deemed to require a separate use case. Ideally, the proposed application should assist the domain expert through this process to the greatest extent possible, and be supported by its back-end infrastructure so as to enable this with ease.

**Table 4**
Summary of water and wastewater network input data.

| Entity type | Number of entities | Number of properties |
|---|---|---|
| Sensors at clean assets | 29 | 5 |
| Hydrants | 261 | 18 |
| Control valves | 46 | 18 |
| Clean mains | 1517 | 19 |
| Water meters | 26 | 15 |
| System valves | 485 | 19 |

The proposed solution for this use case attempted to leverage ICT greatly to assist the decision maker, with an emphasis on back-end support and interoperability. This made significant use of the Drools rule engine, and custom semantic rules, alongside the described hybrid knowledge management platform. By prioritising this processing at the platform layer, the resultant knowledge could be shared by multiple applications, for example if the same issue affected different business processes and hence different expert decision makers. By offering a single point of truth with a range of data and contextualisation methods to various applications, the response of the organisation can be streamlined, coordinated, and more effective.

As a core software requirement, the inference engine needed to have the ability to detect problems in the network, and then determine the network entities affected by the problem. This would serve as decision support for operations managers, by empowering them with knowledge about the cause and impact of the problem, for example this would help to identify customers affected by a network blockage and hence proactively engage with them as opposed to waiting for customers to issue a complaint.

### 5.2. Knowledge-based decision support

ICT was leveraged across the 4 stages identified within the scope of this use case. This is now described for each stage in turn before the application logic is presented in detail.

Identification of faults occurred through rule-based detection. The concept of an alert was modelled in the domain ontology, as well as the problem itself which caused the alert, and object properties then connected these to the physical network entities, their topological representation, and the related sensors and properties. Each alert had an associated 'alert condition', which could be a complex fault detection algorithm, but a simple 'acceptable range' was used for proof of concept. This range then had an upper and lower bound, which an SWRL rule was able to evaluate against the latest observation from the appropriate sensor and change the status of the alert if needed, as shown in the next subsection. This acknowledged the issue within the ICT system, and notified the expert of the issue through the GUI, which could be supplemented with automated email or text notifications if suitable.

Assessment of the issue was deemed to require an intuitive presentation of the cause of the alert and its implications, with the ability for deeper interrogation of underlying data and logic. This was achieved through a combination of rule-based inference, graphical presentation of knowledge and data exploration functionality. Firstly, the rule engine determined the affected network entities, as well as the likely problem which caused the alert, and its severity and detection time. By updating the knowledge base with this information, it was then discovered by the application through the Hypercat API, so as to present knowledge about the problem in an intuitive and graphical manner. Finally, underlying raw data could be interrogated through the GUI, following discovery of the sensors and their endpoints through the Hypercat API. This returned dynamic data from the time series database to present graphs of the sensor readings relevant to the issue, as well as static

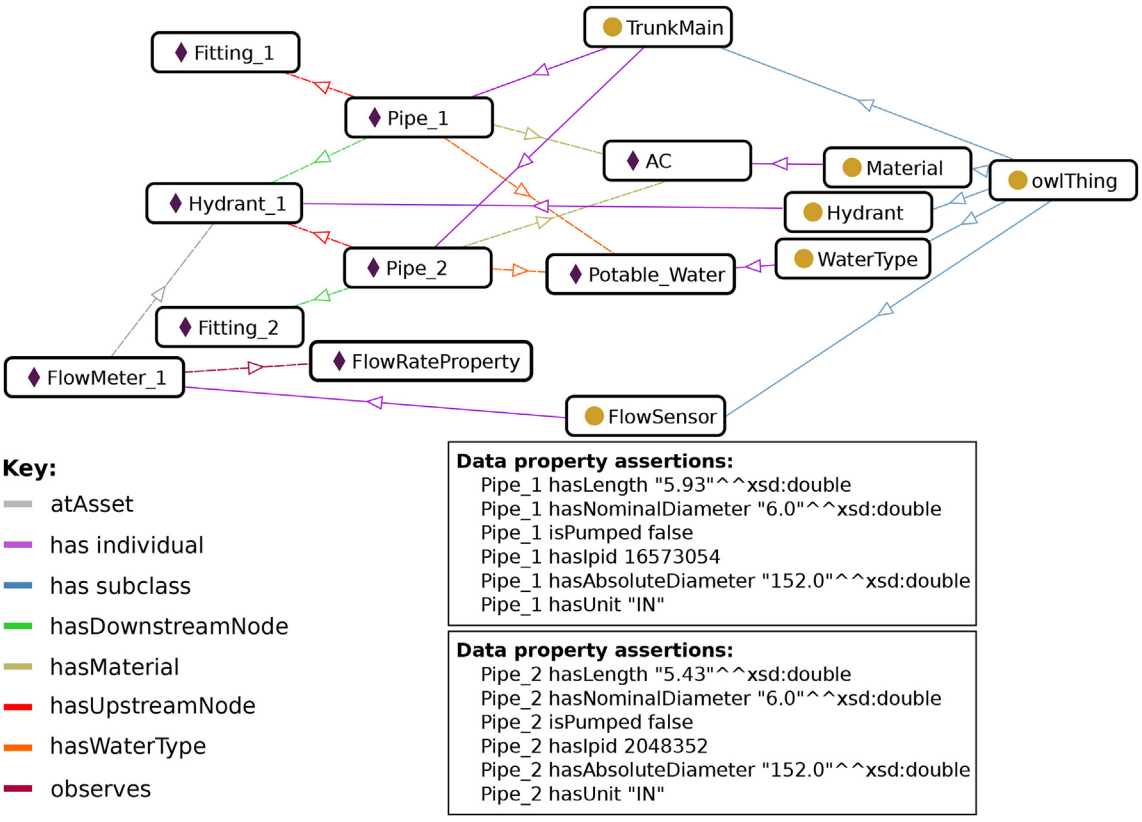


**Fig. 6.** Example of graph database manifestation showing partial graph, object properties, and data properties.

data about the network entities from the knowledge base.

The described rules would also assist with impact mitigation, by identifying the customers likely to be affected by the issue and exposing their details in a secure manner to the appropriate member of staff. Also, by including knowledge regarding the organisation's performance indicators, the inference engine could offer targeted information and suggested actions. This would support the stage of resolution, which could also be supported by extending the knowledge-base to cover the asset management processes, people, and organisations, as this could identify nearby people able to resolve the issue. The rules utilised throughout this process of decision support are now described in detail, as they form a core part of the contribution.

### 5.3. Inference engine rules

The required inference capability for this use case is illustrated in Fig. 7 below, where the green arrows indicate explicit knowledge, and red arrows represent inferred knowledge.

Towards achieving the inference ability described, 28 SWRL rules were written, which were utilised by the inference engine alongside the OWL axioms present in the domain ontology. An SWRL rule follows the standard 'if THIS then THAT' approach of rule based logic, where the 'THIS' and 'THAT' portions consist of SWRL atoms. An SWRL atom is a statement of truth, such that 'if THIS (statement is true) then THAT (statement is also true)'. A full description of SWRL is available from W3C (2004). SWRL built-ins include functions for comparisons, maths, and string manipulation etc. This section now describes the rules produced for each use case, and presents them in SWRL syntax.

#### 5.3.1. deployedAtEntity

As sensors were not explicitly described in terms of the node which they were deployed at, inferring their asset-deployment relationship was fundamental to the knowledge needed to contextualise the capability of the deployed sensors. This capability was provided by rule 1.

$$Sensor(?S) \wedge atAsset(?S, ?A) \wedge TopologicalNetworkEntity(?E) \wedge atAsset(?E, ?A) \Rightarrow deployedAtEntity(?S, ?E) \quad (1)$$

#### 5.3.2. observes

Whilst it may be explicitly stated in the knowledge base that a sensor observes a certain property, the SSN ontology also offers an alternative modelling pattern, such that a sensor is deemed to have a 'measurement capability', which is then itself related to the observed property. Where this was the case, it was useful to infer the direct link between the sensor and the property, to facilitate the previous rules. This was accomplished through rule 2.

$$hasMeasurementCapability(?S, ?MC) \wedge forProperty(?MC, ?P) \Rightarrow observes(?S, ?P) \quad (2)$$

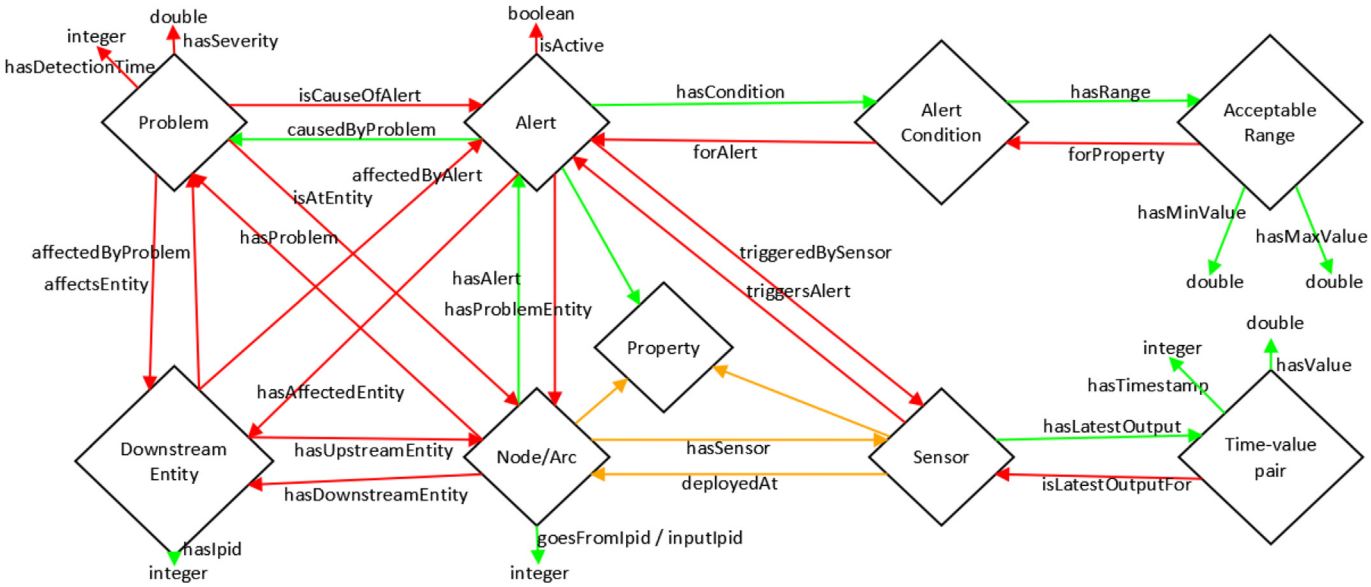


**Fig. 7.** Problem detection and alert propagation inference use case requirements.

### 5.3.3. isActive

This property determines whether an alert is active; relating an alert to a Boolean literal. Inferring this shows the key benefits of the knowledge based approach utilised: as the sensor descriptions, asset descriptions, alert descriptions, and dynamic data were all semantically related, it was possible to directly infer whether or not a sensor reading should trigger an alert. If an alert had an acceptable range, and it was related to a specific sensor, and that sensor's latest reading fell outside of the acceptable range, the alarm was triggered. This could occur if the reading was greater than the allowable maximum, or smaller than the allowable minimum, as otherwise the alarm is not active. This was formalised through rules 3, 4, and 5.

### 5.3.4. hasDownstreamEntity

In order to generalise pipes, pumps, and reservoirs etc. and determine what is upstream or downstream of an entity, it was useful to use the IPID values held in the legacy GIS database to infer knowledge about flow chronology through the entities. This allowed later inference of whether an entity was affected by any given problem, and greatly simplified those rules. Rules 6 and 7 therefore evaluated this for each entity.

$$goesFromIpid(?p,?i) \wedge hasIpid(?u,?i) \Rightarrow hasUpstreamEntity(?p,?u) \wedge hasDownstreamEntity(?u,?p) \quad (6)$$

$$goesToIpid(?p,?i) \wedge hasIpid(?d,?i) \Rightarrow hasUpstreamEntity(?d,?p) \wedge hasDownstreamEntity(?p,?d) \quad (7)$$

### 5.3.5. hasProblem

In the situation that an alert was active, and the alert was caused by a certain problem, it was beneficial to relate the problem entity to the problem directly, which rule 8 achieved.

$$\begin{aligned} &WaterAlert(?A) \wedge hasAlertCondition(?A,?AC1) \wedge hasAcceptableRange(?AC1,?AR) \\ &\quad \wedge hasMaxValue(?AR,?Xmax) \wedge Sensor(?S) \wedge triggersAlert(?S,?A) \\ &\quad \wedge hasLatestOutput(?S,?TVP) \wedge hasValue(?TVP,?X) \wedge swrlb:greaterThan(?X,?Xmax) \\ &\quad \Rightarrow isActive(?A,true) \end{aligned} \quad (3)$$

$$\begin{aligned} &WaterAlert(?A) \wedge hasAlertCondition(?A,?AC1) \wedge hasAcceptableRange(?AC1,?AR) \\ &\quad \wedge hasMinValue(?AR,?Xmin) \wedge Sensor(?S) \wedge triggersAlert(?S,?A) \\ &\quad \wedge hasLatestOutput(?S,?TVP) \wedge hasValue(?TVP,?X) \wedge swrlb:lessThan(?X,?Xmin) \\ &\quad \Rightarrow isActive(?A,true) \end{aligned} \quad (4)$$

$$\begin{aligned} &WaterAlert(?A) \wedge hasAlertCondition(?A,?AC1) \wedge hasAcceptableRange(?AC1,?AR) \\ &\quad \wedge hasMinValue(?AR,?Xmin) \wedge hasMaxValue(?AR,?Xmax) \wedge Sensor(?S) \wedge triggersAlert(?S,?A) \\ &\quad \wedge hasLatestOutput(?S,?TVP) \wedge hasValue(?TVP,?X) \wedge swrlb:lessThanOrEqual(?X,?Xmax) \wedge swrlb \\ &\quad :greaterThanOrEqual(?X,?Xmin) \\ &\quad \Rightarrow isActive(?A,false) \end{aligned} \quad (5)$$

$$hasAlert(?E, ?A) \wedge forProblem(?A, ?P) \wedge isActive(?A, true) \Rightarrow hasProblem(?E, ?P) \quad (8)$$

*5.3.6. hasAffectedEntity*

Tracing the impact of a problem downstream in a water network to determine further problems which the problem could cause, and its negative consequences for customers, was both highly beneficial and challenging, due to the system complexity. Further, if a problem was reported at a downstream entity, one expert task is tracing backwards in the value chain to determine if an upstream problem could be causing it. Rule 9 aimed to empower water experts by telling them if an entity is affected by any upstream problems. Further, the knowledge of an entity being affected by an upstream problem could be used automatically by a further rule to infer a likely problem at that entity, and even a required action to proactively mitigate the overall impact of the initial problem. Note that 'hasProblemEntity' is a sub-property of 'hasAffectedEntity', which allowed this inference to propagate to all downstream elements.

$$WaterAlert(?A) \wedge isActive(?A, true) \wedge hasAffectedEntity(?A, ?E) \wedge hasDownstreamEntity(?E, ?D) \Rightarrow hasAffectedEntity(?A, ?D) \quad (9)$$

*5.3.7. affectedByProblem*

Rule 10 continued the benefits of rule 9 by directly linking the downstream entity with the problem which it was affected by. Note that 'hasProblem' is a sub-property of 'affectedByProblem', allowing the inference to propagate to all downstream entities.

$$hasDownstreamEntity(?E, ?D) \wedge affectedByProblem(?E, ?P) \Rightarrow affectedByProblem(?D, ?P) \quad (10)$$

*5.3.8. hasSeverity*

This inference ability explored the possibility of evaluating how severe a problem was, based on how far outside the acceptable range a latest sensor reading was. This was achieved in a simple manner by finding the relative absolute distance which the reading is outside of the acceptable range. This could be explored further with more specific use cases, and more domain knowledge about the criteria for problem severity, such as the likely total future impact on the organisation's KPIs. However, the work showed that the knowledge-based approach allowed further knowledge to be derived quite easily about the current situation, to empower decision makers without them having to do their own analysis of the data. Firstly, in the case of the reading being greater than the maximum allowable value, severity was defined by 11.

$$\text{if } \mathrm{Val}_{\mathrm{Actual}} > \mathrm{Val}_{\mathrm{max}}: \quad \text{Severity} = \frac{\mathrm{Val}_{\mathrm{Actual}} - \mathrm{Val}_{\mathrm{Max}}}{\left(\frac{\mathrm{Val}_{\mathrm{Max}} - \mathrm{Val}_{\mathrm{Min}}}{2}\right)} \quad (11)$$

If a problem caused an alert, and the alert was based on an acceptable range, and the alert was triggered by a sensor whose latest reading was above that range, the severity of the problem was calculated as per equation (11), which was implemented in SWRL as rule 12.

$$\begin{aligned} &Problem(?P) \wedge isCauseOfAlert(?P, ?A) \wedge WaterAlert(?A) \wedge hasAlertCondition(?A, ?AC1) \\ &\quad \wedge hasAcceptableRange(?AC1, ?AR) \wedge hasMinValue(?AR, ?Xmin) \wedge hasMaxValue(?AR, ?Xmax) \\ &\quad \wedge Sensor(?S) \wedge triggersAlert(?S, ?A) \wedge hasLatestOutput(?S, ?TVP) \\ &\qquad \wedge hasValue(?TVP, ?X) \wedge swrlb : greaterThan(?X, ?Xmax) \wedge swrlb \\ &\qquad : subtract(?x1, ?Xmax, ?Xmin) \wedge swrlb \\ &\qquad : divide(?x2, ?x1, 2) \wedge swrlb \\ &\qquad : subtract(?SevAbs, ?X, ?Xmax) \wedge swrlb \\ &\qquad : divide(?SevRel, ?SevAbs, ?x2) \\ &\qquad \Rightarrow hasSeverity(?P, ?SevRel) \end{aligned} \quad (12)$$

In parallel to the above rule, the opposite logic held when the sensor reading was below the minimum acceptable range, where the severity was calculated as per equation (13) below, and implemented as rule 14.

$$\text{if } \text{Val}_{\text{Actual}} < \text{Val}_{\text{min}}: \quad \text{Severity} = \frac{\text{Val}_{\text{Min}} - \text{Val}_{\text{Actual}}}{\left(\frac{\text{Val}_{\text{Max}} - \text{Val}_{\text{Min}}}{2}\right)} \tag{13}$$

$$\begin{aligned} &\mathit{Problem}(?P) \wedge \mathit{isCauseOfAlert}(?P, ?A) \wedge \mathit{WaterAlert}(?A) \wedge \mathit{hasAlertCondition}(?A, ?AC1) \\ &\quad \wedge \mathit{hasAcceptableRange}(?AC1, ?AR) \wedge \mathit{hasMinValue}(?AR, ?Xmin) \\ &\quad \wedge \mathit{hasMaxValue}(?AR, ?Xmax) \wedge \mathit{Sensor}(?S) \wedge \mathit{triggersAlert}(?S, ?A) \\ &\qquad \wedge \mathit{hasLatestOutput}(?S, ?TVP) \wedge \mathit{hasValue}(?TVP, ?X) \wedge \mathit{swrlb} : \mathit{lessThan}(?X, ?Xmin) \wedge \mathit{swrlb} \\ &\qquad : \mathit{subtract}(?x1, ?Xmax, ?Xmin) \wedge \mathit{swrlb} \\ &\qquad : \mathit{divide}(?x2, ?x1, 2) \wedge \mathit{swrlb} \\ &\qquad : \mathit{subtract}(?SevAbs, ?Xmin, ?X) \wedge \mathit{swrlb} \\ &\qquad : \mathit{divide}(?SevRel, ?SevAbs, ?x2) \\ &\qquad \Rightarrow \mathit{hasSeverity}(?P, ?SevRel) \end{aligned} \tag{14}$$

### 5.3.9. hasDetectionTime

Given that the knowledge base was iteratively updated as new sensor readings are received, and alerts may not be viewed immediately, it was deemed beneficial to inform decision makers exactly when a problem was first observed. This was achieved by noting the time at which the sensor's latest reading was outside the acceptable range, but the sensor's previous reading was inside the acceptable range. Rules 15 and 16 therefore state that if an alert has an acceptable range, and is triggered by a sensor, and the sensor's latest reading is outside that range, but its previous reading was inside the range, then the detection time of the problem is the latest reading's timestamp.

$$\begin{aligned} &\mathit{Problem}(?P) \wedge \mathit{isCauseOfAlert}(?P, ?A) \wedge \mathit{WaterAlert}(?A) \wedge \mathit{hasAlertCondition}(?A, ?AC1) \\ &\quad \wedge \mathit{hasAcceptableRange}(?AC1, ?AR) \wedge \mathit{hasMinValue}(?AR, ?Xmin) \wedge \mathit{hasMaxValue}(?AR, ?Xmax) \\ &\quad \wedge \mathit{Sensor}(?S) \wedge \mathit{triggersAlert}(?S, ?A) \wedge \mathit{hasLatestOutput}(?S, ?TVP) \wedge \mathit{hasValue}(?TVP, ?X) \\ &\qquad \wedge \mathit{swrlb} : \mathit{greaterThan}(?X, ?Xmax) \wedge \mathit{hasTimestamp}(?TVP, ?time) \wedge \mathit{hasPreviousOutput}(?S, ?TVPprev) \\ &\qquad\quad \wedge \mathit{differentFrom}(?TVP, ?TVPprev) \wedge \mathit{hasValue}(?TVPprev, ?Xprev) \wedge \mathit{swrlb} \\ &\qquad : \mathit{lessThanOrEqual}(?Xprev, ?Xmax) \wedge \mathit{swrlb} \\ &\qquad : \mathit{greaterThanOrEqual}(?Xprev, ?Xmin) \\ &\qquad \Rightarrow \mathit{hasDetectionTime}(?P, ?time) \end{aligned} \tag{15}$$

$$\begin{aligned} &\mathit{Problem}(?P) \wedge \mathit{isCauseOfAlert}(?P, ?A) \wedge \mathit{WaterAlert}(?A) \wedge \mathit{hasAlertCondition}(?A, ?AC1) \\ &\quad \wedge \mathit{hasAcceptableRange}(?AC1, ?AR) \wedge \mathit{hasMinValue}(?AR, ?Xmin) \wedge \mathit{hasMaxValue}(?AR, ?Xmax) \\ &\quad \wedge \mathit{Sensor}(?S) \wedge \mathit{triggersAlert}(?S, ?A) \wedge \mathit{hasLatestOutput}(?S, ?TVP) \wedge \mathit{hasValue}(?TVP, ?X) \\ &\qquad \wedge \mathit{swrlb} : \mathit{lessThan}(?X, ?Xmin) \wedge \mathit{hasTimestamp}(?TVP, ?time) \wedge \mathit{hasPreviousOutput}(?S, ?TVPprev) \\ &\qquad\quad \wedge \mathit{differentFrom}(?TVP, ?TVPprev) \wedge \mathit{hasValue}(?TVPprev, ?Xprev) \wedge \mathit{swrlb} \\ &\qquad : \mathit{lessThanOrEqual}(?Xprev, ?Xmax) \wedge \mathit{swrlb} \\ &\qquad : \mathit{greaterThanOrEqual}(?Xprev, ?Xmin) \\ &\qquad \Rightarrow \mathit{hasDetectionTime}(?P, ?time) \end{aligned} \tag{16}$$

## 5.4. Inference use case testing

The rules were tested individually during development to ensure they were able to infer the desired knowledge from the desired explicit knowledge, and to ensure that obvious similar situations didn't invoke false positive inferences, which the presented rule set passed. However, use case based testing of the rules in unison is a far more reliable test of the performance of the inference engine, as is far more likely to produce false positives due to the increased complexity in the explicit knowledge. This section therefore presents the use case based testing of the rules, by illustrating the explicit knowledge in an example instance of the use case, then presenting the inferred knowledge, followed by discussing the time taken, the robustness of the test, and the value of the inference achieved.

The rules were tested on a middle range laptop; a Samsung 900X, with an Intel i5 1.7 GHz processor and 8 GB of memory, on 64-bit Windows 7. The rules were tested in the Protégé software; given the computer specification and the testing environment it is expected that performance would be better within a dedicated high performance server or a cloud computing environment. The rules were tested such that the entire domain ontology was reasoned over, as well as the individuals specifically relevant to the use case. The input and output Aboxes sizes were determined directly through the RDFlib Python library by command line.

To test the inference capability, an instance of the use case was defined, whereby a reservoir node is connected formally to a level sensor, and has a tree of downstream nodes and arcs. It was considered that the sensor's latest reading indicated that the reservoir's water level was too low, and the ability of the inference engine to provide decision support was tested. The described test case is illustrated in Fig. 8, which also shows the named individuals

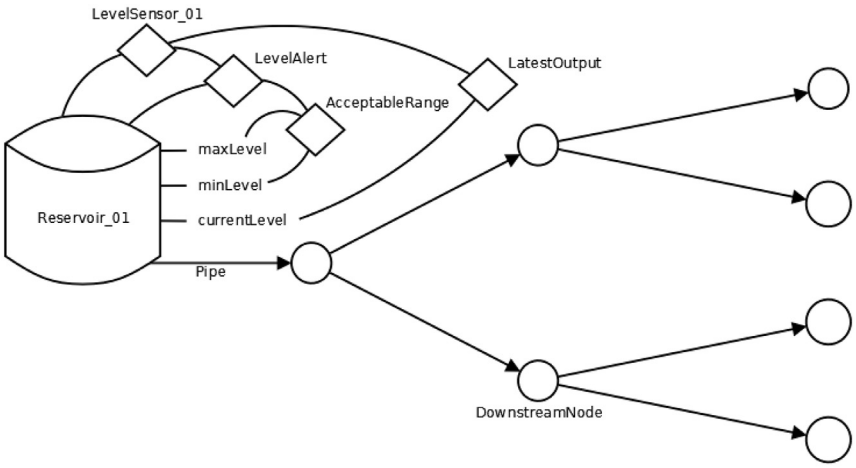


**Fig. 8.** Problem and alert inference test case illustration.

for the alert, and acceptable range and latest output.

The main desired outcomes of the inference engine were the triggering of the alert, the link between the problem and downstream entities, and knowledge about the time and severity of the problem. Following the application of the inference engine on the test knowledge base of 56 triples, the Abox contained 972 triples, meaning that 916 triples had been inferred, of 7185 total inferred axioms. This inference occurred in 1427 ms on the first instance (without caching), which reduced to circa 450 ms after caching. The desired knowledge primarily centres on the problem and downstream entity named individuals, so Fig. 9 displays the Abox knowledge at these entities following the inference. This shows that the node is linked to its upstream and downstream entities, is 'affected by' the active alert, and is 'affected by' the low level problem. Fig. 9 also shows that the problem individual is linked to all of the downstream nodes, and its severity and timestamp have been inferred.

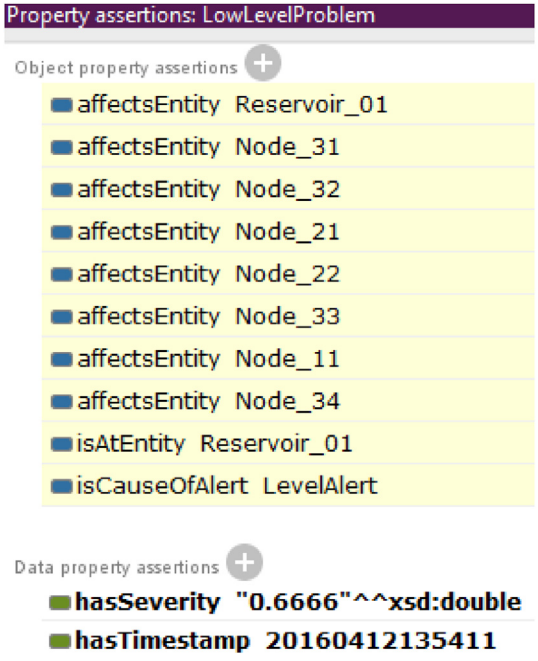


**Fig. 9.** Excerpt of resultant Abox knowledge after problem and alert inference testing.

Based on the inference achieved through the application of the inference engine to the test case, Fig. 10 below highlights some of the key inferred knowledge. Specifically, this shows that knowledge about the reservoir problem was inferred, and was linked directly to downstream entities. By integrating system knowledge in this manner, little work would be required to automatically infer required actions due to the impact of the problem on customers, as shown at the right hand side of Fig. 10. The knowledge-based system approach therefore allows the impact of problems to be more comprehensively checked; a task which is otherwise prohibited by the time taken, due to the complexity of the system. The inference ability demonstrated is a significant first step towards an integrated knowledge-based approach, and serves as proof of concept that the approach allows decision makers to be better informed in managing the water network.

Whilst further research around the potential of this rule-based approach is outside the scope of this paper, some key avenues are suggested:

- The knowledge of an entity being affected by an upstream problem could be used automatically by a further rule to infer a likely problem at the downstream entity, and even a required action to proactively mitigate the overall impact of the initial problem.
- If an entity is affected by an upstream problem, but has another upstream entity, then the former entity is only partially affected by the upstream problem. This could be explored further for the case of two upstream entities, both with different problems, or to infer more specific knowledge about how badly affected an entity is by an upstream problem, based on the relative flow rate it derives from the problematic upstream entity.
- The inference could also be extended with provenance knowledge: by determining if a sensor often malfunctions, or has 'drifted' and needs recalibrating, it would be possible to assign a 'reliability metric' to a sensor observation and subsequently infer the likelihood of a false positive when triggering an alert, which would further inform decision makers.

## 6. Tool implementation and evaluation

A decision support tool was developed as a proof-of-concept stage, so as to demonstrate the benefits of the proposed platform, within the context of the use case established in the previous section. The tool aimed to extend the state of the art of GIS tools, as

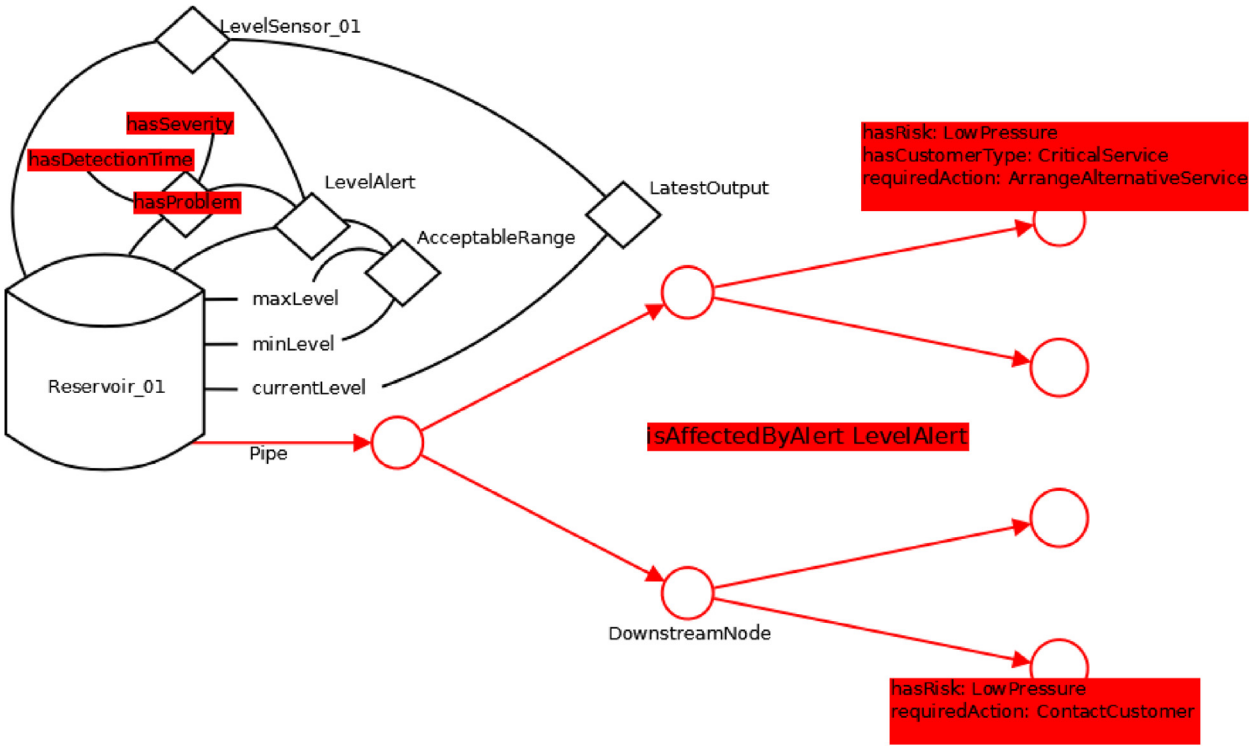


**Fig. 10.** Key knowledge inferred and extendable through the alert and problem inference testing.

well as typical water utility dashboards. The following subsection describes the interface from a user and technological perspective, and the subsequent subsection provides evidence of the software's performance.

### 6.1. Interactions with the knowledge management APIs

Given the described use case, a decision support tool was built as a client application for the knowledge management platform described. This utilised the various APIs exposed in a coherent manner to highlight the value of the approach, which is now described.

The first use of the knowledge management platform was a call to the Hypercat API to retrieve a list of pilot sites and knowledge bases, followed by a second call to discover the online sensors and alerts. The discovered SPARQL endpoints of each sites' knowledge base were then queried to retrieve descriptions of the water network objects such as pipes, assets, and sewer overflows. Once a sensor had been selected, as described in the following subsection, the timeseries endpoint was queried to retrieve its data for graphing. On discovering an active alert, the SPARQL endpoint was queried to retrieve further information such as the affected network entities and the problem's severity, based on the results of the inference engine's algorithms. This demonstrates the value of i) the Hypercat endpoint in aiding the discovery of remote resources and signposting their interfaces, and ii) the ontological grounding, in providing a common terminology, meaning and uniform resource identifiers for describing objects.

### 6.2. Decision support interface

The decision support tool used the Google Maps API (Google Developers, n.d.) to visualise the water network assets in an intuitive manner in a web-browser. CSS was used to mobile-optimise the page, such that it could be used on-site and away from utility workstations. Information about the assets was made available through context boxes on clicking assets. On clicking a sensor, an info window was displayed which showed a graph of the sensor's latest readings, which could be expanded for further investigation. These functions are illustrated in Fig. 11 below. It was envisaged that a decision maker may use these functions during normal asset management and operational management of the network, but the main use case considered was responding to a network fault.

If a fault was detected by the inference engine, the side menu was uncovered and an alert icon was shown. On clicking this icon, the alert investigation state was entered. On entering this state, the colour of the affected network entities was changed to the alert's colour (this provisioned for clarity in the case of multiple alerts), and basic alert information was shown in the side menu. The user could then click the alert information to present a greater level of detail, or could click network entities to view information pertaining to the alert. This state is illustrated in Fig. 12.

The tool's functions were programmed in pure Javascript and AJAX, to showcase the platform's various functions in a simple manner. Firstly, the network asset locations and descriptions were retrieved via a SPARQL query to the platform, and parsed from the returned JSON object. The sensor locations and descriptions were retrieved from the Hypercat API of the platform, and used to interoperate with the timeseries data API. This API was then used to retrieve the dynamic sensor data, to populate the graph of the sidebar. A more detailed interaction with the dynamic data could easily be achieved by using an existing Javascript library as a wrapper for the KairosDB API. The tool checked for active alerts at regular intervals through SPARQL queries, although this could have been achieved through a subscription-based approach. On entering the alert investigation state, the full description of the alert was retrieved from the knowledge base, including the impact inference knowledge produced by the SWRL rules and Drools engine. This

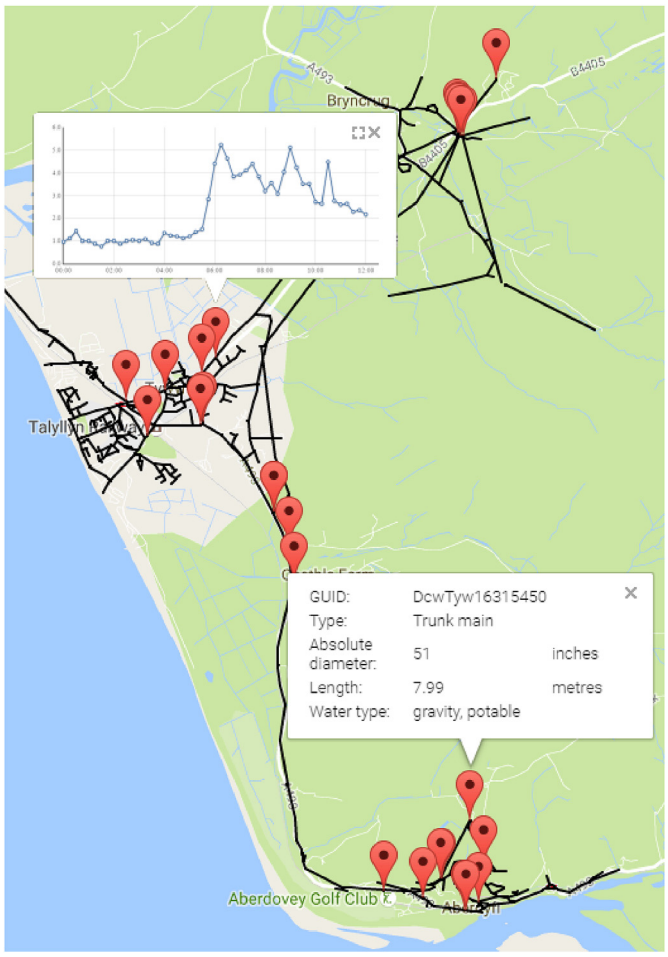


**Fig. 11.** Illustration of the user interface in network monitoring mode.

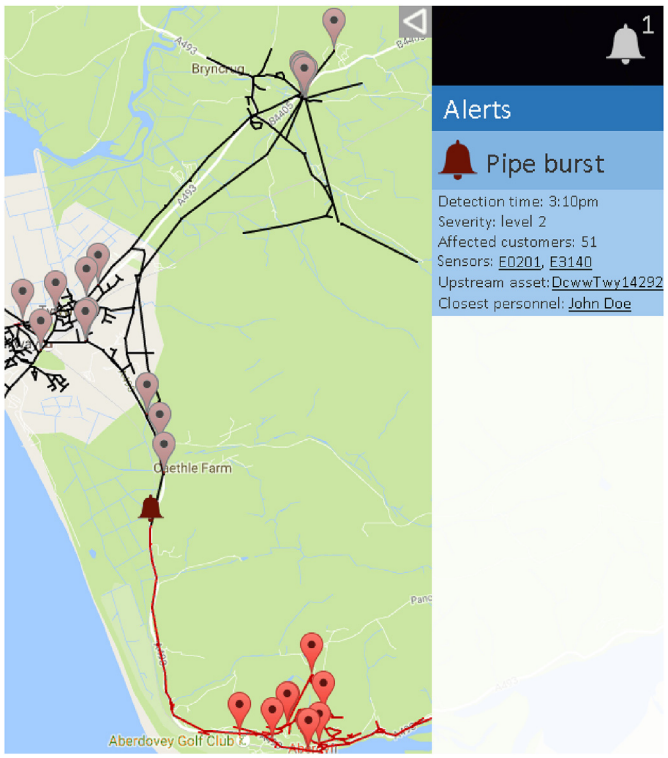


**Fig. 12.** Illustration of the user interface in alert mode.

was then used to update the map, and to populate the information in the sidebar.

### 6.3. Software performance

Towards evaluating the performance of the software for the given use case, this primarily involved determining the response time to network faults, and the accompanying user experience in receiving decision support. Given that the tool was only intended as a proof of concept for the Semantic Web of Things approach, the latter of these is outside the scope of this paper. However, an 'in vitro' test of the responsiveness to alerts, and a discussion of the tool's responsiveness in general, are feasible and valuable.

The total response time can be evaluated as the sum of the time for the sensor reading to be updated in the knowledge base, the time taken for the inference engine to reason over the updated readings, and the time taken for the active alert to be represented in the GUI. As the first of these three components depends entirely on the sensor network and communication technologies at a site, it is omitted here, as the current contribution is at the application layer, and is agnostic to the lower level technology choices.

The time taken for the inference engine to reason over the knowledge base has been explored in section 5.3, where a time of 450 ms was observed following caching. As this time is sub-second, it is deemed satisfactory. The time taken to retrieve knowledge about the active alerts from the SPARQL endpoint was found to be 550 ms after caching, with an initial time of 3000 ms, and with memory consumption increasing from 113 MB to 800 MB after caching. Typically, at present, sensor readings are only reported up to as frequently as every 15 min (so as to preserve battery life) in water networks. Given this, the overall latency observed in the ICT solution is deemed satisfactory. Further testing could compare the scalability of technology choices for each component, against foreseeable scenarios of big data volumes and velocities for the water sector.

It is especially pertinent to discuss the suitability of the semantic approach in regards to the temporal component of the target use case, given the urgency of raising appropriate alerts, and the potential criticism of semantic technologies for their handling time. To this end, further testing could be conducted by increasing the telemetry reporting frequency near alert limits and observing the response time, to test the scalability of the approach with regards to big data velocity. However, the developed interface is primarily meant to provide decision support following alert discovery, where the semantic back-end provides discoverability and context to the time series data. It would be preferential for alert management and other low latency, mission critical software to subscribe to the telemetry devices via the Kafka pub-sub approach, where the semantic parts of the platform are used for higher-order analytics, optimisation, and decision support software. In this way the architecture supports reactivity by promoting interoperability through the ontology at design-time and low latency through the event bus at run-time.

## 7. Discussion and conclusion

This paper has proposed a knowledge management platform which aims to facilitate utility decision support software in a more powerful and accessible manner than existing solutions. A tool was presented which leveraged the benefits of the platform to provide higher-order and more integrated insight to decision makers for a

real pilot site in Wales, based on the use case of responding to a network fault. The platform supports the extension of existing web-based applications (Walker and Chapra, 2014) by furthering and applying the Internet of Things paradigm in smart water networks, through comprehensive semantic information provision and expert rule inference capabilities.

The main advantages of the proposed solution are i) the exposure of knowledge at varying levels of semantic richness, ii) scalability of the dynamic data storage, iii) automated discoverability of resources through standardised mechanisms, iv) industry-leading security, and v) robust grounding and extensibility through a domain ontology.

Firstly, the knowledge management platform exposes both timeseries data, and rich semantic web data. The former is achieved through a KairosDB server and accompanying API, provisioning for data-driven applications. The latter is achieved through an implementation of the Apache JENA libraries, which host a rigorous domain ontology and a site-specific knowledge bases. A mechanism was discussed which achieves the automated conversion of GIS and EPANET data to RDF/XML data, federated to the candidate domain ontology. This data is exposed as a standard SPARQL endpoint via a RESTful API. This is a critical aspect of the proposal, as the RDF data provides a rich contextualisation of the dynamic data in a coherent and machine-interpretable manner, enabling more powerful, confident, and quicker interoperability to be achieved with client applications, as well as extensibility. Ultimately, it would be beneficial for timeseries observation data and RDF data to be handled homogeneously, and this represents a long-term ambition of SWoT research. However, the proposed approach derives value from the combination of these data types alongside mature tooling, and could be extended to incorporate advances towards this integration agenda as they occur, whilst delivering value in the meantime. As well as serving data, a set of SWRL rules was curated from expert knowledge, to augment the inference capabilities of the system in line with a proposed use case of fault detection and impact inference. This rule base could be extended to include more use cases and more generic expert knowledge, but demonstrates the value of the approach in leveraging semi-structured or incomplete data towards a more complete description of the state of the network.

The dynamic data server was based on KairosDB, which is itself an implementation of Cassandra (KairosDB, n.d.). This is a leading edge big data storage solution for timeseries data, and offers the benefits of robustness, low response time, and seamless scalability. This contrasts the alternative options of hosting the dynamic data at a centralised RDF repository, or as a traditional MySQL database, or as a set of distributed linked data. Whilst the proposed solution performed well, further work could explore a quantitative comparison of these options at the data volumes and velocities likely in the foreseeable future in the water sector.

It was acknowledged that significant progress and value is evident in the Internet of Things field, and so recent promising work in the field was integrated with the knowledge-based approach to provide resource discoverability. The accessibility and simplicity of the Hypercat standard meant that this process was a small effort, and provided a valuable standardised means for discovering sensors on the target water networks, as well as key semantic information, and API endpoint and type descriptions. The Hypercat subscription service recommendations were not adopted, as the solution's previously chosen event-based architecture better supported the many components outside of the Hypercat ecosystem. An important aspect of future work will be to integrate the platform with the W3C Web of Things approach, once their suite of standards has been fully endorsed, although the presented work is likely to be complementary to these standards, and represents a useful contribution to the ongoing discourse.

The previously described APIs were governed by a robust security solution, which was implemented as a dedicated governance service, that controlled access to knowledge stored within the system using a role based access control approach. Additionally, security of information in transit was provided by encryption using TLS/SSL.

Finally, the proposed domain ontology represents a critical step for the smart water industry, as it integrates existing standards and a representative GIS schema with further smart water concepts, as well as the SSN ontology. The grounding in a comprehensive ontology, which includes significant abstraction, allows the system to be extensible in the event of new smart water concepts emerging, or adaptation to suit the needs of a specific stakeholder or use case. The ontological approach also supports the reuse of the system knowledge within other domains, and vice versa, such as if an application aimed to integrate a pumping station operation scheme with its local energy resources, or dynamic pricing tariffs.

A proof-of-concept decision support tool was presented, which aimed to demonstrate the benefits of the various functions of the platform. This provided a graphical interface for visualising the target network on a map, alongside dynamic sensor data, customer data, and alert descriptions, via a multiple level-of-detail approach. The tool extends traditional GIS systems and existing approaches used in industry, as well as integrating several typically siloed utility ICT systems. As well as this integration, the key extensions of the state of the art demonstrated were the inference capabilities, whereby the affected households could be more easily identified in the event of a network fault, and suggested actions could be presented to decision makers. Although the paper focuses on a specific use case to demonstrate the applicability of the proposed approach, the proposed water utility decision support is generic as evidenced by the business process diagrams, use cases, sequence diagrams, scenario descriptions, the various requirements, and competency questions that have informed the research as reflected by the scope of the underpinning ontology and its scalable heuristics delivered through expert rules.

Given the arguments presented, and in the context of the use case presented, the benefits and functions of the system presented would not have been possible with the standard rule systems evident in the industry (Servelec Technologies, n.d.), as they don't integrate the various types of knowledge (social, sensor, network) in the seamless manner that the ontology does. Further, the requirements outlined in section 3.1 describe a step change towards the capabilities of modern web applications, beyond currently evident ICT systems in the water industry and literature. Traditional knowledge management approaches fail to fulfill one or more of these requirements, hence justifying the value of the semantic web approach.

As the potential for smart water interventions is broad, the work needed a clear scope. As discussed in section 3.1, this scope manifested through a scenario-based approach to requirements engineering, and resulted in an understanding of desired functionality, out-of-scope statements, and an understanding of what the software could solve with little additional work. This also led to an understanding of the value proposition of the main novelty of the work: the integration of IoT (via Hypercat), semantic web, and inference technologies, which was validated through experimentation. Specifically, the approach is well suited to integrating data silos, providing discoverability and context to IoT devices for innovative and lightweight application development, and unifying the data schema and semantics used in a water domain enterprise. This supports the reuse of software, and hypothetically data, across organisations through standardisation. This has the benefit of accelerating progress in the field, reducing barriers to innovation

within an enterprise by reducing vendor tie-in, and allowing the development of more advanced knowledge-based analytics software for the domain. The main drawbacks of the approach would be the need to adapt the domain ontology developed, and instantiate this for each target system, as well as addressing the likely skills gap surrounding ontologies within a water utility aiming to adopt the approach. However, the semi-automated reuse of existing enterprise schemas and data mitigates the upfront investment in adopting an ontological approach, and the simplified API developed addressed part of the skills-gap problem, although this remains a largely open issue.

The work conducted serves as a proof of concept for this new approach, although more work would be required to explore the solution space further before building production-grade software based on the approach. Specifically, further work is required to fully understand the integration of a Semantic Web of Things approach alongside legacy systems, given the prohibitive expense of overhauling entire systems. Typical best practice is to implement an IoT layer above existing systems, in which case further work would be required to synchronise the SWoT layer with legacy GIS and telemetry layers, and potentially also hydraulic models. Further work is also required to build broader industry consensus around a standard ontology for the smart water domain, to fully explore the role of streaming RDF data, and to fully explore the suitability of inference and knowledge-based software within smart water use cases.

Overall, the proposed solution goes beyond existing research in the water or IoT fields, and represents a step-change towards a Semantic Web of Things for the water sector. The domain ontology, SWRL rules and inference engine provide comprehensive semantic context to big data from network sensors. Also, by exposing lower and higher-order knowledge and resource discoverability in a scalable and secure manner, the platform more powerfully supports interoperability for leading edge and next generation applications.

## Data availability

Information on the data underpinning the results presented here, including how to access them, can be found in the Cardiff University data catalogue at http://doi.org/10.17035/d.2018.0045386608.

## Acknowledgments

This work was supported by the European Commission under the FP7 WISDOM (grant number 619795) project, as well as the EPSRC and BRE.

## References

Allen, M., Preis, A., Iqbal, M., Whittle, A.J., 2013. Water Distribution System Monitoring and Decision Support Using a Wireless Sensor Network. IEEE, pp. 641–646. https://doi.org/10.1109/SNPD.2013.97.

ALMANAC, 2015a. D6.1 A Scalable Data Management Architecture for Smart City Environments. http://www.almanac-project.eu/downloads/deliverables/ (accessed 16.12.17).

ALMANAC, 2015b. D8.2 Application Definition - Water Management. http://www.almanac-project.eu/downloads/deliverables/ (accessed 16.12.17).

Cahn, Amir, 2013. Shaping the Architecture of Smart Water Networks. https://www.swan-forum.com/wp-content/uploads/sites/218/2016/03/water__sewerage_journal_shaping_the_architecture_of_smart_water_networks.pdf?x12236 (accessed 16.12.17).

Apache Jena - Jena Ontology API, n.d. https://jena.apache.org/documentation/ontology/ (accessed 2.24.15).

Babel, M.S., Islam, M.S., Das Gupta, A., 2009. Leakage management in a low-pressure water distribution network of Bangkok. Water Sci. Technol. Water Supply 9, 141. https://doi.org/10.2166/ws.2009.088.

Bellini, P., Benigni, M., Billero, R., Nesi, P., Rauch, N., 2014. Km4City ontology building vs data harvesting and cleaning for smart-city services. J. Vis. Lang. Comput. 25, 827–839. https://doi.org/10.1016/j.jvlc.2014.10.023.

BSI, 2014. Smart City Concept Model- Guide to Establishing a Model for Data Interoperability. https://shop.bsigroup.com/upload/268968/PAS%20182_bookmarked.pdf (accessed 2.24.15).

BSI, 2016. BSI PAS 212:2016 Automatic Resource Discovery for the Internet of Things Specification. https://shop.bsigroup.com/upload/276605/PAS212-corr.pdf (accessed 2.24.15).

Calbimonte, J.-P., Sarni, S., Eberle, J., Aberer, K., 2014. XGSN: an open-source semantic sensing middleware for the web of things. In: Joint Proceedings of the 6th International Workshop on the Foundations, Technologies and Applications of the Geospatial Web, TC, pp. 51–66. http://ceur-ws.org/Vol-1401/paper-04.pdf (accessed 2.24.15).

CityPulse, 2014. citypulse_d3.1:Semantic Data Stream Annotation for Automated Framework. http://www.ict-citypulse.eu/page/sites/default/files/citypulse_d3.1_v1.3.pdf (accessed 2.24.15).

Complexible, n.d. Complexible/pellet FAQ. https://github.com/Complexible/pellet/wiki/FAQ#different-results (accessed 5.19.16a).

Complexible, n.d. GitHub - Complexible/pellet. https://github.com/Complexible/pellet (accessed 5.19.16b).

Compton, M., Barnaghi, P., Bermudez, L., Garca-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C., Herzog, A., Huang, V., Janowicz, K., Kelsey, W.D., Le Phuoc, D., Lefort, L., Leggieri, M., Neuhaus, H., Nikolov, A., Page, K., Passant, A., Sheth, A., Taylor, K., 2012. The SSN ontology of the W3C semantic sensor network incubator group. Web Semant. Sci. Serv. Agents World Wide Web 17, 25–32. https://doi.org/10.1016/j.websem.2012.05.003.

Creaco, E., Lanfranchi, E., Chiesa, C., Fantozzi, M., Carrettini, C.A., Franchini, M., 2016. Optimisation of leakage and energy in the Abbiategrasso district. Civ. Eng. Environ. Syst. 33, 22–34. https://doi.org/10.1080/10286608.2015.1135136.

CTRL+SWAN, n.d. Cloud Technologies & ReaL time monitoring + Smart WAter Network. http://www.eip-water.eu/CTRL_SWAN (accessed 2.18.16).

CUAHSI, 2008. CUAHSI Hydrologic Information System - Hydrologic Ontology for Discovery. http://his.cuahsi.org/ontologyfiles.html (accessed 10.23.15).

Curry, E., Derguech, W., Hasan, S., Hassan, U.U., 2014. Waternomics D3.1.2-Linked Water Dataspace. http://waternomics.eu/wp-content/uploads/D3.1.2.-Linked-Water-Dataspace-Compressed.pdf (accessed 2.24.15).

Dam, K.H. van, 2009. Capturing Socio-technical Systems with Agent-based Modelling. Delft University of Technology, Delft, the Netherlands.

EIP Water, n.d. AugMent - Water Monitoring for Decision Support (AG124) — EIP Water. http://www.eip-water.eu/AugMent (accessed 2.18.16).

Gangemi, A. n.d. DOLCE+DnS Ultralite ontology. http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite (accessed 11.12.17).

Gómez-Pérez, A., Suárez-Figueroa, M.C.S., Villazón-Terrazas, B., 2008. NeOn methodology for building ontology networks: ontology specification. Methodology 118.

Google Developers, n.d. Google Maps JavaScript API. https://developers.google.com/maps/documentation/javascript/(accessed 9.11.16).

Gröger, G., Plümer, L., 2012. CityGML Interoperable semantic 3D city models. ISPRS J. Photogrammetry Remote Sens. 71, 12–33. https://doi.org/10.1016/j.isprsjprs.2012.04.004.

Gyrard, A., Bonnet, C., Boudaoud, K., 2014. Domain Knowledge Interoperability to Build the Semantic Web of Things. W3C Workshop on the Web of Things, Germany. https://www.w3.org/2014/02/wot/papers/gyrard-1.pdf (accessed 9.11.16).

Hauser, Andreas, Foret, Nicolas, Combellack, Stuart, Coome, Jonathan, Lopez, Quintilia, Hernandez, Elkin, Kharkar, Salil M., Rasekh, Amin, Koenig, Michal, Marcotorchino, Remy, Damour, Nicolas, 2016. Communication in Smart Water Networks. SWAN. https://www.swan-forum.com/wp-content/uploads/sites/218/2016/06/SWAN-White-Paper_Communication-Protocols.pdf?x12236 (accessed 16.12.17).

Horsburgh, J.S., Tarboton, D.G., Piasecki, M., Maidment, D.R., Zaslavsky, I., Valentine, D., Whitenack, T., 2009. An integrated system for publishing environmental observations data. Environ. Model. Software 24, 879–888. https://doi.org/10.1016/j.envsoft.2009.01.002.

Huang, M., Maidment, D.R., Tian, Y., 2011. Using SOA and RIAs for water data discovery and retrieval. Environ. Model. Software 26, 1309–1324. https://doi.org/10.1016/j.envsoft.2011.05.008.

IEEE Standards Committee, IEEE Standards Coordinating Committee 21 on Fuel Cells, P, Dispersed Generation and Energy Storage, Institute of Electrical and Electronics Engineers, IEEE-SA Standards Board, 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. Institute of Electrical and Electronics Engineers, New York, N.Y.

INSPIRE Thematic Working Group Utility and Government Services, 2013. D2.8.III.6 Data Specification on Utility and Government Services Technical Guidelines. European Commission Joint Research Centre. http://inspire.ec.europa.eu/id/document/tg/us (accessed 11.12.17).

ISO/IEC JTC1, 2014. Smart Cities Preliminary Report. https://www.iso.org/files/live/sites/isoorg/files/developing_standards/docs/en/smart_cities_report-jtc1.pdf (accessed 11.12.17).

Jara, A.J., Olivieri, A.C., Bocchi, Y., Jung, M., Kastner, W., Skarmeta, A.F., 2014. Semantic web of things: an analysis of the application semantics for the IoT moving towards the IoT convergence. Int. J. Web Grid Serv. 10, 244–272. https://doi.org/10.1504/IJWGS.2014.060260.

KairosDB, n.d. https://kairosdb.github.io/. (accessed 9.11.16).

Kolozali, S., Bermudez-Edo, M., Puschmann, D., Ganz, F., Barnaghi, P., 2014. A knowledge-based approach for real-time IoT data stream annotation and processing. In: Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing (CPSCom), IEEE. IEEE, pp. 215–222.

Lea, R., Blackstock, M., 2014. City hub: a cloud-based IoT platform for smart cities. In: IEEE 6th International Conference on Cloud Computing Technology and Science (CloudCom), pp. 799–804. https://doi.org/10.1109/CloudCom.2014.65.

Lee, S.W., Sarp, S., Jeon, D.J., Kim, J.H., 2015. Smart water grid: the future water management platform. Desalination Water Treatment. 55, 339–346. https://doi.org/10.1080/19443994.2014.917887.

Manyika, James, Woetzel, Jonathan, Dobbs, Richard, Chui, Michael, Bisson, Peter, Bughin, Jacques, Aharon, Dan, 2015. Unlocking the Potential of the Internet of Things. McKinsey & Company. https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world (accessed 11.12.17).

Mounce, S.R., Pedraza, C., Jackson, T., Linford, P., Boxall, J.B., 2015. Cloud based machine learning approaches for leakage assessment and management in smart water networks. Procedia. Eng. 119, 43–52. https://doi.org/10.1016/j.proeng.2015.08.851.

Mutchek, M., Williams, E., 2014. Moving towards sustainable and resilient smart water grids. Challenges 5 (1), 123–137. https://doi.org/10.3390/challe5010123.

Nemirovski, G., Nolle, A., Sicilia, Ballarini, I., Corado, V., 2013. Data integration driven ontology design, case study smart city. In: Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics. ACM. https://doi.org/10.1145/2479787.2479830.

Pfisterer, D., Rmer, K., Bimschas, D., Kleine, O., Mietz, R., Truong, C., Hasemann, H., Kröller, A., Pagel, M., Hauswirth, M., et al., 2011. SPITFIRE: toward a semantic web of things. IEEE Commun. Mag. 49, 40–48.

RDFlib, 2014. GitHub - RDFLib/rdflib: RDFLib Is a Python Library for Working with RDF, a Simple yet Powerful Language for Representing Information. https://github.com/RDFLib/rdflib (accessed 5.19.16).

RERUM, n.d. RERUM deliverable D2.3: System Architecture. https://bscw.ict-rerum.eu/pub/bscw.cgi/d18321/RERUM%20deliverable%20D2_3.pdf (accessed 17.12.17).

Reynolds, L., 2014. SWIM a Semantic Ontology for Interoperability. http://www.semanticwater.com/resources/Laurie-Reynolds-2014.pdf (accessed 17.12.17).

Robles, T., Alcarria, R., de Andrés, D.M., Navarro, M., Calero, R., Iglesias, S., López, M., 2015. An IoT based reference architecture for smart water management processes. Journal of Wireless Mobile Networks. Ubiquitous Comp. Dependable Appl. 6 (1), 423.

Sahlmann, K., Schwotzer, T., 2015. MOCAP: towards the semantic web of things. In: Posters and Demos at the International Conference on Semantic Systems, pp. 59–62. Vienna, Austria.

Schenk, C., 2010. A Systems-based Generic Decision Support System Application to Urban Water Management. Thse No 4663. Swiss Federal Institute of. Technology, Lausanne (EPFL), Switzerland. www.switchtraining.eu/switch-resources (accessed: 18.12.17).

Sensus, 2012. Water 20/20: Bringing Smart Water Networks into Focus. https://c.ymcdn.com/sites/www.ncsafewater.org/resource/collection/A0650A28-4C94-471B-B98E-B0DFD4F76C35/Water_T_AM_09.10_Walsby.pdf (accessed: 18.12.17).

Servelec Technologies, n.d. SCOPE Telemetry and SCADA: https://www.servelec-group.com/technologies/products-and-services/telemetry-scada/. (accessed 18.12.17).

Shu, Y., Liu, Q., Taylor, K., 2016. Semantic validation of environmental observations data. Environ. Model. Software 79, 10–21. https://doi.org/10.1016/j.envsoft.2016.01.004.

Sottara, D., Bragaglia, S., Mello, P., Pulcini, D., Luccarini, L., Giunchi, D., 2012. Ontologies, Rules, Workflow and Predictive Models: Knowledge Assets for an EDSS. International Environmental Modelling and Software Society (iEMSs).

Sottara, D., Correale, J.C., Spetebroot, T., Pulcini, D., Giunchi, D., Paolucci, F., Luccarini, L., 2014. An Ontology-based Approach for the Instrumentation, Control and Automation Infrastructure of a WWTP. International Environmental Modelling and Software Society (iEMSs), 7th Int. Congress on Env. Modelling and Software, San Diego, Ca, USA. http://www.iemss.org/sites/iemss2014/papers/iemss2014_submission_286.pdf (accessed 18.12.17).

Stewart, R.A., Willis, R., Giurco, D., Panuwatwanich, K., Capati, G., 2010. Web-based knowledge management system: linking smart metering to the future of urban water planning. Aust. Plan. 47, 66–74. https://doi.org/10.1080/07293681003767769.

SWEET, n.d. SWEET Overview. https://sweet.jpl.nasa.gov/. (accessed 10.23.15).

Thompson, K., Kadiyala, R., 2014. Leveraging big data to improve water system operations. Process Eng. 89, 467–472. https://doi.org/10.1016/j.proeng.2014.11.235.

Uceda-Sosa, R., Srivastava, B., Schloss, R.J., 2011. Building a highly consumable semantic model for smarter cities. In: AI for an Intelligent Planet Conference (Barcelona, Spain).

Vamvakeridou-Lyroudia, L., Mansfield, L., Moore, R.V., Hauser, A., Akhavan, R., Farnham, T., Laspidou, C., Pesquer, L., McCann, J., 2015. ICT4Water Cluster Report on Recommendations for Standards and Standardisation in the European SMART Water Market. http://www.i-widget.eu/images/pdf/CCWI_iWIDGET_ICT4Water_Standards_report_v2.pdf (accessed 11.12.17).

Vilches-Blázquez, L.M., Ramos, J.A., López-Pellicer, F.J., Corcho, O., Nogueras-Iso, J., 2009. Hydrontology: a Global Ontology of the Hydrographical Domain. http://mayor2.dia.fi.upm.es/oeg-upm/index.php/en/ontologies/107-hydrontology (accessed 10.23.15).

W3C, 2004. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. https://www.w3.org/Submission/SWRL/ (accessed 1.18.16).

W3C, 2017. Web of Things (WoT) Thing Description W3C First Public Working Draft 14 September 2017. https://www.w3.org/TR/2017/WD-wot-thing-description-20170914 (accessed 16.12.17).

Walker, J.D., Chapra, S.C., 2014. A client-side web application for interactive environmental simulation modeling. Environ. Model. Software 55, 49–60. https://doi.org/10.1016/j.envsoft.2014.01.023.

WatERP, 2013. D1.3 Generic Ontology for Water Supply Distribution Chain. http://waterp-fp7.eu/Downloads/deliverables/D1.1_Generic_taxonomy_for_water_supply_distribution_chain_v1.4.pdf (accessed 16.12.17).

Wong, B.P., Kerkez, B., 2016. Real-time environmental sensor data: an application to water quality using web services. Environ. Model. Software 84, 505–517. https://doi.org/10.1016/j.envsoft.2016.07.020.

Zhao, W., Beach, T., Rezgui, Y., 2016. Optimization of potable water distribution and wastewater collection networks: a systematic review and future research directions. IEEE Trans. Syst. Man Cybern. Syst 46, 659–681. https://doi.org/10.1109/TSMC.2015.2461188.