





RESEARCH ARTICLE

Semantic agent framework for automated flood assessment using dynamic knowledge graphs

Markus Hofmeister^{1,2,3} , Jiaru Bai¹, George Brownbridge³, Sebastian Mosbach^{1,2,3} , Kok F. Lee², Feroz Farazi¹, Michael Hillman³, Mehal Agarwal², Srishti Ganguly², Jethro Akroyd^{1,2,3}  and Markus Kraft^{1,2,3,4,5} 

¹Department of Chemical Engineering and Biotechnology, University of Cambridge, Cambridge, UK

²Cambridge Centre for Advanced Research and Education in Singapore (CARES), Singapore, Singapore

³CMCL Innovations, Cambridge, UK

⁴School of Chemistry, Chemical Engineering and Biotechnology, Nanyang Technological University, Singapore, Singapore

⁵The Alan Turing Institute, London, UK

Corresponding author: Markus Kraft; Email: mk306@cam.ac.uk

Received: 20 June 2023; **Revised:** 14 February 2024; **Accepted:** 16 April 2024

Keywords: derived information framework; digital twin; flood assessment; knowledge graph; ontology

Abstract

This article proposes a framework of linked software agents that continuously interact with an underlying knowledge graph to automatically assess the impacts of potential flooding events. It builds on the idea of connected digital twins based on the World Avatar dynamic knowledge graph to create a semantically rich asset of data, knowledge, and computational capabilities accessible to humans, applications, and artificial intelligence. We develop three new ontologies to describe and link environmental measurements and their respective reporting stations, flood events, and their potential impact on population and built infrastructure as well as the built environment of a city itself. These coupled ontologies are deployed to dynamically instantiate near real-time data from multiple fragmented sources into the World Avatar. Sequences of autonomous agents connected via the derived information framework automatically assess consequences of newly instantiated data, such as newly raised flood warnings, and cascade respective updates through the graph to ensure up-to-date insights into the number of people and building stock value at risk. Although we showcase the strength of this technology in the context of flooding, our findings suggest that this system-of-systems approach is a promising solution to build holistic digital twins for various other contexts and use cases to support truly interoperable and smart cities.

Impact Statement

Given an ever-growing abundance of smart city data, streamlined and computer-aided data handling is essential for evidence-based and timely decision-making. This holds even more true during crisis situations and disaster management to ensure the safety and well-being of individuals and protect city infrastructure. We propose an ecosystem of connected autonomous software tools that continuously interact with an underlying knowledge graph to combine various flood-related information and automate the impact assessment for potential flood hazards. Our system dynamically ingests and connects real-time data from multiple previously isolated sources and automatically determines the number of people and buildings as well as the total building stock value at risk of flooding.

1. Introduction

Cities have an incomparable amount of urban data that can boost innovation and provide decision support from strategic planning to day-to-day operations. Despite the abundance of data, it often remains fragmented in silos. Numerous smart city applications have emerged to leverage those ever-increasing amounts of data; however, most of these solutions either focus on individual domains or provide tailored platform solutions to combine datasets for specific use cases using proprietary data models. On the contrary, FAIR data principles (Wilkinson et al., 2016) promote open findability, accessibility, interoperability, and reusability of available data to maximize the value of individual pieces of information, foster collaboration, and promote more holistic perspectives. Connected and enriched data can be seen as a strategic asset, providing valuable insights, allowing timely and evidence-based decision-making, and enabling myriad automation opportunities. Real interoperability, however, requires not just merging data from different domains, sources, and temporal dimensions, but creating and instantiating the underlying knowledge models, allowing for comprehensive assessments of cross-domain effects.

Ontologies play a crucial role in enabling intelligent systems to comprehend both the conceptual and causal aspects of real-world phenomena (Sermet and Demir, 2019), thereby facilitating effective knowledge inference from ever-increasing sensor data in the environmental and urban domain. The use of semantics in smart city modeling enables the discovery and analysis of data based on spatial, temporal, and thematic context (Wang et al., 2020) and enhances both the quantity and quality of information available from large-scale sensor networks. While numerous ontologies exist to help disambiguate heterogeneous urban information (Wang et al., 2020; Mughal et al., 2021), most of the available ontologies focus on conceptualizing static domains instead of representing actual data. Depending on their scope, they provide more or less detailed representations of certain smart city aspects, but are hardly linked or utilized to instantiate actual live data feeds; however, exactly this dynamism would be required to create truly interoperable smart city ecosystems which remain current in time.

Floods are among the most devastating natural disasters, causing extensive damage to people, properties, and the environment (Pour et al., 2020; Mughal et al., 2021). Managing flood risk effectively requires an accurate and timely assessment of potential impacts on human lives, infrastructure, and the economy (Scheuer et al., 2013). Hence, identifying and extracting relevant pieces of information as well as making accurate inferences from various data sources rapidly is critical. Although several publicly available flood assessment tools support some consolidated insights, all of them remain limited to individual domains. For example, a live flood map for the UK (Met Office and Environment Agency, 2022) shows current flood warnings and alerts, together with current readings for river, sea, groundwater, and rainfall levels, and the expected flood risk over the next 5 days; however, no indication of potential impacts in terms of people and built infrastructure at risk is provided. Nevertheless, such cross-domain awareness has been shown to have a positive impact throughout all phases of the disaster management cycle (Oktari et al., 2020), especially as floods are very complex phenomena involving a large number of stakeholders and domain experts to collaborate seamlessly (Wrachien et al., 2012). Ontology-driven systems have been proposed to create a universal understanding across the different stakeholders to enable semantic interoperability, flexibility, and reasoning support (Sinha and Dutta, 2020).

Knowledge graph technology can be used to instantiate ontologies and connect data about various aspects of urban environments into a network of entities and their relationships. The use of knowledge graphs has gained traction as a vital technology for offering machine-interpretable, semantic information about real-world entities on a large scale. For example, the recently developed geographic knowledge graph WorldKG (Dsouza et al., 2021) offers an ontological instantiation of geographic entities in the OpenStreetMap dataset to promote the use of semantic geographic knowledge across various real-world applications, such as event-centric and geospatial question answering as well as geographic information retrieval. Although this work significantly enhances previous efforts, such as LinkedGeoData (Stadler et al., 2012) or YAGO2geo (Karalis et al., 2019), which solely focused on instantiating entities instead of also providing respective class definitions, the proposed ontology seems too light-weight to represent comprehensive connections between related entities. Johnson et al. (2022) have presented a scalable

workflow for merging multiple geospatial datasets to create a comprehensive knowledge graph of urban infrastructure data. Furthermore, machine learning models are applied to the graph to infer and populate missing data in order to ensure the availability of important information for emergency responders during flood events. Buildings and demographics at risk of flooding can be queried; however, no dynamic data assimilation is supported and a new graph needs to be created on demand for each new analysis, lacking continuous insights into real-world situations.

The World Avatar dynamic knowledge graph is designed as an extensible semantic system to foster interoperability and effectively address cross-domain questions (Akroyd et al., 2021). It combines ontologies (i.e., data definitions) with actual data instances (i.e., from [open] APIs), and computational services operating on the instantiated data (i.e., so-called agents). Autonomous computational agents accomplish tasks such as updating the knowledge graph, simulating systems, or transmitting responses to the physical world. Based on Semantic Web technologies, the World Avatar intends to overcome the limitations of previous platform approaches, such as cumbersome data ingestion pipelines or potential lock-in effects. The effectiveness of this approach has been demonstrated in its ability to create ecosystems of connected digital twins that provide real-time information about the world's state, intelligently control complex systems, or support system design through elaborate what-if analyses. The World Avatar constitutes a versatile system to conduct geospatial (Akroyd et al., 2022), temporal (Savage et al., 2022), as well as cross-domain scenario analyses (Eibeck et al., 2020). The derived information framework (Bai et al., 2024) can be deployed to track data provenance within the knowledge graph and ensures that newly instantiated data automatically traverses through the graph, including changes to all dependent information when required.

The *purpose of this article* is to address the identified lack of dynamism and cross-domain interoperability when assessing potential flooding events. Three coupled ontologies are developed as foundational knowledge models to dynamically assimilate and connect real-world data feeds related to flooding. A sequence of connected autonomous software agents continuously monitors the knowledge graph to re-evaluate the likely impacts of imminent floods with regards to people and buildings at risk whenever relevant inputs become updated. The deployment of a knowledge graph native solution to track data provenance and dependencies enables the automatic cascading of information to maintain an up-to-date worldview evolving in time. The system is hosted in the cloud using a containerized implementation approach to ensure collaborative deployment together with a unified visualization interface.

The structure of this article is as follows: [Section 2](#) summarizes previous technical works and provides a review of relevant ontologies as well as used data sources; [Section 3](#) introduces the target use case and details both the ontology and agent development to create a dynamic knowledge graph-based digital twin to address it; [Section 4](#) describes the deployment of the developed system and highlights its results; and [Section 5](#) concludes the work.

2. Background

The Semantic Web (Berners-Lee et al., 2001) is an extension of the World Wide Web with the aim of making web content machine-readable and interoperable by adding structured metadata. It involves the use of ontologies and the Resource Description Framework (RDF) (Klyne and Carroll, 2004) as standard for representing such metadata. The Semantic Web aims to enable a “Web of Data” that can be easily understood and processed by machines, thus facilitating the development of more sophisticated and intelligent web-based and automated applications.

2.1. Ontologies and knowledge graphs

An ontology provides a conceptual description of a certain domain of interest. It describes relevant *concepts* (also known as classes), relationships between these concepts (referred to as *properties*), and restrictions and rules describing these relations explicitly. Properties are relationships whose domain and range are defined in terms of concepts, where *object properties* connect an instance of one class (the

domain of the property) to another instance of a class (the range of the property) and *data properties* link an instance of a class to an actual data element, a so-called Literal. A so-called Terminological Component (TBox) defines the classes and properties that can exist within the ontology, while the Assertion Component (ABox) contains specific instances of those classes as well as their object and data properties. Ontologies can be represented in various forms, such as formal representations using Description Logic (Baader et al., 2007) or the widely accepted Web Ontology Language (OWL) (W3C, 2012), which is currently considered the standard by the World Wide Web Consortium (W3C) (Allemang and Hendler, 2011).

Ontologies can be designed in a modular fashion: upper ontologies (also known as top-level ontologies) capture general knowledge about a certain field by providing basic notions and concepts, and domain ontologies refine these notions to include more detailed knowledge valid for a particular application or domain (Sinha and Dutta, 2020). Users of a common ontology commit themselves to ask queries and make assertions in a way that is consistent, but not complete, with respect to the knowledge model specified by the ontology (i.e., supporting the open world assumption). Hence, ontologies ease knowledge sharing and reuse without sharing actual (and potentially proprietary) data and enable knowledge to be machine-processable for better information retrieval (Sinha and Dutta, 2020). Strict formalization allows for additional automation as well as knowledge discovery, reasoning, or the inference of new and implicit information. Reasoners such as HerMiT (Data and Knowledge Group, 2019) can be used to check the consistency of an ontology and to deduce indirect subclass relationships. Ontology-Based Data Access (OBDA) (Botoeva et al., 2016) using tools like Ontop (Xiao et al., 2020) enables access to data from a variety of structured sources (e.g., relational databases) using ontologies, allowing for more efficient and effective data integration and management.

Representing data using ontologies results in the formation of directed graphs, so-called knowledge graphs (KGs), where nodes define concepts, instances, or data, and edges denote their relationships (i.e., properties). KGs provide an extensible data structure that is well suited to represent arbitrarily structured data and which can be hosted decentralized (i.e., distributed over the internet) using Semantic Web technology in the form of Linked Data (Berners-Lee, 2006; Bizer et al., 2011). As KG resources can be uniquely identified via Internationalized Resource Identifiers (IRIs), Semantic Web data can unambiguously be linked to one another across the web to allow for the identification of related information as well as the creation of a collaborative knowledge graph, where every concept and relation can be referenced back to its original definition. Linked Data fosters FAIR data principles (Wilkinson et al., 2016), improves machine readability, enhances clarity by identifying and dissolving inconsistencies, provides additional context information, and increases discoverability of information across isolated data sources.

Knowledge graphs can be stored in graph databases, also known as triple stores, such as RDF4J or Blazegraph (Blazegraph, 2020a). Graph databases are designed to host RDF data (and thus KGs) in the form of subject-predicate-object triples and can be queried and updated using SPARQL (Aranda et al., 2013), a query language designed to interact with semantic information.

2.2. *The World Avatar*

The World Avatar (TWA) project intends to create an all-encompassing model of our world, with a current emphasis on automation and decarbonization in chemistry (Farazi et al., 2020; Bai et al., 2022; Kondinski et al., 2023), process and energy industry (Devanand et al., 2020; Atherton et al., 2021), and smart cities and city planning (Chadzynski et al., 2022, 2023a, 2023b). TWA aims to provide a technology agnostic and scalable architecture based on open standards and protocols to create a collaborative knowledge-model based system to foster interoperability along three themes (Akroyd et al., 2021): (1) providing cross-domain insights into the current state of physical assets in the real world, (2) controlling real-world entities, and (3) facilitating complex what-if scenario analyses. TWA builds on Semantic Web technologies and is implemented as dynamic knowledge graph which can be distributed across the web, combining the ontological descriptions of relevant concepts and instances with semantically annotated

computational agents to operate upon them. The combination of ontological descriptions, instantiated data, and automated agents makes TWA a powerful, extensible, and FAIR-compliant system for representing and reasoning about complex domains of knowledge. The overall design idea is depicted in Figure 1.

As a dynamic KG, computational agents are an integral part of TWA and provide versatility and dynamism to the overall system. Similar to a micro-service architecture, TWA follows a system-of-systems approach, trying to avoid large monolithic models and breaking them down into sequences of individual agents with tailored capabilities, such as ingesting real-world data, interacting (autonomously) with instantiated information, restructuring the KG, or assembling new composite agents to accomplish more complex tasks (Zhou et al., 2019). The design of TWA as dynamic KG has been demonstrated as a suitable approach to implement a comprehensive ecosystem of connected digital twins to describe the behavior of complex systems (Akroyd et al., 2021).

2.3. Derived information framework

The derived information framework (DIF) provides a knowledge graph native solution to track data provenance and dependencies within a dynamic KG (Bai et al., 2024) by semantically annotating how a specific piece of information can or has been derived from other data instantiated within the KG. Another feature provided by the framework is its ability to automatically detect outdated agent outputs in order to trigger a subsequent re-evaluation when necessary. The underlying idea is that all active agents share the same worldview and can be chained together by using the same instance IRIs: by declaring the output instances of one agent operation as input instances of another agent operation, entire cascades of information updates can be executed automatically. This design removes the need for direct agent-to-agent communication and allows for a distributed ecosystem of agents solely connected via their input/output resources within the KG. A short introduction is provided below and illustrated in Figure 2, while the interested reader is referred to Bai et al. (2024) for more details.

The term *derivation* denotes the fact that a specific piece of information (i.e., a specific instance) has been obtained or computed from other instantiated information. OntoDerivation (Bai et al., 2024) has been proposed as light-weight ontological markup to denote such dependencies between related instances

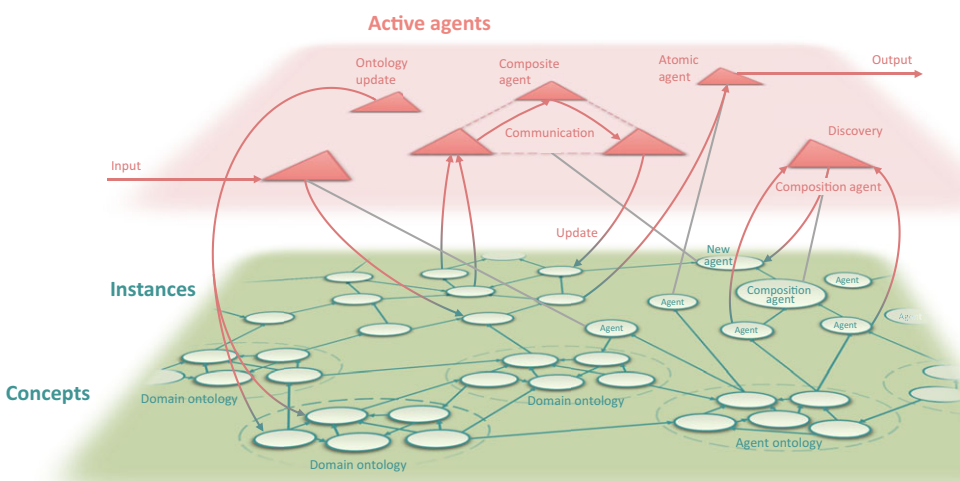


Figure 1. The design of the World Avatar dynamic knowledge graph. The World Avatar consists of three principal components: (1) ontological description of relevant domains (i.e., concepts), (2) actual data instantiated based on those ontologies (i.e., instances), and (3) automatable computational agents to operate on the knowledge graph. Image reproduced from Akroyd et al. (2021) under a CC BY 4.0 license.

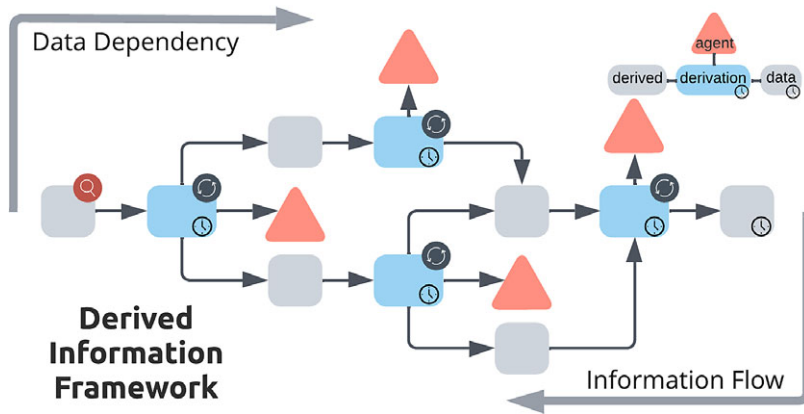


Figure 2. Chain of derivations. The derived information framework facilitates automatic information cascading in a dynamic KG by capturing dependencies at the instance level, including details about associated agents: Which agent is responsible for computing a specific instance? Which inputs are used in deriving that output? And is that output instance itself potentially input to another derivation? Image reproduced from Bai et al. (2024) under a CC BY 4.0 license.

explicitly and re-uses OntoAgent (Zhou et al., 2019) to specify the agent instance responsible for a specific derivation task. While OntoAgent describes the computational capabilities of agents conceptually, OntoDerivation concentrates on the instance level, that is, connects a specific agent instance with the actual input and output instances processed and generated by it, respectively. Upon initialization, each derivation as well as all its pure inputs are marked with a timestamp according to the W3C standard (Cox et al., 2020). Throughout the lifespan of a derivation, these timestamps determine whether a derivation is still up to date or considered outdated. Whenever a derivation output is requested within the KG, the framework compares the timestamp of the derivation instance with the timestamps of its inputs. In case any of the inputs have been amended after the last derivation calculation (i.e., has a more recent timestamp than the derivation instance), an update of the output instance is conducted before returning the requested information.

The framework differentiates between two types of derivations, namely *synchronous* and *asynchronous* derivations, to cater for different response times upon receipt of a query: The *synchronous* mode utilizes the agent's HTTP endpoint for communication, making it faster and suitable for applications that require immediate responses. Conversely, *asynchronous* mode communicates exclusively through the KG (i.e., using specific status markup to track the progress of a derivation computation) to suit slower and computationally more expensive agent operations. Regardless of the communication protocol, all derivations are instantiated consistently: Outputs (i.e., representing derived information) are connected to their respective derivation instance via a `belongsTo` relationship. Inputs (i.e., a set of source information instances) are related to the derivation instances via an `isDerivedFrom` relationship. The connection between a derivation instance and the respective agent is denoted with an `isDerivedUsing` relationship. Although there is no limit to the number of inputs or outputs of a single derivation, any output instance cannot be associated with more than one derivation instance. Chains of derivations can be formed if an instance `belongsTo` to a certain derivation (i.e., as one of its output entities), but also has an `isDerivedFrom` relationship with another derivation instance (i.e., representing one of its input entities). The framework supports derivation dependencies from basic linear chain to non-linear polytree to generic directed acyclic graph.

2.4. Available public data sources

A comprehensive search for available open data sources and existing ontologies concerning built infrastructure as well as environmental measurement and flood-related data has been conducted, with

primary focus on the UK. Relevant static sources and application programming interfaces (APIs) are summarized below and the interested reader is referred to [Supplementary Material SI.5](#) as well as Hofmeister et al. (2023) for more details.

2.4.1. Environmental observations data

The Met Office DataPoint API provides open access to both weather observations and forecasts (e.g., temperature, wind speed and direction, humidity, etc.) for thousands of stations across the UK (Met Office, 2022). While forecasts cover a 5-day period, actual observation values are provided for the previous 24 hours. Similarly, the UK Air Information Resource (UK-AIR) provides real-time air quality data for various pollutants via a machine-readable Sensor Observation Service (Department for Environment, Food, and Rural Affairs, 2023), including nitrogen dioxide, particulate matter, and ozone. The Environment Agency (EA) provides several API endpoints with (near) real-time information related to flooding and flood risk: The Real Time flood-monitoring API (Environment Agency, 2021b) provides a continuously updated list of current flood alerts and warnings, together with applicable flood areas, as well as live readings of water levels and flows recorded at various measuring stations along rivers and other water bodies. Furthermore, precipitation and hydrological data about river levels, river flows, groundwater levels, and water quality are provided via EA's Hydrology API (Environment Agency, 2021a). As the official government agency responsible for environmental protection and regulation, EA operates its own UK-wide monitoring system to collect river level, hydrological, and rainfall data. With regards to flood risk management, EA deploys a widely recognized early warning dissemination system as part of its flood risk reduction and adaption strategy, including proactive communication to the wider public (Pathak and Eastaff, 2014). The agency employs domain experts specialized in hydrology and flood risk assessment to collect and disseminate accurate information, and, in collaboration with the Met Office, forms the Flood Forecasting Centre which produces the best-combined understanding of flood risk based on weather forecasts, flood forecasts, catchment conditions, and the operational status of flood defenses (Flood Forecasting Centre, 2017). They generate a daily flood risk forecast, assessing the likelihood of flooding 5 days into the future.

2.4.2. Flood warnings data

The EA Real Time flood-monitoring API (Environment Agency, 2021b) provides a listing of all current flood alerts and warnings with applicable flood areas as well as further meta information (e.g., severity and associated water bodies) and is updated every 15 minutes. Flood alerts and warnings refer to different warning stages: *flood alerts* express that flooding is possible and it is important to stay alert and vigilant, while *flood warnings* refer to situations when flooding is expected and immediate action is required to protect people and properties. Since 1996, the Environment Agency has been designated as the lead authority for issuing accurate, reliable, and timely flood warnings to both emergency responders and the wider public in the UK (Pathak and Eastaff, 2014). The agency's system offers an end-to-end approach, including live data acquisition, data quality control, post-event data collection, reporting, and archiving. EA's early warning strategy integrates traditional methods, like using measured water levels as triggers, with advanced modeling and forecasting, including relatively simple techniques like correlating peak levels/flows across sites as well as linked rainfall-runoff models with hydrological channel routing. Probabilistic and deterministic weather forecasts from the Met Office automatically feed into flood forecasting and warning processes, while measured rainfall intensities are used for real-time validation purposes (Pathak and Eastaff, 2014). The performance of flood forecast models is evaluated regularly and continuous model recalibration cycles based on validation datasets from a series of past flood events ensure current, meaningful, and accurate flood forecasts. While the exact methodology remains undisclosed, in part due to the exclusive access to more detailed data granted to emergency responders, the public warnings data provided is highly trusted by authorities and public services (Met Office and Environment Agency, 2022) as well as academia, both as data source (Barker and Macleod, 2019; Wolf et al., 2022) and benchmark (Smith et al., 2009). As the focus of our work lies on the methodology of

integrating flood-related data from various domains and sources, the interested reader is referred to Pathak and Eastaff (2014) for more details about models as well as validation and data workflows deployed by the Environment Agency.

2.4.3. Building data

The Ordnance Survey (OS), as the national mapping agency of Great Britain, provides several open (e.g., OpenMap Local) and premium (i.e., Building Height Attribute) datasets describing the physical characteristics of the built environment in various levels of detail (Ordnance Survey, 2022). While the Building Height Attribute (BHA) data represents the most granular data about individual buildings, including their base polygon, building height, and ground elevation, the OpenMap Local contains building data on a more aggregated level and lacks information about building heights. Premium datasets are generally license-restricted; however, made available via Digimap (EDINA Digimap Ordnance Survey Service, 2022) for educational and research purposes. The Unique Property Reference Number (UPRN), which constitutes a unique and officially maintained identifier assigned to every addressable location in the UK, can be used to cross-links building information across datasets. The Department for Levelling Up, Housing & Communities offers open Energy Performance Certificate (EPC) data (Department for Levelling Up, Housing, and Communities, 2022) via three dedicated APIs for domestic, non-domestic, and display (i.e., mostly public buildings) certificates. This data contains property-level information about energy efficiency, key construction characteristics (i.e., number of rooms, total floor area, building type, etc.), high-level usage classification as well as address and location details, and is updated every 4–6 months. His Majesty's Land Registry publishes several public datasets related to residential property sales on a monthly basis: The UK House Price Index (UKHPI) (HM Land Registry, 2022d) captures the monthly change in the value of residential properties on different levels of geospatial granularity. And the Price Paid Data (PPD) (HM Land Registry, 2022b) contains information about actual prices and dates of residential property sales, including address and property type.

2.4.4. Population data

The OpenPopGrid (Murdock et al., 2015) provides an open-gridded population dataset for England and Wales based on the Office for National Statistics (ONS) 2011 Census as well as OS OpenData. It aims to enhance the spatial accuracy of the ONS population dataset by redistributing the population to actual residential areas.

2.5. Existing domain ontologies

Knowledge model-based systems require structured and curated data to be represented in the form of ontologies. The following section introduces several relevant existing ontologies together with their limitations, with further details provided in [Supplementary Material SI.6](#).

2.5.1. Sensor and measurement ontologies

Several sensor ontologies have been proposed in the literature, each with its own design principles, coverage, and target applications: The Sensor, Observation, Sample, and Actuator (SOSA) ontology (Janowicz et al., 2019) provides a light-weight, modular, and self-contained core ontology for describing basic concepts related to sensors, observations, and actuators. While its simplicity compared to other ontologies fosters re-use, its limited coverage is not suitable for all sensor-related applications. The Semantic Sensor Network (SSN) ontology (World Wide Web Consortium, 2017) extends SOSA with more specialized and domain-specific concepts to provide a more expressive ontology to describe sensors and their observations, samples, and procedures used, observed properties, and actuators. While its comprehensive coverage of standardization is one of the key advantages, some aspects of the ontology may be overly complex for certain use cases. The Modular Environmental Monitoring (MEMOn) ontology (Masmoudi et al., 2020) defines a set of concepts and relationships for describing environmental

monitoring equipment, including sensors, together with clear guidelines for mapping sensor data to the ontology. While MEMOn's domain specificity ensures an accurate representation of environmental monitoring data, it may be overly specific for various applications. The Smart Appliances REference (SAREF) ontology (ETSI, 2020b) aims to provide a standardized framework for representing smart appliances and associated data and is designed in a modular fashion. Compared to SSN and SOSA, SAREF has a narrower scope, focusing specifically on smart appliances rather than sensors and observations more broadly; however, it can easily be extended.

Several ontologies have been proposed to describe environmental water resources and associated sensor readings; however, the focus has mainly been on either aligning heterogeneous data from various sensor web resources (Liu et al., 2012; Dutta and Morshed, 2013; Wang et al., 2020) or supporting water quality monitoring (Xiaomin et al., 2016). A hydrological sensor web ontology-based on the SSN ontology has been developed to align semantics and support collaboration between individual water-related sensor networks (Wang et al., 2018), primarily in response to natural disasters such as floods. It introduces hydrological domain classes and establishes relevant reasoning rules.

2.5.2. Flood ontologies

Various ontologies have been proposed for natural disaster management in general (Klien et al., 2006; Kollarits et al., 2009) as well as flooding in particular (Wrachien et al., 2012; Agresta et al., 2014; Khantong et al., 2020; Mughal et al., 2021). Recent systematic reviews of existing flood ontologies confirm the increasing usage of ontological approaches for flood knowledge and disaster management (Sinha and Dutta, 2020; Mughal et al., 2021). Ontologies have been developed for different aspects and types of flood disasters, including urban flooding, flood risk and environmental assessment, and stakeholder and response management. The majority of the 14 flood ontologies reviewed by Sinha and Dutta (2020) are formal application ontologies built around small tasks performed during the response phase of flood disasters (Sinha and Dutta, 2020). Among these, very few intend to conceptualize the broader domain knowledge of flood disasters, but are mostly scenario and/or task-specific. The number of classes varies significantly between 4 and 410; however, most of the ontologies are modular to simplify the complexity of conceptualizing large-scale spatiotemporal systems, such as floods (Sinha and Dutta, 2020; Mughal et al., 2021). The study reveals that there is no standard flood ontology available in the field. The most commonly re-used ontologies comprise the Semantic Web for Earth and Environmental Terminology (SWEET) ontologies (Buttigieg et al., 2018) and the MONITOR risk management ontology (Kollarits et al., 2009).

SWEET ontologies define a hierarchy of many flood risk-related terms and are often referenced for a variety of environmental concepts (Buttigieg et al., 2018). The Environmental Ontology (ENVO) is a domain ontology describing various environment types across several levels of granularity. Its *environmental hazard* module contains suitable concepts to represent a single flood and flooding in general, including more detailed descriptions of typical kinds of flooding; however, no concepts to describe the social or economic damage of floods are considered. Several domain ontologies have been proposed to resolve ambiguity in information exchange and enable seamless collaboration between various stakeholders during flood disaster management (Khantong et al., 2020; Mughal et al., 2021). The focus of these ontologies is mainly on conceptualizing the structure and sequence of relevant information, involved organizations, roles as well as the interplay between them rather than representing the actual data itself. Kollarits et al. (2009) have developed MONITOR as formalized knowledge model to describe the relations between natural, social, and built environments, potentially hazardous events, and several risk assessment and management terms. The proposed risk model has been refined by Scheuer et al. (2013) proposing a tailored ontology for multi-criteria flood risk assessment.

Although most of the proposed ontologies are claimed to be available in OWL, only ENVO could be found in coded form online. This matches previous findings by Sinha and Dutta (2020), which significantly hinders the re-usability of previous efforts and partially undermines the fundamental idea of common ontologies to enable machine-interpretability.

2.5.3. Building ontologies

OntoCityGML has been proposed as a comprehensive ontology (Chadzynski et al., 2021) for three-dimensional geometrical city objects based on the CityGML 2.0 standard (Gröger et al., 2012). CityGML is an open data model and XML-based format developed by OGC to describe urban environments by providing a common definition of the basic entities, attributes, and relations within a 3D city model. OntoCityGML provides a multi-scale model with five consecutive Levels of Detail (LoD), where the geometric representation of any building is successively refined from LoD0 to LoD4. A single building can have multiple spatial representations in different LoDs at the same time, for example, a simple 2D footprint polygon as LoD0 and a set of 3D surfaces confining the building volume as LoD1 representation. OntoCityGML primarily focuses on the geospatial representation of buildings to support spatial analyses and city planning. Further information about a building (e.g., building usage, year of construction, energy rating) can be encoded using predefined code lists or so-called `genericAttributes`. Although this approach provides further information about a building, it does not fulfill the requirements for semantically Linked Data.

A review of 40 metadata schemas for different phases of the building life cycle is provided by Pritoni et al. (2021). Five popular ontologies/schemas have been scrutinized in detail with regards to suitability and potential gaps in building modeling, namely the Brick Schema (2022), the BOT ontology (Rasmussen et al., 2021), the RealEstateCore ontology (RealEstateCore Consortium, 2020), SAREF including extensions (ETSI, 2020a) as well as SSN/SOSA (World Wide Web Consortium, 2017; Janowicz et al., 2019). The ontologies differ in perspective of how they represent building information, from focus on topologies of buildings and their sub-components (BOT) to sensors (SSN/SOSA) or devices (SAREF) and assets (Brick Schema). General-purpose ontologies such as BOT, SSN/SOSA, and the core of SAREF tend to leave gaps that require a modeler to supplement them with extensions or external schemas. Conversely, application ontologies like Brick, RealEstateCore, and SAREF4BLDG may not be as broadly applicable but offer domain-specific features. While the Brick Schema seems most suited for large, complex building automation, the Haystack ontology suits smaller-scale systems. Key ontologies are briefly discussed in [Supplementary Material SI.6.3](#), and the interested reader is referred to Pritoni et al. (2021) for more details.

Despite the fact that the proposed ontologies differ in how exactly they represent building sub-components and their relationships, most of them are extensible and compatible with one another. Further perspectives on buildings are provided by specific domain ontologies, such as the iCity Building ontology (Katsumi, 2021) providing a foundational vocabulary to represent building-related data within the urban context or the Domain Analysis-Based Global Energy Ontology (DABGEO) (Cuenca et al., 2020) fostering interoperability across the energy domain, with a focus on smart home and smart city energy management (Cuenca et al., 2020). The DABGEO ontology (Cuenca and Larrinaga, 2019) contains a tailored sub-module for knowledge about infrastructure and buildings, containing classes, properties, and axioms to represent static building features (e.g., surface, material), geometrical details (rooms, floors) as well as internal and external environmental conditions (e.g., room temperature).

2.6. Representation of geospatial and time series data

Common graph databases are optimized to handle large-scale RDF data and perform complex queries efficiently; however, native support for geospatial capabilities is limited in most off-the-shelf triple stores (Akroyd et al., 2022). Furthermore, the representation of large quantities of time series data where each data point is instantiated as an individual triple with full semantic markup can pose performance issues due to the vast amount of (partially redundant) statements.

Numerous guidelines for encoding geospatial data in RDF have been published, including the GeoSPARQL (OGC, 2012) as well as CityGML (OGC, 2021) standards, both created by the OGC. GeoSPARQL forms the de-facto standard for representing and querying geospatial data on the Semantic Web, including an extension to the SPARQL query language for processing geospatial data. The level of support for GeoSPARQL varies among different triple stores, but remains limited and inconsistent

(Chadzynski et al., 2021; Jovanovik et al., 2021): While RDF4J, for example, provides “partial GeoSPARQL support” (Eclipse Foundation, 2021), Blazegraph (which is used in this study) does not support GeoSPARQL at all and instead offers a custom subsystem for simple geospatial queries (Blazegraph, 2020b).

3. Methodology development

3.1. Use case

The first key motivation behind this work is to instantiate a rich knowledge basis of built environment as well as transient and live environmental sensor data for a set of autonomous agents to subsequently act upon. Connecting previously isolated data sources enables both humans and software agents to make better evidence-based decisions, especially if the various data sources provide complementing perspectives on related domains, for example, bringing together data about potential flood events (i.e., severity, areal extent) with data about buildings (i.e., building locations, building usages, property values). This requires the development of domain ontologies to represent relevant building, flood, and environmental observation information as well as multiple input agents to continuously ingest data feeds from physical entities in the real world.

Secondly, the agent-enabled autonomous cascading of information through the KG shall be demonstrated with an automatic re-assessment of potential flood impacts using the DIF (Bai et al., 2024): Current flood alerts and warnings are steadily instantiated as part of creating the underlying knowledge basis. Each alert or warning pertains to a specific geographical extent, posing potential risk to both people and built infrastructure within that area. The proposed agent framework ensures that whenever a new flood alert or warning gets instantiated or already instantiated information gets updated, the potential impact with regards to the number of people and buildings as well as the estimated building stock value at risk gets re-assessed.

The use case is implemented for a mid-size coastal town in the UK, King’s Lynn, and schematically illustrated in Figure 3. While this work focuses on the technical details and implementation, gained insights and the added value for flood risk management are discussed elsewhere (Hofmeister et al., 2023) in more detail.

3.2. Ontology development

Typically, ontology development is not a goal in itself, but follows specific usage objectives, such as semantic search, reasoning, or allowing cross-domain interoperability. Competency questions are often

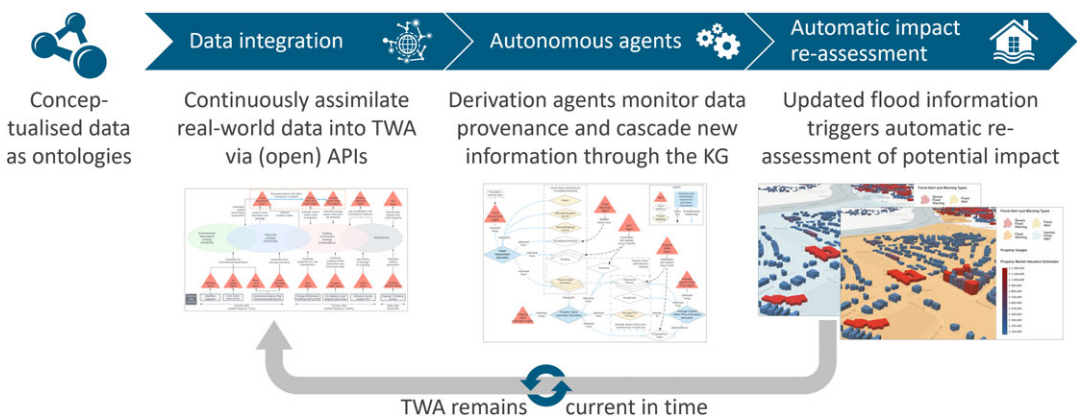


Figure 3. Schematic of use case implementation. For details see Figures 10, 17, and 20, respectively.

used to assess whether an ontology provides a specific enough description of the domain of interest (Noy and McGuinness, 2001). A good set of competency questions should target TBox, ABox, and the combination of both. A satisfactory response to predefined competency questions is one way to assess the suitability of a proposed ontology.

Ontology development can follow a “top-down” or “bottom-up” approach, or a combination of both. The former targets broad applicability by defining high-level concepts first before adding any increasingly specific terminologies, while the latter focuses on introducing tailored concepts to represent specific applications or data sources first (i.e., before any optional generalization). A hybrid approach initially defines salient concepts before either generalizing or specializing them as needed. In this article, a hybrid approach is used, guided by available data as well as the target use case, while ensuring a sufficient level of generality to foster re-usability. Ontology development is an iterative process (Noy and McGuinness, 2001), consisting of (1) defining the domain and scope of the ontology, (2) identifying core concepts and relationships for the intended use case, (3) re-using existing concepts and ontologies to the extent possible to foster integration between applications and leverage previous efforts by domain experts, and finally (4) encoding classes, properties, individuals, and restrictions (i.e., value types, cardinality, domain and range restrictions) in OWL format.

The following subsections delineate the development of three domain-specific ontologies and their interconnections. Although the ontologies strongly reflect the data and structure of the resources detailed in [Section 2.4](#), they are kept as general as practicable to ease re-usability. Concepts from existing ontologies are re-used where applicable and links to equivalent external concepts are introduced to enrich the meaning of instantiated data. Re-using established ontologies builds on the collective understanding of a domain developed and codified by subject matter experts. It leverages and enhances prior domain expert efforts for accurate representation of concepts, relationships, and rules widely shared among practitioners, which can further be refined and/or extended to accommodate use case-specific requirements.

The names of ontological classes and properties are written in typewriter font. All three ontologies rely on the established Ontology of units of measure (OM) (Rijgersberg et al., 2020) to represent measured or reported quantities, numerical values, and associated units. Furthermore, an aligned representation of geospatial information using GeoSPARQL (OGC, 2012) is used. All proposed ontologies have been checked for consistency using the Hermit reasoner in the Protege editor (Musen, 2015) to identify and eliminate logical contradictions. Sets of competency questions have been formulated to evaluate the ontologies’ abilities to capture relevant information about their respective domain of interest, with subsets provided in [Supplementary Material SI.7](#). A formal representation of the ontologies using description logic (Baader et al., 2007) is provided in [Supplementary Material SI.8](#), with the OWL versions being publicly available on GitHub.

3.2.1. Geospatial and time series representation

The following approaches are used to overcome limitations in representing geospatial and time series data (see [Section 2.6](#)):

To avoid using custom-typed Literals to encode geospatial coordinates suitable for Blazegraph’s geospatial engine (as done in previous studies: Chadzynski et al., 2021; Akroyd et al., 2022), OBDA via Ontop (Xiao et al., 2020) is used with the actual geospatial information being stored in a PostGIS database. PostGIS is an open-source spatial database extender for PostgreSQL, enabling the storage and retrieval of geospatial data. It adds support for geographic objects and geospatial functions, making it a powerful tool for managing spatial information within a relational database. Ontop, on the other hand, is a semantic data virtualization system that allows users to query relational databases via SPARQL as if they were triple stores. It “hides” the actual data source and exposes the data in terms of the mapped ontological concepts and relationships based on predefined mappings between SQL column and ontology concept names. Ontop facilitates the transformation of relational data into a semantic representation, promoting interoperability and advanced querying capabilities. Although Ontop is not fully compliant with OGC GeoSPARQL 1.0, it supports the main geospatial properties and functions. Hence, geospatial information

can be served as virtual triples according to provided OBDA mappings and combined with actually instantiated data in Blazegraph.

To overcome issues with instantiating myriad individual time stamps, a light-weight time series ontology (Lee et al., 2021) has been developed together with a dedicated *TimeSeriesClient*: upon creation, each time series gets instantiated within the KG with full semantic markup, while the actual tabular data (i.e., time stamp and data value) get stored in an associated relational database (e.g., PostgreSQL). The client provides a robust mapping between any data instance in the KG and the respective values stored in the relational database. Adding or deleting time series data only adds or deletes data points in the database, while deleting an entire time series via the *TimeSeriesClient* also deletes all associated relationships from the KG.

3.2.2. Environmental Measurement Station ontology

Several sensor ontologies have been proposed in the literature (see Section 2.5.1); however, most of them focus on detailed representations of measurement devices and procedures together with actual measurement values. In contrast, the Environmental Measurement Station ontology (OntoEMS) aims to establish an aligned conceptualization for environmental measurement observations from a variety of sources. OntoEMS disregards a comprehensive sensor and procedure representation, as such information is mainly unavailable from public APIs (such as the Met Office DataPoint, EA Real Time flood-monitoring API, or UK-AIR), and focuses on the nature of reported quantities, their geo-location, and time series data for historical readings and forecasts. It represents a light-weight top-level ontology designed to accommodate various data sources without imposing strict limitations on the required data coverage, while still providing an appropriate semantic representation of measured quantities. Given its modular design, OntoEMS can easily be extended with more detailed domain knowledge.

As depicted in Figure 4, the central concept is a *ReportingStation*, which represents any entity reporting environmental observations and/or forecasts. This general term is used because most of the screened data feeds from public APIs do not contain detailed information about the actual measurement devices. Instead, only high-level metadata regarding the location and nature of the supplied data is given, along with the actual readings data. A *ReportingStation* can refer to a physical asset that houses one or more actual sensors/measurement devices or a virtual station that provides certain readings. To enable geospatial capabilities using GeoSPARQL, each *ReportingStation* is defined as a *rdfs:subClassOf* a *geo:Feature*. All reported quantities are defined as a *rdfs:subClassOf* a *om:*

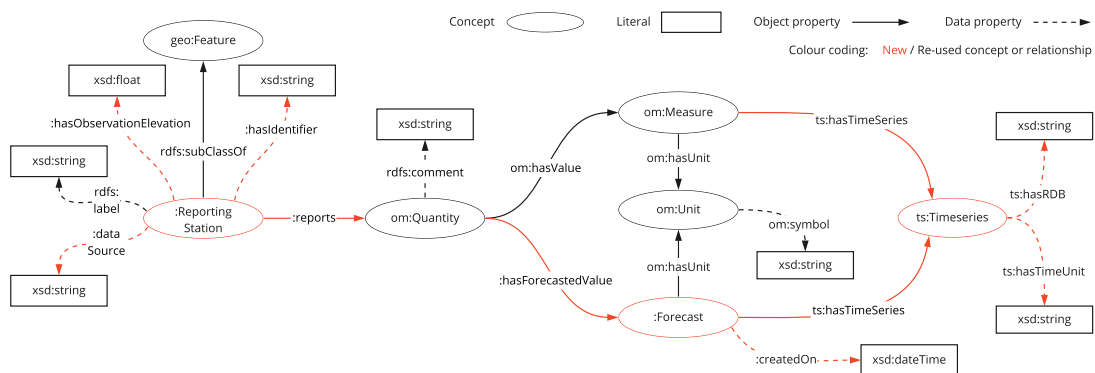


Figure 4. OntoEMS top-level ontology. OntoEMS represents the top-level ontology to represent environmental reporting stations (e.g., measurement stations) and associated readings of environmental observations (including time series values). Further domain knowledge can be incorporated with domain ontologies enriching *ontoems:ReportingStation*, *om:Quantity*, and *om:Measure*. All referenced namespaces are declared in Appendix A.1, with missing explicit namespace declarations referring to *ontoems*. Newly introduced concepts and relationships are depicted in red, while re-used ones are shown in black.

Quantity and time series data are represented using the Time Series ontology (Lee et al., 2021). Figure 5 illustrates this definition for WaterLevel and Rainfall measurements, which are declared as subclasses of om:Height, which itself is a subclass of om:Quantity. References to equivalent external concepts are captured using owl:equivalentClass or rdfs:subClassOf relationships, depending on the actual definition of the concept.

The OntoEMS top-level ontology can easily be refined to provide a more detailed description of specific domains, as illustrated in Figure 6 for water-level reporting stations. In this case, a WaterLevelReportingStation subclass is introduced to include additional information about attached water bodies, such as the catchment or river name, as well as the relative location to other stations along the same river (i.e., up- and downstream stations). A Range and Trend concept are included to provide additional context for current water level measurements. Such domain-specific extensions facilitate the direct import of application-specific ontologies, such as the RDF data model utilized by the EA Real Time flood-monitoring API (Environment Agency, 2021b). Furthermore, additional ABox level assertions can be incorporated, for example, to link instantiated air pollutants with equivalent instances in the European

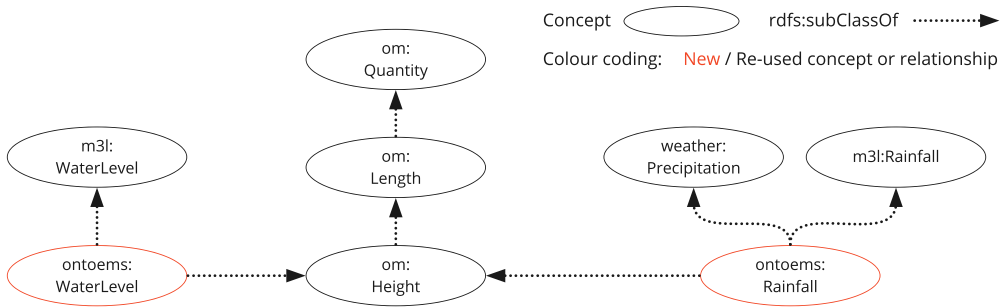


Figure 5. Example definition of a reported quantity. OntoEMS design requires reported quantities to be subclasses of om:Quantity; exemplarily depicted for rainfall and river level measurements, including references to further ontologies to leverage the power of Linked Data.

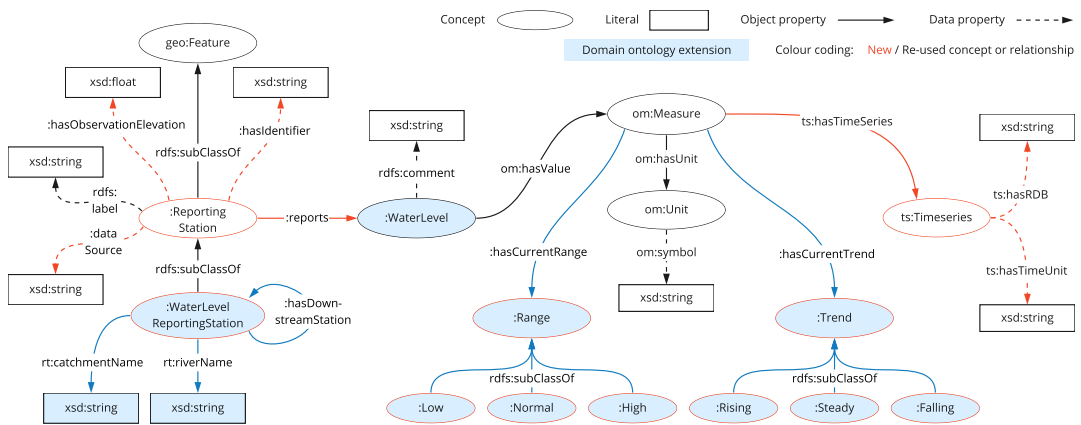


Figure 6. Example OntoEMS domain ontology extension. The OntoEMS top-level ontology can be refined to suit needs for more detailed domain representations by elaborating ontoems:ReportingStation, om:Quantity, and om:Measure; exemplarily shown for water-level reporting stations monitoring the water level in rivers. All referenced namespaces are declared in Appendix A.1, with missing explicit namespace declarations referring to ontoems. Newly introduced concepts and relationships are depicted in red, while re-used ones are shown in black.

General Multilingual Environmental Thesaurus (GEMET) (European Environment Information and Observation Network, 2021) using `owl:sameAs` relationships.

Several amendments to the Ontology of units of measure have been proposed (i.e., the inclusion of new quantities and corresponding units), which are currently being tracked in a fork of the official ontology. It is intended that these amendments will be submitted as a pull request in the future.

3.2.3. Flood Risk ontology

The aim of the proposed Flood Risk ontology (OntoFlood) is twofold: (1) to provide a reasonably general conceptualization of the flood risk domain, while (2) providing a suitable description for the available flood warning and forecast data provided by the EA Real Time flood-monitoring and Flood Forecast APIs (Environment Agency, 2021b; Flood Forecasting Centre, 2017). Compared to many previous ontology efforts, the focus is less on establishing a full-fledged conceptual representation of the entire domain and more on providing a semantic description of relevant phenomenological data to instantiate and operationalize actual data feeds. The majority of previously identified ontologies (see Section 2.5.2) is not publicly available and direct re-use of core concepts and relations is not possible. Hence, the majority of required concepts and relationships is created independently in OntoFlood and links to previously published ontologies are only included for a few classes where useful to enrich the semantic meaning (i.e., ENVO (Buttigieg et al., 2016), SWEET (Buttigieg et al., 2018)). Nevertheless, key design choices in OntoFlood are based on previous conceptualizations to build on existing domain knowledge. Several concepts are borrowed directly from the data model used by the EA Real Time flood-monitoring API (Environment Agency, 2021b) (prefixed with `rt`). Compared to previous flood ontologies, the proposed ontology represents the areal extent of a flood by any arbitrarily shaped polygon instead of linking a flood to a particular predefined region or district. This allows for more versatile geospatial representation, similar to the `flood_hazard_map` concepts developed by Scheuer et al. (2013). However, the primary purpose is an accurate assessment of affected buildings and people using geospatial queries, as compared to pure visualization purposes.

OntoFlood is capable of describing floods and associated impacts in three different stages: actual flooding events, flood alerts and warnings (i.e., expected flooding events), and flood forecasts (i.e., potential flooding events, but less certain). The core concept to describe a flood is the `envo:flood` class, which is defined as a `rdfs:subClassOf` of `soph:Event`. Each event is associated with a certain time interval and a `Location` which provides information about the parent `AdministrativeDistrict` as well as the actual `ArealExtentPolygon` of the flood. The `ArealExtentPolygon` is defined as subclass of a `geo:Feature` and the encompassing `AdministrativeDistrict` shall either be the corresponding IRI from ONS Geography Linked Data (Office for National Statistics, 2022) or refer to it via an `owl:sameAs` relationship. This enables geospatial capabilities, such as the identification of certain `geo:Features` within a flood polygon or insights into most relevant flood areas in a particular county. An event can `resultIn` a monetary `Impact` (positive or negative) describing the total monetary consequences of its occurrence. In the case of flooding, this value describes the total (potential) damage to all affected infrastructure assets. More detailed insights into the (potential) effects of a `envo:flood` are represented using the `affects` relationship, connecting to `Population` and infrastructure assets, such as `Buildings`, representing both the count and estimated monetary value.

The `rt:FloodAlertOrWarning` is the key concept for the flood warnings and alert stage, which represents the key data provided by the flood-monitoring API (Environment Agency, 2021b). Each alert or warning `warnsAbout` a potential `envo:flood` event and is associated with a certain `Severity` and `rt:FloodArea`. Required levels of severity are defined and instantiated as `ABox` together with the `Tbox` of the ontology. The concept of a `FloodAlertOrWarningHistory` is introduced to store key information (i.e., date of issue/change, severity, impacts) of previously issued flood alerts and warnings for a particular `rt:FloodArea`. This allows us to investigate potential trends in both frequency and severity of flood warnings for a certain area and tailor efforts where to take preventive measures. The flood forecast stage is centered around the `FloodForecast` concept, which predicts a potential `envo:`

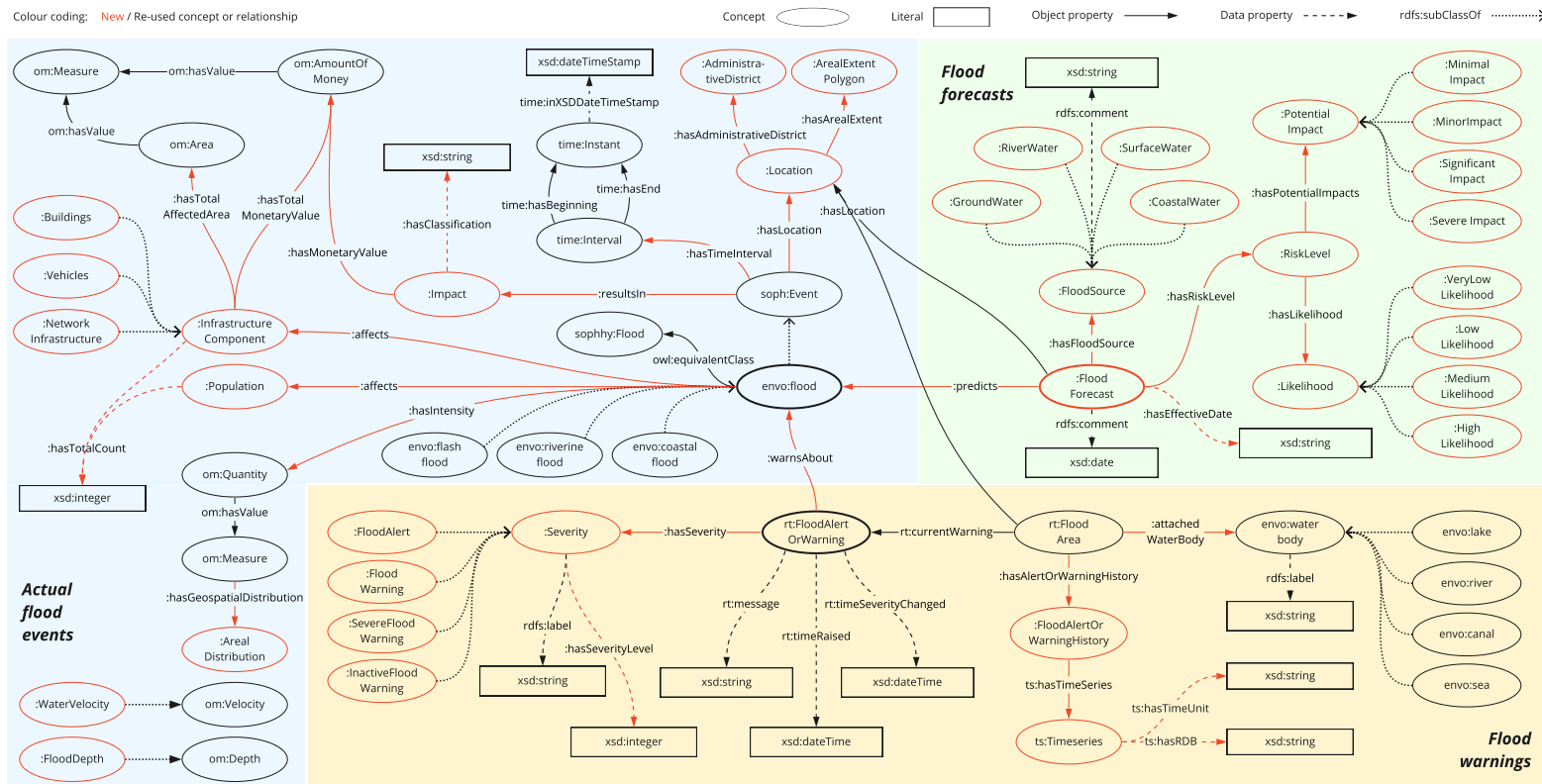


Figure 7. OntoFlood ontology (extract only). The OntoFlood ontology describes flood events and their (potential) impacts in three stages: (1) actual flood events, (2) flood warnings, and (3) forecasted floods. The ontology has been designed to represent available data from the Environment Agency Real Time flood-monitoring API (Environment Agency, 2021b) and assess built infrastructure as well as population at risk. All referenced namespaces are declared in Appendix A.1, with missing explicit namespace declarations referring to `ontoflood`.

flood event. As a flood forecast is less certain and, hence, less quantitative than a `rt:Flood-AlertOrWarning`, it is mainly characterized by its `RiskLevel` expressing a hazard potential defined by expected likelihood and impact.

3.2.4. Building Environment ontology

The Building Environment ontology (OntoBuiltEnv) aims to create a lightweight ontology to represent properties (i.e., buildings and flats) suitable for a variety of use cases within TWA. It is designed to integrate with existing agents and knowledge representations, such as OntoCityGML and the City Energy Analyst (CEA) Agent (Chadzynski et al., 2023a). Compatibility and interoperability with previous building instantiations based on OntoCityGML is ensured by introducing a link between corresponding building instances across both ontologies. While OntoCityGML represents all geospatial and geometric aspects of a building (i.e., in various LoDs), OntoBuiltEnv provides a semantic representation of additional building information (i.e., to replace previously used non-semantic `ExternalReference` and `genericAttribute`). The ontology is designed to represent publicly available building data based on the Energy Performance Certificate and HM Land Registry APIs (Department for Levelling Up, Housing, and Communities, 2022; HM Land Registry, 2022c) and follows a use case specific approach, considering only a required subset of all available data for now, that is, focusing on key construction properties as well as property market value relevant information. However, the design ensures extensibility to accommodate additional use cases in the future and already captures building usage as well as solar photovoltaic descriptions to support ongoing efforts to refine the CEA agent.

The ontology reuses existing concepts from multiple screened ontologies (see Section 2.5.3), such as DABGEO or iCity Building (Cuenca and Larrinaga, 2019; Katsumi, 2021). A few public concepts from the custom HM Land Registry data model (HM Land Registry, 2022b) are included to represent property market data (prefixed with `lrppi`). A `Property` represents the central concept of the ontology, with `dabgeo:Building` and `Flat` as relevant subclasses. While each `Property` holds multiple relationships regarding its location, construction characteristics, and market valuation details, certain information is restricted to buildings only, for example, elevation and roof area specifications (see Figure 8). The `dabgeo:Building` class is the key concept to represent buildings and the `hasOntoCityGMLRepresentation` relationship connects it with an external IRI representing the corresponding OntoCityGML instance containing all geospatial information. To enrich semantics, `owl:equivalentClass` relationships are included to refer to building definitions in further ontologies. This ensures linking and easier integration of potentially more elaborate building models in the future (e.g., BIM representation). All buildings are again represented as `geo:Feature` to enable geospatial capabilities using GeoSPARQL (e.g., to assess whether a building is located within a particular flood polygon). Each property is connected with an `icontact:Address` containing detailed address information, including `owl:sameAs` links to corresponding statistical ONS geography IRIs for instantiated `PostalCodes` and `Administrative-Districts` to enable further navigation and geospatial analyses.

Figure 9 illustrates the current representation of `PropertyUsage` types, which has been aligned between available data from the API and requirements from the CEA agent. Furthermore, several subclasses have been introduced to formalize relevant property types (e.g., maisonette or house), built forms (e.g., terraced or detached), and key construction components. The `hasIdentifier` relationship is used to provide a unique identifier for each instantiated building, required for external referencing or instance matching. In UK context this corresponds to the UPRN of the property, which can be used to unambiguously identify each property and potentially assimilate further data in the future. The `hasLatestEPC` refers to the identifier of the underlying energy performance assessment of the instantiated data and can be used to assess whether this data still reflects latest information available from the API.

3.3. Agent framework

The developed ontologies are used by a set of input agents to dynamically assimilate data from a variety of sources (see Section 2.4) as well as multiple autonomous agents which subsequently operate upon it. As

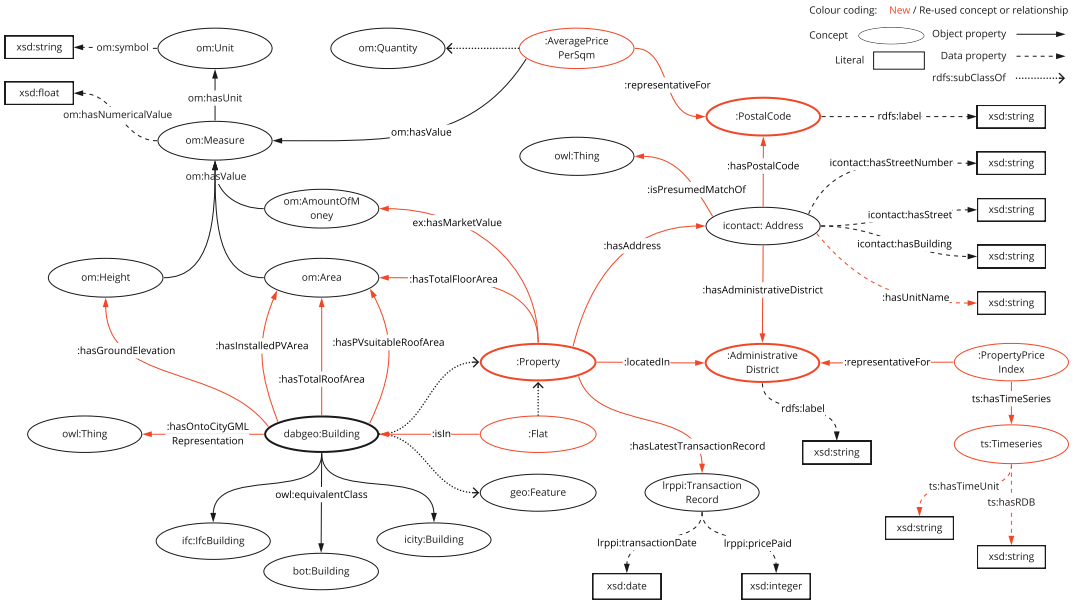


Figure 8. OntoBuiltEnv ontology part 1 (extract only). The OntoBuiltEnv ontology provides a semantic description for properties (i.e., both buildings and flats), including location and address details as well as information about previous sales transactions and current market value estimates. The ontology has been designed to represent available data from both Energy Performance Certificates (Department for Levelling Up, Housing, and Communities, 2022) and His Majesty’s Land Registry (HM Land Registry, 2022a), while seamlessly integrating with OntoCityGML (Centre Universitaire d’Informatique at University of Geneva, 2012) for the geospatial representation of buildings. All referenced namespaces are declared in Appendix A.1, with missing explicit namespace declarations referring to ontobuiltenv.

introduced in Section 3.1, latest flood warnings from the EA API get instantiated on an ongoing basis, before being processed by a sequence of agents to understand the share of population which is potentially affected, the number of buildings located within the flood area and how that translates into building stock value at risk of flooding.

The overall ecosystem of interacting agents is schematically depicted in Figure 10 to provide a high-level overview of the interplay between agents, external data sources, and ontologies within TWA. The sequence of key agent interactions in the automated flood impact assessment is depicted in Figure 11. While the initial data instantiation relies on active user input, subsequent tasks operate autonomously, with agents communicating directly through the knowledge graph via DIF. All derivation agents (i.e., agents deriving new information based on existing entities using the DIF) honor the OntoAgent (Zhou et al., 2019) ontology to refer to associated inputs, outputs, and their respective service descriptions. All agents are implemented in Python or Java and packaged as individual Docker containers. Details about agent deployment are provided in Section 4.

An overview of key agent tasks and dependencies (i.e., due to information maintained by other agents) is provided in Table 1, before explaining each agent in detail below. Activity diagrams are used to illustrate key agent logic in a simple manner, avoiding unnecessary complexities that might impede a comprehensive understanding; however, a few pseudocode snippets and example SPARQL queries are provided in Supplementary Materials SI.2 and SI.3 for illustration purposes. While the pseudocode reveals further details about the underlying algorithms, it needs to be noted that the actual implementation might differ for performance considerations or the use of established libraries, such as pandas.

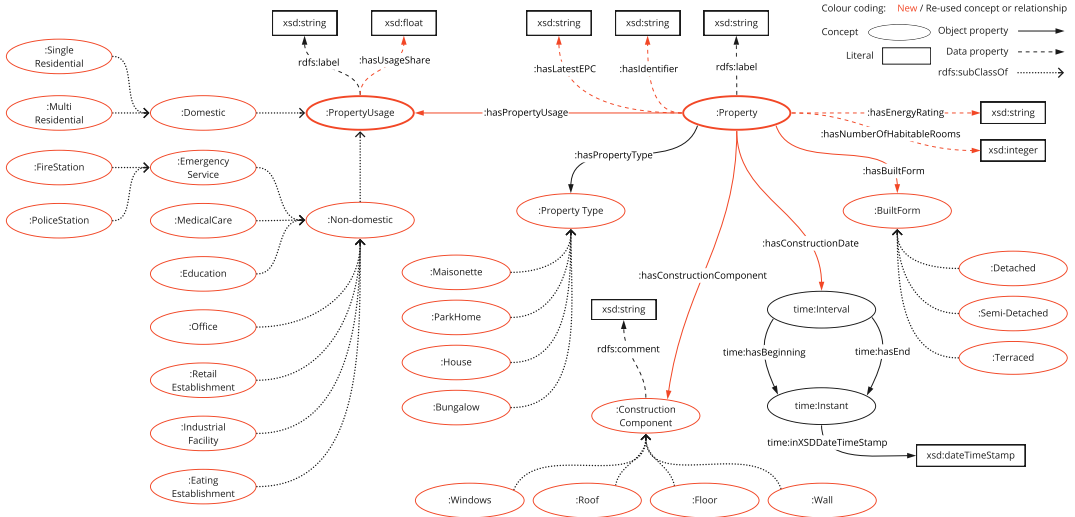


Figure 9. OntoBuiltEnv ontology part 2 (extract only). The OntoBuiltEnv ontology provides relevant concepts to represent properties, including their usage classification, major construction components, property type and built form, and energetic characteristics.

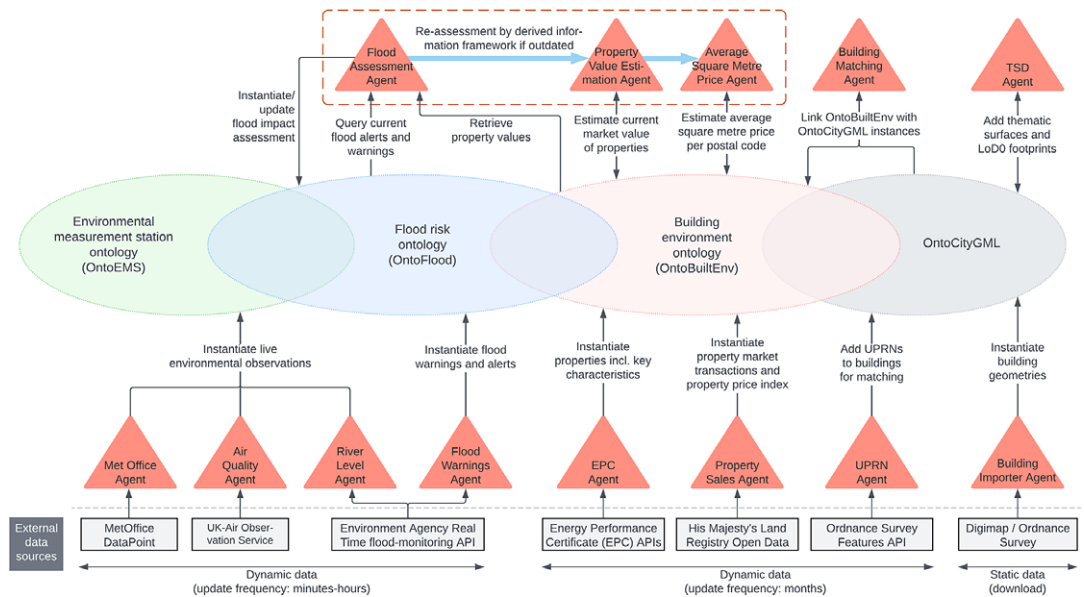


Figure 10. Agent Ecosystem. Schematic depiction of all agents involved in the flood assessment use case. Input agents (i.e., all agents in the bottom row of the figure) interact with external data sources to instantiate (or update) data within the KG, while other software agents operate (autonomously) on the instantiated data.

3.3.1. Met Office Agent

The *Met Office Agent* recurrently queries the Met Office DataPoint API (Met Office, 2022) for latest weather observation and forecast data and instantiates it according to the OntoEMS ontology. This agent also serves as template for other OntoEMS input agents to keep TWA in sync with the real world. A

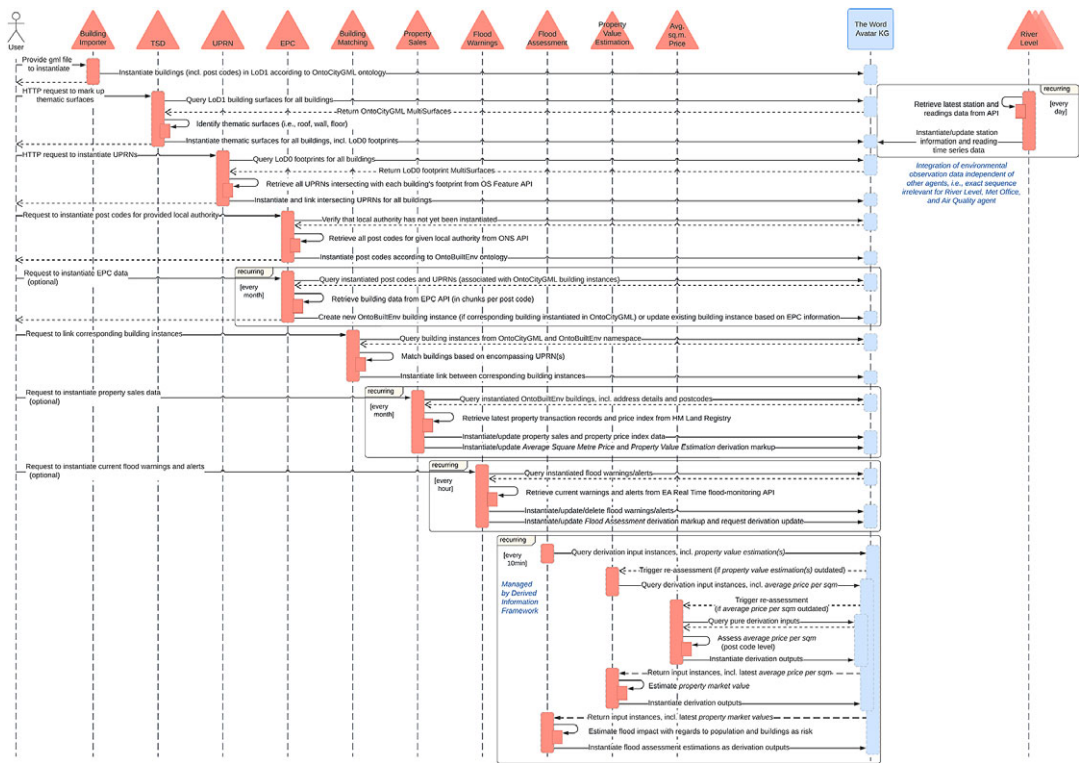


Figure 11. Agent Sequence. Sequence diagram of key agent interactions and dependencies required for the automated flood assessment. While initial data instantiation requires active user input, recurring tasks occur automatically and agents communicate directly via the knowledge graph.

detailed Unified Modeling Language (UML) diagram is provided in Figure 12. The agent offers dedicated endpoints to update (1) instantiated reporting stations, (2) reported quantities (i.e., measured and/or forecasted parameters), and (3) the actual time series data of those quantities. Additionally, an endpoint to update all instantiated information is provided, performing tasks 1–3 sequentially: Upon invocation (either via HTTP request or scheduled background task), the agent queries all available reporting stations from both the API and TWA. In case stations are available from the API, but not TWA, they get instantiated. Subsequently, all available stations and their associated readings are queried from both the API and TWA. Similarly, potentially missing quantities are instantiated and the associated time series instances to store actual readings get initialized. Lastly, the latest time series data are queried from the API for all instantiated quantities and added to the respective TimeSeries instances. Potentially, duplicated timestamps are overwritten to ensure that latest information is available from TWA at all times.

3.3.2. Air Quality Agent

The Air Quality Agent recurrently queries air pollutant concentration data from the UK-AIR Sensor Observation Service (Department for Environment, Food, and Rural Affairs, 2023) and instantiates it according to the OntoEMS ontology. It follows the same implementation approach as the Met Office Agent, with consecutive updates of reporting stations, measured quantities, and actual time series data. Similarly, only not yet instantiated stations and readings are added to the KG on subsequent agent invocations. To leverage the power of Linked Data, each unambiguously identifiable air pollutant reading is linked to its equivalent concepts of the European air quality e-Reporting initiative (European Environment Information and Observation Network, 2021) via an owl:sameAs relationship. This provides

Table 1. Agent overview

Agent	Agent type	Key task(s)	Required predecessor
Met Office Agent	Input agent	Instantiate latest weather observation and forecast data	
Air Quality Agent	Input agent	Instantiate latest air pollutant observation data	
River Level Agent	Input agent	Instantiate latest water level readings (as well as water flows and rainfall data for some locations)	
Building Importer Agent	Input agent	Instantiate buildings from CityGML files (using OntoCityGML ontology)	
TSD Agent	Update agent	Enhance instantiated LoD1 buildings into LoD2 by identifying thematic surfaces	Building Importer Agent
UPRN Agent	Input agent	Instantiate UPRNs for instantiated buildings	TSD Agent
EPC Agent	Input agent	Instantiate properties with their energy and construction characteristics based on EPC data (using OntoBuiltEnv ontology)	
Building Matching Agent	Update agent	Link corresponding building instances between OntoBuiltEnv and OntoCityGML (based on their UPRN)	UPRN Agent, EPC Agent
Property Sales Agent	Input agent	Instantiate latest property sales transactions and price index data	EPC Agent
Average Square Meter Price Agent	Derivation agent	Estimate average square meter price of properties per postcode	Property Sales Agent
Property Value Estimation Agent	Derivation agent	Estimate the current market value of properties	Property Sales Agent
Flood Warnings Agent	Input agent	Continuously instantiate latest flood alert and warnings data	
Flood Assessment Agent	Derivation agent	Estimate the number of people and buildings as well as the total building stock value at risk of flooding from any flood warnings	Flood Warnings Agent

Note. Overview of all agents, their key task(s), and their required predecessors, that is, agents which need to be executed prior to (1) instantiate required data within TWA or (2) instantiate required derivation markup to reflect information interdependencies. Input agents instantiate new data into TWA, update agents operate upon instantiated data to enrich/connect individual instances, and derivation agents deduce new information based on existing entities using the DIF.

additional information about relevant protection targets, (commonly used) units, and measurement equipment, besides further metadata. Moreover, this eases alignment and interoperability with other data sources and stations potentially to be incorporated later.

3.3.3. River Level Agent

The *River Level Agent* retrieves sensor observation data from the EA flood-monitoring API (Environment Agency, 2021b) and instantiates it according to the OntoEMS ontology. It downloads all available data in RDF format once per day and updates the KG by assimilating new station and readings information as well as adding new time series data. The agent instantiates all provided readings, not just river levels, including water flow rates, rainfall, and wind measurements. For `WaterLevelReportingStations` at rivers the agent enriches the RDF data with additional information about stage scales, such as

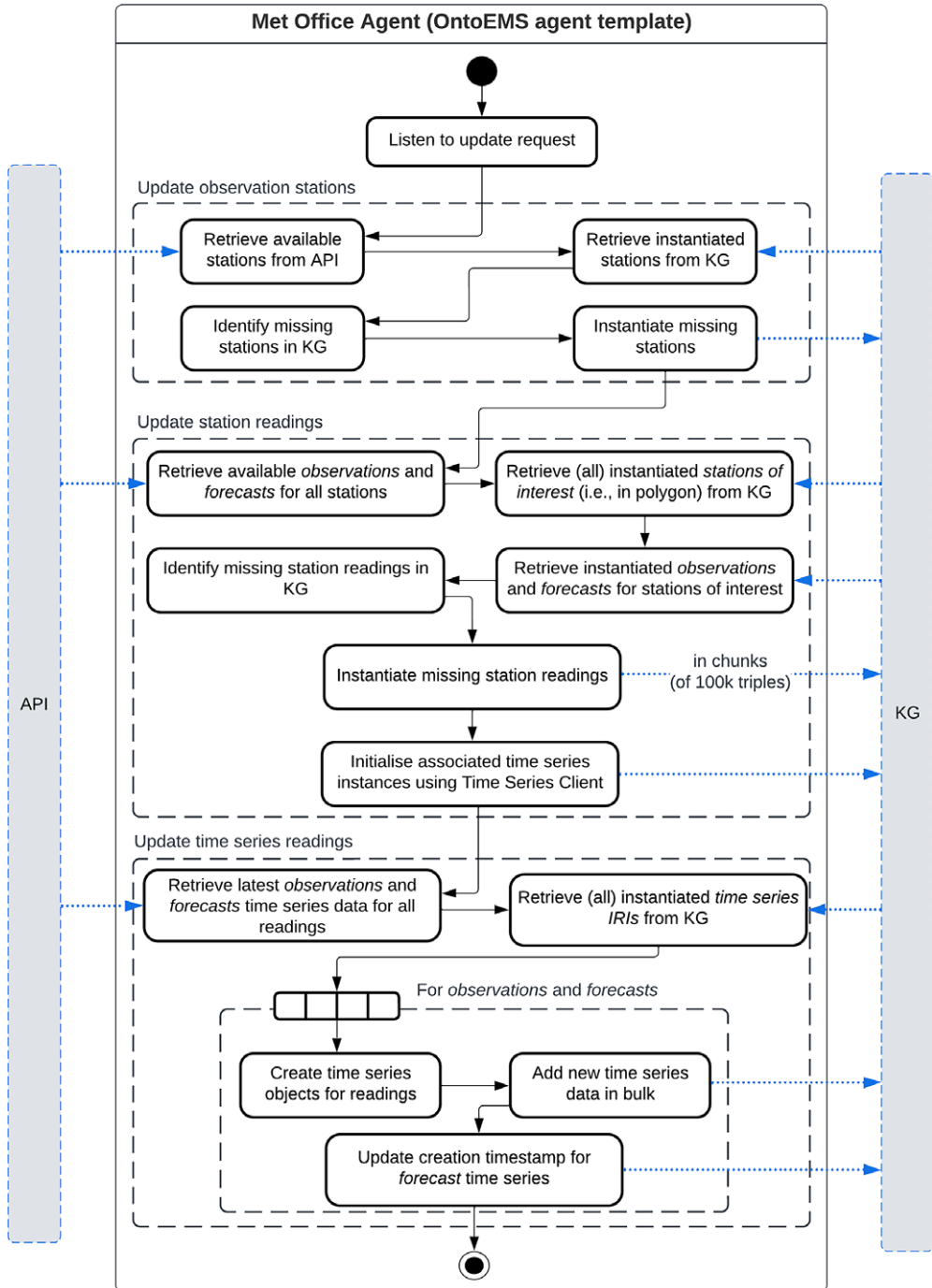


Figure 12. Met Office Agent. The Met Office Agent recurringly queries the Met Office DataPoint API (Met Office, 2022) for both latest weather observation and forecast data and instantiates it according to the OntoEMS ontology. This agent also serves as template for other OntoEMS input agents to keep TWA in sync with the physical world.

the reference datum and typical low/high readings, and adds relationships to potential downstream stations. Data about the relative location of stations is obtained from another government service (Met Office and Environment Agency, 2022) through web scraping. The agent also derives and instantiates the current `Range` and `Trend` for each `WaterLevelReportingStation` with associated stage scale instance. The `Range` is determined by comparing the latest water level reading with typical low and high readings, while the `Trend` is based on an assessment of the readings over the last 12 hours. A rising or falling `Trend` is instantiated if the difference between the last and first value in that period exceeds $\pm 10\%$, respectively.

3.3.4. Building Importer Agent

The *Building Importer Agent* imports entire CityGML files and instantiates respective buildings according to the `OntoCityGML` ontology (Chadzynski et al., 2023a, 2023b). The agent supports building representations in multiple LoDs; however, given the available data from OS, this work relies on building instantiations in LoD1, with buildings represented as simple block models describing their overall shape and layout (i.e., building footprints as polygon). Additional building data, such as building height or associated postcode, are also instantiated according to `OntoCityGML`, either as dedicated data property (e.g., building height) or as `genericAttribute` (e.g., postcode).

3.3.5. Thematic Surface Discovery Agent

The *Thematic Surface Discovery (TSD) Agent* enhances instantiated `OntoCityGML` LoD1 building representations to LoD2 (Chadzynski et al., 2023a). The agent identifies wall, roof, and ground polygons and adds explicit semantic annotations to complement their pure geometrical perspective in an `OntoCityGML` compliant form. The agent also offers an endpoint to only identify and restructure the ground surface, that is, footprint, of a building. As this work focuses on LoD1 building representations, most of the thematic surface restructuring is not necessary; however, the ground surface classification is a prerequisite for the instantiation of UPRN information by the *UPRN Agent*.

3.3.6. Unique Property Reference Number Agent

The *Unique Property Reference Number (UPRN) Agent* enriches instantiated `OntoCityGML` building instances with a unique identifier, that is, their UPRN given the UK context of this work. Depending on the provided HTTP request parameters, the agent processes individual buildings (i.e., single `OntoCityGML` city object with provided IRI) or targets all buildings instantiated in a provided triple store namespace. For each target building, the agent queries the OS Features API (Ordnance Survey, 2022) with the building's bounding box to retrieve all enclosed UPRNs. Subsequently, all returned UPRNs are tested against the ground surface to exclude UPRNs not intersecting the building itself. Only intersecting UPRNs are then instantiated and linked to the respective building(s).

3.3.7. Energy Performance Certificate Agent

The *Energy Performance Certificate (EPC) Agent* retrieves data from all three EPC APIs (Department for Levelling Up, Housing, and Communities, 2022) (i.e., domestic, non-domestic, and display certificates) and instantiates relevant building data according to the `OntoBuiltEnv` ontology. To manage the size of individual API requests, the data is queried per postcode and for all three APIs subsequently. To instantiate all postcodes for a particular local authority, an initial request must be sent to the agent. New postcodes are only instantiated if the local authority is not already present in TWA. All instantiated postcodes get linked to corresponding ONS IRIs using the `owl:sameAs` relationship. A detailed UML diagram is provided in [Supplementary Material SI.1](#), with key agent logic described below:

The agent requires an available SPARQL endpoint to retrieve instantiated `OntoCityGML` buildings, since EPC data is only assimilated for properties with available geospatial representation. Initially, all instantiated postcodes are queried from TWA, and the latest EPC data for all instantiated postcodes are retrieved from the respective API endpoint. Returned EPC data with a corresponding UPRN instantiated

in *OntoCityGML* gets instantiated according to the *OntoBuiltEnv* ontology: either as standalone building or as property belonging to a parent building if the associated *OntoCityGML* building contains multiple UPRNs. Only outdated EPC data is instantiated, while already instantiated data are skipped. The agent determines information for parent buildings by aggregating data of contained child properties: by summing up actual values (e.g., number of rooms and floor area), using the most common value (e.g., EPC rating code, property type), or concatenating distinct values of child properties (e.g., property usage, description of construction components). After ingesting new information and to ensure that updated information is detectable by the DIF, the IRIs of all (potential) pure inputs managed by the *EPC Agent* (i.e., *Property*, *PostalCode*, and *FloorArea* instances) are retrieved and associated timestamps are added or updated.

To enable visualization using TWA's unified visualization interface, the geospatial representation of all buildings (i.e., their 2D footprints and elevations) must be uploaded to PostGIS. The agent provides all necessary functionality to do so; however, corresponding building instances in *OntoBuiltEnv* and *OntoCityGML* need to be matched beforehand using the *Building Matching Agent*.

3.3.8. Building Matching Agent

The *Building Matching Agent* links buildings instantiated according to *OntoBuiltEnv* with their *OntoCityGML* equivalents. While *OntoBuiltEnv* provides the overall semantic description of properties, *OntoCityGML* is solely used for the geospatial representation of buildings. Linking corresponding instances allows the combination of both of these complementary perspectives. Matching is conducted using UPRNs as unique identifiers and realized by instantiating one additional relationship between matched IRIs, namely *hasOntoCityGMLRepresentation*. For details please refer to [Supplementary Material SI.1](#).

3.3.9. Property Sales Agent

The *Property Sales Agent* is an input agent which queries HM Land Registry Open Data and instantiates it according to the *OntoBuiltEnv* ontology. More precisely, it queries latest property sales transactions and price index data from the Price Paid (HM Land Registry, 2022b) and the UK House Price Index Linked Data (HM Land Registry, 2022d), respectively, via the publicly available SPARQL endpoint (HM Land Registry, 2022c). After instantiating new property sales data, the agent also instantiates the relevant derivation markups to allow for the automatic assessment of the *AveragePricePerSqm* per *PostalCode* as well as the *PropertyValueEstimation* of individual buildings. A detailed UML diagram is provided in [Supplementary Material SI.1](#), with key agent logic described below:

Initially, all instantiated properties are retrieved from TWA, including their full address information. Subsequently, for each postcode latest transaction records (i.e., latest transaction date and price paid) are retrieved for all unique addresses from Land Registry's SPARQL endpoint. Since the PPD does not contain UPRN information, sales transactions are assimilated based on address matching. To attach a previous sales record to an instantiated property, both postcode and property type (i.e., building or flat) must match. However, to account for minor discrepancies between instantiated address information (i.e., based on EPC data) and addresses from HM Land Registry, fuzzy matching of the concatenated string of street name, street number, building name, and unit name is used. If the fuzzy match exceeds a minimum confidence score (i.e., 95) both addresses are considered equivalent and the retrieved *TransactionRecord* is instantiated for the respective property. The Python library *Fuzzywuzzy* (Cohen 2020) is used and two matching algorithms have been compared, that is, the Levenshtein distance and the Damerau-Levenshtein algorithm. Additionally, the performance of multiple scoring methods (i.e., simple ratio, partial ratio, token sort ratio, and token set ratio) has been investigated. The token set ratio calculates the similarity score based on the ratio of the length of the longest common sub-string to the total length of the two strings being compared after breaking them into individual words and accounting for differences in word order and word frequency. Manual comparisons of the matching results suggest that the Levenshtein distance with the token set ratio performs best. While the scoring method has a significant influence, the difference between both scoring algorithms is very minor.

Having assimilated all property sales transaction information, the `PropertyPriceIndex` (i.e., the UKHPI) is instantiated and/or updated for all instantiated administrative districts (i.e., local authorities, as they are the most granular regions for which the UKHPI is published). Finally, the agent also updates the timestamps of amended pure derivation inputs (i.e., `TransactionRecord` and `PropertyPriceIndex` instances) and instantiates/updates the required markups for both the `AveragePricePerSqm` and the `PropertyValueEstimation` derivation. Both of these derivations are initialized as synchronous derivations to compute and instantiate an initial evaluation immediately. As the `PropertyValueEstimation` depends on the `AveragePricePerSqm`, the latter one is marked up first to ensure availability of the required derivation input. Both markup methods are part of the *Property Sales Agent* and described in more detail in the following two paragraphs.

Average Square Meter Price Derivation Markup: This method creates the semantic markup required by the DIF to enable the automatic assessment of the `AveragePricePerSqm` for each instantiated `PostalCode`: for each instantiated `PostalCode` an `OntoDerivation` instance is initialized to be handled by the *Average Square Meter Price Agent* and connecting all input instances via an *isDerivedFrom* relationship. The method also handles postcodes without previous sales transaction records. In such cases, simply no transaction record instances get connected to the derivation instance as inputs, which ensures that the *Average Square Meter Price Agent* in turn queries data from nearby postcodes from the Land Registry SPARQL endpoint. A detailed UML diagram of the markup method is provided in [Supplementary Material SI.1](#), with the key logic described below:

Initially, all unique `PostalCodes` are queried from TWA, followed by the retrieval of all associated `TransactionRecords` and the representative `PropertyPriceIndex`. If already instantiated (i.e., due to previous derivation computation), also the `AveragePricePerSqm` instance for the postcode is retrieved. If no `AveragePricePerSqm` instance exists yet, a synchronous derivation for new information is initialized to connect all `TransactionRecords` of that `PostalCode` as well as the associated `PropertyPriceIndex` with a newly instantiated derivation instance and get an initial assessment computed immediately by the *Average Square Meter Price Agent*. Additionally, not yet instantiated timestamps are added to all input instances. Otherwise, `TransactionRecord` instances not yet connected with the existing `AveragePricePerSqm` instance of the respective postcode (i.e., transaction record has been instantiated after previous average price assessment) are retrieved and added to the existing derivation instance. Finally, a derivation update is requested to ensure the instantiated `AveragePricePerSqm` is up-to-date.

Property Value Estimation Derivation Markup: Similar to the *Average Square Meter Price Derivation Markup* method, this method maintains the required derivation markup for the automatic assessment of the `PropertyValueEstimation` of instantiated buildings. A detailed UML diagram is provided in [Supplementary Material SI.1](#), with the key logic described below:

Initially, all instantiated properties are retrieved, together with their associated sales `TransactionRecord` and `FloorArea` as well as representative `PropertyPriceIndex` and `AveragePricePerSqm` (if available). If already instantiated (i.e., due to previous derivation computation), also the current `PropertyValueEstimation` instance is queried. If no market value estimate exists yet for a building, a synchronous derivation for new information is initialized based on all available inputs for that building and requested for immediate initial assessment by the *Property Value Estimation Agent*. As the `AveragePricePerSqm` derivation is marked up first, its value should already be instantiated when conducting the property value estimation markup. In the unlikely event that this is not the case, the derivation can also be initialized with the `AveragePricePerSqm` derivation instance as input. In case a property already has an instantiated `PropertyValueEstimation`, the need for a potential update is evaluated: In case the instantiated property value *isDerivedFrom* the representative `AveragePricePerSqm` and the `FloorArea`, but a sales `TransactionRecord` is actually instantiated for the property (i.e., a new sales transaction has been made available in HM Land Registry after the last time the property value has been estimated), the respective `TransactionRecord` is added as additional input to the existing derivation and an update is requested to re-assess the current market value. Furthermore, not yet instantiated timestamps are added to all input instances.

3.3.10. Average Square Meter Price Agent

The *Average Square Meter Price Agent* is a derivation agent to estimate the average square meter price of properties on a postcode level. The required input data comprises total `FloorArea` and latest sales `TransactionRecord` for properties as well as the representative `PropertyPriceIndex`. A detailed UML diagram is provided in [Supplementary Material SI.1](#), with key agent logic described below:

Upon invocation, the agent verifies whether received input instances are suitable to assess the `AveragePricePerSqm`, that is, both a `PostalCode` and `PropertyPriceIndex` are available (i.e., have been marked up properly). Subsequently, the number of `TransactionRecords` associated with the derivation instance (i.e., available for current postcode) is derived. The agent requires a minimum threshold of transaction instances to compute a meaningful average (e.g., minimum of five transactions). If less (or even no) transactions are available, data from nearby postcodes are included: the agent retrieves all postcodes within the same Super Output Area from ONS public SPARQL endpoint (Office for National Statistics, 2022), queries the instantiated number of transactions for each of them from TWA and orders the postcodes by increasing euclidean distance from current one. Lastly, the nearest postcodes required to fulfill the minimum number of transactions are extracted and all associated `TransactionRecords` are used in the assessment.

Having retrieved a sufficient set of `TransactionRecords`, each sales price is normalized with the total `FloorArea` of the property and scaled using the `PropertyPriceIndex` to derive today's equivalent value for each historical transaction. Finally, the arithmetic mean is computed and instantiated as current `AveragePricePerSqm` for the respective `PostalCode`.

3.3.11. Property Value Estimation Agent

The *Property Value Estimation Agent* is a derivation agent to calculate the current market value estimate of properties instantiated in TWA. The estimated market value of any property can be determined by either (1) re-calibrating the latest available historical transaction price to today's market conditions or (2) multiplying the total floor area of the property with a representative average square meter price. A detailed UML diagram is provided in [Supplementary Material SI.1](#), with key agent logic described below:

Initially, the agent verifies that sufficient inputs are provided (i.e., have been marked up properly), meaning that either a historical `TransactionRecord` and `PropertyPriceIndex` or total `FloorArea` and `AveragePricePerSqm` instances are available. The first combination is prioritized as actual previous sales transactions for a certain property are considered better proxies for the current market value compared to the more general approach using an average square meter price. Hence, the latter set of inputs shall only be used as fall-back for properties without any previous sales information. After computing the `PropertyValueEstimation` based on available inputs, it is instantiated in TWA as derivation output according to `OntoBuiltEnv`.

Although significantly more elaborate methods exist to estimate property prices, that is, considering numerous influential factors, such as micro-location, building characteristics, usage classification, and so on, this simplified evaluation approach suffices for an initial proof-of-concept. Kept intentionally simplistic for this phase of our research, it will likely be subject to further refinement in future iterations.

3.3.12. Flood Warnings Agent

The *Flood Warnings Agent* is an input agent which queries flood alert and warning data from the EA Real Time flood-monitoring API (Environment Agency, 2021b) and instantiates it according to the `OntoFlood` ontology. For readability, both alerts and warnings are referred to as flood warnings in the following. A detailed UML diagram is provided in [Figure 13](#), with the key agent logic described below:

Initially, all instantiated `FloodAlertOrWarnings` are queried from TWA and matched against available, and hence currently active, ones from the API: (1) Whenever new flood warnings are issued by the API, the agent initially verifies whether the affected `FloodArea` has already been instantiated (i.e., as flood areas are frequently re-used by EA). If the `FloodArea` is already instantiated, the corresponding instance is retrieved. Otherwise, a new instance is created, including all meta information

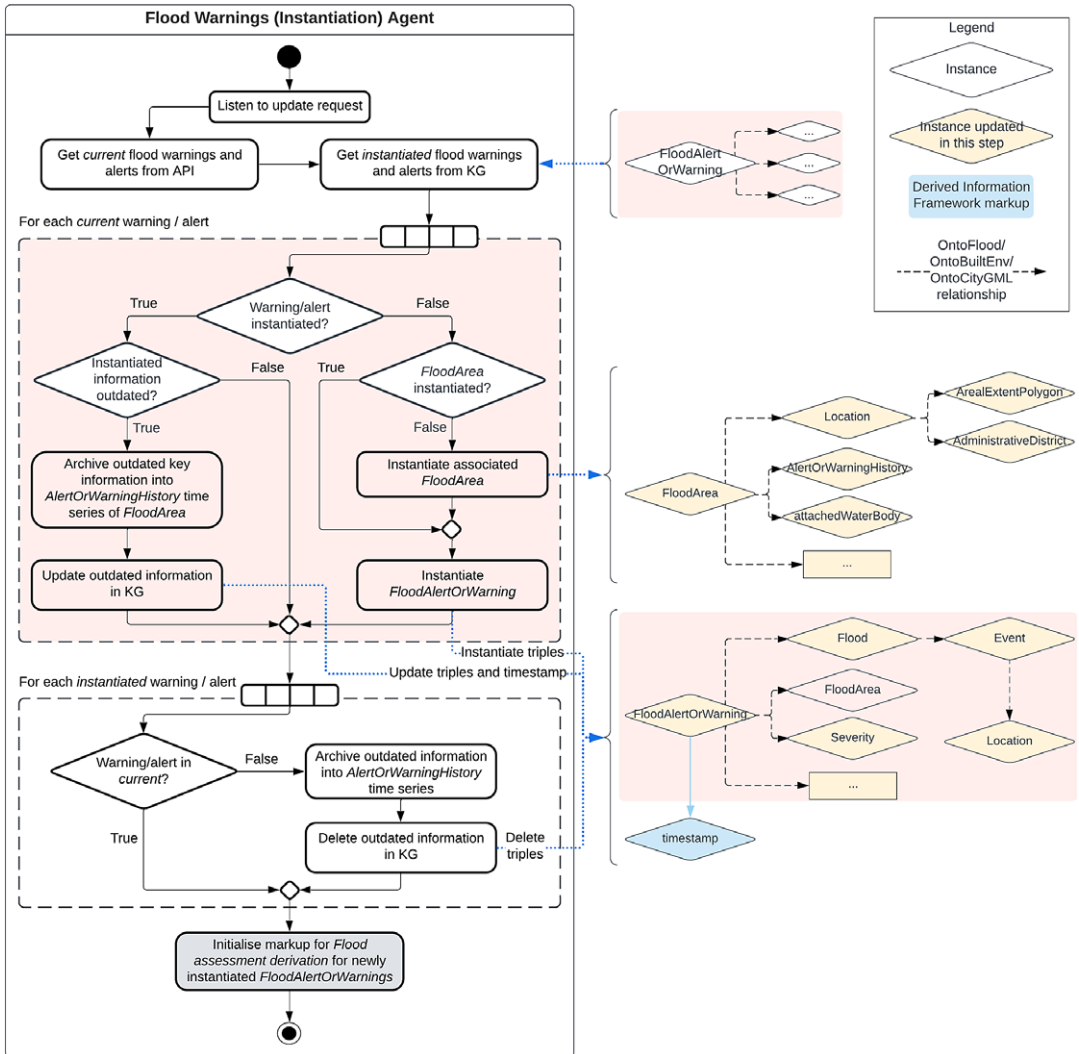


Figure 13. Flood Warnings Agent. The Flood Warnings Agent recurringly (i.e., every hour) queries the EA Real Time flood-monitoring API (Environment Agency, 2021b) and instantiates current flood alerts and warnings. Newly raised alerts/warnings are instantiated (including the instantiation of associated flood area(s)) and already existing ones are updated. Ceased alerts/warnings are deleted from the KG, while associated areas are kept for future reference. Each instantiated alert or warning receives a derived information markup to connect its derivation instance with all relevant inputs for a flood impact assessment. This markup is either newly instantiated or updated (details in Figure 14). Exemplary pseudocode and SPARQL queries for the sections highlighted in red are provided in Supplementary Materials SI.2 and SI.3.

about geospatial extent, attached water body, and so forth. Subsequently, a new FloodAlertOrWarning is instantiated, containing all relevant details about severity, warning message as well as the relationship to the applicable FloodArea. (2) Existing but outdated FloodAlertOrWarnings get updated with latest API information on severity and message details; however, some information, such as the relations to associated FloodArea and Flood event IRI, are kept unaltered. (3) Any previously instantiated warnings that are no longer available from the API are archived. Archiving shall create a log

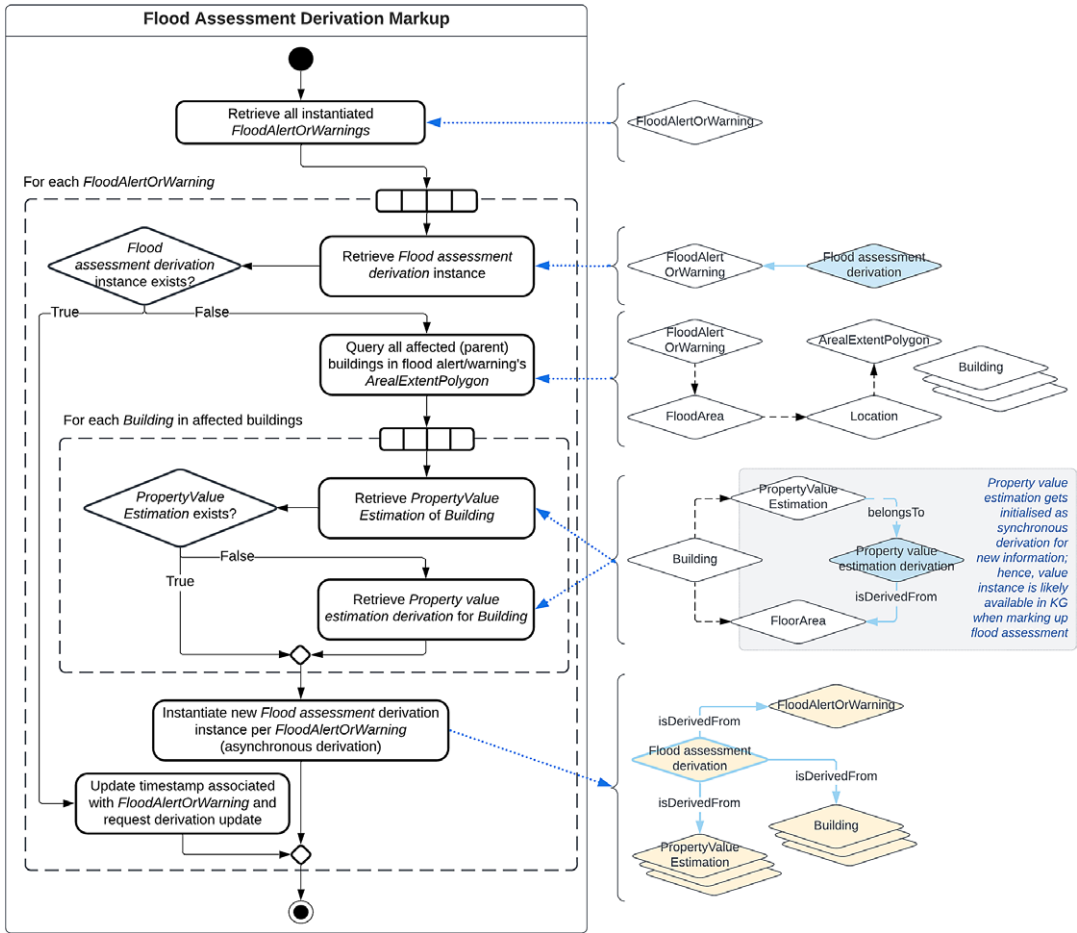


Figure 14. Flood Assessment Derivation Markup. The Flood Assessment Derivation Markup is a method of the Flood Warnings Agent to connect instantiated flood alert/warning information with corresponding flood assessment derivations. Furthermore, all potentially affected buildings (i.e., buildings located within the flood area polygon) are determined and attached to the derivation instance. Those relationships are required to specify the input instances for each flood assessment and allow the Flood Assessment Agent (see Figure 15) to automatically detect any outdated information and trigger a re-evaluation of potential impacts when they are accessed.

of the flood warnings history for a particular FloodArea to allow for elaborate flood risk analyses; however, this feature is not yet fully implemented and obsolete FloodAlertOrWarning instances simply get deleted from TWA, together with all relationships, derivation markup, associated timestamps, and derivation outputs. However, derivation inputs and associated flood areas are not affected, as they may be associated with or re-used by other flood warnings.

After assimilating latest flood warnings, the agent also instantiates the relevant derivation markup and adds or updates associated input timestamps to allow for automatic impact (re-)assessment by the Flood Assessment Agent (details in paragraph below).

Flood Assessment Derivation Markup: This method is part of the Flood Warnings Agent and handles the required derivation markup for the automatic assessment of potential impacts associated with individual flood warnings. It creates and maintains an OntoDerivation instance for each FloodAlertOrWarning, adds all necessary information about the responsible derivation agent, and connects it

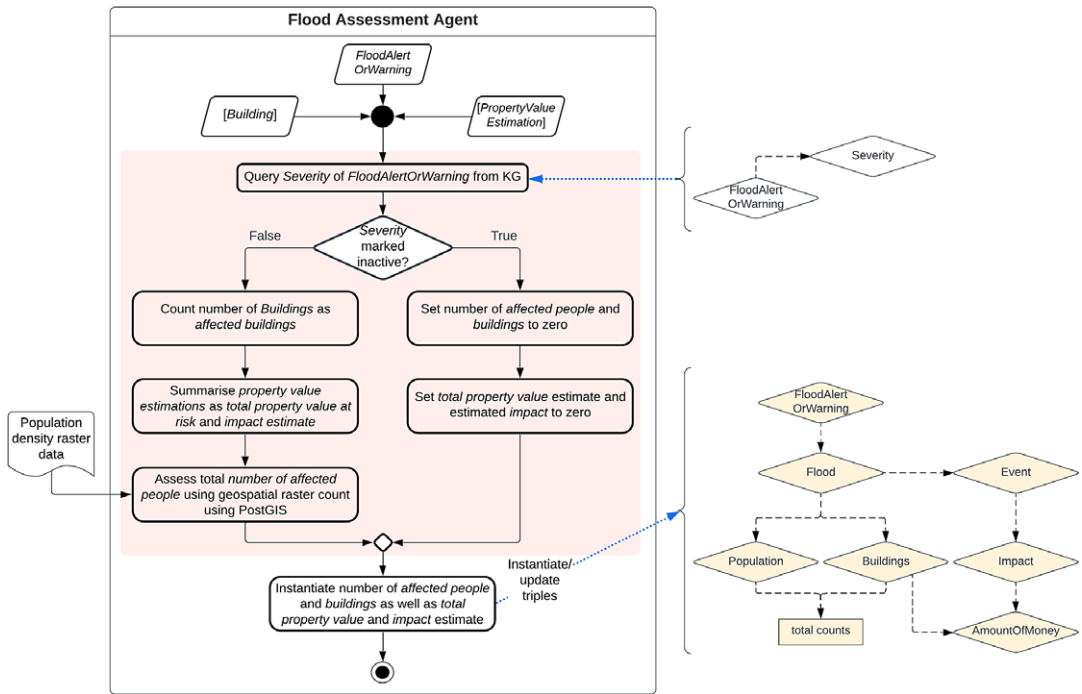


Figure 15. Flood Assessment Agent. The Flood Assessment Agent uses the DIF to assess potential impacts of (anticipated) floods with regards to (1) number of people at risk, (2) number of buildings at risk, and (3) estimated building stock value at risk. The agent is implemented as asynchronous derivation agent which monitors the instantiated information within a specified triple store namespace at predefined frequency. Pure input instances required to evaluate the impacts of a flood warning are marked up as part of the instantiation of new flood alerts and warnings (see Figure 13). A more detailed pseudocode example for the section highlighted in red is provided in Supplementary Material SI.2.

with all actual input instances via an `isDerivedFrom` relationship. A detailed UML diagram of the markup method is provided in Figure 14, with the key logic described below:

For newly instantiated `FloodAlertOrWarnings`, the agent retrieves a list of all `Buildings` within the flood area by using geospatial querying to assess which building footprints fall within the `ArealExtentPolygon` of the flood. Subsequently, for each of these buildings, the agent retrieves the instantiated `PropertyValueEstimation` instance if available. If not, it retrieves the corresponding derivation instance. Finally, an asynchronous derivation is instantiated to connect the flood assessment derivation instance with all potentially affected `Building` IRIs, their `PropertyValueEstimation` IRIs, and the respective `FloodAlertOrWarning` instance. Any potentially not yet instantiated timestamps for pure inputs are added, and an initial assessment is requested, which will be executed the next time the *Flood Assessment Agent* monitors the triple store.

For already instantiated but updated `FloodAlertOrWarnings`, the timestamp of the flood warning instance is updated and a derivation update for the existing derivation IRI is requested. As ceased flood warnings are deleted from TWA, including their derivation markup, no handling is required for lifted warnings.

3.3.13. Flood Assessment Agent

The *Flood Assessment Agent* is a derivation agent to estimate the number of people and buildings at risk of flooding as well as the total monetary value of the potentially affected building stock. The required input data for each flood assessment comprises a collection of `Building` and `PropertyValueEstimation`

instances and the respective `FloodAlertOrWarning` instance itself. A detailed UML diagram is provided in Figure 15, with key agent logic described below:

Initially, the agent verifies whether the marked-up input instances are suitable to perform a flood impact assessment, that is, that exactly one `FloodAlertOrWarning` instance is provided and the set of `PropertyValueEstimation` instances does not exceed the number of related `Building` instances. Missing property value estimations for some buildings as well as completely absent building and value estimation inputs are allowed (i.e., however, would result in an impact assessment of zero).

Subsequently, the `Severity` of the `FloodAlertOrWarning` is retrieved. For inactive flood warnings, all potential impacts are set to zero. Otherwise, the impacts are assessed: (1) The number of people affected is determined by conducting a geospatial count over the population density raster data within the boundary of the `ArealExtentPolygon` associated with the `FloodAlertOrWarning` of interest. (2) The number of buildings at risk is assessed by summing up all `Building` IRIs that are marked up as potentially affected derivation inputs. And (3) the total monetary value at risk is estimated by retrieving and summing up the property market values from marked-up `PropertyValueEstimation` IRIs. Finally, all potential flood impacts are instantiated according to `OntoFlood` as derivation outputs. While this assessment logic is intentionally simplistic for this proof-of-concept and concentrates solely on tangible property loss when evaluating economic impact, it showcases the framework's general capabilities for automated impact assessments; however, the technological capability behind the assessment is highly adaptable and supports future refinements as necessary.

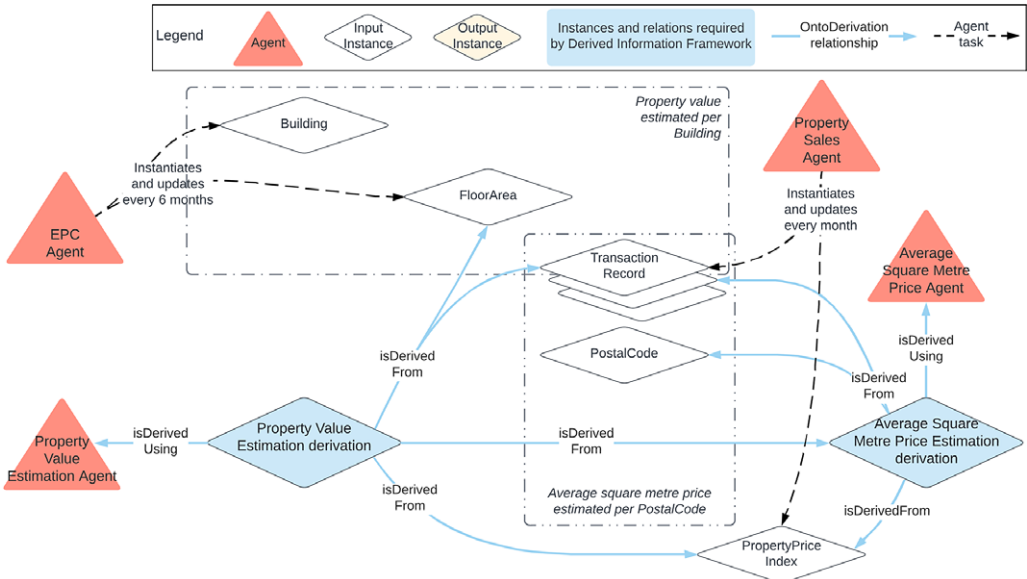
3.4. Derived information cascade

The automated flood impact assessment is based on the interplay of three autonomous derivation agents, namely the *Flood Assessment Agent*, the *Property Value Estimation Agent*, and the *Average Square Meter Price Agent*. In reverse order, these agents compute outputs which in turn act as input to the previously listed agent(s), resulting in a sequence of agent interactions to assess the potential impacts of a flood alert or warning.

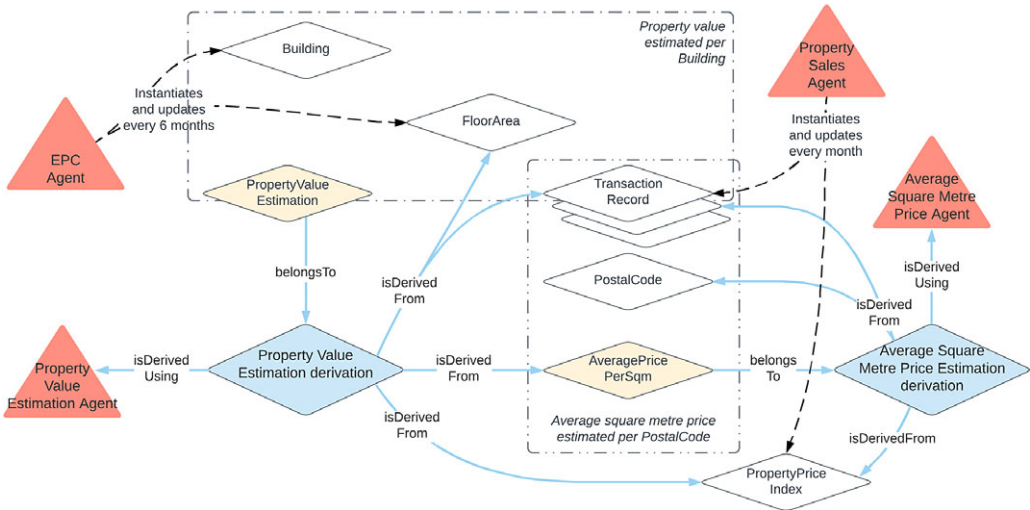
The `AveragePricePerSqM` is assessed on the postal code level and depends on the following inputs: (1) the `PostalCode` for which to assess the average price, (2) a list of previous property sales `TransactionRecords` in the applicable area, and (3) the representative `PropertyPriceIndex` for that postal code. The `PropertyValueEstimation` is assessed for each individual building and can either be derived based on (1) the latest `TransactionRecord` for that property (if available), and (2) again the representative `PropertyPriceIndex` or (3) the `FloorArea` of the given property, and (4) the representative `AveragePricePerSqM` for its location. As depicted in Figure 16, this chains together both the *Property Value Estimation Agent* and the *Average Square Meter Price Agent* to estimate the current market value of any given property.

The relevant real-world input data is assimilated on a monthly basis by both the *EPC Agent* and *Property Sales Agent*. While the initial derivation markup for both derivations is conducted as part of the latter one (see Figure SI.3 in the [Supplementary Material](#)), both agents ensure to update attached timestamps whenever they update an instantiated piece of information. There are two options to initially mark up the chained derivations: either (1) solely instantiate the derivation markup or (2) instantiate the derivation markup and request an immediate instantiation of newly derived information. In the first case, the initial derivation assessment will only be conducted once the required outputs are requested by a user or agent (i.e., at the next scheduled execution of the asynchronous derivation agent or upon HTTP request). Until then, the markup looks like depicted in Figure 16a). The latter case (depicted in Figure 16b)) immediately triggers an initial derivation assessment and instantiates the respective outputs in the KG (including re-connecting the derivation markup). This scenario also reflects situations where derived information is found outdated and re-evaluated based on updated pure inputs.

As both the average price per square meter and the property market value assessments are relatively fast and computationally inexpensive, both derivations are instantiated subsequently as synchronous derivations for new information to immediately compute and instantiate the newly derived information



(a) Derivation markup at creation. In case the AveragePricePerSqM is not yet instantiated, the PropertyValueEstimation derivation can be instantiated with the upstream derivation as input. The upstream derivation is automatically evaluated and instantiated once the PropertyValueEstimation is requested.



(b) Derivation markup after instantiation. After initial evaluation and instantiation of the AveragePricePerSqM, the PropertyValueEstimation derivation is automatically re-connected to it instead of its derivation instance. Assessing the property value will now only trigger an update of the upstream derivation if the instantiated AveragePricePerSqM is outdated, otherwise, it remains unaltered.

Figure 16. Derivation markup for PropertyValueEstimation. The estimated market value of any property depends on (i.e., isDerivedFrom) the FloorArea and latest available Transaction-Record of the property as well as the AveragePricePerSqM and the PropertyPriceIndex of the associated postal code and administrative district, respectively (details in Figure SI.6 in the Supplementary Material). As the AveragePricePerSqM is a derived quantity itself, there are two potential scenarios as detailed in (a,b).

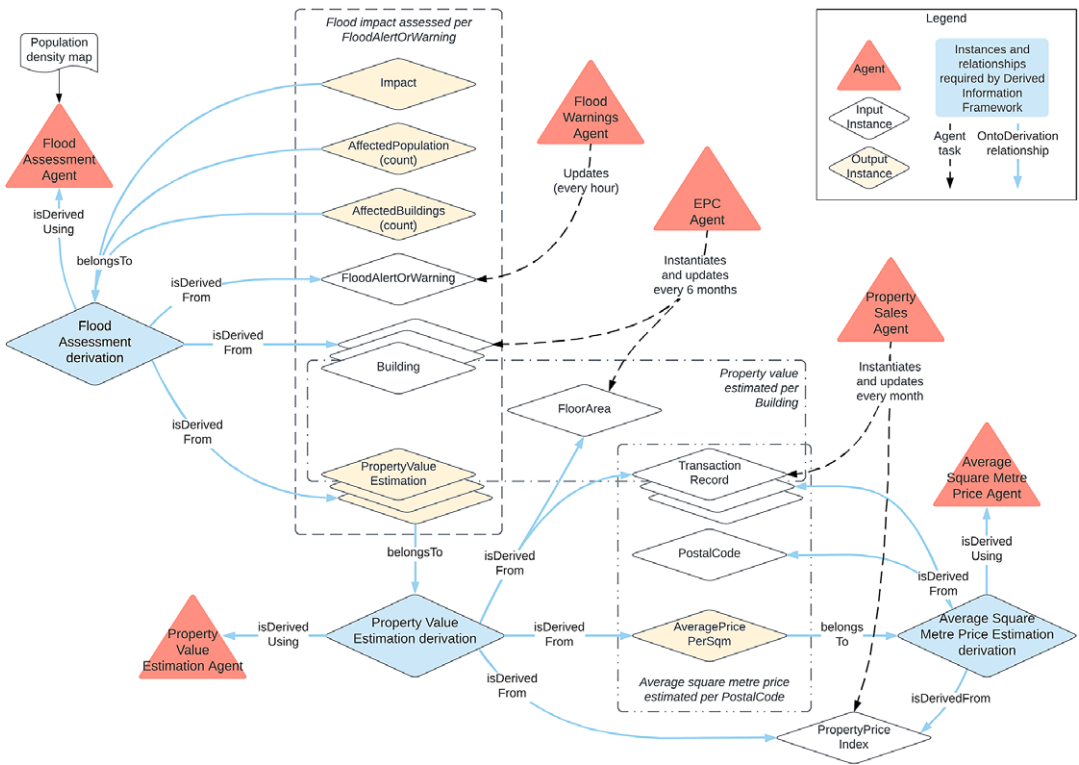


Figure 17. Flood assessment markup (instantiated). The potential impacts of a flood alert or warning (i.e., number of people, buildings, and total property value at risk) isDerivedFrom the FloodAlertOrWarning instance itself as well as all Buildings and respective PropertyValueEstimations within the associated flood area. Depending on the status of the property value estimation derivation, a new flood assessment can trigger a cascade of up to three derivation agents in sequence: Flood Assessment Agent requiring input instances to be updated by the Property Value Estimation Agent, which in turn relies on outputs of the Average Square Meter Price Agent.

(i.e., case 2). Hence, the AveragePricePerSqm shall already be instantiated when creating the PropertyValueEstimation markup, making the depicted case in Figure 16b the default scenario. Since all required inputs can be assumed to be instantiated, this also eases the further markup for the overall flood impact assessment (depicted in Figure 17).

The flood impacts assessment is evaluated for each instantiated FloodAlertOrWarning and depends on (1) the FloodAlertOrWarning itself as well as (2) a collection of Building instances and (3) associated PropertyValueEstimation. One single flood warning/alert can stretch across multiple postal codes with thousands of buildings. Hence, the potential impact assessment is expected to take some time and is thus marked up as asynchronous derivation. The Flood Assessment Agent monitors the status of instantiated derivations with a predefined frequency to detect potentially outdated inputs which could affect the impact of a flood over the lifetime of the warning. Those inputs include both the Severity and ArealExtent of the alert/warning as well as newly instantiated or updated property information, such as FloorArea or sales TransactionRecords, for potentially affected buildings or the PropertyPriceIndex for impacted administrative districts. When an update is requested, the Flood Assessment Agent uses a sequence of calls to the Average Square Meter Price Agent and Property Value Estimation Agent to update relevant derived information and generate a current flood impact estimate.

4. Implementation and deployment

The overall implementation follows a containerized approach offering a flexible, scalable, and platform-independent way to deploy and manage individual agents. Each agent is implemented as an individual Docker container and deployed to an overarching Docker stack enabling inter-container communication. All Docker images are published on GitHub's package repository to ease deployment and reproducibility. The design supports both (remote) server and local deployment to facilitate a truly distributed knowledge graph with decentralized hosting of data and computational capabilities. To ensure reliable service availability, the entire stack for this proof-of-concept is deployed remotely in the cloud.

Within each stack, all agents have access to the same triple store (i.e., Blazegraph) as well as an instance of PostGIS, GeoServer, and Ontop to overcome limitations in triple-store native handling of geospatial information (see Sections 2.6 and 3.2.1). The proposed approach stores geospatial information in PostGIS (i.e., using standard libraries such as GDAL to perform data ingestion), while providing a corresponding OBDA mapping to allow access to the data according to the corresponding ontology using regular SPARQL syntax. Furthermore, PostGIS can be directly linked to GeoServer to support streaming as well as (dynamic) styling of relevant geospatial information to the World Avatar Visualization Framework (VF). Moreover, PostGIS is used as relational database by the *TimeSeriesClient* to store actual time series data.

Once deployed, the developed agent framework continuously integrates sensor data (i.e., weather, river levels, air quality) and flood warning information for the entire UK, along with building characteristics and property market data specific to the vicinity of King's Lynn. It is noteworthy that not all data available from the previously referenced APIs gets instantiated; rather, only the pertinent aspects captured by the developed ontologies and relevant to the discussed use case are considered. After several months of uptime, the instantiated dataset comprises approximately 10.5 million triples, accompanied by around 13,000 time series tables as well as 12,000 geospatial entities (i.e., building footprints and flood areas) in PostGIS. To date, we have not encountered any storage issues, and the system operates seamlessly on a standard Digital Ocean machine.

The VF is used as uniform visualization interface for all instantiated data. It is implemented in TypeScript and compiled into a single minified JS file (accompanied by a single minified CSS file). New visualizations can be created as new Docker containers by providing a few configuration files to import the remotely hosted VF library (along with other dependencies) as well as including further data and HTML elements as required. The used mapping provider Mapbox supports the visualization of 2D data (with the option to extrude 2D polygons into basic 3D polyhedrons) from local files or from WMS endpoints (e.g., served via GeoServer). Both meta and time series data for features of interest are queried directly from TWA and displayed on the side panel on the fly upon request.

4.1. Agent deployment

The overall agent deployment consists of two phases: (1) a rather manual pre-processing of geospatial building data together with the initial building instantiation, followed by (2) the ongoing autonomous assimilation of further building data as well as additional dynamic data feeds.

During the initial building instantiation, several OS datasets need to be downloaded, namely, the BHA containing high-resolution building footprints and building height information, the OpenMap Local with coarser building footprints data, the Code-Point with Polygon containing postcode polygons, and the Digital Terrain Model DTM5 with detailed terrain raster data in 5 m resolution. These datasets are processed using QGIS (QGIS Development Team, 2021) to create a single consolidated shapefile containing all relevant building data: Both the BHA and OpenMap Local building footprints are merged to maximize building coverage as some buildings are only available in either of the datasets. In case of overlaps, the BHA data is used. Additionally, corresponding postcodes and building elevations are extracted from Code-Point polygons and DTM5, respectively, and added to each building node. Subsequently, FME (Safe Software Inc., 2022) is used to convert the consolidated shapefile into CityGML format as required by the *Building Importer Agent*. This includes assigning the postcodes as

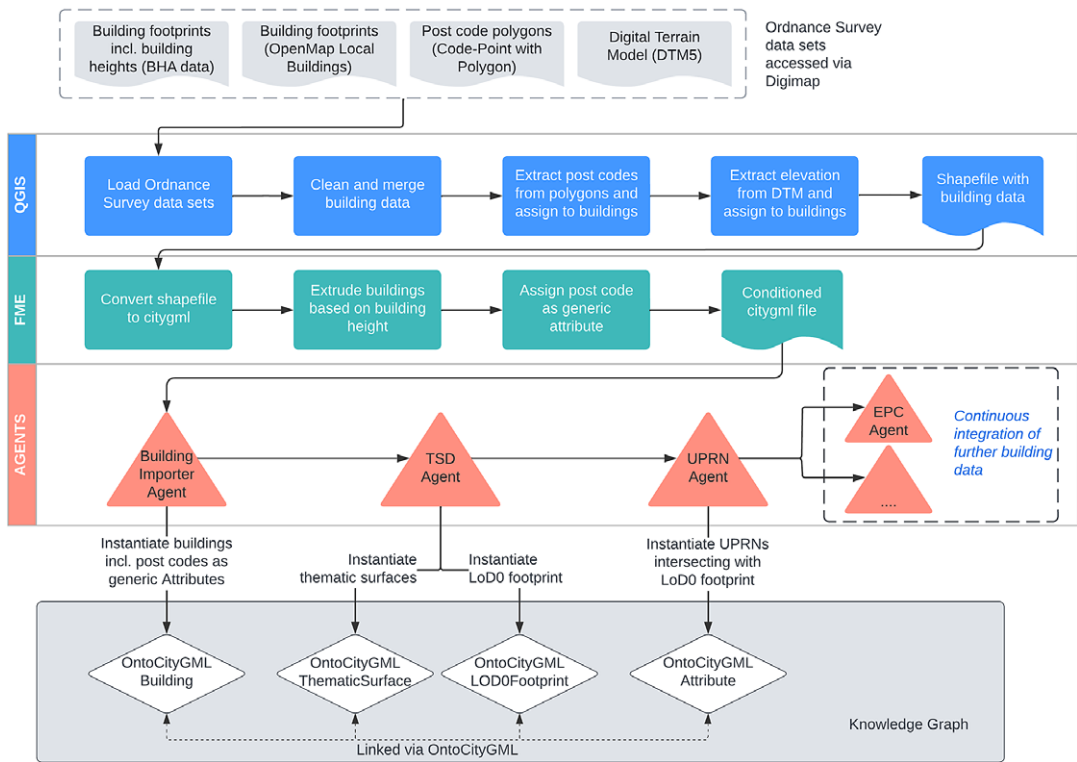


Figure 18. Building instantiation workflow. The instantiation currently still requires some manual steps, that is, geospatial processing using QGIS and FME. After instantiating the buildings using the Building Importer Agent, also the Thematic Surface Discovery and UPRN agents need to be invoked manually to ensure that geospatially represented buildings have UPRNs attached (if available). Further building data instantiation (i.e., EPC Agent, Property Sales Agent) happens automatically once the respective agents are deployed.

Generic Attributes and extruding each building based on its height information. Finally, the created CityGML file is provided to the *Building Importer Agent* to be instantiated according to OntoCityGML. The *TSD Agent* is then called to identify and instantiate both thematic surfaces and LoD0 building footprints before the *UPRN Agent* attaches UPRN information from the OS Feature API to all buildings based on their LoD0 footprints. This marks the end of the manual building data instantiation and the entire workflow is schematically outlined in Figure 18.

After the initial geospatial building instantiation, all remaining agents are deployed to the Docker stack. The *EPC Agent*, which initializes non-OntoCityGML building representations, needs to be deployed first, before any additional building data can be instantiated. While all OntoEMS instantiation agents can populate individual triple store namespaces, all EPC, property sales, and flood warnings data need to get instantiated into the same namespace to enable the derivation agents to monitor and detect relevant updates. Start-up of the *Met Office*, *Air Quality*, and *River Level Agents* automatically registers a recurring background task to assimilate latest data once per day. Similarly, the *EPC* and *Property Sales Agent* ingest latest building-related data on a monthly basis, while current flood warnings are assimilated hourly by the *Flood Warnings Agent*.

4.2. Consolidated visualization

Based on ontologies to represent both data and inherent knowledge, TWA provides a semantic ecosystem to interact with a variety of data from previously isolated sources in an aligned format. TWA creates a rich



Figure 19. Consolidated visualization. A built-in Visualization Framework provides a uniform interface to retrieve and visualize data from TWA. It creates an aligned visualization of previously isolated data sources side by side to foster fact-based decision-making and enable insights across domains.

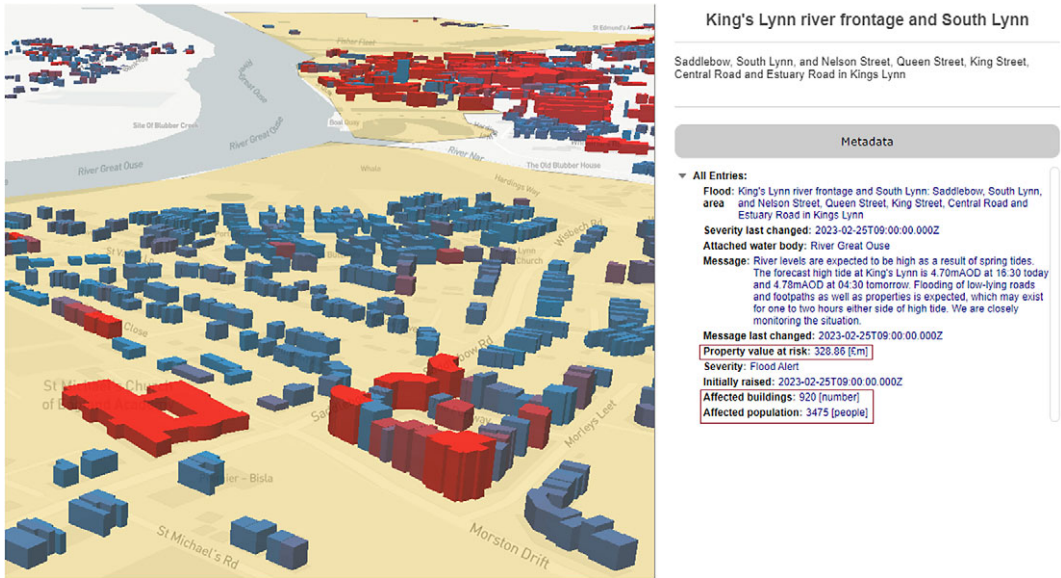
knowledge representation by combining both rather static (i.e., buildings) and dynamic (i.e., environmental measurements and flood warnings) data in one single system, enriching individual pieces of information with meaning and providing additional context.

The World Avatar Visualization Framework provides a web-based interface to visualize data from TWA. Figure 19 shows an example visualization of the current use case in the smart city context, combining consolidated building information with environmental measurements and forecasts from various data providers. It offers a mutual and aligned visualization of previously isolated weather, air quality, and river level data, including information about associated measuring stations as well as corresponding time series data. This empowers citizens to see related information (e.g., river level readings as well as current and projected rainfall data or air pollutant concentration measurements and projected wind conditions) in one single system, as compared to various independent locations or websites. The available building data provides detailed information about key construction characteristics, energy performance, usage types, previous and current market value estimates, and so forth, and is maintained by a set of continuously running input agents to keep it current in time. The integrated map-based visualization enables both geospatial and time series analyses, for example, to understand building usages in a certain area.

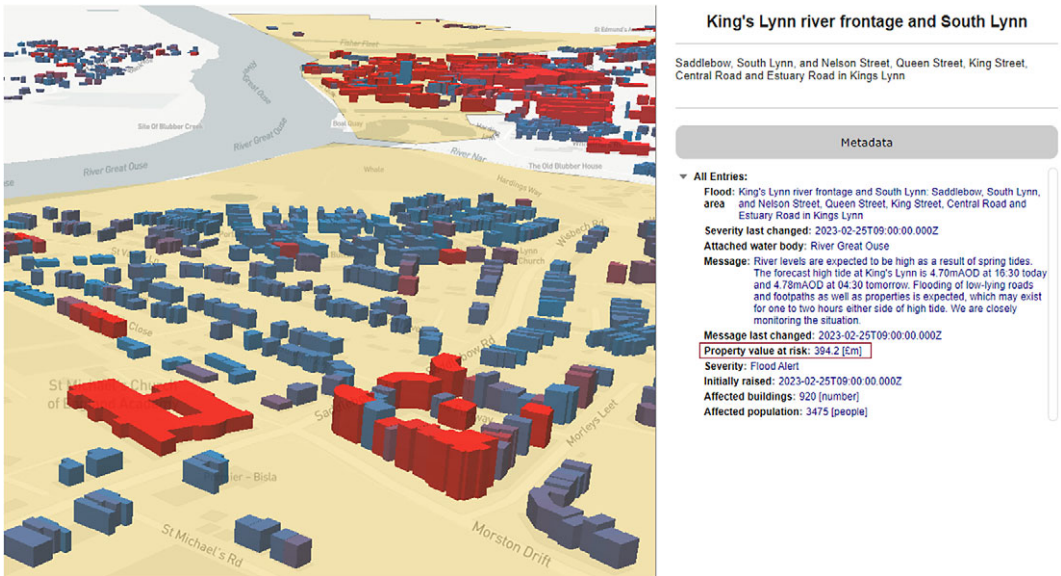
4.3. Cross-domain flood impact assessment

The *Flood Assessment Agent* continuously monitors a predefined triple store at a specified frequency and automatically re-evaluates any outdated flood impact derivation instances. Figure 20 illustrates this intrinsic dynamism by showing an initial flood impact assessment as well as an updated one after an increase in property prices side by side:

Initially, the *Flood Warnings Instantiation Agent* ingests a new flood alert into TWA and marks it up as an asynchronous derivation. The next time the derivation agent processes all requested derivations, it estimates the potential flood impacts of the raised alert. This initial flood impact assessment indicates that the issued alert may affect 3475 people living in the associated flood area, with 920 buildings at risk,



(a) Initial assessment. The raised flood alert is expected to put 3475 people at risk and could affect 920 buildings with an estimated market value of £328.9m.



(b) Updated assessment. The change in property price index triggers an automatic re-assessment of total property value at risk, which is now estimated with £394.2m.

Figure 20. Automated flood assessment. The automated re-evaluation of potential flood impacts is depicted for two subsequent assessments of the same flood alert with a simulated property price index hike of +20% in between. (a) Depicts the initial assessment of properties at risk. During a subsequent evaluation by the Flood Assessment Agent, the updated *PropertyPriceIndex* is identified by the framework, triggering an update of all *PropertyValueEstimations* before computing the impact estimate. Corresponding changes to the overall property value at risk are shown in (b). Property market value changes can be seen from updated colors of individual buildings (very mildly though) or easier from the side panel in aggregated form.

valued at £328.9 m (Figure 20a). The building colors in the figure represent property market value estimates, with red indicating high and blue indicating low property prices.

After the initial assessment, a 20% increase in the property price index is instantiated to simulate a rise in the value of residential properties. This update results in all property value estimations becoming outdated and, accordingly, also affects the dependent flood assessment derivation instance. The *Flood Assessment Agent* automatically detects and updates the affected flood assessment derivation instance the next time it monitors the triple store. The new evaluation still shows that 3475 people and 920 buildings are at risk; however, the total property value has increased to £394.2 m (Figure 20b). While this simulated index hike is for illustrative purposes only, it demonstrates the system's dynamic capabilities., for example, to automatically reflect adjustments in the monthly UKHPI in any subsequent flood assessment. Similarly, updated flood alert/warning data (i.e., change in severity, warning message details), newly instantiated buildings, or outdated property value instances would result in an automatic re-assessment. Whenever a flood alert is marked as inactive at the flood-monitoring API, potential impacts are assessed as zero and all assessment triples are ultimately deleted from TWA once the flood alert vanishes from the API. This dynamism ensures that TWA is inherently evolving with time to provide up to date insights into current flood situations. To further illustrate this aspect, additional screenshots of an actual flood warning situation and its evolution throughout a day are provided in [Supplementary Material SI.4](#).

The proposed framework is well suited to provide actionable insights during disaster situations. The performance for various scales of flooding events has been tested on a virtual machine (4 Intel Xeon Gold 6248 CPUs with 2.50 GHz and 32 GB DIMM RAM) hosted on Digital Ocean. Tests have been conducted for three flood warnings, involving around 400, 900, and 5100 buildings, respectively. Processing times have been measured from the moment the framework picks up the derivation for flood impact (re-) assessment to the timestamp when all information is up-to-date. Each test has been repeated at least three times to obtain meaningful results. As expected, the evaluation time scales linearly with the number of affected buildings, with the current implementation processing approximately 3–10 buildings per second (i.e., 10 buildings per second if all underlying information is still up to date and 3 buildings per second if all underlying information requires updating). It should be noted, however, that the actual processing time is contingent upon the specific use case and is likely to evolve with more sophisticated impact estimation methods. Moreover, the framework is designed to be scalable and multiple *Flood Assessment Agents* could be deployed simultaneously to enhance performance as required.

Limitations: Despite providing this proof-of-concept for a semantic software agent-based automated flood assessment, several limitations of the work shall be emphasized: Firstly, the assessment does not include all buildings, but only considers the actually instantiated share. As the building instantiation workflow is based on EPC data, only buildings with available EPC information are part of the analysis; however, it has been observed, that EPC data is only available for slightly above 50% of the buildings in the vicinity of King's Lynn (Hofmeister et al., 2023), which results in an expected underprediction of monetary impact from flood warnings. Secondly, the current approach of estimating property market values is intentionally simplistic. While this is sufficient for a proof-of-concept, it overlooks several crucial factors which can significantly affect property prices, such as usage classification (i.e., domestic vs. non-domestic), retrofitting, and micro-location, just to name a few. Hence, the results shall only be interpreted as rough estimates and the evaluation approach shall be refined in the future to allow for a more elaborate assessment. Thirdly, the current impact assessment is limited to the vulnerable share of population as well as the number and value of potentially affected buildings, while other aspect of the built environment (e.g., roads or further network assets) are currently neglected. Lastly, it should be noted that the process of enriching instantiated buildings with corresponding sales transactions relies on fuzzy address matching between EPC and HM Land Registry data, as UPRNs are not provided in Land Registry's transaction record data. Unfortunately, free text address information are not always provided in an aligned fashion. For instance, mismatches in address lines, the use of building names instead of numbers, and differences in granularity with regards to building (sub-)numbers have been observed. Such discrepancies can result in erroneous instantiations of previous sales transactions, either by attaching a transaction record to the wrong building instance (i.e., another building with higher confidence score of fuzzy match) or not instantiating a transaction record

for a building instance at all (i.e., required confidence score of fuzzy match not reached). However, it needs to be noted that various scoring methods and confidence scores have been screened to minimize this problem.

5. Conclusions

This work proposes a collection of agents that is able to synthesize multiple previously isolated data feeds into an automated impact assessment for imminent flood hazards. Three connected ontologies are developed to dynamically instantiate a semantically rich representation of buildings, various environmental observations, and flooding-related information. An ecosystem of autonomous input agents has been developed and deployed to continuously assimilate multiple data feeds from disconnected sources, including both rather static and near real-time data, to ensure that the World Avatar remains current in time. The capabilities to provide cross-domain insights and decision support for smart city and disaster management are demonstrated by bringing together live data about potential flood events with information about buildings as well as further environmental observations in their vicinity. A unified visualization interface is provided to interact with and analyze the instantiated data. Beyond the visualization, this work provides access to all incorporated publicly available data sources via one single SPARQL endpoint, using aligned knowledge models based on the developed ontologies.

Furthermore, the agent-based autonomous cascading of information enabled by the derived information framework is demonstrated: a re-assessment of potential flood impacts is triggered and automatically executed whenever a new flood alert or warning is instantiated or instantiated information for any relevant input is updated. The impact of potential flooding events is assessed with regards to the number of likely affected people as well as the number and estimated market value of buildings at risk. Together with the ecosystem of continuously running input agents, this ensures that up-to-date insights into potential flooding situations and their consequences are available at all times. While the currently implemented assessment logic is intentionally simplistic and concentrates solely on tangible property loss for this proof-of-concept, it showcases the framework's general effectiveness for automated impact assessments. The primary contribution of this work is the technological capability behind the assessment, with the underlying evaluation methodology being adaptable for refinement as necessary and the demonstrated information cascading infrastructure providing a promising approach to fully automate smart city workflows and foster interoperability. In disaster management, this is expected to improve timely decision-making and initiate proactive measures to mitigate adverse effects of imminent flood hazards, ultimately contributing to the safety and well-being of vulnerable population and infrastructure. The proposed system is easily deployable to any other city in the UK, and this work is currently ongoing.

Nomenclature

ABox	Assertional Component (of an ontology)
API	Application Programming Interface
BHA	Building Height Attribute (OS Premium dataset)
CEA	City Energy Analyst
DIF	Derived Information Framework
EA	Environment Agency
ENVO	Environmental Ontology
EPC	Energy performance certificate
GEMET	General Multilingual Environmental Thesaurus
GeoSPARQL	Geographic Query Language for RDF Data
IRI	Internationalized Resource Identifier
KG	Knowledge Graph
LoD	Level of Detail
MEMOn	Modular Environmental Monitoring (ontology)

OBDA	Ontology-Based Data Access
OGC	Open Geospatial Consortium
ONS	Office for National Statistics
OS	Ordnance Survey
OWL	Web Ontology Language
PPD	(HM Land Registry's) Price Paid Data
RDF	Resource Description Framework
SAREF	Smart Appliances REference (ontology)
SOSA	Sensor, Observation, Sample, and Actuator (ontology)
SPARQL	SPARQL Protocol and RDF Query Language
SSN	Semantic Sensor Network (ontology)
SWEET	Semantic Web for Earth and Environmental Terminology
TBox	Terminological Component (of an ontology)
TWA	The World Avatar
UK-AIR	UK Air Information Resource
UKHPI	UK House Price Index
UML	Unified Modeling Language
UPRN	Unique Property Reference Number
VF	Visualization Framework
W3C	World Wide Web Consortium

Supplementary material. The supplementary material for this article can be found at <http://doi.org/10.1017/dce.2024.11>.

Data availability statement. No new or proprietary data were created or analyzed in this study. All used data is retrievable from public APIs and manipulated using software agents publicly available on GitHub. All the codes developed are available on The World Avatar GitHub repository: <https://github.com/cambridge-cares/TheWorldAvatar>: (1) The latest versions of all developed ontologies are available in the *JPS_Ontology* sub-directory; (2) The source code of all referenced agents is available in the *Agents* sub-directory; and (3) Detailed deployment instructions to reproduce the work are available in the *Kings Lynn Project* sub-directory.

Author contribution. Conceptualization: M.H., J.B., S.M., K.F.L., J.A., M.K.; Data curation: M.H.; Funding acquisition: M.K.; Investigation: M.H.; Methodology: M.H., J.B., S.M., K.F.L., F.F.; Project administration: M.H.; Resources: G.B., S.M., J.A., M.K.; Software: M.H., J.B., G.B., K.F.L., M.A., S.G.; Supervision: S.M., J.A., M.K.; Validation: M.H., J.B.; Visualization: M.H., M.H.; Writing—original draft: M.H.; Writing—review and editing: M.H., J.B., S.M., F.F., M.K.; All authors approved the final submitted draft.

Funding statement. This research was supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme. A part of this study has been undertaken in the context of DOME 4.0 project, which has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No 953163. M.H. acknowledges financial support provided by the Cambridge Trust and CMCL. J.B. acknowledges the financial support provided by CSC Cambridge International Scholarship from Cambridge Trust and China Scholarship Council. M.K. gratefully acknowledges the support of the Alexander von Humboldt Foundation.

Attribution statement. This work contains OS data © Crown copyright and database rights 2022 (100025252). Furthermore, it contains HM Land Registry data © Crown copyright and database right 2020 as well as public sector information licensed under the Open Government License v3.0. Lastly, this showcase uses Environment Agency flood and river level data from the real-time data API (Beta).

Competing interest. The authors declare none.

References

- Agresta A, Fattoruso G, Pasanisi F, Tebano C, De Vito S and Di Francia G (2014) An ontology framework for flooding forecasting. In Murgante B, Misra S, Rocha AMAC, Torre C, Rocha JG, Falcão MI, Taniar D, Apduhan BO and Gervasi O (eds.), *Computational Science and Its Applications – ICCSA*, Vol. 2014. Cham: Springer International Publishing, pp. 417–428.
- Akroyd J, Harper Z, Soutar D, Farazi F, Bhawe A, Mosbach S and Kraft M (2022) Universal digital twin: Land use. *Data-Centric Engineering* 3, e3.

- Akroyd J, Mosbach S, Bhawe A and Kraft M** (2021) Universal digital twin – A dynamic knowledge graph. *Data-Centric Engineering* 2, e14.
- Allemang D and Hendler J** (2011) *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*, 2nd Edn. San Francisco, CA: Morgan Kaufmann.
- Aranda CB, Corby O, Das S, Feigenbaum L, Gearon P, Glimm B, Harris S, Hawke S, Herman I, Humfrey N, Michaelis N, Ogbuji C, Perry M, Passant A, Polleres A, Prud'hommeaux E, Seaborne A and Williams GT** (2013) SPARQL 1.1 Overview, W3C Recommendation 21 March 2013. World Wide Web Consortium (W3C). Available at <https://www.w3.org/TR/sparql11-overview/> (accessed April 2023).
- Atherton J, Xie W, Aditya LK, Zhou X, Karmakar G, Akroyd J, Mosbach S, Lim MQ and Kraft M** (2021) How does a carbon tax affect Britain's power generation composition? *Applied Energy* 298, 117117.
- Baader F, Calvanese D, McGuinness DL, Nardi D and Patel-Schneider PF** (eds.) (2007) *The Description Logic Handbook*, 2nd Edn. Cambridge: Cambridge University Press.
- Bai J, Cao L, Mosbach S, Akroyd J, Lapkin AA and Kraft M** (2022) From platform to knowledge graph: Evolution of laboratory automation. *JACS Au* 2(2), 292–309.
- Bai J, Lee KF, Hofmeister M, Mosbach S, Akroyd J and Kraft M** (2024) A derived information framework for a dynamic knowledge graph and its application to smart cities. *Future Generation Computer Systems* 152, 112–126.
- Barker J and Macleod C** (2019) Development of a national-scale real-time twitter data mining pipeline for social geodata on the potential impacts of flooding on communities. *Environmental Modelling & Software* 115, 213–227.
- Berners-Lee T** (2006) Linked Data – Design Issues. Available at <http://www.w3.org/DesignIssues/LinkedData.html> (accessed April 2023).
- Berners-Lee T, Hendler J and Lassila O** (2001) The semantic web. *Scientific American* 284(5), 28–37.
- Bizer C, Heath T and Berners-Lee T** (2011) Linked data: The story so far. In Sheth A (ed.), *Semantic Services, Interoperability and Web Applications: Emerging Concepts*. Hershey, PA: IGI Global.
- Blazegraph** (2020a) Blazegraph™DB. Available at <https://blazegraph.com> (accessed February 2023).
- Blazegraph** (2020b) Geospatial Support in Blazegraph. Available at <https://github.com/blazegraph/database/wiki/GeoSpatial> (accessed March 2023).
- Botoeva E, Calvanese D, Cogrel B, Rezk M and Xiao G** (2016) OBDA beyond relational DBs: A study for MongoDB. In Lenzerini M and Peñaloza R (eds.), *Proceedings of the 29th International Workshop on Description Logics (DL 2016), Volume 1577 of CEUR Workshop Proceedings*. Cape Town, South Africa: CEUR-WS. Available at <http://ceur-ws.org> (accessed November 2023).
- Brick Schema** (2022) Brick Ontology. Available at <https://brickschema.org/ontology> (accessed April 2023).
- Buttigieg P, Mungall C, Rueda C, McGibney L, Cox S, Duerr R, Fils D, Graybeal J, Huffer B, Narock T, Ramachandran B, Soiland-Reyes S and Whitehead B** (2018) SWEET ontology suite v3.0.0: Development, alignments and use cases. In *American Association of Geographers 2018 (AAG 2018)*. New Orleans, LA: SIP. Available at <https://github.com/ESIPFed/sweet> (accessed February 2023).
- Buttigieg PL, Pafilis E, Lewis SE, Schildhauer MP, Walls RL and Mungall CJ** (2016) The environment ontology in 2016: Bridging domains with increased scope, semantic density, and interoperation. *Journal of Biomedical Semantics* 7, 57. Available at <https://raw.githubusercontent.com/EnvironmentOntology/envo/master/envo.owl> (accessed February 2023).
- Centre Universitaire d'Informatique at University of Geneva** (2012) Available at <http://cui.unige.ch/isi/onto/citygml2.0.owl> (accessed November 2023).
- Chadzynski A, Krdzavac N, Farazi F, Lim MQ, Li S, Grisiute A, Herthogs P, von Richthofen A, Cairns S and Kraft M** (2021) Semantic 3D city database – An enabler for a dynamic geospatial knowledge graph. *Energy and AI* 6, 100106.
- Chadzynski A, Li S, Grisiute A, Chua J, Hofmeister M, Yan J, Tai HY, Lloyd E, Tsai YK, Agarwal M, Akroyd J, Herthogs P and Kraft M** (2023a) Semantic 3D city interfaces—Intelligent interactions on dynamic geospatial knowledge graphs. *Data-Centric Engineering* 4, e20.
- Chadzynski A, Li S, Grisiute A, Farazi F, Lindberg C, Mosbach S, Herthogs P and Kraft M** (2022) Semantic 3D city agents—An intelligent automation for dynamic geospatial knowledge graphs. *Energy and AI* 8, 100137.
- Chadzynski A, Silvennoinen H, Grisiute A, Farazi F, Mosbach S, Raubal M, Herthogs P and Kraft M** (2023b) Semantic 3D City Inferences – Multi-domain Reasoning on Dynamic Geospatial Knowledge Graphs (submitted for publication). Available at <https://como.ceb.cam.ac.uk/preprints/303/> (accessed December 2023).
- Cohen A** (2020) Fuzzywuzzy: Fuzzy String Matching in Python. Available at <https://pypi.org/project/fuzzywuzzy/> (accessed March 2023).
- Cox S, Little C, Hobbs JR and Pan F** (2020) Time Ontology in OWL. Available at <https://www.w3.org/TR/owl-time/> (accessed March 2023).
- Cuenca J and Larrinaga F** (2019) DABGEO: Domain Analysis-Based Global Energy Ontology. Available at <https://innoweb.mondragon.edu/ontologies/dabgeo/> (accessed April 2023).
- Cuenca J, Larrinaga F and Curry E** (2020) DABGEO: A reusable and usable global energy ontology for the energy domain. *Journal of Web Semantics* 61–62, 100550.
- Data & Knowledge Group** (2019) Hermit OWL Reasoner. Available at <http://www.hermit-reasoner.com/> (accessed April 2023).
- Department for Environment, Food & Rural Affairs** (2023) UK-AIR Sensor Observation Service (Beta Release). Available at https://uk-air.defra.gov.uk/data/about_sos (accessed April 2023).

- Department for Levelling Up, Housing & Communities** (2022) Energy Performance of Buildings Data. Available at <https://epc.opendatacommunities.org/docs/api> (accessed April 2023).
- Devanand A, Karmakar G, Krdzavac N, Rigo-Mariani R, Foo EYS, Karimi IA and Kraft M** (2020) OntoPowSys: A power system ontology for cross domain interactions in an eco industrial park. *Energy and AI 1*, 100008.
- Dsouza A, Tempelmeier N, Yu R, Gottschalk S and Demidova E** (2021) WorldKG: A world-scale geographic knowledge graph. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*. New York: ACM. Available at <https://www.worldkg.org/> (accessed May 2023).
- Dutta R and Morshed A** (2013) Performance evaluation of south esk hydrological sensor web: Unsupervised machine learning and semantic linked data approach. *IEEE Sensors Journal 13*(10), 3806–3815.
- Eclipse Foundation** (2021) RDF4J Documentation – GeoSPARQL. Available at <https://rdf4j.org/documentation/programming/geosparql/> (accessed April 2023).
- EDINA Digimap Ordnance Survey Service** (2022) Available at <https://digimap.edina.ac.uk/> (accessed April 2023).
- Eibeck A, Chadzynski A, Lim MQ, Aditya K, Ong L, Devanand A, Karmakar G, Mosbach S, Lau R, Karimi IA, Foo EYS and Kraft M** (2020) A parallel world framework for scenario analysis in knowledge graphs. *Data-Centric Engineering 1*, e6.
- Environment Agency** (2021a) Hydrology Data API. Available at <https://environment.data.gov.uk/hydrology/doc/reference> (accessed February 2023).
- Environment Agency** (2021b) Real Time flood-monitoring API. Available at <https://environment.data.gov.uk/flood-monitoring/doc/reference> (accessed February 2023).
- ETSI** (2020a) SAREF Extension for Building. Available at <https://saref.etsi.org/saref4bldg/v1.1.2/> (accessed March 2023).
- ETSI** (2020b) SAREF: the Smart Applications REference Ontology. Available at <https://saref.etsi.org/core/> (accessed February 2023).
- European Environment Information and Observation Network** (2021) General Multilingual Environmental Thesaurus. Available at <https://www.eionet.europa.eu/gemet/en/themes/> (accessed February 2023).
- Farazi F, Akroyd J, Mosbach S, Buerger P, Nurkowski D, Salamanca M and Kraft M** (2020) OntoKin: An ontology for chemical kinetic reaction mechanisms. *Journal of Chemical Information and Modeling 60*(1), 108–120.
- Flood Forecasting Centre** (2017) FGS Public API. Available at <https://api.foursources.metoffice.gov.uk/docs/flood-guidance-statement-api-public#> (accessed February 2023).
- Gröger G, Kolbe TH, Nagel C and Häfele K-H** (2012) OGC City Geography Markup Language (CityGML) Encoding Standard, Version 2.0.0. Available at <http://www.opengeospatial.org/standards/citygml> (accessed March 2023).
- HM Land Registry** (2022a) HM Land Registry Open Data. Available at <https://landregistry.data.gov.uk/> (accessed April 2023).
- HM Land Registry** (2022b) Price Paid Linked Data. Available at <https://landregistry.data.gov.uk/app/root/doc/ppd> (accessed April 2023).
- HM Land Registry** (2022c) Price Paid Linked Data SPARQL Endpoint. Available at <http://landregistry.data.gov.uk/landregistry/query> (accessed April 2023).
- HM Land Registry** (2022d) UK House Price Index Linked Data. Available at <https://landregistry.data.gov.uk/app/ukhpi/doc> (accessed April 2023).
- Hofmeister M, Brownbridge G, Hillman M, Mosbach S, Akroyd J, Lee KF and Kraft M** (2023) Cross-Domain Flood Risk Assessment for Smart Cities using Dynamic Knowledge Graphs (submitted for publication). Available at <https://como.ceb.cam.ac.uk/preprints/308/>.
- Janowicz K, Haller A, Cox SJ, Le Phuoc D and Lefrançois M** (2019) SOSA: A lightweight ontology for sensors, observations, samples, and actuators. *Journal of Web Semantics 56*, 1–10.
- Johnson JM, Narock T, Singh-Mohudpur J, Fils D, Clarke KC, Saksena S, Shepherd A, Arumugam S and Yeghiazarian L** (2022) Knowledge graphs to support real-time flood impact evaluation. *AI Magazine 43*(1), 40–45.
- Jovanovic M, Homburg T and Spasić M** (2021) A geosparql compliance benchmark. *ISPRS International Journal of Geo-Information 10*(7), 487.
- Karalis N, Mandilaras G and Koubarakis M** (2019) Extending the YAGO2 knowledge graph with precise geospatial knowledge. In *Lecture Notes in Computer Science*. New York: Springer International Publishing, pp. 181–197.
- Katsumi M** (2021) Building Ontology. Available at <http://ontology.eil.utoronto.ca/icity/Building/1.2/> (accessed April 2023).
- Khantong S, Sharif MNA and Mahmood AK** (2020) An ontology for sharing and managing information in disaster response: An illustrative case study of flood evacuation. *International Review of Applied Sciences and Engineering IRASE 11*, 22–33.
- Klien E, Lutz M and Kuhn W** (2006) Ontology-based discovery of geographic information services—An application in disaster management. *Computers, Environment and Urban Systems 30*(1), 102–123.
- Klyne G and Carroll JJ** (2004) Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation 10 February 2004. World Wide Web Consortium (W3C). Available at <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/> (accessed April 2023).
- Kollarits S, Wergles N, Siegel H, Liehr C, Kreuzer S, Torsoni D, Sulzenbacher U, Papez J, Mayer R, Plank C, Maurer L, Cimarosto S and Bukta E** (2009) MONITOR – An Ontological Basis for Risk Management. Technical Report, Heriot Watt University. Available at http://www.macs.hw.ac.uk/~yjc32/project/ref-ontology/ref-ontology%20examples/MONITOR_Base_Ontology_Report_1_0.pdf (accessed February 2023).
- Kondinski A, Bai J, Mosbach S, Akroyd J and Kraft M** (2023) Knowledge engineering in chemistry: From expert systems to agents of creation. *Accounts of Chemical Research 56*(2), 128–139.

- Lee KF, Hofmeister M, Mosbach S and Tsai YK** (2021) Time Series Ontology. Available at https://github.com/cambridge-cares/TheWorldAvatar/tree/main/JPS_Ontology/ontology/ontotimeseries (accessed February 2023).
- Liu Q, Bai Q, Kloppers C, Fitch P, Bai Q, Taylor K, Fox P, Zednik S, Ding L, Terhorst A and McGuinness D** (2012) An ontology-based knowledge management framework for a distributed water information system. *Journal of Hydroinformatics* 15 (4), 1169–1188.
- Masmoudi M, Karray MH, Ben Abdallah Ben Lamine S, Zghal HB and Archimede B** (2020) Memon: Modular environmental monitoring ontology to link heterogeneous earth observed data. *Environmental Modelling & Software* 124, 104581. Ontology available at <https://github.com/MEMOntology/memon> (accessed February 2023).
- Met Office** (2022) MetOffice DataPoint API. Available at <https://www.metoffice.gov.uk/services/data/datapoint/api-reference> (accessed April 2023).
- Met Office and Environment Agency** (2022) Check for flooding. Available at <https://check-for-flooding.service.gov.uk/river-and-sea-levels> (accessed April 2023).
- Mughal MH, Shaikh ZA, Wagan AI, Khand ZH and Hassan S** (2021) ORFFM: An ontology-based semantic model of river flow and flood mitigation. *IEEE Access* 9, 44003–44031.
- Murdock A, Harfoot A, Martin D, Cockings S and Hill C** (2015) OpenPopGrid: An Open Gridded Population Dataset for England and Wales. GeoData, University of Southampton. Available at <http://openpopgrid.geodata.soton.ac.uk/> (accessed April 2023).
- Musen MA** (2015) The protégé project: A look back and a look forward. *AI Matters* 1(4), 4–12.
- Noy N and McGuinness D** (2001) *Ontology Development 101: A Guide to Creating Your First Ontology*. Stanford Knowledge Systems Laboratory, Technical Report KSL-01-05.
- Office for National Statistics** (2022) ONS Geography Linked Data: SPARQL 1.1 Query: Endpoint. Available at <https://statistics.data.gov.uk/sparql> (accessed April 2023).
- OGC** (2012) OGC GeoSPARQL – A Geographic Query Language for RDF Data. Available at <https://www.ogc.org/standards/geosparql/> (accessed March 2023).
- OGC** (2021) CityGML. Available at <https://www.ogc.org/standard/citygml/> (accessed March 2023).
- Oktari RS, Munadi K, Idroes R and Sofyan H** (2020) Knowledge management practices in disaster management: Systematic review. *International Journal of Disaster Risk Reduction* 51, 101881.
- Ordnance Survey** (2022) Ordnance Survey Data Hub. Available at <https://osdatahub.os.uk/> (accessed April 2023).
- Pathak J and Eastaff R** (2014) Flood forecasting and early warning: An example from the UK environment agency. In Singh A and Zommers Z (eds), *Reducing Disaster: Early Warning Systems for Climate Change*. Amsterdam: Springer Netherlands, pp. 185–207.
- Pour SH, Wahab AKA, Shahid S, Asaduzzaman M and Dewan A** (2020) Low impact development techniques to mitigate the impacts of climate-change-induced urban floods: Current trends, issues and challenges. *Sustainable Cities and Society* 62, 102373.
- Pritoni M, Paine D, Fierro G, Mosiman C, Poplawski M, Saha A, Bender J and Granderson J** (2021) Metadata schemas and ontologies for building energy applications: A critical review and use case analysis. *Energies* 14(7), 2024.
- QGIS Development Team** (2021) QGIS Geographic Information System.
- Rasmussen M, Pauwels P, Lefrançois M, Schneider G, Hviid C and Karlshøj J** (2021) Building Topology Ontology. Available at <https://w3c-lbd-cg.github.io/bot/> (accessed April 2023).
- RealEstateCore Consortium** (2020) Real Estate Core Ontology. Available at <https://doc.realestatecore.io/3.2/core.html#> (accessed April 2023).
- Rijgersberg H, Willems D and Top J** (2020) Ontology of Units of MEASURE. Available at <http://www.ontology-of-units-of-measure.org/page/om-2> (accessed April 2023).
- Safe Software Inc** (2022) FME Software ©.
- Savage T, Akroyd J, Mosbach S, Krdzavac N, Hillman M and Kraft M** (2022) Universal digital twin: Integration of national-scale energy systems and climate data. *Data-Centric Engineering* 3, e23.
- Scheuer S, Haase D and Meyer V** (2013) Towards a flood risk assessment ontology – Knowledge integration into a multi-criteria risk assessment approach. *Computers, Environment and Urban Systems* 37, 82–94.
- Sermet Y and Demir I** (2019) Towards an information centric flood ontology for information management and communication. *Earth Science Informatics* 12, 541–551.
- Sinha PK and Dutta B** (2020) A systematic analysis of flood ontologies: A parametric approach. *Knowledge Organization* 47, 138–159.
- Smith PJ, Hughes D, Beven KJ, Cross P, Tych W, Coulson G and Blair G** (2009) Towards the provision of site specific flood warnings using wireless sensor networks. *Meteorological Applications* 16(1), 57–64.
- Stadler C, Lehmann J, Höffner K and Auer S** (2012) LinkedGeoData: A core for a web of spatial open data. *Semantic Web* 3(4), 333–354.
- W3C** (2012) OWL Web Ontology Language Overview. Available at <https://www.w3.org/TR/owl-features/> (accessed April 2023).
- Wang C, Chen N, Wang W and Chen Z** (2018) A hydrological sensor web ontology based on the ssn ontology: A case study for a flood. *ISPRS International Journal of Geo-Information* 7(1), 2.
- Wang X, Wei H, Chen N, He X and Tian Z** (2020) An observational process ontology-based Modeling approach for water quality monitoring. *Water* 12(3), 715.

- Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, Blomberg N, Boiten JW, da Silva Santos LB, Bourne PE, Bouwman J, Brookes AJ, Clark T, Crosas M, Dillo I, Dumon O, Edmunds S, Evelo CT, Finkers R, Gonzalez-Beltran A, Gray AJ, Groth P, Goble C, Grethe JS, Heringa J, 't Hoen PA, Hooft R, Kuhn T, Kok R, Kok J, Lusher SJ, Martone ME, Mons A, Packer AL, Persson B, Rocca-Serra P, Roos M, van Schaik R, Sansone SA, Schultes E, Sengstag T, Slater T, Strawn G, Swertz MA, Thompson M, van der Lei J, van Mulligen E, Velterop J, Waagmeester A, Wittenburg P, Wolstencroft K, Zhao J and Mons B (2016) The FAIR guiding principles for scientific data management and stewardship. *Scientific Data* 3(1), 160018.
- Wolf K, Dawson RJ, Mills JP, Blythe P and Morley J (2022) Towards a digital twin for supporting multi-agency incident management in a smart city. *Scientific Reports* 12(1), 16221.
- World Wide Web Consortium (2017) Semantic Sensor Network Ontology. Available at <https://www.w3.org/TR/vocab-ssn/> (accessed February 2023).
- Wrachien DD, Garrido J, Mambretti S and Requena I (2012) Ontology for flood management: A proposal. In Proverbs D, Mambretti S, Brebbia C and Wrachien DD (eds), *Flood Recovery, Innovation and Response III*. Southampton: WIT Press.
- Xiao G, Lanti D, Kontchakov R, Komla-Ebri S, Güzel-Kalaycı E, Ding L, Corman J, Cogrel B, Calvanese D and Botoeva E (2020) The virtual knowledge graph system ontop. In Pan JZ, Tamma V, d'Amato C, Janowicz K, Fu B, Polleres A, Seneviratne O and Kagal L (eds), *The Semantic Web – ISWC 2020*. Cham: Springer International Publishing, pp. 259–277.
- Xiaomin Z, Jianjun Y, Xiaoci H and Shaoli C (2016) An ontology-based knowledge modelling approach for river water quality monitoring and assessment. *Procedia Computer Science* 96, 335–344.
- Zhou X, Eibeck A, Lim MQ, Krdzavac N and Kraft M (2019) An agent composition framework for the J-park simulator – A knowledge graph for the process industry. *Computers & Chemical Engineering* 130, 106577.

A. Appendix

A.1. Namespaces

BOT: <<https://w3id.org/bot#>>.

DABGEO: <<http://www.purl.org/oema/infrastructure/>>.

BIMERR: <<http://bimerr.iot.linkeddata.es/def/building#>>.

DB: <<http://www.google.com/digitalbuildings/0.0.1/facilities#>>.

DERIV: <<https://www.theworldavatar.com/kg/ontoderivation/>>.

ENVO: <<http://purl.obolibrary.org/obo/>>.

GEO: <<http://www.opengis.net/ont/geosparql#>>.

GEOLIT: <<http://www.bigdata.com/rdf/geospatial/literals/v1#>>.

ICITY: <<http://ontology.eil.utoronto.ca/icity/Building/>>.

IFC: <https://standards.buildingsmart.org/IFC/DEV/IFC4/ADD2_TC1/OWL#>.

ICONTACT: <<http://ontology.eil.utoronto.ca/icontact.owl#>>.

LRPPI: <<http://landregistry.data.gov.uk/def/ppi/>>.

M3L: <<http://purl.org/iot/vocab/m3-lite#>>.

OM: <<http://www.ontology-of-units-of-measure.org/resource/om-2/>>.

OCGML: <<http://www.theworldavatar.com/ontology/ontocitygml/citieskg/OntoCityGML.owl#>>.

ONTOBUILTENV: <<https://www.theworldavatar.com/kg/ontobuiltenv/>>.

ONTOEMS: <<https://www.theworldavatar.com/kg/ontoems/>>.

ONTOFLOOD: <<https://www.theworldavatar.com/kg/ontoflood/>>.

ONTOUOM: <<https://www.theworldavatar.com/kg/ontouom/>>.

OWL: <<http://www.w3.org/2002/07/owl#>>.

RDF: <<http://www.w3.org/1999/02/22-rdf-syntax-ns#>>.

RDFS: <<http://www.w3.org/2000/01/rdf-schema#>>.

RT: <<http://environment.data.gov.uk/flood-monitoring/def/core/>>.

SIO: <<http://semanticscience.org/resource/>>.

SKOS: <<http://www.w3.org/2004/02/skos/core#>>.

SOPH: <<http://sweetontology.net/phen/>>.

SOPHHY: <<http://sweetontology.net/phenHydro/>>.

TIME: <<http://www.w3.org/2006/time#>>.

TS: <<https://www.theworldavatar.com/kg/ontotimeseries/>>.

WEATHER: <<https://www.auto.tuwien.ac.at/downloads/thinkhome/ontology/WeatherOntology.owl#>>.

XSD: <<http://www.w3.org/2001/XMLSchema#>>.

Cite this article: Hofmeister M, Bai J, Brownbridge G, Mosbach S, Lee KF, Farazi F, Hillman M, Agarwal M, Ganguly S, Akroyd J and Kraft M (2024). Semantic agent framework for automated flood assessment using dynamic knowledge graphs. *Data-Centric Engineering*, 5, e14. doi:10.1017/dce.2024.11