computers

Article

Ontology Development for Detecting Complex Events in Stream Processing: Use Case of Air Quality Monitoring

Rose Yemson [1], Sohag Kabir [1,*], Dhavalkumar Thakker [2] and Savas Konur [1]

[1] School of Computer Science, AI, and Electronics, University of Bradford, Bradford BD7 1DP, UK; r.a.yemson@bradford.ac.uk (R.Y.); s.konur@bradford.ac.uk (S.K.)
[2] School of Computer Science, University of Hull, Cottingham Road, Hull HU6 7RX, UK; d.thakker@hull.ac.uk
* Correspondence: s.kabir2@bradford.ac.uk; Tel.: +44-1274-232212

Abstract: With the increasing amount of data collected by IoT devices, detecting complex events in real-time has become a challenging task. To overcome this challenge, we propose the utilisation of semantic web technologies to create ontologies that structure background knowledge about the complex event-processing (CEP) framework in a way that machines can easily comprehend. Our ontology focuses on Indoor Air Quality (IAQ) data, asthma patients' activities and symptoms, and how IAQ can be related to asthma symptoms and daily activities. Our goal is to detect complex events within the stream of events and accurately determine pollution levels and symptoms of asthma attacks based on daily activities. We conducted a thorough testing of our enhanced CEP framework with a real dataset, and the results indicate that it outperforms traditional CEP across various evaluation metrics such as accuracy, precision, recall, and F1-score.

Keywords: complex events; traditional complex event processing; semantic web technology; ontology; Internet of Things; indoor air quality; asthma

check for updates

Citation: Yemson, R.; Kabir, S.; Thakker, D.; Konur, S. Ontology Development for Detecting Complex Events in Stream Processing: Use Case of Air Quality Monitoring. *Computers* **2023**, *12*, 238. https://doi.org/10.3390/computers12110238

Academic Editor: Paolo Bellavista

Received: 29 September 2023
Revised: 7 November 2023
Accepted: 13 November 2023
Published: 16 November 2023

1. Introduction

The rise of sensor networks and smart devices has led to an unprecedented increase in data collection, presenting a significant challenge in real-time processing. The Internet of Things (IoT) has revolutionised the way things are connected and is the most important big data project identified by Gartner [1]. The ability to react quickly to changing trends and provide instantaneous business intelligence has become a crucial factor in determining the success or failure of a company. Detecting complex events enables intelligent decision making by identifying patterns and correlations, and real-time analysis can yield profits and improve decision making. However, real-time streaming is becoming increasingly challenging with the proliferation of billions of IoT devices, including RFIDs, sensors, remote sensing satellites, and intelligent devices. These devices generate vast amounts of heterogeneous event streams that require constant observation and analysis [2]. Among these devices, sensors are commonly used to detect conditions that may lead to complex events. Reacting swiftly in a mission-critical situation can significantly impact various domains, including healthcare, energy, transportation, nuclear, agriculture, and so on. Detecting complex events in real-time, especially when they cannot be identified on the fly, is a formidable challenge in this context. The use of complex event processing (CEP) has become increasingly popular in IoT deployments due to its ability to process real-time event streams effectively in dynamic business environments [3].

Unlike traditional offline analysis methods used in relational databases, CEP continuously processes and analyses high-speed data streams [4]. It is an exceptional tool for managing various data streams, providing actionable insights, excelling at correlating distributed data to identify and respond to critical situations instantaneously [5]. CEP has proven to be efficient at processing high-frequency data streams in real-time [3]. Each stream is considered an event in CEP, with corresponding event objects carrying general meta-data such as time-stamp, event ID, and event-specific information, e.g., sensor ID and measured data. Although a single event may not have a useful meaning, it must be correlated with other events to gain a significant meaning [3]. CEP enables the analysis of continuous streams of events, aiding in identifying the existence of complex sequences of events [6].

Despite the recognised importance of CEP, its expressivity is limited when it comes to detecting large numbers of complex events in real-time. The increasing demand for real-time responses in IoT applications poses challenges in collecting and structuring data in a knowledge base. It is in this context that ontologies emerge as a pivotal tool. They have been extensively investigated for capturing background and expert knowledge from various perspectives, integrating data from heterogeneous sources, building knowledge bases, knowledge acquisition, analysing data streams, and managing knowledge and system dynamism [7,8]. Ontologies can greatly improve event processing by integrating semantics, allowing for the use of definitions, background knowledge, and behaviour rules [9]. This provides a more efficient declarative event processing and detailed descriptions of complex events, patterns, and reactions to situations, which can all be directly transformed into operational efficiency [10].

Incorporating semantics into conventional CEP systems enhances the recognition capabilities of events and provides an enhanced understanding of event implications, resulting in augmented situational awareness and proactive responses. The motivation for this article lies in the following implications of integrating semantics into classical CEP for IoT-enabled systems.

1. Improved event detection: The primary motivator for improved real-time event detection in stream processing is often hindered by the CEP approaches' reliance on syntactic pattern matching, limiting the expressiveness and efficiency of event detection.
2. Context-aware event processing: In IoT scenarios, understanding events in context is critical. Semantic integration enables the consideration of context, leading to more precise event recognition.
3. Expressive event pattern definition: Developing complex event patterns with semantics allows for richer and more nuanced event definitions, accommodating a broader range of real-world scenarios.
4. Real-time event detection: Real-time processing demands rapid event detection. Semantic integration can expedite event recognition, enabling timely responses.
5. Enhanced reasoning and inference: Semantics facilitates advanced reasoning and inference capabilities, empowering systems to make more informed decisions based on detected events.
6. Personalised IoT applications: Personalisation is increasingly crucial in IoT applications. Semantic integration can enable personalised event processing and responses tailored to individual user needs.
7. Scalability and flexibility: Achieving scalability and flexibility in event processing systems is vital. Semantic integration can contribute to the scalability and adaptability of CEP systems.

This research is driven by the need to detect complex events in stream processing for indoor air quality monitoring. This article provides a motivating example of how semantic web technology integration can play a pivotal role in identifying complex events related to environmental factors, asthmatic conditions, asthma symptoms, and daily activities. Therefore, this work endeavoured to find answers to the following questions:

1. How can we organise knowledge about the CEP framework for IoT applications in a way that machines can interpret?
2. How can structured knowledge integration enhance data analysis and complex event detection, and what is the impact of integrating semantic web technologies on the traditional CEP in real-time complex event detection, making it smarter?

To address the outlined motivations and research questions, we proposed the development of a structured ontology tailored to IoT applications meticulously integrated into the CEP framework. Through practical use cases and competency questions (CQs), we demonstrated the efficiency and effectiveness of this integrated approach in detecting complex events within dynamic event streams.

The remainder of this article is organised as follows: Section 2 provides a comprehensive literature review, examining current research in stream processing, complex event detection, ontologies, and their applications in air quality monitoring. Section 3 outlines our methodology, detailing the data collection process, ontology development, and integration into the CEP framework. This section delves into the specifics of ontology development for air quality monitoring, including design, knowledge acquisition, reasoning, and validation. In Section 4, we discuss the integration of ontology into the CEP framework and its impact on event detection. This section also presents a real-world use case for air quality monitoring, showcasing the practical application of our approach. A comprehensive discussion of our findings, highlighting achievements, limitations, and avenues for future research, was also provided in this section. Finally, Section 5 provides a summary of our conclusions, emphasising the contributions and practical significance of this study.

2. Background and Literature Review

2.1. CEP and IoT

CEP has emerged as a popular solution in the field of IoT research, attracting considerable attention from researchers for its widespread use of IoT devices and its ability to uncover a plethora of innovative applications. These applications require efficient and seamless real-time processing, analysis, and correlation of event streams originating from various data sources. In this section, the related work was focused on IoT applications needing CEP, following early work on complex event processing. In the study [11], the application of Timed Petri Net was instrumental in detecting complex events and modelling their RFID complex event-processing engine. Several studies have proposed different techniques and frameworks for detecting complex events, which have evolved over time to provide a reactive solution. One approach involves using primitive events gathered through the Expressive Stream Language (ESL) and a developed semantic for deriving rule patterns. Another approach [12] employs an automata-based system that uses SQL-like SASE language, while another [13] uses temporal constraints for discovering composite events in an RFID event detection application named RCEDA.

Middleware architectures have also been proposed to integrate wireless sensor networks and RFID systems for real-time analysis of streaming data [14]. These middleware use event processing language to define complex events, filter, group, aggregate, and construct them. Dong et al. [15] implemented CEP in their RFID middleware to identify complicated events. Furthermore, event processing systems have been designed using different processing algorithms, such as automata-based processing algorithms, TESLA language, and event meta-modelling [16,17]. These systems have been implemented in enterprise information systems using RFID data. Lastly, a CEP-based information middleware has been proposed to develop real-time trajectory-based services [18]. Dunken et al. [19] suggested a reference architecture using CEP for the real-time analysis of traffic data in decision-support systems. Their Intelligent Transportation Management System prototype uses event-processing agents and rule-based representation of local traffic expertise. However, the proposed solution requires further evaluation for more distributed implementation. Integrating background knowledge representation through ontologies can enable the automatic translation of natural language interfaces into ontological representations within traditional CEP frameworks.

2.2. Ontology for Complex Events

Ontologies play a crucial role in semantic web technologies and provide an explicit specification of conceptualisation. According to Gruber's definition, an ontology is an explicit specification of a conceptualisation, where it defines the vocabulary with which facts about the domain are expressed. With ontologies, it is possible to model knowledge about events and concepts structured in a knowledge base [20]. Numerous studies have underscored the significance of ontologies in diverse domains, showcasing their applicability in understanding complex events. Notably, researchers have utilised ontologies to capture context, detect patterns, and enable semantic correlation of events. For instance, Sasa et al. [21] linked reactive rules with ontologies to enable the detection of similar complex event patterns. Moser et al. [22] focused on semantic event correlation, recognising its necessity in relating events from various sources for pattern detection. Paschke [23] proposed a language for semantic CEP pattern design. In contrast, we propose a background knowledge framework using existing ontology languages, allowing for reasoners and standard OWL language support. In the context of smart homes, 'E-care@home' [24] demonstrated the importance of ontological structures in managing complex events domestically. Similarly, the 'Agri-IoT' project [25] leveraged ontologies for intelligent farming applications, addressing complex event monitoring in agriculture. Semantic ontologies were employed in healthcare to navigate intricate healthcare operations and patient care [26]. Furthermore, in countering terrorism, complex event processing and ontologies were integrated for the real-time detection of online terrorist content [27]. These studies collectively underscore the versatility and applicability of ontologies in diverse domains. However, several gaps remain in the literature. In the context of smart homes, further research should delve into practical implementation challenges and the scalability of ontological context-aware systems. Agriculture requires the exploration of ontological interoperability and standards for seamless data exchange. Healthcare would benefit from practical guidelines for ontological structure implementation. In the realm of counter-terrorism, there is a need to evaluate the real-world effectiveness of integrated systems and their adaptability to evolving online threats. Lastly, a cross-domain gap exists in exploring how ontological structures from various domains can be harmonised to address complex events transcending multiple domains.

2.3. Air Quality Monitoring with CEP

Air pollution poses a significant threat to the health of millions of people globally by contributing to respiratory and cardiovascular diseases. It is crucial to monitor air quality in real-time to prevent health complications and fatalities. The framework proposed in [28] leverages the concepts of CEP, software-defined networks, and Fog Computing to predict air quality in near real-time. The real-time calculation of pollutant concentration and patterns is handled by CEP. In [29], an intelligent transportation system model was introduced that utilises CEP and CPNs to determine traffic levels and AQI. In [30], the authors propose using CEP to calculate air quality levels in Spain and Morocco, extracting only useful information in real-time.

The authors of [31,32] have focused on the relationship between asthma attacks and pollutant exposure. Additionally, Ref. [33] reviewed the impact of outdoor and indoor pollutants on individuals with asthma. They found that these pollutants pose a significant threat to the health of asthma patients and can lead to more asthma symptoms, exacerbations, and reduced lung function. Further reviews were performed by [34] that correlate asthma development to air pollution caused by industrial activities, disinfection by-products, traffic, and e-cigarettes/tobacco. The author highlights the role of air pollutants as stimulants of childhood asthma. The proposed framework in [35] effectively monitors and analyses environmental triggers of asthma attacks, which can aid healthcare professionals in better managing and preventing asthma attacks. The potential for early detection of asthma exacerbations can be facilitated by machine learning algorithms over traditional paper-based action plans, as proposed by the authors in [36]. However, this approach lacks real-time capabilities as the utilisation of smartphone apps linked to digital spirometers and inhalers for electronic data collection will significantly enhance the predictive ability of these algorithms. The proposed approach by the authors in [37] effectively identifies asthma risk factors from multiple data streams by emphasising the crucial role of time lags between events for correlation.

In summary, while prior studies have provided foundational knowledge in stream processing, complex event detection, and air quality monitoring, this research aims to bridge the gap by developing and integrating ontologies to improve event detection accuracy and facilitate natural language interfaces for complex event specifications and detection in the context of IoT applications.

3. Ontology Development for Complex Events

Ontology is an explicit specification of a conceptualisation [38]. An ontology is a structured representation of concepts, object property relations, instances, data property relations, axioms, and rules [39]. It is created using a machine-readable language to develop explicit and formal conceptualisations of a specific domain [38]. The three commonly used standards of ontology languages are Web Ontology Language, Resource Description Framework, and Description Logic [40]. The focus of this work is to develop an ontology that structures background knowledge about CEP in a machine-interpretable way, allowing for the detection of complex events.

3.1. Ontology Development Methodology

Ontology has become a standard practice for describing a shared vocabulary in information sharing and reuse [41]. Developing an ontology can be a challenging and time-consuming task, but it is crucial to follow a methodology that is similar to software development [42], as it is an iterative process that involves technical complexity. Developing an ontology does not have a precise methodology, according to Kapoor [43]. Similarly, Noy [44] points out that selecting an ontology design methodology is up to the developer and depends on the intended application or purpose, as noted in Aminu's review [41]. After reviewing the relevant literature, it was discovered that the prevalent methodologies employed include Noy and McGuiness methodology, Gruninger and Fox's methodology, Uschold and King's methodology, and Methontology, in addition to NEON and other methodologies [41].

Noy and McGuinness methodology follows an iterative process that includes incorporating more details in each iteration and making modelling decisions at various stages of the process [44]. However, this approach was not considered in our study due to the lack of ontology evaluation and documentation.

Uschold and King's methodology was designed based on the Enterprise's experience. This methodology was not considered due to its lack of description of activities, processes, techniques, or lifecycle.

The Methontology methodology makes the process of building ontologies at the knowledge level easier, which includes development processes, techniques for each activity, and an evolving prototypes-based life cycle, as described in [45]. Among the known methodologies, Methontology is considered the most developed [46]. However, it lacks consideration for competency questions, and there is a need for recommendations for pre-development processes.

Gruninger and Fox's methodology, TOVE (Toronto Virtual Enterprise), is a methodology based on scenarios that elucidates the functionality of an ontology [47]. The primary goal of this methodology is to identify queries, objects, and predicates in the ontology [48]. Nevertheless, we chose not to utilise this approach for our work since it does not take into account the incorporation of pre-existing ontologies.

NeOn Methodology offers guidelines for developing ontologies without imposing a rigid workflow [49,50]. The method consists of a glossary that explains the processes and activities required for ontology development. Additionally, two ontology life cycle models, namely the waterfall and iterative models, are provided along with methodological guidelines for various processes and activities. These guidelines are explained in detail in terms of goals, inputs, outputs, and associated constraints. Additionally, they are presented procedurally with workflow specifications and supported by empirical examples [49,51,52]. The NeOn methodology emphasises the importance of reusing and re-engineering both ontological and non-ontological knowledge resources, making it a valuable tool for ontology development. This methodology has been demonstrated in [53].

This methodology was chosen for our work due to its effectiveness in constructing ontologies by leveraging existing knowledge-aware resources and adapting them to specific contexts. Additionally, it supports the continuous development of ontology networks in distributed environments, which aligns with the requirements of our study. To ensure that the ontology developed satisfies the competency questions, we must validate it. The methodology proposed here offers a robust framework for ontology evaluation, which includes proving consistency, completeness, properties, relationships, and classes through the use of competency questions (CQs). The approach is characterised by its comprehensive and detailed documentation of every stage of ontology development, which facilitates accurate and rigorous evaluation.

The ontology was developed following the step-by-step waterfall life cycle models, with explicit details and guidelines for every process and activity of ontology development. The choice of the waterfall life cycle model was motivated by its suitability for our specific ontology development needs. The waterfall model is characterised by its structured and sequential approach, where each phase must be completed before moving on to the next. This level of rigour and structure was deemed essential in ensuring the methodical and thorough development of the complex event ontology. This allowed us to provide explicit details and guidelines for each process and activity involved in ontology development, enhancing transparency, traceability, and quality assurance. The proposed methodology for developing the complex event ontology is shown in Figure 1, describing each phase of the life cycle model.

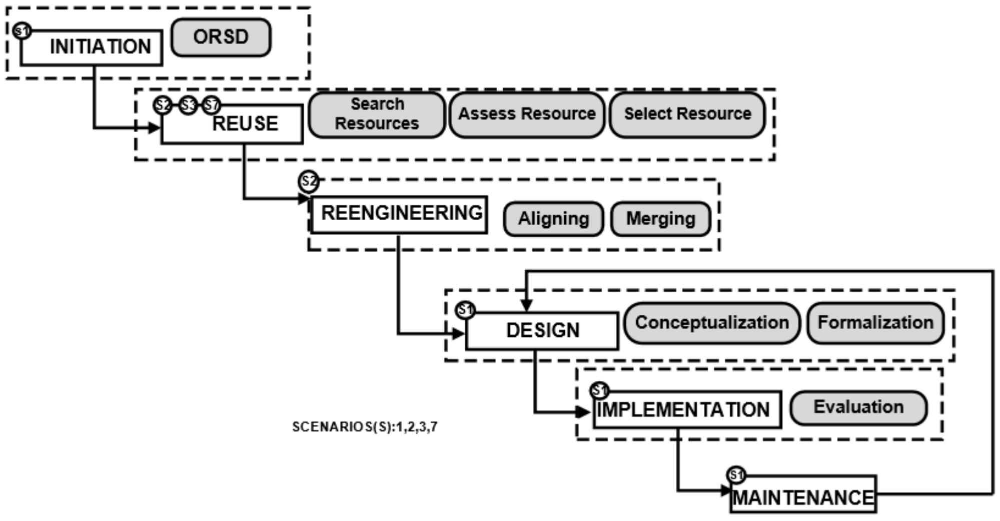
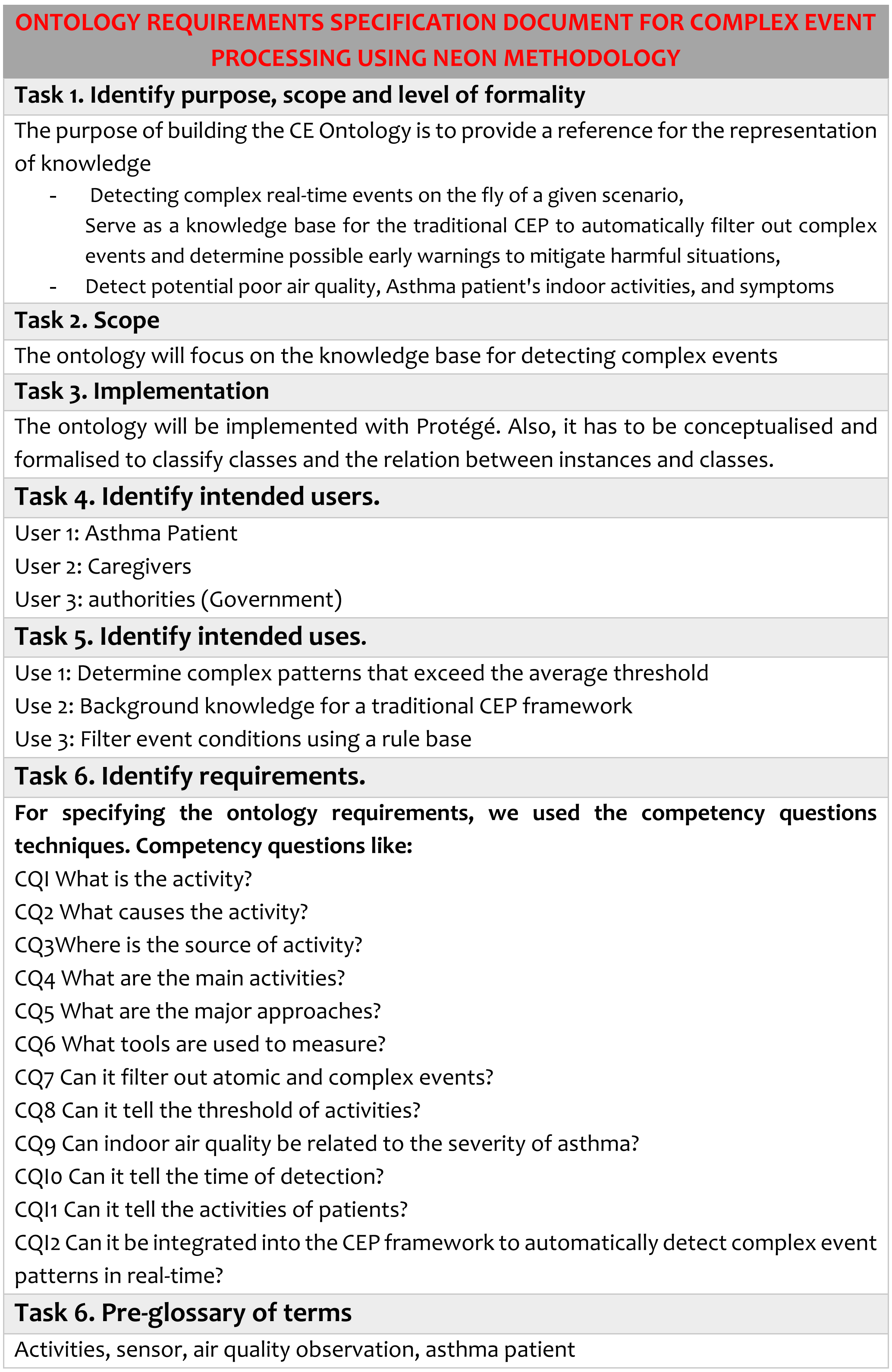


Figure 1. Proposed development method adapted from [53].

Initiation: The first step of our work was to gather all the necessary requirements for developing the ontology and ensuring that it meets all the expected standards. To accomplish our goal, we conducted a thorough investigation by reviewing a vast amount of literature and analysing research documents from different sources. This helped us in developing a highly detailed Ontology Requirements Specification Document (ORSD). The ORSD outlines the ontology's purpose, scope, implementation language, intended users, intended use, and competency questions (CQs) in great detail. The ORSD is depicted in Figure 2.



ONTOLOGY REQUIREMENTS SPECIFICATION DOCUMENT FOR COMPLEX EVENT PROCESSING USING NEON METHODOLOGY

Task 1. Identify purpose, scope and level of formality

The purpose of building the CE Ontology is to provide a reference for the representation of knowledge

- Detecting complex real-time events on the fly of a given scenario,
  Serve as a knowledge base for the traditional CEP to automatically filter out complex events and determine possible early warnings to mitigate harmful situations,
- Detect potential poor air quality, Asthma patient's indoor activities, and symptoms

Task 2. Scope

The ontology will focus on the knowledge base for detecting complex events

Task 3. Implementation

The ontology will be implemented with Protégé. Also, it has to be conceptualised and formalised to classify classes and the relation between instances and classes.

Task 4. Identify intended users.

User 1: Asthma Patient
User 2: Caregivers
User 3: authorities (Government)

Task 5. Identify intended uses.

Use 1: Determine complex patterns that exceed the average threshold
Use 2: Background knowledge for a traditional CEP framework
Use 3: Filter event conditions using a rule base

Task 6. Identify requirements.

For specifying the ontology requirements, we used the competency questions techniques. Competency questions like:

CQI What is the activity?
CQ2 What causes the activity?
CQ3Where is the source of activity?
CQ4 What are the main activities?
CQ5 What are the major approaches?
CQ6 What tools are used to measure?
CQ7 Can it filter out atomic and complex events?
CQ8 Can it tell the threshold of activities?
CQ9 Can indoor air quality be related to the severity of asthma?
CQI0 Can it tell the time of detection?
CQI1 Can it tell the activities of patients?
CQI2 Can it be integrated into the CEP framework to automatically detect complex event patterns in real-time?

Task 6. Pre-glossary of terms

Activities, sensor, air quality observation, asthma patient

Figure 2. Ontology Requirements Specification Document for Complex Event Processing (ORSD).

Reuse: In this phase, we used existing resources to describe the scenario. While an ontology that met all of our requirements could not be found, we reused resources found in the air quality, asthma, and event ontologies. We categorised these ontologies into upper, reusable, and application layers and used the Semantic Sensor Net Ontology for the sensor layer.

Merging: There was a need for merging since our work reused different ontologies of different domains to create a new ontology to suit our goals. Therefore, the ontology was aligned using OpenCyc and Event-Model-F event-based, respectively. These ontologies were selected because of the key features representing the upper ontology, domain ontology, and linking the different ontologies to create complex events.

Re-engineering: Re-engineering non-ontological resources was required to align with our goals as the existing ontologies did not entirely fulfil our goals.

Design: In this phase, we consolidated the requirements collected from earlier phases and defined the essential characteristics of each module of our framework. We then translated this information into a formalised conceptual model, ensuring that all technical details were accurately represented.

Implementation: The implementation of the ontology was carried out using Protégé. The evaluation of the ontology involved a case study that explored the relationship between indoor air quality and an asthma patient. The evaluation process compared the ontology with the ontology requirement specification document, which outlines the ontology requirements and competency questions. This comparison ensured that the ontology was built correctly and in compliance with the specified ontology requirements.

Maintenance: This phase will recur, with updates to the ontology and corrections made during the design phase, following the Waterfall life cycle model.

3.2. Architecture of Proposed Framework

The framework depicted in Figure 3 combines semantic web technology with the conventional CEP detection mechanism, boosting its ability to identify more complex events. The architecture of the framework is composed of Monitoring, Event Analysis, and Application layers. The CEP framework continuously receives events from diverse sources and processes them instantaneously to detect complex events using predefined rules. The framework exploits ontologies and background knowledge developed to incorporate semantic technology, providing an unambiguous and formal representation of entities, concepts, and their interdependencies.

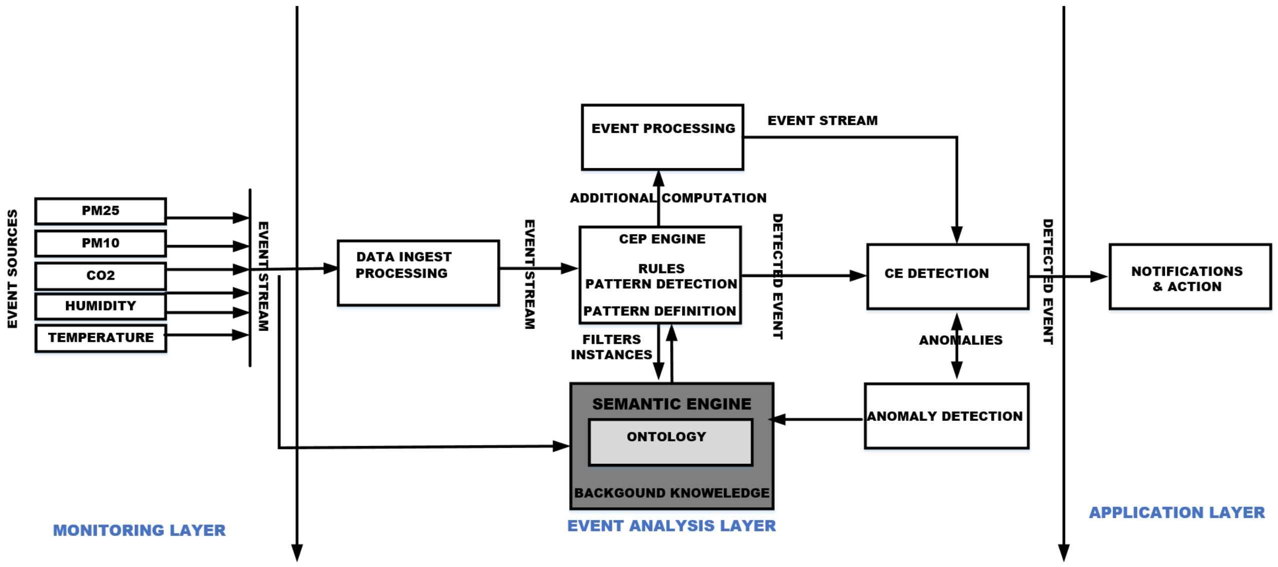


Figure 3. High-level overview of the proposed framework.

3.2.1. Monitoring Layer

This layer is the first layer of the proposed framework that performs the task of gathering and processing the event data from multiple sources and sending them to the event analysis layer for further analysis.

- **Event Source Sub-Layer:** The layer in question is tasked with the responsibility of creating events and transmitting data to the CEP engine for further analysis. The sources from which these events are generated can be any system, device, or application capable of generating events. These events can manifest in different types, including but not limited to financial transactions, temperature readings, and customer behaviour data. In this particular study, the work involved monitoring the continuous real-life activities of asthma patients, including their symptoms and sensor readings from different parameters.
- **Data Ingestion Processing:** Data ingestion is a critical task in the CEP framework's monitoring layer. It involves receiving event data from various sources, parsing, transforming, and ensuring data quality before forwarding it to the CEP engine. The data received from IAQ sensors includes PM10, PM2.5, CO_2, temperature, and humidity, among others.

3.2.2. Event Analysis Layer

The event analysis layer is a crucial component of the framework that handles the various data streams produced by the event sources. This layer acts as the brain of the system and is responsible for filtering, detecting, aggregating, and correlating patterns across multiple event streams. This layer comprises several sub-layers that work together to ensure that the system can handle complex event-processing tasks efficiently and accurately.

- **CEP engine sub-layer:** This layer is responsible for detecting and defining patterns and correlations between various events. It uses predefined rules to detect events, processes them to identify atomic events, and forwards them to the semantic sub-layer for subsequent processing. The event processing pipelines and rules of the framework are implemented using Apache Flink, which facilitates effective and efficient stream processing. The CEP engine can be configured to handle a diverse range of rules or models, such as time-based, context-based, or predictive models, to provide flexible and adaptable solutions for different scenarios. By defining time-based rules, the CEP engine can detect correlations between events and identify patterns and relationships that may occur periodically or at certain time intervals.
  CEP engines use context-aware rules to detect events and take appropriate actions. By analysing contextual information, they can identify triggers such as high CO_2 levels that can cause health issues like asthma. Predictive models can also be used to anticipate events and take proactive measures. For example, the engine may predict that indoor activities like cooking can lead to higher concentrations of particulate matter and provide timely alerts to occupants or take preventive measures.
- **Semantic engine sub-layer:** This sub-layer is critical in the framework. It uses ontology to interpret events and their relationships, enhancing analysis and interpretation beyond individual characteristics. By combining it with the CEP engine sub-layer, it efficiently filters events from multiple atomic sensed events.
  The semantic engine's reasoning mechanism detects intricate event patterns from the ontology's background knowledge. It identifies significant events beyond simple atomic events involving multiple conditions or causality. The framework's ontology can be easily customised and updated to improve the system's event detection performance, as it continually learns from new insights and data. Advanced event analysis and detection are made possible by leveraging the CEP engine and semantic web technology. The system allows for timely interventions and proactive actions by triggering notifications when events exceed predefined thresholds.

- **Event stream processing sub-layer:** This layer processes event streams using the CEP Engine's capabilities. It orchestrates a sequence of events and utilises pattern recognition to identify complex events by leveraging the knowledge stored in the ontology.
- **Complex event detection sub-layer:** The purpose of this sub-layer is to detect complex events by combining the outcomes generated by the CEP engine and the semantic engine sub-layers. It does this by applying predefined patterns to the event stream and leveraging the semantic information about the events. The integration with semantics technology enhances the detection capability, making it more accurate and efficient.
- **Anomaly detection sub-layer:** This module is designed to detect any unusual patterns in real-time event streams using advanced algorithms to identify abnormal patterns, allowing for timely interventions.

3.2.3. Application Layer

The application layer presents analysed data to users, generates reports, triggers actions, and alerts them of critical events. It also permits users to customise analysis rules and integrate the system with other software. In the context of this study, the application layer can produce notifications or alerts to the users as soon as the CEP engine identifies a correlation between indoor activities, indoor air pollutants, and asthma symptoms. For example, if cooking triggers asthma symptoms, the CEP Engine sends an alert to occupants if it exceeds the threshold. As a result, they can either avoid cooking at certain times or ventilate the kitchen. This layer is also designed to generate reports that provide a better understanding of indoor air quality and its effects on human health.

3.3. Description of the Ontology Structure

3.3.1. Classes and Subclasses

This research focuses on several relevant classes that we have identified and defined in our ontology, depicted in Figure 4. Various classes include, but are not limited to, AsthmaSymptom, IndoorAirQuality, IndoorActivity, Sensor, and Measurement, among others. Each of these classes exhibits a unique entity or concept with its respective properties, relationships, and constraints [54].

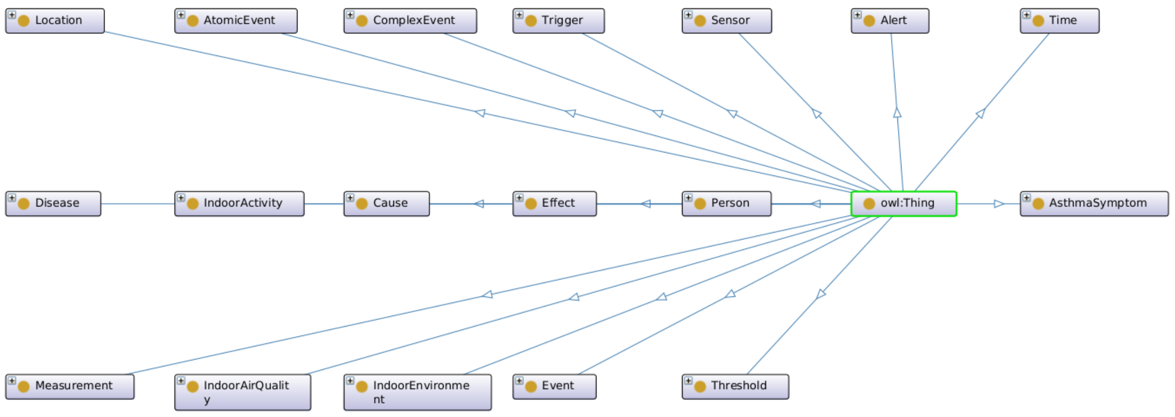


Figure 4. Classes visualised in OntoGraf.

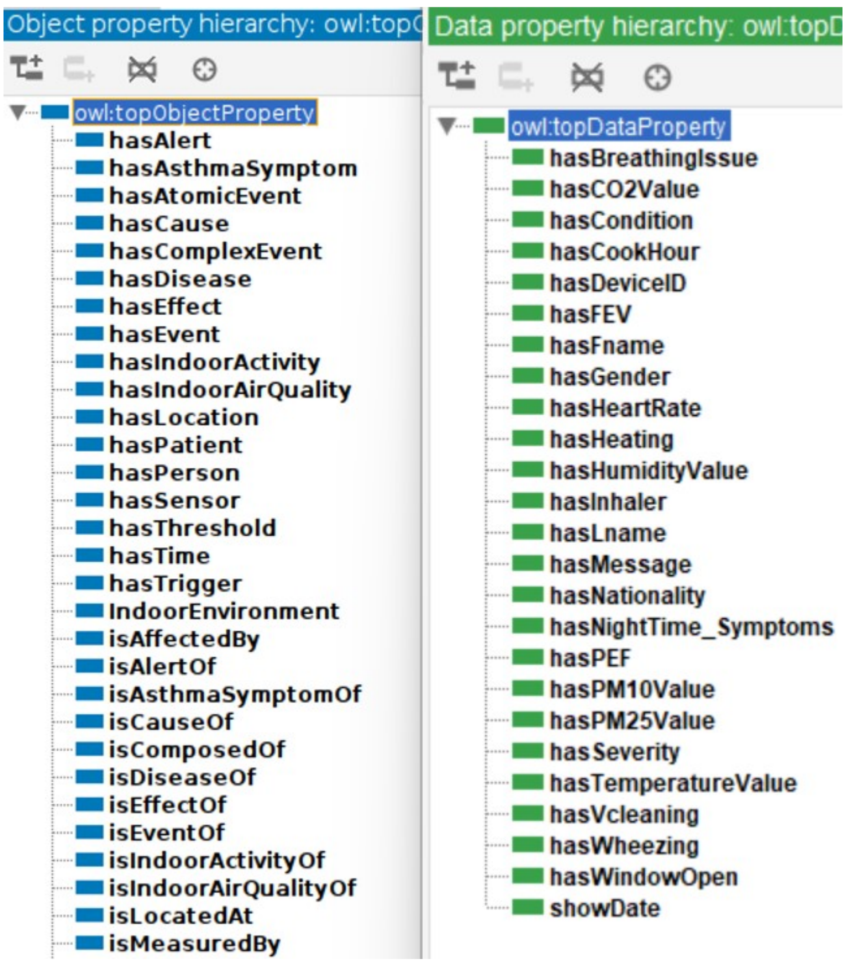
In our ontology, classes and subclasses play a pivotal role in organising concepts within the domain of air quality monitoring and event detection. Here are some key classes and their subclasses along with examples:

```
class Event {
  // Class definition for Event
}
class AtomicEvent extends Event {
  // Class definition for AtomicEvent
  hasEffect: IndoorAirQuality
}
class AsthmaSymptom {
  // Class definition for AsthmaSymptom
  hasEffect: IndoorAirQuality
}
class IndoorAirQuality {
  // Class definition for IndoorAirQuality
  hasCause: Event[]
  hasEffect: AsthmaSymptom[]
}
class IndoorActivity {
  // Class definition for IndoorActivity
  hasEffect: IndoorAirQuality
}
class Sensor {
  // Class definition for Sensor
  //  hasMeasurement: Measurement[]
}
class Location {
  // Class definition for Location
  // hasLocation: IndoorEnvironment
}
class Alert {
  // Class definition for Alert
  // hasAlert:Person
}
class Threshold {
  // Class definition for Threshold
  // hasThreshold: Measurement
}
class Trigger {
  // Class definition for Trigger
  // hasTrigger:AtomicEvent
}
class Disease {
  // Class definition for Disease
  // Represents a disease or medical condition
  // hasDisease: Person
}
class Effect {
  // Class definition for Effect
  // Represents an effect or consequence of an event
  // hasEffect: Event
}
class Person {
  // Class definition for Person
 // hasPersonEvent:Event
}
class Time {
  // Class definition for Time
  // Represents a point in time or a time interval
  // hasTime: Event
```

```
}
class Measurement {
  // Class definition for Measurement
  // Represents a measurement of a certain
  //property or quantity
  // isMeasuredBy:Sensor
}
class indoorEnvironment {
  // Class definition for
  indoorEnvironment
  // Represents the indoor environment
  // hasEffect: IndoorActivity
}
```

3.3.2. Object and Data Properties

Object and data properties are the two properties used for this ontology to describe the relationship between two individuals and the relationship between individuals and enter data values for the classes according to their data format. Some properties defined for disease, such as hasDisease, hasSymptoms, has activity, hasIndoorAirQuality are depicted in Figure 5.



Object property hierarchy: owl:topO

owl:topObjectProperty
- hasAlert
- hasAsthmaSymptom
- hasAtomicEvent
- hasCause
- hasComplexEvent
- hasDisease
- hasEffect
- hasEvent
- hasIndoorActivity
- hasIndoorAirQuality
- hasLocation
- hasPatient
- hasPerson
- hasSensor
- hasThreshold
- hasTime
- hasTrigger
- IndoorEnvironment
- isAffectedBy
- isAlertOf
- isAsthmaSymptomOf
- isCauseOf
- isComposedOf
- isDiseaseOf
- isEffectOf
- isEventOf
- isIndoorActivityOf
- isIndoorAirQualityOf
- isLocatedAt
- isMeasuredBy

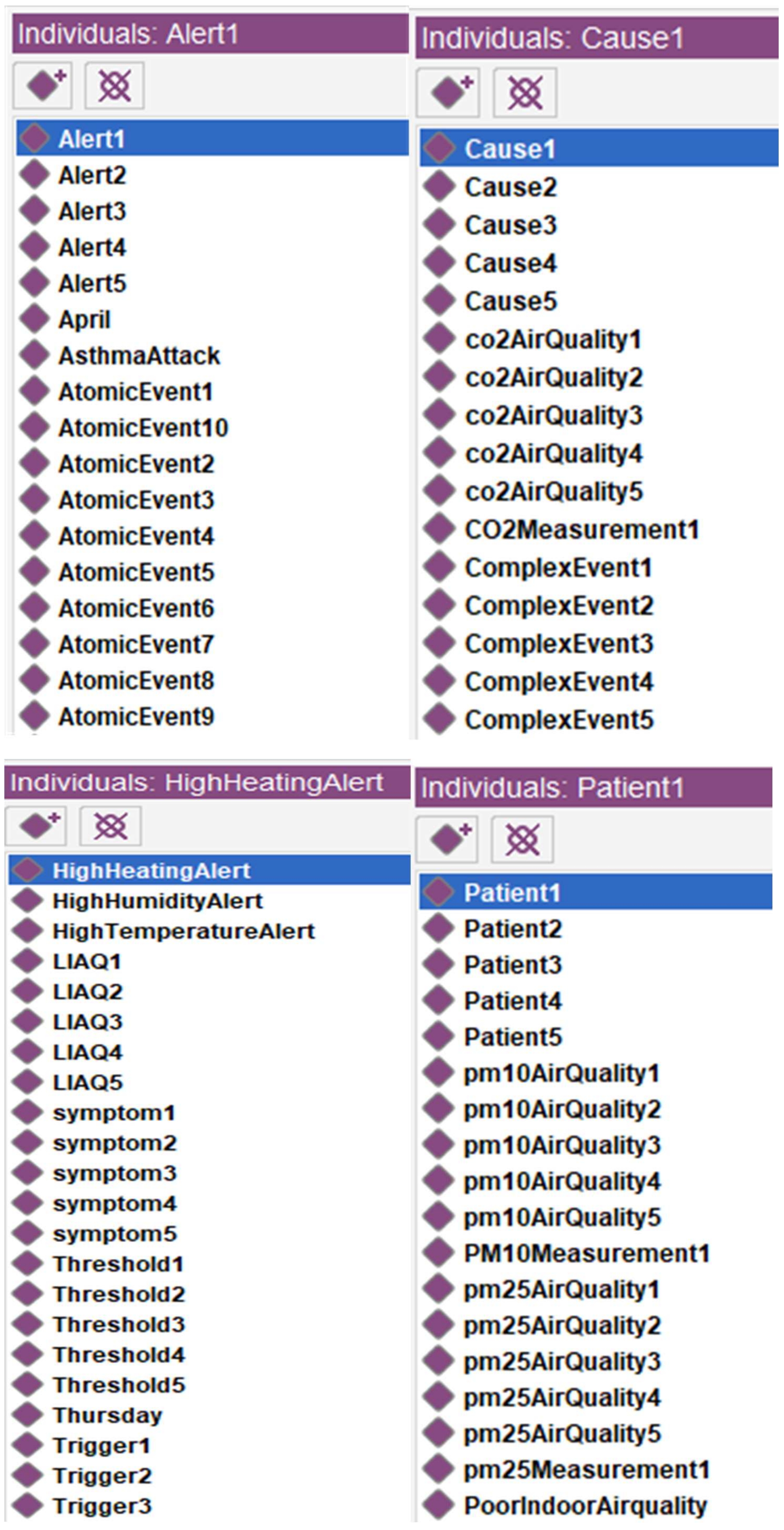
Data property hierarchy: owl:topD

owl:topDataProperty
- hasBreathingIssue
- hasCO2Value
- hasCondition
- hasCookHour
- hasDeviceID
- hasFEV
- hasFname
- hasGender
- hasHeartRate
- hasHeating
- hasHumidityValue
- hasInhaler
- hasLname
- hasMessage
- hasNationality
- hasNightTime_Symptoms
- hasPEF
- hasPM10Value
- hasPM25Value
- hasSeverity
- hasTemperatureValue
- hasVcleaning
- hasWheezing
- hasWindowOpen
- showDate

Figure 5. Some of the Object and Data Properties.

3.3.3. Instances(Individual)

Figure 6 showcases the utilisation of individuals to populate the ontology with real-world data instances, demonstrating the practical application of the ontology.


Individuals: Alert1

Alert1
Alert2
Alert3
Alert4
Alert5
April
AsthmaAttack
AtomicEvent1
AtomicEvent10
AtomicEvent2
AtomicEvent3
AtomicEvent4
AtomicEvent5
AtomicEvent6
AtomicEvent7
AtomicEvent8
AtomicEvent9

Individuals: Cause1

Cause1
Cause2
Cause3
Cause4
Cause5
co2AirQuality1
co2AirQuality2
co2AirQuality3
co2AirQuality4
co2AirQuality5
CO2Measurement1
ComplexEvent1
ComplexEvent2
ComplexEvent3
ComplexEvent4
ComplexEvent5

Individuals: HighHeatingAlert

HighHeatingAlert
HighHumidityAlert
HighTemperatureAlert
LIAQ1
LIAQ2
LIAQ3
LIAQ4
LIAQ5
symptom1
symptom2
symptom3
symptom4
symptom5
Threshold1
Threshold2
Threshold3
Threshold4
Threshold5
Thursday
Trigger1
Trigger2
Trigger3

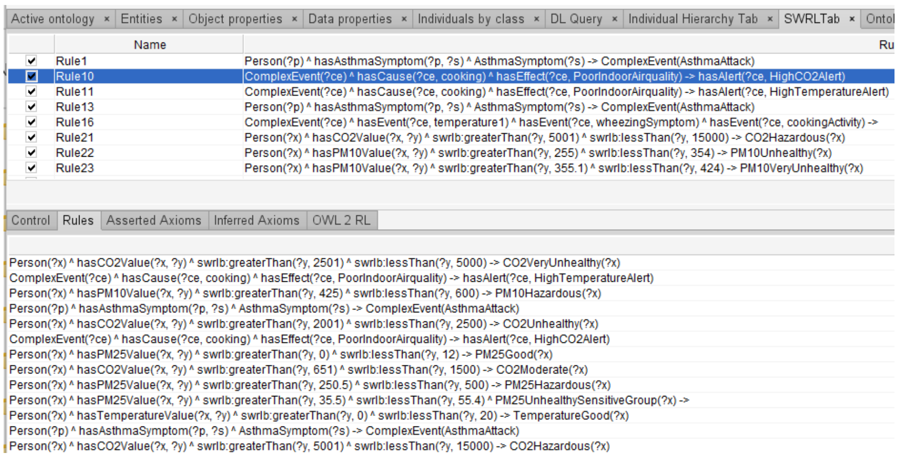
Individuals: Patient1

Patient1
Patient2
Patient3
Patient4
Patient5
pm10AirQuality1
pm10AirQuality2
pm10AirQuality3
pm10AirQuality4
pm10AirQuality5
PM10Measurement1
pm25AirQuality1
pm25AirQuality2
pm25AirQuality3
pm25AirQuality4
pm25AirQuality5
pm25Measurement1
PoorIndoorAirquality

Figure 6. Some of the individuals in the ontology.

3.3.4. SWRL Rules

Figure 7 is an example of the use of SWRL rules in our case. These rules are employed to identify events depending on asthma symptoms, indoor activities, environmental properties, and air quality measurements. By using logical operators such as AND and OR, the rules can combine events to provide a comprehensive analysis. Additionally, temporal patterns can be defined with the use of operators like SEQUENCE, PERIODIC, APERIODIC, and ALL, which allow for the detection of specific event sequences or recurring patterns. Figure 7 illustrates examples of a few of the rules utilised in the proposed ontology.



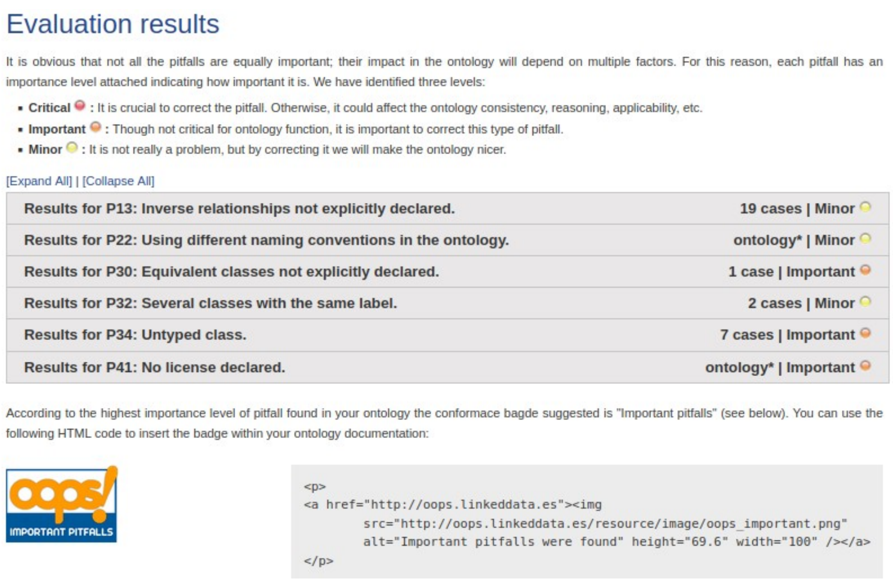
	Name	Ru
✓	Rule1	Person(?p) ^ hasAsthmaSymptom(?p, ?s) ^ AsthmaSymptom(?s) -> ComplexEvent(AsthmaAttack)
✓	Rule10	ComplexEvent(?ce) ^ hasCause(?ce, cooking) ^ hasEffect(?ce, PoorIndoorAirquality) -> hasAlert(?ce, HighCO2Alert)
✓	Rule11	ComplexEvent(?ce) ^ hasCause(?ce, cooking) ^ hasEffect(?ce, PoorIndoorAirquality) -> hasAlert(?ce, HighTemperatureAlert)
✓	Rule13	Person(?p) ^ hasAsthmaSymptom(?p, ?s) ^ AsthmaSymptom(?s) -> ComplexEvent(AsthmaAttack)
✓	Rule16	ComplexEvent(?ce) ^ hasEvent(?ce, temperature1) ^ hasEvent(?ce, wheezingSymptom) ^ hasEvent(?ce, cookingActivity) ->
✓	Rule21	Person(?x) ^ hasCO2Value(?x, ?y) ^ swrlb:greaterThan(?y, 5001) ^ swrlb:lessThan(?y, 15000) -> CO2Hazardous(?x)
✓	Rule22	Person(?x) ^ hasPM10Value(?x, ?y) ^ swrlb:greaterThan(?y, 255) ^ swrlb:lessThan(?y, 354) -> PM10Unhealthy(?x)
✓	Rule23	Person(?x) ^ hasPM10Value(?x, ?y) ^ swrlb:greaterThan(?y, 355.1) ^ swrlb:lessThan(?y, 424) -> PM10VeryUnhealthy(?x)

```
Person(?x) ^ hasCO2Value(?x, ?y) ^ swrlb:greaterThan(?y, 2501) ^ swrlb:lessThan(?y, 5000) -> CO2VeryUnhealthy(?x)
ComplexEvent(?ce) ^ hasCause(?ce, cooking) ^ hasEffect(?ce, PoorIndoorAirquality) -> hasAlert(?ce, HighTemperatureAlert)
Person(?x) ^ hasPM10Value(?x, ?y) ^ swrlb:greaterThan(?y, 425) ^ swrlb:lessThan(?y, 600) -> PM10Hazardous(?x)
Person(?p) ^ hasAsthmaSymptom(?p, ?s) ^ AsthmaSymptom(?s) -> ComplexEvent(AsthmaAttack)
Person(?x) ^ hasCO2Value(?x, ?y) ^ swrlb:greaterThan(?y, 2001) ^ swrlb:lessThan(?y, 2500) -> CO2Unhealthy(?x)
ComplexEvent(?ce) ^ hasCause(?ce, cooking) ^ hasEffect(?ce, PoorIndoorAirquality) -> hasAlert(?ce, HighCO2Alert)
Person(?x) ^ hasPM25Value(?x, ?y) ^ swrlb:greaterThan(?y, 0) ^ swrlb:lessThan(?y, 12) -> PM25Good(?x)
Person(?x) ^ hasCO2Value(?x, ?y) ^ swrlb:greaterThan(?y, 651) ^ swrlb:lessThan(?y, 1500) -> CO2Moderate(?x)
Person(?x) ^ hasPM25Value(?x, ?y) ^ swrlb:greaterThan(?y, 250.5) ^ swrlb:lessThan(?y, 500) -> PM25Hazardous(?x)
Person(?x) ^ hasPM25Value(?x, ?y) ^ swrlb:greaterThan(?y, 35.5) ^ swrlb:lessThan(?y, 55.4) ^ PM25UnhealthySensitiveGroup(?x) ->
Person(?x) ^ hasTemperatureValue(?x, ?y) ^ swrlb:greaterThan(?y, 0) ^ swrlb:lessThan(?y, 20) -> TemperatureGood(?x)
Person(?p) ^ hasAsthmaSymptom(?p, ?s) ^ AsthmaSymptom(?s) -> ComplexEvent(AsthmaAttack)
Person(?x) ^ hasCO2Value(?x, ?y) ^ swrlb:greaterThan(?y, 5001) ^ swrlb:lessThan(?y, 15000) -> CO2Hazardous(?x)
```

Figure 7. SWRL rules.

3.4. Ontology Validation

Figure 8 illustrates the outcomes of the OOPS! evaluation process, which was conducted to find common errors in our designed ontology, particularly in relation to complex event detection. The used tool meticulously crosschecked the ontology against a predefined set of patterns, flagging inconsistencies such as missing or inaccurate class definitions, uncertain modelling choices, and inaccurate usage of relationships. This enabled us to detect any potential problems that could hamper the complete and accurate detection of complex events. The findings obtained from the OOPS! evaluation directed improvements that were made to enhance the effectiveness of the ontology.



Evaluation results

It is obvious that not all the pitfalls are equally important; their impact in the ontology will depend on multiple factors. For this reason, each pitfall has an importance level attached indicating how important it is. We have identified three levels:

- **Critical** ● : It is crucial to correct the pitfall. Otherwise, it could affect the ontology consistency, reasoning, applicability, etc.
- **Important** ● : Though not critical for ontology function, it is important to correct this type of pitfall.
- **Minor** ● : It is not really a problem, but by correcting it we will make the ontology nicer.

[Expand All] | [Collapse All]

Results for P13: Inverse relationships not explicitly declared.	**19 cases | Minor** ●
Results for P22: Using different naming conventions in the ontology.	**ontology* | Minor** ●
Results for P30: Equivalent classes not explicitly declared.	**1 case | Important** ●
Results for P32: Several classes with the same label.	**2 cases | Minor** ●
Results for P34: Untyped class.	**7 cases | Important** ●
Results for P41: No license declared.	**ontology* | Important** ●

According to the highest importance level of pitfall found in your ontology the conformace bagde suggested is "Important pitfalls" (see below). You can use the following HTML code to insert the badge within your ontology documentation:

```
<p>
<a href="http://oops.linkeddata.es"><img
        src="http://oops.linkeddata.es/resource/image/oops_important.png"
        alt="Important pitfalls were found" height="69.6" width="100" /></a>
</p>
```

Figure 8. Summary of the OOPS! with minor pitfalls.

4. Complex Event Detection with Illustrative Example

In the context of real-time data analysis, complex event detection is a vital process that involves detecting meaningful patterns and correlations within a continuous stream of data. CEP techniques play a crucial role in this domain. CEP allows us to define event patterns and rules that trigger specific actions or alerts when these patterns are detected. While CEP has a wide range of applications, this paper focuses on the detection of complex events; in particular, we present event detection algorithms designed to identify instances of high indoor air pollution correlated with asthma symptoms and specific indoor activities.

4.1. Data Description

The authors conducted a systematic study to obtain data for the study, which involved gathering data on indoor activities, IAQ, and asthma symptoms. This extensive dataset was fundamental in comprehending the correlation between different factors and their influence on human health, especially with regard to respiratory conditions such as asthma. The data collected have been used in this article to test the efficiency of the developed framework. Examples of some of the data instances are depicted in Table 1.

Table 1. Example instances of IAQ dataset.

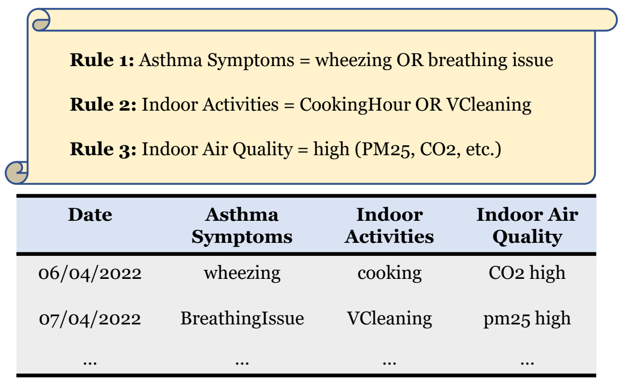
CO_2	PM10	PM2.5	Temperature	Breathing Issue	VCleaning	Heating Hour	Cooking Hour	Wheezing	Inhaler Use	Device ID
872	15.335	14.61	22.79	Bad	1	2	3	Y	Y	LAQ-x
678.7	4.039	3.279	17.67	Good	2	2	3	Y	Y	LAQ-x
519.2	32.03	29.5	16.68	Good	2	3	2	Y	N	LAQ-x
649.5	21.36	20.568	15.67	Bad	2	2	4	Y	Y	LAQ-x
780.2	18.36	16.4	16.53	Good	2	2	4	Y	N	LAQ-x
765.7	42.15	37.9	17.54	Bad	1	1	4	N	N	LAQ-x
650.7	18.01	15.79	17.38	Bad	1	1	2	N	Y	LAQ-x
710	32.8	28.1	16.97	Bad	1	2	4	Y	N	LAQ-x

4.2. Event Pattern Modelling

The following section outlines the various event patterns that have been established to identify complex events associated with asthma exacerbation. To detect high indoor air pollution and its correlation with asthma symptoms, these patterns utilise a combination of indoor air quality readings, indoor activities, and asthma symptoms. The aim is to pinpoint specific scenarios that suggest elevated indoor air pollution levels and their connection to asthma symptoms. Below are two examples of event patterns we have identified for illustrative purposes.

Event Pattern 1: "High indoor air pollution during cooking and VCleaning with wheezing or breathing issues".

This pattern's objective is to detect "High indoor air pollution during cooking or cleaning with wheezing or breathing issues". The rules of this event pattern, with a few example instances, are outlined in Figure 9.



Rule 1: Asthma Symptoms = wheezing OR breathing issue

Rule 2: Indoor Activities = CookingHour OR VCleaning

Rule 3: Indoor Air Quality = high (PM25, CO2, etc.)

Date	Asthma Symptoms	Indoor Activities	Indoor Air Quality
06/04/2022	wheezing	cooking	CO2 high
07/04/2022	BreathingIssue	VCleaning	pm25 high
...	...	...	...

Figure 9. Rules of event pattern 1 with some example instances.

Detected pattern events are represented with RDF triples (semantic web technology) in Listing 1. Each event uses properties like hasdate, hasAsthmaSymptom, hasIndoorActivity, and hasIndoorAirQuality with specific values.

Listing 1. RDF/Turtle Representation.

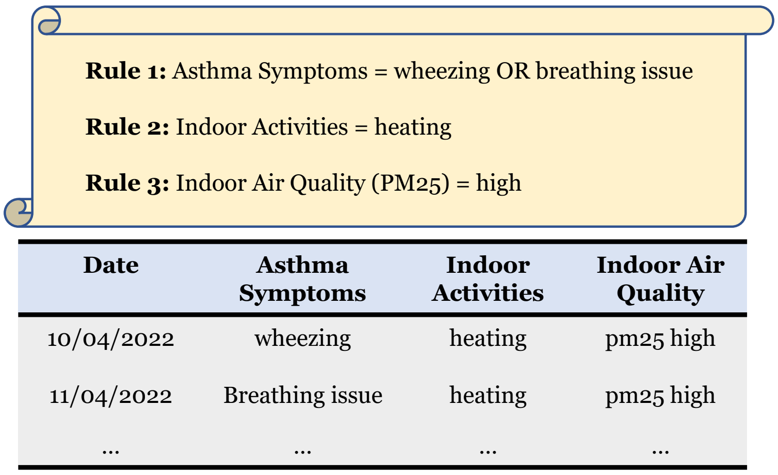
```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rayemson: <http://www.semanticweb.org/rayemson#> .
@prefix onto: <http://www.ontotext.com/> .
rayemson:Event1 a rayemson:Event ;
    rayemson:hasdate "06/04/2022" ;
    rayemson:hasAsthmaSymptom "wheezing" ;
    rayemson:hasIndoorActivity "CookingHour" ;
    rayemson:hasIndoorAirQuality "co2 high".

rayemson:Event2 a rayemson:Event ;
    rayemson:hasDate "07/04/2022" ;
   rayemson:hasAsthmaSymptoms "BreathingIssue" ;
   rayemson:hasIndoorActivity "VCleaning" ;
   rayemson:hasIndoorAirQuality "Pm25 high".
```

This pattern is applicable when trying to comprehend potential causes of asthma symptoms that may arise during cleaning or cooking, particularly when IAQ is at its peak. By recognising this pattern, individuals can proactively implement countermeasures to minimise indoor air pollution when engaging in these activities.

Event Pattern 2: "High indoor air pollution during heating, accompanied by wheezing or breathing issues".

This pattern's objective is to identify instances where indoor air pollution is elevated during heating activities, coinciding with reports of wheezing or respiratory issues. The rules of this event pattern, with a few example instances, are outlined in Figure 10.



Rule 1: Asthma Symptoms = wheezing OR breathing issue

Rule 2: Indoor Activities = heating

Rule 3: Indoor Air Quality (PM25) = high

Date	Asthma Symptoms	Indoor Activities	Indoor Air Quality
10/04/2022	wheezing	heating	pm25 high
11/04/2022	Breathing issue	heating	pm25 high
...	...	...	...

Figure 10. Rules of event pattern 2 with some example instances.

Detected pattern events are represented with RDF triples (semantic web technology) in Listing 2. Each event uses properties like hasdate, hasAsthmaSymptom, hasIndoorActivity, and hasIndoorAirQuality with specific values.

Listing 2. RDF/Turtle Representation.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rayemson: <http://www.semanticweb.org/rayemson#> .
@prefix onto: <http://www.ontotext.com/> .
rayemson:Event3 a rayemson: Event ;
    rayemson:hasdate "10/04/2022" ;
   rayemson:hasIndoorActivity "HeatingHour" ;
   rayemson:hasIndoorAirQuality "high" .
rayemson:Event4 a rayemson:Event ;
    rayemson:hasDate "11/04/2022" ;
   rayemson:hasAsthmaSymptom "BreathingIssue" ;
    rayemson:hasIndoorActivity "HeatingHour" ;
    rayemson:hasIndoorAirQuality "high" .
```

4.3. Event Detection Algorithms

In this subsection, we present the algorithms developed to detect complex events based on the defined event patterns previously described. Each event pattern is associated with a set of rules, and the goal is to detect instances that match these patterns within a continuous data stream.

4.3.1. Algorithm 1: High Indoor Air Pollution during Cooking

Algorithm Description: This algorithm iterates through the data stream and checks each data point against the defined rules for Algorithm 1. When all conditions are met, it appends an asthma exacerbation event to the list of detected events.

Algorithm 1 High Indoor Air Pollution Event Detection

```
1: for each data_point in data_stream do
2:     if data_point['Asthma Symptoms'] in ['wheezing', 'breathing issue'] and
3:       data_point['Indoor Activities'] in ['CookingHour', 'VCleaning'] and
4:       data_point['Indoor Air Quality'] == 'high' then
5:         events.append('Asthma Exacerbation Event')
6:     end if
7: end for
```

4.3.2. Algorithm 2: High Indoor Air Pollution during Heating

Algorithm Description: This algorithm scans the data stream, applying the specified rules for Algorithm Algorithm 2. When the conditions are met, it adds an asthma exacerbation event to the list of detected events.

Algorithm 2 High Indoor Air Pollution Event Detection

```
1: for each data_point in data_stream do
2:     if data_point['Asthma Symptoms'] in ['wheezing', 'breathing issue'] and
3:       data_point['Indoor Activities'] == 'heating' and
4:       data_point['Indoor Air Quality (PM25)'] == 'high' then
5:         events.append('Asthma Exacerbation Event')
6:     end if
7: end for
```

The proposed event detection algorithms provide a systematic approach to detecting complex events related to asthma exacerbation. By leveraging these algorithms, we can analyse real-time data streams and trigger timely alerts and/or interventions when potential asthma exacerbation events are detected.

4.4. Performance Evaluation

We evaluated the performance of the proposed complex event detection approaches using the established evaluation metrics. We also compare the results between conventional CEP and CEP integrated with semantic web technologies.

4.4.1. Evaluation Metrics

Our evaluation of event detection involved the use of precision, recall, and F1-score metrics to assess the effectiveness of each approach utilised. True Positives (TP) were identified as the total number of complex events that were correctly detected by the system. False Positives (FP) were events flagged by the system that were not complex events, while False Negatives (FN) represented the total number of complex events missed by the system. The formula to compute precision is shown in Equation (1), which was used to determine positive prediction accuracy. The recall was calculated using the Equation (2), indicating the proportion of actual positives accurately identified. The F1-score was the harmonic mean of precision and recall and functioned to balance precision and recall. A score of 1 indicated perfect precision and recall, while 0 indicated poor performance.

$$\text{Precision}: \quad \text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall}: \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$\text{F1-score}: \quad \text{F1-score} = \frac{2 \cdot (\text{Precision} \cdot \text{Recall})}{\text{Precision} + \text{Recall}} \tag{3}$$

4.4.2. Comparison of Performance of Traditional CEP and Proposed Enhanced Semantic Web-Integrated CEP

We compared the effectiveness of conventional CEP and CEP with semantic web technologies for detecting complex events in real-time IoT data analysis. The integration of semantic web technologies was found to enhance traditional CEP and improve accuracy. Traditional CEP has proven efficiency in detecting simple events, but as we use more smart things, we need to enhance its capability to find more complicated events. That is why it is useful to use the Semantic Web with CEP to find even more complicated events.

Efficiency: A comparison was conducted to measure efficiency and processing time between conventional CEP and that of the enhanced CEP with semantic web technologies, which is illustrated in Figure 11. Our findings showed that the traditional CEP had a lower processing time than the enhanced CEP. The processing times for different events are shown in the efficiency graph, and it is evident that the integrated approach can handle larger events without a significant increase in processing time. This suggests that the system can scale to meet the demands of real-time event detection in IoT applications.

Accuracy: The use of semantic web technologies enabled CEP to achieve higher levels of accuracy and completeness in generating insights compared to the conventional CEP as outlined in Figure 12. By utilising semantic representations, the proposed framework exhibited enhanced contextual understanding, resulting in more precise and suitable insights, leading to better decision making. The semantic-based CEP framework outperformed traditional CEP in terms of accuracy, precision, and contextual relevance due to semantic web reasoning and inference capabilities.

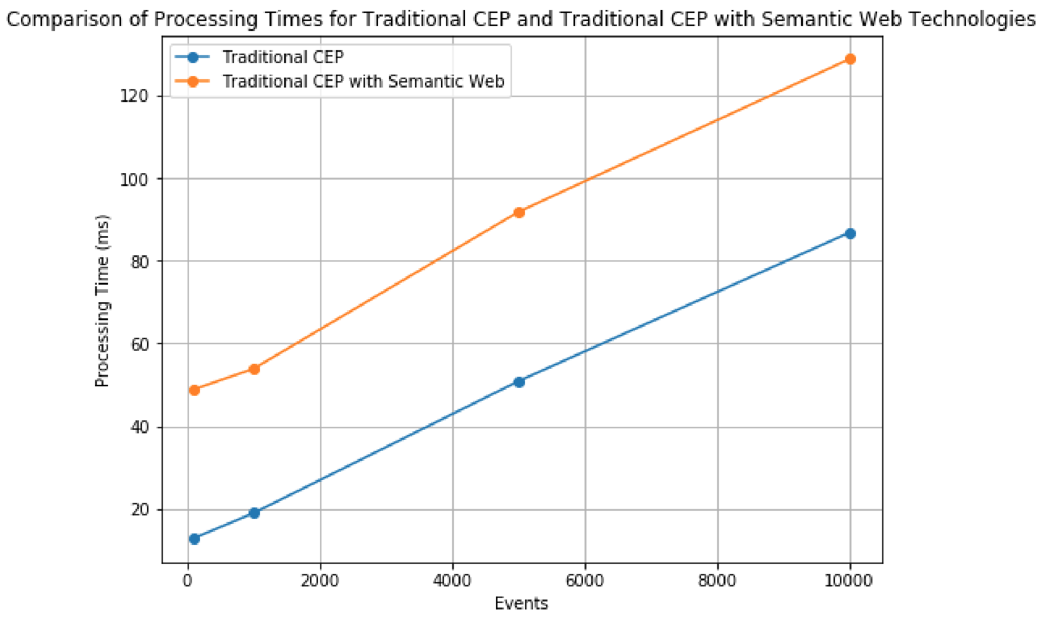


Figure 11. Comparison of processing times for traditional CEP and enhanced CEP.

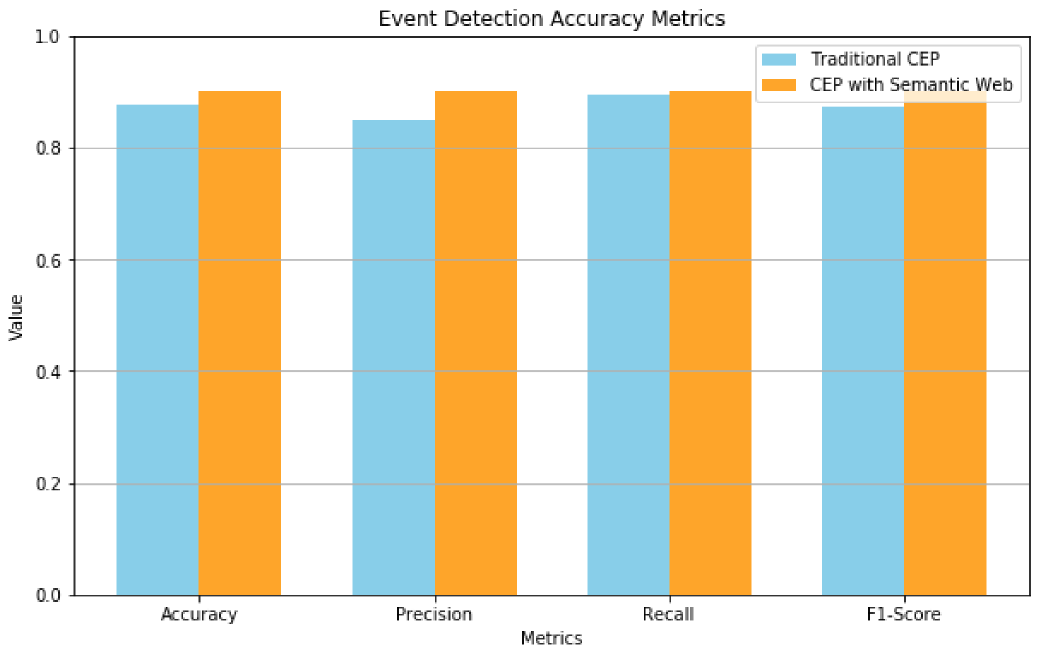


Figure 12. Comparison of detection accuracy of traditional CEP and enhanced CEP.

The performance of two approaches, conventional CEP and CEP enhanced with semantic web technologies, are evaluated side by side using four metrics—Accuracy, Precision, Recall, and F1-Score—which are presented in the chart with blue and orange bars representing conventional CEP and CEP with semantic web technologies, respectively. The chart shows that the proposed enhanced CEP framework outperforms conventional CEP in all four metrics, indicating that the integration of semantic web technologies with traditional CEP significantly improves real-time event detection capacity.

Evaluating both approaches reveals their strengths and weaknesses. Conventional CEP shows strength in processing and resource utilisation for limited environments, but it may not handle complex event relationships or integrate diverse data, limiting its applicability for comprehensive event detection. Conversely, the fusion of semantic web technologies with traditional CEP offers significant benefits to event detection. The integration of semantics provides a more robust data representation, enabling the system to comprehend data at a more abstract level and conduct reasoning over it. This leads to heightened accuracy and completeness in event detection, thereby reducing the occurrence of false positives and false negatives. Furthermore, the utilisation of semantic web technologies has allowed for the effortless integration of data from multiple sources, which has resulted in the system being able to achieve greater contextual awareness and adaptability across a wide range of application domains.

Real-time analysis of large-scale and diverse data streams is crucial for streaming IoT applications. However, the increasing complexity of IoT data poses several challenges that require innovative solutions. One such solution proposed in this article is the integration of semantic web technologies with traditional CEP, which holds enormous potential to address these challenges. By leveraging this integration, complex event patterns can be efficiently handled, enabling timely and informed decision making as well as proactive actions based on real-time insights. A comparative analysis of different approaches can assist researchers and practitioners in selecting the most suitable method for their distinct IoT applications and use cases.

In evaluating the two approaches, we consider both traditional performance metrics and computational burden. Understanding the computational requirements of our methods is essential for practical feasibility. In Table 2, we present the combined computation burden metrics for both Traditional CEP and Semantic Web-integrated CEP.

Table 2. Computation burden metrics for Traditional CEP and Semantic Web-integrated CEP.

Metric	Traditional CEP Approach	CEP with Semantic Web Approach
Avg CPU Usage	70%	45%
Peak CPU Usage	85%	60%
Avg Memory	3 GB	2.5 GB
Peak Memory	4 GB	3.5 GB
Avg Network BW	100 Mbps	80 Mbps
Peak Network BW	150 Mbps	120 Mbps

Comparing the computational burden metrics between traditional CEP and semantic web technology-integrated CEP approaches, we gain a holistic view of their respective performance. This evaluation extends beyond raw speed and accuracy and delves into the practical feasibility of implementation, particularly in resource-intensive real-time scenarios. Our research focuses on real-time data analysis within the context of our CEP framework. We have addressed the computational burden by optimising our algorithms for efficiency, selecting appropriate hardware configurations, and implementing scalability and resource-monitoring strategies. Within the CEP framework, we have harnessed parallel processing, advanced pattern recognition algorithms, low-latency design, and continuous resource monitoring to optimise our CEP framework for superior performance.

5. Conclusions

This article has demonstrated the pivotal role of ontologies in enhancing complex event detection, particularly within the context of air quality monitoring in IoT applications. Our work highlights how ontologies significantly contribute to more precise and context-aware event detection by enriching data semantics, thereby reducing false positives and false negatives and increasing event detection precision. The integration of ontologies with traditional CEP expands the capabilities of event detection, particularly in scenarios characterised by intricate event patterns and diverse data sources. While traditional CEP remains efficient, the addition of semantics improves the quality of insights derived from streaming data. Our evaluation has confirmed the advantages of this integrated approach, offering a robust framework for event detection.

The proposed method for detecting complex events in real-time did not take into account the possibility of anomalies in the data streams. This oversight presents a potential issue as streaming data is constantly evolving, thereby increasing the probability of anomalies and irregular patterns in the data, particularly due to cyber-attacks. Failure to identify these anomalies may lead to erroneous event detection. Hence, it is recommended to investigate techniques for anomaly detection and integrate them with the existing method to reinforce its resilience. Moreover, to improve the performance and effectiveness of the framework, future research should also focus on addressing computational overhead, and streamlining complexity in ontology development. By enhancing the algorithms for anomaly detection, optimising computational efficiency, and developing dynamic ontologies, the integrated framework can become a more efficient and powerful tool for generating real-time insights and detecting events in IoT applications.

Author Contributions: Conceptualization, R.Y.; methodology, R.Y., S.K. (Sohag Kabir), D.T. and S.K. (Savas Konur) writing—original draft preparation, R.Y. and S.K. (Sohag Kabir); writing—review and editing, R.Y., S.K. (Sohag Kabir), D.T. and S.K. (Savas Konur); problem space and formalization, R.Y., S.K. (Sohag Kabir) and D.T.; supervision, S.K. (Sohag Kabir), D.T. and S.K. (Savas Konur); project administration, S.K. (Sohag Kabir). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hung, M. Leading the IoT. Available online: https://www.gartner.com/imagesrv/books/iot/iotEbook_digital.pdf (accessed on 6 February 2023).
2. Ranjan, R.; Thakker, D.; Haller, A.; Buyya, R. A note on exploration of IoT generated big data using semantics. *Future Gener. Comput. Syst.* **2017**, *76*, 495–498. [CrossRef]
3. Yemson, R.; Konur, S.; Thakker, D. A Novel Semantic Complex Event Processing Framework for Streaming Processing. In Proceedings of the 9th International Conference on the Internet of Things, Bilbao, Spain, 22–25 October 2019; pp. 1–4.
4. Khanna, A.; Kaur, S. Internet of Things (IoT), applications and challenges: A comprehensive review. *Wirel. Pers. Commun.* **2020**, *114*, 1687–1762. [CrossRef]
5. Sun, A.Y.; Zhong, Z.; Jeong, H.; Yang, Q. Building complex event processing capability for intelligent environmental monitoring. *Environ. Model. Softw.* **2019**, *116*, 1–6. [CrossRef]
6. Beckstein, S.; Bruns, R.; Dunkel, J.; Renners, L. Integrating semantic knowledge in data stream processing. In Proceedings of the 9th Workshop on Knowledge Engineering and Software Engineering (KESE9), Koblenz, Germany, 17 September 2013.
7. Van Assche, D.; Delva, T.; Haesendonck, G.; Heyvaert, P.; De Meester, B.; Dimou, A. Declarative RDF graph generation from heterogeneous (semi-) structured data: A systematic literature review. *J. Web Semant.* **2022**, *75*, 100753. [CrossRef]
8. Pacaci, A.; Bonifati, A.; Özsu, M.T. Evaluating complex queries on streaming graphs. In Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE), Kuala Lumpur, Malaysia, 9–12 May 2022; pp. 272–285.
9. Steenwinckel, B.; De Paepe, D.; Hautte, S.V.; Heyvaert, P.; Bentefrit, M.; Moens, P.; Dimou, A.; Van Den Bossche, B.; De Turck, F.; Van Hoecke, S.; et al. FLAGS: A methodology for adaptive anomaly detection and root cause analysis on sensor data streams by fusing expert knowledge with machine learning. *Future Gener. Comput. Syst.* **2021**, *116*, 30–48. [CrossRef]
10. Amir, A.; Kolchinsky, I.; Schuster, A. DLACEP: A Deep-Learning Based Framework for Approximate Complex Event Processing. In Proceedings of the SIGMOD'22: Proceedings of the 2022 International Conference on Management of Data, Philadelphia, PA, USA, 12–17 June 2022.
11. Jin, X.; Lee, X.; Kong, N.; Yan, B. Efficient complex event processing over RFID data stream. In Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science (ICIS 2008), Portland, ON, USA, 14–16 May 2008; pp. 75–81.
12. Wu, E.; Diao, Y.; Rizvi, S. High-performance complex event processing over streams. In Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data, Chicago, IL, USA, 27–29 June 2006; pp. 407–418.
13. Wang, F.; Liu, S.; Liu, P.; Bai, Y. Bridging physical and virtual worlds: Complex event processing for RFID data streams. In Proceedings of the International Conference on Extending Database Technology, Munich, Germany, 26–31 March 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 588–607.
14. Wang, Y.; Cao, K.; Zhang, X. Complex event processing over distributed probabilistic event streams. *Comput. Math. Appl.* **2013**, *66*, 1808–1821. [CrossRef]

15. Dong, L.; Wang, D.; Sheng, H. Design of RFID middleware based on complex event processing. In Proceedings of the 2006 IEEE Conference on Cybernetics and Intelligent Systems, Bangkok, Thailand, 7–9 June 2006; pp. 1–6.
16. Cugola, G.; Margara, A. Complex event processing with T-REX. *J. Syst. Softw.* **2012**, *85*, 1709–1728. [CrossRef]
17. Zang, C.; Fan, Y. Complex event processing in enterprise information systems based on RFID. *Enterp. Inf. Syst.* **2007**, *1*, 3–23. [CrossRef]
18. Terroso-Saenz, F.; Valdés-Vela, M.; den Breejen, E.; Hanckmann, P.; Dekker, R.; Skarmeta-Gomez, A.F. CEP-traj: An event-based solution to process trajectory data. *Inf. Syst.* **2015**, *52*, 34–54. [CrossRef]
19. Dunkel, J.; Fernández, A.; Ortiz, R.; Ossowski, S. Event-driven architecture for decision support in traffic management systems. *Expert Syst. Appl.* **2011**, *38*, 6530–6539. [CrossRef]
20. Zhang, Y.; Sheng, V.S. Fog-enabled event processing based on IoT resource models. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 1707–1721. [CrossRef]
21. Sasa, A.; Vasilecas, O. Ontology-based support for complex events. *Elektron. Elektrotechnika* **2011**, *113*, 83–88. [CrossRef]
22. Moser, T.; Roth, H.; Rozsnyai, S.; Mordinyi, R.; Biffl, S. Semantic event correlation using ontologies. In Proceedings of the OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Vilamoura, Portugal, 1–6 November 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 1087–1094.
23. Paschke, A. A Semantic Design Pattern Language for Complex Event Processing. In Proceedings of the AAAI Spring Symposium: Intelligent Event Processing, Stanford, CA, USA, 23–25 March 2009; pp. 54–60.
24. Alirezaie, M.; Renoux, J.; Köckemann, U.; Kristoffersson, A.; Karlsson, L.; Blomqvist, E.; Tsiftes, N.; Voigt, T.; Loutfi, A. An ontology-based context-aware system for smart homes: E-care@ home. *Sensors* **2017**, *17*, 1586. [CrossRef]
25. Kamilaris, A.; Gao, F.; Prenafeta-Boldu, F.X.; Ali, M.I. Agri-IoT: A semantic framework for Internet of Things-enabled smart farming applications. In Proceedings of the 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), Reston, VA, USA, 12–14 December 2016; pp. 442–447.
26. Luschi, A.; Petraccone, C.; Fico, G.; Pecchia, L.; Iadanza, E. Semantic ontologies for complex healthcare structures: A scoping review. *IEEE Access* **2023**, *11*, 19228–19246. [CrossRef]
27. Florea, M.; Potlog, C.; Pollner, P.; Abel, D.; Garcia, O.; Bar, S.; Naqvi, S.; Asif, W. Complex project to develop real tools for identifying and countering terrorism: Real-time early detection and alert system for online terrorist content based on natural language processing, social network analysis, artificial intelligence and complex event processing. In *Challenges in Cybersecurity and Privacy-the European Research Landscape*; River Publishers: Aalborg, Denmark, 2022; pp. 181–206.
28. Guzel, M.; Ozdemir, S. A new CEP-based air quality prediction framework for fog based IoT. In Proceedings of the 2019 International Symposium on Networks, Computers and Communications (ISNCC), Istanbul, Turkey, 18–20 June 2019; pp. 1–6.
29. Díaz, G.; Macià, H.; Valero, V.; Boubeta-Puig, J.; Cuartero, F. An Intelligent Transportation System to control air pollution and road traffic in cities integrating CEP and Colored Petri Nets. *Neural Comput. Appl.* **2020**, *32*, 405–426. [CrossRef]
30. Semlali, B.E.B.; Amrani, C.E. A Stream Processing Software for Air Quality Satellite Datasets. In Proceedings of the International Conference on Advanced Intelligent Systems for Sustainable Development, Tangier, Morocco, 21–26 December 2020; Springer: Cham, Switzerland, 2020; pp. 839–853.
31. AlBalawi, S.M.; Namdeo, A.; Hodgson, S.; Pless-Mulloli, T.; McNally, R.J. Short-term effects of air pollution on daily asthma-related emergency department visits in an industrial city. *J. Public Health* **2021**, *43*, e45–e53. [CrossRef]
32. Caillaud, D.; Keirsbulck, M.; Leger, C.; Leynaert, B.; on behalfof the Outdoor Mould ANSES Working Group. Outdoor Mold and Respiratory Health: State of Science of Epidemiological Studies. *J. Allergy Clin. Immunol. Pract.* **2022**, *10*, 768–784. [CrossRef]
33. Tiotiu, A.I.; Novakova, P.; Nedeva, D.; Chong-Neto, H.J.; Novakova, S.; Steiropoulos, P.; Kowal, K. Impact of air pollution on asthma outcomes. *Int. J. Environ. Res. Public Health* **2020**, *17*, 6212. [CrossRef]
34. Paciência, I.; Cavaleiro Rufo, J.; Moreira, A. Environmental inequality: Air pollution and asthma in children. *Pediatr. Allergy Immunol.* **2022**, *33*. [CrossRef]
35. Gubert, L.C.; da Costa, C.A.; Righi, R.d.R. Context awareness in healthcare: A systematic literature review. *Univers. Access Inf. Soc.* **2020**, *19*, 245–259. [CrossRef]
36. Zhang, O.; Minku, L.L.; Gonem, S. Detecting asthma exacerbations using daily home monitoring and machine learning. *J. Asthma* **2021**, *58*, 1518–1527. [CrossRef]
37. Jalali, L.; Dao, M.S.; Jain, R.; Zettsu, K. Complex asthma risk factor recognition from heterogeneous data streams. In Proceedings of the 2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), Turin, Italy, 29 June–3 July 2015; pp. 1–6.
38. Alfaifi, Y. Ontology Development Methodology: A Systematic Review and Case Study. In Proceedings of the 2022 2nd International Conference on Computing and Information Technology (ICCIT), Tabuk, Saudi Arabia, 25–27 January 2022; pp. 446–450.
39. Hafidh, R.; Sharif, M.S.; Alsallal, M. Smart holistic model for children and youth with special educational needs and disabilities. In Proceedings of the 2019 International Conference on Computing, Electronics & Communications Engineering (iCCECE), London, UK, 22–23 August 2019; pp. 130–135.
40. Staab, S.; Studer, R. *Handbook on Ontologies*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2010.
41. Aminu, E.F.; Oyefolahan, I.O.; Abdullahi, M.B.; Salaudeen, M.T. A review on ontology development methodologies for developing ontological knowledge representation systems for various domains. *Inf. Eng. Electron. Bus.* **2020**, *12*, 28–39.

42. Chatterjee, A.; Prinz, A.; Gerdes, M.; Martinez, S. An automatic ontology-based approach to support logical representation of observable and measurable data for healthy lifestyle management: Proof-of-concept study. *J. Med. Internet Res.* **2021**, *23*, e24656. [CrossRef]
43. Kapoor, B.; Sharma, S. A comparative study ontology building tools for semantic web applications. *Int. J. Web Semant. Technol. (IJWEST)* **2010**, *1*, 1–13. [CrossRef]
44. Noy, N.F.; McGuinness, D.L. *Ontology Development 101: A Guide to Creating Your First Ontology* Stanford University: Stanford, CA, USA, 2001.
45. Stancin, K.; Poscic, P.; Jaksic, D. Ontologies in education–state of the art. *Educ. Inf. Technol.* **2020**, *25*, 5301–5320. [CrossRef]
46. Schultz, D.J. *IEEE Standard for Developing Software Life Cycle Processes*; IEEE: Piscataway, NJ, USA, 1997; pp. 1074–1997.
47. Grüninger, M.; Fox, M.S. Methodology for the design and evaluation of ontologies. In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI-95, Montreal, QC, Canada, 19–21 August 1995.
48. Silva-López, R.B.; Silva-López, M.; Bravo, M.; Méndez-Gurrola, I.I.; Sánchez-Arias, V.G. GODeM: A Graphical Ontology Design Methodology. *Res. Comput. Sci.* **2014**, *84*, 17–28. [CrossRef]
49. Suárez-Figueroa, M.C.; Gómez-Pérez, A.; Fernández-López, M. The NeOn methodology for ontology engineering. In *Ontology Engineering in a Networked World*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 9–34.
50. Zhang, Z.; Yu, P.; Chang, H.C.; Lau, S.K.; Tao, C.; Wang, N.; Yin, M.; Deng, C. Developing an ontology for representing the domain knowledge specific to non-pharmacological treatment for agitation in dementia. *Alzheimer's Dementia Transl. Res. Clin. Interv.* **2020**, *6*, e12061. [CrossRef]
51. de Figueroa, M.d.C.S.; Advisors, B.; Pérez, A.G.; López, M.F. NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse. Ph.D. Thesis, Universidad Politécnica de Madrid, Madrid, Spain, 2010.
52. Pérez, A.G.; Baonza, M.D.F.; Villazón, B. Neon methodology for building ontology networks: Ontology specification. *Methodology* **2008**, 1–18. Available online: http://www.macs.hw.ac.uk/~yjc32/project/ref-ontology/ref-ontology%20building/1-2008-Gomez%20Perez-NeOn-Methodology-OntologySpecification-v3.pdf (accessed on 6 February 2023).
53. Ramírez-Durán, V.J.; Berges, I.; Illarramendi, A. ExtruOnt: An ontology for describing a type of manufacturing machine for Industry 4.0 systems. *Semant. Web* **2020**, *11*, 887–909. [CrossRef]
54. Horridge, M.; Jupp, S.; Moulton, G.; Rector, A.; Stevens, R.; Wroe, C. *A Practical Guide to Building Owl Ontologies Using Protégé 4 and Co-Ode Tools Edition1.2*; The University of Manchester: Manchester, UK, 2009; Volume 107.