Alejandro Llaves Arellano

INTEGRATION OF SENSOR DATA BY MEANS OF AN EVENT ABSTRACTION LAYER

September 2013

Geoinformatik

INTEGRATION OF SENSOR DATA BY MEANS OF AN EVENT ABSTRACTION LAYER

Inaugural-Dissertation
zur Erlangung des Doktorgrades der Naturwissenschaften
im Fachbereich Geowissenschaften
der Mathematisch-Naturwissenschaftlichen Fakultät
der Westfälischen Wilhelms-Universität Münster

vorgelegt von
Alejandro Llaves Arellano
aus Castellón de la Plana, Spanien
im September 2013

Dekan:	Prof. Dr. Hans Kerp
Erstgutachter:	Prof. Dr. Werner Kuhn
Zweitgutachter:	Prof. Dr. Chris Renschler
Tag der mündlichen Prüfung:	18.11.2013.....
Tag der Promotion:	.....

ABSTRACT

Time series of observations reflect the status of environmental properties. Variations in these properties can be considered events when they potentially affect the stability of the monitored environment. An event is anything that happens or is observed as happening at an instant - or over an interval - of time, and which is relevant for the observer. Organisations dedicated to analyse environmental change use institutionalised descriptions of events to define the observable conditions under which events happen. In some applications, like flood monitoring, the response time to emergency events is critical to save lives. Traditional approaches to analyse spatio-temporal data were not designed for the massive amounts of sensor data that are common nowadays. Additionally, the exchange of event-related information among different communities is challenging because of the lack of a universal event model and the ambiguous terminology used to categorise events in some domains.

The goal of this research is to provide a method that facilitates the analysis and integration of sensor data by inferring events from time series of observations. Event definitions are extracted from domain knowledge resources, like scientific papers or technical reports. These definitions are rules based on observable conditions and can be formalised as event patterns. For the analysis of sensor data, I propose to use event processing to detect event patterns in time series of observations. Spatio-temporal properties of the event are inferred from the sensor location and the observation timestamps. The type of the event is provided by the user when the event pattern is registered. For the data integration, I represent event-related information extracted from multiples sources under a common event model. Additionally, I suggest modelling domain knowledge in a layered ontology structure.

A prototype has been developed as a proof of concept. Inferred event instances are published to an event bus, which allows external applications for listening to events. An event model is provided to represent the properties of events inferred from observations. A lightweight ontology for the flood monitoring domain and two application ontologies for the information communities of the use case complete the ontology structure. From a data set, the system is able to infer flood-related events defined by two information communities and to share this information via a knowledge base.

PUBLICATIONS

The main ideas behind this thesis have appeared in the following publications:

Alejandro Llaves and Werner Kuhn. An Event Abstraction Layer for the Integration of Geosensor Data. Submitted to the International Journal of Geographic Information Science, Special Issue on Space-Time Research, 2013

Alejandro Llaves, Henry Michels, Patrick Maué, and Marcell Roth. Semantic Event Processing in ENVISION. In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, pages 25:1–25:9, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-0915-8. doi: 10.1145/2254129.2254161. URL http://doi.acm.org/10.1145/2254129.2254161

Alejandro Llaves and Thomas Everding. Discovering Geosensor Data by means of an Event Abstraction Layer. In Laura Diaz, Carlos Granell, and Joaquin Huerta, editors, *Discovery of Geospatial Resources: Methodologies, Technologies, and Emergent Applications*, chapter 6, pages 112–132. IGI Global, 2012. doi: 10.4018/978-1-4666-0945-7.ch006

Alejandro Llaves. A Theory for Event Processing of Geosensor Data. In Theodor Foerster, Arne Broering, Bastian Baranski, Benjamin Pross, Christoph Stasch, Thomas Everding, and Stephan Maes, editors, *Workshop on Integrating Sensor Web and Web-based Geoprocessing, 14th AGILE Conference. CEUR Workshop proceedings (Vol. 712)*, Utrecht, 2011. CEUR. URL http://ceur-ws.org/Vol-712/paper8.pdf

ACKNOWLEDGEMENTS

First, I would like to deeply thank Prof. Dr. Werner Kuhn for giving me the opportunity of writing my thesis and working at the Institute for Geoinformatics in Münster. During the last four years, he provided full support on my research trying to make things as easy as possible for me. I own him all the experience I gathered as a researcher in Germany.

I also want to thank Prof. Dr. Chris Renschler for his supervision during my time in Buffalo. Working at his research group was a pleasure and helped me to enrich my viewpoint on extreme events and disaster management. I do not want to forget the fruitful conversations with Prof. Dr. Barry Smith and Prof. Dr. David Mark, who gave me the chance of discussing my topic with them. Special thanks to Prof. Dr. Peter Chung and all the colleagues of the GIS Research Center at the Feng-Chia University of Taichung for their hospitality.

This thesis was funded by the ENVISION project. I would like to thank all my project partners, specially those working with me at Ifgi. Without all the discussions and the collaborative work with my colleagues and friends Pat, Henry, and Marcell, this thesis would not have been the same.

I shared very nice moments with the members of the International Research Training Group on Semantic Integration of Geospatial Information. Being part of this group allowed me to meet interesting people from several countries with a common interest in GIScience. I would like to thank our postdocs Jano, Tomi, and Simon for being always up for a talk and for all the constructive comments.

Thanks to the colleagues of the Muenster Semantic Interoperability Lab for the interesting discussions at the brown bag seminars and retreats. In this group I learnt how inspiring and satisfactory can be dealing with interdisciplinary topics in a friendly environment.

I am very grateful to the proofreaders who helped me giving shape to the thesis in the final period: Sven, Jens, Anu, Simon Scheider, Tomi, Carsten, Rui, Carlos Granell, Christoph Stasch, Simon Jirka, and Pat. I am thanking some of them twice, but this is because they deserve it.

Finally, I would like to thank my parents, my sister, and my girlfriend for celebrating with me the good moments and, specially, for their support in the bad ones. The distance between us is what made me always move forward.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

API	Application Programming Interface
CEP	Complex Event Processing
DUL	DOLCE+DnS Ultralite
EDA	Event-driven Architecture
EPL	Event Processing Language
EPS	Event Processing Service
ESP	Event Stream Processing
GIS	Geographic Information System
OGC	Open Geospatial Consortium
OWL	Ontology Web Language
RDF	Resource Description Framework
RDFS	RDF Schema
SAS	Sensor Alert Service
SES	Sensor Event Service
SOA	Service-Oriented Architecture
SOS	Sensor Observation Service
SSN	Semantic Sensor Network
SSO	Stimulus-Sensor-Observation
SWE	Sensor Web Enablement
URL	Uniform Resource Locator
VGI	Volunteered Geographic Information
W3C	World Wide Web Consortium

1 INTRODUCTION

This chapter starts with an introductory example and an overview of the research topic of the thesis. Then, the main motivation for doing research on semantic integration of sensor data is described in section 1.2. Section 1.3 includes the formulation of the research questions. Section 1.4 describes the scope of the thesis and a summary of some research issues not addressed. Finally, the structure of the remainder of the document is outlined.

1.1 OVERVIEW

Weather agencies use different categories to classify environmental phenomena. In the following examples, different rainfall classifications are described for three organisations. The American Meteorological Society suggests classifying the intensity of a rainfall *"at any given time and place"* as LIGHT, if the rate is up to 2.5 millimetres (mm) per hour, with maximum rates below 0.25 mm in six minutes; MODERATE, if the rate is between 2.6 and 7.6 mm per hour, with a maximum rate below 0.76 mm in six minutes; and HEAVY, if the rate is over 7.6 mm per hour or more than 0.76 mm in six minutes.[1]

The British Meteorological Office considers different classifications for rain and rain shower. A rainfall is classified as SLIGHT if the precipitation rate is below 0.5 mm per hour; MODERATE, if the rate is between 0.5 and 4 mm per hour; and HEAVY, if the intensity is above 4 mm per hour.[2]

In Taiwan, the classification used by the Central Weather Bureau, which does not include light or moderate precipitations, defines a rainfall as HEAVY if the accumulation exceeds 50 mm within 24 hours with at least one hour of 15 mm or more accumulated rainfall; EXTREMELY HEAVY, if the rate is above 130 mm within 24 hours; and as subclasses of EXTREMELY HEAVY rainfall, a rainfall can be TORRENTIAL if the rate is between 200 mm and 350 mm within 24 hours, or EXTREMELY TORRENTIAL if the rate is 350 mm or more within 24 hours.[3]

In the last example, it is clearly stated by the agency that the definitions apply to the area of Taiwan. The British Meteorological Office is the United Kingdom's national weather service, so it is assumed that its definitions cover the geographical extent of UK. However,

1 See http://glossary.ametsoc.org/wiki/Rain.

2 See pages 5 and 6 at http://www.metoffice.gov.uk/media/pdf/4/1/No._03_-_Water_in_the_Atmosphere.pdf.

3 See http://www.cwb.gov.tw/V7e/observe/rainfall/define.htm.

in the case of the American Meteorological Society it is not explicitly indicated to which region are they referring: the whole world, America, North America, or the USA. The classification of a rainfall event is highly dependent on the location, so we could ask national weather organisations to get information about classification of rainfall records for small regions. But what happens when we want to analyse such information in a global or a transboundary context? If we think of a global rainfall intensity map, it would probably look different depending on the organisation responsible for creating it. Such map could also be created by an independent organisation by using the different classification criteria provided by several national services and applying them to their corresponding regions. Taking the examples described above, rainfall events with the same intensity could be represented differently (or even not appear on the map) depending on its spatial location. Other problems arise when more complex phenomena are considered, e.g. blizzards. Devaraju and Kauppinen [38] describe three examples of different blizzard definitions provided by two agencies: Environment Canada and NOAA US. How to classify, represent or track a blizzard which started in the USA and crossed the Canadian border if both countries have different rules to define what a blizzard is? These questions serve to illustrate potential problems that may arise when event-related geospatial information about environmental phenomena have to be analysed at a global scale. Models that allow the integration of different views on the same data are necessary to solve such problems.

The comparison of observations over time allows establishing classification criteria that can involve one or more observed properties. These data sets of timestamped values are called time series, if they are ordered in time and measured at regular intervals. Sensors are the main source of time series of observations. A sensor can range from something as simple as a thermometer to more complex devices, like a stream gauge. In this research, the focus is on time series provided by static in situ sensors, which are sensors that are in contact with the medium they are measuring and that do not change their position. One important feature of in situ sensors is that their observations are geolocated. The most common way to detect changes in the environment is by observing and measuring its properties. Variations in these properties can be considered events when they potentially affect the stability of the monitored environment. In this thesis, I will use the term *event* to refer to anything that happens or is observed as happening at an instant - or over an interval - of time, and which is relevant for the observer.

1.2 MOTIVATION

The inference of geospatial information from changes observed in sensor data is challenging for several reasons. In this section, three of those reasons are analysed: the increase of sensor data generation, the spatio-temporal modelling of dynamic phenomena, and the semantic interoperability problems when the information is shared.

1.2.1 *The data avalanche in the Digital Earth*

In 1998, the vice president of the United States of America, Al Gore, described the vision of the Digital Earth [59]. The bottom-up idea depicted a digital representation of our planet which would allow citizens, scientists, and governments sharing and accessing all kinds of geospatial information. Ten years later, Craglia et al. [27] envisaged the *next generation Digital Earth* as a collection of open access interconnected platforms being able to answer questions for both general and domain-oriented problems and addressed to all kinds of information communities. The *nervous system* of the Digital Earth would consist of a network of sensors able to monitor the state of the globe and raise alerts in crisis management situations [33]. Although existing solutions, like Google Earth[4] or NASA World Wind[5], are promising, so far we have not reached Gore's vision regarding access to huge amounts of data (e.g. satellite imagery), exploration of historical data, simulation of future scenarios, and visualisation of complex data [58].

Science has moved from a data-poor to a data-rich environment because of the decrease in the cost of collecting, storing, and processing digital data [104]. The sources of these data are mostly sensors that are becoming common in our environment. Nowadays, sensors are integrated in devices we use daily, such as smartphones, computers, or fridges; and they are also present in highways, traffic lights, buildings, and even in some other natural landscapes [23].

Environmental monitoring is a critical process in areas affected by natural disasters. It is aimed to ensure public safety, to set up continuous information services, and to provide input for spatial decision support systems [114]. Here, the main challenge is the distributed processing and integration of vast amounts of heterogeneous sensor data in real-time. Most current approaches use Web services based on the classic request/response model. Although partly using open Geographic Information System (GIS) standards, they are often unsuitable for the processing of large volumes of data on the fly [114]. This processing can be performed via pull-based models and push services that send out alerts, e.g. if a certain threshold is exceeded, and it requires real-time processing capabilities.

4 Google Earth is available at http://www.google.com/earth/.
5 NASA World Wind is available at http://worldwind.arc.nasa.gov/.

1.2.2 *Modelling spatio-temporal change*

One of the most relevant research questions today is *"How is the Earth's environment changing and what are the consequences for human civilisation?"* [31]. In order to detect such changes and analyse the consequences, we need to observe the Earth's environment. The use of sensor data seems to be a good solution to create a digital representation of the state of the Earth. However, if we want to understand change and prevent potentially negative consequences, we need to interpret these data to extract the relevant information and, of course, to process and store it. Therefore, we need models to represent change.

In general, scientific models include simplifications and approximations to represent aspects of the real world, so that the final model reflects a particular (human) view of reality [31]. One advantage of using spatio-temporal models to represent change is that space and time are two of the most important ordering relations used in human cognition and language for organising knowledge [72].

From an ontological point of view, we can distinguish between entities which are completely present at any time of their existence, e.g. a table, and entities which are made of temporal parts so that, at any time t at which they exist, only their temporal parts at t are present [101], e.g. a race. The former are referred as endurants, whereas the latter are called perdurants. In information systems, endurants and perdurants with unique identities are represented as objects and events, respectively [137]. Traditional spatial information systems represent dynamic geographic phenomena as collections of static objects ordered in time, namely snapshots, thus events are not explicitly represented [22]. This way of modelling does not provide a full picture on how such phenomena affect our environment because events are necessary to capture the mechanisms of change [137, 52].

Although GIS are moving towards spatio-temporal information systems, some domains call for an immediate switch from a static view of our environment to a dynamic focus, for example, environmental change monitoring, transportation, or health. Moreover, the increased use of real-time, mobile and in situ sensors is leading to new potential applications for spatio-temporal data models and systems [52], such as analysis of crowdsourcing information in crisis management.

Modelling geographic phenomena requires a profound understanding of the processes and events related to the phenomena. Usually, these occurrences are strongly connected, so considering just isolated snapshots of the real world is not the best approach to understand what is happening and why. Each observation can be considered as an abstraction (snapshot) of some environmental feature (taken) at a specific moment. If we want to obtain information from the continuous sensor data flows, we need to provide meaning to relevant

fragments of observations (patterns) and analyse where and when they appear.

1.2.3 *Semantic interoperability problems*

The way of accessing geospatial information changed in the last years from local processing and storage to the migration of data and operations to the Web. This would not pose a problem if Web services and data sets were documented properly. However, traditional methods of documentation in GIS do not use machine-readable metadata, causing the loss of context when metadata is shared between systems [85]. Three different levels of heterogeneities are proposed by Bishr et al. [11]: syntactic, schematic, and semantic. Diversity of data syntax or structure may lead to syntactic or schematic interoperability, respectively, but semantic heterogeneity would be caused by different conceptualisations of the same real world entity [80, ch. 3].

In this thesis, I refer to semantic interoperability as the ability of exchanging information so that it preserves the intended meaning with which it was created. Semantic interoperability problems arise when different communities share, integrate, and reuse geospatial data obtained from sensors without a previous agreement on the meaning of such information. The existence of heterogeneous data models together with the often lack of proper metadata pose a challenge for the process of data interpretation. In the case of inferring events from sensor observations, interoperability problems may appear among information communities if there is no agreement on the use of a common vocabulary for the event types, as illustrated in section 1.1. The path going from sensor data to information used by decision makers normally involve varying levels of abstraction, granularity, and organisation. Diverse information communities have different ways to conceptualise the real world, which usually involves the inclusion of stronger assumptions when higher abstraction levels are considered. Several examples of this so called *semantic heterogeneity* are given by Janowicz [72]. For instance, given two services providing wind direction data, one may refer to wind blowing from, whereas the other refers to wind blowing to. Therefore, the surrounding context in data acquisition as well as the assumptions taken in the interpretation should be propagated to upper abstraction levels, to serve the final purpose of supporting the reconstruction of what was originally meant with the data [74]. Geospatial semantics aims at investigating strategies, computational methods, and tools to enhance semantic interoperability, geographic information retrieval, and usability [76, Preface]. In the field of computer science, an ontology is a vocabulary that describes certain reality and a set of assumptions about the intended meaning of such vocabulary [64]. Ontologies of-

ten serve as the agreement that is needed among different parties to avoid semantic interoperability problems.

Addressing interoperability problems is motivated by the heterogeneity of information [80, ch. 3]. A common way to avoid these heterogeneities is by imposing standards. In the field of the Sensor Web, the Sensor Web Enablement (SWE) working group aims at defining data models, encodings, and Web service specifications to overcome issues raised by syntactic heterogeneities [12]. However, semantic heterogeneities present bigger challenges in this field [15], mostly caused by domain-dependent perspectives, multilingual settings, and the lack of application-specific knowledge.

Handling large amounts of environmental sensor data in form of time series can be addressed with event processing tools. Event processing provides methods for reading, creating, transforming or abstracting events [100]. The formalisation of descriptions of events as event patterns allows for creating an abstraction layer on top of observation data. In the event processing field, an event pattern is defined as *"a template containing event templates, relational operators and variables. An event pattern can match sets of related events by replacing variables with values"* [100]. In this thesis, event patterns are rules that define conditions on observed properties (variables). The values for these observed properties are obtained from time series of observations. Event processing tools are also able to manage abstraction layers in real-time, which can be used to improve information integration across different communities. However, the principal problem in this context is the lack of semantic descriptions to define terms for event types and their properties to be understood by multiple applications [47]. There is a need for a common event conceptualisation and a structured set of relations to achieve interoperability among event-driven applications, and to facilitate semantic discovery of geospatial information.

1.3 RESEARCH QUESTIONS

Events are recognised as core geographic concepts that can answer questions about change [87]. The integration of event-related information from various communities is helpful to understand the behaviour of complex environmental phenomena, like floods. The classification criteria for environmental events strongly depends on the domain. Event definitions are extracted from domain knowledge resources, like scientific papers or technical reports. I assume that the domain knowledge to decide what events are relevant for a specific use case is provided by experts. However, the inference of such events from continuous and heterogeneous data streams is a complex task for a person. For this reason, I investigate the different methods to per-

form this task automatically. The results of this research will answer the following question:

RQ1. How to infer events on the fly from time series of observations provided by in situ sensors?

Event processing allows detecting event patterns in time series of observations. The fixed location of in situ sensors serves to geolocate the inferred event. The timestamps of the observations are used to infer the temporal properties of the event. The type of event related to each event pattern is extracted from domain knowledge sources. Additional properties, like the observation data and the source, are also important for a transparent inference procedure.

Semantic interoperability problems arise when information communities exchange event-related information. The reason is that information communities often use ambiguous terms and vocabularies to categorise events. According to Teymourian and Paschke [127], the attempts at creating a foundational ontology valid for all event types have failed so far. In order to address these problems, event models have to accommodate the perspectives of different communities. The second research question analyses these issues:

RQ2. How to represent event-related information so that it can be shared among information communities?

I suggest organising knowledge in a layered ontology structure. The different layers contain conceptualisations specific to an application, a domain, or knowledge common to all domains. The event model, which captures the main properties of events derived from observations, is also part of this structure. All concepts in the domain and application ontologies inherit from the top-level concepts. This helps to integrate the inferred information by restricting the interpretation of terms.

1.4 SCOPE AND LIMITATIONS OF THE THESIS

The goal of this research is providing a method that facilitates the analysis and integration of sensor data by inferring events from time series of observations. In this thesis, I deal with hydrologic time series requested from sensor data services in the context of flood monitoring. With minor changes it should be possible to apply the presented approach to other domains, like landslide monitoring. Sensors providing hydrologic time series often have a fixed position next to the monitored water body. The analysis of data from mobile sensors or human sensors is not addressed in the thesis.

The Event Abstraction Layer is a virtual layer sitting on top of the data layer. The purpose of this virtual layer is to extract event-related

information from time series of observations. The *event abstraction* process refers here to the inference of events as an abstraction of change in the observed phenomena. Event descriptions are manually converted into event patterns. This conversion is constrained by the data sources available. Therefore, event descriptions based on observed properties not present in the available time series are not suitable for this approach. Each event pattern is labeled with an event type, according to domain knowledge. For instance, air temperature going below zero degrees Celsius could be connected to the event of *freezing*. When values in time series fit the conditions of an event pattern, an event of its corresponding type is inferred. As a proof of concept for the presented method, I implemented a prototype of the Event Abstraction Layer.

Some of the aspects related to the main topic of the thesis were not covered by this research. The *propagation of uncertainty* from the sensor data to the inferred events is not addressed in my approach. Most of sensor data services providing environmental data lack machine-readable uncertainty medatada. The UncertWeb project[6] offers tools to manage uncertainty in the context of the Model Web [56].

Event reasoning is out of the scope of this thesis because the focus is on event inference from streams of observations. Enabling the Event Abstraction Layer with event reasoning capabilities can be realised via a semantic reasoner capable of consuming event streams. Domain knowledge describing the potential consequences of an event (or a combination of events) could be formalised as a set of axioms. The stream of events generated by the Event Abstraction Layer would feed the reasoner and new facts could be inferred.

Some event models give special importance to the definition of *roles of the event participants*, above all, those models conceived to represent historical events. Considering the participants on events derived from observations is crucial to attach a spatial location to the inferred event. However, since I only focus on events constructed from data provided by in situ sensors, the spatial location attached to a specific event is the location of the sensor (or group of sensors) that provided the data. Other roles that event participants can play are not relevant for the results of this research.

The presented approach does not include an ontological *representation of the units of measurement* of the sensor data. This fact can cause problems. For instance, when the data sources provide observations in different units. In the current solution, the domain experts in charge of registering the data sources into the system have to take care of this issue.

6 UncertWeb project official website: http://www.uncertweb.org/.

1.5 THESIS OUTLINE

The next chapter introduces the necessary background discussed in the thesis and substantial related work. Chapter 3 describes the Event Abstraction Layer and the extension of an ontology about sensors and sensor networks to model events. The details on the implementation and an architecture overview are given in chapter 4. The description of the use case to test my methodology is presented in chapter 5. In chapter 6, I compare the presented approach to some of the related work solutions. Additionally, the results from two experiments applied to the use case are analysed. The final chapter comprises the conclusion and future work. It answers the research questions raised in the introduction and list some open issues that are not covered in this thesis.

2

BACKGROUND AND RELATED WORK

This chapter starts with an introduction of basic concepts about sensors, environmental monitoring, and Semantic Web. Sections 2.3 and 2.4 present related research on the field of event modelling and processing, respectively. Last section includes an overview on eventing standards.

2.1 SENSORS AND ENVIRONMENTAL MONITORING

A sensor (or geosensor) is a device that measures detectable changes in our environment and can be geographically referenced [27]. The term device has a connotation of physical object that can be mechanical or electrical. This is the reason why in Volunteered Geographic Information (VGI) [57] individuals contributing observations are often referred to as *human* sensors. Sensors can be configured to provide data periodically, hence creating time series of observations. In situ sensing, in contrast to remote sensing, deals with sensors which are in contact with the medium they are sensing. My research focuses on analysing observation time series obtained from in situ sensors.

Some applications require analysing several observation data sources to aggregate and extract higher level information. The components that carry out this kind of tasks are often called virtual sensors. Kabadayi et al. [79] defined a virtual sensor as *"a software sensor as opposed to a physical or hardware sensor. Virtual sensors provide indirect measurements of abstract conditions (that, by themselves, are not physically measurable) by combining sensed data from a group of heterogeneous physical sensors"*. The service to infer events from observations presented in chapter 4 is, in general terms, a virtual sensor.

A wireless sensor network (WSN) consists of a number of sensor nodes that work together to monitor a specific region in order to obtain data about the environment [138]. The application scenarios where WSNs are commonly deployed ranges from natural disaster assessment to health monitoring [138]. Nittel [107] defined geosensor networks (GSN) as specialised applications of WSN technology in geographic space that are able to detect, monitor, and track processes and phenomena.

The technology advances allow us to connect observation data sources with decision makers in real-time, but this is not enough to provide rapid answers. In some domains, responding to certain situations in a timely manner is critical, e.g. an oil spill at sea. The extraction of relevant information helps to find the response to a specific prob-

lem rapidly. A challenge in this context is that data producers are normally not data consumers. The transformation of data into information implies an interpretation procedure, i.e. take on a meaning. Since different information communities can have different application purposes, perspectives, and conceptualisations the results of extracting information can vary depending on the community involved. Therefore, poor metadata management causes misinterpretation and misuse of data.

The term geospatial information community defines a *"collection of people and systems [...] that at least part of the time, share a common digital geographic information language and share common spatial feature definitions. This implies a common world view as well as common abstractions, feature representations, and metadata"* [84]. In the environmental field there are several information communities that, although might share some common goals, use different bodies of knowledge, e.g. hydrologists and geologists. Interoperability problems arise when such communities exchange information globally. The event model presented in this thesis facilitates the solution of these problems.

2.2 SEMANTIC WEB AND SENSOR WEB

The concept of Sensor Web [35] defines a network of distributed and interconnected sensing devices able to monitor uncertain environments. The following analogy gives an idea of the Sensor Web's goal in terms of communication among several platforms using robust protocols: *"the Sensor Web is to sensors what the Internet is to computers"* [34]. The SWE was created to provide a set of standards for managing online sensor networks and the data they produce [12]. The SWE working group defines data models, encodings, and Web service specifications to overcome issues raised by syntactic heterogeneities [12]. As part of the standardisation activities, the SWE working group developed standard specifications to enable sensor data encoding[1], retrieval[2], streaming[3], alerting[4], and notification[5], as well as sensor tasking[6] and encoding of sensor metadata[7] [12]. Such standardisation efforts together with innovation in sensing technologies have pushed the status of the Sensor Web to a mature level [16]. Yet, integration

1 Observations and Measurements (O&M) website is available at http://www.opengeospatial.org/standards/om.
2 Sensor Observation Service (SOS) website is available at http://www.opengeospatial.org/standards/sos.
3 Transducer Markup Language (TML) website is available at http://www.opengeospatial.org/standards/tml.
4 Sensor Alert Service (SAS) best practices document [120].
5 Web Notification Service (WNS) best practices document [121].
6 Sensor Planning Service (SPS) website is available at http://www.opengeospatial.org/standards/sps.
7 Sensor Model Language (SensorML) website is available at http://www.opengeospatial.org/standards/sensorml.

problems are common when dealing with data provided by multiple heterogeneous sensor networks. One of the biggest challenges in this field is dealing with semantic heterogeneities [15].

The term Semantic Web was coined by Tim Berners-Lee to extend the definition of the World Wide Web (WWW) as a Web of (linked) data that can be processed directly or indirectly by machines[10]. The main challenge is to provide well-defined meaning to the Web of documents to enable computers and humans working together. This initiative is led by the World Wide Web Consortium (W3C)[8], which contributes actively to the development and support of standard specifications and technologies such as:

- Resource Description Framework (RDF): data model that allows defining statements in the form of subject-object-predicate triples.
- RDF Schema (RDFS): set of classes intended to give structure to RDF resources.
- Ontology Web Language (OWL): semantic markup language which extends RDF for publishing and sharing ontologies in the Web.
- SPARQL Protocol and RDF Query Language: the query language for RDF.

To contribute to the realisation of the Semantic Web, a list of recommendations are given by the W3C as best practices. They are referenced as Linked Data principles and are supposed to be optimal for the publication, sharing and interconnection of information on the Semantic Web.[9] Linked Open Data (LOD) embraces the Linked Data principles and recommends to add an open license to a data set. Ontologies can be helpful to restrict the interpretation of data in the Web, but they also have their own context. Janowicz [72] suggested using embodiment, sensors, and observations as building blocks to align ontologies and vocabularies contributed from diverse information communities, i.e. grounding the Semantic Web in the Sensor Web.

The Semantic Sensor Web provides a framework for the interoperable exchange and processing of observation data from heterogeneous sensor networks. One of the goals of the Semantic Sensor Web is to solve the interoperability problems detected in the Sensor Web by enriching sensor data and sensor descriptions with spatial, temporal, and thematic metadata [119]. In sections 2.3.4 and 2.4.4, some of the research work that contributed to the growth of the Semantic Sensor Web is reviewed.

8 W3C's website about the Semantic Web is available at http://www.w3.org/standards/semanticweb/.

9 Linked Data principles according to Tim Berners-Lee: http://www.w3.org/DesignIssues/LinkedData.html.

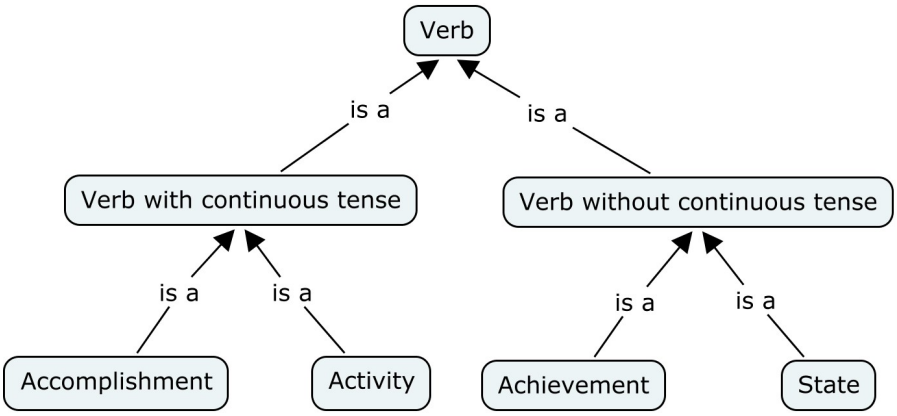

Figure 1: Vendler's classification of verbs.

2.3 EVENT MODELLING

This section presents existing research related to event modelling. First part introduces some views about the nature of events in linguistics, philosophy, and knowledge representation. Section 2.3.2 describes how events are modelled in the foundational ontology DOLCE. In section 2.3.3, modelling approaches in the GIScience domain are explained. The role of events and observations in the Semantic Sensor Network ontology is addressed in section 2.3.4. The last section lists some of the most relevant event models for the general purpose.

2.3.1 *The nature of events*

The nature of events has been the object of discussion in disciplines like philosophy, linguistics, and cognitive sciences for the last decades. This section presents some of the research that contributed to my current understanding of the topic. A more detailed review can be found in [18, 19], the latter including a considerable list of annotated bibliographic entries.

In linguistics, it is common the use of taxonomies for verb classification. Vendler [133] aimed at four divisions of verbs by separating accomplishments and activities (for verbs with continuous tenses, e.g. grow up and walk), and achievements and states (for verbs lacking continuous tenses, e.g. find and love). Figure 1 depicts a concept map with the verb classification introduced by Vendler. Mourelatos [105] modified this classification and suggested to model events and processes as subclasses of occurrences, and accomplishments and achievements as subclasses of events. Figure 2 shows Mourelatos' approach.

In knowledge representation, Allen and Ferguson [5] claimed that events do not exist per se, but that they represent how agents classify relevant patterns of change. This definition points out the importance

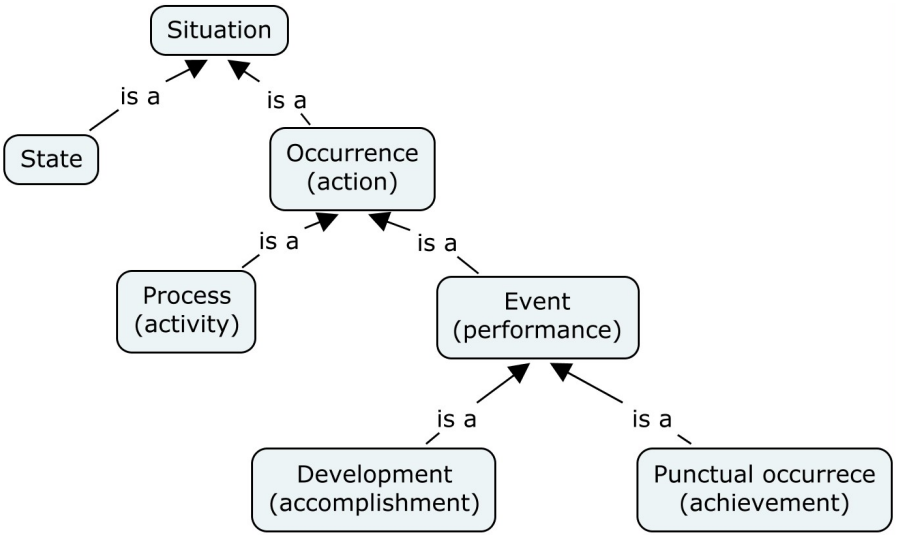

Figure 2: Events, processes, and states according to Mourelatos [105].

of the observer's context for event identification and classification. According to this view, there is no universal law to identify events, since for two agents, an observed change might or might not be relevant. Additionally, it suggests that the classification of events depends on the context of the agent, for instance, the domain of expertise or the application purpose.

Another point of discussion is the distinction between events and processes. Galton [50] claimed that the variation of temporal granularity influences how we perceive occurrences (as punctual or durative events, or as processes) and the dependency relationships between these occurrences. An event occurs over an interval of time and there is no subinterval of that interval over which the event can be said to occur. However, a process is operative at each moment of its existence and, therefore, through each subinterval of any interval over its duration [49]. To distinguish between events and processes, Allen [4] argued that the number of times that an event happens is countable, but the number of times that a process is occurring is not countable. Allen's argument is a simple and effective way of separating events and processes.

The nature of events is relevant to model relations between them. The Interval Calculus defined by Allen [3] includes thirteen qualitative relationships that describe how two events can interact. One of the main features of this calculus is that events are considered to be durative, so it does not contemplate punctual events. It is commonly agreed that models based exclusively on intervals are not complete because of the absence of instants [48]. Ligozat et al. [90] discussed some problems that Allen's calculus does not properly address, suggesting that the choice of the right event calculi depends on the problem to solve.

2.3.2 *Events in DOLCE*

In the context of information science, foundational ontologies (also called upper or top-level ontologies) represent a set of basic concepts and relations among them that can be used across knowledge domains. The main concepts included in the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [101], see figure 3, fit the purpose of modelling the domains of knowledge related to geospatial and geosciences areas [62, 96]. DOLCE+DnS Ultralite (DUL) [54] is a simplification of DOLCE that is used as foundational ontology in my approach. In this section, I describe which is the role of events in DOLCE and DUL.

Based on the previous work of Vendler [133], Mourelatos [105], and Allen [4], among others, Masolo et al. [101] formalised in DOLCE the concept of PERDURANT. A PERDURANT is an entity which extends in time, thus cannot be completely observed at an instant. PERDURANTS can be classified as EVENTS or STATIVES. DOLCE's classification of PERDURANTS is based on their mereological features (see figure 4): when different instances of a PERDURANT are not cumulative, we refer to a STATIVE; otherwise, we talk about EVENTS. A STATIVE can be a STATE or a PROCESS, depending on its homeomericity: a PROCESS is composed of temporal parts that do not represent the whole process by themselves (e.g. running); otherwise, we talk about STATES (e.g. sitting). EVENTS are called ACHIEVEMENTS if they are atomic (e.g. score a goal), otherwise they are ACCOMPLISHMENTS (e.g. buy a book).

The "lite" versions of DOLCE are simplified translations that do not consider modality, temporal indexing, and relation composition.[10] DUL is a simplification and an enhancement of some parts of the DOLCE Lite-Plus ontology[11] and the Descriptions and Situations (DnS) ontology [55]. Following the idea of Allen and Ferguson [5] described in the previous section, DUL focuses on the fact that an event is related to an observable situation, thus different observers may have different views on it. Given various classification criteria, like aspect, agentivity, and participation, DUL does not follow any but suggests the latter. Figure 5 depicts the classification of top-level entities in DUL and the relation between EVENT and OBJECT. This participation relation is useful to attach a spatial location to an EVENT. The core discussion (and the reason to step back from DOLCE's perspective) is about whether dealing with events as observed situations affects the identity of an event. In the method presented in this thesis, event descriptions are derived from observed situations and event identity is not considered.

10 More information about the different versions of DOLCE is available at http://www.loa.istc.cnr.it/DOLCE.html.

11 Documentation available at http://www.loa.istc.cnr.it/Files/DLP_description.txt.

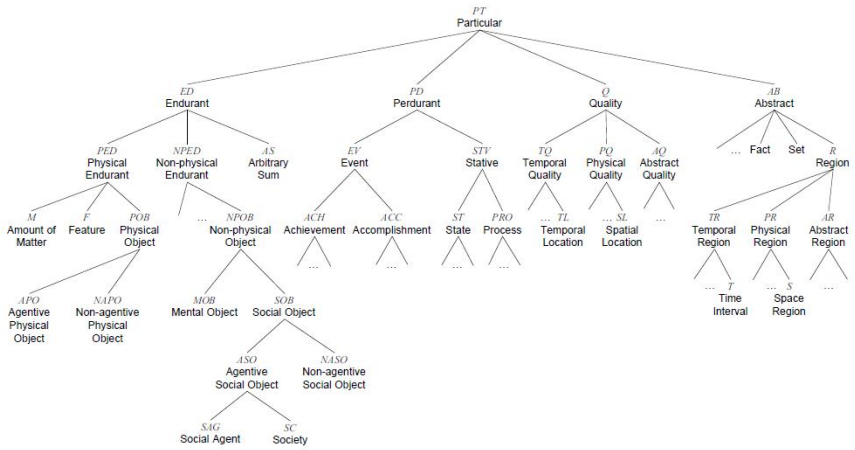

Figure 3: Taxonomy of DOLCE basic categories, from [101].

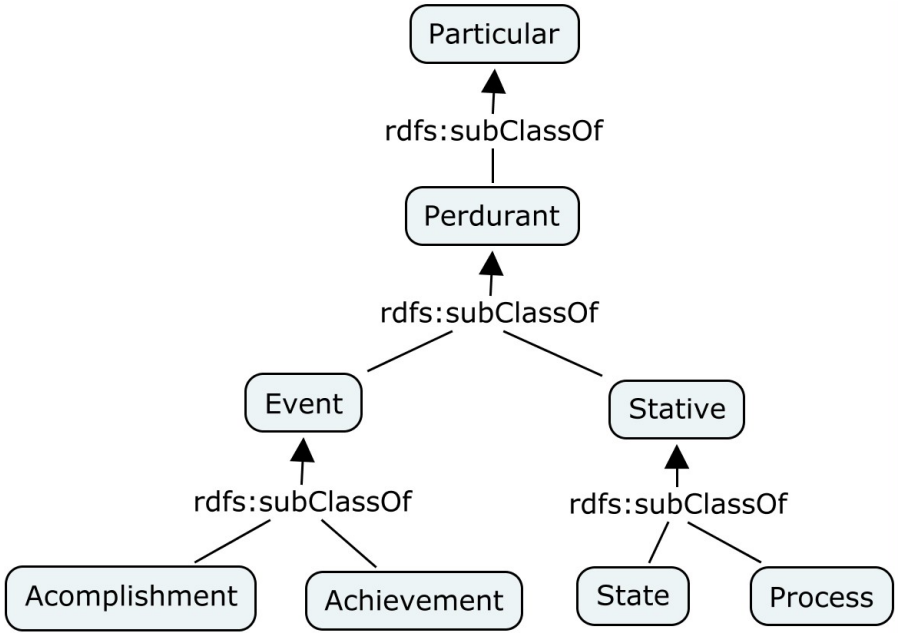

Figure 4: Taxonomy of PERDURANTS in DOLCE.

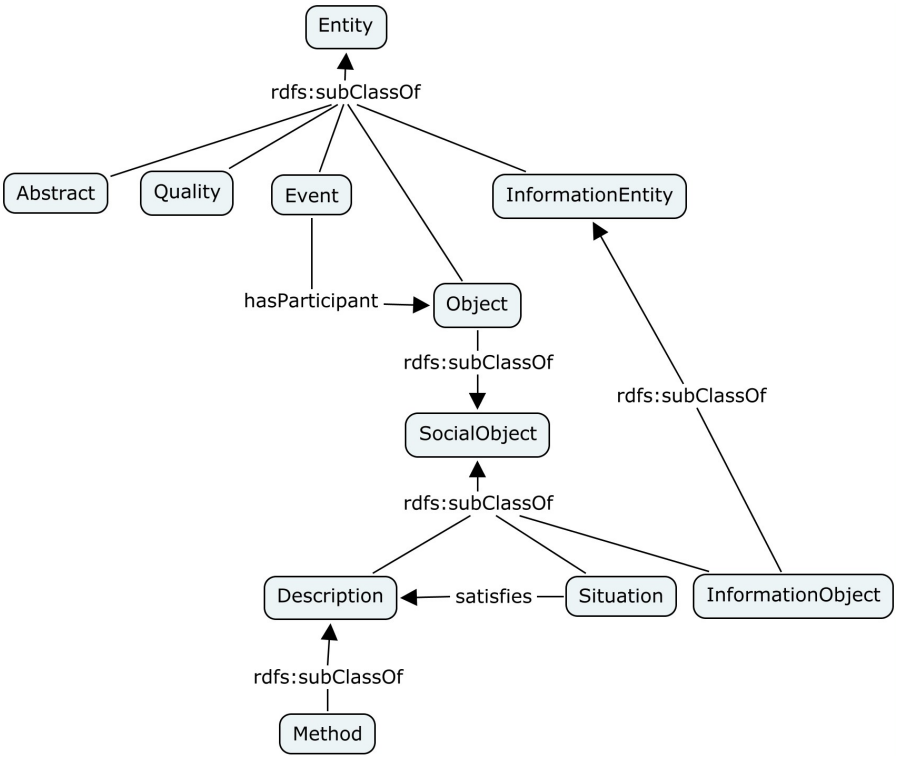

Figure 5: Taxonomy of main entities in DUL.

2.3.3 *Event modelling in GIScience*

This section introduces some of the different event modelling approaches in GIScience that are relevant for my research.

Spatio-temporal models can be divided into two groups: empirical models and dynamic models [69]. The first group includes the traditional spatial information systems mentioned in section 1.2.2 that deal with input, storage, retrieval, management, manipulation, and presentation of data. In the second group, more advanced capabilities are offered, such as analysis, simulation, and prediction. Dynamic models consider the temporal dimension and operate with rules that describe how reality works. The use of event models to represent geospatial information is also characteristic of dynamic models. The model developed for this thesis belongs to the second group of dynamic models.

The Event-based SpatioTemporal Data Model (ESTDM) was created with the aim of easing the analysis of temporal relationships and patterns of change through time [111]. This model represents spatial entities based on fields and temporal entities based on events. At the time of its introduction, most GIS were built upon snapshot models. The ESTDM stores specific changes occurring within a pre-defined geographic area, ordered by time and attached to locations.

The Geospatial Event Model (GEM) introduced the concept of spatio-temporal setting [137]. This model is based on the idea that the snap-

shot paradigm is not enough to explicitly represent changes in our environment. In GEM, events are treated as objects. The main goal is to improve the modelling and management of geospatial phenomena in information systems. While in [135], all entities in the event model are represented as perdurants, in GEM one can distinguish between events, objects and settings. Settings can be spatial, temporal or spatio-temporal. Spatio-temporal settings situate events or objects from a temporal setting to a spatial setting. Worboys and Hornsby claim that geospatial events are situated in spatio-temporal settings. A relevant contribution of this work is the definition of relationships between events, objects, and settings. The limitations of the snapshot paradigm were also used as motivation to develop a generic ontology for modelling, analysing, and retrieving spatio-temporal data about dynamic geospatial phenomena [52]. Galton and Worboys focused on networks embedded in a two-dimensional space and the events that happen or processes that undergo there. An important part of that research refers to the causal relations that appear between events, processes, and states. In my event model, temporal and causal relations between events serve as data provenance indicators. Moreover, relations between events, observations, and sensors (objects) are fundamental to define the event location, although sensors are not explicitly represented in the model.

A different approach was presented by Grenon and Smith [61], where two types of ontologies for endurants and perdurants to represent reality are proposed: SNAP and SPAN, respectively. These two models are interconnected within the framework of the foundational Basic Formal Ontology (BFO).[12] The SNAP ontology represents the relations among enduring entities existing at a moment in time for a given domain and a given granularity, like a snapshot. Thus, each SNAP ontology is indexed to a specific instant of time. On the other hand, SPAN models those entities which *"unfold themselves through time in their successive temporal parts"* for a given domain and at some level of granularity [61]. SPAN entities are not located in space but in space-time. Entities belonging to different ontologies can be integrated via trans-ontological relations. This combined model is intended to bridge the gap existing between ontologies that represent different levels of granularity or based on different entities, e.g. object vs. field. Moreover, it can help to solve problems derived from multi-perspective representations in geographical domains.

The Spatial History Ontology (SHO) extends DOLCE and takes some elements of the CIDOC Conceptual Reference Model (CRM) ontology[13] [62]. SHO is designed to represent different perspectives of the real word given by actors. The goal is to model accounts of his-

12 More information about BFO can be found at http://www.ifomis.org/bfo.

13 The CIDOC CRM ontology is used in cultural heritage documentation. More information is available at http://cidoc-crm.org/.

tories that can be broken down into activities, events and processes. An event in SHO is made of activities performed by its participants within the duration of the event. The reason for choosing DOLCE over BFO was that the purpose of the latter is to provide a formal ontology of reality, whereas in DOLCE it is possible to represent different views on the same fact. Moreover, Grossner claimed that DOLCE is highly appropriate to support GIS applications. SHO supports the representation of historical events derived from news, articles, books, and other documents. One of the purposes of my thesis is finding a solution to represent multiple views on environmental events derived from sensor observations, thus the proposed model for my method present differences to SHO.

Probst [112] suggested to model observations as subclasses of Accomplishment, which is a subcategory of Event in DOLCE. Such observations can be geolocated using (depending on the context) any of the two spatial references of the participant objects in the event: i) the location of the entity that is being observed, or ii) the location of the sensor that performs the observation. This approach allows relating events to spatial locations indirectly, since events in DOLCE only have temporal properties. The temporal properties of the event can be derived from the observation timestamp. A similar approach, adapted to the Semantic Sensor Network ontology, is used in the model presented in this thesis.

2.3.4 *Events and observations in the Semantic Sensor Network ontology*

The Semantic Sensor Network (SSN) ontology [24] describes the domain of sensors and sensor networks. This ontology uses the Stimulus-Sensor-Observation (SSO) design pattern [73].[14] The SSO approach to model observations is based on previous work of Stasch et al. [123] and Kuhn [86]. Kuhn presented a functional ontology of observation and measurement which includes five core elements: Observable, Stimulus, Observer, Observation Value and Observation Process. An Observable is a physical or temporal quality to be observed. It can be inhered in a physical entity or endurant. If the physical quality to be measured is groundwater level, the endurant bearing the quality is the underground body of water. Temporal qualities are inhered in perdurants. An example of temporal quality is the starting time of a rainfall. The concept Stimulus is essential, since an Observable cannot be detected per se. A Stimulus is described as a detectable change in the environment of an Observer. This change can be produced by external agents or by the Observer itself, e.g. a sonar measuring water depth on the sea using sound waves. An Observer perceives a quality and produces a symbol called Observation Value. This symbol is the result of the Observation Process performed by the Observer.

14 More about ontology design patterns in [53].

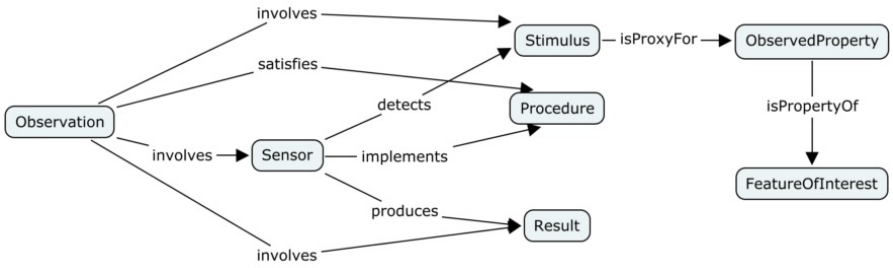

Figure 6: Concept map of the SSO ontology design pattern, from [73].

In [86], the definition of OBSERVER remains open to include any kind of agent with the capability of observing, like sensors or humans. Figure 6 depicts a map with the concepts and relations of the SSO design pattern. Janowicz and Compton [73] described this concept map in the following way: *"Sensors observe properties of features of interest by detecting stimuli, i.e. changes in the physical world, (directly or indirectly) related to these properties and transforming them to another representation as results. Sensors implement a procedure that describes the transformation of stimuli to results. Observations are the context that bring sensor and stimuli together. They are described by procedures that determine how a certain observation has to be carried out"*.

The SSN ontology is aligned to DUL in order to restrict the interpretations of concepts and relations. This implies, for instance, that the concept EVENT used in the SSN ontology is defined in DUL. Opposed to the Probst [112] model, in the SSN ontology observations are not represented as events but as subclasses of DUL:SITUATION[15]. A DUL:SITUATION is a view on a set of entities which is consistent with a description.[16] An SSN:OBSERVATION is a situation in which a sensing method has been used to estimate or calculate a value of a property of a feature of interest.[17] Observations are triggered by stimuli, i.e. events in the physical world. From an ontological point of view, this conceptualisation allows modelling different views on the same event, depending on the observer. Therefore, in SSN and DUL observations can be represented as records of events.

2.3.5 *Generic event models*

In this section, some event modelling approaches for general purpose are described.

15 In the remainder of the thesis, I use the ontology acronym as namespace to differentiate concepts of the two ontologies: DUL and SSN.

16 A complete definition of DUL:SITUATION is available at http://www.w3.org/2005/Incubator/ssn/wiki/DUL_ssn.

17 Definition extracted from from the definition of SSN:OBSERVATION at http://www.w3.org/2005/Incubator/ssn/wiki/Incubator_Report.

The Event Ontology[18] defines the concept of event as *"the way by which cognitive agents classify arbitrary time/space regions"*. An event can have a spatial location, a temporal location, active agents, factors, and products. For spatial and temporal locations the Time Ontology in OWL[19] and the WGS84 Geo Positioning Ontology[20] are used, respectively. The event representation follows the token reification approach, which uses a first-class object to represent every individual event occurrence and a collection of predicates to link the event object to related information [1]. Complex events can be divided into simpler sub-events that would include part of the information belonging to the complex event. The Event Ontology was developed for the domain of information management in music analysis systems, but it can be considered domain-independent.

Ruotsalo and Hyvönen [116] presented an approach to enable interoperability between heterogeneous metadata schemas. The main contribution of this research work is a generic event-based model which consists of a set of relations. Foundational and domain ontologies are used to instantiate perduring and enduring concepts. Connections among such instances are realised via the event-based model. Additionally, a method is provided to transform metadata schemas to the presented event-based schema. The novel idea of this method is to use the domain ontology as a basis for describing, at the same time, the semantics of the metadata elements and the content of the resources [116]. The results of this research were applied to three metadata schemas describing artifacts, paintings, and artists. A recommendation system was implemented to discover new relations across these different data sets.

Event-Model-F [117] is a formal model of events which allows representing space and time, objects and persons, as well as mereological, causal, and correlative relationships between events. It is aligned to DUL and supports modelling multiple views on the same event. For the development of Event-Model-F, two sets of requirements were taken into account: functional and non-functional. The former set refers to what needs to be expressed by the event model and it includes requirements from multiple domains, namely, participation of objects in events, temporal duration of events, spatial extension of objects, structural relationships between events (mereological, causal, and correlational), documentary support for events and objects, and representation of different interpretations of events. Non-functional requirements deal with criteria on how the model has to be designed, including extensibility, axiomatization and formal precision, modularity, reusability, and separation of concerns (domain independence).

18 The Event Ontology is described at `http://motools.sourceforge.net/event/event.html`.

19 The Time Ontology in OWL is described at `http://www.w3.org/TR/owl-time/`.

20 The WGS84 Geo Positioning Ontology is described at `http://www.w3.org/2003/01/geo/`.

The Event-Model-F offers various patterns based on the previously listed functional requirements which support the specialisation of the ontology.

LODE [118] is an ontology for Linked Open Descriptions of Events. Shaw et al. define event as anything that happens over a limited extent in time and that has been reported as an event by some agent. Using linked data, LODE represents event-related information to answer the following questions: what happened, where, when, and who was involved. According to the authors, these answers represent *intersubjective consensus reality*, but in our opinion, different interpretations can be given for each of the listed questions. Properties for event categorisation or event-event relations, e.g. causality, are not included in the model. LODE has a core concept EVENT and a list of properties with focus on spatio-temporal and participation aspects of the represented event. The main contribution of this research is the comparison between existing event models (CIDOC CRM [39], ABC Ontology [88], Event Ontology, EventsML-G2[21], DUL, Event-Model-F, and OpenCYC[22]) with the goal of building a language-independent model that solves interoperability problems via mappings between existing event ontologies.

Van Hage et al. [132] defined events as everything that happens, including fictional events. The purpose of the Simple Event Model (SEM) [132] is modelling events in various domains such as history, cultural heritage, multimedia, and geography, without making assumptions about the domain-specific vocabularies used. The model is intended to be used in the context of the Web of Data, thus it is based on *weak semantics* in order to avoid restrictive conditions. Among other features, SEM also allows to handle different viewpoints on the same event (with respect to roles, time validity, and source of information). For the categorisation of events, the use of classes or individuals from other ontologies as event types is permitted. The mapping between event models can be realised using SKOS.[23] SEM includes four core classes with an associated type: EVENT, ACTOR, PLACE, and TIME. Every class in the model is optional and can be duplicated. Three types of constraints can be applied to properties to restrict its validity: ROLE, TEMPORARY, and VIEW. As opposed to Event-Model-F, SEM does not model relations between events, like causality. However, spatial indexing is supported for SEM instances. The integration of event instances in the Linked Open Data Cloud is facilitated via a Prolog Application Programming Interface (API).

The models presented in this section try to cover all types of events in all domains. In some cases, this implies models that are too flexible to serve interoperability purposes. One example is SEM, in which

21 http://www.iptc.org/site/News_Exchange_Formats/EventsML-G2/

22 http://www.cyc.com/platform/opencyc

23 More information about SKOS available at http://www.w3.org/2004/02/skos/.

every class in the model is optional. The solution of Ruotsalo and Hyvönen is more a mapping between other models than an event model per se, thus it does not clearly define a conceptualisation that can be used for the purpose of this thesis. Nevertheless, models like the Event Ontology and the Event-Model-F include concepts and relations that appear in my model because they are implicit to the common understanding of what is an event, e.g. the concept event itself and its spatio-temporal location. Although my event model is domain-independent, it is designed to represent events derived from environmental sensor observations, which results in specific design choices. For instance, the concept of AGENT used in the Event Ontology has no counterpart in my model because it is assumed that human agents do not participate in the environmental events that I intend to model. Concepts like PRODUCT ("everything produced by an event") or FACTOR ("everything used as a factor in an event"), also included in the Event Ontology, are not present in my event model for being too ambiguous. The reason for not reusing Event-Model-F in my approach lies on its complexity. As described above, it offers multiple design patterns to support modelling requirements which may cover all the aspects of events, but the amount of possible choices is a bit overwhelming. LODE offers a simple and elegant event model, but it does not handle different interpretations of events. Since my idea is to integrate multiple types of events inferred from data, considering the various perspectives of information communities is mandatory. For all these reasons, I opted for an ad-hoc solution by extending the SSN ontology, described in section 2.3.4. The resulting event model is described in section 3.4.

2.4 EVENT PROCESSING

Event processing consists in working with representations of events. The Event Processing Glossary [100] defines event processing as *"computing that performs operations on events, including reading, creating, transforming, or discarding events."* A common example for the use of event processing are graphical user interfaces. They listen to user inputs, generate event representations, and initiate actions. Traditionally, event processing has been used to manage business events that are communicated in the form of messages among the different layers of an IT infrastructure. In the last years, there have been attempts to apply event processing techniques to other domains that need to handle other types of situations than those occurring at the enterprise level, like environmental monitoring.

Information Flow Processing (IFP) refers to the applications in which information generated by heterogeneous sources needs to be collected and processed in a timely manner [29]. The goal of IFP is creating new knowledge as soon as the information is processed. Two models have

emerged among all the existing solutions in the IFP field: Data Stream Processing [8] and Complex Event Processing (CEP) [98]. Stream processing systems oriented to deal with events are called Event Stream Processing (ESP) systems. In the following section, I talk about ESP and CEP. Common architectures and the communication paradigm in event processing are introduced in section 2.4.2. Section 2.4.3 deals with the concept of semantic event processing and some related work. Finally, existing solutions to infer events from observations in the GI-Science domain are analysed.

2.4.1 *ESP vs. CEP*

An event stream is a sequence of events ordered by time. ESP has its origins in active databases and data streams management [100]. A database executes varying queries on a more or less static data set, whereas in ESP the queries are static and executed on a highly dynamic data set. One problem that comes with this approach is that the amount of events that has to be processed may be infinite. Thus, it is not possible to take all events into account for the processing. One solution consists in using views (also called windows) that restrict the available events, for instance to the newest one hundred or to those received in the last minute [8].

CEP is a specialisation of event processing. It deals with the processing of complex events (and not the complex processing of events) [98]. A complex event is an event that has relations to other events like "caused by". For instance, a wild-fire event might be caused by a dry-weather-period event in combination with a dropped-burning-cigarette event [93]. The wild-fire event is complex as it consists of other events. Those events may themselves be complex, like the dry weather event, but also simple, like the cigarette event. CEP can be used to detect certain relations between events. In the example of the wild-fire event, one could try to identify the cigarette that was dropped to find the source of the fire. CEP can also be used to generate new information from sets of events. In this case it could be discovered that during a dry weather period a cigarette was dropped resulting in a high risk for a wild fire. Causality relations are common in CEP, but there are more types of relations, such as temporal relations (e.g. "before" or "after") and spatial relations (e.g. "within area") [98]. The rules for defining events in CEP are called event patterns. They are formalised using an Event Processing Language (EPL). The process of resolving these patterns in sets of events is called pattern matching and performed by software components called pattern matchers [98]. CEP was initially designed to extract information from the events flowing through the layers of the enterprise IT infrastructure, to understand its impact on high level management goals and business processes, and to act upon this information in real-time.

Nowadays, it is also used in the development of applications that infer events from sensor data, e.g. the Sensor Event Service (SES) [41].

ESP and CEP overlap in some aspects, but ESP is considered as a subset of CEP. The reason is that ESP is addressed to event streams whereas CEP is able to process event clouds [99]. An event cloud may contain several unordered event streams, e.g. from heterogeneous sensor networks monitoring a city, thus it requires more complex analysis tools to discover potential relations between events.

2.4.2 *EDA, SOA and the pub/sub paradigm*

Many event processing applications are implemented via an Event-driven Architecture (EDA) [103, 20]. In these architectures, the communication is realised by means of events [47]. However, during the last years, the number of Service-Oriented Architecture (SOA) [43] implementations has increased considerably. In SOA applications, different capabilities like processing and data storage are distributed on different computers and provided as Web services [130]. It is important for the usability of a SOA implementation that the services use well-known standards. Otherwise, the offered service cannot be accessed by external applications, e.g. clients or other services. The interaction between SOA and EDA, sometimes referred as Event-driven SOA or SOA 2.0, can happen in two different manners [103]: either the occurrence of an event triggers the invocation of one or many services, or the output of a service generates an event. The publish-subscribe interaction paradigm (pub/sub) [46] usually appears in EDA and Event-driven SOA systems. In figure 7, the different boxes represent the service components involved. The arrows represent the information flow (messages) in a common scenario:

1. A subscriber expresses interest in a specific event or pattern of events.
2. The broker responds to the subscriber on its request.
3. The publisher publishes events via the broker and is decoupled from subscribers and receivers.
4. If any information matches the interest expressed by the subscriber, a notification is sent to the receiver. In many cases, the subscriber and the receiver belong to the same component.

The power of this event-based interaction paradigm resides in the full decoupling in time, space, and synchronisation between publishers and subscribers/receivers[24]. The prototype presented in chap-

24 In [46], the authors do not distinguish between subscriber and receiver. In case of having different components assuming these roles, they will be decoupled in time and synchronisation, but not decoupled in space, meaning that the subscriber needs to have a reference to the receiver in order to perform the step 1.

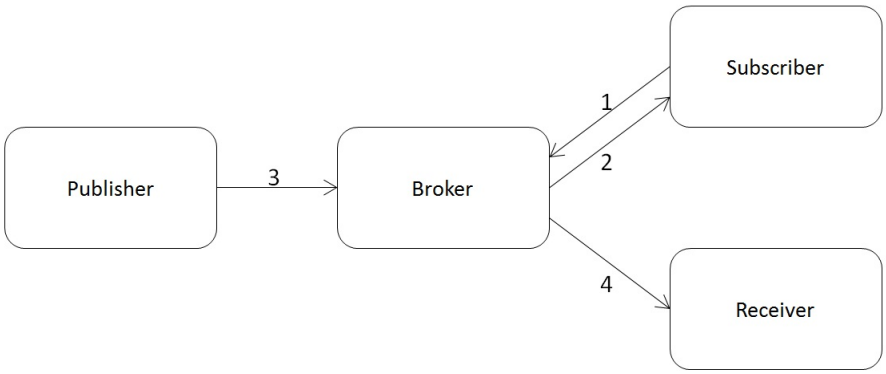

Figure 7: Publish-subscribe interaction paradigm (adapted from [93]).

ter 4 has an Event-driven SOA architecture and follows the pub/sub paradigm to guarantee a loosely-coupled communication between the components.

2.4.3 *Semantic event processing for general purpose*

Semantic structures common to different event processing systems exist and can be used to create a semantic model of event processing Etzion [44]. In this research direction, Teymourian and Paschke [127] defined the concept of *semantic event processing* as the combination of event processing and semantic technologies which allows event processing engines to *"understand what is happening in terms of events and states"*, and to react appropriately. Any system implementing semantic event processing is supposed to include a static knowledge module about the predefined event types, and a real-time analysis module which processes data streams searching for event patterns [127]. In Teymourian's Ph.D. thesis [129], the research goal is to develop a representation methodology for CEP that integrates domain and application ontologies for events, processes, states, actions, and other concepts that are related to change over time. The hypothesis of this dissertation is that the use of ontological knowledge combined with event processing and stream processing techniques enhances the procedure of detection and processing of events. For the evaluation, a healthcare scenario was selected where body sensors connected to patients deliver multiple types of events.

Another example of a semantic event processing system is ETALIS [7], which enables monitoring and specification of changes in near real-time and it is able to perform reasoning over streams of events with respect to background knowledge. In the work of Anicic et al. an event is something that happens or changes the current state of affairs. Two languages are suggested for the specification of event patterns: ETALIS Language for Events (ELE) and Event Processing SPARQL (EP-SPARQL). ELE implements the relations between events defined

in Allen's Interval Calculus [3]. The example used to demonstrate ELE capabilities describes a system to identify traffic bottlenecks. The events analysed to infer a bottleneck event include information about the road, current traffic speed, and sensor location. In order to evaluate whether the events denote slow traffic conditions, and whether events happened in the same road and in the same area, a knowledge base in Prolog is set up. EP-SPARQL enables ETALIS to be used in real-time Semantic Web applications. It extends ELE to allow defining complex event patterns in a SPARQL-like language which are continuously evaluated. Event streams are encoded as RDF triples and background knowledge in RDFS. An example of query could consist in searching for roads where a certain number of traffic events have been reported within a period of time.

SCEPter performs Semantic Complex Event Processing (SCEP) of streaming and archived events using a hybrid solution which combines a CEP engine and a database [139]. For demonstration purposes, the research addresses challenges arising in the domain of Smart Power Grids, where a vast amount of data is generated and exchanged among heterogeneous systems. The proposed event model uses one standard concept to represent a property, e.g. AIRFLOW, and the equivalent concepts are represented as subclasses, e.g. FLOWRATE, AIRRATE, or AIRVOLUME. This is intended to provide integration of data between heterogeneous schemas [139]. Additionally, it includes domain ontologies to model knowledge related to electrical equipment, locations, and buildings, among others. The language used for the patterns is based on SPARQL and supports querying on streams and on archived data. The SCEPter engine is based on the Siddhi CEP engine[25] which performs event processing by using sliding windows, but does not allow batch windows. SCEPter extends Siddhi with an ontology model, stream and query register, annotation module and semantic filter module.

A comparison of my semantic event processing method with the methods described above can be found in section 6.2. The description of other relevant event processing systems (like Cayuga [36] and Drools[26]), event query languages (like XChangeEQ [42]), and rule languages (like Prova[27]) has been excluded because of their non-explicit use of semantic technologies.

2.4.4 *Inferring events from sensor observations in GIScience*

The use of event processing techniques to deal with representations of geospatial events is not a novel concept, but it is not fully established in the GIScience field yet. Traditional views on spatio-temporal mod-

25 More information about Siddhi is available at http://siddhi.sourceforge.net/.
26 More information about Drools can be found at http://www.jboss.org/drools.
27 More information about Prova is available at https://prova.ws.

elling impede the adoption of event processing since it was conceived to deal with business events. However, what an event processing engine does is managing event representations as messages with special temporal properties. The developer or the knowledge engineer designs the structure of the event message. This design flexibility opens the possibility of using the same event processing tools to manage, for instance, representations of bank transactions, car crashes, or landslide occurrences. The decision of giving more or less relevance to the spatial properties, or of considering a timestamp or an interval to represent an event will depend on the application purpose.

Some researchers have demonstrated that is possible to apply data stream mining to infer event-related information from streams of data. The research work of Croitoru [28] has two goals: developing a data-driven method to discover event hierarchies in spatio-temporal data, and studying how events can be used for top-down spatio-temporal data mining. Croitoru claimed that, although change is continuous in our environment, we manage to perceive it discretely as a set of events. Additionally, humans tend to organise events into hierarchies. Among other things, this research work differs from others on the analysis of punctual observations provided by scattered sensors as opposed to dense concentration of points; on the utilisation of a range of scales to discover the hierarchical structure of the data set instead of focusing only on the optimal scale; and on matching hierarchical patterns across sensor observations as opposed to deriving patterns from multiple sensor observations. An example to illustrate how the method works is based on the monitoring of a storm by analysing sensor data provided by the GoMOOS network of buoys[28]. With the conversion of spatio-temporal data into tree structures, the problem of event schema mining is reduced to an ordered tree matching problem. Applying this method to the storm scenario, it is possible to identify an event footprint (hierarchical structure) across different sensors. Using a similar scenario also based on the GoMOOS network, Rude and Beard [115] presented a method to infer high-level events, like storms, from time series of geolocated data. A primitive (or low level) event is defined as a *"subsequence of a time series for which a particular property of a parameter holds, typically indicating a state or change of state over a temporal interval"* [115]. Primitive events are univariate and easily detectable by sensors, and are used as building blocks for multivariate high-level events. Moreover, primitive events can be related to a high-level event by means of three properties: INITIALISES, FORMSBODYOF, and TERMINATES. For example, a river flooding event can be detected by the primitive events of "exceeding high water level threshold" and terminated by "recovery to normal water level".

One example of a semantic event processing system in the GI-Science domain is the Semantic Sensor Observation Service (SemSOS)

28 More information available at http://oceandata.gmri.org/.

[71]. SemSOS is an enhancement of Sensor Observation Service (SOS) that provides a more meaningful representation of sensor data. The method proposed consists in various steps that include: developing a set of ontologies to model the domain of sensors and sensor measurements, enriching sensor observations with semantic annotations, and using the ontology models to perform rule-based reasoning over the annotated observations in order to obtain new knowledge. For the implementation of SemSOS, the SOA is extended with a semantic knowledge base. SemSOS allows executing temporal and thematic queries on historical sensor data by using SPARQL. Results can be used to construct observation collection responses encoded in O&M.

Based on previous work on SemSOS, the Real-Time Feature Streams Infrastructure (RTFS) allows converting real-time heterogeneous sensor data into a stream of high-level events (also called features) [108]. Reasoning over these events is possible by using background knowledge. RTFS starts by transforming raw sensor data to streams of RDF triples. The system abstracts high-level events from the RDF triples with the support of domain ontologies, like a weather ontology. For instance, a RainStorm event has properties HighWindSpeed, RainPrecipitation, and NonFreezingTemperature. As a result, streams of events are generated out of time series of sensor observations and published on the Linked Open Data Cloud.

The Sensing Geographic Occurrences Ontology[29] (SEGO) represents the relations between observations and geographic occurrences reflected in them [37, 38]. This model uses DOLCE as a foundational ontology. Geographic processes act as stimuli that trigger sensors, whereas geographic events are inferred from observations. A geographic event may have parts which are other events, and each of these sub-events can be constituted by geographic processes. The elements that participate in geographic events play different roles. In SEGO, roles represent functional relations and this helps to formulate more symbolic observational queries. A *geo-event* is related to the observation domain through its participants, which bear the observed properties. Institutionalised descriptions [113] define geographic events based on observed properties. The use of institutionalised descriptions of events is an important part of my method, as described in chapter 3.

The solutions reviewed in this section are compared to the method I developed in section 6.2.

2.5 EVENTING STANDARDS

Web Services Notification (WSN) from OASIS[30] is a set of three standards. First, Web Services Base Notification (WS-BaseNotification)

29 Ontology documentation available at http://anusuriya.com/sego/SEGO.htm.
30 https://www.oasis-open.org/

[60], defines operations for Web services to subscribe to certain events, to deliver events to a consumer, and some further operations for alternative event delivery and management of subscriptions. Web Services Brokered Notification (WS-BrokeredNotification) [21] can be implemented on top of the basic operations. It defines additional methods to register publishers at a service that provide new events and manage these registrations. The third specification, Web Services Topics (WS-Topics) [131], defines the use and definition of event topics that can be offered by a Web service. An individual topic can provide access to different events related to the topic.

Web Services Eventing (WS-Eventing) [32] defines a protocol for a Web service (subscriber) to express interest in another Web service (event source) for receiving notifications about events. The subscriber can manage subscriptions via another Web service, named subscription manager, whose reference is provided by the event source. The WS-Eventing specification is a W3C Recommendation. In general, it is similar to the WSN standards suite but with reduced complexity and less restrictions. This offers more freedom when using the standard, but may also lead to incompatibilities between different implementations.

Another standardisation organisation working on eventing standards is the Open Geospatial Consortium (OGC). Various specifications has been released for eventing under the umbrella of SWE. The Web Notification Service (WNS) [122, 121] is a Web service interface that allows receiving events via HTTP and forward them via email or SMS. It is mainly used to notify human decision makers. The Sensor Alert Service (SAS) [120] is a service specification that allows the subscription to sensor measurements using simple filter capabilities. The SES [41] is the natural successor of The SAS and its interface specification is currently available as an OGC discussion paper. The SES implements WS-Notification and offers enhanced filtering capabilities, including CEP and ESP features. In the near future, OGC aims at producing a standard that defines how to enable publish/subscribe functionality for all OGC Web Services in a common way.[31]

31 More information about the work of the PubSub SWG is available at http://www.opengeospatial.org/projects/groups/pubsubswg.

3

INFERRING EVENTS FROM IN SITU SENSOR OBSERVATIONS

This chapter introduces the theoretical aspects behind my method. It includes my definition of event and event abstraction layer, explains the semantic annotation of event patterns, and finally, presents the Event Abstraction ontology.

3.1 MY PERSPECTIVE OF EVENTS

This research focuses on how events can be inferred from time series of sensor data. In chapter 1, I defined an event as anything that happens or is observed as happening at an instant - or over an interval - of time, and which is relevant for the observer. Time series of sensor data reflect the status of environmental properties. Such time series are provided by in situ sensors. I assume that sensor observations are geolocated and chronologically ordered. Sensors allow us measuring properties of phenomena that we, as humans, cannot measure. Sensors are also used to observe remote places that are inaccessible or dangerous for humans, thus saving costs and, potentially, lives. Therefore, sensors are an additional tool that we have to observe our environment. Changes in the observed environment are relevant when they threaten the stability of the environment.

Organisations dedicated to analyse environmental change use descriptions of events to define the observable conditions under which events happen. In CEP, event descriptions are formalised as rules that take the name of event patterns. An event pattern is a template containing relational operators and variables [100]. Event processing tools allow defining event patterns to extract those fragments of data that are considered relevant in a given context. The complexity of event patterns may range from simple rules that describe an observation value above certain threshold, e.g. earthquake magnitude above 5.5, to complex patterns that aggregate values from different sources within a spatio-temporal extent, e.g. average of all precipitation measurements during the last 24 hours for a specific region. When sensor data match an event pattern, an event is inferred.

In computing systems, events are represented as objects. Following the guidelines of the Event Processing Glossary [100], I overload the term *event* to refer to events and their object representations: *"It is tempting to introduce two separate terms such as event and event object. However, in any discussion longer than a paragraph or two, this becomes intolerably clumsy and one finds the distinction being misused, forgotten or*

dropped altogether. For example, using the two separate terms would dictate that event processing [...] should be event object processing. The best solution is to overload the word event. The context of each use becomes the indicator of which meaning is intended.".

The granularity of events is relative to the application purpose. A cosmologist may consider events with a duration of ten years as punctual events because the temporal extent of the research scenario extends over thousands of millions of years. On the contrary, the temporal extent of experiments in molecular biology is normally much smaller and events are finer grained. CEP allows combining simple events to construct more complex events. In my method, events inferred from observations are simple events. Complex events are inferred from other events, which can be simple or complex.

3.2 THE EVENT ABSTRACTION LAYER

Some environmental monitoring applications require to sense and respond to certain changes. In these settings, it is necessary to shift the focus from analysing raw streams of observations to the analysis of higher level pieces of information that indicate change, namely events. Methods that infer and integrate event-related information from available data sources can reduce the response time in emergencies. As described in chapter 1, the main motivation of this research is enabling sensor data integration and semantic interoperability among event-driven applications and systems. In order to do so, a virtual layer has been designed.

The Event Abstraction Layer infers events from time series of observations. Figure 8 illustrates the information flow around the Event Abstraction Layer. On the left side, sensor data services provide time series of observations. The layer analyses time series looking for traces of event patterns. Event patterns describe the conditions under which events happen. When the sensor data reflect such conditions, events are inferred. The Event Abstraction Layer handles mappings between event patterns and event types. These mappings are provided by users from classifications present in domain knowledge. The type of the event is taken from the corresponding mapping of the detected event pattern. Spatio-temporal properties of the event are inferred from the observations. Since the devices providing the observations are in situ sensors, the spatial location of the event is extracted from the sensor locations. The temporal properties of the event are inferred from the timestamps of the involved observations. Additional properties, such as the sensor data used and the source, are also relevant for making the inference procedure more transparent.

One disadvantage of the Event Abstraction Layer is that event patterns are dependent on data sources. This means that the formalisation of event patterns has to be adapted to the available sensor

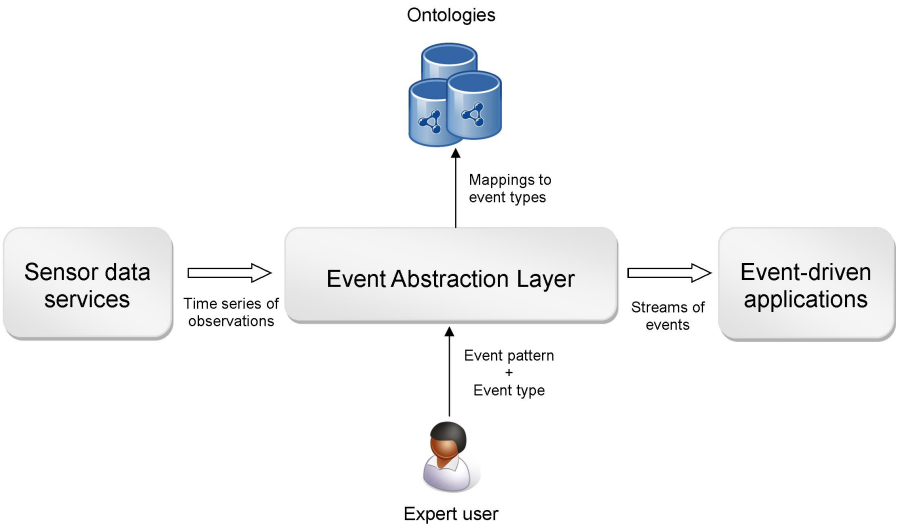

Figure 8: The Event Abstraction Layer analyses time series of observations and generates streams of events.

observations in terms of observed properties, units of measure, and sampling frequency.

The Event Abstraction Layer bridges the gap between sensor data services and event-driven applications. Therefore, event-driven applications do not need to deal with raw sensor data. On the one hand, the Event Abstraction Layer consumes time series of observations. It also handles mappings between event patterns and event types, so that inferred events are automatically classified according to the provided domain knowledge. On the other hand, the Event Abstraction Layer generates a stream of events that are represented under the same model, which is described in section 3.4. Data integration consists in providing users with a uniform view on data residing at different sources [89]. The Event Abstraction Layer enables data integration by representing event-related information extracted from multiple sources under a common event model.

3.3 SEMANTIC ANNOTATION OF EVENT PATTERNS

The *semantic annotation of event patterns* consists in labeling event patterns with event types defined in ontologies. Event types represent categories of a classification related to one or more observed properties, e.g. heavy rainfall. Event patterns describe events quantitatively based on observed properties, e.g. rainfall intensity above 4 millimetres per hour. The mapping between event patterns and event types is extracted from domain knowledge. Figure 9 shows two annotations of event patterns for HEAVY rainfall based on the classifications from two information communities, see section 1.1. The links between

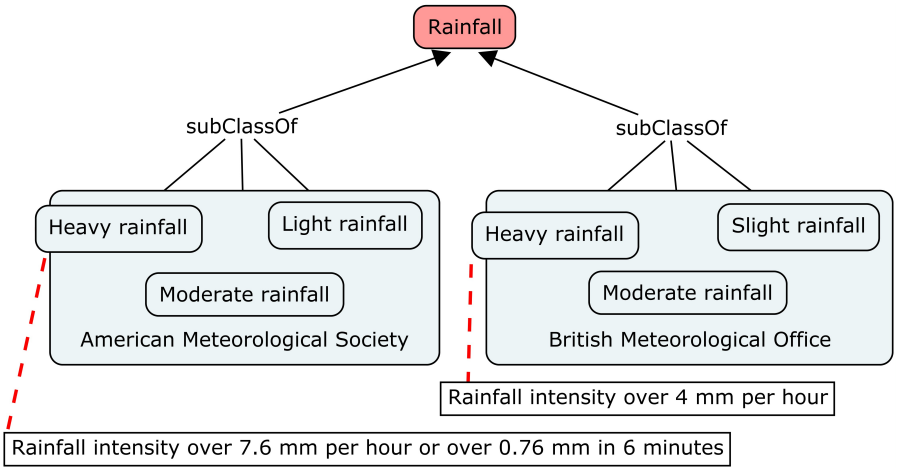

Figure 9: Example of semantic annotation of two patterns for heavy rainfall events.

event patterns (squared boxes) and event types are indicated with dashed lines.

Defining the conditions for the classification of environmental events depends on various factors, such as the location of the region of interest or the availability of historical records of events. Two information communities may use the same event type to refer to different event patterns, as happened with MODERATE and HEAVY rainfall in the introductory example of section 1.1. Two communities may also use the same event pattern to describe different event types. The presented method allows for the two possibilities, but it focuses on the case of different event patterns annotated with the same event type, which seems to be more common in the environmental monitoring domain. The key lies on decoupling event types and event patterns in the formalisation of knowledge.

Event patterns shall not be included in ontologies. This fosters the reusability of ontologies across information communities when they use similar event classifications. For instance, back to the introductory example of section 1.1, if the British Meteorological Office creates an ontology for modelling rainfall classifications and it includes event types related to event patterns, the ontology only can be reused in the context of the British Meteorogical Office. On the contrary, by avoiding the inclusion of event patterns, the same ontology can be reused in the American Meteorological Society context with few modifications. Although this example is a simplification of reality, it illustrates the main idea of ontology reuse among information communities.

I organise application- and domain-specific knowledge in different levels as suggested by Klien and Probst [81], see figure 10. At the bottom, application ontologies model event-related knowledge specific to information communities. In the middle layer, domain ontologies represent a shared conceptualisation within a domain. On top,

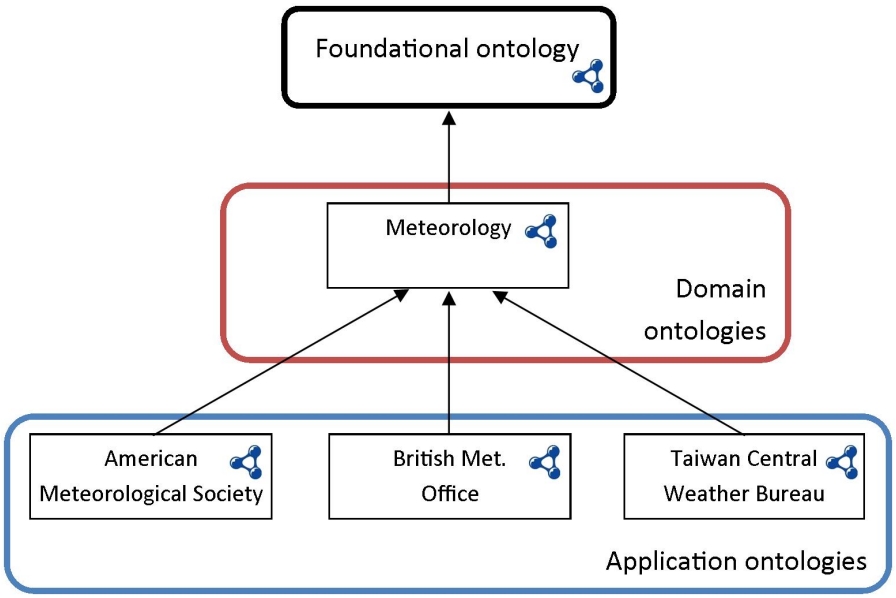

Figure 10: Example of the layered ontology structure.

the foundational ontology defines concepts common to all domains, like event. Event types that are specific to an information community shall be represented in an application ontology, e.g. EXTREMELY TORRENTIAL RAINFALL event in the Taiwanese Central Weather Bureau. Event types that are common to a domain shall be represented in a domain ontology, e.g. RAINFALL event in the meteorological domain. All concepts in application ontologies inherit from concepts defined in domain ontologies because application ontologies specialise domain knowledge. All concepts in domain ontologies inherit from the top-level concepts defined in the foundational ontology. The alignment to a foundational ontology provides a common set of top-level concepts as a reference across domains and applications. The mappings between application ontologies or domain ontologies are also possible, but out of the scope of this thesis.

3.4 THE EVENT ABSTRACTION ONTOLOGY

Attempts at designing a foundational ontology for all types of events have failed [127]. The Event Abstraction ontology[1] does not aim at representing all types of events. It provides a set of concepts and relations to model events inferred from time series of sensor data. The Event Abstraction Layer uses these concepts and relations to represent the generated events.

The Event Abstraction ontology extends the SSN ontology (introduced in section 2.3.4). Both ontologies are aligned to the DUL foundational ontology to restrict the interpretation of concepts and relations.

1 http://wsmls.googlecode.com/svn/trunk/global/Event-abstraction/0.2/.

I reused the SSN ontology because it describes the domain of sensors and sensor networks as a result of a revising seventeen existing sensor-centric and observation-centric ontologies[2]. The SSN ontology is supported by the W3C community and was created with the aim of being reused and extended. Additionally, the ontology design pattern at the SSN ontology core is based on the SSO pattern (see section 2.3.4). The SSO pattern fits my purpose of modelling events inferred from observations, although the concept stimulus is not explicitly modelled. In the context of this research, the stimuli are changes in the environment reflected in time series of sensor data. These stimuli trigger the inference of an event, which is produced by a virtual sensor. In the SSO pattern, the event inference corresponds to a sensor producing a symbol for an observation.

The main concept of the Event Abstraction ontology is the *event abstraction*. To put it simply, an *event abstraction* is the representation of an event. An *event abstraction* represents the perception of certain conditions in time series of sensor data. These conditions are described by an *event abstraction rule* (or event pattern). *Event abstraction rules* are based on observed *properties*, like rainfall intensity. *Event abstractions* are inferred by *event processing agents*, which are virtual sensors. Etzion and Niblett [45] define an *event processing agent* as a software module that processes events. The *event detection procedure* describes the sensing method used by such agents to infer events. For instance, CEP is the procedure used in my implementation, as explained in next chapter. An *event abstraction* is attached to an *event type*. The semantic annotation of event patterns (see previous section) maps *event abstraction rules* to *event types*. This mapping is a formalisation of domain knowledge and allows inferring events automatically. *Event abstractions* are related to a *spatio-temporal region*. The spatial location of the *event abstraction* is extracted from the sensors providing the observations. The temporal location of the *event abstraction* is inferred from the observation timestamps. Moreover, the *event abstraction* is related to the observation *collection* that was used for the event inference.

The four core concepts of the Event Abstraction ontology aligned to SSN and DUL are depicted in figure 11. In the rest of the thesis, I use the abbreviation EABS as namespace to refer to the concepts and relations included in the Event Abstraction ontology. Concepts of the Event Abstraction ontology are depicted in yellow boxes. The concepts and relations of the SSN and DUL ontologies are represented in orange and blue boxes, respectively. The RDFS relations are reused from the RDFS vocabulary.[3] The purpose of figure 11 is illustrating

2 Reviews available at http://www.w3.org/2005/Incubator/ssn/XGR-ssn-20110628/#Review_of_Sensor_and_Observation_ontologies.

3 The RDF Schema Description Language 1.0 is available at http://www.w3.org/TR/2004/REC-rdf-schema-20040210/.

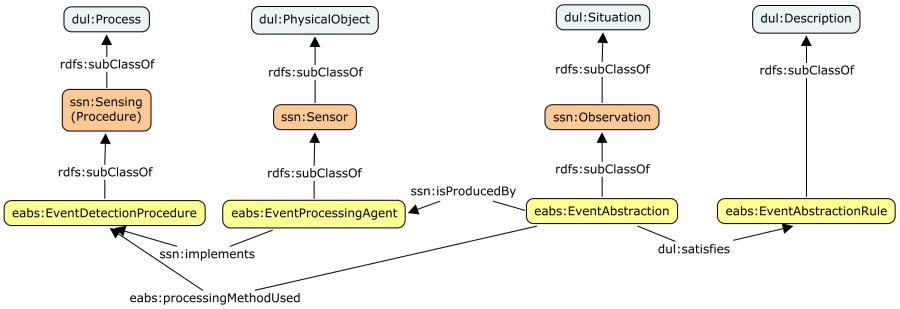

Figure 11: Main concepts of the Event Abstraction ontology aligned to the SSN and DUL ontologies.

from which SSN[4] and DUL[5] concepts are the Event Abstraction ontology concepts derived:

- EABS:EVENTABSTRACTION extends SSN:OBSERVATION[6], which inherits from DUL:SITUATION[7]. The definitions of both superclasses fit the EABS:EVENTABSTRACTION definition because it represents a view (or perspective) of an information community on a data set where a method has been used to infer the properties of an event.

- EABS:EVENTABSTRACTIONRULE inherits from DUL:DESCRIPTION[8] because it defines conditions on observed properties to describe an event.

- An EABS:EVENTPROCESSINGAGENT is subclass of SSN:SENSOR. The SSN ontology defines a SSN:SENSOR as any entity that can follow a sensing method to observe a property of another entity. This definition allows for physical devices and computational methods, among others. Therefore, virtual sensors suit the definition of SSN:SENSOR well.

- EABS:EVENTDETECTIONPROCEDURE extends SSN:SENSING, which is a process (or method) for the estimation of the value of a phenomenon. The EABS:EVENTDETECTIONPROCEDURE is the sensing method used to infer events. The inference procedure can be considered an estimation of the event properties. The event is regarded as a phenomenon that represents a change in the environment.

4 Definitions for SSN concepts can be found at http://www.w3.org/2005/Incubator/ssn/ssnx/ssn.

5 Definitions for DUL concepts are extracted from http://www.w3.org/2005/Incubator/ssn/wiki/DUL_ssn and http://www.loa.istc.cnr.it/ontologies/DUL.owl.

6 An SSN:OBSERVATION is a situation in which a sensing method has been used to estimate or calculate a value of a property of a feature of interest.

7 A DUL:SITUATION is a view on a set of entities which is consistent with a description.

8 A DUL:DESCRIPTION is a conceptualisation that defines a view from a set of observations.

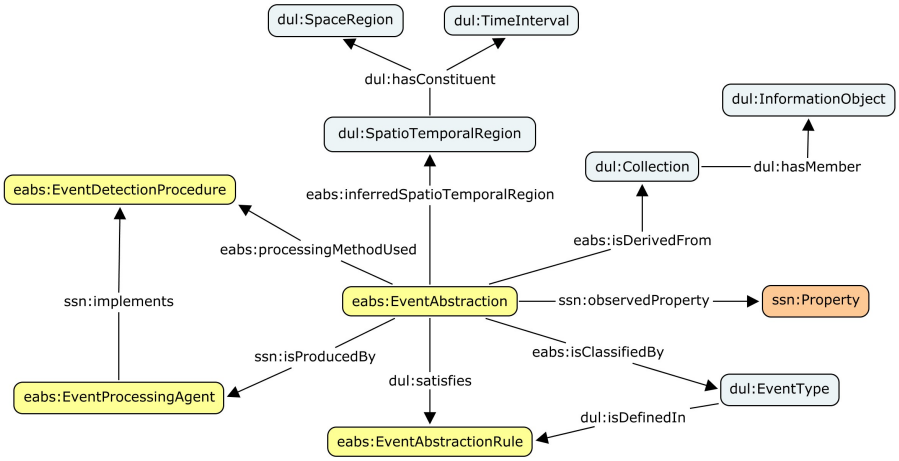

Figure 12: Extended concept map of the Event Abstraction ontology.

A complete concept map of the Event Abstraction ontology is depicted in figure 12. In this figure, I removed the subclass relationships for the sake of clarity. An EABS:EVENTABSTRACTION is an observed situation of change in one or various properties of a geographical entity that is relevant for the application purpose. Here, observing a situation of change means that an event is inferred from sensor data, which is assumed that reflects a situation of change in the environment being observed.

The spatio-temporal attributes of an EABS:EVENTABSTRACTION are specified by the concept DUL:SPATIOTEMPORALREGION. The property EABS:INFERREDSPATIOTEMPORALREGION points to the region derived from the spatial location of the in situ sensor providing the observation and the observation timestamps[9].

EABS:EVENTPROCESSINGAGENTS implement a method to infer events represented by the EABS:EVENTDETECTIONPROCEDURE. The product of the inference procedure is an EABS:EVENTABSTRACTION instance. Such instance satisfies the description of an EABS:EVENTABSTRACTIONRULE that defines an DUL:EVENTTYPE. The EABS:EVENTABSTRACTION represents changes in the observed SSN:PROPERTIES of the sensor data. Depending on the variety of data sources, the EABS:EVENTABSTRACTION can be related to more than one SSN:PROPERTY. A DUL:COLLECTION represents the set of sensor observations that were used to infer an event. This data set is formalised as DUL:INFORMATIONOBJECTS, i.e. pieces of information.

9 In a different application scenario, e.g. remote sensing, it would be more reasonable to use the spatial location of the observed entity instead of the location of the sensor.

The following tables include the concepts and properties used in the Event Abstraction ontology. The definitions of concepts and relations from SSN and DUL ontologies are adapted from their original sources (previously provided in this section). Table 1 defines the properties of the Event Abstraction ontology, and table 2 defines the concepts.

Property	Definition	Domain	Range
EABS:ISCLASSIFIEDBY	Relation between an event abstraction and its event type. This relation has been created to avoid using DUL:ISCLASSIFIEDBY, which has range DUL:EVENT.	EABS:EVENTABSTRACTION	DUL:EVENTTYPE
EABS:PROCESSINGMETHODUSED	Relation between an event abstraction and the method used to infer it.	EABS:EVENTABSTRACTION	EABS:EVENTDETECTIONPROCEDURE
EABS:INFERREDSPATIO-TEMPORALREGION	Relation between an event abstraction and the spatio-temporal region at which it was observed.	EABS:EVENTABSTRACTION	DUL:SPATIOTEMPORALREGION
EABS:ISDERIVEDFROM	Relation between an event abstraction and the observation collection that was used in the inference procedure.	EABS:EVENTABSTRACTION	DUL:COLLECTION
SSN:ISPRODUCEDBY	Relation between a producer and a produced entity.	EABS:EVENTABSTRACTION	EABS:EVENTPROCESSINGAGENT
SSN:OBSERVEDPROPERTY	Relation between an observation and the property that was observed.	EABS:EVENTABSTRACTION	SSN:PROPERTY
SSN:IMPLEMENTS	Relation between an entity that implements a procedure and the procedure.	EABS:EVENTPROCESSINGAGENT	EABS:EVENTDETECTIONPROCEDURE
DUL:SATISFIES	Relation between a situation and its description.	EABS:EVENTABSTRACTION	EABS:EVENTABSTRACTIONRULE
DUL:ISDEFINEDIN	Relation between a concept and its description.	DUL:EVENTTYPE	EABS:EVENTABSTRACTIONRULE
DUL:HASMEMBER	Relation between a collection and the entities of the collection.	DUL:COLLECTION	DUL:INFORMATIONOBJECT
DUL:HASCONSTITUENT	Relation of belonging between two world layerings, e.g. social system has constituent person. The constituent is regarded as a part that belongs to the lower layer.	DUL:SPATIOTEMPORALREGION	DUL:SPACEREGION, DUL:TIMEINTERVAL

Table 1: Properties of the Event Abstraction ontology.

Concept	Definition	Superclass
EABS:EVENTABSTRACTION	Observed situation of change in one or various properties of a geographical entity that is relevant for the application purpose.	SSN:OBSERVATION
EABS:EVENTABSTRACTIONRULE	Rule that describes an event based on observed properties.	DUL:DESCRIPTION
EABS:EVENTPROCESSINGAGENT	Virtual sensor that infer events.	SSN:SENSOR
EABS:EVENTDETECTIONPROCEDURE	Sensing method used to infer events.	SSN:SENSING
SSN:PROPERTY	Observable quality of an event or object.	DUL:QUALITY
DUL:EVENTTYPE	Concept that classifies an event. An event type defines how an event should be interpreted according to a description.	DUL:CONCEPT
DUL:COLLECTION	Container for entities that have one or more properties in common.	DUL:SOCIALOBJECT
DUL:INFORMATIONOBJECT	Piece of information.	DUL:SOCIALOBJECT
DUL:SPATIOTEMPORALREGION	Region formed by a time interval and a spatial region.	DUL:REGION
DUL:SPACEREGION	Region in a dimensional space that is used to localise an entity.	DUL:REGION
DUL:TIMEINTERVAL	Region in a dimensional space that represents time.	DUL:REGION

Table 2: Concepts of the Event Abstraction ontology.

This chapter described the proposed method to infer events from time series of in situ sensor observations. I presented my conception of events, the holistic design of the Event Abstraction Layer, guidelines on how to annotate event patterns, and an ontology to model events inferred from observations. Next chapter explains the implementation of the Event Abstraction Layer.

4

IMPLEMENTATION OF THE EVENT ABSTRACTION LAYER

This section presents an overview of the Event Abstraction Layer architecture and a description of the capabilities provided by the prototype implemented for this research.

4.1 SEMANTIC EVENT PROCESSING ARCHITECTURE

Event processing architectures are divided in three layers [45]: event producers, intermediary processing, and event consumers. Figure 13 depicts the interaction between these layers. Event producers generate events, which are sent to the components in the processing layer. Event consumers are software components that receive events, evaluate them and optionally take some action. Producers and consumers are decoupled through the intermediary processing layer. The arrow that leaves the processing layer and points again to the same layer indicates the possibility of creating new events that can be processed again before being consumed. This division enables loosely coupled event processing systems.

In the proposed architecture, the role of event producer is played by a component of the Event Abstraction Layer. The event producer is a parser that converts O&M observations into CEP objects that the CEP engine can process. These objects have the following properties: sampling time, sensor identifier, spatial location (of the sensor), observed property, observation value, and unit of measurement. In figure 14, the O&M parser is included in the event producers' layer.

The intermediary layer of figure figure 14 contains a CEP engine that allows registering semantically annotated event patterns. That way, each pattern points to an ontology concept that represents an event type. Event patterns define conditions based on properties of CEP objects. Additionally, listeners are attached to event patterns. Listeners can be programmed, using the CEP engine API, to perform actions if an event pattern is matched. Figure 14 shows two examples of listeners. Listener A reacts on objects of type 'circle' and creates event

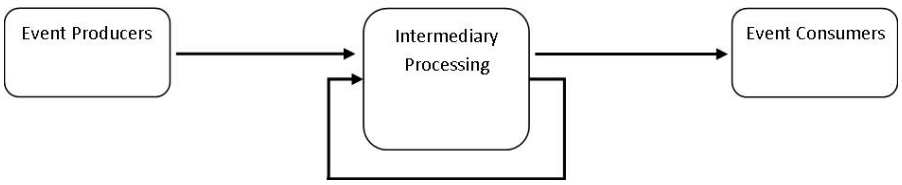

Figure 13: Three basic layers of event processing architectures.

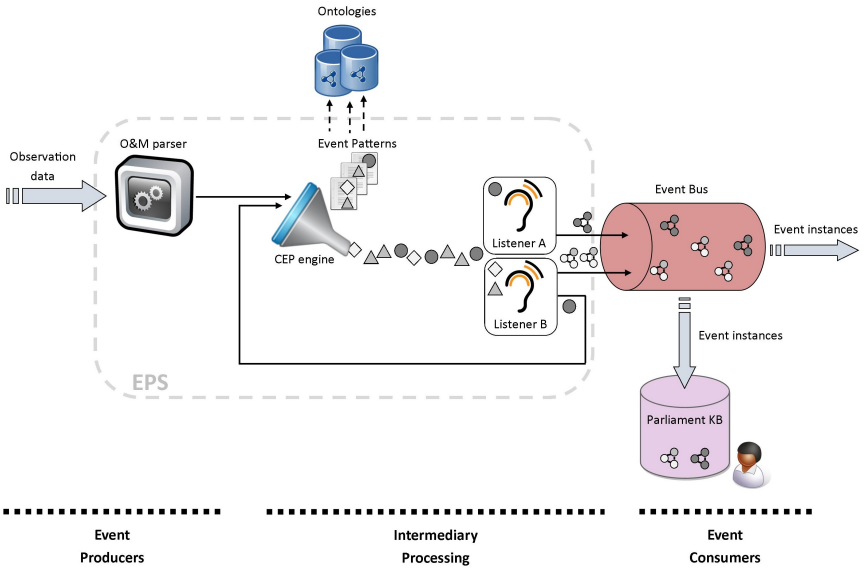

Figure 14: Main components and information flow of the semantic event processing architecture.

instances encoded as RDF triples that are published to the Event Bus. Listener B reacts on situations where diamond and triangle objects are consecutive, and publish RDF event instances to the Event Bus. Moreover, Listener B constructs a 'circle' object that sends back to the CEP engine. This feedback to the CEP engine allows building complex patterns, e.g. diamond-triangle, upon simple event patterns.

External applications, such as a map to display events, can consume the events from the Event Bus. The bus is not a consumer per se, but the channel on which the event instances are published. A copy of every event instance is stored into a knowledge base. In this implementation, I use the Parliament[1] triple store because it offers a SPARQL endpoint compatible with GeoSPARQL, a geographic query language for RDF data [110]. GeoSPARQL was developed to unify the access to geospatial data in the Semantic Web [9].

The Event Abstraction Layer overlaps the event producers and the intermediary processing layers of figure 13. The components included in the grey dashed box of figure 14 implement the functionalities of the Event Abstraction Layer. The Event Processing Service (EPS) is the realisation of this layer and is described in next section.

1 More information at http://parliament.semwebcentral.org/.

4.2 EVENT PROCESSING SERVICE PROTOTYPE

I developed a Web service prototype as a proof of concept for the method presented in the previous chapter: the EPS.[2] The purpose of the EPS is inferring events from time series of sensor data. The intended users of the prototype are domain experts or people with access to domain knowledge.

At the heart of the EPS there is a CEP engine. I used the Esper engine because it provides an open source and well documented API (this implementation is based on the version 4.4.0).[3] The EPS is able to process time series of observations encoded in O&M 1.0. Users register event patterns that are checked against the continuous flow of data in near real-time. Every time a pattern is matched, a timestamped and geolocated representation of the event in RDF is published to the Event Bus. This section presents the implementation of the main functionalities of the EPS: scheduling of sensor data requests, registration of event patterns, and publication of event instances.

4.2.1 *Scheduling sensor data requests*

The EPS offers a method called `registerService` to allow users scheduling requests of time series of sensor observations. Listing 1 shows the signature of the method. The prototype supports instances of the SOS, version 1.0. When an SOS is registered, a new entry is created in the service registry. Then, a `getObservation` request is scheduled to be executed recurrently. The interval between data requests is specified in the parameters. The list below listing 1 describes the `registerService` parameters.

Listing 1: registerService method to schedule observation requests with the EPS.

```
registerService(String serviceURL, String offering, String
    observedProperty, String timeUnit, String numberOfTimeUnits)
```

- The `serviceURL` is the URL of the service without any additional parameters, e.g. "http://uni-muenster.de/WeatherSOS".
- An observation `offering` is a thematic grouping of observations offered by a service [106]. It is similar to what we understand as a map layer in terms of sensor observations, e.g. "urn:ifgi:uni-muenster:weatherSensor:2".
- An `observedProperty` is the property of a phenomenon being observed by a sensor, e.g "urn:ogc:phenomenon:OGC:waterflow".

2 The source code is availabe at https://github.com/allaves/EPS/tree/master/EventProcessingService.

3 Esper for Java available at http://esper.codehaus.org/about/esper/esper.html.

- The time interval to schedule the data requests is defined in `timeUnit`, e.g. "minutes", and `numberOfTimeUnits`, e.g. 7.

Given a registered service, each recurrent `getObservation` request is different. In the appendix, listing 5 contains the template of the `getObservation` request. The begin and end positions of the sampling time change in every new request. The time interval parameters and the time of the system running the service are used to calculate the sampling time. For instance, with a current system time on `April 4th 2008 10:00:00` and a time interval of seven minutes, the first data request would have `April 4th 2008 09:53:00` as `<beginPosition>` and `April 4th 2008 10:00:00` as `<endPosition>`. The second data request, seven minutes later, would have `April 4th 2008 10:00:00` as `<beginPosition>` and `April 4th 2008 10:07:00` as `<endPosition>`.

The O&M parser converts each observation into a CEP object. The CEP engine processes the stream of objects produced by the consecutive requests. The service is easily adaptable to process sensor data streams in different encodings. This step would require the development of an additional parser for the target encoding.

4.2.2 *Registering event patterns*

The EPS allows registering event patterns encoded in Esper's EPL.[4] In Esper's EPL, the `SELECT` clause specifies a view on the CEP object properties. Operators as `sum`, `avg`, `count`, `max`, `min`, among others, are supported. A wildcard (∗) in the `SELECT` clause includes all the object properties.

The `FROM` clause defines one or more streams of objects. In the example of listing 2, `ObservationEvent` is the name of the CEP object class that the O&M parser creates. Conditions can be set on the properties of CEP objects in the stream. The condition on `observer.id` filters the `ObservationEvents` with the property `observer.id` equals to `sensor-0002`. The condition on `observedProperty` filters objects with `observedProperty` equals to `waterlevel`. A time window of one hour is applied to the stream of objects. As a consequence, the average operator (`avg`) defined in the `SELECT` clause is applied to the values of water level observations produced by the sensor `sensor-0002` within one hour. It is out of the scope of this section covering all the Esper's EPL aspects, thus more information about the syntax can be found in the official documentation.[5]

4 EPL is a SQL-like language. In the Esper documentation, event patterns are called EPL statements, see http://esper.codehaus.org/esper-4.4.0/doc/reference/en/html_single/index.html.

5 EPL reference for Esper 4.4.0 available at http://esper.codehaus.org/esper-4.4.0/doc/reference/en/html_single/#epl_clauses.

Listing 2: EPL statement example to calculate the hourly average value of water level for a specific sensor.

```
SELECT avg(value)
FROM ObservationEvent(observer.id = 'sensor-0002',
    observedProperty = 'waterlevel').win:time(1 hour)
```

Event patterns are semantically annotated by users. The EPS method to register annotated event patterns is called `registerStatement`. The method has two inputs: `stm` and `eventType`. The signature is shown in listing 3.

Listing 3: registerStatement call to register event patterns to the EPS.

```
registerStatement(String stm, String eventType)
```

The first parameter is the event pattern encoded in Esper's EPL. The second parameter is a Uniform Resource Locator (URL) representing the event type. `eventType` points to an ontology. In that ontology, the event type is defined as a subclass of DUL:EventType. The mapping between event pattern and event type is stored in the EPS. If the streams of data match an event pattern, an event instance is inferred and published on the Event Bus. The mapping is used to assign a type to that event instance.

4.2.3 *Publication of event instances*

The Event Bus implements the publish-subscribe interaction paradigm [46] (described in section 2.4.2). It uses a RabbitMQ[6] server based on the Advanced Message Queuing Protocol (AMQP) [134]. The EPS creates EABS:EventAbstraction instances and publishes them on the bus. External applications can subscribe to events based on their type, e.g. a map application subscribing to heavy rainfall events.

The properties of an EABS:EventAbstraction instance are inferred from properties of CEP objects. Each event instance is encoded in Notation 3 (N3)[7] and sent to the Event Bus as a text message. I use this encoding because it is easier to read than RDF/XML. Figure 15 depicts the structure of an EABS:EventAbstraction instance. This concept map is an extension of the concept map in figure 12. Question marks represent literals in the event instance. A table including the properties between concepts and literals can be found in the Appendix, section 8.4.[8]

6 http://www.rabbitmq.com/

7 Documentation for N3 is available at http://www.w3.org/TeamSubmission/n3/.

8 An example of an EABS:EventAbstraction instance for a high water level event is available at https://www.dropbox.com/s/pfmxqas2ktibtcy/eventInstanceExample.n3.

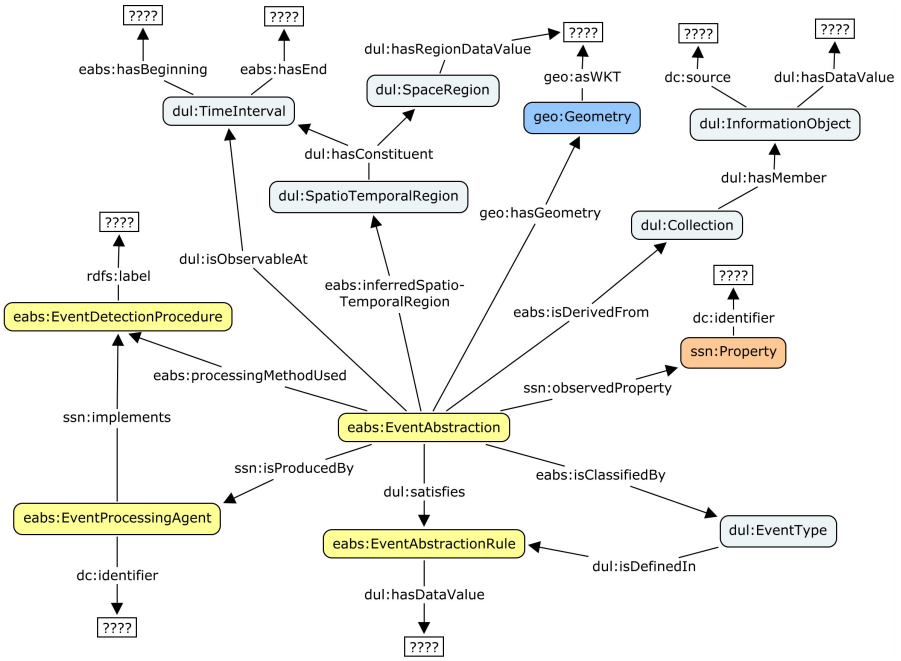

Figure 15: Template for Event Abstraction instances.

Event instances are stored in a Parliament knowledge base.[9] Parliament provides a SPARQL endpoint to access the stored data. In order to enable GeoSPARQL querying of event instances [9], I added to the model a link from EABS:EVENTABSTRACTION to a new concept GEO:GEOMETRY.[10] A GEO:GEOMETRY is a representation of a spatial location [83]. This new link implies that EABS:EVENTABSTRACTION has to be subclass of GEO:FEATURE. A GEO:FEATURE is an entity that can have a spatial location [83]. Subclasses of GEO:FEATURE can *have* a GEO:GEOMETRY. The GEO:ASWKT property allows expressing geometries as RDF literals encoded in Well-Known Text (WKT).

To facilitate the temporal querying of the knowledge base, I also added a direct relation from EABS:EVENTABSTRACTION to its time interval: DUL:ISOBSERVABLEAT. I define the DUL:TIMEINTERVAL of an event using two time instants. From the set of CEP objects that match a pattern, the timestamp of the first is the initial instant. The property for the initial instant is EABS:HASBEGINNING. The timestamp of the last object corresponds to the final instant: EABS:HASEND. For punctual events, the initial and final instants of the DUL:TIMEINTERVAL have the same value. If the event pattern window has size for only one observation, a punctual event is inferred. Punctual events are also inferred when the application purpose is capturing the instant at which certain conditions hold. The literal format of time instants follow the

9 There is a running instance of Parliament available for testing at `http://giv-llaves.uni-muenster.de:8081/parliament`.

10 GEO is the namespace of GeoSPARQL.

XML Schema datatype *dateTime*.[11] The property DUL:ISOBSERVABLEAT links an event abstraction to the interval defined by the observations it was inferred from.

The redundancy on the links between an EABS:EVENTABSTRACTION and its spatio-temporal properties allows users designing queries according to their purpose, either using DUL or GeoSPARQL concepts and relations. I keep the concepts for spatio-temporal region and space region, and the DUL:HASCONSTITUENT properties in the model because they are part of the proper conceptualisation (figure 12). The model extension described in this section only has the purpose of optimising the querying procedure after publication.

In this chapter, I described the implementation of the Event Abstraction Layer via the EPS prototype. This prototype infers events from time series of observations. The EPS realises the annotation of event patterns by using the Event Abstraction ontology and a CEP engine. In order to evaluate this tool, I test it with real data in next chapter.

11 The definition of *dateTime* is available at http://www.w3.org/TR/xmlschema-2/#dateTime.

5

REAL-TIME FLOOD MONITORING IN THE DANUBE RIVER

This chapter describes a flood monitoring scenario to apply the proposed method. First section includes an overview on the motivation. Section 5.2 introduces two organisations with problems to share event-related information in the context of flood monitoring. Section 5.3 explains how to infer events from observations in the described scenario. Final section summarises the conclusions of the chapter.

5.1 OVERVIEW

The countries located along the Danube River collect data to assess hydrological and meteorological conditions. Figure 16 shows the river basin, which covers nineteen countries. I selected two Romanian organisations for the data tests: Romanian Waters National Administration and Hidroelectrica Romania. Romanian Waters National Administration is a governmental body responsible for water management in Romania. Hidroelectrica Romania manages hydroelectric power plants located at the South-West of Romania. Two dams are used by these plants to produce energy and help to protect the villages located downstream the river from floods. Both organisations are interested in obtaining high level information related to floods in order to use it for decision making, but they have two problems: i) there is no real-time management of data collected at the dams, and ii) there is no common model for sharing geospatial information about flood-related events. The former delays responses when flooding situations are detected. The latter leads to interoperability problems when information about such situations is exchanged between the two organisations.

5.2 TWO VIEWS ON THE RIVER FLOODS: A GOVERNMENTAL BODY AND A HYDROELECTRIC POWER PLANT

The Romanian Waters National Administration is the organisation in charge of the water management in Romania.[1] Information about the state of Romanian rivers can be accessed via its online GIS,[2] see figure 17. In this portal, gauging stations measuring water level, discharge, 24-hours precipitation, and air temperature are represented by sym-

1 Official website available at http://www.rowater.ro/default.aspx.

2 Romanian Waters online GIS available at http://gis2.rowater.ro:8989/SituatieHidrologica.html.

Figure 16: Map of the Danube River basin (from Wikimedia Commons).

bols depending on the water level trend: a circle (stable), triangle pointing up (increasing), or a triangle pointing down (decreasing). Different colours classify the status of the river at a specific point based on water level thresholds:[3]

- Green - Normal situation.
- Yellow - Attention threshold exceeded: level at which the risk of flooding is possible after a relatively short time frame. It requires increased vigilance when carrying out activities exposed to flooding.
- Orange - Flooding threshold exceeded: level at which major floods occur. It can lead to flooding of households and socio-economic goods.
- Red - Danger threshold exceeded: level at which special measures are necessary for the evacuation of people and goods, the restriction on the use of bridges and roads, and the operation of hydraulic structures.

Three thresholds for *Attention*, *Flooding*, and *Danger* are defined for each gauging station based on past events and historical data. Figure

3 Descriptions in Romanian are available in the Romanian Waters' Emergency Management Regulations http://www.rowater.ro/daprut/Documente%20Repository/Regulament%20%20gestionare%20situatii%20de%20urgenta%20.pdf, CAPITOLUL II, Art. 11, section (2) B.

Figure 17: Snapshot of the Romanian Waters online GIS.

18 shows the details for the station located at Bazias,[4] including the thresholds of 600, 690, and 700 centimetres (cm), respectively.

Downstream of Bazias, Hidroelectrica Romania manages two hydroelectric power plants. The Iron Gates I and II, located between Romania and Serbia,[5] are the biggest dams in the Danube River. Each of them hosts two power plants, one at the Romanian side and another one at the Serbian side. Hidroelectrica Romania collects data and operates the discharge of the reservoirs in order to avoid upstream and downstream floodings [63]. Unfortunately, these tasks are usually carried out without the full support (in terms of input data) from local authorities, which are responsible for the management of emergency situations. Although sensor networks are in place, the systems collecting and analysing the observations are not fully interoperable and data is often exchanged by phone [63].

5.3 INFERRING EVENTS FROM FLOOD MONITORING OBSERVATIONS

To address the lack of i) real-time management of the data collected at the dams and ii) a common event model to share flood-related information, I describe in this section how to apply the method proposed in chapter 3 to the scenario presented above.

4 The station of Bazias is located at the county of Caras-Severin, with coordinates 44°48′31″N 21°23′25″E, star B in figure 17.

5 The Iron Gates I and II are located at 44°40′15″N 22°31′45″E and 44°18′16″N 22°33′54″E, respectively. In figure 17, Iron Gates I is represented with the star I and Iron Gates II, with the star II.

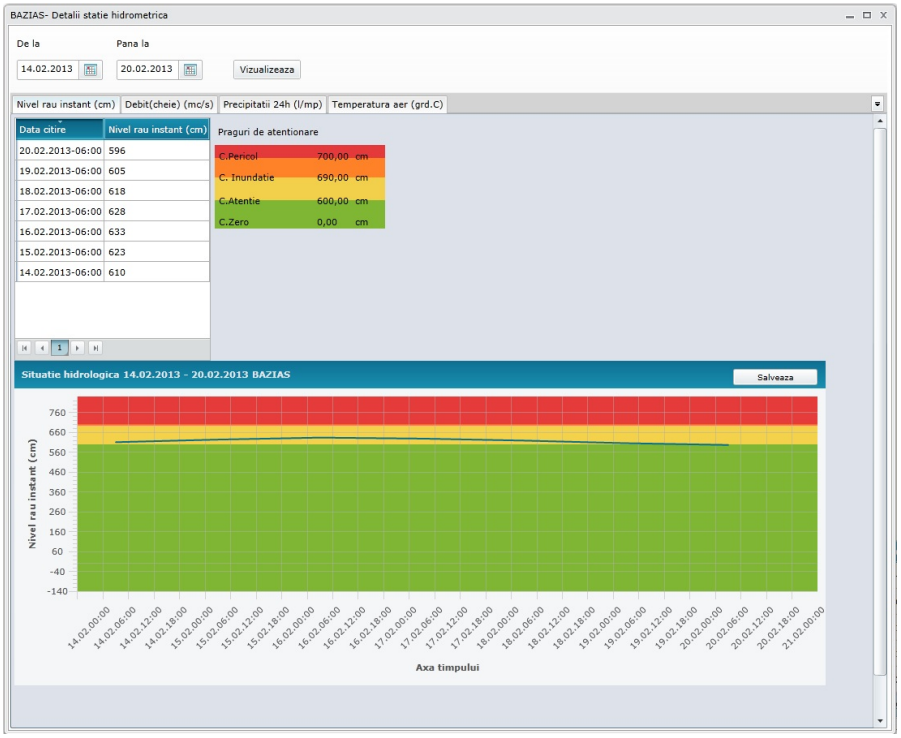

Figure 18: Gauging station interface for Bazias. Thresholds for flood stage categories are defined in the legend of the chart.

5.3.1 *A domain ontology for flood monitoring*

The flood monitoring ontology is the result of a collaborative work with some experts on the domain.[6] Part of the used knowledge is gathered on the technical document that describes the selected scenario [63]. The ontology is a simplification of a river model. The river has properties water level and water flow. Stream gauges deployed near the river measure these properties. Moreover, a river can have reservoirs that use dams to store the water. A dam may have a hydropower plant attached to it.

Figure 19 depicts the main concepts and relations of the flood monitoring ontology. Although the concept map is divided in two parts, all concepts are aligned either to the SSN or DUL ontologies in order to restrict the interpretations of the terms. A FLOODMONITORINGEVENT is anything that happens or is contemplated as happening at an instant - or over an interval - of time, and which is relevant for the flood monitoring application. Since the properties observed in this particular scenario are water level and water flow, the focus is on events that indicate changes on these properties, namely WATERLEVELCHANGES and WATERFLOWCHANGES. The flood monitoring ontology is intended to act as an agreement among the different communities in this sce-

6 The flood monitoring ontology, namespace FLOOD, is available at http://wsmls.googlecode.com/svn/trunk/local/water/0.6/.

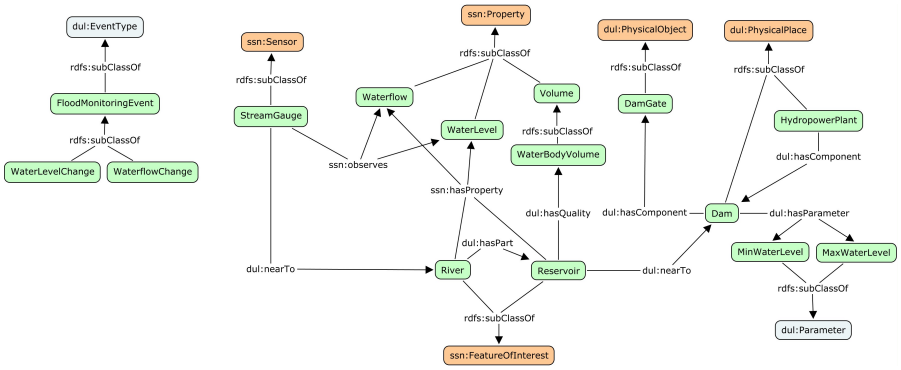

Figure 19: Concept map of the flood monitoring ontology.

nario, which share geospatial information related to the flood monitoring domain. Further specialisations of this ontology to represent application-specific conceptualisations can be added to the model as application ontologies.

5.3.2 *Application ontologies and event patterns*

I developed an application ontology with specific event types for each information community described in section 5.2. The purpose of separating application-specific knowledge is to keep the domain ontology as a reusable resource for other communities. Yet, these application ontologies are aligned to the flood monitoring ontology following the structure proposed in figure 10. Additionally, this section presents the event patterns that I defined for each event type.

Event types and patterns for Romanian Waters

From the Romanian Waters flood stage classification described in section 5.2, I defined nine event types. Figure 20 shows the application ontology with the event types for Romanian Waters.[7] Six of them correspond to crossings of thresholds defined for the *Attention*, *Flooding*, and *Danger* categories. The list below contains their six corresponding event patterns in natural language:

- *Attention threshold exceeded*: two consecutive observations (`obs1`, `obs2`) produced by the same sensor and ordered in time where the value of `obs1` is below the attention threshold and the value of `obs2` is above. The remaining event patterns in this section can be found in the Appendix, section 8.3.
- *Attention threshold deceeded*[8]: two consecutive observations (`obs1`, `obs2`) produced by the same sensor and ordered in time where

7 The application ontology for Romanian Waters, namespace RW, is available at `http://wsmls.googlecode.com/svn/trunk/application/EventType/RomanianWaters/`.

8 "Deceed" is a neologism that corresponds to the antonym of exceed.

the value of `obs1` is above the attention threshold and the value of `obs2` is below.

- *Flooding threshold exceeded*: two consecutive observations (`obs1`, `obs2`) produced by the same sensor and ordered in time where the value of `obs1` is below the flooding threshold and the value of `obs2` is above.
- *Flooding threshold deceeded*: two consecutive observations (`obs1`, `obs2`) produced by the same sensor and ordered in time where the value of `obs1` is above the flooding threshold and the value of `obs2` is below.
- *Danger threshold exceeded*: two consecutive observations (`obs1`, `obs2`) produced by the same sensor and ordered in time where the value of `obs1` is below the danger threshold and the value of `obs2` is above.
- *Danger threshold deceeded*: two consecutive observations (`obs1`, `obs2`) produced by the same sensor and ordered in time where the value of `obs1` is above the danger threshold and the value of `obs2` is below.

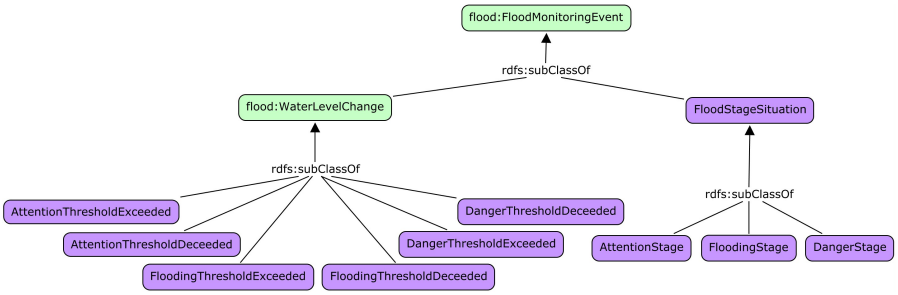

Figure 20: Application ontology containing the event types for the Romanian Waters scenario.

When the EPS detects any of these event patterns, it creates an EABS:EVENTABSTRACTION instance. The spatial location is derived from the location of the gauging station providing the observations. The temporal location is the time instant corresponding to `obs2`. I built the other three event patterns upon the threshold crossings listed above:

- *Attention stage*: two events (`e1`, `e2`) related to the same gauging station and ordered in time where `e1` is of type *attention threshold exceeded* and `e2` is of type *attention threshold deceeded*.
- *Flooding stage*: two events (`e1`, `e2`) related to the same gauging station and ordered in time where `e1` is of type *flooding threshold exceeded* and `e2` is of type *flooding threshold deceeded*.

- *Danger stage*: two events (`e1`, `e2`) related to the same gauging station and ordered in time where `e1` is of type *danger threshold exceeded* and `e2` is of type *danger threshold deceeded*.

When two events match any of these three patterns, the EPS creates an EABS:EVENTABSTRACTION instance with a spatial location derived from the locations of `e1-e2`. The time interval of the event instance is defined by the `e1-e2` time intervals. Regarding temporal relations between events, it is assumed under normal conditions (no error values, no sharp transitions between flood categories, and continuous data) that the relationship between two flood stage events (`A`, `B`), being `A` of higher risk than `B`, will always be:[9] `A` during `B`, `A` starts `B`, or `A` finishes `B`. The event patterns for Romanian Waters encoded in EPL are included in the Appendix, section 8.3.1.

Event types and patterns for Hidroelectrica Romania

Two types of events are relevant for Hidroelectrica Romania in the region of the Iron Gates: *low water level* and *high water level* events. The domain experts of Hidroelectrica Romania define thresholds for these events types. The conditions are based on observations taken at Iron Gates I and II: the water level at Iron Gates I must range between 63.00 mdMA[10] and 69.59 mdMA; and the water level at Iron Gates II must range between 39.40 mdMA and 41.00 mdMA. Figure 21 depicts a concept map with the corresponding application ontology,[11] which is aligned to the flood monitoring ontology. Section 8.3.2 in the Appendix includes the event patterns in EPL derived from experts' knowledge.

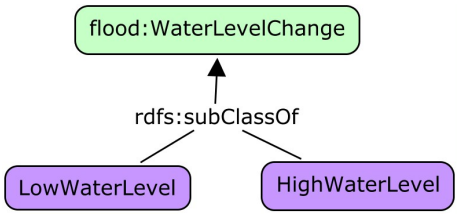

Figure 21: Application ontology containing the event types for the Hidroelectrica Romania scenario.

The data analysed by Hidroelectrica Romania includes also discharge (water flow) and water level observations at the mouth of the Nera River.[12] If the observations exceed the thresholds listed in table 3, the water level is considered too high and the dams are operated.

9 According to Allen's interval algebra [3]
10 Meters above the Adriatic Sea.
11 The application ontology for Hidroelectrica Romania, namespace HR, is available at http://wsmls.googlecode.com/svn/trunk/application/EventType/IronGates/.
12 Tributary that flows into the Danube upstream from the Iron Gates I.

Each row of table 3 describes different conditions for a *high water level* event at the mouth of the Nera River. Listing 4 contains an example of an event pattern from this table encoded in EPL. Approximately, the pattern describes a situation in which the water flow value is between 3000 and 3500 and the water level exceeds 69.81 in the following 24 hours. The `SELECT` clause defines two identifiers: `obs1` and `obs2`. In the `FROM` clause, I specify conditions on three properties of the observations (sensor identifier, observed property, and observation value). `obs1` is a water flow observation and `obs2` is a water level observation. The word `every` indicates that I am looking for every match of this pattern in the data, instead of only detecting the first one. The arrow `->` means that `obs2` follows `obs1` in the time of arrival at the CEP engine. It is assumed that observations are provided in near real-time and that there are no relevant delays. The `WHERE` clause filters only those pairs of observations for which `obs2` was measured within the 23 hours and 59 minutes after the measurement of `obs1`. I use this filter because water flow observations are taken every 24 hours, hence it makes no sense to accept pairs of observations for which the water flow observation, `obs1`, was taken days or even weeks before the water level observation, `obs2`. The remaining patterns from table 3 can be found in the Appendix, section 8.3.

Discharge (m^3/s)	**Water level (mdMA)**
Up to 3000	-
Between 3000 and 3500	69.81
Between 3500 and 4000	69.87
Between 4000 and 4500	69.92
Between 4500 and 5000	69.97
Between 5000 and 5500	70.02
Between 5500 and 6000	70.10
Between 6000 and 6500	70.17
Between 6500 and 7000	70.25
Between 7000 and 7500	70.31
Between 7500 and 8000	70.38
Between 8000 and 11500	70.40
Above 11500	-

Table 3: Discharge and water level thresholds for *high water level* events at the mouth of the Nera River.

Listing 4: Example of *high water level* event pattern for Nera mouth encoded in EPL

```
SELECT obs1, obs2
FROM pattern[every (obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 3000 and 3500) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 69.81)) ]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )
```

5.4 CONCLUSION

This chapter described a flood monitoring scenario to apply the method proposed in chapter 3. The challenges in the flood monitoring scenario are motivated by the necessity of sharing in near real-time geospatial information inferred from time series of observations. To solve the interoperability problems that arise when different information communities interact, I suggested structuring domain- and application-specific knowledge in ad-hoc ontologies. Moreover, I showed how to extract and model event types from text descriptions and tables. Finally, I described some examples of patterns for the target event types. The method is evaluated in the next chapter by using the developed ontologies, the event patterns, and the EPS prototype with real and simulated data.

6 EVALUATION

This chapter presents the evaluation of the methodology to infer events from time series of observations. Section 6.1 describes two data experiments with real and simulated data. Section 6.2 compares the method to other existing solutions introduced in chapter 2.

6.1 TESTING THE PROTOTYPE: EVALUATION EXPERIMENTS

This section describes two experiments designed to evaluate the method presented in chapter 3. The first experiment uses historical data from the Romanian floods in 2006. In the second experiment, I simulated flooding conditions in real-time for a specific region of the Danube River. To interact with the EPS remotely, I developed a Web service client.[1]

6.1.1 *Analysis of historical data: the Romanian floods in 2006*

The location of the experiment corresponds to the Danube River basin, along the border of Romania and Serbia. Different organisations have deployed gauging stations in this region. I selected the stations at Bazias and at the Iron Gates because event patterns and time series for 2006 are available (see chapter 5).

The data set consists of time series for 2006 of water level at the Iron Gates, and water level and discharge observations at Bazias. It is encoded in O&M 1.0 and can be accessed under permission for research purposes.[2]

Section 5.3.2 describes the event patterns used for the experiments. The formalisation of these patterns in EPL is available in the Appendix, sections 8.3.1 and 8.3.2. In total, there are nine patterns for Romanian Waters and fourteen for Hidroelectrica Romania. The client that I developed for this experiment registers all these event patterns properly annotated with their event types into an instance of the EPS.[3] Then, the client schedules two recurrent requests for time series of SOS data. The first request returns all the discharge (`WATER_FLOW`) observations collected during 2006. The second request returns water

1 The client source code is available for download at https://github.com/allaves/EPS/tree/master/EventProcessingServiceClient.

2 The data can be visualised at http://envision.c-s.ro/web/guest/publicdemo. For getting access to the raw data, contact CS-Romania at office@c-s.ro.

3 The source code is available at https://github.com/allaves/EPS/blob/master/EventProcessingServiceClient/test/de/ifgi/envision/eps/thesis/Danube/IGDEALThesisHistoricalDataTest.java.

level (GAUGE_HEIGHT) observations collected for 2006. The client code schedules the requests every three seconds, but requests are dynamic. The time window of the request advances one day every new execution. I added a new method registerServiceForHistoricalData to the EPS client to simulate the polling of data once a day because discharge observations are taken every twenty four hours.

There is a correlation between water level and water flow time series in Bazias and the Iron Gates I. Figure 22 depicts water level and water flow data for Bazias (in blue and red, respectively), and water level data for the Iron Gates I (in green; sensor identifier, *NivelAmontePF1*). The Iron Gates release high volumes of water when there is risk of flooding. Remarkable decreases and increases on the water level indicate the openings and closings of the dam gates. Water flow data for Iron Gates II are not available for this period. Figure 23 shows a chart with water level data for the Iron Gates II (sensor identifier, *NivelAmontePF2*).

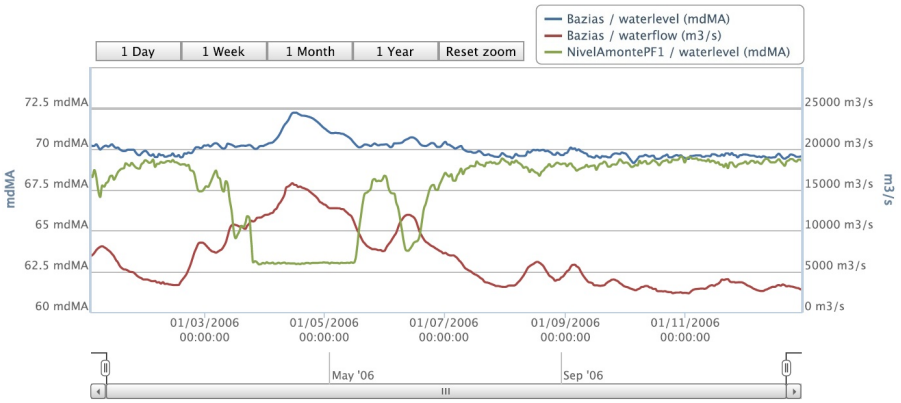

Figure 22: Water level observations for Bazias and NivelAmontePF1 and water flow observations for Bazias during 2006.

The experiment produces a set of one hundred sixty-eight events that are stored in the Parliament knowledge base.[4] Listing 11 includes the SPARQL query used to request the event instance URL, the event type, and the spatio-temporal location of all the inferred events.

Results for Hidroelectrica Romania

The analysis of results[5] for the events related to Bazias and Iron Gates shows that there are:

- Ten *high water level* events located at Bazias,

4 The event instances exported from Parliament are available at https://www.dropbox.com/s/vz2lu5y18liselu/floods2006Romania_results_parliamentExport.n3.

5 SPARQL queries for the analysis of results are available in the Appendix, section 8.5.

- eighteen *low water level* events located at Iron Gates I (from March 28th to May 8th),
- and one hundred *high water level* events located at Iron Gates II (none of them between March 12th and May 22nd).

Note that I defined *low* and *high water level* event patterns for Iron Gates I and II, but only *high water level* patterns for Bazias (see section 5.3.2). Those event patterns for Bazias are based on water flow observations ranging from 3000 to 11500 m^3/s (the latter threshold is called Q1 and indicates a high water flow rate). Therefore, during the two periods in which the discharge surpassed Q1 (March 27th to May 15th and June 10th to June 16th)[6] there are no *high water level* events inferred for Bazias.

A remarkable fact is that there are no *high water level* events for Iron Gates I and no *low water level* events for Iron Gates II during 2006. This indicates, on the one hand, that dam operators always opened the Iron Gates I before the dam reached its maximum capacity. On the other hand, the operators never left opened the Iron Gates II enough time to let the water level reach the minimum allowed. From the analysis of the flood-related events in 2006, I conclude that the opening of Iron Gates I and II in 2006 was coordinated to store or discharge water depending on flood risks, but without reaching in any case the high water level at Iron Gates I and the low water level at Iron Gates II.

Results for Romanian Waters

The analysis of results[7] for the events at Bazias based on the Romanian Waters classification shows that there are:

- Eleven *attention stage* events (the longest one from March 29th to May 18th),
- one *flooding stage* event (from April 9th to May 5th),
- and one *danger stage* event (from April 9th to May 2nd).

The twenty-seven remaining events correspond to exceeded and deceeded flood thresholds. The results validated the hypothesis of section 5.3.2: the temporal relationship between two flood stage events (`A`, `B`), being `A` of higher risk than `B`, will always be:[8] `A` during `B`, `A` starts `B`, or `A` finishes `B`. Additionally, the analysis of the flood stage events located at Bazias overlaps the period in which this area was affected

6 This information has been extracted from the visualisation interface available at http://envision.c-s.ro/web/guest/publicdemo.

7 SPARQL queries are available in the Appendix, section 8.5.

8 According to Allen's interval algebra [3]

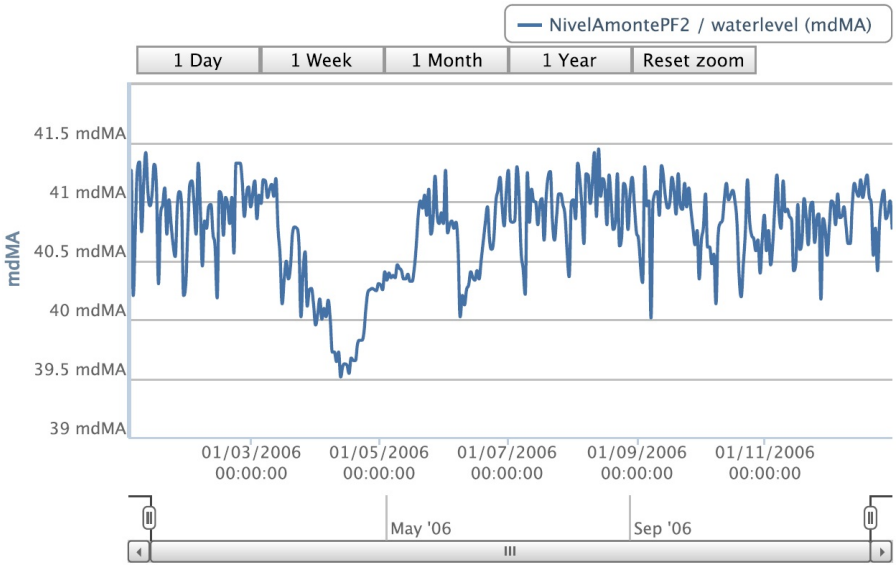

Figure 23: Water level observations for Iron Gates II during 2006.

by floodings. News on the media reported floods in the area for this period of time.[9] The International Commission for the Protection of the Danube River (ICPDR)[10] informed about the floods in this area in a report:[11]

"The Danube levels in Romania ***exceeded average monthly multi-annual values for April and May****, inducing massive flooding in the 12 counties along the Danube River. [...] The Danube levels greatly* ***exceeded the attention, inundation and danger values, resulting in severe flooding*** *of some localities, affecting farm animals, isolated buildings and large agricultural fields. [...] Aid was distributed in many of the* ***affected counties: Caras-Severin****,..."*

The International Federation of Red Cross and Red Crescent Societies (IFRC)[12] informed in various documents about the situation:[13]

*"****Over the past three weeks****, Romania has been hit by the heaviest seasonal flooding in 110 years. Three counties in south-western Romania -* ***Caras-Severin****, Mehedinti and Dolj - and three counties in the south-eastern part of the country - Tulcea, Constanta and Calarsi - have suffered the most."*

9 For news on CNN International, check http://edition.cnn.com/2006/WEATHER/04/15/danube.floods.reut/.
10 Official website of ICPDR at http://www.icpdr.org/.
11 A report titled "The Analysis of the Danube Floods 2006" (pp. 39-45) is available at http://www.icpdr.org/main/resources/analysis-danube-floods-2006.
12 More information about the IFRC can be found at http://www.ifrc.org/.
13 Extracted from the DREF Bulletin no. MDRRO001 (p. 2) from April 26th 2006, available at http://www.ifrc.org/docs/appeals/06/MDRRO00101.pdf.

The integration of different views about the same data helps to extract richer information. For instance, the analysis of all the inferred events shows that during the longest *attention stage* event in Bazias (from March 29th to May 18th), the EPS generated most of the *low water level* events located at Iron Gates I (from March 28th to May 8th). From this fact, I deduce that the dam operators opened the gates before reaching the *attention stage* in Bazias, maybe alerted by forecasts. Moreover, the operators maintained the water level at the minimum allowed at Iron Gates I. The oscillations above and below the minimum allowed level indicate that the gates were not closed until the situation improved at Bazias.

6.1.2 *Simulating floods in real-time*

The second experiment simulates a water level increase and decrease at a particular point of the Danube River. The water flow is only increased (and not decreased) to simulate what happens at the Iron Gates when one of the dam gates is opened.

I selected a gauging station located at Pancevo, Serbia, because I have access to insert observations remotely.[14] The station is located at the Danube River, upstream of Bazias and near Belgrade.

The simulation is split in two parts: i) water flow increase and ii) water level increase and decrease. The former consists in increasing the initial water flow value, set to 2900 m^3/s, by adding 40 to each new simulated observation every ten seconds during three minutes. The simulation of water level variations consists in increasing the initial water level value, set to 65.9 cm, at a rate of 0.5 cm every ten seconds during six minutes. Table 7, in the Appendix, contains the simulated data inserted for the various time steps (t_0, t_1,..., t_{35}).

In total, I created ten event patterns for the following event types: *low* and *high water level*, *attention threshold exceeded*, *attention threshold deceeded*, *flooding threshold exceeded*, *flooding threshold deceeded*, *attention stage*, and *flooding stage*. The event patterns and their corresponding event types are listed in natural language in the Appendix, section 8.3.3. These event patterns are similar to the event patterns defined for the historical data experiment, except that these ones are not extracted from domain knowledge. The formalisation of these event patterns can be found in the application that I developed for this experiment.[15] First, the application registers all event patterns with the mappings to event types. Then, the application schedules two recurrent SOS requests for water level and water flow, respectively. The method used is `registerService`, which returns the latest observa-

14 The gauging station of Pancevo is located at coordinates 44°47′53″N 20°38′13″E.

15 The code of the real-time experiment is available at https://github.com/allaves/EPS/blob/master/EventProcessingServiceClient/test/de/ifgi/envision/eps/thesis/Danube/IGDEALThesisRealTimeDataTest.java.

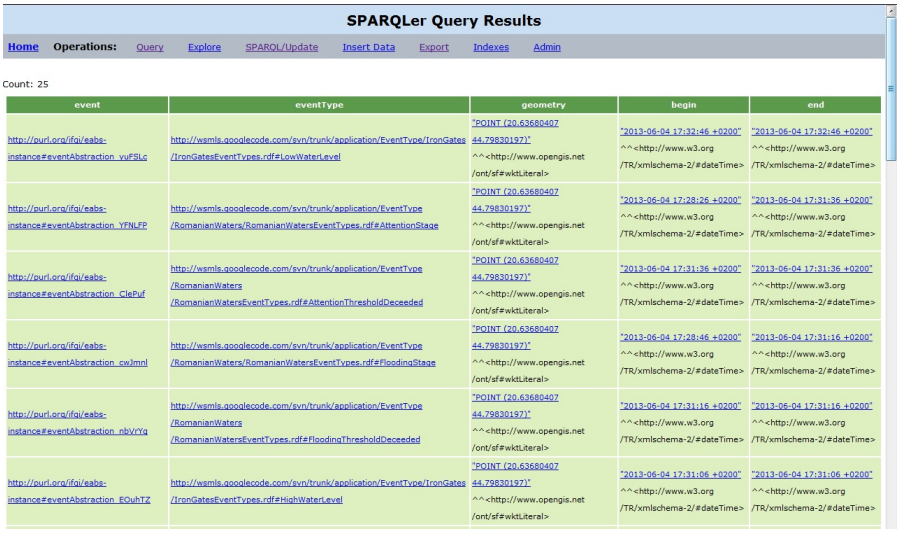

Figure 24: Snapshot of the Parliament KB interface showing the results of the simulation experiment.

tions for all available sensors every ten seconds. After six minutes, the service is stopped.

According to the event patterns registered for this experiment and the simulated data, the EPS should generate the following twenty-five event instances located at Pancevo:

- Eighteen *high water level* events created consecutively from t_8 to t_{25}, both included.
- One *attention stage* event from t_9 to t_{28}.
- One *flooding stage* event from t_{11} to t_{26}.
- One *low water level* event at t_{35}.

To get a list of all the inferred event instances, I queried the knowledge base via the SPARQL endpoint. The SPARQL query of listing 11 requests all the event instances available in the knowledge base, including the instance URL, the event type, and the spatio-temporal location of the event. Figure 24 shows a snapshot of the Parliament interface after executing the query. The resulting event instances of this experiment are available online encoded as N3 triples.[16]

The execution of the experiment delivers to the bus the expected number and type of events as they are inferred. As explained in section 4.2.3, events are published to the Event Bus as text messages. The Event Bus API does not provide a way to register the timestamp of each message arrival at the queue. It is not my purpose to test the performance of the bus. However, I estimated the average delay of

16 The event instances exported from Parliament are available at `https://www.dropbox.com/s/gkes69fod08kx1s/realTimeExperiment_parliamentExport.n3`.

the message arrival. First, I gave values to every time step, (t_0, t_1,..., t_{35}), by adding ten seconds to the initial timestamp (the execution of the experiment application returns the value t_0 and schedules data requests every ten seconds). Additionally, the code adds a timestamp at the inference instant to the header of each message. I listed above what events should be inferred and when. Therefore, I measured the difference between the corresponding time step and the event that should be inferred. After several executions of the real-time experiment in the current server settings, the estimated average delay was always between three and four seconds.[17] The delay can vary depending on the features of the server but with low message rates, similar to the ones in the simulation experiment (and in the Danube's flood monitoring scenario), the delay should be in the order of few seconds.

6.1.3 *Conclusion*

These two experiments demonstrate that the developed prototype is able to infer events from historical and real-time sensor data. The experiments show that the EPS can infer punctual events (e.g. *high water level*). Moreover, the event pattern language used allows for building patterns for more complex events (e.g. *attention stage*) upon punctual event patterns. The EPS creates instances of punctual and durative events using the Event Abstraction model. Instances are published to the Event Bus in near real-time. Storing the event instances in a triple store allows executing SPARQL queries against the knowledge base. For instance, these queries can ask for events related to a specific location, or for events of a specific type. Geospatial information that was not available before can be queried now via a SPARQL endpoint. The benefit of integrating different views from information communities via a common event model is that users can formulate questions about changes involving different applications on the same domain. Yet, the proposed methodology does not impose domain- or application-specific ontologies. As concluded in the experiment with historical data, the inferred information helps to improve the users understanding about the overall observed environment.

6.2 COMPARING MY APPROACH TO EXISTING SOLUTIONS

This section defines a set of comparison criteria and compares the presented methodology to similar solutions. The problem that my method addresses is about inferring event-related information from observations. For the comparison, I selected the solutions from section 2.4.4 that offer a methodology for inferring events from time series of observations. I also included the most relevant methods of se-

17 The EPS is running on Ubuntu 12.04.2 LTS, with an Intel Xeon CPU E5530 @ 2,40 GHz processor and a cache size of 8 KB. The RAM memory is 1 GB.

mantic event processing for general purpose from 2.4.3. The focus of the thesis is on managing interoperability problems among information communities exchanging geospatial information, thus the evaluation of performance measures, such as throughput, is out of the scope of this section.

The criteria for comparison are a mixture of requirements extracted from section 1.2 and related work. Results are included in table 4. The content of each criterion/method cell is the answer to the criterion's question applied to the method. My solution is located at the bottom of the table.

- **Near real-time:** Can this method be scheduled in order to analyse streaming data in near real-time? The compared solutions are conceived to analyse vast amounts of data. In the context of the Digital Earth [59], these solutions would be implemented as part of the *nervous system* [33], see section 1.2.1. Reacting to the information extracted from the data in a timely manner requires real-time processing capabilities [114].

- **Multi-perspective:** Is it possible to define various patterns for the same event type using different property conditions? Previous work in the area of spatial cognition and sensor data analysis call for models and methods that support multiple definitions of domain-specific concepts [14, 38]. This request applied to event inference is translated to suporting multiple definitions of the same event type.

- **Domain-independent:** Is this method applicable to other domains? Separating domain knowledge from application-specific and foundational knowledge, see section 3.3, is essential to build consistent models independently of their implementation [81].

- **Query-ability:** Is it possible to query the generated event-related information? The capability of processing queries is considered fundamental in data integration solutions [89]. As described in section 1.2.1, the Digital Earth also accounts for platforms that are able to answer domain-specific questions by multiple information communities [27].

- **Decoupling:** Is the system loosely coupled in terms of event producers and event consumers? In the pub/sub paradigm, see sections 2.4.2 and 4.1, publishers (event producers) only deal with the production of the event and do not care about receivers (event consumers). Similarly, receivers focus on listening to events, but not on who is publishing them. This decoupling is a desirable property in architectures because it enables scalability at the abstraction level, and allows producers and consumers operating independently [46].

Methods / Criteria	Near real-time	Multi-perspective	Domain-independent	Query-ability	Decoupling
Teymourian's thesis [129]	YES	N/A	YES	YES (SPARQL)	YES
ETALIS [7]	YES	N/A	YES	YES (Event Processing SPARQL)	YES
SCEPter [139]	YES	NO	YES	YES (SCEP query model)	YES
SemSOS [71]	NO	NO	YES	YES (SOS)	YES
RTFS [108]	YES	NO	YES	YES (SPARQL)	YES
SEGO [37]	NO	NO	YES	YES (SQWRL + SPARQL)	NO
Croitoru [28]	NO	YES	YES	YES (Query pattern tree)	N/A
Rude & Beard [115]	NO	YES	YES	NO	N/A
Llaves' thesis	YES	YES	YES	YES (SPARQL)	YES

Table 4: Comparison of the Event Abstraction Layer to similar solutions.

This comparison does not intend to rank solutions for event inference from sensor data. In some cases, the compared solutions do not aim at the same research goal than the Event Abstraction Layer does. Nevertheless, the comparison shows that, for the selected criteria, the methodology developed for this thesis offers similar or better capabilities than the rest described above.

7 CONCLUSIONS AND FUTURE WORK

This chapter answers the research questions proposed in section 1.3. It also includes the contribution of this thesis to the field of semantic event processing and a discussion on future work.

7.1 ANSWER TO RESEARCH QUESTIONS

This thesis investigates how to infer and represent events from time series of in situ sensor observations. The focus is on solving semantic interoperability problems among information communities exchanging event-related information. In this context, I formulated two research questions. The first one deals with the procedure of event inference:

RQ1. How to infer events on the fly from time series of observations provided by in situ sensors?

I propose to use event processing to infer events from time series of observations. First, I define an event as anything that happens or is observed as happening at an instant - or over an interval - of time, and which is relevant for the observer. Information communities use descriptions of events to define the observable conditions under which events happen. Event processing allows using descriptions of events, formalised as event patterns, to extract fragments of sensor data that fit the event pattern. I consider this extraction of data the event inference. As a result of the inference, an event instance is created. Spatio-temporal properties of the event are implicit in the extracted observations. I derive the spatial location of the event from the location of the in situ sensors providing the data. For the temporal location of the event, I use the observation timestamps. The type of the event is provided by the domain expert together with the event description. Other properties of the event, such as the data provenance, are useful to make the inference procedure more transparent.

I use Complex Event Processing (CEP) [98] for event inference because it provides real-time processing capabilities and it allows processing multiple unordered streams of events. To perform semantic event processing, see section 2.4.3, I suggest annotating event patterns with a reference to a domain or application ontology, where the event type is described. The annotation of event patterns enrich the inference procedure. Generated event instances point to the conceptualisation that defines the event type, putting them into context.

As a proof of concept, I developed a prototype called Event Processing Service (EPS). The EPS converts time series of observations into a stream of objects that its CEP engine can process. Additionally, it allows registering event patterns with their corresponding event types. To simulate a continuous flow of observations from the sensor data services to the EPS, it offers a mechanism to schedule recurrent data requests. When the observations reflect the conditions described in a registered event pattern, the EPS creates an RDF event instance with the properties described above. The EPS publish a copy of the event instance to an Event Bus in order to allow external applications subscribing to events. Another copy of the event instance is stored in a knowledge base, which supports SPARQL querying.

The second question addresses the exchange of event-related information among information communities. To avoid semantic interoperability problems in the information exchange, the thesis analyses the properties of existing event models:

RQ2. How to represent event-related information so that it can be shared among information communities?

I propose to model event-related information in a layered ontology structure, as suggested by Klien and Probst [81] (section 3.3). At the bottom, application ontologies define the relevant event types for a specific application. In this context, application ontologies are often derived from classifications provided by information communities, such as flood categories (see chapter 5). Application ontologies are aligned to domain ontologies, which represent a simplified conceptualisation of a specific domain, e.g. hydrology. I suggest modelling domain and application ontologies as microtheories [66] that allow for contradictory representations within a knowledge base, as long as each microtheory is internally coherent [40, 72]. The use of microtheories avoids imposing conceptualisations. On top, the DUL foundational ontology [54] is aimed to provide a common semantic framework, where other ontologies can be anchored. DUL also helps to restrict the meanings of the basic concepts and relations shared across domains and information communities.

To model events inferred from observations, I developed the Event Abstraction ontology (see section 3.4). This model is an extension of the SSN ontology [24] and represents the main research output of the thesis. The EPS creates EventAbstraction instances that are classified with a specific event type. The proposed approach allows for representing different types of events derived from the same data. Additionally, it also supports the formalisation of multiple event descriptions annotated with the same event type, which is a common problem when information communities share event-related information. The Event Abstraction Layer enables sensor data integration by rep-

resenting, under the same event model and in one knowledge base, event-related information extracted from multiple sources.

7.2 CONTRIBUTION

The major contribution of this thesis is in the field of semantic event processing and event modelling. The Event Abstraction ontology is not a general purpose event ontology, but a model designed for the task [81] of inferring events from time series of observations. Nevertheless, the design of the Event Abstraction ontology and the Event Abstraction Layer is thought to cover any domain within this specific type of task. All the compared solutions in section 6.2 support analysing data from different domains. This fact highlights the trend of separating domain modelling issues from the event model and the application-specific details. Ontologies are result of consensus on how to represent specific knowledge. However, ontologies evolve over time instead of remaining as closed and static sets of concepts and relations. In my method, the use of a layered ontology structure, see section 3.3, allows restricting the interpretation of concepts that are shared across domains. Domain ontologies based on microtheories [66, 40] permit to model conceptualisations that are contradictory to each other in the same knowledge base, as far as they are internally coherent. The alignment of these lightweight domain ontologies to a common foundational ontology makes the knowledge structure very modular. Every information community can add their domain or aplication ontology if it is aligned to DUL.

One of the novelties of the presented approach is the connection between event patterns and event types, see section 3.3. Keeping the event patterns out of the ontologies instead of including them as ontology rules allows for multiple descriptions of event types. The possibility of defining multiple patterns for the same event type is either not supported, considered as future work, or not addressed in most of the compared methods in table 4. The exception are the data mining methods, Croitoru [28] and Rude and Beard [115], for which the focus is more on event detection than on classification and representation. The use of query pattern trees [28] and the definition of initiation and termination primitive events as patterns for high-level events [115] result in interesting approaches to formalise event patterns. In terms of interoperability among information communities, it is essential to allow representing different views of the world. The design of the Event Abstraction ontology is thought to accommodate different event patterns for the same event type. This enables that two information communities can use the same domain ontology to represent occurrences derived from observations. The key is on the relationship between quantitative (event patterns) and qualitative (event types) de-

scriptions of occurrences which are represented separately, but linked by (many to one) semantic annotations.

Another interesting aspect of this approach is the semi-automatic population of ontologies. The capability of performing near real-time processing of sensor data is important in environmental monitoring domains [114, 33]. Using the Event Abstraction Layer, users can schedule sensor data requests and register their patterns to infer events. This is a step forward ground the Semantic Web in the Sensor Web Janowicz [72]. This research work is a contribution to the mapping of objective measurements, made by sensors, with subjective judgements [85], in the form of event classifications within information communities. The final purpose of these mappings is to support the creation of more meaningful geospatial information.

7.3 FUTURE WORK

This section collects some issues related to my research that the thesis do not address. To do further research on inferring events from time series of observations, these issues can serve as a starting point.

Spatio-temporal reasoning on events

The presented approach does not perform reasoning on events. The EPS prototype manages event patterns that can involve multiple sensors, but no spatio-temporal awareness is used. For instance, in the case of flood monitoring it would be interesting to model a *flood wave* event as a result of *high water level* events in sensors located downstream consecutively. In this research direction, Worboys and Duckham [136] proposed a modelling framework for the inference of global events based on local observations.

Propagation of uncertainty in sensor data

Accuracy of sensor data is normally not considered in the Sensor Web [125]. Most of sensor data services providing environmental data lack machine-readable uncertainty medatada. As a consequence, the Event Abstraction Layer does not manage the propagation of errors from the sensor data to the inferred events. Uncertainty-enabled sensor data services can help to propagate uncertainty when information is inferred from data [125]. This would require changes in the sensor data parser, the Event Abstraction ontology, and the inference procedure. An example of uncertainty-enabled event inference could add the occurrence probability of a high water level event as a property of the event instance.

Considering additional data sources and formats

This research deals with the analysis of in situ and static sensor observations. Supporting alternative sensor data sources, such as mobile devices, is not addressed. The main variations would affect the inference of the event location, since a mobile sensor is still in contact with the medium it is sensing. The increase of VGI [57] will have an impact on environmental monitoring.applications. People who deploy simple weather stations on their backyards or use smartphones as noise sensors are two examples of the potential contributions of VGI.

Regarding the input format, the implemented prototype only supports data streams encoded as O&M 1.0. For full coverage of all SOS versions, a parser for O&M 2.0 [26] should be written. As described in section 4.1, adding alternative parsers does not involve substantial changes in architecture of the Event Abstraction Layer.

Exploring the Linked Data Cloud

The Event Abstraction Layer generates event instances following the Linked Data principles.[1] Finding the right data sets in the Linked Data Cloud[2] to link new inferred events would allow event consumers navigating to other data sets and, potentially, infer new information. Some of the candidate data sets could be: Linked Sensor Data[3], GeoLinked Data[4], DBpedia[5], and GeoNames[6]. An example of application using GeoNames is a Web service that returns the villages which are closer to a flooding event. As an alternative, the knowledge base can be populated with RDF triples corresponding to geographic features of the area of interest. This would allow for querying the knowledge base directly [9].

The semantics of event types, properties, and values

The Event Abstraction Layer is designed according to the pub/sub paradigm, see section 4.1. An important feature of pub/sub systems is that publishers (event producers) are decoupled from subscribers (event consumers) through the broker (intermediary processing layer) Eugster et al. [46]. The decoupling between event producers and event consumers is realised at three levels: synchronisation, spatial, and temporal [46]. However, the design of the pub/sub paradigm implies that producers and consumers have to know about the semantics of

1 Tim Berners-Lee defined the Linked Data Principles at http://www.w3.org/DesignIssues/LinkedData.html.
2 http://richard.cyganiak.de/2007/10/lod/
3 http://wiki.knoesis.org/index.php/LinkedSensorData
4 http://geo.linkeddata.es/web/guest
5 http://dbpedia.org/About
6 http://www.geonames.org/

the event model Hasan et al. [68]. Hasan et al. presents a solution based on approximate semantic matching to bridge the gap of the fourth dimension for event decoupling: the semantics of event types, properties, and values.

The design of the Event Abstraction Layer does not include a data mediation component. The lack of data mediation management also affects the decoupling between publishers and subscribers (see section 2.4.2). Since subscribers define event patterns according to observable conditions, they must know about the available data sources. In my approach, subscribers must have knowledge about the data sources and the event definitions. Therefore, the possibility of vocabulary-free subscriptions to events is interesting to be considered as future work in order to build full loosely coupled systems.

8 APPENDIX

8.1 EPS SOFTWARE ENGINEERING

8.1.1 *EPS class diagram*

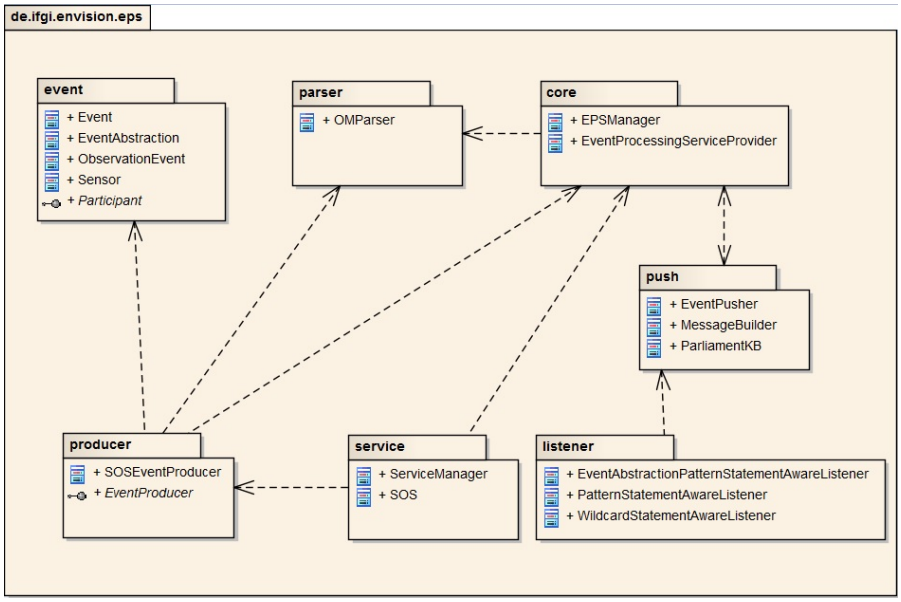

Figure 25: Class diagram of the Event Processing Service.

8.2 SOS DATA REQUEST

Listing 5: getObservations request template used by the registerService call.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetObservation xmlns="http://www.opengis.net/sos/1.0"
  xmlns:ows="http://www.opengis.net/ows/1.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:om="http://www.opengis.net/om/1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/sos/1.0
  http://schemas.opengis.net/sos/1.0.0/sosGetObservation.xsd"
  service="SOS" version="1.0.0" srsName="urn:ogc:def:crs:EPSG
      ::4326">
  <offering>[OFFERING]</offering>
  <eventTime>
    <ogc:TM_During>
      <ogc:PropertyName>om:samplingTime</ogc:PropertyName>
      <gml:TimePeriod>
        <gml:beginPosition>
          [CURRENT_SYSTEM_TIME-TIME_INTERVAL]
        </gml:beginPosition>
        <gml:endPosition>[CURRENT_SYSTEM_TIME]</gml:endPosition>
      </gml:TimePeriod>
    </ogc:TM_During>
  </eventTime>
  <observedProperty>[OBSERVED_PROPERTY]</observedProperty>
  <responseFormat>"om/1.0.0"</responseFormat>
</GetObservation>
```

8.3 EVENT PATTERNS

8.3.1 *Event patterns for Romanian Waters*

The experiments presented in section 6.1 analyse SOS water level observations measured in *metres above the Adriatic Sea* (mdMA). However, Romanian Waters defines thresholds in centimetres (cm). Before registering the event patterns for Romanian Waters, I converted the thresholds from cm to mdMA. For this conversion, I searched for an observation which coincide with one of the thresholds: on January 21st 2013 at 06:00 the water level was 600 cm according to Romanian Waters.[1] In the experiment data set, the water level indicates 70.17 mdMA at that instant. Domain experts recommend to calculate the rest of the thresholds using a reference level, see Listing 6. Table 5 contains the values of the three water level thresholds for Bazias. All the Romanian Waters event patterns formalised for this experiment are listed below. Listing 7 includes the event patterns based on flood thresholds. Listing 8 contains the event patterns for flood stages.

Listing 6: Conversion of flood thresholds for Bazias from cm to mdMA.

```
SOS observation = Reference level mdMA + RW observation

Reference level mdMA = SOS obs - RW obs =
  70.17 - 6.00 = 64.17 mdMA

Flood threshold = 64.17 + 6.90 = 71.07 mdMA

Danger threshold = 64.17 + 7.00 = 71.17 mdMA
```

Thresholds / Units of measurement	**cm**	**mdMA**
Alert stage	600	70.17
Flood stage	690	71.07
Danger stage	700	71.17

Table 5: Thresholds for flood categories at Bazias in cm and mdMA.

1 Online GIS portal available at http://gis2.rowater.ro:8989/SituatieHidrologica.html.

Listing 7: Event patterns for exceeding and deceeding flood stage thresholds at Bazias defined by Romanian Waters.

```
%Attention threshold exceeded:
SELECT obs1, obs2
FROM pattern[every (obs1=ObservationEvent( observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 70.17) ->
 obs2=ObservationEvent( observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.17)) ]

%Attention threshold deceeded:
SELECT obs1, obs2
FROM pattern[every (obs1=ObservationEvent( observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.17) ->
 obs2=ObservationEvent( observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 70.17)) ]

%Flood threshold exceeded:
SELECT obs1, obs2
FROM pattern[every (obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 71.07) ->
 obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 71.07)) ]

%Flood threshold deceeded:
SELECT obs1, obs2
FROM pattern[every (obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 71.07) ->
 obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 71.07)) ]

%Danger threshold exceeded:
SELECT obs1, obs2
FROM pattern[every (obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 71.17) ->
```

```
 obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 71.17)) ]

%Danger threshold deceeded:
SELECT obs1, obs2
FROM pattern[every (obs1=ObservationEvent(observer.id =
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 71.17) ->
 obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 71.17)) ]
```

Listing 8: Event patterns for flood stage periods at Bazias.

```
%Alert stage:
SELECT obs1, obs2
FROM pattern[every (obs1=EventAbstraction(eventType=
 'http://wsmls.googlecode.com/svn/trunk/local/water/0.5/flood.rdf
     #AlertThresholdExceeded')
 -> obs2=EventAbstraction(eventType=
 'http://wsmls.googlecode.com/svn/trunk/local/water/0.5/flood.rdf
     #AlertThresholdDeceeded'))]

%Flood stage:
SELECT obs1, obs2
FROM pattern[every (obs1=EventAbstraction(eventType=
 'http://wsmls.googlecode.com/svn/trunk/local/water/0.5/flood.rdf
     #FloodThresholdExceeded')
 -> obs2=EventAbstraction(eventType=
 'http://wsmls.googlecode.com/svn/trunk/local/water/0.5/flood.rdf
     #FloodThresholdDeceeded'))]

%Danger stage:
SELECT obs1, obs2
FROM pattern[every (obs1=EventAbstraction(eventType=
 'http://wsmls.googlecode.com/svn/trunk/local/water/0.5/flood.rdf
     #DangerThresholdExceeded')
 -> obs2=EventAbstraction(eventType=
 'http://wsmls.googlecode.com/svn/trunk/local/water/0.5/flood.rdf
     #DangerThresholdDeceeded'))]
```

8.3.2 *Event patterns for Hidroelectrica Romania*

Listing 9: Event patterns for low and high water level events at Iron Gates I and II.

```
%Low water level at the Iron Gates I
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 63) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 63)]


%High water level at the Iron Gates I
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 69.59) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 69.59)]


%Low water level at the Iron Gates II
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 39.4) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 39.4)]


%High water level for the Iron Gates II
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value < 41.0) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0034',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 41.0)]
```

Listing 10: Event patterns for high water level events at Bazias.

```
%b. Between 3000 and 3500 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 3000 and 3500) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 69.87)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )

%c. Between 3500 and 4000 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 3500 and 4000) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 69.92)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )

%d. Between 4000 and 4500 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 4000 and 4500) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 69.97)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )

%e. Between 4500 and 5000 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 4500 and 5000) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.02)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )
```

```
%f. Between 5000 and 5500 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 5000 and 5500) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.10)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )

%g. Between 5500 and 6000 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 5500 and 6000) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.17)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )

%h. Between 6000 and 6500 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 6000 and 6500) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.25)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )

%i. Between 6500 and 7000 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 6500 and 7000) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.31)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )
```

```
%j. Between 7000 and 7500 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 7000 and 7500) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.38)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )

%k. Between 7500 and 11500 m3/s:
SELECT obs1, obs2
FROM pattern[every obs1=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0006',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterflow',
 value between 7500 and 11500) ->
 every obs2=ObservationEvent(observer.id=
 'urn:ogc:object:feature:Sensor:CSRO:csro-sensor-0002',
 observedProperty='urn:ogc:def:phenomenon:OGC:1.0.30:waterlevel',
 value > 70.4)]
WHERE obs2.time.between(obs1.time,obs1.time.plus(23 hours 59 min)
    )
```

8.3.3 *Event patterns for real-time experiment*

- IF the discharge is below 3000 m^3/s AND the water level goes below 65.5 cm in the following 10 seconds THEN a HR:LOWWATERLEVEL instance is created.
- IF the discharge is between 3000 and 3500 m^3/s AND the water level exceeds 69.81 cm in the following 10 seconds THEN a HR:HIGHWATERLEVEL instance is created.
- IF the discharge is above 3500 m^3/s AND the water level exceeds 71.0 cm in the following 10 seconds THEN a HR:HIGHWATERLEVEL instance is created.
- IF the discharge is above 3500 m^3/s AND the water level goes below 665 cm in the following 10 seconds THEN a HR:LOWWATERLEVEL instance is created.
- IF the attention threshold (70.17 cm) is exceeded THEN an instance of type RW:ATTENTIONTHRESHOLDEXCEEDED is created.
- IF the attention threshold (70.17 cm) is deceeded THEN an instance of type RW:ATTENTIONTHRESHOLDDECEEDED is created.
- IF the flooding threshold (71.07 cm) is exceeded THEN an instance of type RW:FLOODINGTHRESHOLDEXCEEDED is created.
- IF the flooding threshold (71.07 cm) is deceeded THEN an instance of type RW:FLOODINGTHRESHOLDDECEEDED is created.
- IF an RW:ATTENTIONTHRESHOLDEXCEEDED event is followed by an event of type RW:ATTENTIONTHRESHOLDDECEEDED event THEN an RW:ATTENTIONSTAGE instance is created.
- IF an RW:FLOODINGTHRESHOLDEXCEEDED event is followed by an event of type RW:FLOODINGTHRESHOLDDECEEDED THEN an RW:FLOODINGSTAGE instance is created.

8.4 PROPERTIES OF AN EVENT ABSTRACTION INSTANCE

Property	Definition	Domain	Literal format
EABS:HASBEGINNING EABS:HASEND	Relation between a time interval and a the literal representation of its initial time instant. Equivalent to DUL:HASINTERVALDATE.	DUL:TIMEINTERVAL	XML Schema dateTime
DC:IDENTIFIER (Dublin Core)	Unambiguous reference to the resource within a given context.	SSN:PROPERTY	String
DC:SOURCE (Dublin Core)	Relation to a resource from which the described resource is derived.	DUL:INFORMATIONOBJECT	URL
DUL:HASDATAVALUE	Datatype property that encodes values for an entity.	DUL:INFORMATIONOBJECT	XML
DUL:HASDATAVALUE	Datatype property that encodes values for an entity.	EABS:EVENTABSTRACTIONRULE	String
DUL:HASREGIONDATAVALUE	Datatype property that encodes values for a region.	DUL:SPACEREGION	Well-Known Text
GEO:ASWKT	Relation between a geometry and its literal representation in Well-Known Text.	GEO:GEOMETRY	Well-Known Text
RDFS:LABEL	Property that provides a human-readable resource name.	EABS:EVENTDETECTIONPROCEDURE	String

Table 6: Properties of the Event Abstraction instance.

8.5 SPARQL QUERIES

Listing 11: SPARQL query for all the event instances produced in the historical data experiment.

```
PREFIX eabs: <http://wsmls.googlecode.com/svn/trunk/global/Event-
    abstraction/0.2/EventAbstraction.rdf#>
PREFIX dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>

SELECT ?event ?eventType ?geometryAsWKT ?begin ?end
WHERE { ?event a eabs:EventAbstraction .
        ?event eabs:isClassifiedBy ?type .
        ?type a ?eventType .
        ?event geo:hasGeometry ?geometry .
        ?geometry geo:asWKT ?geometryAsWKT .
        ?event dul:isObservableAt ?temporalLocation .
        ?temporalLocation dul:hasBeginning ?begin .
        ?temporalLocation dul:hasEnd ?end
}
```

Listing 12: SPARQL query to retrieve all the events located at Bazias.

```
PREFIX eabs: <http://wsmls.googlecode.com/svn/trunk/global/Event-
    abstraction/0.2/EventAbstraction.rdf#>
PREFIX dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>

SELECT ?event ?eventType ?begin ?end
WHERE {
   ?event a eabs:EventAbstraction .
   ?event eabs:isClassifiedBy ?type .
   ?type a ?eventType .
   ?event geo:hasGeometry ?geometry .
   ?geometry geo:asWKT "POINT (21.39046182 44.80861612)"^^<http
       ://www.opengis.net/ont/sf#wktLiteral> .
   ?event dul:isObservableAt ?temporalLocation .
   ?temporalLocation dul:hasBeginning ?begin .
   ?temporalLocation dul:hasEnd ?end .
}
```

Listing 13: SPARQL query to retrieve all the events located at Iron Gates I.

```
PREFIX eabs: <http://wsmls.googlecode.com/svn/trunk/global/Event-
    abstraction/0.2/EventAbstraction.rdf#>
PREFIX dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>

SELECT ?event ?eventType ?begin ?end
WHERE {
   ?event a eabs:EventAbstraction .
   ?event eabs:isClassifiedBy ?type .
   ?type a ?eventType .
   ?event geo:hasGeometry ?geometry .
   ?geometry geo:asWKT "POINT (22.53039649 44.675289)"^^<http://
       www.opengis.net/ont/sf#wktLiteral> .
   ?event dul:isObservableAt ?temporalLocation .
   ?temporalLocation dul:hasBeginning ?begin .
   ?temporalLocation dul:hasEnd ?end .
}
```

Listing 14: SPARQL query to retrieve all the events located at Iron Gates II.

```
PREFIX eabs: <http://wsmls.googlecode.com/svn/trunk/global/Event-
    abstraction/0.2/EventAbstraction.rdf#>
PREFIX dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>

SELECT ?event ?eventType ?begin ?end
WHERE {
   ?event a eabs:EventAbstraction .
   ?event eabs:isClassifiedBy ?type .
   ?type a ?eventType .
   ?event geo:hasGeometry ?geometry .
   ?geometry geo:asWKT "POINT (22.565832 44.308002)"^^<http://www
       .opengis.net/ont/sf#wktLiteral> .
   ?event dul:isObservableAt ?temporalLocation .
   ?temporalLocation dul:hasBeginning ?begin .
   ?temporalLocation dul:hasEnd ?end .
}
```

Listing 15: SPARQL query to retrieve Attention Stage events.

```
PREFIX eabs: <http://wsmls.googlecode.com/svn/trunk/global/Event-
    abstraction/0.2/EventAbstraction.rdf#>
PREFIX dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>
PREFIX rw: <http://wsmls.googlecode.com/svn/trunk/application/
    EventType/RomanianWaters/RomanianWatersEventTypes.rdf#>

SELECT ?event ?geometryAsWKT ?begin ?end
WHERE { ?event a eabs:EventAbstraction .
        ?event eabs:isClassifiedBy ?type .
        ?type a rw:AttentionStage .
        ?event geo:hasGeometry ?geometry .
        ?geometry geo:asWKT ?geometryAsWKT .
        ?event dul:isObservableAt ?temporalLocation .
        ?temporalLocation dul:hasBeginning ?begin .
        ?temporalLocation dul:hasEnd ?end
}
```

Listing 16: SPARQL query to retrieve Flood Stage events.

```
PREFIX eabs: <http://wsmls.googlecode.com/svn/trunk/global/Event-
    abstraction/0.2/EventAbstraction.rdf#>
PREFIX dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>
PREFIX rw: <http://wsmls.googlecode.com/svn/trunk/application/
    EventType/RomanianWaters/RomanianWatersEventTypes.rdf#>

SELECT ?event ?geometryAsWKT ?begin ?end
WHERE { ?event a eabs:EventAbstraction .
        ?event eabs:isClassifiedBy ?type .
        ?type a rw:FloodStage .
        ?event geo:hasGeometry ?geometry .
        ?geometry geo:asWKT ?geometryAsWKT .
        ?event dul:isObservableAt ?temporalLocation .
        ?temporalLocation dul:hasBeginning ?begin .
        ?temporalLocation dul:hasEnd ?end
}
```

Listing 17: SPARQL query to retrieve Danger Stage events.

```
PREFIX eabs: <http://wsmls.googlecode.com/svn/trunk/global/Event-
    abstraction/0.2/EventAbstraction.rdf#>
PREFIX dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
PREFIX geo: <http://www.opengis.net/ont/geosparql#>
PREFIX rw: <http://wsmls.googlecode.com/svn/trunk/application/
    EventType/RomanianWaters/RomanianWatersEventTypes.rdf#>

SELECT ?event ?geometryAsWKT ?begin ?end
WHERE { ?event a eabs:EventAbstraction .
        ?event eabs:isClassifiedBy ?type .
        ?type a rw:DangerStage .
        ?event geo:hasGeometry ?geometry .
        ?geometry geo:asWKT ?geometryAsWKT .
        ?event dul:isObservableAt ?temporalLocation .
        ?temporalLocation dul:hasBeginning ?begin .
        ?temporalLocation dul:hasEnd ?end
}
```

8.6 REAL-TIME DATA SIMULATION

Time step / Observed property	**Water flow (m^3/s)**	**Water level (cm)**	**Time step / Observed property**	**Water flow (m^3/s)**	**Water level (cm)**
t_0	2900	65.9	t_{18}	3620	74.9
t_1	2940	66.4	t_{19}	3620	74.4
t_2	2980	66.9	t_{20}	3620	73.9
t_3	3020	67.4	t_{21}	3620	73.4
t_4	3060	67.9	t_{22}	3620	72.9
t_5	3100	68.4	t_{23}	3620	72.4
t_6	3140	68.9	t_{24}	3620	71.9
t_7	3180	69.4	t_{25}	3620	71.4
t_8	3220	69.9	t_{26}	3620	70.9
t_9	3260	70.4	t_{27}	3620	70.4
t_{10}	3300	70.9	t_{28}	3620	69.9
t_{11}	3340	71.4	t_{29}	3620	69.4
t_{12}	3380	71.9	t_{30}	3620	68.9
t_{13}	3420	72.4	t_{31}	3620	68.4
t_{14}	3460	72.9	t_{32}	3620	67.9
t_{15}	3500	73.4	t_{33}	3620	67.4
t_{16}	3540	73.9	t_{34}	3620	66.9
t_{17}	3580	74.4	t_{35}	3620	66.4

Table 7: Simulated data inserted in the SOS for the real-time experiment.

BIBLIOGRAPHY

[1] Samer Abdallah, Yves Raimond, and Mark Sandler. An Ontology-based Approach to Information Management for Music Analysis Systems. In *Audio Engineering Society Convention 120*, Paris, France, 2006. URL http://www.aes.org/e-lib/browse.cfm?elib=13574.

[2] Jochen Albrecht, Brandon Derman, and Laxmi Ramasubramanian. Geo-ontology Tools: The Missing Link. *Transactions in GIS*, 12(4):409–424, August 2008. ISSN 13611682. doi: 10.1111/j.1467-9671.2008.01108.x. URL http://doi.wiley.com/10.1111/j.1467-9671.2008.01108.x.

[3] James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, November 1983. ISSN 0001-0782. doi: 10.1145/182.358434. URL http://doi.acm.org/10.1145/182.358434.

[4] James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, (23):123–154, 1984.

[5] James F. Allen and George Ferguson. *Actions and Events in Interval Temporal Logic*, volume 4. 1994. doi: 10.1093/logcom/4.5.531. URL http://logcom.oxfordjournals.org/cgi/doi/10.1093/logcom/4.5.531.

[6] James F. Allen and Patrick J. Hayes. A common-sense theory of time. In Joshi Aravind, editor, *Proceedings of the Ninth International Joint Conference on Artificial Intelligence (IJCAI'85)*, pages 528–531. International Joint Conferences on Artificial Intelligence, Inc., 1985.

[7] Darko Anicic, Sebastian Rudolph, Paul Fodor, and Nenad Stojanovic. Stream Reasoning and Complex Event Processing in ETALIS. *Semantic Web Journal*, 1:1–5, 2009. URL http://www.semantic-web-journal.net/content/stream-reasoning-and-complex-event-processing-etalis.

[8] Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and issues in data stream systems. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '02, pages 1–16, New York, NY, USA, 2002. ACM. ISBN 1-58113-507-6. doi: 10.1145/543613.543615. URL http://doi.acm.org/10.1145/543613.543615.

[9] Robert Battle and Dave Kolas. Enabling the Geospatial Semantic Web with Parliament and GeoSPARQL. *Semantic Web Journal (SWJ)*, 3(4):355–370, 2012. URL http://iospress.metapress.com/content/h470t26742n7x506/.

[10] By Tim Berners-lee, James Hendler, and Ora Lassila. The Semantic Web, 2001. URL http://sernt55.essex.ac.uk/ce/ce313/site%20for%202011-12%20-%202012-05-01%20-%20v1/communication/10.1.1.115.9584-extract.pdf.

[11] A. Yaser Bishr, H. Pundt, W. Kuhn, and M. Radwan. Probing the Concept of Information Communities - A First Step Toward Semantic Interoperability. In Michael Goodchild, Max Egenhofer, Robin Fegeas, and Cliff Kottman, editors, *Interoperating Geographic Information Systems*, volume 495 of *The Springer International Series in Engineering and Computer Science*, pages 55–69. Springer US, 1999. ISBN 978-1-4613-7363-6. doi: 10.1007/978-1-4615-5189-8_5. URL http://dx.doi.org/10.1007/978-1-4615-5189-8_5.

[12] M. Botts, G. Percivall, C. Reed, and J. Davidson. OGC® Sensor Web Enablement: Overview and high level architecture. In Silvia Nittel, Alexandros Labrinidis, and Anthony Stefanidis, editors, *GeoSensor Networks*, Lecture Notes in Computer Science, pages 175–190. Springer Berlin Heidelberg, 2008.

[13] Don Box, Luis Felipe Cabrera, Craig Critchley, Francisco Curbera, Donald Ferguson, Steve Graham, David Hull, Gopal Kakivaya, Amelia Lewis, Brad Lovering, Peter Niblett, David Orchard, Shivajee Samdarshi, Jeffrey Schlimmer, Igor Sedukhin, John Shewchuk, Sanjiva Weerawarana, and David Wortendyke. Web Services Eventing (WS-Eventing), 2006. URL http://www.w3.org/Submission/WS-Eventing/.

[14] Boyan Brodaric and Mark Gahegan. Experiments to Examine the Situated Nature of Geoscientific Concepts. *Spatial Cognition & Computation*, 7(1):61–95, 2007. doi: 10.1080/13875860701337934. URL http://www.tandfonline.com/doi/abs/10.1080/13875860701337934.

[15] Arne Broering, Krzysztof Janowicz, Christoph Stasch, and Werner Kuhn. Semantic Challenges for Sensor Plug and Play. In James D. Carswell, A. Stewart Fotheringham, and Gavin McArdle, editors, *Web and Wireless Geographical Information Systems*, volume 5886 of *Lecture Notes in Computer Science*, pages 72–86. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-10600-2. doi: 10.1007/978-3-642-10601-9_6. URL http://dx.doi.org/10.1007/978-3-642-10601-9_6.

[16] Arne Broering, Johannes Echterhoff, Simon Jirka, Ingo Simonis, Thomas Everding, Christoph Stasch, Steve Liang, and Rob Lemmens. New Generation Sensor Web Enablement. *Sensors*, 11(3):2652–2699, March 2011. ISSN 1424-8220. doi: 10.3390/s110302652. URL http://www.mdpi.com/1424-8220/11/3/2652/.

[17] Daniel G. Brown, Rick Riolo, and Derek T. Robinson. Spatial process and data models: Towards integration of agent-based models and GIS. *Journal of Geographic Systems*, 7:25–47, 2005.

[18] R. Casati and A.C. Varzi, editors. *Events (The International Research Library of Philosophy)*. Dartmouth Publishing Co., Aldershot, England, 1996.

[19] R. Casati and A.C. Varzi. *50 years of events: an annotated bibliography, 1947 to 1997*. Bowling Green State Univ philosophy, 1997.

[20] K. Mani Chandy and Roy Schulte. What is Event Driven Architecture (EDA) and Why Does it Matter?, 2007. URL http://complexevents.com/wp-content/uploads/2007/07/EDA%20article%20long%20Chandy%20and%20Schulte%2015%20July%202007%20final.pdf.

[21] David Chappel and Lily Liu. Web Services Brokered Notification 1.3. Technical report, OASIS Standard, 2006. URL http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-os.pdf.

[22] Nicholas Chrisman. Beyond the snapshot: changing the approach to change, error, and process. In Max Egenhofer and R. G. Golledge, editors, *Spatial and temporal reasoning in geographic information systems*, chapter 6, pages 85–93. Oxford University Press, USA, 1998.

[23] Michael Compton, Corey Henson, Holger Neuhaus, Laurent Lefort, and Amit Sheth. A Survey of the Semantic Specification of Sensors. In *Proceedings of the 8th International Semantic Web Conference (ISWC 2009), 2nd International Workshop on Semantic Sensor Networks*, Washington DC, 2009.

[24] Michael Compton, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort, Myriam Leggieri, Holger Neuhaus, Andriy Nikolov, Kevin Page, Alexandre Passant, Amit Sheth, and Kerry Taylor. The SSN Ontology of the W3C Semantic Sensor Network Incubator Group. *Web Semantics: Science, Services and Agents on the*

World Wide Web, 0(0), 2012. ISSN 1570-8268. URL http://www.websemanticsjournal.org/index.php/ps/article/view/292.

[25] Simon Cox. Observations and Measurements - Part 1 - Observation Schema. Technical report, Open Geospatial Consortium, 2007. URL http://portal.opengeospatial.org/files/?artifact_id=22466.

[26] Simon Cox. Observations and Measurements - XML Implementation (version 2.0). Technical report, Open Geospatial Consortium, 2011. URL http://portal.opengeospatial.org/files/?artifact_id=41510.

[27] Max Craglia, Michael F Goodchild, Alessandro Annoni, Gilberto Camara, Michael Gould, Werner Kuhn, David Mark, Ian Masser, David Maguire, Steve Liang, and Ed Parsons. Next-generation digital earth: A position paper from the vespucci initiative for the advancement of geographic information science. *International Journal of Spatial Data Infrastructures Research (IJSDIR)*, 3:146–167, 2008. doi: 10.2902/1725-0463.2008.03.art9.

[28] Arie Croitoru. Deriving and Mining Spatiotemporal Event Schemas in In-Situ Sensor Data. In Osvaldo Gervasi, Beniamino Murgante, Antonio Laganà , David Taniar, Youngsong Mun, and Marina Gavrilova, editors, *Computational Science and Its Applications - ICCSA 2008*, volume 5072 of *Lecture Notes in Computer Science*, pages 740–755. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-69838-8. doi: 10.1007/978-3-540-69839-5_54. URL http://dx.doi.org/10.1007/978-3-540-69839-5_54.

[29] Gianpaolo Cugola and Alessandro Margara. Processing flows of information: From data stream to complex event processing. *ACM Comput. Surv.*, 44(3):15:1–15:62, June 2012. ISSN 0360-0300. doi: 10.1145/2187671.2187677. URL http://doi.acm.org/10.1145/2187671.2187677.

[30] D. Davidson. The logical form of action sentences. *Essays on actions and events*, 5:105–148, 1967.

[31] Clodoveu A Davis, Frederico T Fonseca, and Gilberto Camara. Understanding Global Change: The Role of Geographic Information Science in the Integration of People and Nature. In Timothy Nyerges, Helen Couclelis, and Robert McMaster, editors, *The SAGE Handbook of GIS and Society*, pages 123–137. SAGE Publications Limited, Thousand Oaks, CA, 2009.

[32] Doug Davis, Ashok Malhotra, Katy Warr, and Wu Chou. Web Services Eventing (WS-Eventing). Technical report, W3C, 2011. URL http://www.w3.org/TR/2011/REC-ws-eventing-20111213/.

[33] B. De Longueville, A. Annoni, S. Schade, N. Ostlaender, and C. Whitmore. Digital Earth's Nervous System for Crisis Events: Real-Time Sensor Web Enablement of Volunteered Geographic Information. *International Journal of Digital Earth*, 3(3):242–259, 2010.

[34] Kevin A. Delin. The Sensor Web: A macro-instrument for coordinated sensing. *Sensors*, 2(7):270–285, 2002.

[35] Kevin A Delin and Shannon P Jackson. The Sensor Web: A New Instrument Concept. In Michael R Descour and Juha T Rantala, editors, *Proceedings of SPIE*, number January, pages 20–26. Spie, 2001.

[36] Alan Demers, Johannes Gehrke, Biswanath Panda, Mirek Riedewald, Varun Sharma, and Walker White. Cayuga: A General Purpose Event Monitoring System. In *CIDR 2007*, pages 412–422, 2007. URL http://www.ccs.neu.edu/home/mirek/papers/2007-CIDR-CayugaImp.pdf.

[37] Anusuriya Devaraju. *Representing and Reasoning about Geographic Occurrences in the Sensor Web*. PhD thesis, Institute for Geoinformatics, University of Muenster., 2012.

[38] Anusuriya Devaraju and Tomi Kauppinen. Sensors Tell More than They Sense: Modeling and Reasoning about Sensor Observations for Understanding Weather Events. *Special Issue on Semantic Sensor Networks, International Journal of Sensors, Wireless Communications and Control*, 2(1), 2012. URL http://kauppinen.net/tomi/devaraju-kauppinen-sensors-2011.pdf.

[39] Martin Doerr. The CIDOC conceptual reference module: an ontological approach to semantic interoperability of metadata. *AI magazine*, 24(3):75–92, 2003.

[40] Stephanie Duce and Krzysztof Janowicz. Microtheories for spatial data infrastructures - accounting for diversity of local conceptualizations at a global level. In Sara Irina Fabrikant, Tumasch Reichenbacher, Marc Kreveld, and Christoph Schlieder, editors, *Geographic Information Science*, volume 6292 of *Lecture Notes in Computer Science*, pages 27–41. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15299-3. doi: 10.1007/978-3-642-15300-6_3. URL http://dx.doi.org/10.1007/978-3-642-15300-6_3.

[41] Johannes Echterhoff and Thomas Everding. OpenGIS Sensor Event Service Interface Specification (proposed). Technical report, Open Geospatial Consortium, 2008. URL http://portal.opengeospatial.org/files/?artifact_id=29576.

[42] Michael Eckert. Complex Event Processing with XChangeEQ. December 2008. URL http://nbn-resolving.de/urn:nbn:de:bvb:19-94051.

[43] T. Erl. *Service-oriented architecture*. Prentice Hall PTR, 2004.

[44] Opher Etzion. Semantic approach to event processing. In *Proceedings of the 2007 inaugural international conference on Distributed event-based systems*, DEBS '07, pages 139–139, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-665-3. doi: 10.1145/1266894.1266920. URL http://doi.acm.org/10.1145/1266894.1266920.

[45] Opher Etzion and Peter Niblett. *Event Processing in Action*. Manning Publications Co., Greenwich, CT, USA, 1st edition, 2010. ISBN 1935182218, 9781935182214.

[46] Patrick T. H. Eugster, Pascal A Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The Many Faces of Publish/Subscribe. *ACM Computing Surveys*, 35(2):114–131, 2003. doi: 10.1145/857076.857078. URL http://doi.acm.org/10.1145/857076.857078.

[47] Thomas Everding and Johannes Echterhoff. OGC OWS-6 SWE Event Architecture Engineering Report. Technical report, Open Geospatial Consortium (OGC), 2009. URL https://portal.opengeospatial.org/files/?artifact_id=33347.

[48] Antony Galton. A critical examination of allen's theory of action and time. *Artificial Intelligence*, 42(2-3):159–188, March 1990. ISSN 0004-3702. doi: 10.1016/0004-3702(90)90053-3. URL http://dx.doi.org/10.1016/0004-3702(90)90053-3.

[49] Antony Galton. On What Goes On: The Ontology of Processes and Events. In *Proceedings of the 2006 conference on Formal Ontology in Information Systems: Proceedings of the Fourth International Conference (FOIS 2006)*, pages 4–11, Amsterdam, The Netherlands, The Netherlands, 2006. IOS Press. ISBN 1-58603-685-8. URL http://dl.acm.org/citation.cfm?id=1566079.1566083.

[50] Antony Galton. Spatial and temporal knowledge representation. *Earth Science Informatics*, 2(3):169–187, 2009. ISSN 1865-0473. doi: 10.1007/s12145-009-0027-6. URL http://dx.doi.org/10.1007/s12145-009-0027-6.

[51] Antony Galton and Juan Carlos Augusto. Two Approaches to Event Definition. In Abdelkader Hameurlain, Rosine Cicchetti, and Roland Traunmüller, editors, *Database and Expert Systems Applications*, pages 547–556. Springer Berlin Heidelberg, 2002. ISBN 978-3-540-44126-7. doi: 10.1007/3-540-46146-9_54.

[52] Antony Galton and Michael Worboys. Processes and Events in Dynamic Geo-Networks. In M. Andrea Rodríguez, Isabel Cruz, Sergei Levashkin, and Max J. Egenhofer, editors, *GeoSpatial Semantics*, volume 3799 of *Lecture Notes in Computer Science*, pages 45–59. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-30288-9. doi: 10.1007/11586180_4. URL http://dx.doi.org/10.1007/11586180_4.

[53] Aldo Gangemi. Ontology Design Patterns for Semantic Web Content. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *The Semantic Web - ISWC 2005*, volume 3729 of *Lecture Notes in Computer Science*, pages 262–276. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-29754-3. doi: 10.1007/11574620_21. URL http://dx.doi.org/10.1007/11574620_21.

[54] Aldo Gangemi. DOLCE+DnS Ultralite ontology, 2010. URL http://www.loa.istc.cnr.it/ontologies/DUL.owl.

[55] Aldo Gangemi and Peter Mika. Understanding the Semantic Web through Descriptions and Situations. In Robert Meersman, Zahir Tari, and Douglas Schmidt, editors, *Proceedings of On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, pages 689–706. Springer Berlin Heidelberg, 2003. doi: 10.1007/978-3-540-39964-3_44. URL http://dx.doi.org/10.1007/978-3-540-39964-3_44.

[56] Gary Geller. Model Web Development. Technical report, Group on Earth Observations, 2009. URL http://www.earthobservations.org/documents/tasksheets/latest/AR-09-02d.pdf.

[57] Michael Goodchild. Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69:211–221, 2007. ISSN 0343-2521. URL http://dx.doi.org/10.1007/s10708-007-9111-y. 10.1007/s10708-007-9111-y.

[58] Michael F Goodchild, Huadong Guo, Alessandro Annoni, Ling Bian, Kees de Bie, Frederick Campbell, Max Craglia, Manfred Ehlers, John van Genderen, Davina Jackson, Anthony J Lewis, Martino Pesaresi, Gábor Remetey-Fülöpp, Richard Simpson, Andrew Skidmore, Changlin Wang, and Peter Woodgate. Next-generation Digital Earth. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 109(28):11088–94, 2012. doi: 10.1073/pnas.1202383109. URL http://www.pnas.org/cgi/content/long/109/28/11088.

[59] Al Gore. The digital earth: understanding our planet in the 21st century. *Australian surveyor*, 43(2):89–91, 1998.

[60] Steve Graham, David Hull, and Bryan Murray. Web Services Base Notification 1.3. Technical report, OASIS Standard, 2006. URL http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf.

[61] Pierre Grenon and Barry Smith. SNAP and SPAN: Towards Dynamic Spatial Ontology. *Spatial Cognition and Computation*, 4 (1):69–103, 2004.

[62] Karl Eric Grossner. *Representing Historical Knowledge in Geographic Information Systems*. PhD thesis, University of California, Santa Barbara, 2010. URL http://kgeographer.com/assets/Grossner_dissertation_2010.pdf.

[63] Mircea Grosu, Silviu Trasca, Cosmin Udroiu, Mihaela Ionescu, Dora Grosu, and Marius Vilcu. ENVISION Deliverable 1.7 - Real-time Pilot Case Definition and Requirements Specification. Technical report, ENVISION project, 2012. URL http://www.envision-project.eu/wp-content/uploads/2012/05/D1-7_v1-1.pdf.

[64] Nicola Guarino. Formal Ontology and Information Systems. In *Proceedings of FOIS'98*, pages 3–15. IOS Press, 1998. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.29.1776&rep=rep1&type=pdf.

[65] Nicola Guarino. The Ontological Level : Revisiting 30 Years of Knowledge Representation. *Knowledge Creation Diffusion Utilization*, pages 52–67, 2009.

[66] R. Guha, R. Mccool, and R. Fikes. Contexts for the semantic web. In *International Semantic Web Conference, volume 3298 of Lecture Notes in Computer Science*, pages 32–46. Springer, 2004.

[67] Bridgette M Hard, Barbara Tversky, and David S Lang. Making sense of abstract events: Building event schemas. *Memory & cognition*, 34(6):1221–1235, 2006.

[68] Souleiman Hasan, Sean O Riain, and Edward Curry. Approximate Semantic Matching of Heterogeneous Events. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, pages 252—-263, Berlin, Germany, 2012. ACM, New York, NY, USA. URL http://doi.acm.org/10.1145/2335484.2335512.

[69] N. W. J. Hazelton, F. J. Leahy, and I. P. Williamson. Integrating dynamic modeling and geographic information systems. *URISA Journal*, 4(2):47–58, 1992.

[70] Cory Henson, Amit Sheth, and Krishnaprasad Thirunarayan. Semantic Perception: Converting Sensory Observations to Abstractions. *Internet Computing, IEEE*, 16(2):26–34, 2012. ISSN 1089-7801. doi: 10.1109/MIC.2012.20.

[71] Cory A. Henson, Josh K. Pschorr, Amit P. Sheth, and Krishnaprasad Thirunarayan. SemSOS: Semantic sensor Observation Service. In *Proceedings of the 2009 International Symposium on Collaborative Technologies and Systems*, CTS '09, pages 44–53, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-1-4244-4584-4. doi: 10.1109/CTS.2009.5067461. URL http://dx.doi.org/10.1109/CTS.2009.5067461.

[72] Krzysztof Janowicz. The role of space and time for knowledge organization on the semantic web. *Semantic Web Journal*, 1(1): 25–32, 2010.

[73] Krzysztof Janowicz and Michael Compton. The Stimulus-Sensor-Observation Ontology Design Pattern and its Integration into the Semantic Sensor Network Ontology. In Kerry Taylor, Arun Ayyagari, and David De Roure, editors, *The 3rd International workshop on Semantic Sensor Networks 2010 (SSN10) in conjunction with the 9th International Semantic Web Conference (ISWC 2010)*, Shangai, China, 2010.

[74] Krzysztof Janowicz and Pascal Hitzler. The Digital Earth as knowledge engine. *Semantic Web*, 3(3):213–221, 2012.

[75] Krzysztof Janowicz and Pascal Hitzler. Key Ingredients for Your Next Semantics Elevator Talk. In Silvana Castano, Panos Vassiliadis, Laks V. Lakshmanan, and Mong Li Lee, editors, *Advances in Conceptual Modeling*, volume 7518 of *Lecture Notes in Computer Science*, pages 213–220. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-33998-1. doi: 10.1007/978-3-642-33999-8_27. URL http://dx.doi.org/10.1007/978-3-642-33999-8_27.

[76] Krzysztof Janowicz, Martin Raubal, and Sergei (Eds.) Levashkin, editors. *GeoSpatial Semantics: Third International Conference, GeoS 2009, Mexico City, Mexico, December 3-4, 2009, Proceedings*, volume 5892. Springer, 2009.

[77] Krzysztof Janowicz, Sven Schade, Arne Broering, Christoph Stasch, Thomas Everding, and Alejandro Llaves. A RESTful Proxy and Data Model for Linked Sensor Data. *International Journal of Digital Earth*, 2011.

[78] Krzysztof Janowicz, Christoph Stasch, Alejandro Llaves, Sven Schade, and Arne Broering. Are Virtual Sensors Sensors ? 2011.

[79] Sanem Kabadayi, Adam Pridgen, and Christine Julien. Virtual Sensors: Abstracting Data from Physical Sensors. In *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, pages 587—-592. IEEE Computer Society, 2006.

[80] Marinos Kavouras and Margarita Kokla. *Theories of geographic concepts: ontological approaches to semantic integration*. CRC Press, Boca Raton, 2007.

[81] Eva Klien and Florian Probst. Requirements for Geospatial Ontology Engineering. In F. Toppen and M. Painho, editors, *8th Conference on Geographic Information Science (AGILE 2005).*, pages 251–260, Estoril, Portugal, 2005. URL http://www.agile-online.org/Conference_Paper/CDs/agile_2005/papers/79_Eva%20Klien.pdf.

[82] Alexander Klippel, Michael Worboys, and Matt Duckham. Geographic event conceptualization. *Cognitive Processing*, 7(S1):52–54, July 2006. ISSN 1612-4782. doi: 10.1007/s10339-006-0062-x. URL http://www.springerlink.com/index/10.1007/s10339-006-0062-x.

[83] Dave Kolas and Robert Battle. GeoSPARQL User Guide. Technical report, InterOp project, 2012. URL http://ontolog.cim3.net/file/work/SOCoP/Educational/GeoSPARQL%20User%20Guide.docx.

[84] Clifford A. Kottman. The Open GIS Consortium and Progress Toward Interoperability in GIS. In Michael Goodchild, Max Egenhofer, Robin Fegeas, and Cliff Kottman, editors, *Interoperating Geographic Information Systems*, volume 495 of *The Springer International Series in Engineering and Computer Science*, pages 39–54. Springer US, 1999. ISBN 978-1-4613-7363-6. doi: 10.1007/978-1-4615-5189-8_4. URL http://dx.doi.org/10.1007/978-1-4615-5189-8_4.

[85] W. Kuhn. Geospatial semantics: why, of what, and how? *Journal on data semantics III*, 3534:1–24, 2005.

[86] Werner Kuhn. A Functional Ontology of Observation and Measurement. In Krzysztof Janowicz, Martin Raubal, and Sergei Levashkin, editors, *GeoSpatial Semantics*, volume 5892 of *Lecture Notes in Computer Science*, pages 26–43. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-10435-0. doi: 10.1007/978-3-642-10436-7_3. URL http://dx.doi.org/10.1007/978-3-642-10436-7_3.

[87] Werner Kuhn. Core concepts of spatial information for transdisciplinary research. *International Journal of Geographical Information Science*, 26(12):2267–2276, 2012.

[88] Carl Lagoze and Jane Hunter. The ABC Ontology and Model. *Journal of Digital Information*, 2(2), 2006. ISSN 1368-7506.

[89] Maurizio Lenzerini. Data integration: a theoretical perspective. In *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '02, pages 233–246, New York, NY, USA, 2002. ACM. ISBN 1-58113-507-6. doi: 10.1145/543613.543644. URL http://doi.acm.org/10.1145/543613.543644.

[90] Gérard Ligozat, Debasis Mitra, and Jean-François Condotta. Spatial and temporal reasoning: beyond Allen's calculus. *AI Communications*, 17(4):223–233, 2004.

[91] Alejandro Llaves. Integration of Geosensor Data by means of an Event Abstraction Layer. In Christian Kray and Angela Schwering, editors, *Doctoral Colloquium at Conference on Spatial Information Theory (COSIT'11)*, Belfast, Maine, US., 2011.

[92] Alejandro Llaves. A Theory for Event Processing of Geosensor Data. In Theodor Foerster, Arne Broering, Bastian Baranski, Benjamin Pross, Christoph Stasch, Thomas Everding, and Stephan Maes, editors, *Workshop on Integrating Sensor Web and Web-based Geoprocessing, 14th AGILE Conference. CEUR Workshop proceedings (Vol. 712)*, Utrecht, 2011. CEUR. URL http://ceur-ws.org/Vol-712/paper8.pdf.

[93] Alejandro Llaves and Thomas Everding. Discovering Geosensor Data by means of an Event Abstraction Layer. In Laura Diaz, Carlos Granell, and Joaquin Huerta, editors, *Discovery of Geospatial Resources: Methodologies, Technologies, and Emergent Applications*, chapter 6, pages 112–132. IGI Global, 2012. doi: 10.4018/978-1-4666-0945-7.ch006.

[94] Alejandro Llaves and Werner Kuhn. An Event Abstraction Layer for the Integration of Geosensor Data. Submitted to the International Journal of Geographic Information Science, Special Issue on Space-Time Research, 2013.

[95] Alejandro Llaves and Henry Michels. ENVISION Deliverable 4.7 - Event Processing of Observation Data. Technical report, ENVISION project, 2012. URL http://www.envision-project.eu/wp-content/uploads/2013/02/D4.7-1.0.pdf.

[96] Alejandro Llaves, Miha Grčar, Patrick Maué, Henry Michels, Dunja Mladenić, Alexandra Moraru, Maja Škrjanc, Matjaž Rihtar, and Marcell Roth. ENVISION Deliverable 4.1 - Ontology

requirements analysis. Technical report, ENVISION project, 2010. URL http://www.envision-project.eu/wp-content/uploads/2012/02/D4_1_Final.pdf.

[97] Alejandro Llaves, Henry Michels, Patrick Maué, and Marcell Roth. Semantic Event Processing in ENVISION. In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, pages 25:1–25:9, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-0915-8. doi: 10.1145/2254129.2254161. URL http://doi.acm.org/10.1145/2254129.2254161.

[98] D. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley Professional, 2002.

[99] David Luckham. What's the Difference between ESP and CEP?, 2006. URL http://www.complexevents.com/2006/08/01/what%E2%80%99s-the-difference-between-esp-and-cep/.

[100] David Luckham, Roy Schulte, Jeff Adkins, Pedro Bizarro, Albert Mavashev, and Peter Niblett. Event Processing Glossary - Version 2.0. Technical report, Event Processing Technical Society, 2011. URL http://www.complexevents.com/wp-content/uploads/2011/08/EPTS_Event_Processing_Glossary_v2.pdf.

[101] Claudio Masolo, Stefano Borgo, Aldo Gangemi, Nicola Guarino, Alessandro Oltramari, and Ian Horrocks. WonderWeb Deliverable D18 - Ontology Library (final). Technical report, WonderWeb project, 2003. URL http://wonderweb.man.ac.uk/deliverables/documents/D18.pdf.

[102] Patrick Maué, Alejandro Llaves, and Thore Fechner. Context-aware access to ontologies on the Web. In Mathieu D'Aquin, Alexander García Castro, Christoph Lange, and Kim Viljanen, editors, *Proceedings of Workshop on Ontology Workshops and Editors for the Semantic Web (ORES 2010)*, pages 83–94, Crete, Greece, 2010. CEUR. URL http://ceur-ws.org/Vol-596/paper-11.pdf.

[103] Brenda M. Michelson. Event-Driven Architecture Overview. Technical report, Patricia Seybold Group, 2006. URL http://www.omg.org/soa/Uploaded%20Docs/EDA/bda2-2-06cc.pdf.

[104] Harvey J Miller. The data avalanche is here. Shouldn't we be digging? *Journal of Regional Science*, 50(1):181–201, 2010. doi: 10.1111/j.1467-9787.2009.00641.x. URL http://dx.doi.org/10.1111/j.1467-9787.2009.00641.x.

[105] A.P.D. Mourelatos. Events, processes, and states. *Linguistics and philosophy*, 2(3):415–434, 1978.

[106] Arthur Na and Mark Priest. OGC Sensor Observation Service. Technical report, Open Geospatial Consortium, 2007. URL http://portal.opengeospatial.org/files/?artifact_id=26667.

[107] Silvia Nittel. A Survey of Geosensor Networks: Advances in Dynamic Environmental Monitoring. *Sensors*, 9(7):5664–5678, 2009. doi: 10.3390/s90705664. URL http://www.mdpi.com/1424-8220/9/7/5664.

[108] Harshal Patni. Real Time Semantic Analysis of Streaming Sensor Data. Master's thesis, Wright State University, Ohio, 2011.

[109] Harshal Patni, Cory Henson, and Amit Sheth. Linked sensor data. *2010 International Symposium on Collaborative Technologies and Systems*, pages 362–370, 2010. doi: 10.1109/CTS.2010.5478492. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5478492.

[110] Matthew Perry and John Herring. OGC GeoSPARQL - A Geographic Query Language for RDF Data. Technical report, Open Geospatial Consortium, 2012. URL https://portal.opengeospatial.org/files/?artifact_id=47664.

[111] Donna Peuquet and Niu Duan. An Event-based Spatiotemporal Data Model (ESTDM) for temporal analysis of geographical data. *International Journal of Geographical Information Systems*, 9 (1):7–24, 1995.

[112] Florian Probst. Ontological Analysis of Observations and Measurements. In M. Raubal, H. Miller, A. Frank, and M. Goodchild, editors, *Geographic Information Science, 4th International Conference, GIScience 2006*, pages 304 – 320. Springer, Number 4197 in LNCS, 2006.

[113] Femke Reitsma. *A New Geographic Process Data Model*. PhD thesis, University of Maryland., 2004.

[114] Bernd Resch, Manfred Mittlboeck, Fabien Girardin, Rex Britter, and Carlo Ratti. Live Geography - Embedded Sensing for Standardised Urban Environmental Monitoring. *International Journal on Advances in Systems and Measurements*, 2(2&3):156–167, 2009. URL http://www.berndresch.com/download/work/publications/resch_et_al_asm_urban_environmental_monitoring.pdf.

[115] Avinash Rude and Kate Beard. High-Level Event Detection in Spatially Distributed Time Series. In *Eighth International Conference, GIScience 2012, Ohio, Columbus, Sep. 2012, Proceedings LNCS.*, pages 160–172, 2012.

[116] Tuukka Ruotsalo and Eero Hyvönen. An Event-Based Approach for Semantic Metadata Interoperability. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, volume 4825 of *Lecture Notes in Computer Science*, pages 409–422. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-76297-3. doi: 10.1007/978-3-540-76298-0_30. URL http://dx.doi.org/10.1007/978-3-540-76298-0_30.

[117] Ansgar Scherp, Thomas Franz, Carsten Saathoff, and Steffen Staab. F–a model of events based on the foundational ontology dolce+DnS ultralight. In *Proceedings of the fifth international conference on Knowledge capture*, K-CAP '09, pages 137–144, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-658-8. doi: 10.1145/1597735.1597760. URL http://doi.acm.org/10.1145/1597735.1597760.

[118] Ryan Shaw, Raphaël Troncy, and Lynda Hardman. LODE: Linking Open Descriptions of Events. In Asunción Gómez-Pérez, Yong Yu, and Ying Ding, editors, *The Semantic Web*, volume 5926 of *Lecture Notes in Computer Science*, pages 153–167. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-10870-9. doi: 10.1007/978-3-642-10871-6_11. URL http://dx.doi.org/10.1007/978-3-642-10871-6_11.

[119] Amit Sheth, Cory Henson, and Satya S Sahoo. Semantic Sensor Web. *IEEE Internet Computing*, 12(4):78–83, 2008. ISSN 10897801. doi: 10.1109/MIC.2008.87. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4557983.

[120] Ingo Simonis. OGC Sensor Alert Service Candidate Implementation Specification. Technical report, Open Geospatial Consortium, 2006. URL http://portal.opengeospatial.org/files/?artifact_id=15588.

[121] Ingo Simonis and Johannes Echterhoff. OGC Draft Web Notification Service Implementation Specification. Technical report, Open Geospatial Consortium, 2006. URL http://portal.opengeospatial.org/files/?artifact_id=18776.

[122] Ingo Simonis and Andreas Wytzisk. Web Notification Service. Technical report, Open Geospatial Consortium, 2006. URL http://portal.opengeospatial.org/files/?artifact_id=1367.

[123] Christoph Stasch, Krzysztof Janowicz, Arne Broering, Ilka Reis, and Werner Kuhn. A Stimulus-Centric Algebraic Approach to Sensors and Observations. In Niki Trigoni,

Andrew Markham, and Sarfraz Nawaz, editors, *GeoSensor Networks. Third International Conference, GSN 2009, July 13-14, 2009 Proceedings*, volume 5659 of *Lecture Notes in Computer Science*, pages 169–179. Springer, 2009. URL http://ifgi.uni-muenster.de/%7Ejanowicz/downloads/Sensors_and_Observations_GSN2009.pdf.

[124] Christoph Stasch, Sven Schade, Alejandro Llaves, Krzysztof Janowicz, and Arne Broering. Aggregating Linked Sensor Data. In Kerry Taylor, Arun Ayyagari, and David De Roure, editors, *Proceedings of the 4th International Workshop on Semantic Sensor Networks 2011 (SSN11)*, pages 55–68. CEUR, 2011.

[125] Christoph Stasch, Richard Jones, Dan Cornford, Martin Kiesow, Matthew Williams, and Edzer Pebesma. Representing Uncertainties in the Sensor Web. In *Workshop Sensing A Changing World*, pages 1–7, 2012.

[126] Kerry Taylor and Lucas Leidinger. Ontology-driven complex event processing in heterogeneous sensor networks. In *Proceedings of the 8th extended semantic web conference on The semanic web: research and applications - Volume Part II*, ESWC'11, pages 285–299, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-21063-1. URL http://dl.acm.org/citation.cfm?id=2017936.2017959.

[127] Kia Teymourian and Adrian Paschke. Towards semantic event processing. In *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, DEBS '09, pages 29:1–29:2, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-665-6. doi: 10.1145/1619258.1619296. URL http://doi.acm.org/10.1145/1619258.1619296.

[128] Kia Teymourian and Adrian Paschke. Semantic Rule-Based Complex Event Processing. In Guido Governatori, John Hall, and Adrian Paschke, editors, *Rule Interchange and Applications*, volume 5858 of *Lecture Notes in Computer Science*, pages 82–92. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-04984-2. doi: 10.1007/978-3-642-04985-9_10. URL http://dx.doi.org/10.1007/978-3-642-04985-9_10.

[129] Kia Teymourian and Adrian Paschke. Enabling knowledge-based complex event processing. In *Proceedings of the 2010 EDBT/ICDT Workshops*, page 37. ACM, 2010.

[130] M. Hadi Valipour, Bavar AmirZafari, K. Niki Maleki, and Negin Daneshpour. A brief survey of software architecture concepts and service oriented architecture. In *Proceedings of 2nd IEEE*

International Conference on Computer Science and Information Technology, ICCSIT'09, pages 34–38, 2009. URL http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5235004.

[131] William Vambenepe, Steve Graham, and Peter Niblett. Web Services Topics 1.3. Technical report, OASIS Standard, 2006. URL http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf.

[132] Willem Robert Van Hage, Véronique Malaisé, Roxane Segers, Laura Hollink, and Guus Schreiber. Design and use of the Simple Event Model (SEM). *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(2):128–136, 2011. ISSN 1570-8268. doi: 10.1016/j.websem.2011.03.003. URL http://dx.doi.org/10.1016/j.websem.2011.03.003.

[133] Z. Vendler. *Linguistics in philosophy*. Cornell University Press Ithaca, 1967.

[134] S. Vinoski. Advanced Message Queuing Protocol. *Internet Computing, IEEE*, 10(6):87–89, 2006. ISSN 1089-7801. doi: 10.1109/MIC.2006.116.

[135] Michael Worboys. Event-oriented approaches to geographic phenomena. *International Journal of Geographical Information Science*, 19(1):1–28, January 2005. ISSN 1365-8816. doi: 10.1080/13658810412331280167. URL http://www.informaworld.com/openurl?genre=article&doi=10.1080/13658810412331280167&magic=crossref||D404A21C5BB053405B1A640AFFD44AE3.

[136] Michael Worboys and Matt Duckham. Monitoring qualitative spatiotemporal change for geosensor networks. *International Journal of Geographical Information Science*, 20(10):1087–1108, November 2006. ISSN 1365-8816. doi: 10.1080/13658810600852180. URL http://www.informaworld.com/openurl?genre=article&doi=10.1080/13658810600852180&magic=crossref||D404A21C5BB053405B1A640AFFD44AE3.

[137] Michael Worboys and Kathleen Hornsby. From objects to events: GEM, the geospatial event model. In M Egenhofer, C Freksa, and H Miller, editors, *Third International Conference on GIScience 2004*, pages 327–344, Berlin, 2004. SpringerVerlag.

[138] J. Yick, B. Mukherjee, and D. Ghosal. Wireless sensor network survey. *Computer networks*, 52(12):2292–2330, 2008.

[139] Qunzhi Zhou, Yogesh Simmhan, and Viktor Prasanna. SCEPter: Semantic complex event processing over end-to-end data flows.

Technical report, Technical Report 12-926, Computer Science Department, University of Southern California, 2012. URL http://www.cs.usc.edu/assets/001/82856.pdf.

VERSICHERUNG

Hiermit versichere ich, dass ich bisher noch keinen Promotionsversuch unternommen habe.

Castellón, September 2013

Alejandro Llaves Arellano

Hiermit versichere ich, dass ich die vorgelegte Dissertation selbst und ohne unerlaubte Hilfe angefertigt, alle in Anspruch genommenen Quellen und Hilfsmittel in der Dissertation angegeben habe und die Dissertation nicht bereits anderweitig als Prüfungsarbeit vorgelegen hat.

Castellón, September 2013

Alejandro Llaves Arellano

Hiermit erkläre ich, nicht wegen einer Straftat rechtskräftig verurteilt worden zu sein, zu der ich meine wissenschaftliche Qualifikation missbraucht habe.

Castellón, September 2013

Alejandro Llaves Arellano

LEBENSLAUF

Alejandro Llaves Arellano

Geboren am 21. März 1985 in Castellón de la Plana, Spanien.
Familienstand: Ledig.

Eltern:
José Luís Llaves Almagro
María Rosa Arellano Calvo

Schulbildung:	
Grundschule:	1989 - 1999, Colegio Hermanos Ochando, Almazora, Spanien.
Gymnasium:	1999 - 2003, IES Álvaro Falomir, Almazora, Spanien.
Hochschulreife (Abitur):	Mai 2003, Universitat Jaume I, Castellón.
Studium:	
Dipl.-Ing. Informatik	September 2003 - September 2009, Universitat Jaume I, Castellón.
Prüfungen:	
Dipl.-Ing. Informatik	30. September 2009, Universitat Jaume I, Castellón.
Promotionsstudium:	
Geoinformatik	November 2009 - September 2013, Westfälische Wilhelms-Universität Münster.

Tätigkeiten:

Studentische Hilfskraft	Februar 2007 - August 2008, Geospatial Technologies Research Group, Universitat Jaume I, Castellón.
Praktikum	Oktober 2008 - Mai 2009, con terra GmbH und 52°North Initiative for Geospatial Open Source Software GmbH, Münster.
Forschungsstipendiat	November 2009 - März 2010, Institut für Geoinformatik, Westfälische Wilhelms-Universität Münster.
Wissenschaftlicher Mitarbeiter	April 2010 - Juli 2013, Institut für Geoinformatik, Westfälische Wilhelms-Universität Münster.

Beginn der Dissertation:

November 2009, Institut für Geoinformatik.

Betreuer: Prof. Dr. Werner Kuhn.

Castellón, September 2013

_____

Unterschrift