Peer-reviewed version available at *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 385; doi:10.3390/ijgi7100385

Article

GeoSpatial IoT - the Need for an Event-Driven Spatial Data Infrastructure

Matthes Rieke [1,*,†], Lorenzo Bigagli [2,†], Stefan Herle [3,†], Simon Jirka [1,†], Alexander Kotsev [4,†‡], Thomas Liebig [5,†], Christian Malewski [6,†], Thomas Paschke [7,†] and Christoph Stasch [1,†]

1 52°North GmbH, Münster, Germany
2 National Research Council of Italy, Institute of Atmospheric Pollution Research, 50019 Sesto Fiorentino (Florence), Italy
3 RWTH Aachen University, Geodetic Institute and Chair for Computing in Civil Engineering & GIS, Aachen, Germany
4 European Commission, Joint Research Centre, Digital Economy Unit, Ispra 21027, Italy
5 TU Dortmund, Dortmund, 44221, Germany
6 Wupperverband, Wuppertal, 42289, Germany
7 Esri Deutschland GmbH, Kranzberg, Germany
* Correspondence: Matthes Rieke (m.rieke@52north.org; Tel.: +49-251-396371-51)
† All authors contributed equally to this work.
‡ The views expressed are purely those of the authors and may not in any circumstances be regarded as stating an official position of the European Commission.

Abstract: The nature of contemporary Spatial Data Infrastructures lies in the provision of geospatial information in an on-demand fashion. Though recent applications identified the need to react to real-time information in a time-critical way. In particular, research efforts in the field of geospatial Internet of Things have identified substantial gaps in this context, ranging from a lack of standardization for event-based architectures to the meaningful handling of real-time information as ”events”. This manuscript presents work in the field of Event-driven Spatial Data Infrastructures with a particular focus on sensor networks and the devices capturing in-situ measurements. The current landscape of Spatial Data Infrastructures is outlined and used as the basis for identifying existing gaps that retain certain geospatial applications from using real-time information. We present a selection of approaches - developed in different research projects - to overcome these gaps. Being designed for specific application domains, these approaches share commonalities as well as orthogonal solutions and can build the foundation of an overall Event-driven Spatial Data Infrastructure.

Keywords: event-driven architectures, asynchronous transactions, sensor web, spatio-temporal data, real-time data, stream processing, spatial data infrastructures, sensor networks

1. Introduction

Historically, the discoverability and accessibility to geospatial content was difficult, as data was not available online, was encoded through different proprietary formats and was often not documented through metadata [1]. These were among the main technical arguments for the establishment of spatial data infrastructures (SDIs) during the 1990s. Nowadays, the situation is completely different. The rapid emergence of technologies such as the Internet of Things (IoT), combined with the increased importance of the private sector and citizens as users and creators of content is generating unprecedented amounts of data. Not only are the heterogeneity and volumes of data rapidly increasing, but they are produced in near-real time conditions. This, combined, poses completely new challenges regarding the utilisation and processing of data. Geospatial data is no exception. In particular, addressing the (spatio-)temporal dimension requires (i) new analytical methods, (ii) innovative means for encoding and exchanging spatio-temporal data from constrained devices, and (iii) new standards for data interoperability. That is

why [2,3] highlight the need for a theoretical framework that would cater for a process-based approach towards data as a complement to the very well established foundations of contemporary GIScience.

In this manuscript, we elaborate on eventing in GIScience. Our objectives are twofold - on one hand we identify limitations and corresponding challenges of current Spatial Data Infrastructures for an adequate incorporation of time series, and on the other we elaborate on approaches on how to tackle them. The latter is illustrated through specific case studies.

Structurally, our work is divided into five sections. Following a brief introduction (Section 1) we illustrate the current state of 'eventing' in GIScience (Section 2). A description of architectural patterns and the challenges that remain is provided in (Section 3). Subsequently, we discuss different new approaches for Eventing which have been developed for specific application domains (Section 4). Finally, we conclude by summarising our findings and discuss open issues in (Section 5).

2. Eventing in GIScience: State-of-the-art

In this section we (i) provide an overview of the theoretical foundations of eventing in GIScience and share several definitions of selected frequently used terms, (ii) position eventing in the context of spatial data infrastructures, (iii) outline the most relevant standardisation initiatives, and (iv) describe relevant software products.

2.1. Spatial Data Types

During the last years, the availability and granularity of geospatial data has been rapidly growing. Additionally, spatio-temporal data streams provide multiple opportunities in fields such as smart cities, environmental risk assessment, precision farming and disaster response. All of those application domains require algorithms that process massive amounts of spatio-temporal data in near real-time. Historically, spatio-temporal data comes in a variety of forms and representations, depending on the domain, the observed phenomenon and the acquisition approach. Three types of spatio-temporal data streams can be outlined [4,5], namely spatial time series, events, and trajectories.

- A spatial time series consists of tuples (attribute, object, time, location)
- An event is triggered from a spatial time series under certain conditions and contains the tuples verifying these conditions
- A trajectory is a spatial time series for a particular moving object. It contains the location per time and is a series of tuples.

The automated detection of events in spatio-temporal data streams is a well-established research area (see e.g. [6] for an overview). Depending on the application, the event detection can analyse single trajectories (e.g. an aircraft), spatio-temporal measurement time series, or heterogeneous data streams. The examples below highlight the capabilities of each approach:

- Spatio-temporal measurements: A spatio-temporal value can focus on a specific spatially limited observation (e.g. water gauge measurements) or span regions (e.g. traffic flow, air pollution or noise). Sudden changes of such values can indicate anomalies but might also be caused by a failure of measurement equipment. Distinguishing between these two cases is a crucial capability required for spatio-temporal measurement processing
- Individual mobility and group movement: both fields are related in terms of acquisition methods (e.g. surveillance cameras) and make use of pattern or sequence recognition to identify certain events (e.g. detection of unusual behavior [7], hazardous pedestrian densities [8])
- Heterogeneous sensor networks: in the past years the design of sensor stations progressed more and more to integrated small-scale solutions (sensing IoT devices, smart sensors [9]). Therefore, maintainers of established sensor networks look into ways of integrating these into their existing architecture. Applications built for these sensor networks need to deal with the heterogeneity of the different observation devices.

2.2. Spatial Data Infrastructures

Spatial Data Infrastructures (SDIs) built since the term started to be used, in the early 1990s, have brought multiple technological, organisational and legal novelties that have increased the availability of spatial data for online reuse. The resulting benefits have been recognized, for example, in Europe, where a law obliges public sector providers of environmental data to comply with the requirements of the Infrastructure for Spatial Information in Europe (INSPIRE) Directive [10].

In accordance with INSPIRE, data providers shall (i) create metadata, (ii) establish network services, and (iii) harmonise their dataset in accordance with commonly agreed data specifications. The aforementioned network services are to be put in place in order to enable (i) discoverability of data through their metadata, (ii) interactive web-based viewing, (iii) download, and (iv) transformation from non-harmonised to harmonised.

As a classic SDI, INSPIRE has been conceived as a Service-Oriented Architecture (SOA), based on the Client-Server model and on the request/reply Message Exchange Pattern (MEP). Meanwhile, the development of modern applications and advanced distributed systems, such as peer to peer networks, have increasingly highlighted the shortcomings of the request/reply pattern, typical of SOA and classic SDIs.

Download services are particularly interesting with these regards, as they effectively provide access to the spatial (and spatio-temporal) data made available within the SDI. On a conceptual level, download services are typically defined to follow the request/reply communication model, where a client makes a request and the server responds synchronously, with either the requested information or a failure. This provides relatively immediate feedback, but can be insufficient in cases where the client is waiting for a specific event (such as data arrival, server changes, or data updates).

Shortly after SOA, another architectural model has emerged, named Event-Driven Architecture (EDA), which emphasizes the ability to monitor and react to events. There is abundant literature on the relationship between SOA and EDA, whose combination is sometimes referred to as Event-driven SOA, or also SOA 2.0 [11,12]. In synthesis, the two approaches can be seen as complementary to each other, the main difference being the reactive nature of SOA request/reply pattern, as compared to the proactive nature of EDA, characterized by the publish/subscribe interaction, described in the following section. Within this context, a recent activity [13] investigates the use of the OGC SensorThings API as an INSPIRE download service.

2.2.1. The publish/subscribe pattern

The publish/subscribe MEP is distinguished from the request/reply MEP by the asynchronous delivery of messages between a server and a client and the ability for the client (i.e. the Subscriber) to specify an ongoing (persistent) expression of interest. Note that such expression of interest requires the server (i.e. the Publisher) to maintain internal state about subscribing entities and therefore the fundamental publish/subscribe interaction is stateful.

The publish/subscribe MEP can be useful to reduce the latency between event occurrence and event notification, as it is the Publisher's responsibility to publish a message when the event occurs, rather than relying on clients to anticipate the occurrence. The publish/subscribe MEP can also be used to decouple message production from message consumption by allowing messages to be passed through a third party (a message broker). This allows entities in a system to be changed (with respect to implementation, network location, etc.), as long as the agreed-upon message format and publication location remain the same.

Two primary parties characterise the publish/subscribe interaction style: a Publisher, which is publishing information, and a Subscriber, which expresses an interest in all or part of the published information. Publishers and Subscribers are fully decoupled in time, space, and synchronization [14].

Two other actors involved are the actual entity to which data is to be delivered, i.e. the Receiver, and the entity actually delivering the data, i.e. the Sender. In many cases, the Subscriber coincides with the Receiver. However, the two roles may be distinguished, for example in a use case where a system

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

manager, acting as a Subscriber, sets up the information flows from Publishers to a number of other system entities that act as Receivers. Similarly, the Publisher and Sender roles are often implemented by the same entity, but may be segregated. For example, the Sender role may be implemented by a chain of entities, possibly unaware of the ultimate recipient of their messages and of the overall architecture of the system into which they inject messages.

A publish/subscribe model typically includes several functional components, which are variously implemented by the existing technologies (e.g. WS-Notification, WS-Eventing, ATOM):

- Filtering semantics - selecting desired information.
- Subscription semantics - duration of subscription, renewal of subscription, cancellation.
- Publication semantics - what, when, where, who of the published information.
- Wire/delivery protocol - reliable delivery, link failure detection, at-most-once or at-least-once delivery, firewall solution.
- Security - transport encryption, authentication, policy enforcement, etc.
- Capabilities advertisement - i.e., what PubSub capabilities a service offers.

2.3. Relevant standards

2.3.1. Open Geospatial Consortium

As illustrated in Figure 1, the Open Geospatial Consortium (OGC) has conducted much work, in the past, on event-driven models and architectures (see also [15,16]).

In the years 2006-2011, several lines of activities eventually consolidated into a revision of the Sensor Planning Service (SPS 2.0), an interface for querying a sensor about its capabilities and tasking it, and a version-aligned Standard Service Model (SWES 2.0) for the OGC Sensor Web Enablement initiative.

Both standards contained provisions for publish/subscribe interactions, which were increasingly seen as a generally useful enabler for the OGC standard baseline, even beyond the Sensor Web realm. In 2010, a specific PubSub Working Group was started, to define a self standing OGC standard for publish/subscribe, introduced in the next section.

OGC PubSub

The OGC Publish/Subscribe Interface Standard (in short, PubSub) describes a mechanism to support publish/subscribe requirements across OGC data types, such as coverages, features, and observations, and service interfaces, such as Sensor Observation Service (SOS) and Catalogue Service for the Web (CSW). The standard, officially approved in February 2016, is a long-awaited building block in the OGC suite of geospatial standards, and a fundamental enabler towards an Event-driven SDI. It has been successfully applied to several application domains, including the Sensor Web and Aviation [17].

The rationale for PubSub can be summarized as follows:

- OGC service clients are interested in subscribing to information already provided by OGC services.
- OGC services have developed relatively mature and specialized request/reply mechanisms for accessing and filtering the information provided (e.g. the Filter Specification for general-purpose filtering).
- Such filtering semantics can be used for implementing publish/subscribe use-cases and need not be re-invented or generalized.

PubSub consists of two parts: (i) a Core document [18] that abstractly describes the basic mandatory functionalities and several optional extensions, independently of the underlying binding technology; (ii) and a SOAP binding document [19] that defines the implementation of PubSub in SOAP services, based on the OASIS Web Services Notification (WS-N) set of standards [20].

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

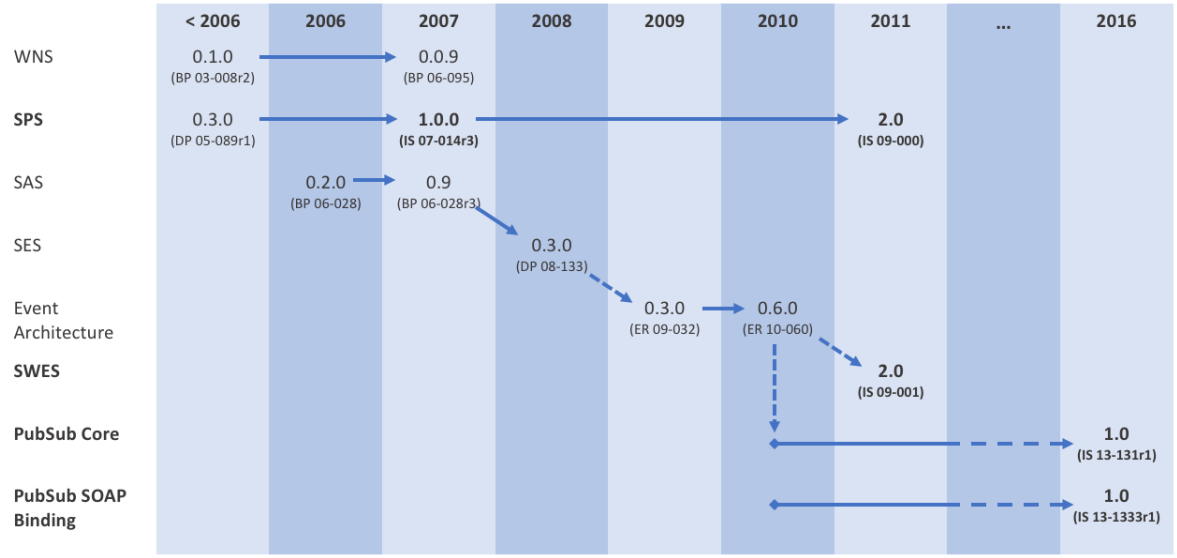

Figure 1. Overview and timeline of relevant OGC eventing-related work in the last decade. The rows show the roadmap of various documents; milestones are indicated with their version and their OGC reference number and type (Best Practice, Discussion Paper, Implementation Standard, Engineering Report); bold fonts highlight official OGC standards. See at the end of the manuscript for a complete list of abbreviations.

The scope of the OGC PubSub Standard Working Group also includes a RESTful binding document (currently in draft status) that will specify how to realize the PubSub functionality in REST/JSON services. Additional extensions have been proposed in the framework of the OGC Testbed-12 initiative: a specific PubSub 1.0 extensions for CSW 2.0.2 and 3.0, leveraging on standard functionalities, data models, and semantics to enable sending notifications based on user-specified area of interest and/or keywords; and a basic mechanism to PubSub-enable any generic OGC Web Service leveraging on its existing request/reply interaction, i.e. usual requests as filters, usual responses as appropriate updates/data pushes, existing semantics and syntax expressiveness [21].

Web Processing Service

The OGC Web Processing Service (WPS) standard is based on the synchronous request/reply model and implements the classical Remote Procedure Call communication paradygm. To mitigate sychronization coupling between clients and servers and avoid blocking the caller thread while waiting for the reply of a remote invocation, the standard allows the client to request a handle through which the actual return value can be accessed in the future, when needed [14].

Orchestration and automated processing has been achieved with WPS-based service architectures [22,23]. Further work has been done on event-driven architectures based on WPS workflows. An extension to the WPS, called the Web Event Processing Service (WEPS) has been developed to allow time series event pattern matching [24]. This architecture is based on classic request/response *Execute* requests to manage user subscriptions to specific events.

2.3.2. IoT Protocols

Events like those defined in section 2.1 are often emitted by resource-restricted devices in IoT settings. In literature, plenty discussion papers can be found about the optimal IoT application protocol to send and collect the related messages (e.g. [25]). In fact, every protocol has its advantages and disadvantages.

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

HTTP

HTTP[1] is well known and simple but because of its request/response model, it cannot be used to emit events and initiate data streams. Also, it has a large overhead and cannot implement a 1-to-n communication. Protocols which use a push-mechanism like a publish/subscribe MEP seem more suitable for event dissemination by IoT devices.

CoAP

The Constrained Application Protocol (CoAP)[2] is based on UDP and implements also the request/response-pattern. It is often termed as "HTTP for UDP" since it has a similar method set compared to HTTP and REST-APIs can be realized. However, it also includes an "observe" feature which can be used to accomplish a server-based push-mechanism. It is very efficient and aims for Wireless Sensor Networks. But since it is based on UDP, it may be problematic in IoT applications.

XMPP

The Extensible Messaging and Presence Protocol (XMPP)[3] is based on XML and implements push-based communication mechanisms. It can be extended by so-called XEPs, some of them are also defined specifically for IoT purposes. Because it is famous for chat clients, it has been implemented in a variety of programming languages with different features. A major drawback is the large overhead which is created by the use of XML encodings. Therefore, it is barely used in IoT settings.

MQTT

The Message Queuing Telemetry Transport (MQTT) [26] is a standardised messaging protocol implementing the publish-subscribe MEP. It is designed to have a minimal overhead for message payloads thus allowing the application in low-bandwidth networks as well as embedded devices. These characteristics make it a perfect fit for IoT applications, still the usage of MQTT has grown beyond.

The core concept of MQTT is the topic. Clients can subscribe to topics in order to receive corresponding messages. Topics can represent a hierarchy which allow subscribers to subset data stream. E.g. a topic such as *sensor-platform-1/NOx* would publish NOx measurements of *sensor-platform-1*. A subscriber may register for the topic *sensor-platform-1/+*, which has a wildcard for the second hierarchy level, and, thus, not solely receives NOx messages but also other phenomenon measurements.

Many software projects have been established for MQTT, ranging from broker implementations to client libraries. Besides the TCP protocol, many brokers support WebSockets to enable seamless integration into (web) user interfaces.

GeoMQTT

GeoMQTT extends the MQTT protocol by spatio-temporal filtering capabilities [27]. While the original protocol tags every message with a topic name and matches the corresponding interests, the spatio-temporal extension also adds a geometry and a time stamp or time period to each published event. Given these three dimensions, the protocol can be used by clients to publish spatio-temporal events or processes. Subscribers may use additional spatial and temporal filters to register their spatio-temporal interests with the message broker, which validates the filters against the incoming events and notifies the corresponding clients. The spatial filter consists of a geometry defined by

---

1 https://tools.ietf.org/html/rfc2616/
2 https://tools.ietf.org/html/rfc7252/
3 https://tools.ietf.org/html/rfc6120/

common encoding standards and a spatial relation according to the DE-9IM model [28]. The temporal filter uses time intervals and a temporal relation which is defined by Allen's interval algebra for interval-interval relations [29] or Vilain's interval-point relations [30].

WebSub

Formerly known as PubSubhubbub [31], WebSub is an open protocol for publish/subscribe communication. It is well-established among web media and related content providers. One of its prominent use cases is the provision of new content from providers (e.g. blogs such as Wordpress or Medium.com) to news feed service providers (e.g. Feedly or Flipboard).

WebSub builds upon the publish/subscribe model and specifically introduces a third party, the Hub. The Hub interacts as an intermediate layer between Publisher and Subscribers. Compared to other publish/subscribe technologies, WebSub focusses on the information that new content or data is available without necessarily distributing the actual content. This is achieved by the concept of the "Light ping": the Publisher informs the Hub about content availability, and the Hub subsequently informs the Subscribers. Besides the content location, the "Light ping" only features a small set of information (e.g. title, abstract) and Subscribers may decide to fetch the full data. WebSub is a lightweight protocol with low bandwidth, but implies that the actual data is available via the HTTP protocol on the providers side.

2.4. Relevant IoT Event Processing Software

2.4.1. Open Source solutions

The increasingly deployed location-aware IoT devices result in an exponentially growing amount of spatio-temporal events and streams. The desire to process these data streams in a real-time manner and to finally support decision-making processes drives the development of suitable tools and algorithms for event processing.

Event processing divides into Complex Event Processing (CEP) and Distributed Stream Processing (DSP). Both can be applied to spatio-temporal data streams. CEP solutions analyse concurrent events and emit derived new events based on their combination. The complex part of such solutions represents fusing and synthesizing events from multiple sources, while the event processing refers to checking the joint events against query patterns. Traditionally, CEP frameworks are implemented with centralized architectures and provide no scalability. DSP focuses on the high volume and velocity of data streams. These tools are scalable and have different requirements than CEP systems. One example is the Kappa architecture described in Kreps [32].

Esper

Esper[4] is a multi-lingual (Java and .NET) CEP framework. It works on streams of data and allows to apply aggregate functions as well as pattern matching, also across multiple data streams. By offering an SQL-like Domain Specific Language, the Event Processing Language (EPL), Esper provides a transparent and easy way of defining these functions and patterns. The EPL allows to apply basic statistical functions such as minimum/maximum, sums and counts or averages. Filtering data can be achieved by applying basic comparisons (e.g. greater than) on the stream data. It supports different types of data windows (time-based, count-based). While Esper is an Open Source project, an Enterprise edition is available which builds upon established Big Data frameworks such as Apache Kafka (see below) to support horizontal scaling and fault-tolerance mechanisms.

[4] http://www.espertech.com/esper/

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

In the context of Eventing in GIScience, Esper can be understood as a framework that is integrated under the hood of specific components, providing essential business logic which is required to support different use cases such as the analysis of time series data stream.

Apache Storm

Apache Storm[5] is a real-time stream processor written in Java and belongs to the group of DSP systems. Apache Storm is used in the research to analyse spatio-temporal data streams. Zhang *et al.* [33] utilise Apache Strom to implement a distributed spatial index for moving objects and, finally, provide real-time traffic decision support. Mattheis *et al.* [34] build a scalable map matching system with Apache Storm.

Storm uses topologies as an application abstraction to model units of computation. A topology forms a directed acyclic graph with different nodes which process the data while the data stream advances in the graph. These nodes can either be input units for data tuples, the spouts, or processing units, the bolts. The flow of the data tuples in the graph and therefore between different physical machines is controlled by different grouping types. For instance, the *field* type groups according to a key in the data tuples. The communication and coordination of the distributed processing is achieved by a manager nodes, the Nimbus. Storm implements a backup and acknowledgement mechanism. It ensures that each data tuple is either processed once or if a failure occurs, it is reprocessed by the topology.

Besides Apache Storm, other DSP solutions are based e.g. on Apache Flink[6] or Apache Spark Streaming[7]. A performance comparison between these different frameworks can be found e.g. in Lopez *et al.* [35]. For all these solutions, the data stream provisioning and delivery plays an important role, which can be accomplished by Apache Kafka.

Apache Kafka

Like *Storm* and *Spark*, *Kafka*[8] is a top-level software project of the Apache Software Foundation. It is designed to be a low-latency stream-processing framework in the Java ecosystem. In settings with distributed computing workers, Kafka has been established as a message broker that operates as a publish/subscribe service and can be used as multi-node load balancer. Events in Kafka are called *messages*. The software abstracts message streams into *topics* which operate as buffers or queues. Each topic is split into a set of partitions. Kafka is designed to work in a cluster environment. By replicating the partitions of a topic to multiple cluster nodes, Kafka provides not only a built-in failover system but also a highly effective architecture to apply distributed stream processing.

Kafka provides a set of APIs which allow developers to plug-in to the platform:

- Consumer API: allows a component to listen to specific topics of interest
- Streams API: intermediate components can apply processes and algorithms on the message stream and provide the result back to Kafka on a dedicated topic
- Connectors: used to integrate components (input or output) that are not part of the platform itself (e.g. databases)

Kafka is a key pillar in Java-based Big Data and IoT architectures (e.g. [36]). It is able to integrate with Apache Hadoop[9] and other related Apache frameworks such as Storm and Spark (see above).

---

5 http://storm.apache.org/
6 https://flink.apache.org/
7 https://spark.apache.org/streaming/
8 https://kafka.apache.org/
9 http://hadoop.apache.org/

2.4.2. Proprietary Solutions

Esri introduced an event based processing framework written in Java back in 2012, which is nowadays called ArcGIS GeoEvent Server. The GeoEvent Server uses different input and output connectors to ingest data streams from a variety of sources and formats, process them in a common event format and distribute the events to different systems inside and outside of Esri's Web GIS platform solution [37].

From the 2018 10.6 release on, the GeoEvent Server started to utilize Apache Kafka as an underlying messaging framework, replacing RabbitMQ that was used in former versions. This allows to harvest the benefits of Apache Kafka, as described above, in order to scale horizontally and run multiple GeoEvent Server in a multi-machine site for parallel event processing [38].

Esri is also using Apache Spark as a framework for distributed processing of data in their ArcGIS GeoAnalytics Server solution (see Figure 2) to analyze large amounts of spatio-temporal data [39]. It will also be used in a new, highly scalable Software as a Service solution that will provide comprehensive geo-spatial analytic options for the IoT world as part of the ArcGIS Online platform. Here a combination of Apache Kafka and Apache Spark is used for both streaming and batch analytics [40][37].

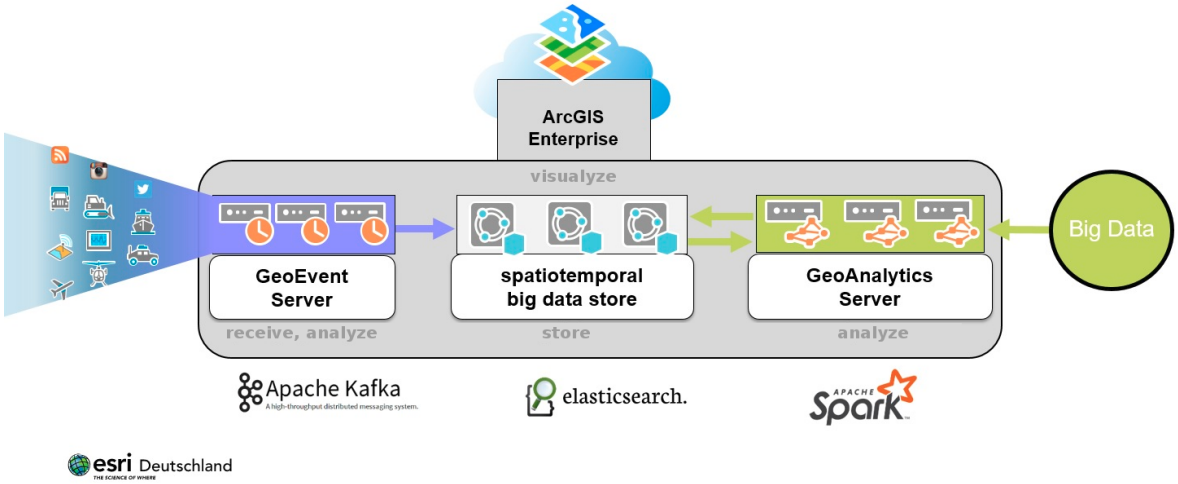

Figure 2. Overview of ESRI spatio-temporal data streaming technology.

Other big players of the IT industry have established messaging systems which enable stream-based processing within their cloud architectures. In 2015 Google introduced *Cloud Pub/Sub* which allows the scalable and performant processing of data streams with a special focus on the requirements of smart devices and IoT applications (high diversity, high throughput). It follows the publish/subscribe pattern (see section 2.2.1) and integrates natively with the other components available in the Cloud platform as well as external components via well-defined APIs. A similar approach was taken by Amazon with the introduction of *AWS IoT Core*, with a more dedicated focus on the integration aspects required for IoT and other smart devices.

3. Gaps in Event-driven SDI

The previous sections have introduced on the one hand the state of the art of spatial data infrastructure concepts and technologies and on the other hand important technological developments to enable event-driven information flows. However, there are still many open research challenges which need to be addressed before enhancing current spatial data infrastructure with event-driven functionalities. These challenges form the motivation for our work and will be introduced in the following subsections.

3.1. Defining and Determining Events

There are many definitions of the term "event" available, and the existing definitions of differ. From a conceptual perspective, we described the characteristics of events in Section 2.1. In addition and from a broader perspective once can rely on the definition that an "event is a notable thing that happens inside or outside your business" [11]. At the same time, this definition opens up a series of further questions regarding the technical definition of events in practical applications.

First of all, it is necessary to define what a "notable thing" is. For the purpose of observation data an event would be a notable observation within an incoming observation data set/stream. This can for example be characterised by a certain range or pattern of observation values. A common example would be the exceedance of threshold values. Other typical events might be properties such as the rate of change (in case of scalar observations), the occurrence of specific values (especially in case of categorical or Boolean observations), the presence of mobile objects within defined areas (geofencing), or the absence of new observation values within a given period of time (e.g. as an indicator for sensor failure).

As a result of the broad range of possible event types, users need a mechanism to define the events they are specifically interested in. Although there have been different previous approaches such the OGC Event Pattern Markup Language (EML) Discussion Paper [41] or the Event Processing Language (EPL) of the Esper framework (see Section 2.4.1), a common interoperable definition language suited to the needs of geospatial observation data streams is still missing.

Another aspect in the context of event definition is the distinction between event filtering and event processing. In the following we will use the first term for the process of delivering only those observations from a data stream that fulfil certain user-defined criteria (e.g. only forwarding those measurements above a given threshold). The second term, event processing, will refer to the generation of new, higher level information from an incoming observation data stream. For example, certain patterns within a data stream may indicate an alter situation. In this case, not the individual measurements themselves are delivered but the higher-level information that an alert has occurred. While event filtering is already supported by PubSub, the generation of new, higher level information is not yet covered by this specification.

3.2. Handling Large Amounts of Observation Data

Another challenge is the handling of large amounts of observation data. Depending on the application scenarios, data from very large sensor networks with potentially very high temporal resolution needs to be analysed. For dealing with such large amounts of data, there are different solutions available (e.g. the Big Data solutions of Esri (see section 2.4.2) or the Technical University of Dortmund (see section 4.3)). However, this does not yet resolve all issues that are specific to geospatial observation networks. For example, due to constraints in the data transmission capacity of wireless sensor networks, a distributed detection of events across many nodes may be needed before centralised data processing tools may become applicable.

Another challenge lies in the heterogeneous nature of input data for event detection. For example many applications require the fusion of data streams from different sources (e.g. weather data and hydrometric measurements in order to generate flood warnings). In this case it may become necessary to work with large data sets that have different temporal and/or spatial resolutions so that further pre-processing steps become necessary.

In summary these different aspects require approaches how to determine the performance measures that need to be fulfilled by a potentially distributed observation network and event processing infrastructure. Relevant parameters for this question may comprise the number of distributed observation data sources, data transmission constraints, the amount of incoming data (e.g. temporal and spatial resolution), but also the number of potential users and the event detection tasks they may submit.

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

3.3. How to Apply Event-driven Data Flows into Spatial Data Infrastructures

Conventional spatial data infrastructures do not provide means for receiving up-to-date data without delay and without manual interaction (see Section 2.2). Typically, such infrastructures are based on pull-based communication patterns. Thus, an approach is needed how to enhance such existing infrastructures without breaking established architectures and communication flows.

One important aspect in this regard is the event-enablement of existing data sources such as sensors, databases, and Web services. While for each of these elements event-driven extensions from mainstream IT are available (see Section 2.4), the integration of these technologies into extended spatial data infrastructures is still an open work item. In this regard the integration of geospatial IT standards with emerging mainstream information technologies will be a special challenge, as many current spatial data infrastructure concepts were developed independently from mainstream approaches beyond the geospatial domain. A central element within spatial data infrastructures are catalogues that allow the discovery of geospatial data sets as well as Web services. These catalogues, however, do not yet explicitly support the discovery of data streams or services for managing event subscriptions. To close this gap, the development of dedicated metadata profiles will be necessary.

Finally, the consideration of event-driven data flows opens up a new dimension of data analysis and processing workflows within spatial data infrastructures. Besides simple ideas such as triggering certain workflows in case of a specific event, new opportunities for chaining and orchestrating fully event-driven, real-time processes become available.

3.4. Security and Reliability

In conventional spatial data infrastructures, the data transfer can be easily verified on the client side. If a client issues a pull-request, the data flow is complete as soon as the client has received the complete response. Thus, there is no significant risk of losing messages. In event-driven infrastructures, however, the client has no a-priori knowledge when to expect an incoming message. For example, if an event is detected in a data stream, a notification is sent to a subscriber. However, the subscriber only know about the event, after the notification has arrived. If the notification is lost, the users would not know about this. Thus, the ability to guarantee the delivery of notifications about events through the whole processing workflow is essential for a multitude of use cases (e.g. emergency response, prevention).

Furthermore, security aspects (e.g. for managing subscriptions and for ensuring that only authorised users receive notifications) need to be considered when enhancing spatial data infrastructures with event-driven data flows.

3.5. Meaningful events

Also the semantics of event notifications are an important aspect that needs to be considered. For example, a formal description of the meaning of a specific notification (e.g. what does “alert” or “threshold exceeded” mean) would increase interoperability between different sources of event streams. Another important aspect is the provenance of event notifications. Usually, users would not only be interested in the notification itself (e.g. an alert message”) but also in the underlying input data that have triggered the notification (e.g. the observed water level and weather observations that have triggered a flood warning). Thus, mechanisms are needed for providing the relevant underlying input data that have caused the transmission of a notification.

3.6. Integrating Event-driven Data Flows into Applications

While the underlying event processing infrastructures may reach a high level of complexity, the usability of that infrastructure for users with potentially no technological background knowledge is of high importance. Thus, we consider a need for further research on how to enable users to define the events they are interested in or to manage event processing and notification workflows. At the same

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

time, also the end-user dissemination of data and events needs further investigation. For example, typical portrayal services such as the OGC Web Map Service are designed for delivering rather static map views. Thus, further work is necessary to develop ideas how dynamic event streams can be integrated into data visualisation components of applications.

4. Application domains

As the requirements and therefore also solutions for provisioning and processing of near-real time data vary in the different domains of GIScience, we describe the developed approaches within four important application domains: environmental monitoring, risk monitoring, disaster response and logistics. Still, eventing is not limited to these domains, but most of the solutions can be transferred to others with minimal effort.

The following sections present solutions which have been developed in the scopes of different research projects. We therefore start with introducing the overall use cases, followed by the importance of real-time data and event processing. The identification of domain specific gaps conclude the individual sections.

4.1. Environmental monitoring

Providing meaningful real time information becomes a more important pillar in several areas of environmental monitoring. Ranging from air quality threshold monitoring to the detection of oil spills, the time-critical interpretation of one or more phenomena is the main task of event architectures in this application domain. This section provides an overview on architectures developed in a set of research projects which all build upon the same design principal.

4.1.1. Architectural Approach

For an event-driven architecture in the environmental monitoring domain the Sensor Web concepts play an important role. The Sensor Observation Service (SOS) is one central component, providing capabilities to manage and provide time series for in-situ observations and measurements. One project building an architecture around this is AirSensEUR [42]. It's goal was to design a low-cost open hardware IoT device (a 'sensor shield', see Figure 3) with a Linux-based software platform for air quality monitoring. The overall objective of the platform is to simultaneously meet the legal requirements of two European Union Directives - the Air Quality Directive with regards to the quality of observation data, and the INSPIRE Directive [10] with regards to data download. The platform targets to measure the quality of ambient air (CO, O_3, NO and NO_2) at low concentration levels. This is achieved through a selection of sensors and a transparent and open data-treatment process. The latest version of the device is extended to measure Radon, CO_2 and particulate matter.

The transfer of data from the sensor shields is handled by the transactional profile of the 52 °North SOS implementation. Data are pushed on a regular interval to the SOS server. Subsequently, a post-processing through the use of an artificial neural network is applied to ensure a high level of measuring quality. The 52 °North Helgoland web client, as well as a smartphone app developed within the MYGEOSS project [43] consume data from the SOS. The smartphone app provides a notification service based on (i) exceedance of air quality thresholds as defined in the EU Air Quality Directive, and (ii) spatio-temporal vicinity of the users to a given sensor shield. These notifications are derived from the time series data through interaction with the 52 °North timeseries API.

Also in the ocean sciences community, several projects within the European Horizon 2020 framework address the need for event-driven data dissemination approaches with regards to environmental observations. The NeXOS project [44] aimed at the development of new sensing technologies in combination with an interoperable Web-based data exchange infrastructure. Many sensors deployed in the NeXOS infrastructure deliver a significant amount of in-situ observation data. While the archiving of these data streams is one important scientific requirement, the detection of relevant events within these streams is another aspect. For example, a hydrophone is capable of

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 August 2018 doi:10.20944/preprints201808.0132.v1

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

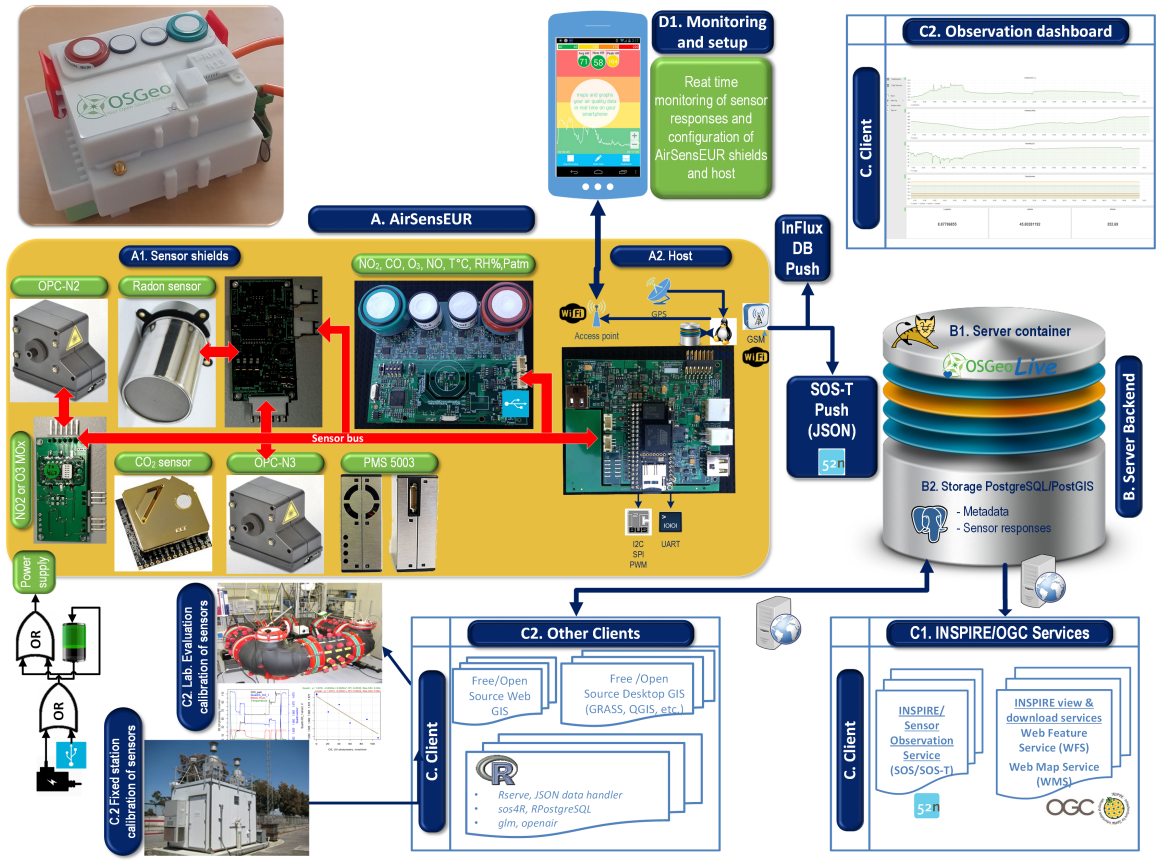

Figure 3. The AirSensEUR platform. Source:[42] (modified)

performing acoustic underwater measurements. These measurements can subsequently be used for detecting marine mammals which are recognisable as certain specific patterns within the acoustic data. An event-driven dissemination of such detection events, will be used for notifying scientists as soon as interesting, new data is available.

A second aspect that is relevant in the oceans research community is the event-driven dissemination of observation data collected by autonomous sensor platforms such as gliders. Often such platforms operate in very remote areas of the world so that the transmission of data can become a cumbersome task - a situation which is well-known in many applications of IoT devices. In the case of remote devices acting in the open oceans, cellular networks are not available and data has be transferred via satellite links. Thus, it is desirable that scientists are provided with an approach to request the automatic and immediate delivery of data about certain user-defined events (e.g. observation of properties beyond a given threshold or specific (critical) operating status data of the platform) while the large volume of all measured data is recovered later on when the platform has reached the shore again. The BRIDGES project [45] aims (among other goals) at a system architecture to address such use cases (see Figure 4).

In comparison to the above, the NoiseCap[10] citizen science experiment developed a customized architectural approach. Based on the outcomes of the Energic-OD project [46], NoiseCap focusses on determining the reliability of commercial cell phones in estimating indoor noise pollution. The system builds on the free and open-source Noise-Planet[11] scientific toolset for environmental noise assessment,

[10] http://essi-lab.eu/noisecap
[11] http://noise-planet.org/

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 August 2018 doi:10.20944/preprints201808.0132.v1

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

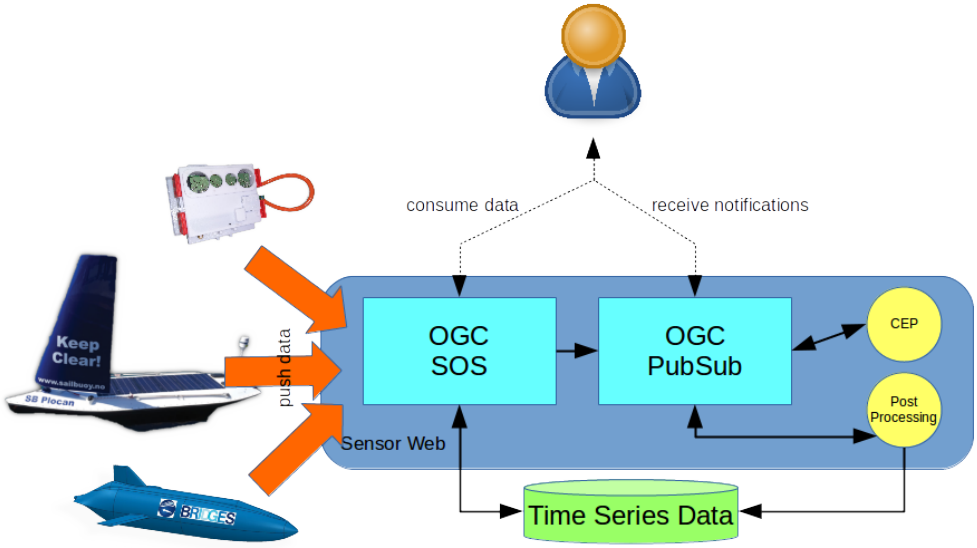

Figure 4. Sensor Web Architecture for Environmental Monitoring.

which includes the NoiseCapture Android application, developed within Energic-OD and allowing users to measure and share their noise environment. Each noise measurement is annotated with its location and can be displayed in interactive noise maps, within the application and on the Noise-Planet portal. NoiseCap focuses on air traffic noise, which is quite well-characterized and identifiable in indoor settings. Besides, this choice improves the reproducibility of the results. Hence, the experiment targets the specific group of citizens living nearby the airport of Florence, Italy.

User participation in NoiseCap is on a completely voluntary basis. In particular, participating citizens are free to choose whether to measure a given landing event or not. Since the measurement protocol requires some preparation (e.g. turning off the television and such), the researchers overseeing the experiment (for an illustration of the activities see Figure 5) have quickly realized the importance of notifying each user with a personalized message, at least a few minutes before the passage of an aircraft over her/his measurement spot (specified by the user during registration). Such "notification service" has been implemented leveraging ATC ADS-B Mode S data. An ad-hoc procedure analyses the Mode S records to identify the aircraft approaching the airport of Florence, whose trajectories are then tracked. When a trajectory matches a specified landing pattern, the service notifies the subscribers with a personalised message providing an estimated time of the overflight.

4.1.2. Lessons Learned

The AirSensEUR project implements an eventing layer in a non-standardised manner while the NeXOS and BRIDGES projects propose an approach based on PubSub [17]. All three initiatives have identified gaps which need to be solved before a stable deployment is possible. This comprises on the one hand the need for more lightweight interfaces (i.e. REST/JSON) instead of XML - due to the remote locations and transmission limitations - and on the other hand a common approach that allows users to define the events they are interested in (i.e. event filters). Furthermore, a need for user-friendly applications to manage event subscriptions and an interoperable approach for defining event filters were identified as future requirements.

The implementation of the NoiseCap experiment has highlighted a substantial lack of free and open standards and solutions for processing spatial time series, such as ADS-B Mode S data, to identify events and apply event pattern matching. Therefore, transferring the developed system to a different geographic area would require substantial adjustments to the underlying analysis of aircraft trajectories.

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

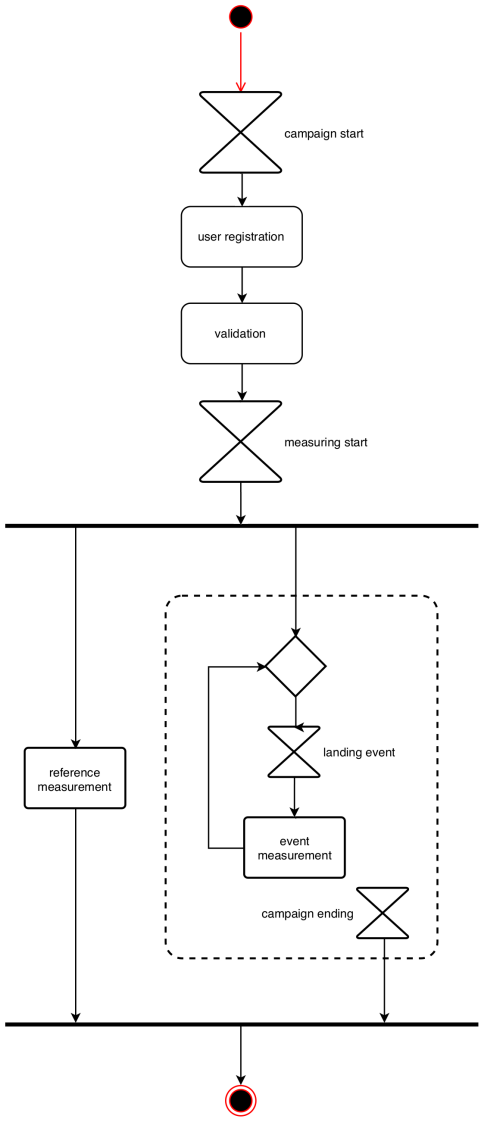

Figure 5. Activity diagram of the NoiseCap experiment.

A standardised approach to filter raw ADS-B data based on an area of interest (e.g. based on GeoMQTT or PubSub) could help to develop similar applications in a sustainable manner. In addition, the lack of support for client-side standards led to rely on proprietary solutions (e.g. Telegram-based push messages) for notifying the volunteers.

4.2. Risk Monitoring

Real-time information and the identification of specific events play a central role in the field of risk monitoring and early warning systems. In addition to phenomena measurements, simulation is a key asset in the system architectures. The timely provision of inputs for such simulations therefore is an important goal and can be achieved by utilizing event streams. On the other hand, the results of a simulation model can also be materialized as a stream of events on its own. In the following we provide architectural insights in the context of two independent use cases.

Seadikes and estuarine dikes embody the main coastal defence structure in Germany and protect the low-lying areas against flooding. Dike failures in these areas may affect millions of people and

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

economic values, which are worth protecting by monitoring the dikes and alerting in time. But today's early warning systems for dike failures are based exclusively on water level forecasts and do not consider other parameters such as wind, waves, currents, heavy rainfalls or the condition of the dike structure itself. In case of simultaneously-occurring adverse events, dikes may fail before the design load is reached. A real-time warning system for dike failures has been established in the context of the EarlyDike project (see [47]) which combines the detection of such different events. The system stands on three pillars: simulation of the environmental impacts by forecasting storm surges and wave loads, monitoring the condition of the dike itself by deploying new sensor technologies to measure the moisture penetration and the deformation and, finally, simulating the extent of a flood in case of a dike failure.

In analogy to seadikes, river dams are critical infrastructure and require continuous monitoring with various observations. Although safety monitoring systems are available at these structures, there remains a residual risk of failure. Potential damages associated with dams usually stem from the failure of the structure due to overtopping, seepage water flow, and deformation. A critical dam failure may claim lives and cause massive material damage. To further minimize the residual risk, measures from different sources must be combined and analysed to extract maximum information. In the research project TaMIS an already existing set of core SDI components has been complemented with additional spatial time series and analysis components to form a dam information system [48]. One of the core requirements and complementing modules of such a system is an eventing module that detects low level events, e.g. threshold violations or sensor failure and communicates detected events to a responsible person.

4.2.1. Architectural Approach

The system architectures developed for seadikes, river dams and urban Infrastructures all share a common technological baseline. The backends are based on a Sensor and Spatial Data Infrastructure (SSDI), which follow the concept of the Sensor Web and, therefore, utilize OGC standards to make the time series data of the simulations and the sensors accessible [49]. In the EarlyDike project, the sensor platforms and the simulations publish their measured or generated events with lightweight IoT protocols since they are adapted to resource-constraint environments. Especially the GeoMQTT protocol (see Section 2.3.2) is used to establish a geo event bus to interconnect sensors, simulators and the SSDI (see Figure 6). In order to connect the SWE service layer the geo event bus adopts concepts of the Sensor Bus by Bröring et al. [50]. The geo event bus ensures the deliverance and provisioning of real-time spatio-temporal events and data in time-sensitive architectures such as early warning systems.

The dam monitoring system developed in the TaMIS project enhances the SensorWeb standards with a lightweight access layer. The Sensor Event API adopts elements of PubSub [18]. It serves a set of predefined rules, detected series events and allows a user to subscribe for a rule of a particular time series. The Series API[12] is based on the Sensor Observation Service Standard [52] and introduces the time series object as a new element in the Sensor Web Enablement. The time series object is crucial to get explicit references among Series API and Sensor Event API.

The eventing module validates incoming observations against a set of threshold rules through database triggers when adding them to a harmonized spatial time series database schema. Stored events are sent via Email to subscribed users in regular intervals. In order to make events accessible through the web a web service has been developed that adopts concepts from the PubSub Standard and provides a JSON/REST API. Figure 7 sketches the system's architecture.

In the context of river dam monitoring where processes are of slow characteristic early access to reliable severe weather warnings is helpful to start dischargement in a water reservoir. Besides

[12] https://github.com/52North/series-rest-api

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 August 2018 doi:10.20944/preprints201808.0132.v1

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

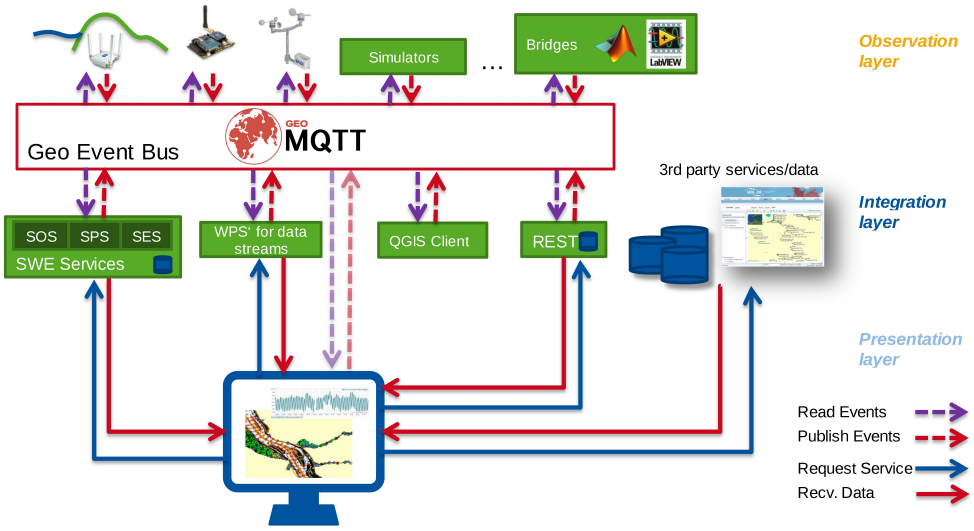

Figure 6. Architecture with GeoEvent bus in the EarlyDike project (adapted from [51]).

real time event pattern matching the developed dam monitoring system accesses severe weather warnings from the Web Feature Service of the Deutscher Wetterdienst (German Meteorological Office). Predefined filters (location, topic) extract those weather warnings that are of interest for the monitored structures.

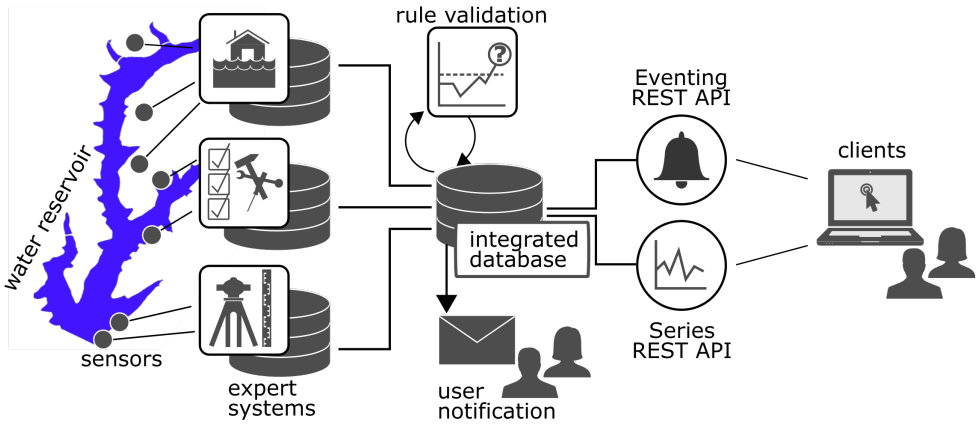

Figure 7. Architecture of the TaMIS project.

4.2.2. Lessons Learned

During the EarlyDike project, missing solutions for emitting spatio-temporal events by low-cost IoT devices and for filtering these events spatially and temporally were identified. This is essential for the project's aims since dikes which protect the hinterland are widespread along the coastline. Because of this gap, the MQTT protocol was extended to spatio-temporal filtering capabilities (see Section 2.3.2). However, neither a standard for spatio-temporal data streams nor a domain-specific language for event-driven pattern matching (e.g. as described in Section 3.1) do exist. Additionally, the integration of spatio-temporal data streams in traditional GIS Software is lacking. Coupling standardized spatio-temporal data streams in GIS software has high potential to improve spatial analysis and allows for real-time operations.

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

One gap within the TaMIS eventing application is the implicit linkage among eventing rules and observation streams (time series). The eventing rules in TaMIS observe threshold overshoots and undershoots of time series. A robust linkage within the web service layer among an eventing rule and a time series is crucial for the eventing logic to work robustly. However, this linkage is currently solved merely implicit at the web service layer and has to be configured explicitly in a client application. In the future this explicitness have to be transferred to the web service layer.

4.3. Disaster Response and Urban Sensing

The proliferation of heterogeneous urban sensors and volunteered geographic data (e.g. from crowd-sourcing) provides novel data-driven methods for emergency response and early detection of hazardous events. Challenging are the combination of these massive real-time data streams from urban sensors and the common reasoning on these heterogeneous data formats. As an example, a traffic loop detector provides a different spatio-temporal granularity than a video camera or a twitter post, but all data sources may contain useful insights on one particular congestion or flooding event in a smart city.

The European FP7 project INSIGHT [53] focussed on the intelligent synthesis and real-time response using heterogeneous data streams. Goal was to gain situation-awareness and perform crisis response based on heterogeneous sensor data streams. Thus, several event detection tasks were addressed by the consortium and a special architecture for spatio-temporal event detection has been proposed.

4.3.1. Architectural Approach

The proposed architecture we introduced in the INSIGHT project[13] [54] has been inspired by the TechniBall system [55] and follows the Lambda architecture design principles for Big Data systems [56]. A sketch of the architecture and the interconnection among the components is presented in Figure 8, a comparison to other event processing frameworks can be found in [5]. Every data stream is analysed individually for anomalies. In this detection functions (e.g. clustering, prediction, thresholds, etc.) on the data streams can be applied. The resulting anomalies are joined at a round table. A final Complex Event Processing component allows the formulation of complex regular expressions on the function values derived from heterogeneous data streams. The predictions comprise future predictions of sensor values [57], but also regression at unobserved locations [58].

In the succeeding VaVeL project[14] the architecture of the INSIGHT project is reused and the application focuses on urban sensor data streams which are combined to gain situation-awareness. One of the project goals is to provide a multi-modal trip planner for the citizen that incorporates predictions and extracted events to avoid delays. Thus, the tram delays are predicted [60], and these predictions are incorporated in the OpenTripPlanner framework[15] to provide route suggestions that avoid delays.

4.3.2. Lessons Learned

Utilizing real-time values and predictions for situation awareness faces some problems which are listed in the following.

- **Utilization:** Using these real-time predictions in a routing framework causes performance issues, we thus improved the applicability of existing transfer pattern algorithms towards dynamic costs [61].

[13] http://www.insight-ict.eu
[14] http://vavel-project.eu
[15] http://www.opentripplanner.org

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 August 2018 doi:10.20944/preprints201808.0132.v1

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

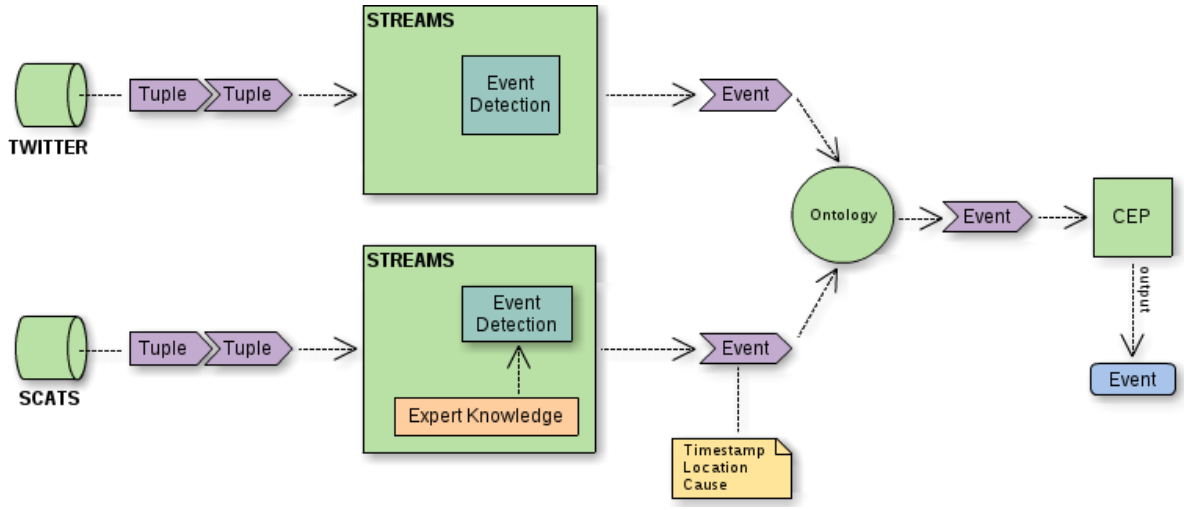

Figure 8. INSIGHT Architecture for event detection from heterogeneous data streams exemplified with two input streams Twitter and traffic loop data derived by SCATS, compare [54,59].

- **Proactive control:** If any person would follow our route recommendations, we would cause novel unexpected traffic congestions at unexpected places. Indeed, utilizing a prediction for a control decision modifies the distribution of the observed data and makes the prediction model invalid. A common approach towards this problem is bandit feedback learning. In [62] we apply on of its algorithms, reinforcement learning, to the problem of traffic control and street network performance was improved.
- **Life-long systems:** A real-time system has to cope with a changing environment. In route planning, e.g., one may not rely on the availability of a certain pathway. Thus, we tackle this problem by a distributed online mapping method [63] to identify obstacles and free spaces in real-time.

4.4. Spatio-Temporal Modelling

Often, geospatial applications rely on the execution of spatio-temporal models in order to determine or predict the distribution of specific properties. One example for such an application is the calculation of pollutants within the public sewage system in case of heavy rainfall events, as it is performed by the COLABIS project (https://colabis.de). Another example is the WaCoDiS project (https://wacodis.fbg-hsbo.de) which aims at the identification of element input in water courses and dams based on the combination of in-situ and remote sensing data.

Although both of these projects deal with different thematic questions, they share a common requirement: The execution of models and analysis algorithms in order to detect certain properties, as soon as updated input data is available. In case of the COLABIS project, this concerns the availability of radar-based rainfall data: based on the updated radar data, the latest rainfall data can be used for updated flood model runs that predict the distribution of pollutants in the sewage system. In case of the WaCoDiS project, the corresponding data analysis algorithms shall be executed as soon as new remote sensing data is available that fulfils certain quality requirements (e.g. no cloud coverage, etc.).

Thus, both projects would benefit from an automated trigger as soon as new suitable data is available so that new processing workflows can be initiated.

4.4.1. Architectural Approach

Both of the above mentioned projects are still work in progress. Within the COLABIS project, currently a time-based trigger is used. As the time interval in which radar updates are available is

approximately known, this information can be used for triggering the workflow execution in regular intervals. However, for further optimization in case of irregular availability of data, the COLABIS workflow would benefit from an event-based approach as it is currently developed in the WaCoDiS project.

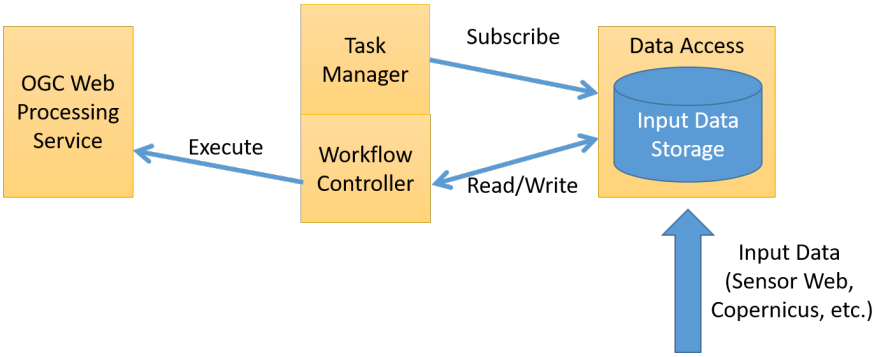

Figure 9. Simplified Overview of the WaCoDiS Workflow Architecture.

Within the WaCoDiS project the workflow partially outlined in Figure 9 is followed. In this case, it is necessary to start a process chain as soon as new remote sensing images are available. For this purpose, all incoming data such as Sensor Web observations or metadata about Copernicus satellite images is stored in a central repository. A task manager holds several subscriptions on this repository, so that it is notified as soon as new data is available, which matches the conditions defined for the subscriptions. As soon as all necessary data for a workflow run is available, a workflow controller reacts on the information about new available data and triggers the execution of the corresponding data analysis processes on OGC Web Processing Service instances.

This publish/subscribe architecture enables automatic process/model execution as soon as the conditions for a new run are fulfilled. As a result, this approach ensures the timely delivery of new information as soon as updated input data is available. Currently, different approaches for enabling the publish/subscribe patterns within the WaCoDiS infrastructure are investigated. Besides the OGC PubSub standard [21] also mainstream IT technologies are evaluated.

4.4.2. Lessons Learned

Currently the presented workflow is still in development so that evaluation results are not yet available. However, in the near future the presented workflow will be further refined and it is expected that follow-up publications will present a detailed analysis of this concept. It is expected that especially the WaCoDiS developments will make a contribution to enabling event-driven data flows in spatial data infrastructures, as outlined in Section 3.3.

5. Discussion and Conclusions

The presented application domains and related projects (Section 4) cover current research activities of the authors. The illustrated architectural approaches could be translated to other relevant domains where real-time spatio-temporal data also play an important role. Currently, new research activities are focussing on fields such as logistics, precision farming or Earth Observation that can build upon the outlined solutions. At the same time, the landscape that we outlined in 2 illustrates the high diversity of approaches in the field of Eventing in geo-information science. Recent research activities have also highlighted the challenge of event processing capabilities in the area of geospatial IoT architectures (see [64]). This diversity and the related challenges in combination with the availability of sophisticated software solutions for stream processing evolve into a challenging research area. This is especially

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 385; doi:10.3390/ijgi7100385

relevant in the context of IoT technologies, networks, and devices which promise to contribute an additional source of real-time data to spatial data infrastructures.

Following the analysis of systems which have been developed in various application domains we identified several existing gaps. The commonly occurring issues that need to be tackled in the coming years in order to take full advantage of the uptake of eventing in GI Science are outlined below.

- *Inconsistencies between classic data access methods and event-driven approaches:* The implementation of forms of asynchronous client-server interaction is considered highly relevant to most geospatial application scenarios. Nonetheless, the analysis of existing SDIs shows that the majority of spatio-temporal data is provided either in a static way or via standardized Web services. Still, these Web services rely on the request/reply pattern. Also the current OGC standard baseline is still mainly based on synchronous web service capabilities, which have insofar primarily addressed the request/reply model. Thus, while waiting for an event, a client must repeatedly request the desired information (polling). This has undesirable side effects: if a client polls frequently this can increase server load and network traffic, and if a client polls infrequently it may not receive a message when it is needed. These issues are accentuated when event occurrences are unpredictable, or when the delay between event occurrence and client notification must be small. The establishment of the publish/subscribe pattern could eradicate these shortcomings. It could be argued that a growing number of applications may benefit from an interoperable and uniform way of implementing event-driven architectures. Possibly, the transition to an Event-driven SOA could see a parallel transition to an Event-driven SDI, backed by some or all of the presented standards and solutions.
- *Heterogeneous approaches for defining event patterns:* While the OGC PubSub standard aims at providing a common approach how to enable push-based communication patterns within SDIs, it does not address how the events shall be described that data consumers are interested in. Even though there were previous efforts to address this topic (e.g. the OGC Event Pattern Markup Language Discussion Paper [65]), the way of defining event patterns is still mostly depending on the used technologies. However, to ensure full interoperability, also the harmonization of approaches for defining event patterns is necessary. This would also comprise extensions of the OGC PubSub standard. An interesting field of work would be the addition of event stream processing capabilities [65]. Stream processing goes beyond the capability of basic data filtering. It generates higher level information from raw data stream (e.g. event patterns such as "temperature < 20 followed by temperature >= 20"). This would allow the integration of concepts such as Complex Event Processing into the standard and could be realized as a dedicated event processing extension of the OGC PubSub standard.
- *Standardization and corresponding support in established GIS software.* At the moment the integration of event-based communication flows in established GIS software is still a gap. Even though there are dedicated components such as Esri's GeoEvent Server the integration into typical desktop GIS is still missing as it would require new concepts how to handle such information flows within the existing graphical user interfaces. At the same time new lightweight, standardized solutions (e.g. based on REST and JSON) would be needed to allow the development of sophisticated client solutions. However, the availability of such lightweight interoperability standards is still a gap which needs to be resolved (potentially through faster, lightweight standardisation approaches). In this context, the definition of additional interface bindings and profiles for OGC PubSub delivery methods is an important task. In order to allow smooth and interoperable integration of OGC compliant Publish/Subscribe services, such profiles are crucial. The integration of modern and well-established technologies such as AMQP, MQTT and JMS with specific profiles is therefore a central goal.
- *Bidirectional integration of IoT devices in event-driven SDI:* Until now, SDIs are mainly dealing with data but not with the devices delivering this data. However, the direct integration of IoT devices (e.g. resource-constrained platforms, sensors, actuators, moving objects) would create

new opportunities. On the one hand this would open up a new dimension in SDIs by having IoT devices as direct sensors delivering live input about their environment. On the other hand, the bi-directional integration of IoT devices would also make it possible to define and execute specific sensing tasks or to even interact with the environment.

- *Lack of semantic interoperability of geospatial events:* At the moment, the semantics of events occurring an event-enabled SDI are not yet harmonised from a semantic point of view. For the future it would be important to have common semantics for events (e.g. new IoT device registered in an SDI, threshold exceedance of detected values) to achieve full interoperability. At least for a common foundation of typical events, a dedicated vocabulary or ontology would be desirable.

In summary, it can be concluded that the integration of IoT devices and the enablement of event-based communication patterns would be a valuable enhancement of SDIs. While first relevant results are already available, there is still a broad range of challenges which need to be addressed before achieving the goal of Event-Driven Spatial Data Infrastructures.

Acknowledgments: The contributions of Stefan Herle (gia - RWTH Aachen University) were supported by the project EarlyDike (German Federal Ministry of Education and Research, programme Geotechnologien, under grant agreement No 03G0847A). The contributions of Matthes Rieke and Simon Jirka (52°North GmbH) were supported by the project COLABIS (German Federal Ministry of Education and Research, programme Geotechnologien, under grant agreement No 03G0852C) as well as the project WaCoDiS (German Federal Ministry of Transport and Digital Infrastructure, programme mFUND, under grant agreement No 19F2038D). Thomas Liebig (TU Dortmund) was supported by the European Horizon 2020 programme under grant agreement No 688380 "VaVeL: Variety, Veracity, VaLue: Handling the Multiplicity of Urban Sensors" and by the Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis", project B4 "Analysis and Communication for Dynamic Traffic Prognosis". The contributions of Christian Malweski (Wupperverband) were supported by the project TaMIS (German Federal Ministry of Education and Research, programme Geotechnologien, under grant agreement No 03G0854B). Lorenzo Bigagli would like to credit his colleagues Roberto Salzano and Massimiliano Olivieri for their contributions to the NoiseCap project.

Author Contributions: Matthes Rieke contributed to the introduction, state-of-the-art, gaps, applications and discussions chapters. Lorenzo Bigagli contributed to the state-of-the-art, gaps, applications and discussions chapters. Stefan Herle worked on the state-of-the-art, applications and conclusion chapters. Simon Jirka contributed to the introduction, gaps, applications and discussions chapters. Alexander Kotsev contributed to the introduction, state-of-the-art, applications and discussions chapters. Thomas Liebig contributed to the introduction, state-of-the-art and applications chapters. Christian Malewski contributed to the gaps, applications and conclusions chapter. Thomas Paschke contributed to the state-of-the-art and conclusions chapter. Christoph Stasch contributed to the state-of-the-art and the gaps chapters.

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

Abbreviations

The following abbreviations are used in this manuscript:

ADS-B	Automatic Dependent Surveillance - Broadcast
AMQP	Advanced Message Queuing Protocol
ATC	Air Traffic Control
CEP	Complex Event Processing
CoAP	Constrained Application Protocol
CSW	Catalogue Service for the Web
DSP	Distributed Stream Processing
EDA	Event-Driven Architecture
EML	Event Pattern Markup Language
EPL	Event Processing Language
GIS	Geographic Information System
GIScience	Geographic Information Science
INSPIRE	Infrastructure for Spatial Information in Europe
IoT	Internet of Things
JMS	Java Message Service
JSON	JavaScript Object Notation
MEP	Message Exchange Pattern
MQTT	Message Queuing Telemetry Transport
OGC	Open Geospatial Consortium

PubSub	OGC Publish/Subscribe Interface Standard
REST	REpresentational State Transfer
SAS	Sensor Alert Service
SDI	Spatial Data Infrastructure
SES	Sensor Eventing Service
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SOS	Sensor Observation Service
SPS	Sensor Planning Service
SSDI	Sensor and Spatial Data Infrastructure
SWE	Sensor Web Enablement
SWES	SWE Service Model
WEPS	Web Event Processing Service
WNS	Web Notification Service
WPS	Web Processing Service
XML	eXtensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

References

1. Rajabifard, A.; Feeney, M.E.F.; Williamson, I.P. Future directions for SDI development. *International Journal of Applied Earth Observation and Geoinformation* **2002**, *4*, 11–22.
2. Galton, A. Outline of a formal theory of processes and events, and why GIScience needs one. International Workshop on Spatial Information Theory. Springer, 2015, pp. 3–22.
3. Schade, S.; Craglia, M. A Future Sensor Web for the Environment in Europe. EnviroInfo, 2010, pp. 529–539.
4. Liebig, T.; Morik, K. Report on end-user requirements, test data, and on prototype definitions. Technical Report FP7-318225 D5.1, TU Dortmund and Insight Consortium Members, 2013.
5. Souto, G.; Liebig, T. On Event Detection from Spatial Time series for Urban Traffic Applications. In *Solving Large Scale Learning Tasks: Challenges and Algorithms*; Michaelis, S.; Piatkowski, N.; Stolpe, M., Eds.; Springer International Publishing, 2016; Vol. 9580, pp. 221–233.
6. Aggarwal, C.C. *Outlier Detection*; Springer: New York, NY, USA, 2013.
7. Florescu, S.; Körner, C.; Mock, M.; May, M. Efficient Mobility Pattern Stream Matching on Mobile Devices. Proc. of the Ubiquitous Data Mining Workshop (UDM 2012), 2012, pp. 23–27.
8. Dodge, S.; Weibel, R.; Lautenschütz, A.K. Towards a taxonomy of movement patterns. *Information visualization* **2008**, *7*, 240–252.
9. del Rio, J.; Toma, D.; Martinez, E.; C. OReilly, T.; Delory, E.; Pearlman, J.; Waldmann, C.; Jirka, S. A Sensor Web Architecture for Integrating Smart Oceanographic Sensors into the Semantic Sensor Web. *IEEE Journal of Oceanic Engineering* **2017**, *PP*, 1–13.
10. EC. Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE). *Published in the official Journal on the 25th April* **2007**.
11. Michelson, B.M. Event-Driven Architecture Overview. Technical report, Elemental Links, 2006. First issue: Patricia Seybold Group, Feb 2002.
12. Bukhsh, Z.A.; van Sinderen, M.; Singh, P. SOA and EDA: a comparative study - similarities, difference and conceptual guidelines on their usage. Twelfth International Conference on e-Business; ICE-B, , 2015; pp. 213–220.
13. Kotsev, A.; Schleidt, K.; Liang, S.; van der Schaaf, H.; Khalafbeigi, T.; Grellet, S.; Lutz, M.; Jirka, S.; Beaufils, M. Extending INSPIRE to the Internet of Things through SensorThings API. *Geosciences (2076-3263)* **2018**, *8*.
14. Eugster, P.T.; Felber, P.A.; Guerraoui, R.; Kermarrec, A.M. The Many Faces of Publish/Subscribe. *ACM Comput. Surv.* **2003**, *35*, 114–131.
15. Everding, T.; Echterhoff, J. OGC OWS-6 SWE Event Architecture Engineering Report. Technical Report 09-032, OGC, 2009.
16. Echterhoff, J. OWS-7 Event Architecture Engineering Report. Technical Report 10-060r1, OGC, 2010.
17. Bigagli, L.; Rieke, M. The new OGC Publish/Subscribe Standard - applications in the Sensor Web and the Aviation domain. *Open Geospatial Data, Software and Standards* **2017**, *2*, 18.
18. OGC. *OGC Publish/Subscribe Interface Standard 1.0 – Core*, 1.0 ed.; OGC, 2016.

19. OGC. *OGC Publish/Subscribe Interface Standard 1.0 SOAP Protocol Binding Extension*, 1.0 ed.; OGC, 2016.
20. Graham, S.; Hull, D.; Murray, B. Web Services Base Notification 1.3. Technical report, OASIS, 2016.
21. Bigagli, L.; Vretanos, P.P.A.; Lawrence, M.; Papeschi, F.; Martell, R. Testbed-12 PubSub/Catalog Engineering Report. Technical Report OGC 16-137r2, Open Geospatial Consortium, 2017.
22. Westerholt, R.; Resch, B. Asynchronous Geospatial Processing: An Event-Driven Push-Based Architecture for the OGC Web Processing Service. *Transactions in GIS* **2015**, *19*, 455–479.
23. Herle, S.; Blankenbach, J. Enhancing the OGC WPS interface with GeoPipes support for real-time geoprocessing. *International Journal of Digital Earth* **2017**, *11*, 48–63.
24. Schumann, G.; Lieberman, J. Incident Management Information Sharing (IMIS) Internet of Things (IoT) Architecture Engineering Report. Technical report, OGC, 2018.
25. Chen, Y.; Kunz, T. Performance evaluation of IoT protocols under a constrained wireless access network. 2016 International Conference on Selected Topics in Mobile Wireless Networking (MoWNeT), 2016, pp. 1–7.
26. ISO/IEC 20922:2016, Information technology – Message Queuing Telemetry Transport (MQTT) v3.1.1. https://www.iso.org/standard/69466.html. Accessed: 2018-02-10.
27. Herle, S.; Blankenbach, J. GeoPipes using GeoMQTT. Geospatial Data in a Changing World: Selected papers of the 19th AGILE Conference on Geographic Information Science; Sarjakoski, T.; Santos, M.Y.; Sarjakoski, L.T., Eds., 2016, pp. 383–398.
28. OGC. *OpenGIS Implementation Standard for Geographic information - Simple feature access*, 1.2.1 ed.; OGC, 2011.
29. Allen, J.F. Maintaining knowledge about temporal intervals. *Communication of ACM* **1983**, pp. 832–843.
30. Vilain, M.B. A System for Reasoning About Time. AAAI, 1982, Vol. 82, pp. 197–201.
31. Fitzpatrick, B.; Slatkin, B.; Atkins, M.; Genestoux, J. Pubsubhubbub core 0.3–working draft. *Project Hosting on Google Code* **2010**.
32. Kreps, J. Questioning the Lambda Architecture. The Lambda Architecture has its merits, but alternatives are worth exploring. https://www.oreilly.com/ideas/questioning-the-lambda-architecture. Accessed: 2018-05-23.
33. Zhang, F.; Zheng, Y.; Xu, D.; Du, Z.; Wang, Y.; Liu, R.; Ye, X. Real-Time Spatial Queries for Moving Objects Using Storm Topology. *ISPRS International Journal of Geo-Information* **2016**, *5*.
34. Mattheis, S.; Al-Zahid, K.K.; Engelmann, B.; Hildisch, A.; Holder, S.; Lazarevych, O.; Mohr, D.; Sedlmeier, F.; Zinck, R. Putting the car on the map: A scalable map matching system for the Open Source Community. GI-Jahrestagung, 2014.
35. Lopez, M.A.; Lobato, A.G.P.; Duarte, O.C.M.B. A Performance Comparison of Open-Source Stream Processing Platforms. *2016 IEEE Global Communications Conference (GLOBECOM)* **2016**, pp. 1–6.
36. Wiener, P.; Simko, V.; Nimis, J. Taming the Evolution of Big Data and its Technologies in BigGIS - A Conceptual Architectural Framework for Spatio-Temporal Analytics at Scale. Proceedings of the 3rd International Conference on Geographical Information Systems Theory, Applications and Management (GISTAM 2017). SCITEPRESS, 2017, pp. 90–101.
37. What is ArcGIS GeoEvent Server? http://enterprise.arcgis.com/en/server/latest/get-started/windows/what-is-arcgis-geoevent-server.htm. Accessed: 2018-06-21.
38. Real-Time and Big Data GIS: The Road Ahead - Esri Federal GIS Conference 2018. http://proceedings.esri.com/library/userconf/fed18/papers/fed-126.pdf. Accessed: 2018-06-21.
39. Discovering GeoAnalytics (or what is GeoAnalytics, and why should I care?). https://communityhub.esriuk.com/geoxchange/2016/12/22/discovering-geoanalytics-or-what-is-geoanalytics-and-why-should-i-care. Accessed: 2018-06-21.
40. Real-Time and Big Data GIS at massive Scale - Esri Federal GIS Conference 2017. http://proceedings.esri.com/library/userconf/fed17/papers/fed_107.pdf. Accessed: 2018-06-21.
41. Everding, T.; Echterhoff, J. Event Pattern Markup Language (EML). Technical report, Open Geospatial Consortium, 2008.
42. Gerboles, M.; Spinelle, L.; Kotsev, A.; Signorini, M.; Srl, L. AirSensEUR: An Open-Designed Multi-Sensor Platform for Air Quality Monitoring. Proceedings of the Fourth Scientific Meeting EuNetAir, Linkoping, Sweden, 2015, pp. 3–5.
43. Roglia, E.; Craglia, M. MYGEOSS Project. *GeoInformatics* **2015**, *18*, 38.

44. Memè, S.; Delory, E.; Felgines, M.; Pearlman, J.; Pearlman, F.; del Rio, J.; Martinez, E.; Masmitja, I.; Gille, J.; Rolin, J.F.; Golmen, L.; Hareide, N.R.; Waldmann, C.; Zielinski, O. NeXOS - Next generation, cost-effective, compact, multifunctional web enabled ocean sensor systems. OCEANS 2017 - Anchorage, 2017, pp. 1–10.
45. Rieke, M.; Jirka, S. Interface standards for applications of deep and ultra-deep glider. Technical report, BRIDGES, 2016.
46. Latre, M.; Lopez-Pellicer, F.J.; Zarazaga-Soria, F.J.; Mazzetti, P.; Nativi, S. State of the art on geographical information and open data technologies: a European-project-based approach. *International Journal of Geographical Information Science* **2016**.
47. Krebs, V.; Herle, S.; Schwab, M.; Quadflieg, T.; Gries, T.; Blankenbach, J.; Schüttrumpf, H. Implementation of sensor-based dike monitoring by smart geotextiles. SCACR2017, International Short Course on Applied Coastal Research, Santander, Spain. ICCE, 2017.
48. Stasch, C.; Pross, B.; Gräler, B.; Malewski, C.; Förster, C.; Jirka, S. Coupling sensor observation services and web processing services for online geoprocessing in water dam monitoring. *International Journal of Digital Earth* **2018**, *11*, 64–78, [https://doi.org/10.1080/17538947.2017.1319977].
49. Broering, A.; Echterhoff, J.; Jirka, S.; Simonis, I.; Everding, T.; Stasch, C.; Liang, S.; Lemmens, R. New Generation Sensor Web Enablement. *Sensors* **2011**, *11*, 2652–2699.
50. Bröring, A.; Maué, P.; Janowicz, K.; Nüst, D.; Malewski, C. Semantically-Enabled Sensor Plug and Play for the Sensor Web. Sensors, 2011.
51. Herle, S.; Becker, R.; Blankenbach, J. Smart sensor-based geospatial architecture for dike monitoring. IOP Conference Series: Earth and Environmental Science, 2016, Vol. 34.
52. OGC. *OGC Sensor Observation Service Interface Standard*, 2.0 ed.; OGC, 2012.
53. Kinane, D.; Schnitzler, F.; Mannor, S.; Liebig, T.; Morik, K.; Marecek, J.; Gorman, B.; Zygouras, N.; Katakis, Y.; Kalogeraki, V.; Gunopulos, D. Intelligent Synthesis and Real-time Response using Massive Streaming of Heterogeneous Data (INSIGHT) and its anticipated effect on Intelligent Transport Systems (ITS) in Dublin City, Ireland. Proceedings of the 10th ITS European Congress, Helsinki, 2014, pp. 1–12.
54. Schnitzler, F.; Liebig, T.; Mannor, S.; Souto, G.; Bothe, S.; Stange, H. Heterogeneous Stream Processing for Disaster Detection and Alarming. IEEE International Conference on Big Data. IEEE Press, 2014, pp. 914–923.
55. Gal, A.; Keren, S.; Sondak, M.; Weidlich, M.; Blom, H.; Bockermann, C. Grand Challenge: The TechniBall System. Proceedings of the 7th ACM International Conference on Distributed Event-based Systems; ACM: New York, NY, USA, 2013; DEBS '13, pp. 319–324.
56. Marz, N. *Big data : principles and best practices of scalable realtime data systems*; O'Reilly Media, 2013.
57. Liebig, T.; Piatkowski, N.; Bockermann, C.; Morik, K. Dynamic Route Planning with Real-Time Traffic Predictions. *Information Systems* **2017**, *64*, 258–265.
58. Liebig, T.; Xu, Z.; May, M. Incorporating Mobility Patterns in Pedestrian Quantity Estimation and Sensor Placement. In *Citizen in Sensor Networks*; Springer Berlin Heidelberg, 2013; pp. 67–80.
59. Artikis, A.; Weidlich, M.; Schnitzler, F.; Boutsis, I.; Liebig, T.; Piatkowski, N.; Bockermann, C.; Morik, K.; Kalogeraki, V.; Marecek, J.; Gal, A.; Mannor, S.; Gunopulos, D.; Kinane, D. Heterogeneous Stream Processing and Crowdsourcing for Urban Traffic Management. Proc. 17th International Conference on Extending Database Technology (EDBT), Athens, Greece, March 24-28, 2014. OpenProceedings.org, 2014, pp. 712–723.
60. Heppe, L.; Liebig, T., Real-Time Public Transport Delay Prediction for Situation-Aware Routing. In *KI 2017: Advances in Artificial Intelligence: 40th Annual German Conference on AI, Dortmund, Germany, September 25–29, 2017, Proceedings*; Kern-Isberner, G.; Fürnkranz, J.; Thimm, M., Eds.; Springer International Publishing: Cham, 2017; pp. 128–141.
61. Liebig, T.; Peter, S.; Grzenda, M.; Junosza-Szaniawski, K., Dynamic Transfer Patterns for Fast Multi-modal Route Planning. In *Societal Geo-innovation: Selected papers of the 20th AGILE conference on Geographic Information Science*; Bregt, A.; Sarjakoski, T.; van Lammeren, R.; Rip, F., Eds.; Springer International Publishing: Cham, 2017; pp. 223–236.
62. Liebig, T.; Sotzny, M. On Avoiding Traffic Jams with Dynamic Self-Organizing Trip Planning. In *13th International Conference on Spatial Information Theory (COSIT 2017)*; Clementini, E.; Donnelly, M.; Yuan, M.; Kray, C.; Fogliaroni, P.; Ballatore, A., Eds.; Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik: Dagstuhl, Germany, 2017; Vol. 86, *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 17:1–17:12.

63. Shaik, N.; Liebig, T.; Kirsch, C.; Müller, H., Dynamic Map Update of Non-static Facility Logistics Environment with a Multi-robot System. In *KI 2017: Advances in Artificial Intelligence: 40th Annual German Conference on AI, Dortmund, Germany, September 25–29, 2017, Proceedings*; Kern-Isberner, G.; Fürnkranz, J.; Thimm, M., Eds.; Springer International Publishing: Cham, 2017; pp. 249–261.
64. Kamilaris, A.; Ostermann, F.O. Geospatial Analysis and the Internet of Things. *ISPRS International Journal of Geo-Information* **2018**, *7*.
65. Everding, T.; Echterhoff, J.; Jirka, S. Event Processing in Sensor Webs. *GeoInformatik 2009* **2009**, *35*, 11–19.