Chapter 11
Real-Time Data Processing

Fabiana Fournier and Inna Skarbovsky

Abstract To remain competitive, organizations are increasingly taking advantage of the high volumes of data produced in real time for actionable insights and operational decision-making. In this chapter, we present basic concepts in real-time analytics, their importance in today's organizations, and their applicability to the bioeconomy domains investigated in the DataBio project. We begin by introducing key terminology for event processing, and motivation for the growing use of event processing systems, followed by a market analysis synopsis. Thereafter, we provide a high-level overview of event processing system architectures, with its main characteristics and components, followed by a survey of some of the most prominent commercial and open source tools. We then describe how we applied this technology in two of the DataBio project domains: agriculture and fishery. The devised generic pipeline for IoT data real-time processing and decision-making was successfully applied to three pilots in the project from the agriculture and fishery domains. This event processing pipeline can be generalized to any use case in which data is collected from IoT sensors and analyzed in real-time to provide real-time alerts for operational decision-making.

11.1 Introduction and Motivation

To stay relevant and competitive, modern enterprises must continuously monitor events of interest, assess changing conditions, and make fast decisions. The continuous flow of event streams, such as customer orders, bank deposits, invoices, social media updates, market data, Global Positioning System (GPS)-based location information, signals from Supervisory Control and Data Acquisition (SCADA) systems, and temperature from sensors and IoT devices, are analysed to help enterprises respond in real-time to changing market and environmental conditions. Furthermore, with the emergence of the Internet of Things (IoT), organisations are taking advantage of the high volumes of data produced by sensors for real-time situational

F. Fournier (✉) · I. Skarbovsky
IBM Research—Haifa, University of Haifa Campus, Mount Carmel, 3498825 Haifa, Israel
e-mail: fabiana@il.ibm.com

C. Södergård et al. (eds.), *Big Data in Bioeconomy*,
https://doi.org/10.1007/978-3-030-71069-9_11

awareness and real-time insights. IoT generates a huge amount of high-speed real-time data in different formats from a vast number of sources that must be analysed quickly for timely responses. IoT sensors enable decision-makers to continuously monitor and track various parameters that help them in their day-to-day operations.

Traditionally, organisations used to store data in databases and then process and analyse it after storage using batch processing. As mentioned above, the unexpected growth in the number of events due to advanced operations, massive sensor adoption, mobile devices, and high-speed networks has resulted into an exponential increase in data volume. Moreover, organisations need to be increasingly capable of extracting insights from real-time business events, because data loses value with the passage of time. Many of today's common applications such as fraud detection, algorithmic trading, network monitoring, predictive maintenance, and sales and marketing require the processing of data in real time. *Event Stream Processing* (ESP) has evolved to cope with the analysis of real-time streaming data.

To understand the essence of ESP, let's decompose the name to its three basic terms: event + stream + processing. An *event* is an occurrence within a particular system or domain; it is something that has actually happened or is contemplated as having happened in that domain. The word *event* is also used to refer to a programming entity that represents such an occurrence in a computing system [1]. A *stream* is a constant and continuous flow of events that navigate into and around companies from thousands of connected devices, IoT, and any other sensors. An *event stream* is a sequence of events arranged in some order, typically by time. Enterprises generally have three different kinds of event streams: business transactions, such as customer orders, bank deposits, and invoices; information reports, such as social media updates, market data, and weather reports; and IoT data, such as GPS-based location information, signals from SCADA systems, and temperature measurements from sensors [2]. *Processing* is the final act of analysing all this data in real-time.

ESP is the processing of continuous event data streams in real time. It helps identify the patterns and anomalies within these data streams that are important to an enterprise, such as event correlation, causality, and timing. ESP also enables organisations to respond quickly to critical events, thus saving time, money, and resources. It is also known as real-time streaming analytics, streaming analytics, and (complex) event processing [3].

Specifically, stream analytics provided by ESP platforms [4]:

- Support situation awareness through dashboards and alerts by analysing multiple kinds of events in real-time.
- Benefit decision-makers of different verticals to make data-driven decision and take proactive action before the occurrence of an event.
- Enable smarter anomaly detection and faster responses to threats and opportunities.
- Help shield business people from data overload by eliminating irrelevant information and presenting only alerts and distilled versions of the most important information.

Event Processing (EP) is a paradigm where streams of events are analysed to extract useful insights of real-world events [5]. EP systems associate precise semantics with the information items being processed: these are notifications of events that happened in the external world and were observed by sources, also called event producers [6]. The EP engine is responsible for filtering and combining such notifications to understand what is happening in terms of higher-level events (aka complex events, composite events, or situations) to be notified to sinks, called event consumers. EP systems detect complex patterns of incoming items involving sequencing and ordering relationships. An example of such a situation is the flagging of a *suspicious account* that is detected whenever there are at least three events of large cash deposits within 10 days to the same account. Event processing is in essence a paradigm of reactive computing: a system observes the world and reacts to events as they occur. It is an evolutionary step from the paradigm of responsive computing, in which a system responds only to explicit service requests.

A vast number of recent applications of EP can be found in health informatics, astronomy, telecommunications, electric grids and energy, geography, and transportation [5]. In the DataBio project, event processing applications have been developed and deployed for the domains of agriculture and fisheries, as described in the pilots section. [See Parts V and VII of this book].

11.2 Market

The massive surge in data generation and the increasing demand for real-time analysis of streaming data are expected to boost the growth of the ESP market. According to the Event Stream Processing Market—Global Forecast to 2023 report from December 2018 [3], the global ESP market size is projected to reach USD 1.838 billion by 2023, growing at a compound annual growth rate (CAGR) of 21.6% during the forecast period. The market analysis by application in Europe shows that the predictive maintenance segment is expected to grow from USD 29.2 million in 2018 to USD 81.0 million by 2023, at the highest CAGR of 22.7% during the forecast period. The market analysis by verticals in Europe shows that the ESP market by vertical is expected to grow from USD 689.9 million in 2018 to USD 1838.0 million by 2023, at a CAGR of 21.6% during the forecast period. Furthermore, the market size of the banking, financial services, and insurance (BFSI) vertical is expected to have the largest market size and projected to grow from USD 37.6 million in 2018 to USD 95.8 million by 2023, at a CAGR of 20.6% during the forecast period. This can be attributed to the growing adoption of IoT-based connected devices. All the verticals are undergoing digital transformation, which has created the need for analysing real-time data to achieve a competitive advantage in the market.

Gartner [4] characterises ESP systems as transformational, meaning they have the potential to change the way organisations interact with information to such a degree that they have a demonstrable impact on organisations' business models. Three factors are driving the expansion of ESP:

- The growth of IoT and digital interactions is making event streams ubiquitous.
- Business is demanding continuous intelligence for better situation awareness and faster, more personalised decisions.
- Vendors are launching new products, many of them open source or partly open source, giving the impression of lower acquisition costs.

From the analysts' reports covered, it's clear that ESP solutions have the potential to enable new ways of doing business; companies who have not yet adopted such systems should consider doing so in the near future. Furthermore, the bioeconomy domains investigated in DataBio (i.e., agriculture, forestry, and fishery), and not mentioned in the reports so far, have a unique opportunity to be innovative by embracing this technology.

11.3 Technical Characteristics

Event processing systems are a departure from traditional computing architectures that employ synchronous, request-response interactions between client and servers. In reactive applications, decisions are driven by events. Conventional architectures are not fast or efficient enough for some applications, because they use a "save-and-process" paradigm in which incoming data is stored in databases in memory or on disk, and then queries are applied. When fast responses are critical, or the volume of incoming information is extremely high, application architects instead use a "process-first" EP paradigm; here, logic is applied continuously and immediately to the "data in motion" as it arrives. EP is more efficient because it computes incrementally, in contrast to conventional architectures that reprocess large datasets, often repeating the same retrievals and calculations as each new query is submitted.

As mentioned above, the goal of an EP engine is to notify its users immediately upon the detection of a pattern of interest. Data flows are seen as streams of events, some of which may be irrelevant for the user's purposes. Therefore, the main focus is on the efficient filtering out of irrelevant data and processing of the relevant. Obviously, for such systems to be acceptable, they must satisfy certain efficiency, fault tolerance, and accuracy constraints, such as low latency and robustness.

As previously stated, EP is a technique in which incoming data about what is happening (event data) is processed more or less as it arrives to generate higher-level, very useful summary information, known as complex events. Event processing platforms have built-in capabilities for filtering incoming data, storing windows of event data, computing aggregates, and detecting patterns. In essence, EP software is any computer program that can generate, read, discard, and perform calculations on events. A complex event is an abstraction of one or more raw input events. One complex event may be the result of calculations performed on a few or on millions of events from one or more event sources. A situation may be triggered by the observation of a single raw event but is more typically obtained by detecting a pattern over the flow of events. Many of these patterns are temporal in nature [7], but they

can also be spatial, spatio-temporal, or modal [1]. Event processing deals with these functions: get events from sources (event producers), route these events, filter them, normalise or otherwise transform them, aggregate them, detect patterns over multiple events, and transfer them as alerts to a human or as a trigger to an autonomous adaptation system (event consumers). An application or a complete definition set made up of these functions is also known as an Event Processing Network (EPN) [1].

Generally speaking, complex event processing (CEP) software offers two major components: a high-level language for programmers to easily describe how to process the incoming events and an infrastructure engine for the processing of the data streams in real-time. Events of different formats are gathered from different event producers. The event producers can be of different types, including financial feeds, news feeds, weather sensors, application logs, video streams collected from surveillance cameras, etc. The EP engine is the brain that carries out multiple types of processing on event streams, based on predefined rules. The processing includes simple filtering, counting, averaging, aggregating, of simple event processing operations, as well as more complex processing, such as pattern matching and event prediction (forecasting). Event consumers are parties that are interested in mining valuable information from the event streams, e.g., software agents, users of web/mobile applications, etc. [5].

The design of event processing applications includes the design of both the functional properties and the non-functional properties. While functional requirements define what an event processing system should do, non-functional requirements place constraints on how the system will do so. The design of requirements is implementation-specific and is carried out in either hand-coded fashion or using modern dedicated event processing tools by IT developers familiar with the event processing engine and the particular way to bypass the engine's limitations.

The event logic necessary to specify the event-driven application is typically provided by domain experts who know the domain and can express the event rules. However, the task of defining the event definitions can be tedious and difficult even for experts. To alleviate this task, in some engines the event definitions can be learned in an automated way using machine learning techniques (e.g., [8] and [9]).

Non-functional requirements include scalability, usability, availability, security, and performance objectives. Not all of these requirements apply equally to all applications, so when designing an event processing application, one needs to consider which of them are important for the case in hand. A survey in the area of non-functional requirements can be found in [10].

There is no standard for event processing languages and programming models. As a result, each event processing tool uses its own terminology and semantics. For example, the IBM PROactive Technology Online (PROTON) open source tool[1] applied in the DataBio project follows the semantics presented in Etzion's and Niblet's book [1].

[1] https://github.com/ishkin/Proton/.

11.4 Event Processing Tools

CEP has already built up significant momentum, manifested in a steady research community and a variety of commercial and open source products [6]. Today, a large variety of commercial and open source event processing tools are available to architects and developers who are building event processing applications. These are sometimes called event processing platforms, streaming analytics platforms, (complex) event processing systems, event stream processing systems, or distributed stream computing platforms (DSCPs). DSCPs such as Amazon Web Services Kinesis[2] and open source offerings including Apache Samza,[3] Spark,[4] and Storm[5] were introduced in recent years. In particular, Apache open source projects (Storm, Spark, and Samza) have gained a fair amount of attention and interest [11, 12].

Event processing systems are general purpose development and runtime tools that are used by developers to build custom, event-processing applications. The tools allow this to be done without having to re-implement the core algorithms for handling event streams, as they provide the necessary building blocks to build the event-driven applications. In comparison, DSCPs are general-purpose platforms without full native EP analytic functions and associated accessories. However, they are highly scalable and extensible, and usually offer an open programming model so developers can add the logic to address many kinds of stream processing applications, including some EP solutions. Today, there are already some implementations that take advantage of the pattern recognition capability of EP systems along with the scalability capabilities that DSCPs offer and provide a holistic architecture. For example, the PROTON open source event processing tool applied in the DataBio project has a Storm version (ProtonOnStorm), which allows PROTON's engine to run in a distributed manner on multiple machines using the Storm infrastructure.

A recent Gartner report from 2019 [4] states that more than 40 ESP products are available on the market.

Sample vendors include EsperTech, EVAM, IBM, Microsoft, Oracle, SAP, SAS, Software AG, the Apache Software Foundation, and TIBCO Software.

11.5 Experiences in DataBio

As mentioned previously, event-driven applications were developed for the agriculture and fisheries sectors in the DataBio project. More specifically, two agricultural implementations were developed. One focuses on monitoring temperature and air pressure measurements from sensors in the field (using SensLog) and sending warnings concerning a possible upcoming freeze. This application is designed to alert

[2] https://aws.amazon.com/kinesis/.

[3] https://samza.incubator.apache.org/.

[4] https://spark.apache.org/streaming/.

[5] https://storm.apache.org/.

farmers before the occurrence of freezing temperatures that can destroy crops. The second application monitors different crop parameters to predict disease and pest infestation in various types of crops and sends alerts and warnings if these are found. Crop parameters are gathered by GAIATrons and pushed to PROTON for further analysis of this data, where temporal analysis of trends is carried out to allow proactive measures.

In fisheries, PROTON monitors engine parameters to send alerts in real-time regarding potential engine problems before damage will be caused to the engine and therefore to the tuna fishing vessel. An event-driven application informs crew members to act in advance to avoid critical machinery faults prior to their occurrence. PROTON has been deployed on board the vessel and is integrated with the VTT OpenVA tool to visualise the alarms and warnings in real-time as they are detected by the event processing engine.

For detailed information on these implementations, refer to the relevant pilots sections in Part V and VII of this book.

These applications follow the event-driven paradigm and fit into the "Generic pipeline for IoT data real-time processing and decision making" articulated in the course of the project and presented in Deliverable 4.4 of the project [13]. This generic pipeline is an example of a pattern that fits the two aspects of generalisation. The main characteristic of this generic pipeline is the collection of real-time data coming from IoT devices to generate insights for operational decision-making, by applying real-time data analytics on the collected data.

Figure 11.1 depicts the common data flow among three pilots of the DataBio project: two in agriculture ("Prediction and real-time alerts of diseases and pests

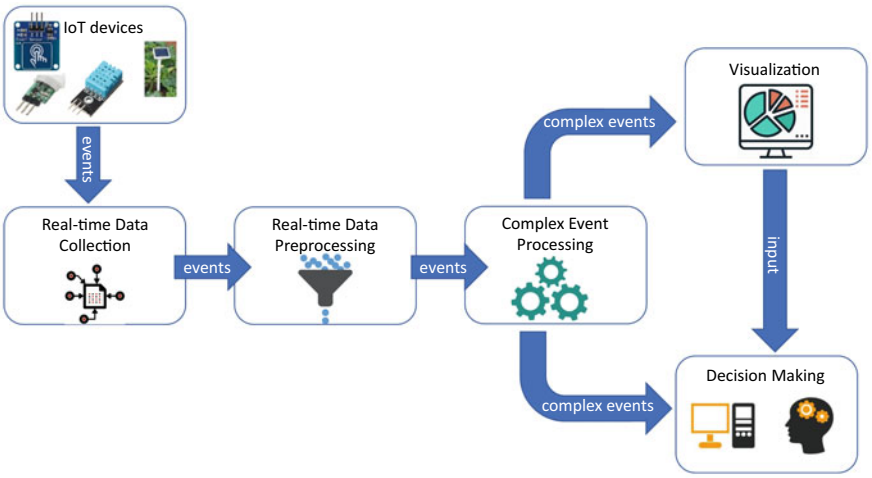

Fig. 11.1 Data flow for real-time IoT data processing and a decision-making generic pipeline

breakouts in crops" and "Cereals and biomass crop") and one in fisheries ("Monitoring, real-time alerts, and visualization for operation efficiency in tuna fishery vessels").

Streaming data from IoT sensors are collected in real-time, from sources such as agricultural sensors, machinery sensors, and fishing vessels' monitoring equipment. These streaming data (aka events) can then be pre-processed to lower the amount of data to be further analysed. Pre-processing can include filtering the data (filtering out irrelevant data and filtering in only relevant events); performing simple aggregation of the data; and storing the data (e.g., on the cloud or using other storage models, or even simply on a computer's file system) such that conditional notification on data updates to subscribers can be done. After being pre-processed, data enters the CEP component for further analysis, which generally means finding patterns in time windows (temporal reasoning) over the incoming data to form new, more complex events (aka as situations or alerts/warnings). These complex events are emitted to assist in the decision-making process that is carried out by humans ("human in the loop") or automatically by actuators (e.g., sensors that starts irrigation in a greenhouse following a certain alert). The situations can also be displayed using visualisation tools to assist humans in the decision-making process. The idea is that the detected situations can provide useful real-time insights for operational management, such as preventing possible pest infestations in crops or machinery failure.

Figure 11.1 shows the end-to-end flow. In essence, all components except the data producers (i.e., sensors) and a data consumer (either human or automatic) can be optional. The level of analysis of the data and its level of abstraction is driven by the specific use case. Sometimes, some filtering on the data is enough, while in other cases, the CEP component performs all types of analysis in a central manner. Communication between the software components is performed using standard RESTful APIs, while communication between IoT devices and the *Real-time data collection* component is based on standard IoT communication protocols (e.g., MQTT).

As mentioned above, the Generic pipeline for IoT data real-time processing and decision making is a generalization of three of the project's pilots, but it is also a specification of the top-level pipeline devised in the project as shown in Fig. 11.2 [13].

11.6 Conclusions

The major factors driving the growth of the ESP market are the increasing demand for IoT and smart devices, and the growing focus on drawing real-time insights to gain a competitive edge. IoT provides numerous opportunities for ESP vendors, such as real-time remote management, monitoring, and insights from connected devices, such as mobile phones or connected cars.

ESP is one of the key enablers of continuous intelligence and other aspects of digital business. It has transformed financial markets and become essential to smart electrical grids, location-based marketing, supply chain, fleet management, and other

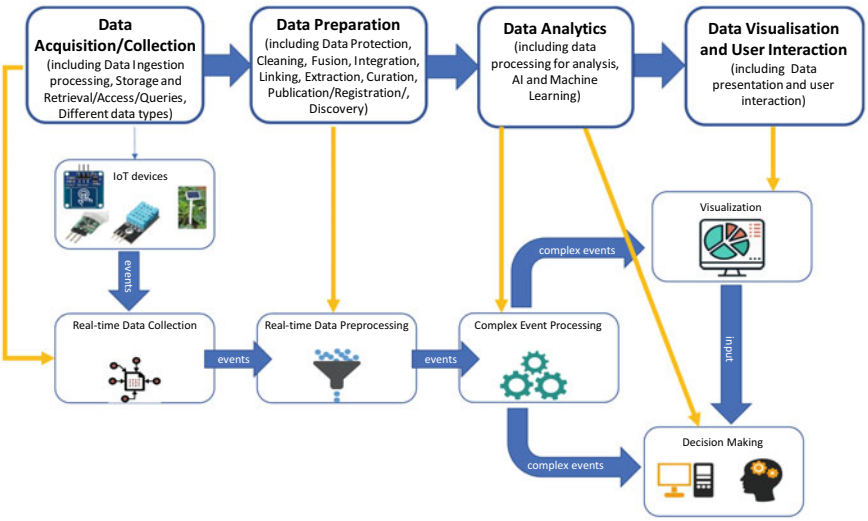

Fig. 11.2 Mapping of the steps of the top-level pipeline to the steps of the generic pipeline for data flow for real-time IoT data processing and decision-making

transportation operations. From the analysts' reports covered, we can conclude that ESP solutions can enable new ways of doing business; thus, companies who have not yet done so should consider adopting ESP systems. Furthermore, the bioeconomy domains investigated in DataBio (i.e., agriculture, forestry, and fishery) that are not mentioned in reports so far, have a unique opportunity to be innovative by embracing this technology. In DataBio, we have already paved the way for such applications by applying event-driven solutions in pilots in both the agriculture and fishery domains.

The generic pipeline for IoT data real-time processing and decision making has been applied to three pilots in the project from the agriculture and fishery domains and, as such, can be seen as a "pipeline design pattern". Conceptually, it can also be applied to other domains beyond fisheries and agriculture. Basically, use cases from any domain in which data is collected from IoT sensors and analysed in real-time to provide real-time alerts for operational decision-making can be adapted to this generic pipeline.

For example, sensor readings from a supply chain scenario in which objects are monitored for tracking and tracing can be collected for further processing by a CEP engine to detect potential delays. The detected situations can be displayed to operators so they can take action if such delays are detected (e.g., reschedule trajectory). Another use case can be found in a classical manufacturing process, in which machinery sensors are monitored to detect potential failures. The sensor data in the factory can be collected and transmitted to a CEP engine, which can detect potential failure situations and emit alerts to aid in decision-making (e.g., stop the machine, replace a part, etc.).

References

1. Etzion, O., & Niblett, P. (2010). *Event processing in action.* Manning Publications Company.
2. Watts S. (2018). *What is stream processing? Event processing explained*, [Online]. At: https://www.bmc.com/blogs/event-stream-processing
3. Sreedhar, B., & Naim, S. (2018). *Event stream processing market—Global forecast to 2023.* A Markets and Markets Report (published December 2018).
4. Hare, J., & Schlegel, K. (2019). *Hype cycle for analytics and business intelligence, 2019.* Gartner report # G00369713 (published on 18 July 2019).
5. Dayarathna, M., & Perera, S. (2017). Recent advancements in event processing. *ACM Computing Surveys*, *1*(1). https://doi.acm.org/10.1145/3170432
6. Cugola, G., & Margara, A. (2012). Processing flows of information: From data stream to complex event processing. *ACM Computing Surveys*, *44*(3).
7. Etzion, O. (2010). Temporal aspects of event processing. Handbook of distributed event based system.
8. Margara, A., Cugola, G., & Tamburrelli, G (2014). Learning from the past: Automated rule generation for complex event processing. In *Proceedings of the 8th ACM international conference on distributed event-based systems (DEBS2014)*, (pp 47–58).
9. Artikis, A., Sergot, M., & Paliouras, G. (2014). An event calculus for event recognition. *IEEE Transactions on Knowledge and Data Engineering (TKDE).*
10. Etzion, O., Rabinovich, E, & Skarbovsky, I. (2011). Non-functional properties of event processing. In *Proceedings of the Fifth ACM international conference on distributed event-based systems (DEBS 2011)* (pp. 365–366).
11. Biscotti, F. et al. (2014). *Market guide for event stream processing*. Gartner report #G00263080 (published on August 14, 2014).
12. Vincent, P. (2014). *CEP tooling market survey*. December 3, 2014, https://www.complexevents.com/2014/12/03/cep-tooling-market-survey-2014/. Accessed September 25, 2020.
13. Plakia, M. et al. (2020). *D4.4—Service documentation*, [Online]. At: file:///C:/Data/DataBio/WP4/D4.4/DataBio_D4_4ServiceDocumentation_v1_2_2020_03_13_EXUS.pdf.