University of Namur

Initiation à la démarche scientifique

IHDCB339

State of the Art: Complex Event Processing for Internet of Things

Author
Kenny Warszawski

Supervisor
Moussa Amrani
Pierre-Yves Schobbens

July 28, 2019

1 Introduction

The Internet of Things is omnipresent and the amount of connected devices is increasing day by day. The Internet of Things is different from a classic information system by its capability for integrating with the physical world. Those devices are used in plenty of sectors (health, industry, transport, home automation, ...) and their users can be both professionals and individuals. Nowadays, we can see the apparition of connected devices for home to facilitate our daily life. (e.g: Google Home, Nest, Philips Hue) IoT is also an interesting technology for Smart Cities use case. We can imagine a city where traffic lights are optimised with the city mobility to avoid traffic jams for example. However, an important problem arises in this type of architecture. How is it possible to handle such an important data traffic efficiently ? Indeed, if an entire city has a huge amount of connected objects, the data flow to be processed is massive. Therefore efficient data processing mechanisms need to be put in place to handle such flows.

Ajouter du texte, pas assez long Parler des CPS, Fraud Detection, Health, Block chain ?

2 Context

2.1 Internet of Things

The Internet of Things(IoT) is a novel paradigm that is rapidly gaining ground in the scenario of modern wireless telecommunications. The basic idea of this concept is the pervasive presence around us of a variety of things or objects – such as Radio-Frequency IDentification (RFID) tags, sensors, actuators, mobile phones, etc. – which,through unique addressing schemes, are able to interact with each other and cooperate with their neighbors to reach common goals. [1]

The phrase "Internet of Things" was mentioned the first time by Kevin Ashton, the co-founder of the Auto-ID Center at MIT, as the title of one of his presentation. For him, today computers and the internet more generally is dependent from human intervention. The information technology architecture are all designed around the hardware, software interactions, etc but an important thing is generally set aside: people. People have a limited time, attention and accuracy. Hence they are making mistakes about the real world. Thus, the data generated by humans isn't the exact representation of the reality in an Information System.

A thing can be any connected device from our daily life that is able to capture information from the physical world by sensors and send it through a network. This information can be kept by another layer able to process the data and make it meaningful by an application.

2.2 IoT architecture

The architecture of an Internet of Things solution can be implemented with various technologies and in very different infrastructures. However, a generic high level architecture is common to all IoT projects:

Perception Layer: The perception layer or device layer is the layer where physical objects will be able to capture data via sensors. These sensors are able to collect real world data. These data can be or not be processed upstream to add meta-data by the connected object itself and will be forwarded to the Network Layer.

Network Layer: The Network Layer or Transmission Layer. This layer transfers the informations from to Perception Layer to the Middleware Layer. This transfer can be accomplished by different communication technologies like Wifi, 3g, Bluetooth, etc.

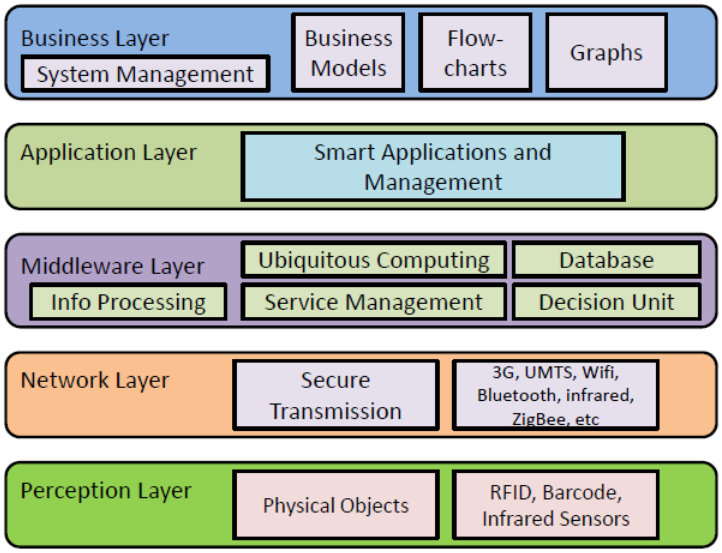

Figure 1: Generic Architecture

Middleware Layer: The Middleware Layer is the common layer for all physical objects of an IoT architecture. The middleware is responsible to take automatic decisions based on informations coming for the devices and store the results in a database.

Application and Business Layer: The Application Layer is where smart applications are running to produce complex processes. These applications will consume data from connected objects previously processed by the middleware to act effectively on a specific situation. These application can transmit actions to execute on a particular type of service. The Business Layer is important for the global management of the system. This one is rather a layer of global monitoring which produces graphs and statistics at the business level.

2.3 Data Flow

The previous architecture can be simplified to highlight the layer of architecture on which this state of the art will focus.

This state of the art is focusing on the "Computation and Processing System" layer because at that stage, the data processing is the most critical and the data flow is the highest. Obviously, the Communication Layer must

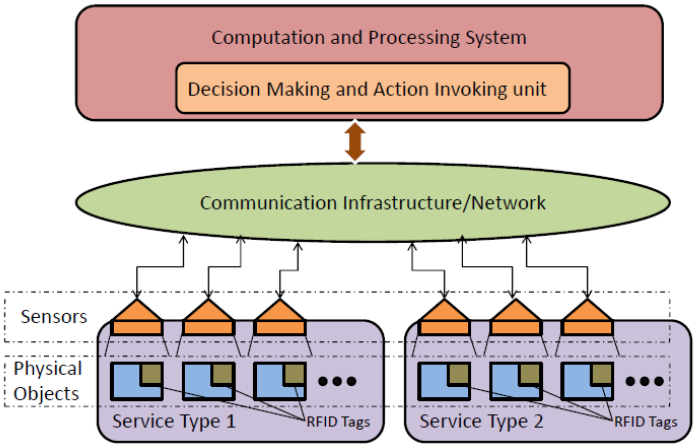

Figure 2: Simple IoT Architecture

be robust enough to transmit the data efficiently but this topic will not be tackled in this article.

3 Database and Data Stream Management Systems

Real-time applications are consuming the data in a different way comparing to a traditional system. On a traditional information system using a Database Management Systems (DBMS), the information stored in the database related to an specific entity is a "static" data persisted. The information stored in an aggregation of the reality. Only the current state of the information is relevant for the system. This type of Data management doesn't fit all real-time applications' needs. A real-time application needs to correlate each events coming into the system. Unfortunately, an IoT solution is sending too much data into the system. It's not possible for a traditional database to store, query and respond in a timely manner.

The Data Stream Management Systems are designed to handle potentially infinite and fast changing event streams. Obviously, it is not possible to query the all history of events that came from the beginning of the solution and respond in milliseconds. Different approaches exist to limit the amount of data processed. Roughly, compression techniques exists to summarize a

set of data already queried. Another technique consists of setting a time window to pick a portion of data in the time.

4 Complex Event Processing

4.1 Definition

Complex Event Processing (CEP) is a set of methods and techniques for tracking and analysing real-time streams of informations and detecting patterns or correlations of unrelated data (complex events) that are of interest to a particular business. [5] The complex event processing is the engine of real-time applications that needs to have real-time informations from an environment to respond as fast as possible. A standard Request/Reply with synchronous processes cannot correlate different type of events and respond in a timely manner. Complex Event Processing mechanisms are often used in Event Driven architectures. This architecture fits with the Internet of Things needs. This kind of architecture is driven by events sent by connected objects. Then, the services at the application layer are consuming those events.

4.2 CEP Engine

The CEP engine is consuming the events sent by the physical objects on the Communication Layer. This engine can define a bunch of rules where each event received will be evaluated. A rule represents a complex condition that can be based on multiple events and on a time window. For example, if all connected cameras in a house are sending that there aren't any move in the house in a time window of 60 seconds, an event saying that nobody is in the house will be sent. Then, some specific actions can be taken by connected devices in the house.

The technologies implementing CEP concepts are querying Time Series Databases where each event received are stored. Thoses databases are optimised for queries based on a time window. In a lot of CEP framework like Apache Flink, those CEP rules are describe like an SQL Query and uses Event Pattern Languages.

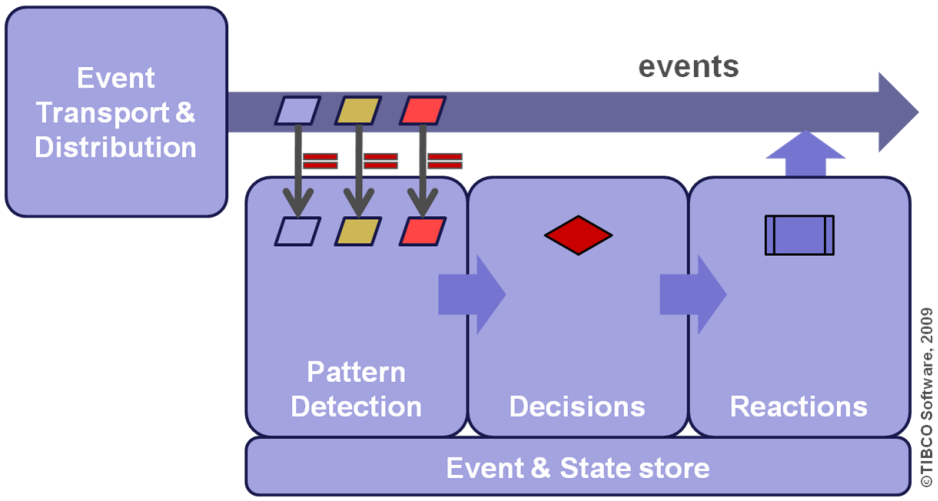

Figure 3: CEP Pattern Decision Reaction

4.3 Event Pattern Languages

An Event Pattern Language (EPL) is a language used by CEP engines in order to describe the relations between events that match a specific pattern. The syntax of EPL's are quite similar to SQL queries. The Example 1 is retrieving dates where the temperature was bigger than 30 degrees Celsius.

Example 1:

```
SELECT Thermometer.date
FROM DataSource
WHERE temperature > 30
```

A plenty of primitive operators exists in Event Pattern Languages to describe the rule's behaviour. The following operators are coming from ThingML framework but are available in most of CEP frameworks:

Selection: Filters relevant events based on the values of their attributes.[2] For example, we can select all Air Pressure events between 101300 Pa and 101400 Pa.
Projection: Extracts or transforms a subset of attributes of the events.[2]
Window: Defines which portions of the input events to be considered for detecting pattern.[2] For example, a rule can concern the last 30 seconds events.

Conjunction: Consider the occurrences of two or more events.[2] This operator is basically a logical 'AND' operator.
Disjunction: Consider the occurrences of either one ore more events in a predefined set.[2] This operator is basically a logical 'OR' operator.
Sequence: Introduces ordering relations among events of a pattern which is satisfied when all the events have been detected in the specified order.[2]
Repetition: Considers a number of occurrences of a particular event.[2]
Aggregation: Introduces constraints involving some aggregated attribute values.[2] For example, the average temperature of all Weather events is an aggregation.
Negation: Prescribes the absence of certain events.[2] This operator is basically a logical 'NOT' operator.

5 Computing Levels

The main preoccupations of a real-time solution in IoT is to have the lowest latency between all devices. To process all data coming from sensors, CEP engines can help reducing the processing time. The CEP engines can be placed at different level of the IoT architecture. The place where Complex Event Processing is used depends of the business needs.

5.1 Cloud Computing

The Cloud Computing places the computing complexity at the cloud level. All the data coming from the sensors have to transit by the connected object, sometimes by a middleware and then it can be consumed by a CEP Engine in the cloud. The Cloud Computing is common for near real-time applications where a little bit of latency is not critical.

This solution is optimal for the near real-time cloud monitoring because all events coming from the devices are sent to the cloud. Then a precise metrics history can be retrieved. Obviously all data persisted have a cost. This type of solution is extremely demanding in storage space if all the history is persisted.

Of course the Cloud Computing can be coupled with other levels of computing and it's often the case. The Cloud Computing can be placed on the top of a multi site IoT solutions where millions of data are coming from thousands of connected objects in the world. Then this solution can reflect a High Level overview of a global IoT system and available all over the world.

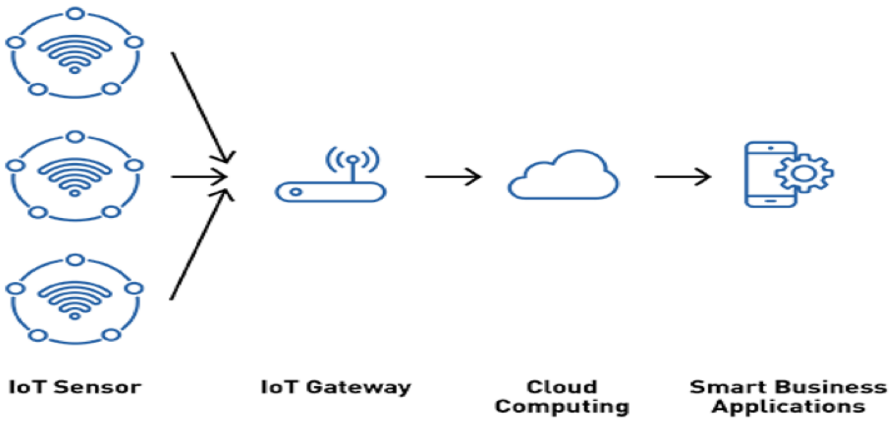

Figure 4: Cloud Computing

5.2 Fog Computing

The Fog Computing places the computing intelligence at the local network level. Typically the complexity is put between the Cloud and the IoT sensors. The devices are sending their data to a Middleware and the CEP engine is processing the data locally and send the processed events to a service running in the Cloud or on a local server. This method offers the CEP performances on a closer level where the data are created. Then the applications where latency is crucial, the Fog Computing can fit with their business needs. For example, the auto mobile or aerospace sector are more and more connected and the reactivity of their system needs to be as fast as possible.

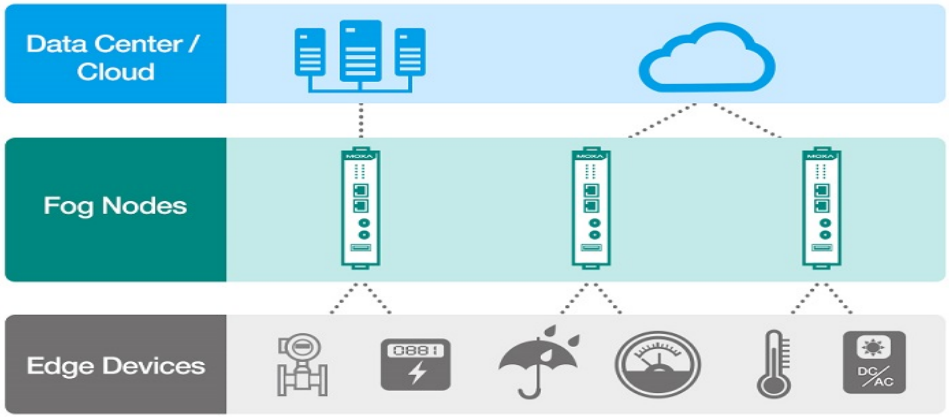

Figure 5: Fog Computing

It can happen that the edge devices are sending too much data and very fast. Then the fog computing can still respond too slowly because of the massive data reception. The CEP philosophy can be put at the lowest level with Edge Computing.

5.3 Edge Computing

Edge computing refers to the enabling technologies allowing computation to be performed at the edge of the network,on downstream data on behalf of cloud services and upstream data on behalf of IoT services. Here we define "edge" as any computing and network resources along the path between data sources and cloud data centers.[**Edge Computing - Vision and Challenges**]

The Edge Computing places the computing complexity at the Physical Layer. This method reduces the network traffic to the Fog Nodes and the Cloud because less informations will be transferred. Therefore it reduces the risk of data congestion and the CEP engines are not flooded. Edge Computing is generally used by industries in Cyber Physical Systems[1].

[1]Cyber-Physical Systems (CPS) are integrations of computation and physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa [3]

6 Publish-Subscribe Architecture

7 IoT Platform extension

8 Problematic

9 Research Methodology

10 Conclusions

List of Figures

1 Generic Architecture 3
2 Simple IoT Architecture 4
3 CEP Pattern Decision Reaction 6
4 Cloud Computing 8
5 Fog Computing 9

References

[1] D. Giusto et al. *The Internet of Things*. Springer, 2010.

[2] An Lam and Øystein Haugen. "Complex Event Processing in ThingML". In: vol. 9959. Oct. 2016, pp. 20–35. ISBN: 978-3-319-46612-5. DOI: 10.1007/978-3-319-46613-2_2.

[3] Edward A. Lee. "Cyber Physical Systems: Design Challenges". In: (Jan. 2008).

[4] D. Robins. "Second International Workshop on Education Technology and Computer Science". In: 2010.

[5] I. Schmerken. *Deciphering the myths around complex event processing*. Wall Street & Technology, May 2008.

[6] Wu Y et al. "RFID enabled traceability networks: A survey". In: (2011).