Towards IoT Analytics. A vf-OS Approach

Víctor Anaya[1], Francisco Fraile[2]
CIGIP
Universitat Politècnica de València
Valencia, Spain
{[1]vanaya, [2]ffraile}@cigip.upv.es

Armando Aguayo
ICE - Information Catalyst
Manchester, UK
armando.aguayo@informationcatalyst.com

Oscar García
ICE - Information Catalyst
Valencia, Spain
oscar.garcia@informationcatalyst.com

Ángel Ortiz
CIGIP
Universitat Politècnica de València
Valencia (Spain)
aortiz@cigip.upv.es

***Abstract*—New challenges faced by the current Industrial Internet of Things (IIoT) are stringent latency, capacity constraints, uninterrupted devices with intermittent connectivity cannot be solved with centralized cloud computing architecture [1]. Sending all the data to the cloud will require prohibitively high network bandwith [2]. vf-OS – Virtual Factory Open Operating System – is a project funded by the H2020 Framework Programme of the European Commission under Grant Agreement 723710, with the purpose to provide an Open Operating System for Virtual Factories composed of a kernel, application programming interface, and middleware specifically designed for the factory of the future. This paper presents the Virtual Open Operating System (vf-OS) approach to IoT Analytics, and how vf-OS IO (Input/Output) components and vf-OS analytics can be used to capture data from sensors and Program Logic Controllers (PLCs) to generate and run machine learning models to do analytics in the cloud and in the edge. The main contribution of the paper is the proposal of new architectural solutions and providing alternative scenarios for developers developing applications for the manufacturing sector.**

Keywords—driver, IoT, Edge Computing, Analytics, IIoT

I. INTRODUCTION

Nowadays, there is an increasing rise of technologies empowering the industry 4.0, such as sensors, networking technologies based on optimised communication protocols, or ad hoc specific and secured computation devices addressing the manufacturing industry. As the price of those technologies is lower and lower, the amount of sensorization and therefore the amount of information captured is bigger, with some reports forecasting the raise from 22.0 billion sensing devices in 2016 to 50 billion by 2020 [3]. The capture of that information comes with the tradeoff of extracting value from it. The need to make decisions in real time make the situation worse, meaning that information is captured with a millisecond frequency, and though computing and storage facilities are easier to access than ever before, the load and complexity of information gathering, storage and analysis is significant. Computing information in the cloud or in powerful datacenters was a first approach. In the other hand, network latency associated with the urge to real-time decisions, and security concerns still raised when sharing private information regarding processes while manufacturing is pushing edge computing as a complementary solution in some scenarios. According to IDC news, the 45% of IoT-created data will be stored, processed, analyzed on the edge of the network by 2019 [4].

Edge computing and edge analytics are shown to be relevant in these scenarios, where some computation, storage and networking assets are moved to the edge, that is, on premises or even at the same computing device as sensors. Benefits associated to edge computing are lower latency and response time, minimizing upward and downward network traffic, reducing load on cloud, data center computation offloading and energy efficiency [5].

Edge computing technologies diverge in approach and scope. The four most well-known technologies approaches are:

- Fog: Fog computing brings cloud computing to the proximity of end users. Introduced by Cisco, with the analogy that real-life fog is closer to the earth than clouds. Fog deploys a virtualized fog platform closer to end users. Used in applications that require low-latency and benefit from the closeness of fog computation, storage and networking infrastructure.
- Cloudets: as fog, cloudlet provide a third-tier nearer the users, with the difference that the main purpose is the provisioning of computational power to mobile devices.
- Micro datacenters: introduced by Microsoft, micro datacenters are an extension of today's hyper-scale cloud data centers. They are self-contained secure computing environments including computation, storage and networking equipment to run customer applications. Micro datacenters are applicable when applications require real-time data processing.
- Mobile edge computing (MEC): MEC brings cloud computing capabilities to the edge of cellular networks. MEC servers process request from mobiles and process them or filter petitions to the cloud.

Complementarily, analytics, in the context of this paper, is the application of machine learning algorithms based on statistical analysis that, based on predictive modelling, forecast the results facilitating the decision making.

The goal of this paper is to explain how the Virtual Open Operating System (vf-OS) addresses IoT analytics on the edge. vf-OS is an operating system designed for manufacturing

978-1- 5386-7097-2/18/$31.00 @2018 IEEE

companies to leverage enterprise assets and resources, reducing the complexity for generic application developers to generate industrial-robust and secure manufacturing and logistic applications. vf-OS ecosystem (see Fig. 1) is composed of the main platform, made-up of a stack of modularized and extensible components that could be plugged into the operating system core, and a marketplace where vApps (applications developed using the services and easy to use tools) and vf-OS assets.

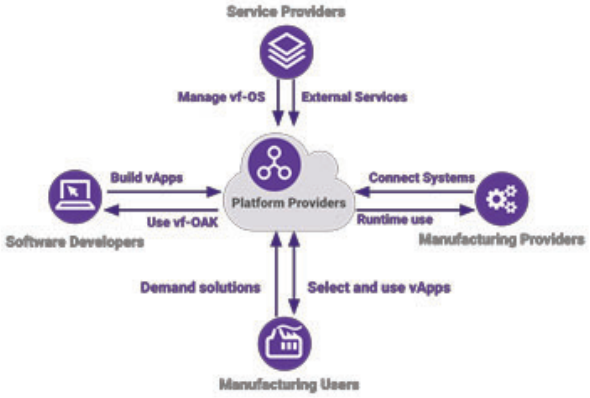

Fig 1. vf-OS ecosystem

The main vf-OS platform can be deployed on-premises or in the cloud, in the simpler scenarios, but as the different modules composing the vf-OS platform are self-contained, REST accessible software components, other deploying alternatives are applicable.

II. VF-OS IoT Approach

vf-OS platform is a modular-oriented operating system, where some of the modules, as the vf-OS OS kernel, are mandatory, and other are optional. The vf-OS IO (Input/Output) module is the set of components in the vf-OS architecture addressing the input and output connections of vf-OS platform to its environment. Among them, the vf-OS Driver component provides a collection of reference implementations to collect data from, and send commands to, physical devices such as industrial automation devices: PLCs, smart sensors, RFID readers, etc. Drivers hide the complexity of communicating to physical devices from the rest of the vf-OS components

Fig 2 shows the high-level internal structure diagram of a Driver component, highlighting the main modules, their interconnections and the connection with other Platform components.

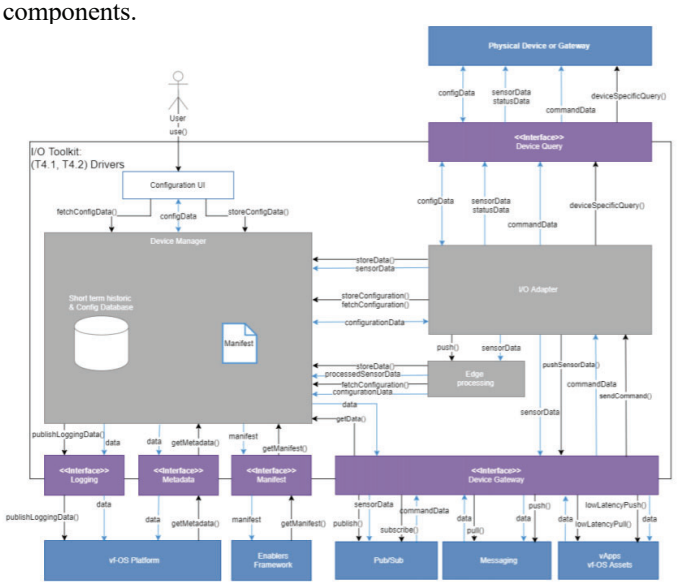

Fig 2. Driver component architecture

The vf-OS Driver components consist of the following main modules:

- Device Manager: This module manages the Device data (configuration data, services end-point metadata, manifest file, etc). Thus, it is a repository of all the data used by other modules and interfaces. It allows storing and fetching configuration data and sensor data in a short-term historic, getting the driver metadata and the manifest file.
- I/O Adapter: This module manages read/write operations with the devices, encapsulating device specific queries and managing sensor and actuator command data, configuration data and status data. The I/O Adapter module connects to the Device Manager to download the configuration data to the device, to update the configuration data with the current configuration of the device and to save short-term sensor data for historic processing. The Device interface can also pull sensor or status data on demand from the physical device via the I/O adapter. Optionally, the I/O Adapter may push data to the Edge Processing module and to the Virtual Device interface.
- Edge Processing: The Edge Processing module allows to optionally implement low-level data processing close to the device. This allows the implementation of basic functions (eg filtering, manipulation, transformation, or statistics) to optimise data communications with physical devices the rest of the vf-OS components. The Edge processing module is configured through the Configuration Interface, which allows setting arithmetic or statistic functions to be applied to the sensor data.

The following sections explain the technology implementation of the Driver component, and the Driver Data Model with an example.

This work was supported in part by the European Commission under the Grant Agreement 723710

A. *vf-OS IO Driver implementation*

The vf-OS IO components are nodejs loopback.io applications. Loopback.io [6] is an open-source Node.js framework for creating APIs and connecting them with backend data sources. Loopback is built on top of Express and can generate a functional end-to-end REST APIs from a data model definition. The data model definition in the vf-OS Driver component was done along with public contract API, publicly accessible at [7], through a Swagger [8] specification in OpenAPI [9]. The client interface used to manage graphically the drivers (see Fig 3) is developed using reactjs [10].

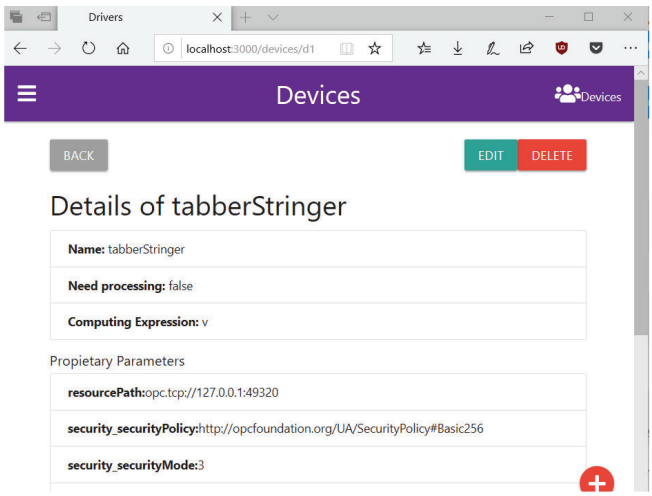

Fig 3. Driver Manager UI

The stack of technologies used in the component is all nodejs compliant making each driver component easily deployable in IoT devices such as Raspberry PIs and easy deployable as Docker images.

B. *vf-OS IO Driver data model*

The vf-OS proposal considers that devices are composed of sensors, and sensors are connected to controllers (PLCs). When reading data from sensors information can be lightly processed. Once processed, the information is stored, if accordingly configured, on-premises. Before pushing the information to the vf-OS platform.

Fig 4 depicts the configuration of a device using the IO component. The device is configured pointing to an OPC UA server and has four sensors associated. Sensor 1 is detailed in the figure, showing that after reading the value of the sensor with the nodeId specified, the value is doubled before sending the result. The device is configured to gather information from sensors 1, 2 and 4 and push them to a regression algorithm loading an already trained model stored. The result of the analytics is pushed to the vf-OS messaging component to the vf-OS platform, where data can be stored, aggregated, mapped, analysed, etc.

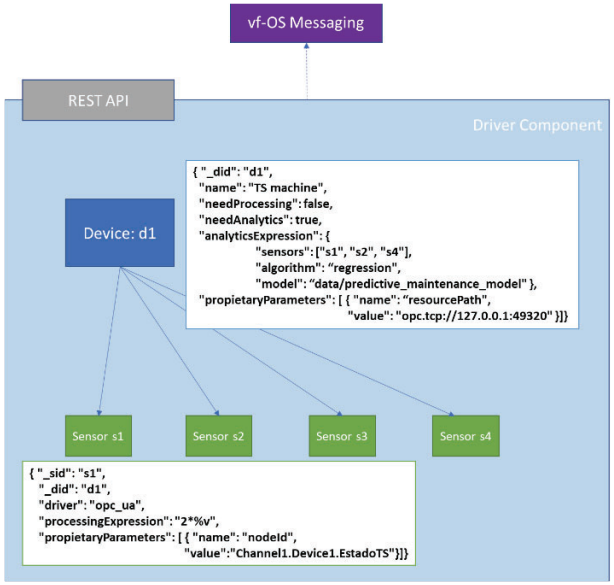

Fig 4. Driver Component partial model example

III. VF-OS ANALYTICS APPROACH

A. *vf-OS Analytics*

vf-OS is composed of a set of modules one of them being the Data Management containing a Data Analytics module. The Data Analytics module core is based on H2O.ai . H2O combines the power of highly advanced algorithms, the freedom of open source, and scalable in-memory processing for Big Data on one or many nodes.

The Data Analytics purpose is to provide business analytic functionalities to derive events from stream and historic process data within the manufacturing domain.

The Data Analytics component workflow is illustrated in Fig 5, where the process starts with the connection plugins enabling access to the data sources to transform the data into one or more data frames. Once the data is on its data frames form, the execution of the selected ML algorithm takes place in an iterative way until obtaining a valid model (algorithm training). The final model could be exported as a POJO/MOJO file and utilised by the user (Developer) to build its own application.

Data Analytics H2O solution is a Java-based solution that provided with a powerful execution environment can scale to a huge amount of data tackling through its integration with Apache Spark and Hadoop. The vf-OS Analytics component is a centralized data analytics solution, meaning high latency and high computing resources available to run on big volume of data.

Data Analytics H2O includes machine learning algorithms supporting supervised and unsupervised scenarios. The core of the analytic algorithms is based on the combination of modern statistical-machine-learning and linear-algebra based algorithms (e.g. SVM, CRF, LDA, Mixture-Models), and traditional data-mining algorithms (e.g. decision trees and rules, K-Means, association rules, etc.).

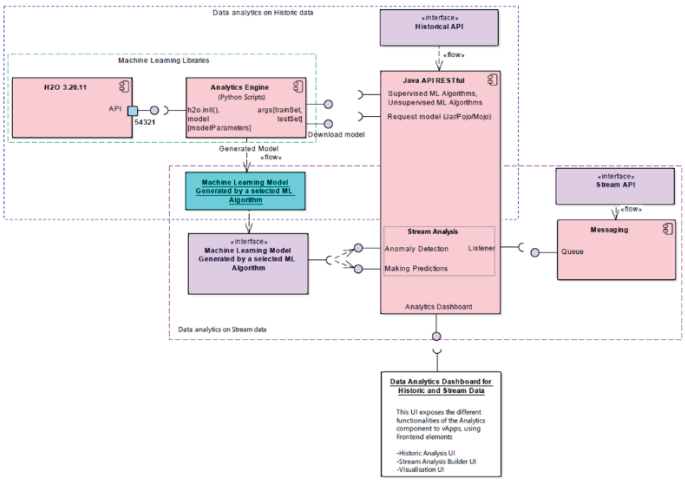

Fig 5. vf-OS Data Analytics component

1) Historic Data Analytics

The Historic Data Analytics execute an off-line analytical processing of sensor inputs, this will include machine learning algorithms supporting supervised and unsupervised scenarios. The core of the analytic algorithms will be based on the combination of the modern statistical-machine-learning linear-algebra based algorithms (e.g. SVM, CRF, LDA, Mixture-Models) and traditional data-mining algorithms (e.g. decision trees and rules, k-means, association rules), the Historic analytics generates a Model which can be used for the developer as ML libraries for vfApps.

The Historic Data Analytics process starts by either the API RESTful or Data Analytics Dashboard which executes an off-line analytical processing of sensor inputs, The user defines the training and testing sets, choose a Machine learning algorithms (supporting supervised and unsupervised scenarios) and provides its execution parameters, this information is used by the Analytics Engine and the H2O ML modules to process the Analytics queries to train and validate a ML model , this model can be downloaded to be used as POJO/MOJO files.

2) Stream Data Analytics

Sensor stream analytics is a research area which deals with the processing continuous streams of on-line sensor data. The result of the processing is provided in a form that is useful results for decision makers and the corresponding software systems which have the ability to act on the provided analytics. The tasks required for processing streaming data usually include: Detecting anomalies from the stream such as outliers [11]; Prediction of the future evolution of the data, including predictive maintenance of machinery and other types of prediction in systems control [12]; Interpretation of data with techniques such as decision trees and rules from data as well as various forms of visualisations[13]. The detection of complex events from heterogeneous data streams (for example, sensorial data used for a factory-wide monitoring systems) is becoming a promising area to support applications for monitoring, detection, and online responses[14].

In relation to the Stream Data Analytics, based on the Generated Model and/or a set of rules, this process is able to listen to a Queue in the Messaging Module and executes the Stream Analysis process which usually includes: Detecting anomalies from the stream such as outliers; Prediction of the future evolution of the data, including predictive maintenance of machinery and other types of prediction in systems control and Monitoring to generated alarms related to the System state and some kinds of Information Visualisation by charts.

B. vf-OS IoT Analytics

IoT Analytics is the use of machine learning algorithms to generate results from data capture from one or more devices or sensors that can be locally distributed. Specifically, IoT Analytics approach within vf-OS, means local, low demanding computation on data, with machine learning models generated potentially on smaller sets of data vf-OS. With the purpose to provide integrated lightweight analytics to devices and sensors, the vf-OS IO component considers the configuration of analytics expressions at a device level.

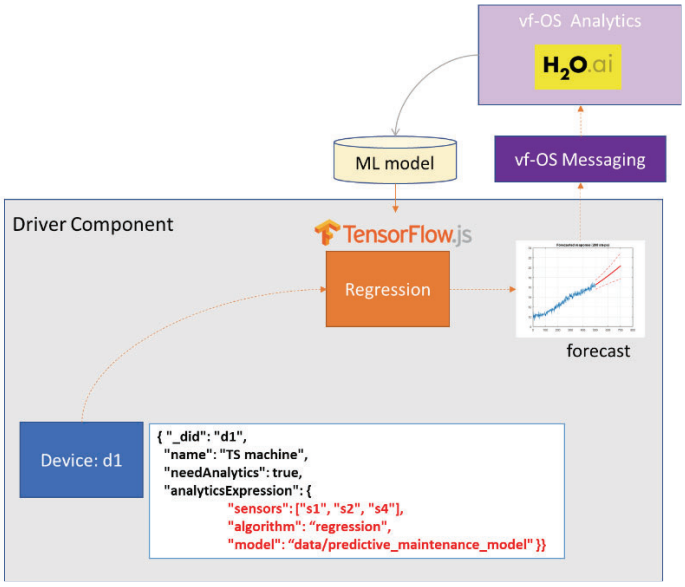

Fig 6. vf-OS IoT Analytics

IoT analytics is a conceived within vf-OS as a lightweight application of basic machine learning algorithms Edge computing pushes the computation of feasible algorithms to the edge of the network, near the devices and sensors, reducing the network overload and latency, and at the same time assuring that aggregated, processed and sometimes anonymised data is sent to the cloud.

vf-OS Driver Edge Analytics is based on tensorflow javascript. TensorFlow.js [15] is an open source WebGL-accelerated JavaScript library for machine intelligence. Tensorflow.js is compliant with the stack of javascript and nodejs technologies of the driver, and run on the same environment, saving latency and network overload.

The initial vf-OS IoT analytics approach is to provide a couple of algorithms out of the box for data sensor analysis, such as a feed-forward neural network for classification problems and value prediction. This paper doesn't cover the training of models that could be done at a driver level or with vf-OS analytics and consumed by tensorflow after migrating the model.

C. *vf-OS Analytics vs vf-OS IoT Edge Analytics*

Following, Table I summarizing the scope of the the two analytic approaches within vf-OS. Summing up, vf-OS IoT analytics purpose is to reduce the amount of information sent to the cloud, avoid network latency time and increase the response time to limited time-consuming algorithm while keeping locally private data. On the other hand, vf-OS Analytics provides a centric analytics hub where multiple devices, multiple systems from potentially disperse locations and companies profit from powerful computation resources and storage to analyze and produce decisions.

TABLE I. VF-OS ANALYTICS VS VF-OS IO EDGE ANALYTICS

	Vf-OS Analytics	**vf-OS IoT analytics**
Usage of computational resources	Massive parallel data processing	PCs or mobile powered processing
Amount of data	High/medium	Low
Running in the same environment	No	Yes
Scope	Strategic	Operational

The conclusion from the comparison between these two approaches is their complementarity in different scenarios.

D. *vf-OS platform Edge deployment*

vf-OS drivers are self-contained loopback / nodejs applications that can be run in docker environment independently from other vf-OS components at edge computing hardware technologies. The vf-OS Data Analytics component is a java server machine runnable as a docker that can be deployed in simple scenarios, at the cloud, along with other vf-OS platform components such as vf-OS data storage vf-OS data mapping and vf-OS messaging component (see Fig 7). vApps plugged into the vf-OS platform will benefit the services offered by this synergy among components.

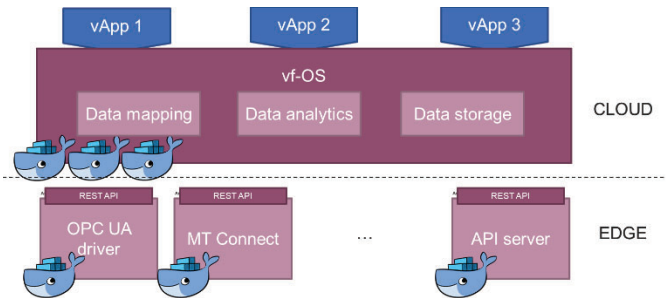

Fig 7. vf-OS platform and IO architecture deployment

The base-connecting component between vf-OS modules is the vf-Os messaging component based on a RabbitMQ implementation [16] and deployable as a docker machine. vf-OS components connect to it through Advanced Message Queuing Protocol [17] using a client library offered.

IV. PREDICTIVE MAINTENANCE

Although the vf-OS IoT proposal is generic and addresses any manufacturing company, one scenario to validate our proposal has been taken from our pilot use case with Mondragon Assembly. Mondragon Assembly designs and manufactures systems and equipment for process automation for different sectors such as automotive, medical, household, electrical appliances, and also cutting-edge photovoltaic (PV) modules. One of the vApps to be developed in vf-OS is vfFailurePrevention. vfFailurePrevention uses predictive maintenance to anticipate failure events in the Tabber Stringer (TS) machine. The vApp will send notifications to maintenance managers at both MASS and VS when failures are likely to happen and they use this information to plan equipment maintenance operations more efficiently.

Predictive maintenance is relevant in asset-intensive industries such as manufacturing and logistics with characteristics such as:

- Cost to production downtimes.
- High urgency repairs.
- Investment in new machinery and parts.

Predictive Maintenance has already been addressed by the literature [18] and a range of alternative proposal and the application of different algorithms exists. From a data analytics point of view, predictive maintenance data is time series data. Data is a time series if it is a sequence of quantitative observations about a system or process and made at successive points in time [19]. Commonly, the points in time are equally spaced. Examples of time series data include gross domestic product, sales volumes, stock prices, weather attribute. Datasets include a timestamp, a set of sensor readings collected at the same time as timestamps, and device identifiers. The goal of predictive maintenance is to predict at a given time, whether the equipment will fail in the near future. Some of the machine learning approaches to solving predictive maintenance are:

- Regression: Predict the Remaining Useful Life (RUL), or Time to Failure (TTF). Li [20] uses deep convolution neural networks for that purpose.
- Binary classification: Predict if an asset will fail within a certain time frame (e.g. next 3 days).
- Multi-class classification: Predict if an asset will fail in different time windows.

vf-OS IoT Analytics offers a binary classification algorithm offering the prediction to fail in the next 3 days. The driver will pack the sensors measures in a tensor, the data structure in tensorflow, and will call tensorflow neural network algorithm already trained. The dataset to train the algorithm will have the structure shown in Table II.

TABLE II. DATASET STRUCTURE

timestamp	s1	s2	…	Sn
2018-07:30 19:00:20	105.12	true	1.004	27.12

V. Conclusions

The Industrial Internet of Things (IIoT) is characterized by an increasing mechanism of monitoring operational processes in the search of complex patterns to improve performance. The amount of data gathered by sensors is exponential compared to production data, that, together with security concerns [21] make edge computing an increasing trend in the industry. This paper has presented vf-OS initial approach to cope with edge computing and edge analytics concerns and how their vf-OS data analytics component and vf-OS driver component can be used complementarily according to specific needs.

The vf-OS proposal is consistent with vf-OS principles, based in bringing generic development tools to developers that are not manufacturing domain experts, with the purpose to increase the manufacturing applications catalogue available to manufacturing and logistic companies. With that goal in common, the vf-OS Driver component provides common layers out of the box with computing, logging, security and lightweight analytics capabilities.

vf-OS IoT Analytics works collaboratively with vf-OS Analytics, being able to consume vf-OS Analytics models for running real-time edge predictions. In the other hand, vf-OS IoT Analytics predicts events and values that pushed to the vf-OS Analytics in the cloud, can be aggregated with manufacturing orders, and other external information together with input from the user to predict higher-level, intra and/or inter-enterprise analytics supporting non real-time critical decision making.

In order to highlight the practical applicability of vf-OS to the industry, a potential example of the application of IoT Analytics to the predictive maintenance has been presented.

Acknowledgment

The current work is part of vf-OS – Virtual Factory Open Operating System – is a project funded by the H2020 Framework Programme of the European Commission under Grant Agreement 723710.

References

[1] Chiang M., Zhang T. Fog and IoT: an overview of research opportunities, IEEE Internet Things J. 3 854–864 (2016)

[2] Ai Y., Peng M., Zhang K. Edge computing technologies for Internet of Things: a primer. Digital Communications and Networks. volume 4, Issue 2, Pages 77-86 (2018)

[3] Ahmed E., Yaqoob I., Hashem I., Khan I., Ahmed A., Imran M., Vasilakos A., The role of big data analytics in Internet of Things, Computer Networks, Volume 129, Part 2, ISSN 1389-1286 (2017) .

[4] IDC predictions 2016. https://www.idc.com/research/viewtoc.jsp?containerId=259856. Last accessed 2018/07/30

[5] Bilal K., Khalid O., Erbad A., Khan S. Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers. Computer Neworks. (2018).

[6] Loopback Homepage, https://loopback.io/, last accessed 2018/07/23.

[7] vf-OS technical specification Homepage, https://vfos-docs.ascora.eu/#drivers. Last accessed 2018/7/30.

[8] Swagger Homepage, https://swagger.io/, last accessed 2018/07/23.

[9] OpenAPI Homepage, https://swagger.io/specification/, last accessed 2018/07/23.

[10] React js Homepage. https://reactjs.org, last accessed 2018/7/30

[11] Chandola, V., Banerjee, A., & Kumar, V. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, *41*(3), 15 (2009)

[12] Li, M., Ganesan, D., & Shenoy, P. Presto: feedback-driven data management in sensor networks. IEEE/ACM Transactions on Networking (TON), 17(4), 1256-1269 (2009)TensorFlow.js Homepage. https://js.tensorflow.org. last accessed 2018/07/31.

[13] Javed, W., McDonnel, B., & Elmqvist, N. Graphical perception of multiple time series. Visualization and Computer Graphics, IEEE Transactions on, 16(6), 927-934 (2010)

[14] Etzion, O., & Niblett, P. Event processing in action. Manning Publications Co (2010)

[15] TensorFlow.js Homepage. https://js.tensorflow.org. last accessed 2018/07/31.

[16] RabbitMQ Homepage, https://www.rabbitmq.com, last accessed 2018/07/25. RabbitMQ Homepage, https://www.rabbitmq.com, last accessed 2018/07/25.

[17] AMQP specification. ISO/IEC 19464:2014. http://standards.iso.org/ittf/PubliclyAvailableStandards/c064955_ISO_IEC_19464_2014.zip, last accessed 2018/07/25.

[18] McWilliam R., Khan S., Farnsworth M., Bell C. Zero-maintenance of electronic systems: Perspectives, challenges, and opportunities. Microelectronics Reliability, Volume 85, Pages 122-139, ISSN 0026-2714 (2018)

[19] Prakash P., Pal A. Practical Time Series Analysis. Packt Publishing. ISBN: 9781788290227 (2017)

[20] Li X., Ding Q., Sun J. Remaining useful life estimation in prognostics using deep convolution neural networks. Reliability Engineering & System Safety, Volume 172, Pages 1-11, ISSN 0951-8320 (2018)

[21] Fraile, F., Tagawa, T., Poler, R., & Ortiz, A. Trustworthy Industrial IoT Gateways for Interoperability Platforms and Ecosystems. *IEEE Internet of Things Journal* (2018). DOI: 10.1109/JIOT.2018.2832041