ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA032

BIG DATA ARCHITECTURES FOR LOGGING AND MONITORING LARGE SCALE TELESCOPE ARRAYS*

A. Costa†[1], E. Sciacca [1], G. Tosti[2], J. Schwarz[3],
P. Bruno [1], A. Calanducci [1], A. Grillo [1], F. Vitello [1], V. Conforti[4], F. Gianotti[4],
U. Becciani[1], S. Riggi[1]
[1]INAF, Osservatorio Astrofisico di Catania, Catania, Italy
[2]Università di Perugia, Dipartimento di Fisica e Geologia, Perugia, Italy
[3]INAF, Osservatorio Astronomico di Brera, Brera, Italy
[4]INAF, Osservatorio di Astrofisica e Scienza dello Spazio di Bologna, Bologna, Italy

Abstract

Large volumes of technical and logging data result from the operation of large scale astrophysical infrastructures. In the last few years several "Big Data" technologies have been developed to deal with a huge amount of data, e.g. in the Internet of Things (IoT) framework.

We are comparing different stacks of Big Data/IoT architectures including high performance distributed messaging systems, time series databases, streaming systems, interactive data visualization. The main aim is to classify these technologies based on a set of use cases typically related to the data produced in the astronomical environment, with the objective to have a system that can be updated, maintained and customized with a minimal programming effort.

We present the preliminary results obtained, using different Big Data stack solution to manage some use cases related to quasi real-time collection, processing and storage of the technical data, logging and technical alert produced by the array of nine ASTRI telescopes that are under development by INAF as a pathfinder array for the Cherenkov astronomy in the TeV energy range.

INTRODUCTION

Internet of Things (IoT) is an emerging technology that is becoming an increasing topic of interest among technology giants and business communities. IoT components are interconnected devices over the network, which are embedded with sensors, software and smart apps to collect and exchange data with each other or with cloud/data centres. The data generated by IoT devices is large in volume and random in nature and needs to be analyzed using Big Data analytics engine (see e.g. [1]) in order to extract the critical information or to understand behavioural patterns.

ASTRI [2] (*Astrofisica con Specchi a Tecnologia Replicante Italiana*) started in 2010 to support the development of technologies within the Cherenkov Telescope Array[3]. The first result of the project was the construction of a prototype telescope now installed at the astronomical INAF site at the Etna volcano in Sicily. The next phase of the project, currently underway, is the construction of a series of nine units of ASTRI telescopes (named ASTRI Mini-Array) [4].

This paper presents the logging and monitoring software architecture that is under development for the ASTRI mini-array telescopes that takes advantage of this new technological evolution to be prepared for the challenges related to the operation of the telescopes including the reliability, availability and maintainability of all its sub-systems and auxiliary devices.

ASTRI TELESCOPES

The logging and monitoring system takes into consideration all the telescopes and their subsystems [5]. Each telescope includes the system to control the telescope motion (such as the Mount Control System–MCS for the motion of the mechanical structure and the Active Mirror Control–AMC for controlling the primary and secondary mirrors) and the camera activities. In addition, in the mini-array site, we foresee some auxiliary systems to detect the environmental condition (such as the Weather Station–WS) and to assess the quality of the observations (such as the All Sky Camera–ASC). Figure 1 shows the telescopes sub-systems and auxiliary systems handled by the logging and monitoring software.

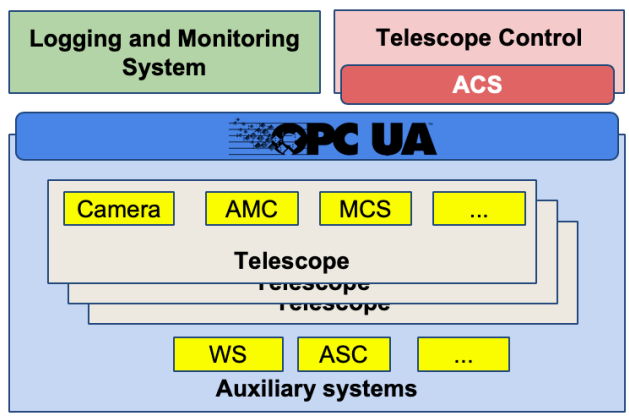

Figure 1: Telescopes sub-systems and auxiliary systems handled by the logging and monitoring software.

Starting from the experiences with the ASTRI prototype, we estimate a technical/operational data load of about 14

* This work was partially supported by the ASTRI "Flagship Project" financed by the Italian Ministry of Education, University, and Research (MIUR) and led by the Italian National Institute of Astrophysics (INAF).
† alessandro.costa@inaf.it

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA032

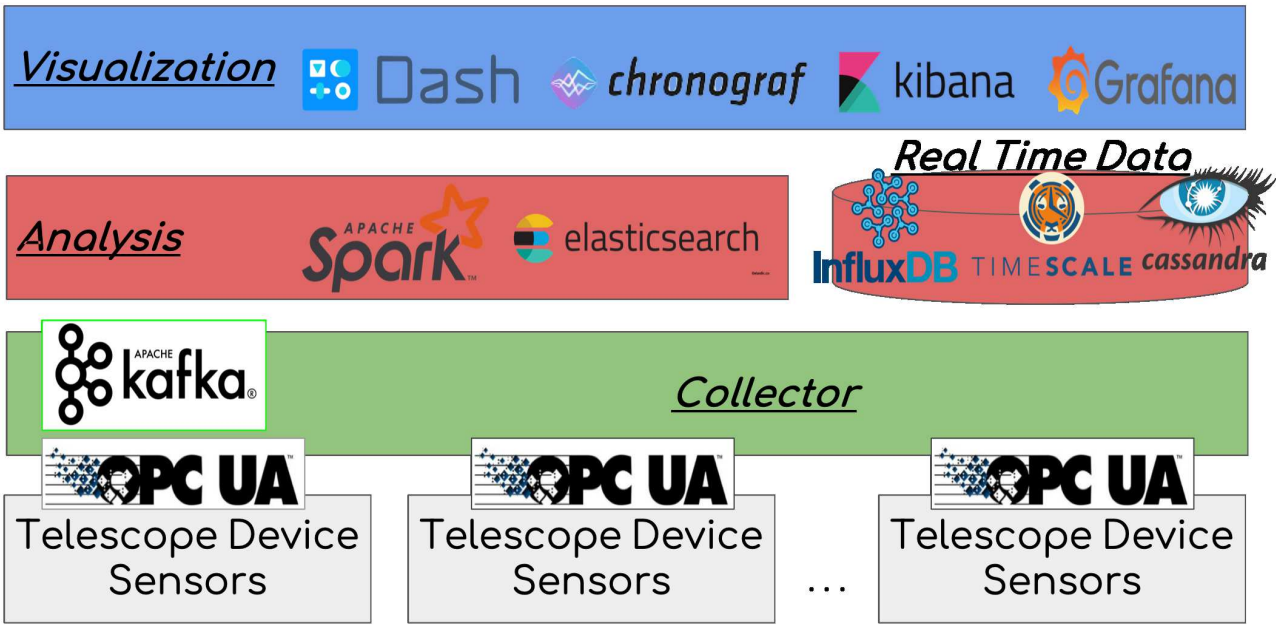

Figure 2: Logging and monitoring software stack layers and employed IoT technologies and tools.

GB per day considering around 20.000 monitoring points and 1 Hz sampling rate.

The interface protocol between the high-level control software and all the hardware assemblies is OPC-UA [6, 7] while the control components are implemented upon the ALMA Common Software (ACS) middleware [8]. Thus, the use of OPC-UA allows the decoupling of the access peculiarities of each assembly with the hardware control systems.

IOT SOFTWARE ARCHITECTURE

The logging and monitoring system software stack has been implemented as standalone modules that can be built and run independently. The architecture comprises three layers (as depicted in Fig. 2):

Collector: it gathers together the IoT messages from the connected telescope devices which are captured by a message broker and are sent to the streaming application for processing.

Analysis: this layer comprises a streaming application which consumes IoT data streams and processes them for data analysis. The IoT data processor stores the processed data in real time databases.

Visualization: this component retrieves data from databases and send them to web pages in fixed intervals so data are refreshed automatically. Dashboard displays data in charts and tables using a responsive web design and it is accessible on desktop as well as mobile devices.

Data Workflow

Devices parameters are sent through OPC-UA protocol and described by an Interface Control Document (ICD) [9]. The ICD comprises, in form of tables, for each control or monitoring point, a complete description of the information required, e.g. data type, OPC-UA node and connected alarms. In our application, the IoT data producer is an OPC-UA simulator application (one for connected device that generates IoT data events). IoT Data are collected using an OPC-UA client and sent through an Apache Kafka [10] Producer over a given topic serialized using the Apache Avro [11] shown in Fig. 3.

Figure 3: Apache Avro schema employed to serialize the telescope IoT data and a sample serialized OPC-UA data from the Weather Station device.

A Kafka Consumer get the deserialized data from the subscribed topic and a Spark [12] application analyzes the data stream and insert the analysis results into a non-relational database optimized for real-time and Big Data applications (e.g. Cassandra [13] and InfluxDB [14]).

Finally, monitoring data are visualized through interactive dashboards. Figure 4 shows sample dashboards using Grafana [15] and Python Dash [16].

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA032

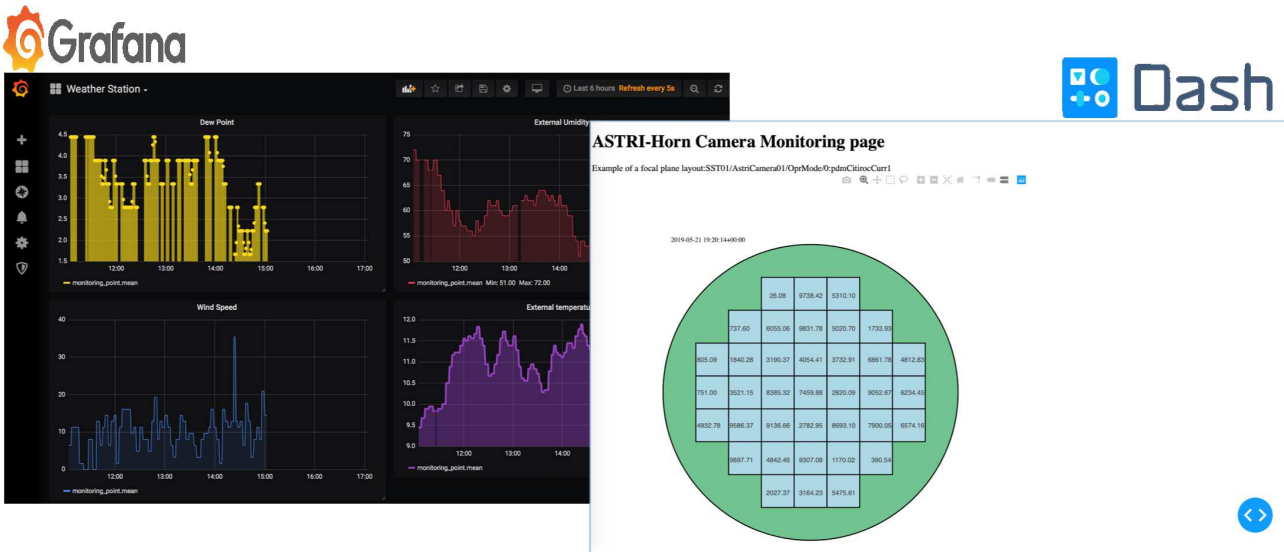

Figure 4: Sample interactive dashboard visualization obtained using Grafana (left figure) and Dash (right figure).

Cloud Deployment

As the technology stack to implement the architecture is made up of several independent components that can be easily swapped out, we deploy each of them as a separate Docker [17] container. This allowed us to start building, testing locally and sharing them easily with the team working on the project.

Once these stages were completed, we started to integrate them and test many modules together. At that purpose, we are using the GARR Cloud Platform [18], based on OpenStack [19] and Kubernetes [20] to orchestrate the execution of the many modules needed within the ASTRI Virtual Datacenter (see Fig. 5).

CONCLUSION AND FUTURE DEVELOPMENTS

We presented a system aimed at monitoring and logging the data needed to improve the operational activities of a large scale telescope array. The prototype was designed and built considering the latest software tools and concepts coming from Big Data and Internet of Things. The software stack is based on open source software minimizing the needs for software development.

We are now planning scalability tests on the GARR Cloud infrastructure to evaluate the system performances simulating different virtual telescopes and scaling the various architectural modules according to the workload. Moreover we are working on a Web dashboard that will allow the team to automatically increase or decrease live the number of simulated devices.

Future works are planned to integrate Machine Learning algorithms to perform anomaly detection and failure predictions. Finally, the system will be deployed on the forthcoming ASTRI mini-array.

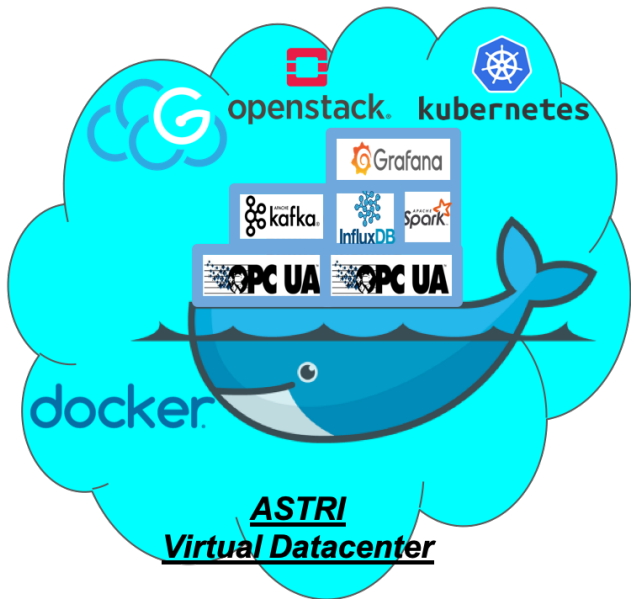

Figure 5: Logging and monitoring software stack deployed at the GARR Cloud Platform (based on OpenStack and Kubernetes) within the ASTRI virtual datacenter using the Docker container based virtualization.

REFERENCES

[1] E. Pena *et al.*, "Framework to use modern big data software tools to improve operations at the paranal observatory," in *Proc. SPIE 10704, Observatory Operations: Strategies, Processes, and Systems VII*, vol. 10704, 2018, pp. 925–935. doi:10.1117/12.2312096

[2] ASTRI project. http://www.brera.inaf.it/astri/

[3] B. Acharya *et al.*, "Introducing the CTA concept," *Astroparticle physics*, vol. 43, pp. 3–18, 2013. doi:10.1016/j.astropartphys.2013.01.007

[4] G. Pareschi, "The ASTRI SST-2M prototype and mini-array for the Cherenkov Telescope Array (CTA)," in *Ground-based*

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA032

and Airborne Telescopes VI, vol. 9906, 2016, 99065T. doi:10.1117/12.2232275

[5] E. Antolini *et al.*, "Telescope Control System of the ASTRI SST-2M prototype for the Cherenkov Telescope Array," in *Proc. ICALEPCS'17*, Barcelona, Spain, 2017, pp. 1266–1270. doi:10.18429/JACoW-ICALEPCS2017-THMPL04

[6] OPC Foundation. https://opcfoundation.org

[7] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC unified architecture*. Springer Science & Business Media, 2009.

[8] G. Chiozzi *et al.*, "The ALMA common software: a developer-friendly CORBA-based framework," in *Advanced Software, Control, and Communication Systems for Astronomy*, vol. 5496, 2004, pp. 205–218. doi:10.1117/12.551943

[9] C. Tanci *et al.*, "The ASTRI mini-array software system (MASS) implementation: a proposal for the Cherenkov Telescope Array," in *Software and Cyberinfrastructure for Astronomy IV*, vol. 9913, 2016, p. 99133L. doi:10.1117/12.2231294

[10] Apache Kafka. https://kafka.apache.org/

[11] Apache Avro. https://avro.apache.org/

[12] Apache Spark. https://spark.apache.org/

[13] Apache Cassandra. http://cassandra.apache.org/

[14] InfluxDB. https://www.influxdata.com/

[15] Grafana. https://grafana.com/

[16] Python Dash. https://dash.plot.ly/

[17] Docker. https://www.docker.com/

[18] GARR Cloud Platform. https://cloud.garr.it/

[19] OpenStack. https://www.openstack.org/

[20] Kubernetes. https://kubernetes.io/