Optimization of Data Flow and Display for Real-Time Visualization of Clinical Data

Maryam BOUMRAH [1], Samir GARBAYA[2], Amina RADGUI [3]

Abstract – Apache Kafka provides real-time streaming of enormously massive data within largely distributed architectures. The use of Kafka streaming tools for monitoring medical data in real-time allows early decision making, the personalization of treatment and immediate feedback for clinicians. However, the main existing visualization tools used along with Kafka stream stay limited to the existing off the shelf Web-based applications such as Grafana, Kibana and Chronograf. To design a new web interface specific to medical data visualization, the challenges of their real-time nature are imposed.
This paper describes the arrangement of medical data flow pipeline using Kafka and the tools used to reach an affluent real-time visualization on a Web server dedicated for patients' monitoring in home-based therapy condition.

1 INTRODUCTION

In the recent years, Apache Kafka became one of the most important platforms which allows building real-time pipeline streams between devices and applications due to its scalability, durability, low latency and distributed nature [1]. When it comes to the need of combining data visualization with Kafka data stream, the ready to use web applications such as Grafana and Kibana are the most recalled. However, medical monitoring dashboards have their specifications and require customization. Building independent web application adapted to healthcare monitoring in real time is challenging when it requires the connection to Kafka servers.

This paper presents the implementation of a pipeline of medical data visualization using Kafka. The challenges faced during the development of this work are discussed along with the proposed solutions to the problems.

2 RELATED WORK

Apache Kafka is a real time distributed platform based on public-subscribed messaging system. It is an open-source project developed at LinkedIn and it is available with Apache Software Foundation [2].
In healthcare, Ta et al. [3] proposed an architecture for big data medical analytics based on Kafka combined with Hadoop, Apache Storm, and NoSQL Cassandra. A real-time processing distributed scheme for the self-health data from a variety of wearable devices was designed by Cai et al.[4] using Kafka and Storm for handling stream data and making decisions without any delay. Alharbi et al.[5] suggested a real-time prediction system to help avoiding heart rate risk in real time. They used Apache Kafka and Apache Spark during the online phase to predict the heart rate in advance.

In the context of existing systems that combined Kafka stream with visualization, Hanamanthrao et al. [6] constructed a full data pipeline using Apache Kafka, Apache Spark for streaming and elastic search, Kibana to query and visualize clickstream data respectively. Similarly, Kuduzet al.[7] used Kibana to provide visualization capabilities on top of the content indexed by Elasticsearch to their system. Their approach was designed using Logstash and Apache Kafka for processing various types of data collected from various IoT devices through LoRa WAN gateways and transform it to uniform way. Beermann et al.[8] used Grafana to implement the ATLAS distributed computing monitoring dashboards which is based on Kafka and Spark for data processing and stream.
The existing systems using Kafka for healthcare are less likely to include dashboards with real-time processing and visualization. In other fields, the systems that combined Kafka along with a visualization board in real-time are limited to the use of out off the shelf visualization web applications.

3 PROPOSED APPROACH

Medical data can flow from different sources carrying out various structures and types. In the implementation of the proposed system, sensing data are mainly used including vital signs (ECG, Blood Pressure, Respiration rate, Temperature) and Kinematic data. These data are time oriented and require a minimum of latency to stay significative.

Based on Kafka architecture, the medical data is ingested into the Kafka producers from different sources and sensors. Every patient is represented by a Topic where the data is transferred into partitions according to their types. As shown in Fig. 1, the data were rearranged into consumer groups of the same types to connect the flow to the data analysis engine.

4 SYSTEM IMPLEMENTATION

The development of the system was implemented on a MacOs machine (Core i7 processor, 12GB RAM) equipped with Kafka 2.13-2.6.1 server, zookeeper and kafka manager. The system was developed using PyCharm 2021.3 and Python 3.9.0.

5 SYSTEM TESTING AND RESULTS

The data used for the testing the implemented system included three datasets; a calibrated database of kinematics and EMG of the forearm and hand [9],Vital signs dataset including heartbeat, blood pressure and respiration rate [10] and a dataset of Post-stroke upper limb kinematics[11]. The data types extracted from these datasets for testing included:

- **Vital signs** (ECG, Blood pressure, Respiration rate)
- **EMG (**magnitude of muscle force of upper limb)
- **Upper Limb Angle joints** (Shoulder, elbow, wrist and hand fingers)

The visualization in real-time of Kafka streamed data is challenging, on one side because of the parallel streams and on the other side because of the accumulated amount of data with time flow. As a first temptation of implementing the real-time display of the flowing data, Dash plotly library based on SVG (Scalable Vector Graphics) and multiprocessing method were used. For testing the performance of the system initially, four streams of data were injected into the data pipeline at the same time. The data were displayed properly and with acceptable latency for a short duration. However, after few minutes and when the number of data streams increases, the live graphs slow down with initial latency of 3 seconds until they completely freeze, and the flow of data is suspended.

The multiprocessing in Python guarantees the parallelism of multiple flows as every process do not share the same memory space. However, multithreading is better choice for the current system. This is because the threads are lighter and less likely to cause overload. They are easier, faster, and safer.

The SVG is an easy option for rendering high-quality vector graphics, but its performance is limited. As an alternative solution, the WebGL provides a JavaScript API that allows to create GPU-accelerated graphics.

As a result of combining the use of Multithreading method and WebGL, the visualization in real-time of the data flow in the implemented system is smooth, continuous in the time and supports significant amount of data streams (9 types of heavy streams while keeping very acceptable latency that does not exceed 100 ms. An example of the displayed data is given in Fig .2.

6 Conclusion

This paper proposed an approach of real-time visualization for healthcare data using Apache Kafka. The test of the initial implementation showed the existence of data flow blockage and important latency in the visualization. The system was improved by changing the methods of processing and graphic display which led to smooth data flow and acceptable visualization latency even for big data flow. The result of this approach for the optimization of data streaming was approved by medical practitioners who considered that the visualized information was useful for the support of medical decision making.

References

[1] F. R. Khan, “Apache Kafka and real-time data streaming,” no. January, 2021.

[2] B. R. Hiraman, M. C. Viresh, and C. K. Abhijeet, “A Study of Apache Kafka in Big Data Stream Processing,” *2018 Int. Conf. Information, Commun. Eng. Technol. ICICET 2018*, pp. 1–3, 2018, doi: 10.1109/ICICET.2018.8533771.

[3] V. D. Ta, C. M. Liu, and G. W. Nkabinde, “Big data stream computing in healthcare real-time analytics,” *Proc. 2016 IEEE Int. Conf. Cloud Comput. Big Data Anal. ICCCBDA 2016*, pp. 37–42, 2016, doi: 10.1109/ICCCBDA.2016.7529531.

[4] J. Cai and Z. Jin, “Real-time Calculating Over Self-Health Data Using Storm,” no. Icmmcce, pp. 2061–2066, 2015, doi: 10.2991/icmmcce-15.2015.398.

[5] A. Alharbi, W. Alosaimi, R. Sahal, and H. Saleh, “Real-Time System Prediction for Heart Rate Using Deep Learning and Stream Processing Platforms,” *Complexity*, vol. 2021, 2021, doi: 10.1155/2021/5535734.

[6] R. Hanamanthrao and S. Thejaswini, “Real-time clickstream data analytics and visualization,” *RTEICT 2017 - 2nd IEEE Int. Conf. Recent Trends Electron. Inf. Commun. Technol. Proc.*, vol. 2018-Janua, pp. 2139–2144, 2017, doi: 10.1109/RTEICT.2017.8256978.

[7] N. Kuduz and S. Salapura, “Building a Multitenant Data Hub System using Elastic Stack and Kafka for Uniform Data Representation,” *2020 19th Int. Symp. INFOTEH-JAHORINA, INFOTEH 2020 - Proc.*, no. March, pp. 18–20, 2020, doi: 10.1109/INFOTEH48170.2020.9066286.

[8] T. Beermann *et al.*, “Implementation of ATLAS Distributed Computing monitoring dashboards using InfluxDB and Grafana,” *EPJ Web Conf.*, vol. 245, p. 03031, 2020, doi: 10.1051/epjconf/202024503031.

[9] N. J. Jarque-Bou, M. Vergara, J. L. Sancho-Bru, V. Gracia-Ibáñez, and A. Roda-Sales, “A calibrated database of kinematics and EMG of the forearm and hand during activities of daily living,” *Sci. Data*, vol. 6, no. 1, pp. 1–11, 2019, doi: 10.1038/s41597-019-0285-1.

[10] S. Schellenberger *et al.*, “A dataset of clinically recorded radar vital signs with synchronised reference sensor signals,” *Sci. Data*, vol. 7, no. 1, pp. 1–11, 2020, doi: 10.1038/s41597-020-00629-5.

[11] A. Schwarz, J. P. O. Held, and A. R. Luft, “Post-stroke upper limb kinematics of a set of daily living tasks,” Mar. 2020, doi: 10.5281/ZENODO.3713449.

- [1] *Centre d'études doctorales Télécoms et Technologies de l'Information (CEDOC-2TI), INPT, Rabat - Maroc*
  E-mail : boumrah.maryam@inpt.ac.ma
- [2] *Laboratoire END-ICAP - INSERM, U1179, Arts et Metiers Institute of Technology, CNAM, LIFSE, HESAM University, F-75013 Paris, France*
  E-mail : Samir.GARBAYA@ensam.eu
- [3] *Centre d'études doctorales Télécoms et Technologies de l'Information (CEDOC-2TI), INPT, Rabat - Maroc*
  E-mail : radgui@inpt.ac.ma@inpt.ac.ma

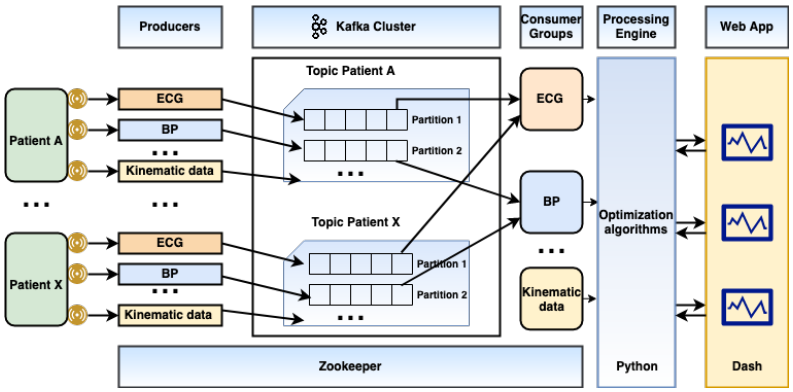

Fig.1 Data pipeline and code implementation.

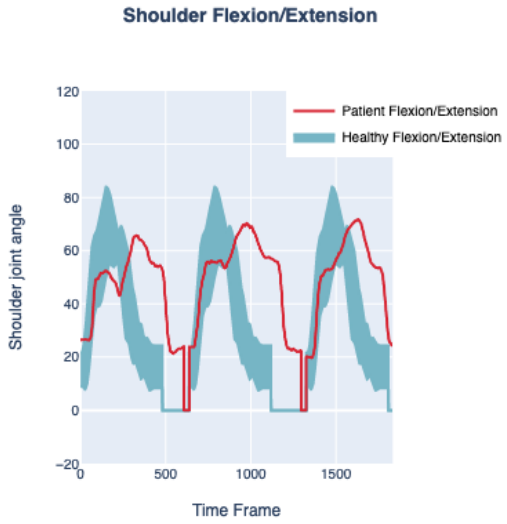

Fig.2 Graph of shoulder joint angle data stream.