engineering proceedings

MDPI

Proceeding Paper

Developing a Cross-Platform Application for Integrating Real-Time Time-Series Data from Multiple Wearable Sensors †

Pataranit Sirithummarak * and Zilu Liang *

Faculty of Engineering, Kyoto University of Advanced Science, Kyoto 615-8577, Japan
* Correspondence: nothing2say.develop@gmail.com (P.S.); liang.zilu@kuas.ac.jp (Z.L.)
† Presented at the 10th International Electronic Conference on Sensors and Applications (ECSA-10), 15–30 November 2023; Available online: https://ecsa-10.sciforum.net/.

Abstract: This research presents a cross-platform application, developed with Flutter, for the efficient integration and management of real-time data from wearable sensors including Apple Watch, Android Wear, and Empatica E4. Compatible with iOS and Android, the app collects various physiological signals for easy analysis by health professionals. Utilizing InfluxDB, a time-series database, our development ensures efficient data handling, even from multiple sources, and enables real-time analytics. This robust, scalable tool signifies a notable advancement in mHealth, offering seamless data integration and management for those utilizing wearable sensor technology in healthcare research and practice.

Keywords: cross-platform application; real-time data; time-series database; wearable sensors

Citation: Sirithummarak, P.; Liang, Z. Developing a Cross-Platform Application for Integrating Real-Time Time-Series Data from Multiple Wearable Sensors. *Eng. Proc.* **2023**, *58*, 4. https://doi.org/10.3390/ecsa-10-16185

Academic Editor: Stefano Mariani

Published: 15 November 2023

1. Introduction

In health research, the importance of collecting quality data is indisputable. The recent advance in consumer-grade wearable technologies further demands technical support for data integration [1–3]. However, domain experts often lack software development skills for integrating health data collected with different wearable sensors, posing a risk of data loss, compromising the quality of data, and impeding data-centered research.

Our research is inspired by the need to support researchers dealing with wearable sensor data [4]. The complexity of developing mobile applications compatible with these sensors often extends beyond the expertise of data scientists, demanding excessive time and resources that could be better spent on core research objectives. Walch et al.'s difficulty [5] in collecting and processing health data from the Apple Watch, characterized by manual submission, inconsistent sampling rates, and data loss, inspired our research to address these complexities.

To address this challenge, we developed a solution that enables domain experts to overcome technical barriers and optimize data collection and integration. We present a cross-platform application, specially designed for seamless integration of data from wearable sensors for health data research. This app streamlines the collection, management, and analysis of time-series data, effectively addressing key design considerations such as the optimal method for developing mHealth apps, the feasibility of Flutter as a cross-platform framework for these apps, and the most suitable time-series database for efficient data storage and real-time tracking in mHealth.

The proposed application transcends these challenges, offering a streamlined approach to data collection, equipped with real-time monitoring, and designed with a focus on ensuring the acquisition of quality data which is integral to insightful health studies. By circumventing the technical intricacies often associated with app development and data integration, we provide a tool that stands as a nexus between efficient data collection and impactful health research, marking a significant stride in the convergence of technology and health research.

2. Methods

We introduce "ZensorsFlow", a cross-platform application designed to seamlessly integrate and manage real-time data from multiple wearable sensors, addressing the evolving needs of mHealth and real-time health monitoring.

2.1. Development Environment and Integration Process

2.1.1. Cross-Platform Mobile Application Development

- Flutter Framework:

ZensorsFlow is grounded in Flutter for both iOS and Android, a decision inspired by the SDK's proficiency in ensuring a consistent user experience across varied platforms and its flexibility [6]. The app leverages Flutter's streamlined process of generating unified code for diverse operating systems, markedly boosting developmental efficiency. Dart, Flutter's intrinsic language, is pivotal due to its asynchronous features, ensuring adept real-time sensor data handling. Our choice of Flutter is not just a technical decision but a strategic initiative to balance performance, development speed, and user engagement seamlessly.

2.1.2. Integration with InfluxDB Cloud Service

- Real-time Data Streaming and Management:

The Flutter-based mobile application was integrated with InfluxDB cloud service to facilitate the real-time streaming and efficient management of data from wearable devices, ensuring seamless and scalable data handling. InfluxDB's adeptness at handling time-series data is underscored by its top rating in performance studies in the mHealth and IoT spheres [7–9]. This empirical backing solidified our selection of InfluxDB for optimized real-time data management in ZensorsFlow.

2.2. Wearable Devices Data Acquisition

2.2.1. iOS—Apple Watch

- Native Development Using Swift:

The Apple Watch's sensor data is accessed through a native development approach using Swift, harnessing the watch's intricate sensor capabilities to collect real-time data.

- Watch Connectivity and Platform Channels:

The Watch Connectivity framework is utilized for the seamless transmission of sensor data between iOS devices and Apple Watches. The data collected is then relayed to the Flutter application through platform channels, ensuring real-time integration and a unified data presentation.

2.3. Android—Android Wear

- Hybrid Development Using Flutter and Native Methods:

The Android Wear application is primarily developed using Flutter with the integrated Flutter Wear plugin, and Kotlin, which is employed for native development to access sensor capabilities.

- Data Layer API and Platform Channels:

The Data Layer API is instrumental in bridging the communication between the Android mobile app and wear app, comparable to the Watch Connectivity framework in iOS. This native environment tool ensures seamless data transmission between Android devices. The data from native environment are transferred to the Flutter application using platform channels.

2.4. Empatica E4 Integration

Integration of the Empatica E4 across iOS and Android is achieved through a native development approach, leveraging the E4link SDK officially provided by Empatica

(Cambridge, MA, USA). This official SDK ensures the precise and real-time acquisition of sensor data, aligning with the data collection from the Apple Watch on iOS and Android Wear on Android.

In both iOS and Android environments, the real-time data from the Empatica E4 is communicated to the main Flutter application through platform channels, ensuring seamless data integration across the platforms.

3. Data Collection and Real-Time Streaming

3.1. Data Types and Collection Methodology

- Physiological Signals:

We are equipped to collect a variety of physiological signals from Apple Watch, Android Wear, and Empatica E4, each offering unique datasets essential for comprehensive health monitoring. Apple Watch and Android Wear provide raw acceleration data and processed heart rate data. The real-time collection of these data types is optimized through native development. The E4, however, provides a richer dataset. It captures Inter-Beat Interval (IBI), Blood Volume Pulse (BVP), 3-axis acceleration, Electrodermal Activity (EDA), and skin temperature. Our application is engineered to synchronize these diverse data types and sampling rates seamlessly, enhancing the precision and reliability of collected health data.

3.2. Real-Time Data Streaming to InfluxDB

- Data Integration and Management:

An exposition of the mechanism through which real-time data, collected from the wearable devices, is streamed, integrated, and managed effectively using the InfluxDB cloud service.

In summary, our development is rooted in Flutter and unifies iOS and Android user experiences, while real-time data management is optimized with InfluxDB. Native development in Swift and Kotlin ensures precise wearable data capture, enriched by the E4link SDK for Empatica E4 integration. The application adeptly handles varied physiological signals and sampling rates. Figure 1 illustrates the systematic architecture, and Figure 2 delineates the app interaction and data flow, showcasing the seamless orchestration of data acquisition, integration, and management.

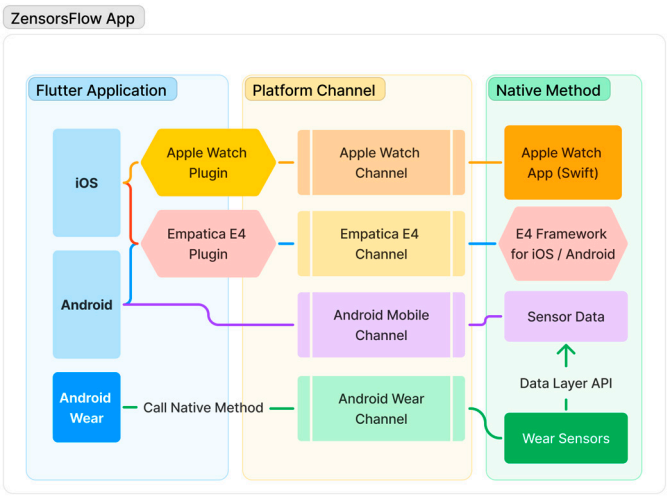

Figure 1. ZensorsFlow's cross-platform system architecture diagram.

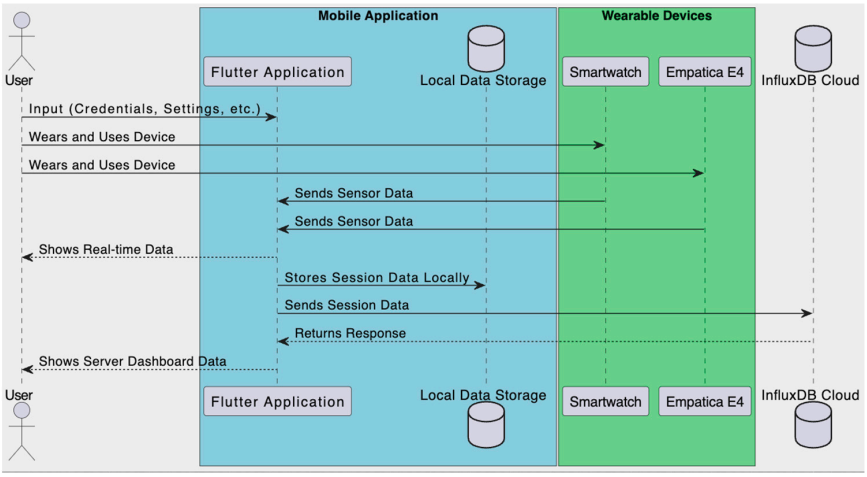

Figure 2. ZensorsFlow's interaction and data sequence diagram.

4. Results and Discussion

4.1. Results

The utility of our application, ZensorsFlow, can be delivered to end-users immediately upon installation. Users are guided through a simplified process of setting up InfluxDB and generating an API key. Following this initialization, the app invites users to a streamlined interface where the database configuration is a one-time, user-friendly process.

The application's operation is intuitive. Users easily connect their wearable devices, initiating real-time data recording. Our app distinguishes itself by supporting simultaneous data collection from multiple devices, a feature tested rigorously.

Data, once recorded, are instantly sent to the pre-configured InfluxDB, and users can monitor real-time data via the dashboard. A notable feature is the option to export data to CSV files for in-depth analysis, enhancing the app's utility for detailed health insights.

4.2. Discussions

The use of Flutter's applicability has both strengths and constraints. While we benefited from its cross-platform compatibility and the efficiency of Hot Reload, challenges like limited connectivity with wearables and extended build times surfaced.

Future research is needed to further refine and optimize the efficiency and usability of our application.

4.3. Future Works

The future development would include integrating low-code platforms like FlutterFlow to simplify and expedite the development process. We aim to expand device support, focusing on enhancing Android Wear integration while bolstering the app's security, performance, and utility.

5. Conclusions

We have presented the development and evaluation of "ZensorsFlow", a cross-platform application that we developed to streamline the process of real-time data integration from a variety of wearable sensors, including, but not limited to, Empatica E4, Android Wear, and Apple Watch. This development addresses the pressing need for efficient and user-friendly tools in the collection and management of health data.

Through rigorous testing and evaluation, we confirmed Flutter's robust capability to develop mobile health applications that seamlessly synchronize with wearable technologies. This affirms Flutter as a viable framework, marrying flexibility and performance, indicative of its potential prominence in future mHealth app developments.

In addressing the development constraints, particularly notable with the Apple Watch which necessitates native development to fully exploit its capabilities, we established

an innovative and harmonious integration of platform-specific developments for both Android and iOS. We engineered platform channels, bridging the Flutter application to native functionalities, ensuring a seamless synchronization of real-time sensor data. With Android Wear, the comprehensive environment of Flutter enabled streamlined application development and data integration. Conversely, for the Apple Watch, we utilized Swift for native development, effectively tapping into its intricate sensor capabilities. These diversely sourced data are cohesively channeled to the Flutter application via platform channels, affirming a consistent and unified data integration and user experience across the diverse platforms.

Furthermore, our exploration into the realm of time-series databases led to us identifying InfluxDB as the optimal choice. Characterized by its efficiency in real-time data management, InfluxDB stands out as a pivotal tool in handling the vast and dynamic datasets generated by sensor technology, confirming our hypothesis concerning database efficacy.

In light of these findings, we hope that ZensorsFlow will serve as a cornerstone in the advancement of mHealth, paving the way for the collection of high-quality multimodal health datasets. Our contributions are poised to not only address the existing challenges but also to catalyze future innovations in the seamless and efficient integration of wearable sensor data for enhanced health research and clinical applications.

Author Contributions: P.S. conceptualized the project, Z.L. provided feedback and discussions; P.S. and Z.L. wrote and proofread the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The source code of the ZensorFlow can be accessed at https://github.com/KUAS-ubicomp-lab/ZensorFlow.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liang, Z.; Ploderer, B.; Liu, W.; Nagata, Y.; Bailey, J.; Kulik, L.; Li, Y. SleepExplorer: A visualization tool to make sense of correlations between personal sleep data and contextual factors. *Pers. Ubiquitous Comput.* **2016**, *20*, 985–1000. [CrossRef]
2. Bertrand, L.; Cleyet-Marrel, N.; Liang, Z. Recognizing Eating Activities in Free-Living Environment Using Consumer Wearable Devices. *Eng. Proc.* **2021**, *6*, 58.
3. Ploderer, B.; Rodgers, S.; Liang, Z. What's keeping teens up at night? Reflecting on sleep and technology habits with teens. *Pers. Ubiquitous Comput.* **2023**, *27*, 249–270. [CrossRef]
4. Sirithummarak, P.; Liang, Z. Investigating the Effect of Feature Distribution Shift on the Performance of Sleep Stage Classification with Consumer Sleep Trackers. In Proceedings of the 2021 IEEE 10th Global Conference on Consumer Electronics (GCCE), Kyoto, Japan, 12–15 October 2021; pp. 242–243.
5. Walch, O.; Huang, Y.; Forger, D.; Goldstein, C. Sleep Stage Prediction with Raw Acceleration and Photoplethysmography Heart Rate Data Derived from a Consumer Wearable Device. *Sleep* **2019**, *42*, zsz180. [CrossRef] [PubMed]
6. Ameen, S.Y.; Mohammed, D.Y. Developing Cross-Platform Library Using Flutter. *Eur. J. Eng. Technol. Res.* **2022**, *7*, 18–21. [CrossRef]
7. Hao, Y.; Qin, X.; Chen, Y.; Li, Y.; Sun, X.; Tao, Y.; Zhang, X.; Du, X. TS-benchmark: A Benchmark for Time Series Databases. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering, Chania, Greece, 19–22 April 2021; pp. 588–599. [CrossRef]
8. Mostafa, J.; Wehbi, S.; Chilingaryan, S.; Kopmann, A. SciTS: A Benchmark for Time-Series Databases in Scientific Experiments and Industrial Internet of Things. In Proceedings of the 34th International Conference on Scientific and Statistical Database Management, Los Angeles, CA, USA, 19–22 April 2021; pp. 1–11.
9. Gangadhar, S. The Real Time Environmental Time Series Data Analysis Using Influx DB. *Int. J. Adv. Sci. Innov.* **2020**, *1*. [CrossRef]