Building a Cloud-based Platform for Personal Health Sensor Data Management

Yang Li, Li Guo, Chao Wu, Chun-Hsiang Lee, and Yike Guo*, *Senior Member, IEEE*

***Abstract*—Recent developments of modern technologies such as cloud computing, wearable sensor devices and big data have significantly impacted people's daily lives, and offer real potential for an Internet-wide, people-centric ecosystem. These advances in technology will considerably extend human capabilities in acquiring, consuming and sharing personal health information. A future in which we are all equipped with devices and sensors that passively collect data and interpret our health and activity status is not far away. The key challenge that we will be facing is how to effectively manage and use that big data. In this paper, we propose a cloud-based platform for health sensor data management, named Wiki-Health, which will provide a potential solution for storing, tagging, retrieving, analysing, comparing and searching health sensor data.**

I. INTRODUCTION

The growing global popularity of smartphones has resulted in new ways of gathering information through an array of embedded sensors such as cameras, microphones, accelerometers, proximity sensors and GPS. The latest generation of professional wearable health sensors can easily connect to smartphones and track significant physiological parameters. Existing wearable fitness sensors such as the Fitbit tracker [1], Zephyr heart rate monitor [2], and Withings blood pressure monitor [3] have already been available on the market for some time, and allow users to automatically collect data on their walking steps, activity levels, food intake, heart rate and blood pressure. This has provided a more efficient and convenient way to collect personal health information. The scale and richness of mobile sensor data being collected and analysed is rapidly growing. However, it is still difficult for users to manage or utilise that collected data in their preferred methods. From the provider's perspective, such massive growth of these big health sensor data creates both data manageability and collaboration challenges.

Health sensors and devices have different characteristics such as sample rate, number of channels and format of the result. For example, most Electrocardiography (ECG) devices have a high temporal resolution, typically at sampling rates between 200 and 1000 Hz, while GPS location readings are taken and stored every few minutes. When such data is aggregated and/or integrated for use in a new application the user is very immediately faced with the multi-resolution nature of that data. Moreover, users of the data will have their own conceptual multi-scale levels of abstraction in their models.

Rapidly evolving modern technologies such as cloud computing, Internet of Things and intelligent data analysis have created new opportunities for improving everyday life. In addition, we can also visualise the role these technological innovations will play in the healthcare sector, spearheading a shift in focus from only offering better healthcare services to people with problems to helping everyone achieve a healthier lifestyle.

To address the aforementioned issues, we present the design and implementation of a platform named Wiki-Health, which will take advantage of cloud computing for personal health sensor data management. Wiki-Health is not only designed to solve the problems of managing and storing the high volume, velocity and variety of data, but also to offer support tools for users to create applications and analysis. This will allow them to effectively utilise all of the data while reducing the complexity of dealing with its diversity.

II. SYSTEM DESIGN

Figure 1 illustrates the overall architecture of Wiki-Health. The Wiki-Health system is designed in three logical layers: data storage, query and analysis and application.

A. Data Storage Layer

The data storage layer is a logical tier that contains a non-relational database, a relational database, CACSS cloud storage [4, 5], an ontology database, and data sources from third party databases.

Health sensor devices such as ECG and Electroencephalography (EEG) employ a number of data channels and possess significantly higher temporal resolution than common environmental sensors like temperature and humidity. Such health sensor data, therefore, demands a scalable, fast and efficient system in order for it to be stored and processed. Traditional relational databases are designed for efficient transaction processing of small amounts of information in a large database. The data sets stored in these databases have no pre-defined notion of time unless timestamp attributes are explicitly added. Non-relational databases store data in a key-value structure. They use looser consistency models than traditional relational databases, thus providing higher scalability and better performance [6-11].

*This research is partially supported by the Innovative R&D Team Support Program of Guangdong Province (NO. 201001D0104726115), China.

*Please direct all inquiries to the corresponding author Yike Guo at y.guo@imperial.ac.uk.

Yang Li, Chao Wu, Chun-Hsiang Lee and Yike Guo* are with the Department of Computing at Imperial College London, UK (email:{yang.li09, c.wu09, c.lee08, **y.guo**}@imperial.ac.uk).

Li Guo is with the School of Computing, Engineering and Computer Science at the University of Central Lancashire, UK (email:lguo@uclan.ac.uk).

We chose HBase [11, 12] to be the foundational non-relational database storage for storing the sensor data in Wiki-Health. It is designed to provide fast real time read/write data access. Cloud storage is another service that helps users reduce the costs, complexities and risks in managing large data growth. CACSS [4, 5] is a generic cloud storage system we have developed to help users reduce the costs, complexities and risks in managing large data growth. It allows access to the data through the S3 compatible interface. In Wiki-Health, CACSS cloud storage system is used to store additional unstructured data as attachments to the sensor data.

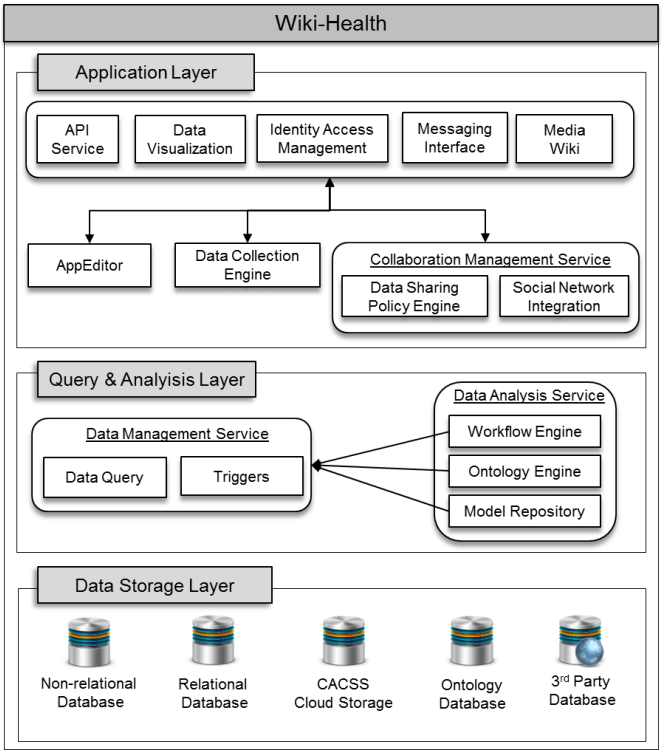

Figure 1. The architecture of Wiki-Health system

Sensor metadata can be thought as descriptions of other data. It can contain background information such as type, accuracy and location of each sensor. Sensor data refers to the actual measurements of sensors. As shown in Figure 2, a hybrid data storage model is used in Wiki-Health to achieve high performance in data access and operations. Sensor metadata and sensor data are completely separate. The system uses MYSQL relational database to store all user information, in addition to sensor metadata such as sample rate, data format, sensor type and other Wiki-Health structured data. All sensor readings, sensor data tags, and other types of sensor data are stored in the non-relational database and CACSS Cloud Storage System.

In Wiki-Health's hybrid data storage model, each data stream maps and holds all the information of the actual health sensor device. A single data stream can have many data units. Such data units are useful for health devices that provide multiple channel readings at the same time. For example, a data stream for a blood pressure device contains data units for systolic and diastolic pressure and heart rate measurements. Data points contains sensor reading data, data attachment index, tag mappings, block mappings and other user defined data. They are stored as a collection of blocks addressed by an index using the data stream ID together with a timestamp (as shown in Figure 3). Data blocks and tags are designed and implemented for developers and users to be able to add more dimensions to the data so that required data can be found more easily and accurately. Data summaries can be used for storing a summary or intermediate analysis result to improve the response time on retrieving certain analysis and query results. A data stream can have multiple triggers. Triggers hold action and condition information. They are used to perform actions when a defined condition is reached. For example, such an action can be configured to alert a caregiver when a certain threshold or unusual reading is discovered.

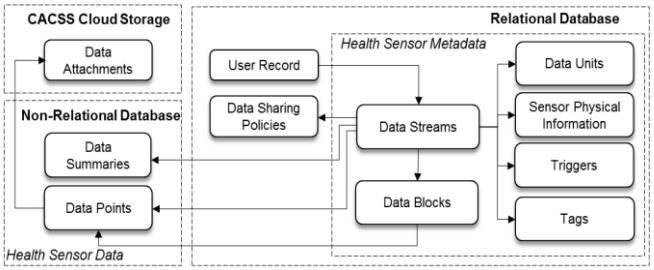

Figure 2. The hybrid data storage model

$$\begin{bmatrix} \boldsymbol{streamid-timestampX} \\ \begin{bmatrix} v: unit\_id1 & 2.0 \\ v: unit\_id2 & 6.2 \\ v: unit\_id3 & 1.6 \end{bmatrix} \\ \begin{bmatrix} t: tag1 \\ t: tag2 \end{bmatrix} \\ \begin{bmatrix} b: blockid1 \\ b: blockid2 \end{bmatrix} \\ \begin{bmatrix} ud: userdefined1 & data \\ ud: userdefined2 & data \end{bmatrix} \\ \begin{bmatrix} \mathrm{a}: attachment1 & fileloc1 \\ a: attachment2 & fileloc2 \end{bmatrix} \end{bmatrix}$$

Figure 3. Data points storage format

B. *Query and Analysis Layer*

The query and analysis layer is used for different data management and data analysis purposes. The data management service component handles all of the generic data access to different data sources in the data storage layer and triggers event actions under pre-defined conditions. The data analysis service consists of the workflow engine, ontology engine and model repository. Figure 4 depicts the analysis framework overview of the Wiki-Health platform.

The workflow engine component is a runtime environment to schedule and execute the workflows and processes constructed by the AppEditor (described in the application layer section). The workflow engine component is implemented using JBOSS jBPM [13, 14].

The ontology engine component manages and queries the ontology via an API that supports the SPARQL language [15]. It is implemented with Virtuoso [16] and dotNetRDF [17] library. The ontology is adopted in the system for two main reasons: first, ontology is used to annotate and describe the data source, and provide the semantic substrate to manipulate various formats of health data sources, such as flexible query and data validation. Second, ontology provides the basis for

our data federation and fusion. For multiple data sources, we fuse them with common upper ontology. In general, ontology makes it possible for aggregating and/or integrating different resolutions and formats of sensor data.

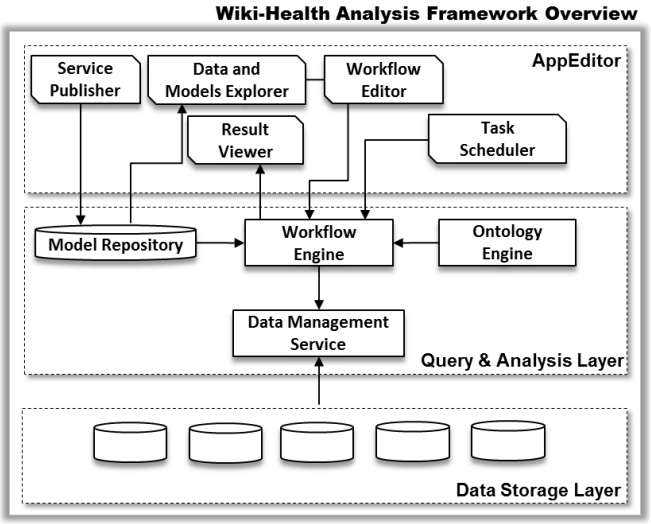

Figure 4. Wiki-Health Analysis Framework Overview

One of the goals of our system is to create an ecosystem where users can voluntarily contribute their data and models. The model repository serves this purpose; it stores all user and system defined functions, models and scripts for reuse of the data and knowledge. The model repository is currently implemented with the R [18] environment deployed on top of IC-Cloud [19, 20]. The submitted models are stored as R source files.

C. *Application Layer*

The application layer consists of all the components required for managing data access, data collection, data security, data sharing and other features that support online collaboration. The API service component offers a web-based interface for access to all of the functionalities of Wiki-Health. The identity access management component is a separate service that provides authorisation and access control of all requests. The messaging interface is used to connect Wiki-Health to external third party API and services. The data visualisation component is used to generate different graphical representations of raw sensor data as well as processed data.

The media wiki component is implemented using the open source MediaWiki [21] system. Wiki-Health interacts with the wiki component through MediaWiki's API to insert and update contents. It provides a collaborative way of allowing users to publish and share content relating to the data acquisition, data analysis models, methods and workflows, based on the health sensor data stored in Wiki-Health. Sharing and collaboration are both key features that make the health sensor data management useful and convenient.

The collaboration management service comprises a data sharing policy engine to resolve and manage data sharing policies defined for data streams. These policies ensure that data can only be accessed by specific users at prerequisite times and from the right locations. The data sharing policy engine is implemented using Drool tools [22]. The social network integration component creates a data-sharing social environment for users to share their data and communicate with other people who have common needs and who want to learn more about themselves.

The data collection engine is intended to liberate a user's personal health data collected through different providers, devices and platforms, by allowing a user to create a copy of all of his or her data and maintain it in the Wiki-Health platform through the third party API.

AppEditor is an online development environment, which provides a graphical user interface to integrate data from different sensors, apply analytic models, construct the workflow, and publish the final service as a sensor application for future data analysis and queries. AppEditor has been discussed in more detail in [23].

III. CURRENT IMPLEMENTATION STAGES

A. *Wiki-Health Mobile Data Collection Application*

The purpose of labelling, tagging or annotating sensor data is often to identify the correct event, stimulus or cause associated with a corresponding sequence of sensor data. Such labels are useful for data analysis algorithms. Therefore, we have developed a mobile application as part of the platform to help users collect and tag sensor data from smartphones.

Figure 5 shows some screenshots of the current mobile application. To summarise, the application performs periodic gathering of data from every available sensor of the device in order to record the activities of the user. It also allows users to tag and enter additional information about their activities during specific hours of a selected date. Such information is annotated and linked to the collected sensor data. All data is first stored locally in a private database from the application before uploading to the Wiki-Health platform through the API. This mechanism is designed to improve privacy control–users can remove any unwanted collected data, so that only user selected data will be uploaded.

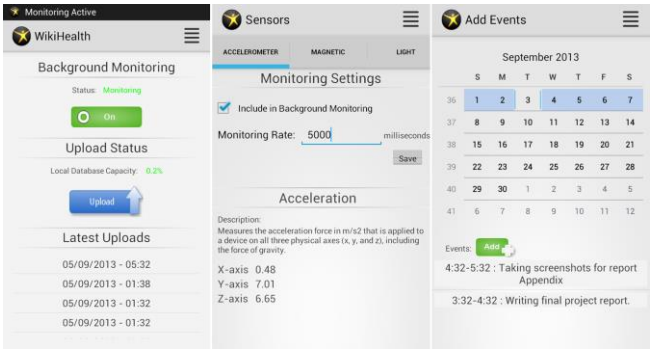

Figure 5. Screenshots of Wiki-Health mobile app for the data collection

B. *AppEditor*

Various graphical shapes are used in AppEditor to represent different elements in a data analysis workflow as shown in Table 1.

TABLE 1 AVAILABLE ELEMENTS IN THE APPEDITOR

Element (shape)	Description

Data Source (circle)	Select single or multiple available sensor data sources.
Filter (diamond)	Query the data source.
Model (rectangle)	Apply various prediction, mining and analysis models on selected data.
Connector (triangle)	Control the flow of the application, such as loops and conditions.
Visualiser (square)	Visualise the output of the application.

Figure 6 shows an example of an EEG data analysis workflow composed by the AppEditor: four available EEG data sources are selected and the data is filtered by the date (22_Aug_2013). Then, several other analysis models are applied in sequence to obtain the final results.

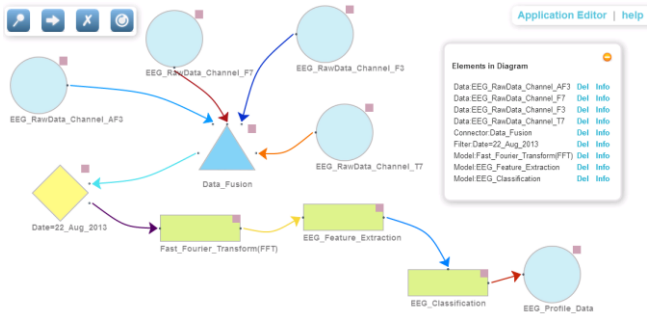

Figure 6. GUI of AppEditor prototype

C. *Wiki-Health GUI*

Figure 7 shows the web-based GUI of the Wiki-Health health data browser with some collected data. In general, the GUI provides an interactive way to view data and it allows users to share selected health and activity information with others. All system components outlined by this paper are deployed across different virtual machines on top of IC-Cloud [19, 20]. IC-Cloud is a generic IaaS cloud computing infrastructure. It allows us to design and quickly compose a cloud computing environment in a flexible manner.

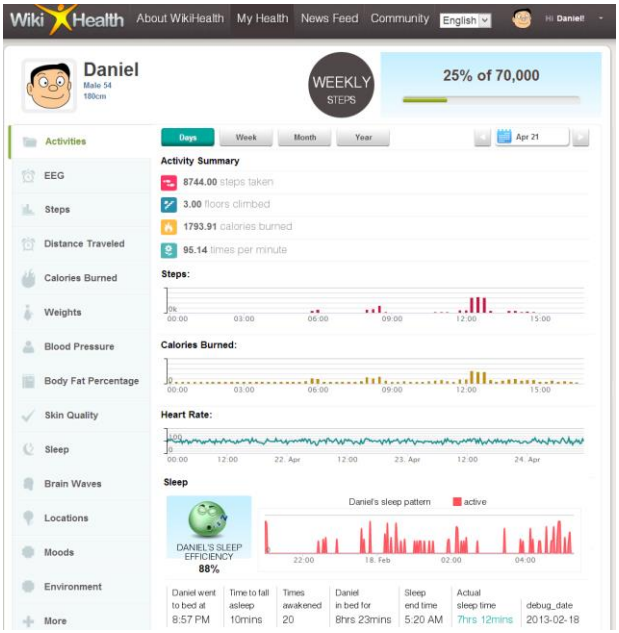

Figure 7. GUI of Wiki-Health web-based health data browser

IV. CONCLUSION

This paper presents the design rationale and implementation of Wiki-Health, a cloud-based platform for personal health sensor data management. The proposed hybrid storage model helps achieve high performance for both data accessing and analysis operations. We have also introduced a collaborative approach for health sensor data management that allows users not only to collect, label, tag, annotate, update and share sensor data, but also to create, reuse and remix data analysis results and models.

Our next step is to carry out a series of systematic studies using real world cases (constant ECG monitoring and long term EEG) to evaluate the performance of the Wiki-Health system. In addition, we will further develop the existing platform and its associated tools so that the collected data can be used not only by data analysis specialists but also by ordinary users to help them better understand themselves, learn from others and lead them to achieve a healthier lifestyle.

REFERENCES

[1] "Fitbit," http://www.fitbit.com/.
[2] "Zephyr," http://www.zephyranywhere.com/.
[3] "Withings," www.withings.com.
[4] Y. Li, L. Guo, and Y. Guo, "CACSS: Towards a Generic Cloud Storage Service," in CLOSER, 2012, pp. 27-36.
[5] Y. Li, L. Guo, and Y. Guo, "An Efficient and Performance-Aware Big Data Storage System," *Cloud Computing and Services Science*, pp. 102-116: Springer, 2013.
[6] N. Leavitt, "Will NoSQL databases live up to their promise?," *Computer*, vol. 43, no. 2, pp. 12-14, 2010.
[7] I. T. Varley, A. Aziz, C.-s. A. Aziz, and D. Miranker, "No relation: The mixed blessings of non-relational databases," 2009.
[8] M. Stonebraker, "SQL databases v. NoSQL databases," *Communications of the ACM*, vol. 53, no. 4, pp. 10-11, 2010.
[9] R. Cattell, "Scalable SQL and NoSQL data stores," *ACM SIGMOD Record*, vol. 39, no. 4, pp. 12-27, 2011.
[10] D. Carstoiu, A. Cernian, and A. Olteanu, "Hadoop Hbase-0.20.2 performance evaluation." pp. 84-87.
[11] A. Khetrapal, and V. Ganesh, "HBase and Hypertable for large scale distributed storage systems," *Dept. of Computer Science, Purdue University*, 2006.
[12] A. HBase. http://hbase.apache.org/.
[13] M. Cumberlidge, *Business Process Management with JBoss JBPM: A Practical Guide for Business Analysts; Develop Business Process Models for Implementation in a Business Process Management Sytem*: Packt Publishing, 2007.
[14] J. Koenig, "JBoss jBPM white paper," *JBoss Labs*, 2004.
[15] E. Prud'Hommeaux, and A. Seaborne, "SPARQL query language for RDF," *W3C recommendation*, vol. 15, 2008.
[16] "Virtuoso," http://virtuoso.openlinksw.com.
[17] "dotNetRdf library," http://www.dotnetrdf.org.
[18] R. Gentleman, *R programming for bioinformatics*: CRC Press, 2008.
[19] Y.-K. Guo, and L. Guo, "IC cloud: Enabling compositional cloud," *International Journal of Automation and Computing*, vol. 8, no. 3, pp. 269-279, 2011.
[20] L. Guo, Y. Guo, and X. Tian, "IC cloud: a design space for composable cloud computing." pp. 394-401.
[21] "MediaWiki," http://www.mediawiki.org.
[22] "DROOL," http://www.jboss.org/drools/.
[23] C.-H. Lee, D. Birch, C. Wu, D. Silva, O. Tsinalis, Y. Li, S. Yan, M. Ghanem, and Y. Guo, "Building a generic platform for big sensor data application." pp. 94-102.