Providing Interoperability between Wearable Devices and FHIR-based Healthcare Systems

Arthur T. Cabral
Unisinos
São Leopoldo, Brasil
arthurtcabral@gmail.com

Gabriel Souto Fischer
Unisinos
São Leopoldo, Brasil
gsfischer@edu.unisinos.br

Rodrigo da Rosa Righi
Unisinos
São Leopoldo, Brasil
rrrighi@unisinos.br

Cristiano A. da Costa
Unisinos
São Leopoldo, Brasil
cac@unisinos.br

Alex Roehrs
Unisinos
São Leopoldo, Brasil
alexr@unisinos.br

Sandro José Rigo
Unisinos
São Leopoldo, Brasil
rigo@unisinos.br

Blanda Mello
Unisinos
São Leopoldo, Brasil
blandamellus@gmail.com

ABSTRACT

The manufacturers of IoT-based wearable devices present different ways to deliver the captured vital signs. Healthcare third-party systems need to adapt to the different approaches, rising complexity and acts against any idea of seamless integration. In addition to the interoperability problem, we observed that the state-of-the-art does not present alternatives that capture and analyze vital signs in such a way that they can be used to diagnose the user's health status automatically. This article presents the HealthTranslator model, which provides interoperability by integrating multiple wearable devices and delivering their data through an HL7 FHIR healthcare standard format. More precisely, our contribution combines different user-oriented periodicity strategies to collect vital signs with data enrichment over an FHIR file, effortlessly delivering valuable insights regarding the user's health status. We developed a microservices-based prototype that collects data on different APIs from different companies/wearables. The resources were encouraging, where data from Xiaomi, Apple, Garmin, and Samsung devices were read and delivered in an FHIR format. Thinking at the intelligent cities level, the provided interoperability is essential to generate the new vision of healthcare, where users and patients will be 24/7 monitored from their own homes.

KEYWORDS

wearable devices, FHIR, healthcare systems

1 INTRODUCTION

As a medical adherence and home monitoring tool, the Internet of Things (IoT), known as Internet of Medical Things (IoMT), can significantly enhance healthcare globally[2, 7, 13]. IoT-powered technologies and sensors can be found almost everywhere to gather, monitor, and greatly enhance regular patient conditions, redefining how healthcare facilities and systems act to save lives[9]. Many articles in computer science applied to the medical area use IoT devices to deliver Artificial intelligence (AI) and observations [10]. In general the IoT is part of a medical device layer that communicates with higher ones, including fog and cloud computing, as well as AI-based solutions. Concerning the theme of IoT in vital signs collection, companies commonly provide WebAPIs (Application Program Interface) based on cloud computing [16]. Using these hooks, we can implement, for example, a model that presents a direct connection between the WebAPI of a particular company and a particular server dedicated to a third-party application.

IoT healthcare systems generally have three layers: body area sensor network, Internet-connected gateways, and AI-extensive cloud data support[9]. Commonly, gateways act as a hub between a sensor and cloud layers. The literature generally presents the gateway layer, which transfers the data captured by the IoT sensors to the other layers or applications of a particular system. We also observed that the works promoting the gateway layer do not address the theme of the periodicity of vital signs reading according to the user's symptoms (for example, feverish body temperature or cardiologic problems)[13]. We believe that utilizing this information to determine an optimal frequency for collecting users' data is crucial for promptly and accurately reporting abnormal conditions.

Aiming to provide communication between different wearable IoT devices and other software layers or applications it is relevant to implement an API that ensures interoperability between these technologies. However, we noted that the current landscape of IoT healthcare systems reveals a gap in developing models that integrate wearable health devices with third-party applications to prediagnose users transparently. Also, this integration can be enhanced with data enrichment, which allows information on user health status using vital signs data. While IoT frameworks have shown promise in monitoring and collecting vital signs through body area sensor networks and transmitting the data through gateway layers to cloud services, more emphasis still needs to be explored on utilizing and standardizing this information.

By establishing robust connections between wearable health devices and sophisticated algorithms, it is possible to analyze and interpret vital signs data in a more accurate and timely way, allowing the feasibility to compute prediagnostic insights. We did not locate articles that implement a gateway to consolidate the data collected from devices into a single file utilizing the HL7 FHIR health file standard.[1, 8, 9, 12]. Moreover, we observed the need for works that use the data enrichment technique to generate valuable insights over captured vital signs. It is also important to mention that a module or algorithm that aims to determine a specific periodicity for collecting the vital signs considering the individual

In: Proceedings of the Brazilian Symposium on Multimedia and the Web (WebMedia'2024). Juiz de Fora, Brazil. Porto Alegre: Brazilian Computer Society, 2024.

ISSN 2966-2753

Table 1: Comparison among the related work

Ref.	Data Enrichment	Supported devices	Scalability
[4]	Not covered	Use cases in medical scenarios.	Use of RESTful API, Node.js, microservices-based architecture and NOSQL databases.
[9]	Improvement of data considering optimization, security, syntax, semantics, and protocol aspects	IoT devices in general, presenting an use case in MIoT area	Use of broker plugins and converter plugins. Optimization in data conversion by avoiding redundant computations.
[8]	Ontology	ECG sensor device	Use of ActiveMQ, ElasticSearch, Hadoop ecosystem and MariaDB database
[5]	Use an ontology for data enrichment. Definition of constraints and situations with basis in health data reading	Medical IoT devices in general	Not covered
[17]	An initial filtering function that removes any unwanted wireless data, such as duplicate data from the same network router or IoT devices	IoT devices, in general	Use of MongoDB database with two levels: level 1 data contains all the raw data received from the wireless sensor nodes, which is stored in the original format for future data recovery purposes. Level 2 contains all the processed data.
[12]	Not covered	Medical IoT devices	Use of fog nodes and an algorithm for using them
[14]	Not covered	Temperature sensor, pulse sensor, enzymatic sensor and immunosensor	Three-layers architecture: Perception, Network, and Application
[6]	Conversion of text into audio content	Mobile Galvanic Skin and Response sensor	Not covered
[1]	Not covered	IoT devices in general presenting two use cases in health area	MW2MW Integration

characteristics of each person is also a gap presented by the literature. Periodicity is essential not only to provide a better treatment and care to user viewpoint, but also to save network bandwidth when monitoring digital health in a s,art cities scale.

This article presents the HealthTranslator model, which provides interoperability by integrating multiple wearable devices, delivering their data through HL7 FHIR (Fast Healthcare Interoperability Resources) healthcare standard. FHIR is a protocol that defines a set of rules and specifications for exchanging electronic health care data. It is designed to be flexible and adaptable for use in various settings and with different healthcare information systems. FHIR aims to enable the seamless and secure exchange of healthcare information so patients can receive the best care. We developed a microservices-based prototype that collects data on different APIs from different brands of wearable devices. The resources were encouraging, where data from Xiaomi, Apple, Garmin, and Samsung devices were read and delivered in an FHIR format. Considering the smart cities perspective, interoperability is essential to generate the new vision of healthcare, where users and patients can be fully monitored from home. We define our contribution as twofold.

- We provide a user-by-user adaptation regarding the periodicity to collect vital signs depending on the health status of each patient;
- Using data from different vital signs of a person, we developed a data enrichment strategy over an FHIR file in such a way as to provide an automatic prognosis of the healthcare situation of such a user.

2 RELATED WORK

Table 1 compares the selected papers when considering the scope of healthcare, interoperability and IoT. Taking into account the articles concerned explicitly with devices that collect health data[8], [12], and [14], we have the collection of the following vital signs: heart rate, temperature, blood pressure, oxygen saturation, blood measurements (Blood circulation, glucose, lactate, calcium, potassium), immunity data. However, we do not have the use of a standard healthcare-orientred API that natively supports data enrichment. These features are relevant not only to export the collect data to higher layers targeting system integrations, but also to favor some combination and analysis of vital signs towards providing prediagnosis of any diasease. Another relevant topic addressed by the current state-of-the-art is scalability. Addressing this thematic in the context of healthcare, [4] provides the use of RESTful API, NOSQL databases, and cloud principles. The authors used Node.js in the Web Server, as it avoids the multithreading burden by employing a nonblocking single-thread pattern and can efficiently serve multiple concurrent clients by operating asynchronously, employing the event-loop mechanism, which is particularly suited for microservices architecture. We observed that adaptations in the data capturing period was not addressed as possibility to generate scalability when dealing with a massive number of users.

3 HEALTHTRANSLATOR MODEL

This section describes the HealthTranslator model. Its main goal is to provide a unified format of API to read vital signs collected by different IoT devices. More precisely, it offers data in a single format using the HL7 FHIR healthcare standard. This is pertinent both to ensure interoperability between different systems and to enable data enrichment-oriented vital sign-based pre-diagnose of users health. Thus, stakeholders like Medical applications, academic projects, and personal studies that need to colect data from vital signs would benefit from not having to adapt to the different ways of connecting to particular WebAPIs of each company. HealthTranslator can be seen as a middleware, acting between WebAPIs to collect vital signs and applications that need to use such data.

3.1 Design Decisions

In addition to interoperability and data enrichment to enable user pre-diagnosis, HealthTranslator also addresses the management of the period to trigger the reading vital signs of each user. We are working with six project decisions: (i) the IoT devices covered by HealthTranslator must be wearable devices, like smart bracelets and smartwatches, from Amazfit, Fitbit, Garmin, Samsung, and Xiaomi; (ii) the vital signs considered by the proposed model are body temperature, heart rate, heart rate variability, oxygen saturation, and respiration rate; (iii) the output file must be in HL7 FHIR standard; (iv) the format of this file is clear-text and expressed in XML or JSON; (v) the rules for the periodicity of vital sign readings need to consider the intervals of normal and abnormal situations;

(vi) the output file must have an observation about the health of the users based on the vital signs data read.

3.2 Architecture

As a middleware, interoperability and abstract are keywords in the HealthTranslator context. We see Healthtranslator acting at the edge, as an app for cellphones or a hub or gateway assembled with single board computers (such as Arduino, Raspibery Pi or Orange Pi). HealthTranslator comprises four microservices: Contact Module, Periodicity Definition Module, Vital Signs Collector Module, and Vital Signs File Maker Module. Figure 1 illustrates the architecture of the proposed model. All microservices have been developed following the set of Domain Driven Design (DDD) principles for code organization, privileging the existence of a business layer, making the system modeling based on the reality of the business [11]. In addition, we modeled reliable communication between two microservices with JSON files and HTTP connections (through port 80 and using the TCP protocol at the transport layer). Each mivroservice accesses particular database tables through SQL.

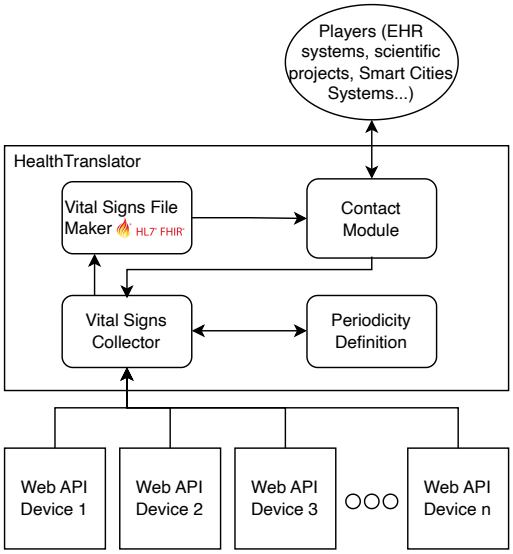

Figure 1: HealthTranslator Architecture

We start the HealthTranslator operation with a request from a player to query vital signs collected by one or more IoT devices. To accomplish this, it is necessary to specify the type of file desired: XML or JSON. Through a PubSub organization, this player subscribes to receive specific patients' health data. Afterwards, a signal is sent to the Vital Signs Collector Module. This last consults the APIs of the companies' devices for collecting health data. The standard periodicity for checking vital signs is every six hours, considering what the Australian Commission on Safety and Quality in Health Care defends. Thus, the Vital Signs Collector Module consults the APIs, and then this module sends the data to two other modules: Vital Signs File Maker and Periodicity Definition. The first is responsible for analyzing the vitals, writing an observation about them and finally creating a file with the FHIR pattern. The second determines the periodicity of API consultation by the Vital Signs Collector Module. It is essential to mention that if the player wants to stop the subscription for receiving the vital signs data of a specific patient, the player can use a stop function that registers the cancellation in the database.

The Vital Signs Collector Module consults each of the different APIs of the IoT devices to collect vital signs. After receiving an authorization signal for this action, this module will query the APIs in accordance with the rules provided b the the Periodicity Definition Module. In general, the APIs use the REST architecture for making the data available and the JSON format for it. Therefore, a JSON object will be instantiated for each data read from the API, being the first data representation after readings. The object will comprise properties to cover the reading of the following vital signs: body temperature, heart rate, oxygen saturation, and respiration rate. while body temperature, oxygen saturation, and respiration rate are single values, the heart rate is an array of values obtained between two periodicity calls. Table 2 presents the reference numbers used to compute the periodicity to collect vital signs. In addition, these numbers are also important to construct the observation when doind data enrichment over a FHIR file.

Table 2: Observations produced by the Vital Signs File Maker

Vital signs	Value	Observation
Body temperature	$x < 36.2$	Low temperature
Body temperature	$x > 37.1$	Fever
Heart rate	$x < 60$	Bradycardia
Heart rate	$x > 100$	Tachycardia
Oxygen saturation	$x < 95$	Low oxygen saturation
Respiratory rate	$x < 12$	Low respiratory rate
Respiratory rate	$x > 20$	High respiratory rate
Heart rate (x) and respiratory rate (y)	$x < 60$ and $y < 12$	Possible hemorrhage
Heart rate (x) and body temperature (y)	$x > 100$ and $y > 37.1$	Possible generalized infection
All	All normal	Normal vital signs

The Periodicity Definition Module considers a query of every six hours. This module also checks if the captured values are within the normality of vital signs, considering the values understood as usual for human health. The Vital Signs File Maker Module, in its turn, receives the data sent by the Vital Signs Collector Module and inserts it in a file that has the FHIR standardization. Algorithm 1 presents how an observation of a particular user is construct from an input of vital signs. Without it, a stakeholder would receive only the raw data requested. For example, in the case of a request for a user's body temperature, we would have only this data and a particular timestamp. Finally, Table 2 shows the observations produced by our data enrichement technique. These observations have been written based on [15], where we have the characteristics of the vital signs themselves (for example, "Low oxygen saturation" when the oxygen saturation is below 95). Algorithm 1 presents the code responsible for generating the observation.

As HealthTranslator use-case, we present the situation where many patients with a specific illness are monitored. Here, suppose that a group of 20 patients has a flu, and a player is testing the effects of a medication on this group. It is also assumed that these patients have vital signs collection devices of different brands. For this monitoring purpose, instead of adapting to the different WebAPIs provided by the companies, we would just adapt the third-party applications to read FHIR data generated by HealthTranslator. Over time, we will receive the files with the vital signs, already

having a pre-diagnosis of the patient, so easing the reading and interpretation of his/her vital signs.

Algorithm 1: Writing an observation on the user vitals

```
Data: VitalSignsToBeProcessed
Result: FHIR Observation
begin
    while VitalSignsToBeProcessed do
        if BodyTemperature < 36.2 then
            Observation ← Observation + "Low temperature."
        end
        if BodyTemperature > 37.1 then
            Observation ← Observation + "Fever."
        end
        if HeartRate < 60 then
            Observation ← Observation + "Bradycardia."
        end
        if HeartRate > 100 then
            Observation ← Observation + "Tachycardia."
        end
        if OxygenSaturation < 95 then
            Observation ← Observation + "Low oxygen saturation."
        end
        if RespiratoryRate < 12 then
            Observation ← Observation + "Low respiratory rate."
        end
        if RespiratoryRate > 20 then
            Observation ← Observation + "High respiratory rate."
        end
        if RespiratoryRate < 12 and HeartRate < 60 then
            Observation ← Observation + "Possible hemorrhage."
        end
        if BodyTemperature > 37.1 and HeartRate > 100 then
            Observation ← Observation + "Possible generalized infection."
        end
        if All vitals are normal then
            Observation ← "Normal vital signs."
        end
        FHIRFiles ← "GenerateFHIRFiles()"
    end
    return FHIRFiles;
end
```

4 EVALUATION METHODOLOGY

We developed a prototype that was composed by a collection of microservices, one for each developed module. We are working with the WebAPI from Apple, Xiaomi, Samsung and Fitbit. At this moment, only the JSON HL7 FHIT output is allowed. For testing purposes, we are simplifying the data aquisition task: we are using a file that presents a collection of vital signs. The use of a dataset will establish the initial test load. This dataset includes the following vital signs: body temperature, heart rate, oxygen saturation, respiration rate. More precisely, in the dataset we have 23 hours of records, where the users presented moments: (i) their vital signs were typical and; (ii) the values are different from normality. Thus, we have a configuration that allows us to test the prototype and the generation of observations over different cases.

We conducted unit tests to evaluate the correct funcioning of each module separately. According to [3], unit testing is widely practised in industry. Here, ACM suggests that software testing should be integrated into Computer Science and Software Engineering curricula. The test cases implemented for testing the Vital Signs File Maker Module were the following ones: (i) observe with normal values should set observation to "Normal vital signs"; (ii) observe with low body temperature should set observation to "Low temperature"; (iii) observe with high body temperature should set observation to "Fever"; (iv) scenario 4: Observe with low respiratory and heart rates should set observation to "Possible haemorrhage"; (v) observe with low heart rate should set observation to "Bradycardia"; (vi) observe with high heart rate should set observation to "Tachycardia"; (vii) observe with low oxygen saturation should set observation to "Low oxygen saturation"; (viii) observe with low respiratory rate should set observation to "Low respiratory rate"; (ix) observe with high respiratory rate should set observation to "High respiratory rate"; (x) observe with multiple conditions should set observation in accordance with the characteristics of these conditions; (xi) make observations with multiple users, and WebAPIs should set observations for each user.

5 RESULTS

The tests on individual modules of HealthTranslator were successful and met all established criteria. HealthTranslator facilitates communication between WebAPIs of mobile devices that collect vital signs. The Contact Module successfully received and recorded subscription requests from new users in an internal database. It then signaled the Vital Signs Collector Module to begin data collection. The functional test for this process took less than 3 seconds, involving two users and two WebAPIs. The Vital Signs Collector Module effectively consulted the relevant dataset, triggered by either the Contact Module or the Periodicity Calculator Module, to provide data for both the Periodicity Calculator Module and the Vital Signs File Maker Module. Using the Newtonsoft.Json library, data was accurately converted and quickly saved in the database.

The Periodicity Calculator module interacted adequately with the Vital Signs Collector Module, so computing the consultation period based on the vital signs. Using the polling technique, the module efficiently retrieved the information from the database, ensuring accurate timing for subsequent queries. We highlight that the processing time for individual requests remained consistently below 3 seconds considering a scenario with two users and 2 WebAPIs. Vital Signs File Maker Module was tested with unit and functional testing. We noted that the module correctly generated the observations according to the user's vital signs, which indicates the success of the application of the data enrichment technique. As expected, six objects have been generated in FHIR format using the "Patient" and "Observation" resources. The processing time of this module was, in general, like the others tests: less than 3 seconds. FHIR standardization was correctly used in the observation files, which ensures better compatibility of HealthTranslator with other healthcare applications. The standardization relied on using "Patient" and "Observation" resources, representing respectively the user's vital signs the observation-based prediagnosis of the user health status.

6 CONCLUSION

This work explored the data enrichment technique for writing a description of the health status of a user based on his/her vital signs. Using the FHIR standard together with well-known XML and JSON file types, we see HealthTranslator as a pertinent hub to integrate vital signs with the current global-scale applications running in hospital centers. In addition, the model proposed performs an intelligent reading of users' vital signs: growing up the frequency when detecting the deterioration of the vital signs. As future work, we intend to decrease the periodicity for checking vital signs to pre-diagnose more health conditions.

ACKNOWLEDGMENTS

This research was partially funded by CAPES, FAPERGS and CNPq.

REFERENCES

[1] Gianluca Aloi, Giancarlo Fortino, Raffaele Gravina, Pasquale Pace, and Claudio Savaglio. 2021. Simulation-Driven Platform for Edge-Based AAL Systems. *IEEE Journal on Selected Areas in Communications* 39, 2 (Feb 2021), 446–462. https://doi.org/10.1109/JSAC.2020.3021544

[2] Talita Alves, Paulo Rettore, and Bruno Santos. 2023. A Mobility Model of The Internet of Things. In *Anais do XXIX Simpósio Brasileiro de Sistemas Multimídia e Web* (Ribeirão Preto/SP). SBC, Porto Alegre, RS, Brasil, 221–229. https://sol.sbc.org.br/index.php/webmedia/article/view/25883

[3] Gina R. Bai, Justin Smith, and Kathryn T. Stolee. 2021. How Students Unit Test: Perceptions, Practices, and Pitfalls. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1* (Virtual Event, Germany) *(ITiCSE '21)*. Association for Computing Machinery, New York, NY, USA, 248–254. https://doi.org/10.1145/3430665.3456368

[4] Riccardo Berta, Ahmad Kobeissi, Francesco Bellotti, and Alessandro De Gloria. 2021. Atmosphere, an Open Source Measurement-Oriented Data Framework for IoT. *IEEE Transactions on Industrial Informatics* 17, 3 (March 2021), 1927–1936. https://doi.org/10.1109/TII.2020.2994414

[5] Mathieu Bourgais, Franco Giustozzi, and Laurent Vercouter. 2021. Detecting Situations with Stream Reasoning on Health Data Obtained with IoT. *Procedia Computer Science* 192 (2021), 507–516. https://doi.org/10.1016/j.procs.2021.08.052 Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 25th International Conference KES2021.

[6] Vagner Figueredo de Santana and Leandro Marega Ferreira Otani. 2021. Measuring Quantitative Situated User Experience with a Mobile Galvanic Skin Response Sensor. In *Proceedings of the XX Brazilian Symposium on Human Factors in Computing Systems* (Virtual Event, Brazil) *(IHC '21)*. Association for Computing Machinery, New York, NY, USA, Article 37, 7 pages. https://doi.org/10.1145/3472301.3484339

[7] Gabriel Souto Fischer, Gabriel de Oliveira Ramos, Cristiano André da Costa, Antonio Marcos Alberti, Dalvan Griebler, Dhananjay Singh, and Rodrigo da Rosa Righi. 2024. Multi-Hospital Management: Combining Vital Signs IoT Data and the Elasticity Technique to Support Healthcare 4.0. *IoT* 5, 2 (2024), 381–408. https://doi.org/10.3390/iot5020019

[8] Iqram Hussain and Se Jin Park. 2021. Big-ECG: Cardiographic Predictive Cyber-Physical System for Stroke Management. *IEEE Access* 9 (2021), 123146–123164. https://doi.org/10.1109/ACCESS.2021.3109806

[9] Abdul Jaleel, Tayyeb Mahmood, Ahsen Tahir, Shehzad Aslam, and Ubaid Ullah Fayyaz. 2022. Autonomic interoperability manager: A service-oriented architecture for full-stack interoperability in the Internet-of-Things. *ICT Express* 8, 4 (2022), 507–512. https://doi.org/10.1016/j.icte.2021.10.010

[10] Dimitrios Markopoulos, Anastasios Tsolakidis, Nikitas Karanikolas, Aikaterini Marinagi, and Christos Skourlas. 2024. Applying Soft System Methodology for a clearer understanding of the future Intensive Care Units. In *Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics* (, Lamia, Greece,) *(PCI '23)*. Association for Computing Machinery, New York, NY, USA, 163–170. https://doi.org/10.1145/3635059.3635084

[11] Microsoft. 2022. Design a DDD-oriented microservice. Available at: <https://learn.microsoft.com/en-us/dotnet/architecture/microservices/microservice-ddd-cqrs-patterns/ddd-oriented-microservice>. Accessed in: 25 July 2023.

[12] Naif Al Mudawi. 2022. Integration of IoT and Fog Computing in Healthcare Based the Smart Intensive Units. *IEEE Access* 10 (2022), 59906–59918. https://doi.org/10.1109/ACCESS.2022.3179704

[13] Saroj Kumar Nanda, Sandeep Kumar Panda, and Madhabananda Dash. 2023. Medical supply chain integrated with blockchain and IoT to track the logistics of medical products. *Multimedia Tools and Applications* (04 Mar 2023). https://doi.org/10.1007/s11042-023-14846-8

[14] Osama Rehman, Zaroon Farrukh, Asiya Al-Busaidi, Kyungjin Cha, Simon Park, and Ibrahim Rahman. 2021. IoT Powered Cancer Observation System.. In *The 9th International Conference on Smart Media and Applications* (Jeju, Republic of Korea) *(SMA 2020)*. Association for Computing Machinery, New York, NY, USA, 313–318. https://doi.org/10.1145/3426020.3426111

[15] Honnesh Rohmetra, Navaneeth Raghunath, Pratik Narang, Vinay Chamola, Mohsen Guizani, and Naga Rajiv Lakkaniga. 2021. AI-enabled remote monitoring of vital signs for COVID-19: methods, prospects and challenges. *Computing* (29 Mar 2021). https://doi.org/10.1007/s00607-021-00937-7

[16] Oshani Seneviratne. 2024. Enabling Data Interoperability for Decentralized, Smart, and Connected Health Applications. In *Proceedings of the 8th ACM/IEEE International Conference on Connected Health: Applications, Systems and Engineering Technologies* (, Orlando, FL, USA,) *(CHASE '23)*. Association for Computing Machinery, New York, NY, USA, 214–215. https://doi.org/10.1145/3580252.3589433

[17] Fan Wu, Chunkai Qiu, Taiyang Wu, and Mehmet Rasit Yuce. 2021. Edge-Based Hybrid System Implementation for Long-Range Safety and Healthcare IoT Applications. *IEEE Internet of Things Journal* 8, 12 (June 2021), 9970–9980. https://doi.org/10.1109/JIOT.2021.3050445