Specification of Quality of Context Requirements for Digital Phenotyping Applications

Luís Eduardo Costa Laurindo(✉), Ivan Rodrigues de Moura, Luciano Reis Coutinho, and Francisco José da Silva e Silva

Laboratory of Intelligent Distributed Systems (LSDi), Federal University of Maranhão (UFMA), 65080-805 São Luís, Brazil
{luis.laurindo,ivan.rodrigues,fssilva}@lsdi.ufma.br, luciano.rc@ufma.br

Abstract. Digital phenotyping applications use sensor data from personal digital devices (e.g., smartphones, smart bands) to quantify moment-to-moment human phenotype at the individual in-situ level. Ensuring the quality and distribution of the data used is essential requirement in the domain of these applications. Context Quality (QoC) refers to the Information Quality (QoI) used and the Quality of Service (QoS) level of information distribution. QoI is measured by parameters that define how reliable the information is. On the other hand, QoS is provided by specifying the quality of service for distributing context data. Some aspects can degrade the QoC of the application, such as information from sensors being imprecise, wireless communication technologies used in the acquisition and distribution of information, scalability problems can cause information delay, and intermittent connection due to user mobility can result in data loss. Therefore, this study conceives a process for incorporating QoC requirements and a Domain-Specific Language (DSL) to specify these requirements in digital phenotyping applications. A case study was carried out where the scenario of an application for monitoring workers' health was considered. It was possible to prove the expressiveness and simplicity of the proposed language when using it to define the instances of the application classes responsible for the acquisition and distribution of context information.

Keywords: Digital Phenotyping · Acquisition and Distribution · Quality of Context (QoC) · Incorporation of QoC Requirements · Domain-Specific Language

1 Introduction

Smartphones and wearable devices are part of people's daily lives and have sensors that capture the user's context, and environment [13,18]. Computational methods can use this context information to make inferences about social, behavioral, and cognitive aspects of individuals [12,14]. For example, it is possible

A. Tsanas and A. Triantafyllidis (Eds.): PH 2022, LNICST 488, pp. 651–665, 2023.
https://doi.org/10.1007/978-3-031-34586-9_43

to identify if the user is performing some physical activity using location and acceleration data or if he is carrying out a conversation based on the data captured by the smartphone's microphone.

Digital Phenotyping is a research area that aims to use context data from mobile devices to infer the individual's health status [19]. Research in digital phenotyping focuses on developing new solutions to complement and extend traditional sources of clinical data. Digital phenotyping solutions need to ensure an acceptable level of data quality to obtain greater decision-making accuracy due to their applicability in the healthcare field. Quality of Context (QoC) refers to the Quality of Information (QoI) used as context information and the Quality of Service (QoS) for distributing this information [2]. With the resources offered by QoC, it is possible to establish quality contracts between service providers and consumers, select better context sources, increase application efficiency by optimizing energy and bandwidth consumption by adapting the frequency of information dissemination and improving the quality of the user experience [5].

In the literature, one can find several parameters related to QoC to specify the application quality requirements [8,10]. For example, in an application that makes decisions in near real-time, the age parameter is crucial, as outdated information may not represent the individual's current context. Furthermore, by distributing this information through the smartphone with battery limitations and an increase in energy consumption when sending data over the network, it is soon possible to specify the desired frequency for distributing this information.

Digital phenotyping applications run in environments that can degrade QoC. First, they use sensor data which can generate inaccurate and erroneous readings [6]. Second, they deal with intermittent connection by allowing individuals to move and situations where the current network does not meet application requirements, resulting in data loss. Finally, a digital phenotyping infrastructure allows the monitoring a set of individuals, and problems related to scalability may occur. Therefore, it is necessary to specify QoC parameters to meet the requirements of each application in order to evaluate the context information used in them. In addition to evaluating the information, it is necessary to evaluate the quality of the distribution service. It is also critical to monitor compliance with these requirements to resolve potential issues that degrade the QoC of the application.

When designing these applications, it is necessary to develop or use software platforms to perform the acquisition, inference, and distribution of context information. Digital phenotyping is currently a very active research area [11,12]. Regarding middleware platforms and several application proposals were presented, such as [1,7,15,19,20]. However, the treatment of QoC in the field of digital phenotyping is still incipient in the literature, especially in middleware platforms focused on this area. Therefore, when considering the importance and requirements of these applications, this study conceives an approach that has a process for incorporating QoC requirements and a Domain-Specific Language (DSL) for specifying these requirements in digital phenotyping applications.

The article is divided as follows: Sect. 2 presents the theoretical foundation; the 3 section exposes related works; Sect. 4 presents the proposed solution;

Sect. 5 presents a case study; and finally, Sect. 6 presents the conclusions and future works.

2 Background

2.1 Digital Phenotyping

Torous et al. [19] define Digital Phenotyping as the "moment-by-moment quantification of human phenotype at the individual level in-situ using data from smartphones and other personal digital devices." Personal digital devices are present in the individual's daily life (for example, smartphones, smartbands) that collect a set of data on behavioral and health aspects useful in the domain of these [14] applications.

According to Mendes et al. [11] the digital phenotyping process (illustrated in Fig. 1) begins with the collection of raw data from sensors, whether physical (e.g., GPS, accelerometer, heart rate) or virtual (e.g., phone calls, screen time on, apps used). After collecting the raw data, behavioral and health events are inferred. Behavioral events represent actions performed by the individual (e.g., the time interval in which he socialized with the family). Health events represent the individual's physiological state based on vital signs (e.g., heart rate, blood pressure, blood oxygenation). After the inference of these events, behavioral patterns are obtained (e.g., the pattern of mobility, sociability, and physical activity). These patterns refer to routine situations of the individuals (e.g., the individual always sleeps, from Monday to Friday, at 23:00). Finally, these behavioral patterns are used in prediction and diagnostic applications.

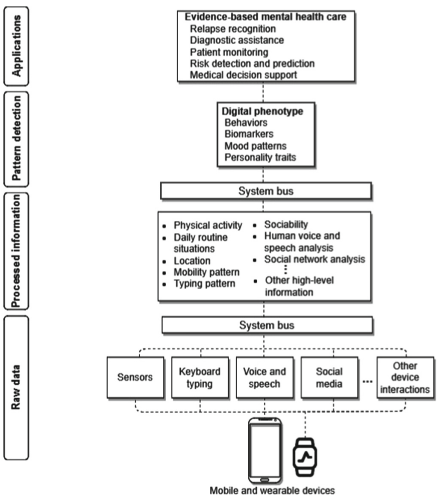

Fig. 1. The process of digital phenotyping [11].

The development of digital phenotyping applications requires components responsible for collecting context data from sensors embedded/connected to personal digital devices. In addition, developers must implement software components to infer behavioral events and distribute collected and inferred data to external servers via wireless communication infrastructure. Finally, on the server, it is necessary to implement components for managing large volumes of data and machine learning models to infer behavioral patterns related to health aspects. [11].

2.2 Quality of Context

Context is the information used to characterize the situation of an entity (e.g., people, objects, or places) that influence an agent's decisions [4]. Below are described just some of the various parameters present in the literature used to quantify the degree of information quality, in addition to allowing the definition of the quality of the distribution service [8,10].

The **Accuracy** represents an estimate of how close the context information is to the actual value. The device manufacturer normally provides the accuracy value when the information comes from a physical sensor. The **Confidence** estimates the degree of certainty of the information provided by the information source. The **Measurement Interval** indicates the time interval between successive readings. The **Delay** represents the time elapsed between sending the message by the producer and its arrival at the consumer. Even though it is a QoS parameter, it is possible to insert the computed time in the context data as a meta-information related to QoI. **Completeness** indicates how complete is the context information received by a consumer. It can be calculated based on the ratio between the sum of the weights of the available attributes and the sum of the weights of the attributes required by the consumer. The **Validity Time** indicates the validity of the information. The information producer can specify the validity of the information as he has produced it. However, the consumer can decide whether or not to accept the shelf life specified by the producer. The **Age** can also be used to check the validity time of the information. It indicates the difference between the current instant and the information measurement time. Finally, the **Total Delivery Time** indicates the time from the measurement of the information to its delivery to the consumer.

To define the quality level of the information distribution service, we have the **Reliability** that determines whether the distribution service should adopt the best effort policy (best-effort), when there is no guarantee of delivery of information, or use delivery and retransmission if the context information is not delivered to the recipient. The **Refresh Rate** allows the consumer to define how often context information should be received regardless of how often the data is produced. The **Delivery Time** indicates the maximum time a consumer is willing to wait for information. The **Latency Control** defines an additional delay to the producer and consumer of the information. By setting a delay, messages are grouped into a queue and sent or received in a single burst. The **History** allows the consumer to store the information for some time feasible for him. By using this parameter **Order of Destination**, the messages stored in the History can be organized according to their timestamp of publishing or receiving. The consumer can still specify the validity time of the information with the parameter **Lifetime**. Through this parameter, the consumer defines when messages should be removed from the History. **Retention** allows the information producer to indicate that he wants to retain the last message sent to new consumers as they arise. Finally, **Liveness** allows producers to send consumers the status of their service, indicating whether they are still active. The consumer must indicate whether he wants to receive this alert.

3 Related Work

We used as related works the studies selected in the Systematic Literature Review (RSL) conducted by Mendes et al. [11]. The study authors presented software platforms designed to support digital phenotyping studies. Based on this review, we present in this section works that conceive software platforms to perform the acquisition and distribution of context information.

AWARE [7] is a reusable Android software platform focusing primarily on the acquisition, distribution, and context inference. The platform provides a library that allows the developer to design their application. Its architecture is composed of the Aware Client and Aware Server layers. The Aware Client layer is responsible for acquiring context information from physical and virtual sensors. After the acquisition, information is processed, and high-level situations are inferred. Processing is carried out through plugins. Each plugin is responsible for some situation of interest (e.g., sociability, mobility, physical activity, sleep). The Aware Server layer has a cloud database and dashboard capabilities for information visualization. The information presented in real time is sent via the Message Queuing Telemetry Transport (MQTT)[1] protocol. The study addresses some measures to minimize data loss. The first is to use MQTT's QoS requirements, and the second is to prevent data collection processes from being interrupted by Android.

SituMan [17] is a reusable software platform that identifies situations in the individual's routine based on data from smartphone sensors. Situation inference is used to solicit self-reports at opportune times. It allows collection related to the location and activity that the individual is performing. The authors recognize that the sensor data can sometimes generate inaccurate data, as in the case of the GPS used by the authors in the study, but the solution does not address the provision of QoC mechanisms. When using inaccurate location information, it may not represent the exact location where the individual is.

Beiwe [19] is a platform that collects raw data from sensors and smartphone usage aspects. Two main components make up Beiwe: a web app and a mobile app. The web application allows researchers to specify the application's content, the sensors used in data collection, and its refresh rate. The mobile application is responsible for acquiring and distributing context information. They are stored in

[1] https://mqtt.org/.

a buffer to be sent later. Buffering information is one of the important QoS requirements, as, at certain times, the application may not have an internet connection, so this information is stored to be sent later when establishing the connection.

Purple Robot [15] is a framework that supports the creation of mobile applications which collect data from sensors through an authoring tool web that helps researchers who do not know software to develop their applications to acquire data from smartphone sensors to carry out their studies.

Funf [1] provides a set of functionality that allows the collection and distribution of context information. Funf has a Funf Manager component responsible for selecting the sensors used to collect context information and configuring the data collection frequency to minimize battery consumption. The possibility of changing the data sampling rate is relevant when considering device battery limitations and application bandwidth consumption. The solution also has a buffer where data is temporarily stored. The Funf sent this data every three hours to a server in the cloud. Being able to temporarily store information in a buffer is also a QoS requirement, as constantly accessing the network can consume increasing device power consumption.

Finally, Sensus is a tool that consists of two [20] mobile applications. The first one is responsible for collecting data from sensors from the monitored individual's smartphone. A server processes the collected data in the cloud. The other application is used by the healthcare professional to manage the study. The study is managed by a protocol designed by the professional. The protocol contains the information necessary to apply the study and the sensor data collected from the subject's smartphone.

3.1 Considerations

Analyzing the related works, we identified that the platforms designed to acquire and distribute context information within the scope of digital phenotyping do not broadly address the QoC requirements necessary for these applications. However, the treatment of QoC in the domain of digital phenotyping is still incipient in the literature, particularly in middleware platforms focused on this area.

Each application may require different levels of QoC. Information that does not meet the requirements required by the application may be useless for the context. Therefore, this study contributes to the literature in proposing mechanisms that facilitate the specification of QoC requirements in digital phenotyping applications, considering both QoI and QoS aspects.

4 Proposed Solution

4.1 Process

The proposed solution has a process for incorporating QoC requirements in developing these applications and monitoring their execution. Figure 2 presents the steps of this process. The process consists of five steps, namely: specification, transformation, implementation, evaluation and monitoring, and finally, visualization.

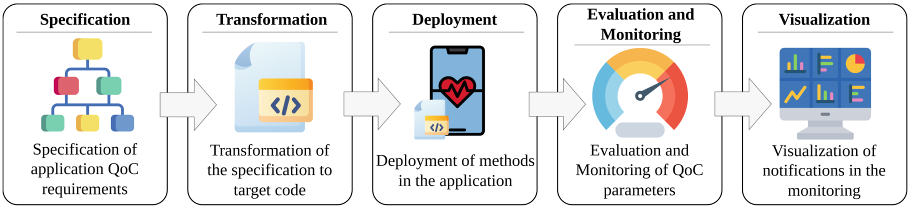

Fig. 2. Process for incorporating QoC requirements into digital phenotyping applications.

The first step is to specify the application's QoC requirements with the help of a domain language. A DSL is conceptualized as a set of models, defined by a metamodel, that corresponds with an abstract syntax and is represented by one or more concrete syntaxes [9]. After formalizing the requirements, the transformation to the target code occurs automatically through a conversion process. After the transformation stage, the developer, in possession of the artifact generated based on the DSL, can use it in the design of his application by importing the generated source code into his project. Next, in the evaluation and monitoring stage, the context information is selected based on the specified QoC parameters, and event logs are generated regarding fulfilling the requirements. Finally, the visualization step involves projecting these data events onto the Dashboard tool.

4.2 Proposed Metamodel

For the design of the DSL, a problem domain analysis was first performed to identify the concepts, abstractions, and relationships between the entities. This domain analysis produced an abstract syntax that corresponds to a metamodel. The proposed metamodel was architected based on the Eclipse Modeling Framework[2] (EMF) pattern. EMF consists of a modeling framework based on a data model with code generation capabilities. Figure 3 presents the abstract syntax of the proposed DSL. She defines all identified concepts and their respective relationships. The concepts must be described to be easily understood by the user. The abstract syntax also includes structural metamodel semantics to define the rules and constraints of relationships between domain classes.

In digital phenotyping, applications consume context information from, for example, sensors. In addition, these applications distribute this information to other applications. Consumption and dissemination of context information are defined in the metamodel as services. Each service is associated with one or more context information. Each context information comprises a specific type of data (e.g., heart rate), a unit of measurement (e.g., bpm), and the source precision value.

[2] https://www.eclipse.org/modeling/emf/.

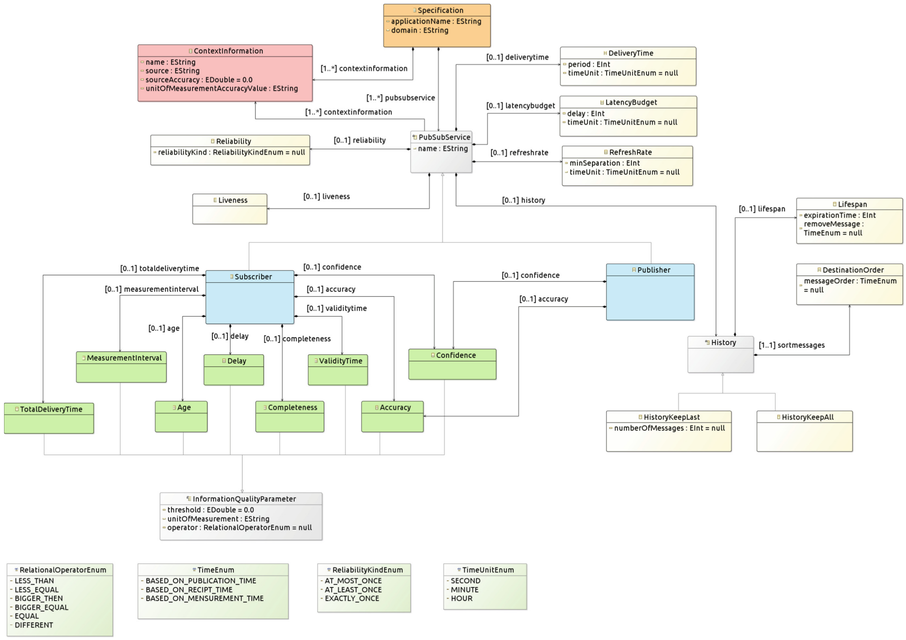

Fig. 3. Proposed metamodel.

For each service, it is possible to associate different QoC parameters. The QoI parameters, highlighted in green, are specified to ensure that the information received or sent will have a certain degree of quality required by the application. When defining a QoI parameter, it is necessary to define a threshold value, its equivalent unit of measure, and a relational operator. The relational operator is used to compare the value of the QoI parameter contained in the information with the specified value. For this, the context information must be annotated with the QoI meta-information.

The QoS parameters, highlighted in yellow, are specified to guarantee the distribution service quality. For example, it is possible to set the reliability level of data delivery by specifying the Reliability parameter and its type. By setting it to At Least Once, the service disseminates the information and will receive a puback to confirm receipt. For the Exactly Once type, a handshake is performed. This action is required to confirm the delivery of the information. If there is no such confirmation, the context information is retransmitted. It is also possible to set the QoS level when consuming the context information. For example, we can set the service that receives data from the heart rate sensor to have a Refresh Rate of 1 s.

4.3 Transformation and Incorporation of QoC Requirements

To incorporate the specified QoC requirements, it is necessary to use Middleware platforms with the mechanisms to implement the requirements in the proposed metamodel. Therefore, the process of transforming the model into Middleware-specific source code begins with the creation of code generator modules. Figure 4 presents the process necessary to carry out the transformation of the specification into the target code.

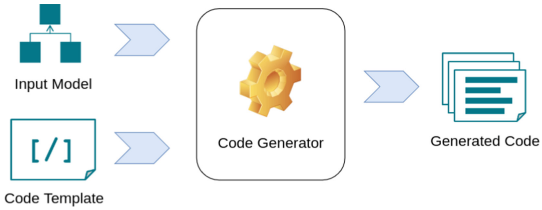

Fig. 4. Target code generation process for Middleware.

Implementing the MOF Model to Text Language (MTL) standard can be used to transform an EMF model into target code. The Acceleo tool implements the MTL standard and comprises two main structures: models and queries. Templates have a set of Acceleo instructions for generating text. The queries are performed using the Acceleo Query Language (AQL) and are responsible for extracting information from the EMF specification. After generating the destination codes referring to the services that consume (Subscribers) and distribute (Publishers) context information, the developer can incorporate them into the application.

5 Case Study

5.1 Worker Health Monitoring Mobile System

Mobile health monitoring systems are applications that run on mobile devices and aim to infer users' health status based on information derived primarily from sensors. These applications use wireless communication technologies to acquire data from wearable devices of monitored users and disseminate this information. As a case study, we implemented a mobile app that monitors workers' vital signs and physical activity during working hours.

We designed a worker health monitoring app to connect devices with Bluetooth communication technology. In addition to performing the collection, the application infers situations about the worker's health status and disseminates this information through the MQTT protocol to consumer applications. One is a SpringBoot[3] application that runs on a cloud server. It is responsible for

[3] https://spring.io/projects/spring-boot.

receiving the data by storing it. Another application consists of a Dashboard that presents to the health professional at a time close to the worker's current state of health. The Dashboard also allows the professional to consult historical data. Figure 5(a) shows the Polar H10 device connected to the mobile application. Figure 5(b) shows that the mobile application collects data from heart rate, electrocardiogram, and activity sensors. Finally, Fig. 6 shows the Dashboard presenting the data in real-time.

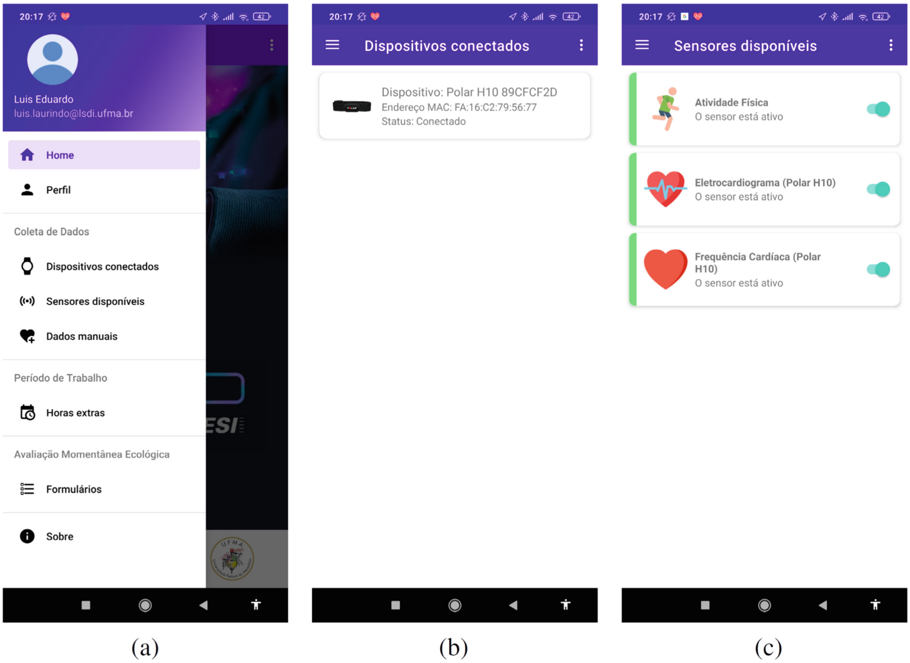

Fig. 5. (a) Screen that shows options menu. (b) Screen that shows connected devices. (c) Screen showing available and active sensors.

The CDAL/CDDL Middleware platform was used for data acquisition and distribution. It is an extension of the M-Hub/CDDL Middleware [8,16], designed to facilitate the development of Internet of Things (IoT) applications with QoC requirements. Middleware is composed of two layers, they are: the Context Data Acquisition Layer (CDAL) and the Context Data Distribution Layer (CDDL). The CDAL layer runs on Android mobile devices to acquire context data from Smartphone and smart personal devices that have technologies such as Bluetooth Classic (BT) and Bluetooth Low Energy (BLE). On the other hand, the CDDL is responsible for processing and distributing the context data obtained through the CDAL and can be executed on Android devices, desktop applications and cloud servers. It provides developers of mobile applications, which use data from

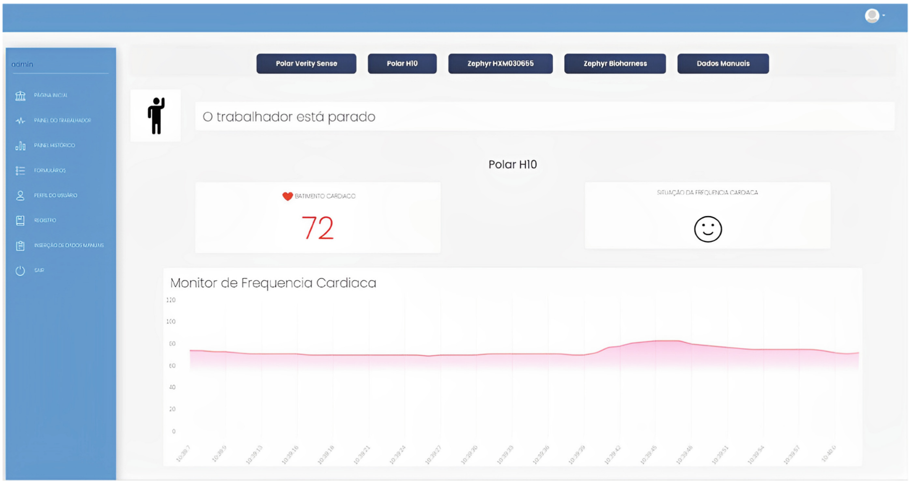

Fig. 6. Presentation of near real-time data through the dashboard.

sensors from smartphones and wearable devices, mechanisms to ensure QoC requirements. The developer must explicitly program QoC requirements.

5.2 System Requirements

Digital phenotyping applications run in environments that have characteristics that can degrade QoC. Therefore, when designing these applications, it is necessary to consider some aspects to ensure their quality. The following describes some requirements that these continuous monitoring applications must have. We incorporated all the mentioned requirements in the application developed in this case study.

- **Identify active sensors:** The system must identify when a sensor is no longer active.
- **Inference from situations:** The system must provide information related to the worker's health status based on the sensors' vital signs information and activities performed, coming from sensors with a certain degree of imprecision.
- **Mobility support:** The system must guarantee data delivery even considering intermittent connections.
- **Resource-saving:** The system should try to minimize battery consumption and network interface usage of devices used by the worker when performing context data collection and distribution.

5.3 Specification of QoC Requirements

After identifying the requirements necessary to guarantee the quality of the application, the specification of the QoC requirements was carried out based on the parameters contemplated by the proposed metamodel. Figure 7 presents, in summary, how the application's QoC requirements were specified.

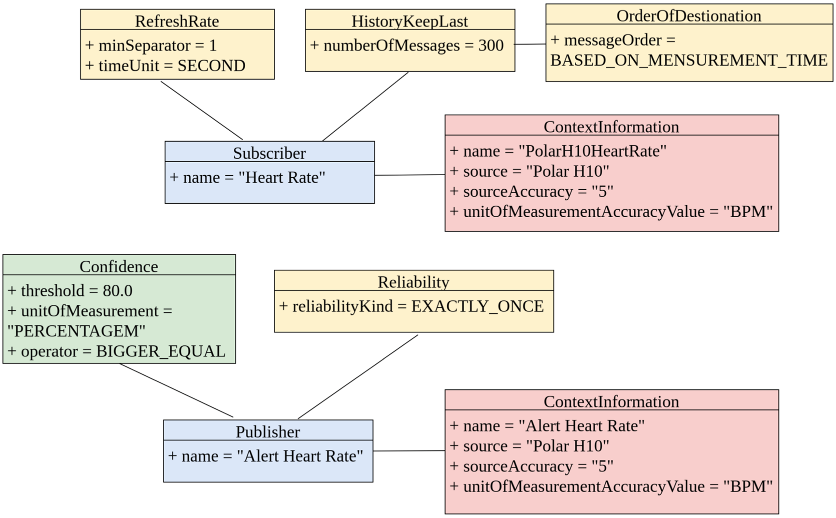

Fig. 7. Specification of QoC requirements for services that consume (Subscriber) heart rate information and disseminate (Publisher) heart rate alert information.

The monitoring application allows connection to devices that have Bluetooth technology. Bluetooth has a distance limit to keep an active connection. In the monitoring environment, in which the monitored user can move away from the smartphone that receives data from the device, it is necessary to notify consumer applications about the status of these services. The application developed in the case study publishes context information from several sensors: heart rate, breathing, blood pressure, and oxygen saturation. For each sensor, the Liveness parameter was defined. This parameter indicates that the service that publishes the sensor information will notify the consuming applications, informing them if they are still active or not.

Health professionals accompanying workers during working hours want to receive near real-time alerts when standard vital signs such as normal, altered, or critical ranges are detected. These alerts are essential to provide the professional with an indicator of the worker's current state of health for a quick reaction depending on the case's urgency. They are generated based on information from physical sensors. As the information from these sensors has a certain degree of imprecision, the information derived has a certain degree of confidence. We calculate the confidence of the alerts via an algorithm that measures the degree of confidence of a situation inferred based on the imprecision of the information source presented in [3]. In addition, the application provides an alert for heart rate (AlertHeartRate) and breathing rate (AlertBreathingRate), as these data are collected continuously. As specified in the requirements, to avoid false alerts about the actual situation of the worker's health status, the mobile application disseminates only the alerts with a degree of 80% confidence to the dashboard. It should be noted that continuous information must be generated by the sensor every 1 s, as specified by the RefreshRate parameter, for quick decision-making.

The mobile monitoring application uses the Activity Recognition API[4] developed by Google to identify the worker's activity. This API, in addition to providing the activity, also provides the confidence level of the situation. To ensure that the activity provided really matches the activity performed by the worker, a threshold of 80% was defined for the confidence parameter. As a result, the monitoring application will only receive situations with a confidence level of 80%.

Worker mobility can cause intermittent connection. This is a problem when you have essential items that must be delivered to a consumer application, such as alerts and information that is not collected continuously (e.g., blood pressure, glucose, and oxygen saturation). It is worth mentioning that a conventional sensor collects the blood glucose data, and the information is manually entered into the application. As specified in the requirements, the Reliability parameter is defined for this information to guarantee the delivery of the data. Each service that consumes context information has a History to store the information when there is no established internet connection or server connection. The Historic performs the function of a Buffer. Upon reestablishing the connection, the service disseminates the stored information.

Mobile devices have energy limitations due to the use of batteries that need to be recharged. The health monitoring system uses the user's smartphone to collect, infer and distribute information. In a standard data distribution model, the information is sent to the measure made available by the sensors. With each new data, network access is requested to send it. As per the Android documentation, each request to the network interface generates a reasonable power consumption. Therefore, for the worker monitoring application, a LatencyBudget parameter was defined that defines a delay for sending the data in a grouped way. As specified in the requirements, the delay is defined as 60 s, that is, the application will group the data generated in this interval and time and send them. This results in the number of network interface request the application makes.

6 Conclusion and Future Work

This study proposed a process to incorporate QoC requirements in digital phenotyping applications. The process has five steps, they are: specification of requirements, the transformation of the specification into target code, deployment of the code in the application, evaluation, and monitoring of the requirements, and finally, the visualization of the monitoring logs in a dashboard tool. From the proposed process, the study conceives a metamodel used to specify QoC requirements considering the structure of digital phenotyping applications in acquiring and distributing context information.

[4] https://developers.google.com/location-context/activity-recognition.

We have developed a system with QoC requirements for monitoring workers during working hours. It is worth mentioning that the specification transformation process was performed manually by the developers. The platform has two main applications. The first is a mobile application that collects signal and activity information from workers. The second is a dashboard that provides health professionals with near real-time information and historical data on the worker's health status. Through the case study, it was possible to observe that based on the proposed metamodel, it is possible to formalize QoC requirements that guarantee the quality of applications for digital phenotyping in the health field. In future works, we propose the automatic transformation of the specification into target code, evaluation and monitoring of the specified requirements, and dealing with conflicting situations between the specified QoC requirements.

Acknowledgments. This work was supported by the Coordination for the Improvement of Higher Education Personnel - Brazil (CAPES - Finance code 001); INCT of the Future Internet for Smart Cities (CNPq 465446/2014-0, CAPES 88887.136422/2017-00, and FAPESP 14/50937-1 and 15/ 24485-9); National Council for Scientific and Technological Development - CNPq (311608/2017-5, 420907/2016-5, 312324/2015-4); and the State of Maranhão Research Funding Agency - FAPEMA (via grant BM QUOTA UFMA - AGREEMENT N° 01/2021 FAPEMA/UFMA).

References

1. Aharony, N., Pan, W., Ip, C., Khayal, I., Pentland, A.: Social fMRI: investigating and shaping social mechanisms in the real world. Pervasive Mob. Comput. **7**(6), 643–659 (2011)
2. Bellavista, P., Corradi, A., Fanelli, M., Foschini, L.: A survey of context data distribution for mobile ubiquitous systems. ACM Comput. Surv. (CSUR) **44**(4), 1–45 (2012)
3. Bezerra, E.D.C., Teles, A.S., Coutinho, L.R., da Silva e Silva, F.J.: Dempster-shafer theory for modeling and treating uncertainty in IoT applications based on complex event processing. Sensors **21**(5), 1863 (2021)
4. Bolchini, C., et al.: And what can context do for data? Commun. ACM **52**(11), 136–140 (2009)
5. Buchholz, T., Küpper, A., Schiffers, M.: Quality of context: what it is and why we need it. In: Workshop of the HP OpenView University Association (2003)
6. Cho, S., Ensari, I., Weng, C., Kahn, M.G., Natarajan, K.: Factors affecting the quality of person-generated wearable device data and associated challenges: rapid systematic review. JMIR Mhealth Uhealth **9**(3), e20738 (2021)
7. Ferreira, D., Kostakos, V., Dey, A.K.: AWARE: mobile context instrumentation framework. Front. ICT **2**, 6 (2015)
8. Gomes, B.D.T.P., et al.: A middleware with comprehensive quality of context support for the internet of things applications. Sensors **17**(12), 2853 (2017)
9. Harel, D., Rumpe, B.: Modeling languages: Syntax, semantics and all that stu. N/A n/a, pp. 1–28 (2000)
10. Jagarlamudi, K.S., Zaslavsky, A., Loke, S.W., Hassani, A., Medvedev, A.: Requirements, limitations and recommendations for enabling end-to-end quality of context-awareness in IoT middleware. Sensors **22**(4), 1632 (2022)

11. Mendes, J.P., et al.: Sensing apps and public data sets for digital phenotyping of mental health: systematic review. J. Med. Internet Res. **24**(2), e28735 (2022)
12. Moura, I., et al.: Mental health ubiquitous monitoring supported by social situation awareness: a systematic review. J. Biomed. Inform. **107**, 103454 (2020)
13. Ometov, A., et al.: A survey on wearable technology: history, state-of-the-art and current challenges. Comput. Netw. **193**, 108074 (2021). https://doi.org/10.1016/j.comnet.2021.108074, https://www.sciencedirect.com/science/article/pii/S1389128621001651
14. Saccaro, L.F., Amatori, G., Cappelli, A., Mazziotti, R., Dell'Osso, L., Rutigliano, G.: Portable technologies for digital phenotyping of bipolar disorder: a systematic review. J. Affect. Disord. **295**, 323–338 (2021)
15. Schueller, S.M., Begale, M., Penedo, F.J., Mohr, D.C.: Purple: a modular system for developing and deploying behavioral intervention technologies. J. Med. Internet Res. **16**(7), e3376 (2014)
16. Silva, M., et al.: Neighborhood-aware mobile hub: an edge gateway with leader election mechanism for internet of mobile things. Mobile Netw. Appl. **27**(1), 276–289 (2020). https://doi.org/10.1007/s11036-020-01630-3
17. Teles, A.S., et al.: Enriching mental health mobile assessment and intervention with situation awareness. Sensors **17**(1), 127 (2017)
18. Statista: Number of smartphone subscriptions worldwide from 2016 to 2027 (2022). https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/. Accessed 13 Apr 2022
19. Torous, J., Kiang, M.V., Lorme, J., Onnela, J.P., et al.: New tools for new research in psychiatry: a scalable and customizable platform to empower data driven smartphone research. JMIR Ment. Health **3**(2), e5165 (2016)
20. Xiong, H., Huang, Y., Barnes, L.E., Gerber, M.S.: Sensus: a cross-platform, general-purpose system for mobile crowdsensing in human-subject studies. In: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 415–426 (2016)