MyHealth: A System for Monitoring Non-communicable Diseases

58

Ronitti Juner da Silva Rodrigues, Ivo Palheta Mendes, and Wanderley Lopes de Souza

Abstract

Noncommunicable Diseases (NCDs) represent one of the greatest health challenges and cause global concern due to the high costs and the high number of deaths. It has a major impact on Brazil's public health system and affects all socioeconomic levels, mainly the poor and the elderly. Among the various risk factors, a patient's lifestyle may lead to complications in their health. Therefore, to prevent and control NCDs it is necessary to promote changes in lifestyle, and to monitor patients. For certain NCDs such as hypertension and diabetes, patients can actively participate in the monitoring process by recording clinical data measured through appropriate equipment. We present in this paper MyHealth, a system that aims to continuously monitor NCDs' patients, supporting in this way the decision-making by health professionals. This system can contribute to reduce the number of physician's appointments, to increase the treatment accuracy, and to provide a better quality of life.

Keywords

Pervasive & Ubiquitous Computing · E-health care and technology · Telemedicine · Patients' monitoring · Non-communicable diseases

58.1 Introduction

Noncommunicable Diseases (NCDs) are one of the major health challenges, and has caused global concern due to its magnitude and high social cost [1]. The four major groups, cardiovascular disease, cancer, diabetes, and chronic respiratory diseases, have caused about 70% of deaths worldwide [2]. In addition, around 15 million people between the ages of 30 and 70 die each year [3].

In Brazil, NCDs account for about 73% of total deaths [2]. They represent a direct impact on the health system, and are among the main causes of hospitalizations [3]. All socioeconomic levels are affected, especially the most vulnerable groups, such as the poor and the elderly [3].

Several risk factors contribute to the development of NCDs and the worsening of people's health. Some of these risk factors, such as physical inactivity, and unhealthy eating, can be changed since they are related to lifestyle [3]. For preventing and controlling NCDs, it is necessary to promote habit changes, and to provide a comprehensive care to patients [1].

Some NCDs, such as diabetes and hypertension, allow for the self-monitoring of patients. They can record their own clinical data, and provide them to the health professionals, e.g., during their physician's appointments, becoming in this way active participants in the healthcare process. In Brazil, this is a common practice, mainly when clinical data are involved, such as blood pressure, glucose levels, and weight.

In order to help in the healthcare of NCD patients, we developed MyHealth system by employing Ubiquitous Computing technologies. The NCD clinical data are collected by these patients using appropriated medical equipment, and recorded on their mobile devices. These data are sent automatically to the cloud for further treatment, and can be accessed and analyzed by health professionals, supporting in this way their decision-making related to the adjustments on treatment, and changes in the lifestyle of the NCD patients.

This paper is further structured as follows: Sect. 58.2 discusses some related works; Sect. 58.3 addresses the problem

R. J. da Silva Rodrigues (✉) · I. P. Mendes · W. L. de Souza
Department of Computing (DC), Federal University of São Carlos (UFSCar), São Carlos, SP, Brazil
e-mail: ronittijuner@ufscar.br; desouza@ufscar.br

S. Latifi (ed.), *17th International Conference on Information Technology–New Generations (ITNG 2020)*, Advances in Intelligent Systems and Computing 1134,
https://doi.org/10.1007/978-3-030-43020-7_58

investigated in this work, and gives an overview of MyHealth system; Sect. 58.4 describes the methodology employed for developing this system; Sect. 58.5 deals with the design, and implementation of MyHealth; and Sect. 58.6 presents our concluding remarks and gives recommendations for future work.

58.2 Related Work

The literature presents several works looking for solutions to monitor NCD patients. We selected four of them for reporting in this paper, and for comparing with our work.

A system is proposed in [4] for monitoring NCD patients, where clinical data are automatically collected through medical equipment with wireless capacities. These data are transmitted to a server, becoming available for analysis by health professionals. In addition, patients can ask for physician's appointments, and can send out a SOS for immediately care.

A framework based on ontologies is proposed in [5] for customizing a mobile application to monitor NCD patients. This customization involves parameters, such as the NCD type, patient preferences, monitoring procedure required by the physician, prescribed medication, and patient environment. All this information determines the data to be acquired from the patient, through questionnaires, and/or sensors.

A prototype based on Arduino, and on an Android application is proposed in [6] for monitoring patients with diabetes mellitus. A glucometer with Bluetooth capacity is employed to measure the glucose level, and to transfer it to a mobile application. This application is connected to a server that provides this clinical data for medical service purposes, or to be used during physician appointments.

An approach based on machine learning is proposed in [7] for diagnosing and monitoring heart disease through a mobile application. The clinical data are collected by sensors, and sent to the mobile application via Bluetooth. Then, they are relayed to the cloud for treatment, where classifiers are employed to determine the presence or absence of heart disease.

All these works aim to monitor patients with some specific NCD, and they employ sensors or medical equipment with wireless capacities to directly communicate with mobile devices. Since in Brazil these medical equipment are expensive, mainly for low-income populations, the Brazilian Unified Health System (SUS) provides some of them for free [8]. This was our main motivation for developing MyHealth, a system that allows the patients to manually collect NCD clinical data using SUS's equipment, and to store these data in their mobile device, which is a very popular technology in Brazil.

58.3 Problem Description

Since the NCDs are the most significant group of diseases in Brazil, the Brazilian Ministry of Health launched in 2011, in partnership with other institutions, the Strategic Action Plan to Tackle Noncommunicable Diseases [9]. This plan has three main foundations: surveillance, information, evaluation and monitoring; health promotion; and comprehensive care [9]. A set of actions and investments, for the prevention and control of NCDs, were defined to be applied up to 2022. One main goal of comprehensive care is to promote the continuous follow-up of NCD patients [3].

Ubiquitous Computing and Cloud Computing have been employed in Mobile Health (mHealth), mainly in the design of patient monitoring systems. According to the Brazilian Institute of Geography and Statistics (IBGE), in 2017 about 78.2% of the Brazilian population had a mobile phone, and about 74.9% of Brazilian households had internet access, where 97% of this access was done through mobile phone [10].

Considering the Brazilian public policies for prevention and follow-up of NCD patients, and the widespread use of mobile phones in Brazil, we developed the MyHealth system for assisting the monitoring of NCD patients. An overview of this system is illustrated in Fig. 58.1.

MyHealth is composed of three main modules: Mobile App, a mobile application that alerts the patient to collect his clinical data, allows the patient to register this data in his mobile device, and sends automatically the registered data to the cloud; API Server provides an Application Programming Interface (API) for storing and retrieving the patient clinical data in the cloud; Web App provides and interface for accessing and visualizing the patient clinical data in the cloud.

The patient is responsible for collecting his clinical data using the medical equipment provided by SUS (e.g., pressure monitor, scale, and blood glucose monitor), and for recording

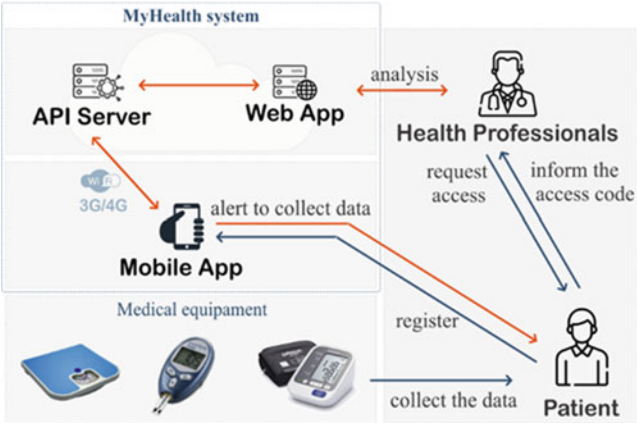


Fig. 58.1 Overview of MyHealth system

these data in the Mobile App. The health professionals (e.g., physician, physiotherapist, and nutritionist) are responsible for accessing the patient clinical data using the Web App, and for analyzing these data for decision-making regarding the patient health.

58.4 Development Methodology

MyHealth was developed according to Test-Driven Development (TDD), an agile software development method, based on short development cycles, in which automated tests are described previously to the production of code [11]. These tests specify and validate what the code must do [12].

Each TDD cycle is divided into three stages: write an automated test for a small functionality, and considering that this test was developed before the functionality code, it should be fail; then develop a minimum code in order for this functionality to pass the test; and refactor this functionality code for optimization purpose [13]. This cycle avoids code duplication, makes the code clearer, simple and bug-free [11]. Figure 58.2 illustrated this cycle with the *Collect Data* functionality of MyHealth, which allows the patient for recording his clinical data in the Mobile App.

As shown in Fig. 58.2, first a test for the persistence validation of the patient's clinical data in the database is written. This test should receive a valid identifier of the data registration in the database, and since the *Collect Data* functionality was not implemented yet, it will be fail. Then, a minimum code for the *Collect Data* functionality is produced for passing this test. Finally, this code is refactored for eliminating duplicates, and for increasing it readability and maintainability, always ensuring that it continues to pass this test.

58.5 MyHealth Design

MyHealth was designed in three main phases: *system specification*, where the system requirements were identified, analyzed, and formal described; *system architecture*, where the system was structure into modules; and *system implementation*, where the system functionalities were implemented and tested using TDD.

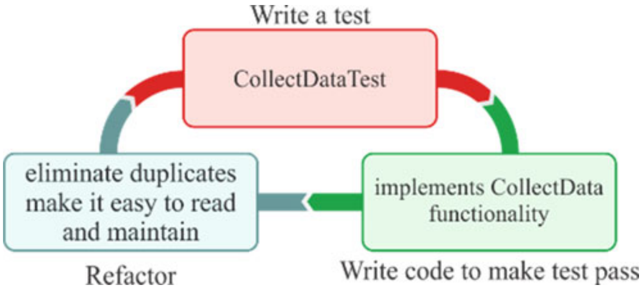

Write a test
CollectDataTest
eliminate duplicates make it easy to read and maintain
implements CollectData functionality
Refactor
Write code to make test pass

Fig. 58.2 TDD cycle for *Collect Data* functionality

58.5.1 System Specification

At this phase the system requirements were captured, and specified employing the Use Case diagram, and the Sequence diagram of Unified Modeling Language (UML). Figure 58.3 shows the Use Case diagram for Mobile App, focusing in the main actors, functionalities, and interactions of this application.

Figure 58.3 shows two actors interacting with Mobile App: *Patient* and *Background Service*. The first one represents the NCD patient who must register himself in MyHealth, using the *Register Patient* functionality, and must record his clinical data, using the *Collect Data* functionality. The second one represents a service that runs in background at patient's mobile device, for alerting the patient to collect his clinical data, using the *Alert Patient* functionality, and for sending this data to the cloud, using the *Synchronize Data* functionality.

Each use case can be refined through a Sequence diagram, for allowing the visualization in time sequence of the interactions needed to perform the corresponding functionality. Figure 58.4 shows an example of this diagram for the *Collect Data* functionality.

The following time sequence, for performing the *Collect Data* functionality, is shown in Fig. 58.4: first the Patient must insert his clinical data into an application form that is

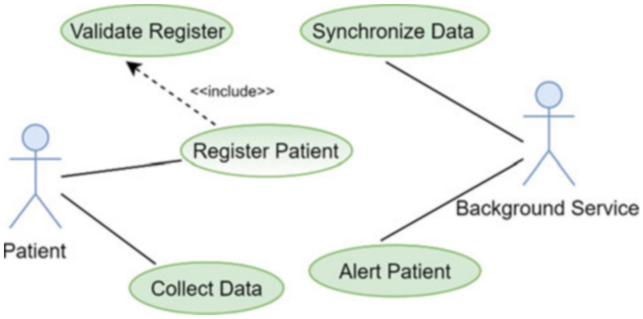


Fig. 58.3 Use Case diagram for Mobile App

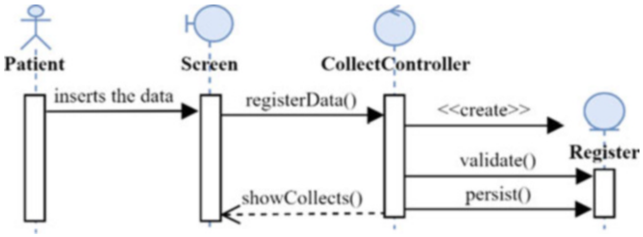


Fig. 58.4 Sequence diagram for *Collect Data* of Mobile App

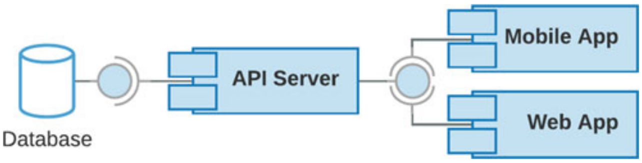


Fig. 58.5 MyHealth architecture

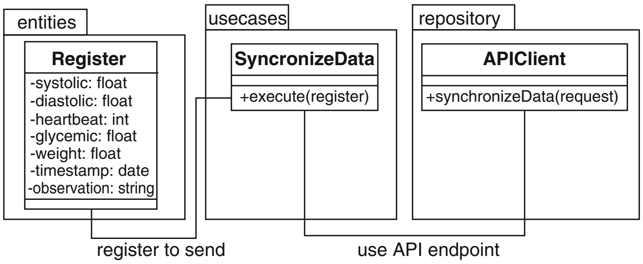


Fig. 58.6 Class diagram for *Syncronize Data* functionality

displayed in the Screen of his mobile device; then this data is sent to a CollectController, which creates a new Register instance, validates the data, and persists it into the database; and finally a printing of all patient clinical data is displayed in the Screen of his mobile device.

58.5.2 System Architecture

At this phase the system was structured, according to the client-server architecture, into three modules: *API Server*, *Mobile App*, and *Web App*. Figure 58.5 shows this architecture.

API Server is the only module that communicates directly with the database, providing a secure API to Mobile App and Web App client modules to store and retrieve all MyHealth information.

Mobile App is an application for mobile devices with Android operating system. It employs 3G/4G or Wi-Fi networks for internet connection, which is needed only for the patient registration, and for data synchronization. The patient's clinical data is stored in a local and small database of his mobile device. This application also allows for the patient obtaining an access token, who must forward it to the health professional, for enabling him to access the patient information through the Web App module.

The Mobile App components are organized according to the Clean Architecture, which aims for separating responsibilities across layers, and whose main rule is "the component dependencies can take place only from the outermost to the immediately innermost layer" [14]. Figure 58.6 shows an example of this component organization through the UML Class diagram, focusing on the *Syncronize Data* functionality.

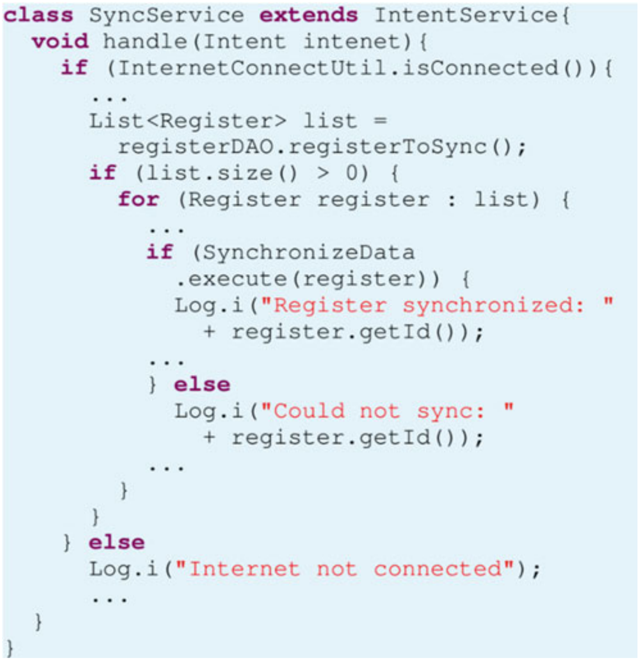

```
class SyncService extends IntentService{
  void handle(Intent intenet){
    if (InternetConnectUtil.isConnected()){
      ...
      List<Register> list =
        registerDAO.registerToSync();
      if (list.size() > 0) {
        for (Register register : list) {
          ...
          if (SynchronizeData
            .execute(register)) {
            Log.i("Register synchronized: "
              + register.getId());
          ...
          } else
            Log.i("Could not sync: "
              + register.getId());
          ...
        }
      }
    } else
      Log.i("Internet not connected");
      ...
  }
}
```

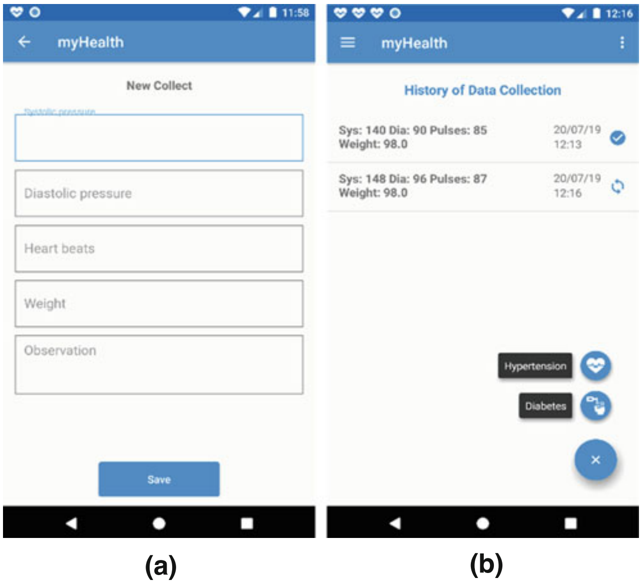
Fig. 58.7 Excerpt of the *SyncService* class

As shown in Fig. 58.6, there are three main layers: *repository* is the outermost layer, it encapsulates the APIClient class that converts the data received from SyncronizeData class to the format to be sent to the API Server; *usecases* is the intermediate layer, it encapsulates the SyncronizeData class that implements the business rules of this functionality, and depends on the Register class; and *entities* is the innermost layer, it encapsulates the Register class that has no dependencies with another class.

Web App is an application hosted in the cloud, which enables health professionals for accessing and visualizing their patient's clinical data through charts. This access is granted by each patient forwarding a token to his health professional. In order to generate this token, Mobile App and Web App modules communicate through the API Server.

58.5.3 System Implementation

At this phase, the functionalities defined in the specification phase were implemented. For example, an important requirement of Mobile App is to allow the patient to record his clinical data regardless the mobile device is connected or not to the internet. For carrying out this requirement, a service that runs in the background on the mobile device was implemented. This service checks every minute if the device is connected to the internet and if so calls the *SynchronizeData* functionality, which in turn sends the clinical data to the cloud. Figure 58.7 shows an implementation excerpt of this service, focusing on the *SyncService* class.



New Collect

Systolic pressure

Diastolic pressure

Heart beats

Weight

Observation

Save

History of Data Collection

Sys: 140 Dia: 90 Pulses: 85 20/07/19
Weight: 98.0 12:13

Sys: 148 Dia: 96 Pulses: 87 20/07/19
Weight: 98.0 12:16

Fig. 58.8 Mobile App UI: (**a**) data collecting (**b**) data history

Two other important requirements of Mobile App are to ensure compatibility with most Android-based mobile devices, and to ensure security of the data stored on these devices. For carrying out the first one, Mobile App was implemented using the Android development API version 19, becoming this application compatible with 96.2% of the Android-based mobile devices [15]. For carrying out the second one, the data is stored using the SQLite library with the extension SQLChipher, which provides 256-bit AES encryption for the database.

Figure 58.8 shows two screenshots of the Mobile App User Interface (UI): (a) the UI for allowing the patient to record his clinical data, where the data fields are displayed according to the monitored NCD; and (b) the UI for allowing the patient to visualize his clinical data already recorded, and the ones already sent to the cloud.

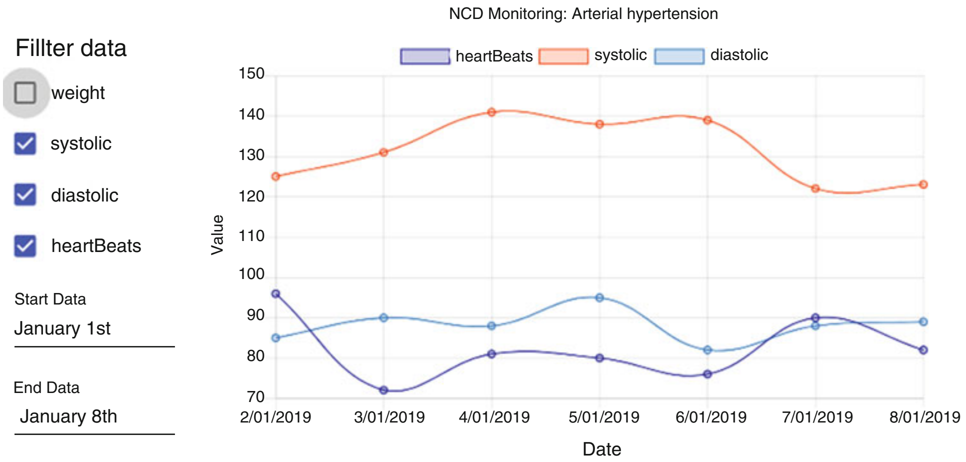
Web App was implemented in JavaScript using the ChartJS framework, which is responsible for generating the patient clinical data graph. Figure 58.9 shows the UI displayed to the health professional for the patient clinical data visualization. These data are presented through a dynamic chart, where filters, such as the type of clinical data, and time interval to be analyzed, can be specified. The health professional can also obtain details of the clinical data recorded by the patient on a specific day (e.g., time of registration, and observations), by positioning the cursor at certain points on the chart.

Finally, the API Server was implemented in JavaScript language, and runs over Node, a server-side JavaScript code interpreter. This module uses the Express framework for the implementation of its API, employs the HTTPS protocol for communicating with the other modules of MyHealth, and uses the PostgreSQL database.

58.6 Conclusion

This paper presented MyHealth, a system that employs Ubiquitous Computing technologies to support the healthcare of patients with certain NCDs. This system allows for storing, treating, and sending to the cloud the patients' clinical data, which are collected by them using low cost medical equipment, some of these provided for free by the Brazilian Unified Health System. These treated data are them made available to the health professionals, supporting in this way the decision-making process.

Our next work is to evaluate the MyHealth performance, and the usability of this system as perceived by patients and health professionals. As future work, we intended to extend MyHealth with technologies of Internet of Things (IoT) to allow the automatic collecting of patients' clinical data by wearable sensors.



Filler data

□ weight

☑ systolic

☑ diastolic

☑ heartBeats

Start Data
January 1st

End Data
January 8th

Fig. 58.9 Web App UI for visualizing the patient's clinical data

Acknowledgments This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior—Brasil (CAPES)—Finance Code 001.

References

1. Mendis, S., World Health Organization: Global Status Report on Noncommunicable Diseases 2014. World Health Organization, Geneva (2014)
2. World Health Organization: Noncommunicable Diseases Progress Monitor, 2017. World Health Organization, Geneva (2017)
3. Malta, D.C., Oliveira, T.P., Santos, M.A.S., Andrade, S.S.C.D.A., Silva, M.M.A.D.: Avanços do Plano de Ações Estratégicas para o Enfrentamento das Doenças Crônicas não Transmissíveis no Brasil 2011–2015. Epidemiologia e Serviços de Saúde. **25**, 373–390 (2016)
4. Liang, B.J., Ma, X.M., Wu, J.J., Yan, J.D.: Design and construction of mobile chronic diseases management system. In: 2nd IEEE International Conference on Computational Intelligence and Applications, pp. 518–522. ICCIA, Karachi (2017)
5. De Almeida, V.P., Endler, M., Haeusler, E.H.: A framework for customizing the mobile and remote monitoring of patients with chronic diseases. In: 16th International Conference on e-Health Networking, Applications and Services, Healthcom 2014, pp. 329–334. IEEE, Piscataway (2014)
6. Sabbir, A.S., Bodroddoza, K.M., Hye, A., Ahmed, M.F., Saha, S., Ahmed, K.I.: Prototyping arduino and android based m-health solution for diabetes mellitus patient. In: 1st International Conference on Medical Engineering, Health Informatics and Technology, pp. 1–4. IEEE, Piscataway (2017)
7. Dharmasiri, N.D., Vasanthapriyan, S.: Approach to heart diseases diagnosis and monitoring through machine learning and iOS mobile application. In: 2018 18th International Conference on Advances in ICT for Emerging Regions, ICTer, pp. 407–412. IEEE, Piscataway (2019)
8. Brazil: LEI N° 11.347, DE 27 DE SETEMBRO DE 2006. Dispõe sobre a distribuição gratuita de medicamentos e materiais necessários à sua aplicação e à monitoração da glicemia capilar aos portadores de diabetes inscritos em programas de educação para diabéticos. Diário Oficial da União, 28 setembro. Brasília (2006). [Online]. http://www.planalto.gov.br/ccivil_03/_Ato2004–2006/2006/Lei/L11347.htm. Accessed 26 July 2019
9. Ministry of Health of Brazil: Report on the 3rd Forum on the Strategic Action Plan to Tackle Noncommunicable Diseases in Brazil. Ministry Health of Brazil, Rio de Janeiro (2018)
10. Brazilian Institute of Geography and Statistics (IBGE): PNAD Contínua TIC: Internet chega a três em cada quatro domicílios do país, [Online]. https://agenciadenoticias.ibge.gov.br/agencia-sala-de-imprensa/2013-agencia-de-noticias/releases/23445-pnad-continua-tic-2017-internet-chega-a-tres-em-cada-quatro-domicilios-do-pais (2017). Accessed 26 July 2019
11. Beck, K.: Test Driven Development: By Example. Addison-Wesley Longman Publishing Co., Inc., Boston (2002)
12. Xu, S., Li, T.: Evaluation of test-driven development: an academic case study. In: Software Engineering Research, Management and Applications 2009, pp. 229–238. Springer, Berlin (2009)
13. Bajaj, K., Patel, H., Patel, J.: Evolutionary software development using test driven approach. In: International Conference and Workshop on Computing and Communication, Vancouver, BC, Canada (2015)
14. Martin, R.C.: Clean Architecture: A Craftsman's guide to Software Structure and Design. Prentice Hall Press, Upper Saddle River (2017)
15. Google: Distribution dashboard. [Online]. https://developer.android.com/about/dashboards. Accessed 28 July 2019