DIYbetes: a Mobile Platform for Empowering Type 2 Diabetes Patients

Mário Zenha-Rela
RedLight Software Inc.
IPN Business Center
3030 Coimbra, Portugal
mzrela@redlightsoft.com

Ana Paula Matos
School of Psychology
University of Coimbra
3001 Coimbra, Portugal
apmatos@fpce.uc.pt

Lèlita Santos
School of Medicine
University of Coimbra
3004 Coimbra, Portugal
lelitasantos@fm.uc.pt

Miguel Antunes
RedLight Software Inc.
IPN Business Center
3030 Coimbra, Portugal
mantunes@redlightsoft.com

João Diogo Costa
RedLight Software Inc.
IPN Business Center
3030 Coimbra, Portugal
jdcosta@redlightsoft.com

Marta Oliveira
RedLight Software Inc.
IPN Business Center
3030 Coimbra, Portugal
moliveira@redlightsoft.com

***Abstract*—Type 2 Diabetes is a growing pandemic that already affects 347 million people in the world, which means that one in every 20 people is affected by the disease. Since Diabetes is a disease with significant behavioral roots, managing it involves a shift in behavior. A multidisciplinary team of researchers from Medicine, Psychology and Information&Communication Technologies have gathered to build a mobile platform that addresses behavior change of patients by helping them avoid bad decisions and pushing them to have a healthier lifestyle. In this paper, we describe the technology behind the DIYbetes platform and how it is a paradigm of a networked solution and system intelligence that supports a demanding set of non-functional (quality) requirements such as availability, latency, throughput, modifiability, maintainability, testability and security.**

***Keywords*–mobile systems, cloud, dependability, diabetes, healthcare.**

I. Introduction

Diabetes Mellitus (DM) is a group of metabolic diseases in which a person has high blood sugar levels. Untreated, diabetes can cause many complications, namely cardiovascular disease, chronic renal failure, and retinal damage. Adequate treatment of diabetes is thus important, as well as blood pressure control and lifestyle factors such as stopping smoking and maintaining a healthy body weight. There are three main types of diabetes mellitus: Type 1 DM (T1DM) results from the body's failure to produce insulin, and currently requires the person to inject insulin or wear an insulin pump. Type 2 DM (T2DM) results from insulin resistance, a condition in which cells fail to use insulin properly, sometimes leading to absolute insulin deficiency as in T1DM. The third main form, gestational diabetes, occurs when pregnant women without a previous diagnosis of diabetes develop a high blood glucose level. It may precede the development of T2DM.

All forms of diabetes have been treatable since insulin became available in 1921. T2DM, especially in early stages, may be controlled with a lifestyle change as a number of lifestyle factors are known to be important to the development of T2DM, including obesity, lack of physical activity, poor diet, stress, and urbanization. The World Health Organization estimates that in 2012 about 347 million people had diabetes, with T2DM making up about 90% of the cases [1]. The same report estimates that in 2004 only, about 3.4 million people died as a direct consequence of DM. This disease incidence keeps increasing rapidly, with the direct number of deaths attributable to DM estimated to increase 66% between 2008 and 2030. In terms of costs, the International Diabetes Federation estimates that 418 billion dollars have been used for the management and treatment of this disease in 2010, with this figure expected to grow to about 561 billion dollars in 2030. DM occurs throughout the world, but is more common in the more developed countries. However, the greatest increase in prevalence is expected to occur in Asia and Africa, where most patients will probably be found by 2030. The increase in incidence in developing countries follows the trend of urbanization and lifestyle changes, namely into a 'westernized' diet. In fact, the World Health Organization (WHO) characterizes T2DM, which represents 90% of DM cases, as a disease almost exclusively derived from unhealthy lifestyle practices [2], [3].

Having described the problem and justified its relevance, in the next sections we describe an effort to address it using technology in a sensible way. DIYbetes (http://DIYbetes.org) is a government-funded initiative to promote Type 2 Diabetes patients self-help using mobile platforms (smartphones and tablets). It was designed with inputs from diabetes medical experts and psychologists that have been dealing with the disease for more than three decades, and a team of Information&Communication professionals. In Section II the objectives of the project are highlighted stressing how important is to build a solution that empowers patients, rather than technical wizzardry. Then, in Section III the core non-functional (quality) attributes of the platform are presented, as an effective solution needs to deal with much more than functionality (e.g., privacy, usability). Armed with this background we arrive at the technical core of the paper (Section IV) where the architecture of the solution is presented, and some trade-offs are discussed. Section V closes the paper with the project current status and further evolutions underway.

II. Platform objectives

The DIYbetes project main objective is to create a user-centered technical solution that will allow T2DM patients to gain better awareness of their daily habits and track their disease's evolution. This is achieved by promoting self-awareness and empowerment, while reducing the need for unsustainable in-person appointments due to distance, increase of prevalence and unsustainable increase in costs by healthcare organizations and authorities. DIYbetes is an application running mainly on smartphones and tablets (Figure 1) and also accessible via a web portal. It is designed to be integrated into the service-chain of healthcare professionals to allow recording of glucose

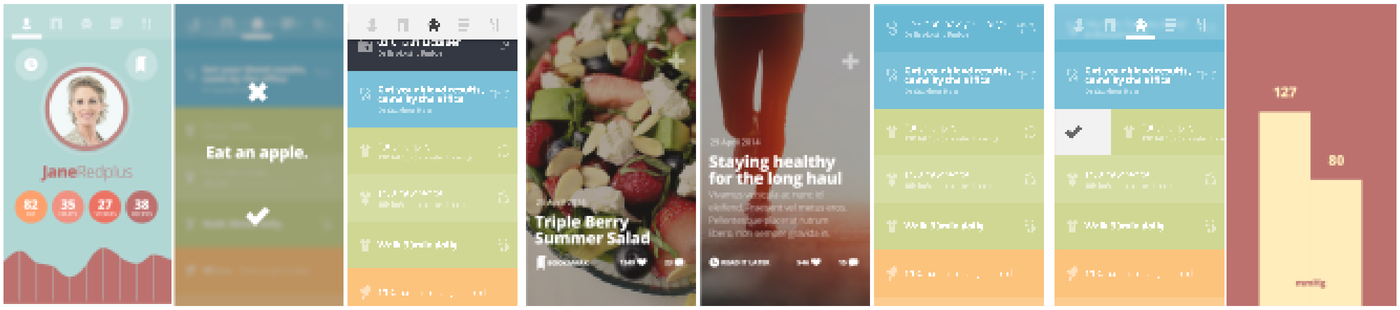

Figure 1. Some screens extracted from the DIYbetes mobile app.

levels and other disease related information, as well as support direct communication with the healthcare professionals [4]. The interactions between patients and their doctors will allow the former to feel more supported. A distinctive feature of this application is the perceived increase in quality of service by the patients due to the follow-up, without costly travel and appointment costs that would be required for such a personalized interaction. To summarize, the DIYbetes application is intended as a bridge between patients and their accompanying health professionals, as it has been designed not only as a logger, but as an information vehicle where diabetics can monitor their information and physicians can have a closer, and continuously up-to-date, look of their patient's records [5], and communicate directly, particularly when distances would prevent this kind of monitoring from happening without such solution (e.g., large area countries without an extensive transportation infrastructure). In synthesis, the technical solution:

- Improves diabetes control by the patient.
- Provides a communication channel between patients and health professionals.
- Aims to reduce the diabetes induced complications and mortality.
- Delivers personalized disease related information.

III. Approach

We shall now present the state-of-art for related approaches, followed by the core requirements that serve as architectural drivers for the solution built.

A. Current approaches

There are several thousand mobile applications to manage DM, both at GooglePlay! and the AppStore [6]-[9]. Most are one person initiatives by technically-savvy patients, others are solid commercial offerings, and a large number is delivered by healthcare institutions to help their patients deal with the disease. However, most of these applications target T1DM patients, due to the criticalness of the condition. This is unfortunate because T2DM represents about 90% of DM patients worldwide. Another reason for the lack of T2DM solutions derives from the fact that this condition is prevalent with age, normally above 45 years, while T1DM occurs at an early age, starting at 9 years or earlier. The lack of technological literacy of elder patients has been preventing the widespread deployment of such solutions. However, several technology observatories have reported a significant growth of tablets among the older population [10], which might be due to the larger screen size and easiness of reading for sight constrained individuals. This growth, while more evident in the more developed coutries (those where the prevalence of T2DM is higher) opens the opportunity for a multiplatform solution as DIYbetes. Currently, most DM solutions focus on logging glucose values or diabetes safe recipes. A remarkable outlier is the Glooko mobile app [11] whose focus is to support people with diabetes interacting with their blood glucose readings and add lifestyle context including carbs, insulin & medication, and activity data. While its main target are T1DM patients, it has been successfully used by T2DM patients too. Despite its very interesting motivational approaches, the fact is that T2DM patients are more receptive to motivation tips targeting their age bracket. That is a major focus of this project and the reason why we have brought into the project psychologists with more than three decades experience with T2DM patients.

As referred above, addressing T2DM requires a behavior change, namely a healthier diet and regular exercise. This requires a service managed by physicians that keep patients motivated. It is manifest that only technology can address the ever growning number of patients [12] by streamlining the treatments, and motivating patients to follow healthier behavior/habits. Hence, more than a logger of glycose readings, this project aims to be a personal lifestyle management tool intended to coach people with diabetes.

B. Requirements as architectural drivers

There are a set of core requirements that have a significant impact in the design decisions associated with the DIYbetes platform. These requirements are the architectural drivers organized as 1) business constrains, 2) technological constrains, 3) functional requirements, and 4) quality attributes.

The *business and technical constrains* are the business and technical decisions made upfront the design. The *functional requirements* specify what the system must execute, but without the details of what does not have an architecture impact (for example colors of the displays and kinds of widgets). The *quality attributes* describe the system's behavior i.e., the properties that will describe how a system achieves a certain functionality. The project's functional requirements include items such as 'support for multiple devices and interfaces' so that users can record and monitor their activities anywhere. This feature allows physicians to follow more closely their patients activities, specify objectives, and act upfront when the results are not the ones expected. Their role is to specify achievements and objectives in order to keep users committed, engaged, and help them to cure their disease.

The system also works as a coach for the physician suggesting challenges based on the users' objectives and health values. Logging meals, exercise, and health values it is a dull activity, so the system identifies the patterns in users' behaviors in order to streamline those tasks. Finally, the platform tries to influence and motivate patients to feel proud of their achievements.

As highlighted by functional requirements, the platform handles sensitive health and personal data. Due to this fact, security is a key quality attribute and an important technological constrain in the system. Both system's information and infrastructures must respect the legal personal data protection regulations, namely the european General Data Protection Regulation (GDPR) [13].

Additionally to the GDPR compliance, this project aims to gather new sound scientific evidence with regard to the patient's compliance. That is why usability is a top concern, since the daily logging needs to be simple and consistent in the different interfaces so that users keep logging their data.

As the business constraints also highlight, the flexibility to adapt and integrate different business models (commercialize new services, integrate new treatment plans, and integrate multiple devices) allows the platform to adapt to market trends, and explore new markets while keeping support costs low. In fact, the project is aligned with the most recent US healthcare legislation (March 2015), namely the Centers for Medicare and Medicaid Services (CMS) Stage 3 proposed rule [14] that requires that more than 35 percent of all patients seen by the provider or discharged from an hospital should received a secure message using the EHR's electronic messaging function or in response to a secure message sent by the patient. The proposal also calls for more that 15 percent of patients to contribute patient-generated health data or data from a non-clinical setting.

As can be seen from this overview, the complexity of this platform is very significant. In the next Section, we present and describe the architectural solution that supports such demanding requirements.

IV. TECHNICAL SOLUTION

In this Section, we present the technical solution that supports the project's goals and constraints. We shall also dicuss some of the design decisions that were taken, as well as some trade-offs that an actual system always imposes.

A. Overview

In Figure 2, we present the context view of the platform. It describes, from a dynamic perspective, the system boundaries and how external actors (people or systems) interact with the system.

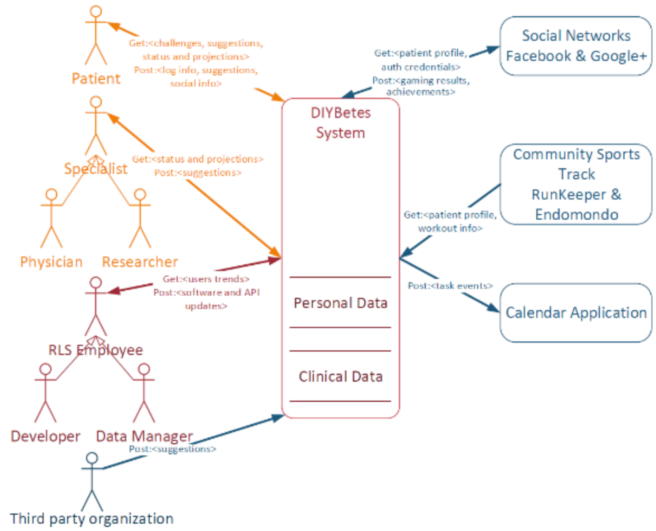

Figure 2. Context view

Patient to DIYbetes System

This data flow represents log info (health values, meal logs, workout sessions, and drugs intakes), suggestions (examples of healthy restaurants/menus in a given location, workout places, and recipes), and social info (events, new friends, and comments). This information is both stored and managed in order to identify new suggestions/tasks for other users that share the same profile. The log info is stored in the Clinical Data repository while the users' profile and social data is stored in a Personal Data repository.

Patients have tasks like performing a workout, log a meal, take drugs, and/or log their clinical values (instant sugar, pulse, blood pressure, blood sugar level readings, HbA1c, and weight). If the patient accepts the task and provides the requested values, these are processed by the system. Thus, the system can identify patients' trends, calculate projections towards health objectives, and identify the sub-set of tasks that are suitable considering this patients' progress. All this information is then stored in the Clinical Data Repository.

DIYbetes System to Patient

This data flow represents challenges (new tasks that the user has to perform in a daily basis or sporadically, where these tasks are defined in accordance with the patients' objectives), status and projections (based on the patients' trends the system identifies how distant patients are from their objectives), and suggestions (examples include places to workout, restaurants, recipes, menus, exercises, meal plans, and so on). This information is read and updated from Personal Data and Clinical Data repositories.

Sporadically, or in specific time intervals, each patient receives a list of individual and group challenges known as 'tasks'. These tasks coach patients to adopt beneficial behaviors, and target patients' health and lifestyle objectives. The system identifies new tasks based on the patients' progress in previously assigned tasks. Additionally, as users' submit their daily logs, the system calculates current progress and related projections. Thus, patients query this information sporadically to monitor their progress, combine different indicators, and verify the areas where they need to improve. Based on the patients' progress, location, and habits, the system provides new suggestions (meals, plans, workout places, and restaurants) designated to help users to improvie their behaviors.

 ISBN: 978-1-61208-422-0

Specialist to DIYbetes System

This data flow represents the inputs provided by a specialist (physician or researcher). The specialists view patients' progress in the system, submit updates on their patients' plan, accept or reject suggestions provided by the system, and provide some logic to identify new treatment patterns. The system stores these updates, and uses this information to validate and refine their algorithms. Context Information Specialists are studying patients trends and reactions to treatments. This study is conducted based on the logs provided by patients and stored in the Clinical Data repository. Whenever specialists identify new tasks, they submit these improvements in the system. The system then uses this information to update the patients' plan. Additionally, specialists provide objectives and thresholds for their patients, which are used to update patients' plans. Specialists can also request from the technical support team new algorithms, statistical functions, and new ratios. Although these improvements are submitted by the technical team (RedLight Software (RLS) employees) the knowledge comes from the specialists.

RLS employee

The RLS technical staff can develop new Application Programming Interfaces (APIs) to extend the DIYbetes features, or modify the services in order to increase the performance in the system. These APIs are plugged into the system (the required changes are made in the system's interfaces), or when a new infrastructure is deployed, and both updates are ready to be used by the system. This can occur e.g., when context information A specialist requests RLS to include a new statistical function or algorithm, or when the DIYbetes steering commitee identifies a new opportunity that translates into a new feature. In both these situations a developer reviews the system's design, identifies the required software, develops a new API, and deploys the software/hardware in the system, without interrupting the system's execution.

Third Party Organizations

Context Information Third Party organizations like restaurants, health clubs, hospitals, and laboratories, wish to promote/suggest their services. These organizations can use the DIYbetes system interface to promote information related with their products and services. Hence, the DYIbetes system can use third party products as suggestions/recommendations, and forward them to the patients when their profile is somehow related with the suggestion. Therefore they can request RLS to include/extend interfaces that can be used by their services and information sources. When this connection is established, the third party organization is able to send, for example, healthy menus, restaurants, or gyms near the patients' location.

Social Networks

A patient or specialist can choose to register/login using their social network credentials. The DIYbetes system forwards to users (patients, researchers, or physicians) the social network authentication page, which they use to login in their social networks. After authentication succeeds in social networks, they provide user identification to the DIYbetes service, which then forwards a session to the user. The system then requests the user's profile, which is forwarded by the associated social network. The DIYbetes system besides authenticating users, also requests profile information (such as e-mail, age gender, and location). The system uses this profile data, thus refining patient's tasks, identifying related groups (groups based on the patients' profile like location, age, gender, and so on), and managing patients' statistics.

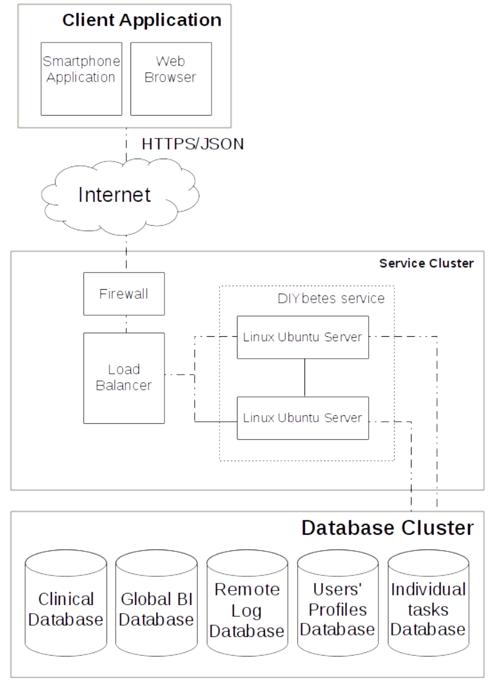

Figure 3. Deployment view

B. Deployment View

We shall now discuss where each part of the architecture will live at deployment. The Deployment View (Figure 3) describes the environment of each system's application and service highlighted in the context view.

Client Application Layer

The *Client Application Layer* is responsible for the support of the DIYbetes client applications, both web and mobile, used by patients and specialists (researchers and physicians).

Smartphone Application

The *Smartphone Application* is responsible for the interface between the patient(s) and the system. It allows patients to log their tasks (workout sessions, daily meals, drugs intakes, and health values), specify and update objectives, and manage their individual and group tasks. The Smartphone Application has location features related with the management of new suggestions (patients receive restaurants, meals, and workout places related with their location), and context aware tasks. It includes connection interfaces designated to support external sensors, which have the objective of complementing daily logs, as well as an internal database, which allows using some application features without connectivity. However, the dynamic information, like social updates, gets deprecated without connectivity to the DIYbetes Service. This component has several technical restrictions namely a network connection (wireless and mobile networks, 3G and 4G). It uses HTTPS and JSON protocols.

Web browser

The *Web browser* is responsible for the interface between the user (patient, physician, and researcher) and the DIYbetes Service. The Web browser allows patients to monitor their progress (thus, users can view graphs with their progress towards their objectives), verify their task results, manage social and gaming updates, manage objectives and alarms, and configure the connection with their physicians. The Web browser allows physicians to monitor their patients' progress, manage statistics, update patients' plans and objectives, and specify alarms. Each Web browser includes cookies that are responsible for holding the session with the DIYbetes service.

Service Cluster Layer

The *Service Cluster Layer* is responsible for supporting the DIYbetes servers and support infrastructures. This Service Cluster is composed by multiple servers distributed across different geographic locations. Although each global location (Europe, US, and Canada) has their own servers, this layer comprises all the infrastructures for each of these service instances.

Load Balancer

The *Load Balancer* is the service responsible for the distribution of the workload for different DIYbetes servers. The workload distribution is configured based on the geographic location of the services (each area has a DIYbetes service instance), since each region has its own restrictions. The load balancer performs automatic fail over recovery, so when a server is not available, this entity forwards the requests to a different server with lower workload. The deploy of a load balancer should follow three phases, 1) initially it is deployed in one of the service machines. With an increase of the workload 2) a load balancer is deployed in a dedicated machine. Finally, 3) a cluster of load balancers is available if a since load balancer instance is not able to support all the accesses.

DIYbetes service

The *DIYbetes service* is a cluster of service instances running in a group of server, virtual or physical. Each geographic region has their on DIYbetes Service instance. Each DIYbetes Service instance is independent from each other, they only share the Global BI Database.

Linux Ubuntu Server

The *Linux Ubuntu Server* is the server responsible for holding the service infrastructure related with the REST DIYbetes Service. It also manages the HTTPS requests and forwards them to the REST DIYbetes Service and supports the interaction with the Database Cluster through postgres-adapter. This is a critical component of the platform, so the architecture was designed in order to be able to support stringent latency and throughput quality attributes: above 10.000 responses in 1 second; above 10.000 simultaneous requests with a maximum latency of 5 seconds and average latency below 1 second.

Database Cluster

The *Database Cluster* is a cluster of database servers (virtual or physical). Each geographic location has their group of database servers (Individual Tasks/Objectives Database, Group Tasks/Objectives Database, Alarms/Recipes/Drugs Database, and User Profile Database) and clinical database servers (Clinical Database). The Global BI and Remote Log Databases are shared between different DIYbetes services from each region.

Quality Attributes

Due to the criticality of the platform a large set of quality attributes were considered since the projects very beginning, namely latency, throughput, availability, modifiability, maintainability, testability and security. Each quality attribute was thoughly described and relevant test cases performed.

C. Design decisions

A number of design decisions were taken that involved a number of trade-offs. This decision process is detailed below:

1) A relational database was the choice for data repositories that involve queries with multiple joins. Thus, most of the operations require support for these type of relations between data. NoSQL Databases can be used to cache logs when they arrive at the system and before they are processed for statistics purposes. *Trade-off:* Dynamic languages allow to include new features and modules at runtime, and the access to database require less effort to program. However, this simplicity hides poor performance in functions where the complexity is non-linear (e.g., sequence of whiles).
2) The web interface is implemented in a dynamic language (Ruby), and the threads and message queues are using JRuby (Java is used in the thread engine and message queues structures). Thus, threads and message queues have a good access scheduler and performance in Java.
3) Ruby simplifies the integration with non-dynamic languages, so functions where the complexity is non-linear will be later (re)implemented in Java and dynamically plugged into the code. This approach partially handles the performance problem.
4) REST was adopted as the service technology. This approach allows to share logic between web and mobile access interfaces, and it is supported by dynamic languages, which allow to change components at runtime. *Trade-off:* the use of REST requires less effort to implement, but it only should be used when there is a hierarchy of accesses. This hierarchy is appropriate considering the project's information structures, since specific users have access to specific features. If such structure did not exist, web services would be a better option. Moreover, since REST will be used, the service will not have a WSDL that clarifies the structure of the service and the methods that it provides. Thus, we had to devote additional

 ISBN: 978-1-61208-422-0

effort to document the API that is used to integrate services with external parties.

5) The system currently does not support clinical protocols (HL7 or DCOM), since not all healthcare systems implement them, or can be integrated with external systems. Thus, supporting this diversity was postponed due to the size of the technical team.
6) The service will have two keys, which services shall use for read or write information from the Clinical Database. These keys are required to authenticate services in these databases, but no service will have the 2 keys simultaneously, in order to limit database read and write operations. *Trade-off:* The approach described promotes security, but does not protect against attacks that result from the organization staff. Thus, an external database was added that has all the database and OS logs. Therefore, all authentications in the hosting machine, access to the database, all database reads and/or writes are logged.
7) It was decided to deploy two parallel servers, so that we are able to install updates without making the service unavailable. Additionally, no effort will be required to implement complex availability mechanisms. If a server has a fault, the other server is able to support users' requests.
8) The project considers an internal Load Balancer, since it allows the organization specifying custom rules to balance the workload (examples of rules include geography location, identification of the users, hours of the day, and so on). Moreover, a custom approach simplifies the deployment of different A/B testing scenarios. Finally, this approach is essential to promote service uploads without the need to have the service totally unavailable. *Trade-off:* a custom load balancer implies more processing at the server side and additional effort to configure and deploy the rules that will manage Load Balancer operations. Considering the importance of the system's flexibility and performance, this drawback does not justify a different approach. Moreover, a local Load Balancer shall be used, since it allows to reuse the same container technology adopted in the service infrastructure (Nginx). However, this local load balancer has a negative impact in the system's performance. Finally, while Linux OS has more security mechanisms than Windows OS, it has less that OpenBSD. However, we did not consider that OS security mechanisms are a stringent requirement. To manage this issue there is a remote, independent OS BD, which will log all database accesses and server authentications.
9) Each server runs a complete instance of the service, which simplifies the scalability and configurability issues, since if more processing power is needed more servers with more service instances can be added and only minor configurations are required in the Load Balancer.
10) Linux OS was the OS selected for test and product platforms, since most of the organization developers are used to its environment and it adequately fits the purpose.

V. Conclusion

In this paper, we presented a multiplatform solution (mobile and web), that addresses behavior change of Type 2 Diabetes patients, and described the set of technologies underlying the DIYbetes platform. It is a complex piece of technology that adresses a set of demanding non-functional (quality) attributes in order to operate in harsh conditions with a low technological literate population. The current solution has just been deployed in Portugal for the patients of the national healthcare system (literally all interested citizens) and will be further expanded to other geographies. A partnership with a USA-based company will allow testing the platform for a different language and geography.

Acknowlegment

This project was was supported by a grant from the national COMPETE SI&IDT program (project nr. 34006/2013, DIYbetes). The authors would also like to thank the contribution from all the patients, doctors and nurses involved.

References

[1] Diabetes Action Now, World Health Organization and the International Diabetes Federation, 20pp. 2004 ISBN: 924159151X

[2] International Diabetes Federation: Epidemiology and morbidity. [http://www.diabetesatlas.org/content/diabetes-and-impaired -glucose-tolerance]

[3] Berger B, Stenstrom G, Sundkvist G. Incidence, prevalence, and mortality of diabetes in a large population. A report from the Skaraborg Diabetes Registry. Diabetes Care. 1999;22:773-8.

[4] Kerr EA, McGlynn EA, Adams J, Keesey J, and Asch SM, "Profiling the Quality of Care in Twelve Communities: Results from the CQI Study," Health Affairs, Vol. 23, No. 3, May/June 2004, pp. 247-256.

[5] Mosa A., Yoo I.,Sheets L, A Systematic Review of Healthcare Applications for Smartphones, BMC Medical Informatics and Decision Making 2012.

[6] Mobilehealthnews (2010,Oct), http://mobilehealthnews.com/9168/why-the-att-welldoc-deal-matters/ [21] Jasinski,D.J. (2012, Apr) IBM Healthcare, interview.

[7] Wu RC, Morra D, Quan S, Lai S, Zanjani S, Abrams H, Rossos PG: The use of smartphones for clinical communication on internal medicine wards. J Hosp Med 2010, 5:553-559.

[8] Soto RG, Chu LF, Goldman JM, Rampil IJ, Ruskin KJ: Communication in critical care environments: mobile telephones improve patient care. Anesth Analg 2006, 102:535-541.

[9] Hasvold PE, Scholl J: Disrupted rhythms and mobile ICT in a surgical department. International Journal of Medical Informatics 2011, 80:e72-e84.

[10] Adults' Media Use and Attitudes Report 2014, Independent regulator and competition authority for the UK communications industries, Ofcom Research Document, April 2014.

[11] Glooko is a mobile application from a Silicon Valley company that delivers a mobile and cloud based, diabetes & chronic disease management system. https://glooko.com/ [retrived March 2015]

[12] Diabetes: Fatos e Números 2012, Relatório Anual do Observatório Nacional de Diabetes (4a edição), 19 de Fevereiro de 2013.

[13] European Parliament legislative resolution of 12 March 2014 on the proposal for a regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation) (COM(2012)0011 – C7-0025/2012 – 2012/0011(COD))

[14] Medicare and Medicaid Programs; Electronic Health Record Incentive Program-Stage 3, online at https://s3.amazonaws.com/public-inspection.federalregister.gov/2015-06685.pdf (March 2015)