MEDINFO 2015: eHealth-enabled Health
I.N. Sarkar et al. (Eds.)
© 2015 IMIA and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.
doi:10.3233/978-1-61499-564-7-118

A Mobile and Intelligent Patient Diary for Chronic Disease Self-Management

William Van Woensel[a], Patrice C. Roy[a], Samina R. Abidi[a,b], Syed SR Abidi[a]

[a] NICHE Research Group, Faculty of Computer Science, Dalhousie University, Halifax, Canada
[b] Medical Informatics, Faculty of Medicine, Dalhousie University, Halifax, Canada

Abstract

By involving patients in their own long-term care, patient self-management approaches aim to increase self-sufficiency and reduce healthcare costs. For example, electronic patient diaries enable patients to collect health data autonomously, increasing self-reliance and reducing strain on health professionals. By deploying patient diaries on mobile platforms, health data collection can occur at any time and place, increasing the mobility of chronic patients who typically need to enter health data frequently. Importantly, an opportunity also arises for mobile clinical decision support, where health feedback is directly issued to patients without relying on connectivity or remote servers. Regardless of the specific self-management strategy, patient and healthcare provider adoption are crucial. Tailoring the system towards the particular patient and toward institution-specific clinical pathways is essential to increasing acceptance. In this paper we discuss a mobile patient diary realizing both the opportunities and challenges of mobile deployment.

Keywords:

Self-Management; Mobile Health; Clinical Decision Support Systems; Health Alerts and Notifications.

Introduction

In ageing societies chronic illnesses are becoming increasingly prevalent, negatively affecting people's quality of life and increasing strain on public healthcare systems [1, 2]. To rein in healthcare costs and increase self-sufficiency, patients with chronic illness are increasingly encouraged to self-manage their illness in a home-based setting [3-5]. Related approaches for patient engagement and education involve providing educational and motivational resources [4] and increasing patient self-efficacy by applying behavioural theories [5]. Electronic patient diaries empower patients to autonomously collect health data about their vitals, medications, hospital visits, and symptoms, thus avoiding manual collection by health workers and associated clinic visits. In particular, digital patient diaries accessible through mobile devices (i.e., smart phones and tablets) have the potential to capture patient health data in any setting and at any time. As such, mobile patient diaries provide an up-to-the-minute health profile of the patient and can supply relevant health alerts based on the current profile. Commercial health measurement devices are already accompanied by their own mobile health apps, such as iBGStar and OneTouch Verio Sync blood glucose monitors and iHealth and Withings blood pressure monitors. Moreover, major companies such as Google (Google Fit) and Apple (HealthKit) are developing mobile toolkits for collecting, analyzing, and monitoring personal health data.

As mobile platforms become more sophisticated, new opportunities arise to develop mobile patient diaries that go beyond typical patient data collection and reporting functions. We argue that mobile patient diaries can be extended to incorporate advanced functionalities, offering (i) mobile clinical decision support, (ii) personalized patient interactions, and (iii) data synchronization with centralized electronic medical records. Proactive decision support based on the current patient profile generates relevant interventions to adverse health conditions. By implementing a local Clinical Decision Support System (CDSS), mobile patient diaries are empowered to directly generate timely health alerts, even in situations where wireless connectivity is lacking or the server is unavailable. In previous work [6, 7] we studied how the scale of mobile decision support can be restricted by mobile hardware limitations (i.e., processing, memory). Based on this work we argue that distributed setups are more suitable for mobile systems, in which lightweight, time-sensitive reasoning is deployed locally and heavyweight processes are delegated to the server [2]. For this paper, we studied a lightweight mobile CDSS, configurable with decision rules derived from general or institution-specific clinical guidelines. In doing so, and combined with security and privacy support, we enable healthcare providers to incorporate patient-reported data and mobile CDSS recommendations within their practice.

Connectivity is an important issue in any mobile scenario because lack of Wi-Fi or mobile network coverage may result in temporary or long-term disconnections. One solution is to allow important functionality to be performed offline so disconnections do not limit mobile patient diary usage. As mentioned before, we argue that a mobile CDSS should function independently of connectivity to allow timely health alerts at any time and place. In addition, data entry may be performed offline as well, synchronizing the data with the Electronic Medical Record (EMR) server whenever connectivity is restored. To further ensure the health profile is current and to enable timely health follow-ups, patients should be encouraged to record their health data regularly and respond to alerts raised by the built-in CDSS. We achieve this by supplying personalized and recurring patient interactions, including data entry reminders and health alert notifications. Personalization options include customizing data entry schedules and alert notifications to suit patients' daily lives.

In this paper we present a mobile and intelligent patient diary for managing patients with Atrial Fibrillation (AF). As a core contribution we present a mobile CDSS configured with computerized Clinical Practice Guidelines (CPG) for AF. To improve patient engagement, the patient diary includes a notification engine, customizable with notification schedules that are personalized to suit the patients' lives. We address the connectivity issue by locally deploying a lightweight CDSS, and supporting offline health data collection. Whenever connectivity is restored, data that were collected offline are automatically synchronized with the back-end server. We implemented the patient diary as a cross-platform mobile application, with current deployments on Android and iOS.

The AF-Mobile Patient Diary (AF-MPD) was developed for the self-management of Atrial Fibrillation (AF) in the context of the Integrated Management Program Advancing Community Treatment of Atrial Fibrillation (IMPACT-AF) project[1]. AF is the most commonly sustained irregular heart rhythm, affecting approximately 350,000 Canadians. Patients carry up to five times the normal risk of developing stroke [8], and stroke associated with AF is almost twice as likely to be fatal [9]. AF-MPD is being used by over 2,000 patients within the Canadian province of Nova Scotia as part of the IMPACT-AF project. Although the patient diary focuses on collecting & acting on AF-related health data in particular, it mainly comprises generic components that are re-usable for the self-management of other chronic diseases.

AF-MPD Architecture Overview

Figure 1 shows an overview of the AF-MPD. Below we summarize the functionality of each component.

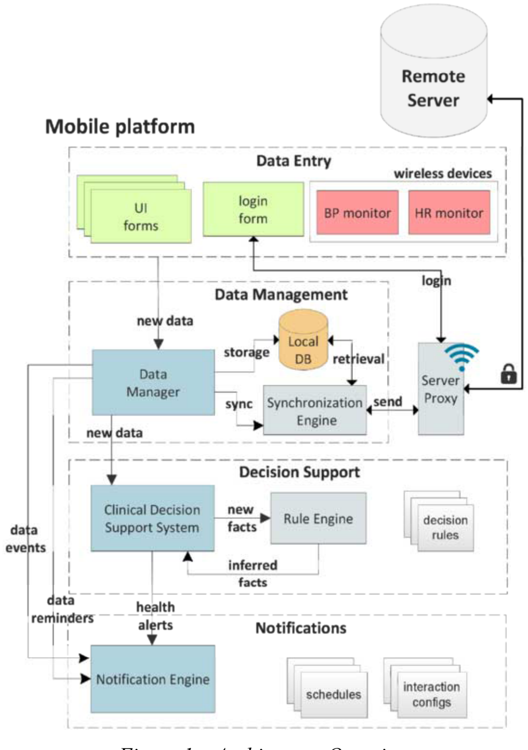


Figure 1 – Architecture Overview

To ensure security and privacy of personal health data, a patient needs to login (**login form**), which requires contacting the server *(login)* for validating the login credentials. Upon authentication, the patient can interact with his AF-MPD by providing health data and reviewing the recommendations. Health data is entered either manually via **UI forms** or automatically via **wireless devices** (e.g., using Bluetooth). This data is passed *(new data)* to the **Data Manager** for persistence in the **Local Database** *(storage)*, and synced with the server *(sync; send)* by the **Synchronization Engine**. If no connectivity is available, the engine will send (*send*) the persistent data *(retrieval)* when connectivity is restored. All server communication occurs via the **Server Proxy**, which maintains a secure server connection. Afterwards, the **Data Manager** sends the new health data *(new data)* to the **CDSS** for inferring new health findings. The **CDSS** incorporates a mobile **Rule Engine** to execute the CPG-based decision rules on the incoming patient data *(new facts)*, and returns the inferred facts *(inferred facts)* as alerts *(health alerts).*

The **Notification Engine** issues notifications to the patient, including health alerts and reminders. The patient diary components invoke the **Notification Engine** for issuing health alerts *(health alerts)* or registering reminders *(data reminders)*. Such reminders or alerts can be repeated to notify the user frequently until a required action occurs (e.g., acknowledging the alert or performing data entry). In other words, we have implemented persistent notifications that have lifecycles influenced by particular events (e.g., data entry). For this purpose, the **Notification Engine** receives relevant events from other components *(data events)*. As discussed later on, these lifecycles are defined by *schedules* and *interaction configurations*. Below, we elaborate on each component.

Data Entry

Health data may be entered into AF-MPD either manually via UI forms or automatically via wireless measurement devices. A variety of such devices are already available, often accompanied by their own monitoring apps. Open and standards-based Bluetooth communication is available via the Health Device Profile [10], and Bluetooth Low Energy (LE) is typically utilized to conserve battery. However, in practice most devices only allow communication with their accompanying applicationss by applying validation protocols, and manufacturers are not keen on sharing access details (similar issues are mentioned in [3]). Short from hacking their systems, this makes it very difficult to integrate off-the-shelf wireless devices into custom mobile apps. For this reason, the mobile patient diary currently only supports manual data entry.

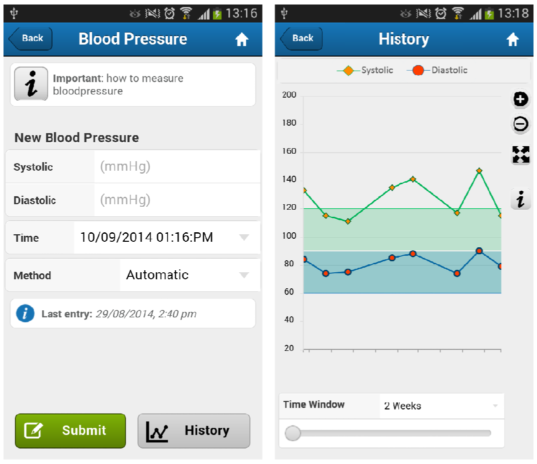
The AF-MPD supplies UI forms for collecting AF-related health data including heart rate, blood pressure (BP), and relevant symptoms or complications. To enter their current medications, patients can search a local copy of the Health Canada drug and health products database[2]. Additionally, patients can track their costs in dealing with their illness (e.g., clinic visits, loss of work). Graphical overviews of health measurements are also made available, allowing patients to track their progress over time. Figure 2 shows the blood pressure form and the graphical overview of the patient's blood pressure history over time. Figure 3 shows screens for recording AF symptoms.

Data Management

Health data is stored persistently in the local database on the mobile device and synchronized with a remote EMR database server when connectivity permits. Data synchronization may occur in two directions as well. At startup time the Synchronization Engine checks for new patient-specific information in the patient's server EMR (e.g., new medications added by the healthcare provider) and adds them to the local database. To increase patient adherence, the Data Management component registers a persistent reminder to enter health data as well as a synchronization reminder to encourage patients to find connectivity in case data synchronization is overdue.

[1] http://impact-af.ca/

[2] http://webprod5.hc-sc.gc.ca/dpd-bdpp/index-eng.jsp



Important: how to measure bloodpressure

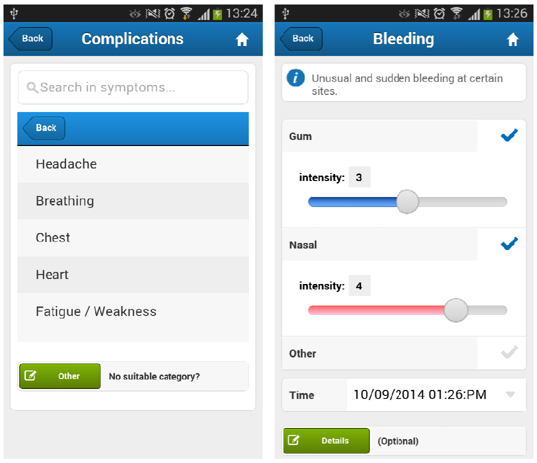
New Blood Pressure

Systolic	(mmHg)
Diastolic	(mmHg)
Time	10/09/2014 01:16:PM
Method	Automatic

Last entry: 29/08/2014, 2:40 pm

Time Window 2 Weeks

Figure 2 – Blood pressure form and overview



Unusual and sudden bleeding at certain sites.

Gum	✓
intensity:	3
Nasal	✓
intensity:	4
Other	
Time	10/09/2014 01:26:PM

Figure 3 – Symptom browsing/searching and entry form

Server Proxy

Security and privacy are critical concerns when sending health data across the Internet. All server communication occurs via the server proxy, which maintains a secure HTTPS connection with the remote server. On the server side, the valid provenance of health data is ensured via a token system. The server proxy sends the patient's login credentials to the remote server, which returns a unique, temporary token to be included in further communication. Matching the included token to the patient indicated in the health data allows the server to rule out invalid data sources.

Decision Support

By incorporating mobile CDSS, the patient diary can directly issue health alerts based on local health data without requiring connectivity or relying on the remote server. On the other hand, mobile hardware limitations may restrict the scope of mobile decision support. To support large-scale and complex decision logic, the reasoning load was distributed [2], such that we deployed lightweight and urgent processes (e.g., problematic measurements) locally, and delegated more heavyweight processes to the IMPACT-AF server.

In previous work, we developed a framework for benchmarking mobile rule engines [6] and performed benchmarks in the clinical field of AF using off-the-shelf rule engines [7]. Importantly, we observed that in the case of limited reasoning scale and complexity, acceptable performance can already be reached. For the AF patient diary we deploy 11 local decision rules that involve checking for certain symptoms, whether heart rate and blood pressure are within acceptable bounds, and whether dangerous health trends exist (e.g., elevated heart rates for an extended period of time). Since the rules only reference the latest or aggregated health measurements, the reasoning dataset is relatively limited. Patient personalization is supported by parameterizing the rules with patient-specific values (e.g., heart rate limits given the patient's age) that are also included in the dataset. The decision support rules were extracted from CPG for the treatment of Atrial Fibrillation given by the Canadian Cardiovascular Society [11] and the European Society of Cardiology [8]. A key feature of our design is the clear separation between the decision support logic and the application logic; allowing both the rules and application to be updated independently. As such, mobile CDSS can be tailored to suit different, institution-specific clinical pathways and other comorbid chronic diseases.

To illustrate the mobile CDSS, consider a patient who enters high BP measurements over a period of three days with an average systolic BP greater than 135 mmHg and/or diastolic BP greater than 85mmHg (prehypertension). After a measurement is added to the database the rule engine applies its configured rules to infer new health findings. After adding the latest measurement, the relevant rule infers an *uncontrolled blood pressure* finding, indicating prehypertension. In response, the CDSS issues a health alert via the Notification Engine. It can be noted that since reasoning is only applied after adding new health data, the local CDSS has only a minimal impact on battery life.

Notifications

Although health data entry is facilitated by mobile patient diaries, patients still need to frequently perform data entry. In addition, follow-up interventions by physicians are only possible when the collected data is synchronized frequently with the remote server. As a result, a second notification category includes reminders that are issued for data entry and server synchronization. Below we discuss the requirements for these kinds of notifications.

Mobile patient monitoring solutions allow issuing notifications at any time and place. However, this also means patients can be notified in situations where they are not able to perform the required action, such as collecting health data. To account for this constraint in our solution, it is possible to repeat notifications until the desired action can be performed. Since the urgency of the notification will typically increase as time passes (e.g., reflecting an increased necessity for health data entry), we apply a time-delayed strategy that increases notification obtrusiveness over time. This has been achieved through an *event-based* and *evolving* notification lifecycle, where (1) the employed interaction resources (e.g., icons, audio and haptic feedback) increases in obtrusiveness over time, together with notification frequency; and (2) the occurrence of events, such as performing the desired action (e.g., data entry), influences the current state of the lifecycle.

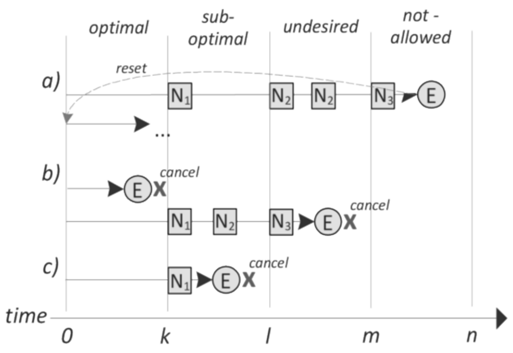


Figure 4 – Example notification lifecycles

In Figure 4 we illustrate several examples of our notification life cycles, including health alerts, data entry, and data synchronization reminders. Notification lifecycles are divided into sequential stages (e.g., optimal, sub-optimal, etc.). To reflect the increased urgency, these progressive stages typically (a) increase the notification's obtrusiveness, by using more intrusive interactions (e.g., sounds and vibrations) and different icons and content templates; and (b) increase the notification frequency per elapsed time. For each progressive stage, interaction configurations and notification frequencies can be predefined for each category (i.e., alerts and reminders). In contrast, registered notifications will each have *individual* lifecycles specifying when each progressive stage starts and finishes; supported events and their effects on the lifecycle; and notification-specific data (e.g., action about which the patient is being reminded) to be filled into content templates.

Each notification lifecycle starts when the notification is registered and is *reset* or *cancelled* when relevant events are received. Figure 4 shows four progressive stages, where the *optimal* stage indicates the initial timespan where no action needs to be taken. The s*ub-optimal, undesired,* and *not-allowed* stages indicate stages with increasing notification obtrusiveness and frequency. In example *a* the system registers a reminder for data entry at install time, where the patient starts receiving reminder instances (N_x symbols) after *k* time has passed. In this case the time *k* depends on the data entry schedule. For example, if the patient needs to enter blood pressures every day, a suitable value for *k* could be 24 hours. Later, the notification frequency and obtrusiveness increase as the *undesired* and *not-allowed* stages are entered until the patient enters the health data. In that case, an event (E circle) is received and the lifecycle is *reset*, starting again from 0. In example *b*, two reminders for data synchronization are registered each time after the patient entered health data while lacking connectivity. In this case, suitable times for *k* depend on the necessity for timely health data uploads (e.g., depending on the urgency of server-side decision support). After *k* time has elapsed, the patient will start receiving reminders of increasing obtrusiveness whereby the start times of subsequent stages (*l* and *m*) will likewise depend on the necessity for timely uploads. For the first reminder, connectivity is re-established before any notification is issued, resulting in an event that *cancels* the reminder's lifecycle. For the second reminder, the patient finally encountered a Wi-Fi hotspot after entering the *undesired* stage, again resulting in the cancellation of the reminder lifecycle. In example *c*, the local CDSS issued an alert after inferring a problematic health issue. In case of alerts, the *optimal* stage will most likely be skipped (setting *k* to 0) so the patient is directly notified of the health issue. In this example, the patient responds immediately after the first notification, generating an event that *cancels* the alert. For instance, patients may respond by simply indicating they will follow-up on the alert, or the alert may include an option to directly dial the clinic or emergency room. The notification lifecycle is terminated after the *not-allowed* stage. We note that notification lifecycles likely depend on the particular clinical setting and chronic illness being managed, and thus should be specified by domain experts.

It is important to note that persistent notifications may result in alert fatigue if not reconciled with the patients' personal schedule [12]. In particular, if patients continuously receive notifications but often are not able to directly perform the associated actions (e.g., health data entry), they will simply start ignoring them. Therefore, we have implemented the functionality to personalize notification lifecycles (within acceptable limits). Currently patients can configure the starting time of the *sub-optimal* stage for *data entry* reminders, thus delaying the time at which they start receiving reminder instances. In doing so, the patient may better align data entry with their own personal schedule. In case this configured time overlaps with subsequent stages (e.g., undesired), the system will display warnings of increased severity during configuration, informing the patient that the current schedule does not allow for an optimal follow-up, but that they should proceed if this is the only timing corresponding to their personal schedule. Setting this time beyond the start time of the *not-allowed* stage is not permitted as it would result in too long a delay for data entry reminders to still have purpose.

Figure 5 shows an example of an Android reminder instance that includes options for directly performing the task (“*Do”*) and personalizing the particular reminder's lifecycle (“*Settings”*). For both our Android and iOS implementations, platform-specific features are utilized to minimize battery usage: on Android, the Notification Engine is only made active by the operating system when notifications need to be issued; on iOS, future notifications to be fired are registered with a platform-specific service. Currently, persistent reminders are registered for data entry and synchronization, while alerts are one-off notifications. Finally, AF-MPD provides patients with a chronological overview of notifications.

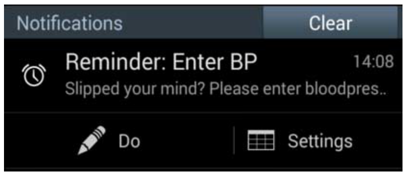

Notifications
Clear
Reminder: Enter BP 14:08
Slipped your mind? Please enter bloodpres..
Do
Settings

Figure 5 – Example Android notification.

Implementation

The mobile patient diary was implemented using the Adobe PhoneGap [13] cross-platform development framework. In this framework, mobile apps are developed using HTML, CSS, and JavaScript, which are deployed as embedded websites in automatically-compiled native apps. Platform-specific components such as databases, platform-specific notification code, and connectivity listeners are then added to these native apps. Currently, the mobile patient diary is deployed on Android and iOS and the necessary platform-specific components were developed for both platforms.

Related Work

Hommersom et al. [3] examine mobile clinical decision support for multiple chronic illnesses. In order to provide clinical decision support, they discuss the possibility of using rule-based systems, Bayesian networks, and case-based reasoning. However, this work seems to be in its early stages, summarizing requirements, challenges, and opportunities of mobile smart assistants but not supplying concrete solution designs. Ambroise et al. [2] present a concrete mHealth solution for collecting health data, targeting a group of chronic illnesses. It features a hybrid architecture with lightweight reasoning occurring on the client side and more heavyweight reasoning taking place on the server. Locally collected health data is synchronized at frequent intervals with the server as well. Although the system has an automatic notification system, no details are given on how notifications are managed. At the same time, no consideration is given to increasing patient adherence and the discussed scenarios do not focus on aligning the system with existing, complex clinical pathways.

iALARM [12] is a language for specifying intelligent alerts and introduces the concept of an alert lifecycle. Alerts are triggered by monitoring temporal clinical data whereby alert reminders are fired in case the problem persists and no appropriate alert response has been given. The proposed language focuses on monitoring medical factors in clinics and issuing alerts to healthcare professionals. As a result, reminders only exist in the context of clinical alerts and cannot be issued separately for performing other self-management actions. In the same vein, issues such as notification lifecycle personalization and indicating the increased urgency of notifications (e.g., by increasing interaction obtrusiveness) are not considered.

Conclusion and Future Work

We presented a mobile intelligent cross-platform patient diary for the self-management of chronic illnesses, focusing on AF. Its main features include a local CDSS that allows for timely health alerts without connectivity and a notification system that supports evolving, event-based, and personalizable notification lifecycles. The supplied patient personalization options, combined with CDSS and notification lifecycles customizable towards different clinical settings, allow for improved adoption by healthcare providers and patients. We note that the patient diary comprises generic, configurable components that are re-usable for self-managing other chronic illnesses as well. The mobile patient diary will be used in a large-scale clinical trial (5,000 AF patients).

Future work includes studying additional opportunities for patient personalization, for instance by automatically suggesting suitable data entry times based on analysis of the patient's schedule (e.g., in iCal format). Since only manual data entry is supported at this point, further efforts also involve attempting to plug in wireless measurement devices and using the mobile device's camera to scan medication barcodes and automatically add them to the system. Importantly, we also aim to apply the patient diary to self-manage other chronic illnesses and investigate the extent to which other types of reasoning (e.g., Bayesian networks) would be more suitable for realizing clinical decision support in those cases.

Acknowledgements

This project is funded by a Bayer Healthcare research grant.

References

[1] World Health Organization. Preventing chronic diseases: a vital investment. WHO Global Report, 2005.

[2] Ambroise N, Boussonnie S, and Eckmann A. A smartphone application for chronic disease self-management. In: Proc Conf MobileMed, 2013; pp. 1-4.

[3] Hommersom A, Lucas PJF, Velikova M, Dal G, Bastos J, Rodriguez J, Germs M, and Schwietert H. MoSHCA – My Mobile and Smart Health Care Assistant. In: Proc IEEE 15th Int Conf on e-health networking, applications and services (HealthCom), 2013; pp. 188-192.

[4] Packer TL, Boldy D, Ghahari S, Melling L, Parsons R, and Osborne RH. Self-management programs conducted within a practice setting: Who participates, who benefits and what can be learned? Patient Educ Couns 2012, Apr; 87(1): 93-100.

[5] Abidi SSR, and Abidi SR. An Ontology-Driven Personalization Framework for Designing Theory-Driven Self-Management Interventions. In: Process Support and Knowledge Representation in Health Care, 2013, pp. 97-112.

[6] Van Woensel W, Al Haider N, Ahmad A, and Abidi SSR. A Cross-Platform Benchmark Framework for Mobile Semantic Web Reasoning Engines. In: P. Mika et al. (Eds.) ISWC 2014, Part I, LNCS 8796. Switzerland: Springer, 2014; pp. 389-408.

[7] Van Woensel W, Al Haider N, Roy PC, Ahmad AM, and Abidi SSR. A Comparison of Mobile Rule Engines for Reasoning on Semantic Web Based Health Data. In: Proc of 2014 IEEE/WIC/ACM Int Joint Conf on Web Intelligence (WI) and Intelligent Agent Technologies (IAT). 2014; pp. 126-133.

[8] The task force for the management of atrial fibrillation of the European Society of Cardiology. Guidelines for the management of atrial fibrillation. European Hearth Journal 2010: 31: 2369-2429.

[9] Lin HJ, Wolf PA, Kelly-Hayes M, Beiser AS, Kase CS, Benjamin EJ, and D'Agostino RB. Stroke severity in atrial fibrillation: The Framingham study. Stroke 1996: 27 (10): 1760-1764.

[10] Health Device Profile (HDP). https://developer .bluetooth.org/TechnologyOverview/Pages/HDP.aspx

[11] CCS Atrial Fibrillation Guidelines Committee. 2014 Focused Update of the Canadian Cardiovascular Society Guidelines for the management of atrial fibrillation. Canadian Journal of Cardiology 2014: 30: 1114-1130.

[12] Klimov D, and Shalar Y. iALARM: An Intelligent Alert Language for Activation, Response, and Monitoring of Medical Alerts. In: D. Riaño et al. Eds. KR4HC 2013/ProHealth 2013, LNAI 8268. Switzerland: Springer, 2013; pp. 128-142.

[13] Adobe PhoneGap. http://phonegap.com/

Address for correspondence

William Van Woensel and Syed Abidi, NICHE Research Group, Faculty of Computer Science, Dalhousie University, E-mail: william.van.woensel@dal.ca & ssrabidi@dal.ca