Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 56 (2015) 49 – 58
Procedia Computer Science

The 12th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2015)

An Adaptive User Interface in Healthcare

Elhadi M. Shakshuki[a]*, Malcolm Reid[a], Tarek R. Sheltami[b]

[a]*Acadia University, 15 University Ave, Wolfville, NS B4P 2R6, Canada*
[b]*King Fahd University of Petroleum and Minerals, Dhahran, 31261, Saudi Arabia*

Abstract

Healthcare is a broad subject with many different challenges, yet it is important and relatable to everyone. The aging Baby Boomer generation is an important healthcare issue today. In Canada, and many other developed nations, the number of citizens reaching the age of retirement and seniority is growing faster than the rate of citizens working and providing health related services. As people age they tend to require more frequent checkups and health services, ultimately putting a bigger resource drain on healthcare infrastructure. New advancements in Computer Science and Engineering are allowing the development of next generation applications with the purpose of providing healthcare services in a cost effective and efficient way. This paper proposes a multi-agent system for tracking and monitoring health data for patients. Furthermore, agents within the system use reinforcement learning techniques to build an adaptive user interface for each human user. The actions and behaviour of users are monitored and used to modify their respective user interface over time. To demonstrate the feasibility of the architecture, two scenarios are provided. We conclude with several possible future directions for this research.

Peer-review under responsibility of the Conference Program Chairs

Keywords: Multi-Agent Systems; Healthcare Technology; Adaptive User Interface; Reinforcement Learning

1. Introduction

Healthcare is a broad subject with many different challenges, yet it is important and relative to everyone. The aging Baby Boomer generation is an important healthcare issue today. In Canada and many other developed nations the number of citizens reaching the age of retirement and seniority is growing faster than the rate of citizens working and providing health related services[1, 2]. In the first case, when people age they tend to require more frequent

* Corresponding author. Tel.: 1-902-585-1524
E-mail address: elhadi.shakshuki@acadiau.ca

Peer-review under responsibility of the Conference Program Chairs
doi:10.1016/j.procs.2015.07.182

checkups and health services, ultimately putting a bigger resource drain on the healthcare system. This is a clear sign that new ideas and innovations are required to help provide the quality of healthcare demanded by the population.

Multi-Agent Systems coupled with advancements in Engineering, such as miniature low power processors, is new a technology from the field of Computer Science offering new means of providing healthcare and related services[3, 4].

In the second case, there is an increasing trend in healthcare monitoring with personal technology, a movement sometimes referred to as *eHealth*, and *mHealth*[5]. People have the desire, and for the first time in history, the technology required to actively monitor and track their own health metrics. In both cases however, a problem exists when a user is unable to operate the interface to his or her technological device. There could be any number of reasons for this, but for the remainder of the paper we will focus on: environmental or physical causes. Environmental causes are external factors like low or bright light, while physical causes are user relative properties such as poor vision or Parkinson's disease. Regardless of which type, these reasons may make the operation of personal devices, and in turn the healthcare application, difficult or impossible. In this paper, we are tackling the following question: What if an interface could adapt over time, to meet the needs of a user? This subject is an active branch of research in the artificial intelligence and multi-agent system communities[6, 7]. The theory behind such an interface requires a multi-agent system to use a machine learning technique that helps to build, test, and evaluate a policy for each user.

The remainder of this paper will focus on building a system to monitor and share the health metrics of specific users. Gathered data can be accessed either by the patient to whom it belongs or a qualified health professional. The health metrics tracked may include properties such as pulse, blood oxygenation, or any attribute that is able to be monitored using digital sensors. Human users, both patient and healthcare professional, access the system through a mobile interface. Furthermore, the system uses reinforcement learning techniques to build an adaptive user interface for each user. A hypothetical scenario for a system like this is monitoring patients in a nursing home.

2. Related Work

This section provides a brief description of the fundamental techniques that are utilized to enhance the user experience.

2.1. Adaptive User Interface and Reinforcement Learning

An adaptive user interface is a user interface, which adapts itself by changing its layout and rearranging the screen elements based on user needs[8, 9]. Such an interface is useful in many areas, particularly for mobile applications. The possible applications of adaptive user interfaces however, are not limited to mobile devices. This concept can be extended to environments such as aircraft displays and military devices, to name two examples [8, 6].

Constructing an adaptive user interface is made possible by the use of models, which form a standard for creating the interface [8]. Two types of models are used for accommodating individual user requirements: (1) an error model, and (2) an interface model [8]. Ultimately, these models describe the behavior of users, which can be later extended into the adaptive interfaces. Models are updated by observing the actions of the user within the system.

Similar research has been conducted with respect to adaptive interfaces for users of mobile devices in an external environment, namely outside[8]. Environmental issues such as fog and low light as well as user issues such as poor motor skills and the complexity of an interface can affect the ability of a user to interact with a mobile device[7, 8]. One strategy presented for overcoming this problem was to use reinforcement learning to build an error model of a user's interactions with the interface and adapt accordingly[7].

The theory of reinforcement learning provides a normative account, deeply rooted in psychological and neuroscientific perspectives on animal behaviour, of how agents may optimize their control of an environment[10]. To use reinforcement learning successfully in real-world situations, agents are confronted with the difficult task of deriving efficient representations of the environment, and use them to generalize past experience to new situations[10].

2.2. Body Area Sensor Network

A Body Area Sensor Network (BASN) is one type of Wireless Sensor Network. BASNs primary goal is gathering information about a human body using sensors located in and on the target body[11]. Although the two types of networks are closely related, BASN have the following distinguishing properties: (1) agents are heterogeneous, (2) agents are few in number, and (3) wireless signal quality is variable[3]. The first property results from the requirement that nodes must be placed in or on the body, and therefore must be built to be as small and unobtrusive as possible. This limits the functionality of each node to only what is required for the task at hand[11]. To illustrate this point, consider two BSAN nodes: one for measuring brainwaves and another for measuring heart rate. Both must be able to communicate, but it would be redundant in size, cost, and power consumption to have the hardware for sensing both stimuli built into each.

The second property also follows from the requirement that nodes must be placed in or on the body and made to be small and unobtrusive. Therefore agents must be few in number because there is no room for multiple nodes of the same functionality. The argument can be made that this reduces the overall redundancy of the system. However, using more nodes requires more power to be expended by the system as well as more equipment to be worn by the user, both of which are ultimately undesirable results[12]. Finally, the last property is a constraint of the environment in which BASN operate. The human body is not a conductive medium for wireless signals and can physically block or interfere with signal transmission between nodes[12]. This problem is challenging, as the body is a dynamically moving object. Therefore, to be truly robust and efficient a BSAN system must take this into account when communication between nodes occurs.

3. Environment

The healthcare environment in this study consists of Doctors, Patients, and Software agents. Patient users (PUs) are the people being monitored by the proposed system architecture. Each patient is monitored via a wireless body area sensor network (BASN). Each BASN can be tailored to meet the needs of the patient and monitor desired vital statistics. The BASN interfaces with a central coordinator device carried by the patient as illustrated in Figure 1. In this paper, we make the assumption that the coordinator device is a smart phone with a touch-screen interface.

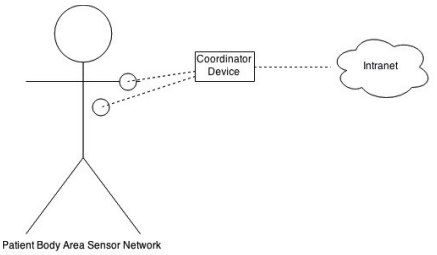

Figure 1: example body area sensor network worn by patient.

The software agents in the environment are: user agent (UA) and resource agent (RA). Each human user is assigned a UA. There will be many user agents in the system, and one resource agent. The RA is responsible for writing historical data to a central server repository. This allows for the later retrieval and analysis of important patient data. An additional role of the Resource Agent is to act as a security authority and validate the credentials of users requesting patient data.

User agents can be operated by either a doctor or a patient, and must behave accordingly. Patients will be wearing a BASN for monitoring health data, which will act as sensor input for the user agent representing them. A patient can use his or her coordinating device to view real-time as well as historical health data that they own. Doctors will have a UA representing them and their authority as a doctor, which in turn provides access to patient health data, both historical and real time. Figure 2 shows a visualization of the system architecture.

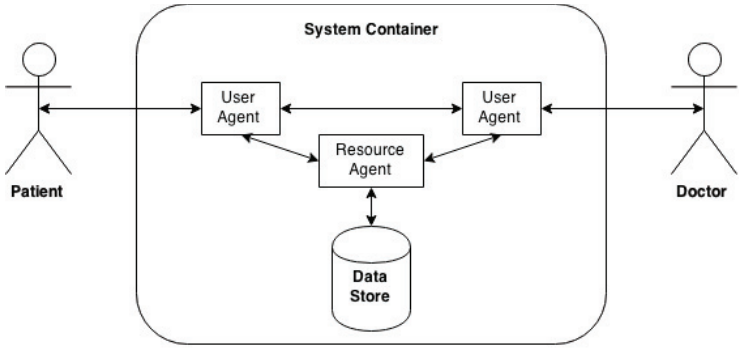

Figure 2: System architecture overview.

4. System Architecture

The system architecture consists of two types of agents: user agent (UA) and resource agent (RA). The function of user agent is to act as the interface for the human user to interact with the system. The function of the resource agent is to act as an authority for storing data as well as retrieving historical data for both patients and doctors. Our focus in this paper is on the user agent that has the capability of enhancing the user experience using reinforcement learning. The User Agent has the following main components: sensor, communication, learning, and user model, as shown in Figure 3.

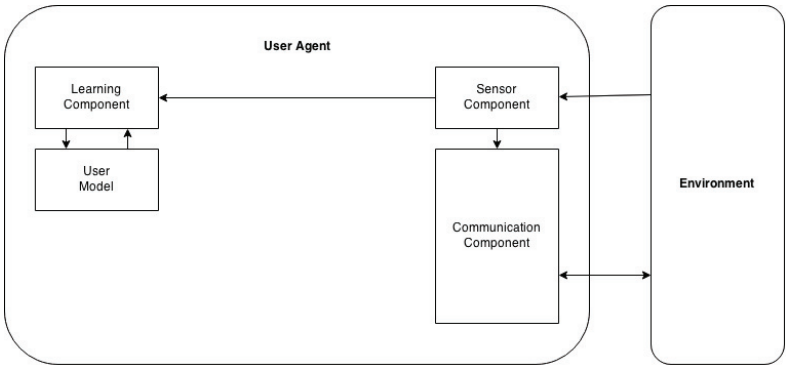

Figure 3: internal architecture of the User Agent, consisting of four components.

The agent's sensor component is responsible for receiving health metrics from the environment via the coordinating device. Once received and sanitized, health data and any relative agent communications are then passed on to the communication and learning components for further processing. The agent's communication component is responsible for two tasks: (1) transmitting data to the resource agent for historical recordkeeping, and (2) responding to requests by doctor operated user agents. The user model component contains data relating to the usage habits of the human user, both their actions as well as errors. This model is initiated to a zero sum default state and is modified over time by the Learning Component as the agent attempts to learn the behavior patterns of the user. Lastly, the following section provides a detailed description of the reinforcement-learning component.

4.1. Reinforcement Learning Component

The reinforcement learning component (RLC) is responsible for learning the behaviour of a user by tracking his or her historical actions as well as errors in interface use. The RLC can be broken down into three sub-components: (1)

learning, (2) evaluating, (3) adapting. See Figure 4 for an illustration of how the sub-components relate and interact with each other and the system environment around them.

Figure 4 shows the steps as information flows through the RLC. The first step consists of data entering the RLC from the agent's sensor component. This data may consist of user action choices, such as viewing a health metric, as well as information about how the user interacted with the system interface. The data is processed by the learning component (LC). This component parses the information received into two categories: *action*, and *behaviour*. The action group consists of user choices and preferences, while the behavior group consists of statistical interactions with the interface, such as coordinates of touch-screen clicks or ambient light levels. In some cases a user action might not be included with the usage data, an example of this would be a patient with poor motor skills being unable to tap a button. In the later example, the learning component needs to learn and recognize different types of behavior. Once actions and behaviours are identified they are passed to the Evaluating Component (EC).

The evaluating component is responsible for pulling relative information, with respect to the received usage data, from the agent's perceived concept of the user, which is stored in the user model component. Next, the EC compares the new usage actions with the existing policy for those actions. If there is no data about that action in the policy already, then nothing happens and the data is passed on to the next component. However, if there is information in the agent's user model then the EC must compare it with the new usage data and determine the differences. Next, data is passed to the Adapting Component (AC).

The adapting component receives the new usage data as well as the related data from the user model for that action. The AC must determine what actions need to be taken and update the user model accordingly. We provide an example scenario in the next section, in which a user has trouble tapping an interface button. The RLC observes the users' difficulty interacting with the button and updates the screen accordingly.

4.1.1. Example of Learning

Lets assume that the user model contains a vector representation of the users previous actions, such as: $UserActions(a_1 \ldots a_n)$. Where there are n actions in the user model. Furthermore, the user model also contains a vector representation of the users erroneous interactions with the user interface: $UserErrors(e_1 \ldots e_n)$. Where n is the number of erroneous actions observed so far.

Hypothetically, a user unable to tap a button on screen may be capable of tapping near the button. Every touch interaction with the interface is observed by the RLC. Therefore, if a touch interaction is within a predefined threshold distance of a known button, or any selectable screen object, the action is identified as erroneous with respect to that action and recorded in the *UserErrors* list. Lets assume that the user in question suffers from hand tremors and is still unable to tap the button described earlier after five attempts.

Each interaction is observed by the RLC and unsuccessful attempts are stored in the *UserError* list. The Evaluating Component within the RLC is able to determine if the new action being observed has a history of being difficult for the user, by examining the actions in the *UserError* list. A threshold value for erroneous interactions with screen objects can be defined, at which point the interface is modified by the Adapting Component to meet the need of the user.

As a hypothetical course of action in this case, the adaptive component expands the size of the button by a factor of 10%.

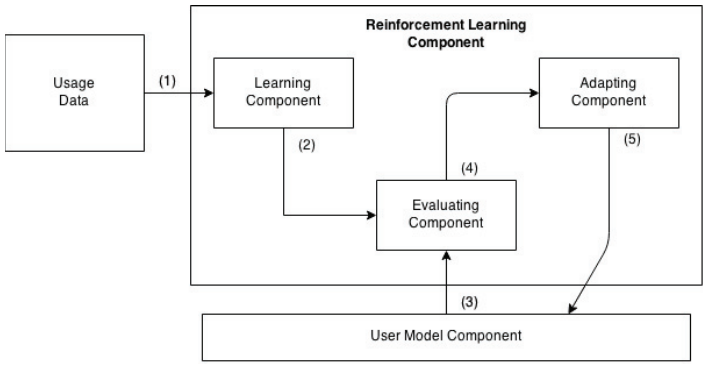

Figure 4: Reinforcement Learning Component.

To visualize how the system works two scenarios are provided. In the first scenario we consider a patient who would like to view his or her pulse data. In the second scenario we consider a doctor who requests the pulse data for the past 24 hours from four of his patients.

4.1.2. Scenario I

The situation describing this scenario is described first followed by the sequences of occurring events between the user and the system. A patient, say Bob, spent the past 40 minutes walking around the building. He would like to view his pulse data for the past hour.

Let's assume we provide the user with a smart device as shown in Figure 5. First, Bob must interface with a coordinating device, specifically the smart device provided to him. Interaction requires launching an application interface to the system. One open, Bob taps the view health data button to view health information. A dialog box will pop up and requiring Bob to select the type of health information he would like to view, in this case heart rate. An additional dialog box will pop up and prompt Bob for an optional time range to view, in this case the range is the previous hour. Figure 5 contains a mock up interface for the view health information screen.

When Bob presses the view health data button, the user agent (UA) representing Bob receives this request and simultaneously passes on the request to the resource agent (RA) and to its own internal learning component. Assuming Bob submitted the request for data at 11:04am, the learning component (LC) receives a user action of *request(pulseData, 10:04, 11:04)*. The LC further receives data from the coordinating device about the users erroneous interaction with the interface. The learning component quantifies observations into two categories of actions and behavior and forwards the data on to the evaluating component.

The evaluating component (EC) reads data from the agent's user model and determines if the received actions and behaviors are new data for the model, or already known. New data, or data requiring modification in the model are passed on to the adapting component.

The adapting component (AC) combines the data in the user model with the current usage data and updates the model. In this scenario, the user model would contain an entry that user Bob has a preference for viewing heart rate data, specifically in the previous 60 minutes. Furthermore, the AC must adapt the interface as needed if the agent perceived difficulty interacting. While this is happening, the Resource agent prepares and delivers the requested data to the user agent for display to the end user Bob.

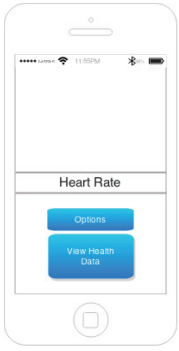

Figure 5: mock interface for requesting health data on the coordinating device.

Figure 6 shows a formal use case diagram for the scenario described above. The two actors are a patient and user agent, which are Bob and his respective user agent in this case. Furthermore, the resource agent is present, but only interacts with the user agent to provide historic data.

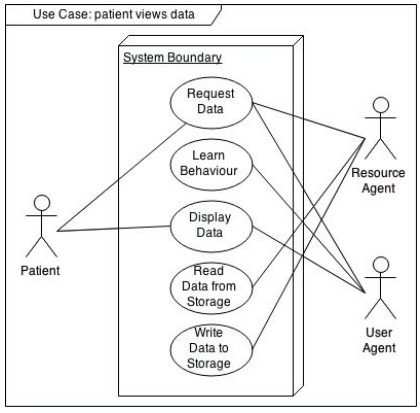

Figure 6: use case diagram of a patient requesting health data.

4.1.3. Scenario II

The situation describing this scenario is described first, followed by the sequences of occurring events between the user and the system. A doctor, Alice, cares for several patients suffering from heart complications. She would like to view the pulse rates of four of her patients, for the past 24 hours.

Lets assume Alice is provided with a smart device, like the one in Figure 5. First, Alice interfaces with the system via the provided coordinating device. This interaction requires launching an application on the device. Next, before performing any operations with other users, Alice must prove her identity by authenticating with credentials known only to her. The Resource Agent evaluates the credentials. Once successfully authenticated, Alice requests to view health data by tapping on the view health data button. A dialog box appears asking Alice the type of health information to view, she will specify heart rate. Another dialog box appears prompting Alice to input the names of patients for whom to the data should be gathered. Finally, the last dialog box appears asking for a time range for the requested data.

Lets assume the four patients are: Bill, Jeb, Bob, and Vail. Furthermore, lets assume that Alice is making this request at 2:30, on January 4, 2015. The request generated by Alice will be *RequestData(Credentials, <Bill, Bob, Jeb, Vail>, <Pulse Rate>, <January 3, 2015, January 4, 2015>)*. Alice's credentials are encrypted and passed along with the request in an attempt to minimize system attack vectors, due to the nature of such personal data. The Resource Agent authenticates the credentials, before any data is actually retrieved.

The second parameter is a list of patient identifiers, while the third is a list of the requested health metrics. Lastly, the fourth and optional parameter is the time range for the request. In the event no time range is specified, the system returns the current known values.

The User agent receives the request and begins a parallel process of: (1) passing this request on to either the related user agents (in the case of real-time data monitoring) or the resource agent (in the case of historical data monitoring), and (2) observing and learning the behaviour of the doctor's actions and interactions with the system.

The reinforcement learning component of the user agent works the same for doctors as it does for patients. That is, the Learning component receives the action request as well as information about erroneous interactions with the interface. In this case, the *UserActions* model would be updated to reflect the new *RequestData* action and its associated preferences. The associated preferences are the patient names, requested health metric, and specified time range in the request. The most frequently associated preferences may be used to streamline the user interface for doctors in a hurry. A formal use case diagram of this scenario describe above is shown in Figure 7. The actors are Alice the doctor, the user agent representing her, and the resource agent.

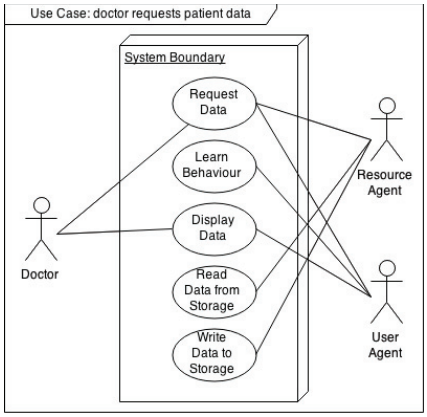

Figure 7: use case diagram of doctor requesting patient information.

5. Implementation

A formal code implementation has not yet been completed for this system; however, some suggestions and recommendations for a starting platform are suggested.

5.1. System Container

The system container is a platform in which agents exist and operate. Management tasks such as the creation and deletion of agents must be achieved with a central control or management resource. Our suggested software package for this task is the Java Agent Development (JADE) Framework. The JADE framework provides the critical agent communication and management features required by our proposed system architecture. JADE is an advantageous choice as an agent platform because it is well maintained and documented. Current academic research is being conducted into the impact and feasibility of developing MAS applications using JADE[13]. An example of our proposed system container running on JADE is shown in Figure 8.

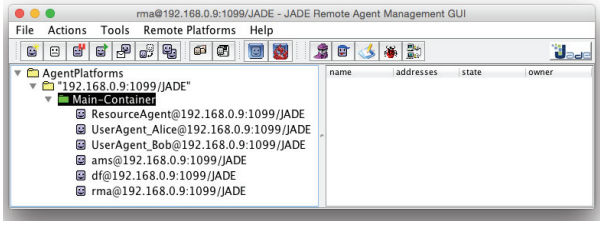

Figure 8: proposed system agents running on the JADE platform.

5.2. Historical Services

Historical services in the context of the proposed system mean the storage of patient health statistics in an easily retrievable manner. While requests to read and write data are handled by the resource agent, the data must still be stored somewhere. The obvious choice for this is a database, of which there are many choices available. Our recommendation for a starting platform Oracle's free and open source MySQL database. This is because of the open source licensing, easy integration with java, and a large number of thoroughly documented examples. There are a number of healthcare applications already using MySQL to manage data[14].

5.3. Body Area Sensor Network

Developing or obtaining the hardware and software required for a functioning body area sensor network is a challenging and expensive task. Alternatively, a simulation of a BASN may be used in the development of our proposed system architecture. The simulation of a patient worn BASN could be accomplished with the JADE platform mentioned previously. Setting up a new system container to represent the BASN worn by the patient. There, the various sensors and coordinating device can be simulated using agents, as shown in Figure 9. This is only intended to be a proof of concept simulation, as eventually a real BASN will be required to evaluate the usefulness of the proposed system.

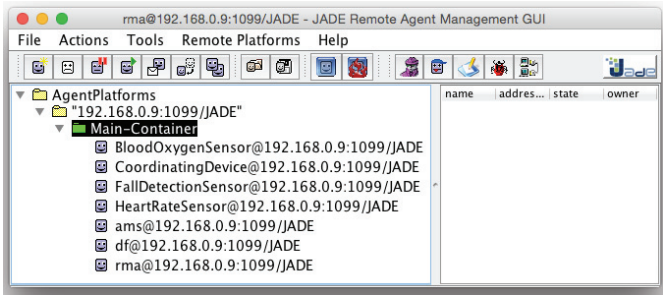

Figure 9: simulated BASN and coordinating device, in JADE.

6. Conclusion

This paper presented architecture for a multi-agent system with the purpose of tracking and reporting human health information, such as heartbeat or blood oxygen levels, to both patients and doctors. The feasibility of the proposed system architecture is demonstrated with two hypothetical scenarios. Furthermore, several software packages are presented as a starting platform for an implementation of this system. The future work for this research includes examining the possibility of using other machine learning techniques such as supervised and unsupervised learning to build more accurate user models. Furthermore, implementing a real world functioning example of this system is the next step in evaluating the feasibility of the proposed system architecture.

Acknowledgements

The authors would like to thank the research and graduate office at Acadia University for their support. The authors also like to thank King Fahd University of Petroleum and Minerals for supporting this research.

References

1. CBC News. (2012, May) CBC News. [Online]. HYPERLINK "http://www.cbc.ca/news/canada/canada-has-higher-proportion-of-seniors-than-ever-before-1.1151526" http://www.cbc.ca/news/canada/canada-has-higher-proportion-of-seniors-than-ever-before-1.1151526
2. CBC News. (2015, January) Why doc? Solution to doctor shortage may be more than money. [Online]. HYPERLINK "http://www.cbc.ca/news/canada/newfoundland-labrador/why-doc-solution-to-doctor-shortage-may-be-more-than-money-1.2929966" http://www.cbc.ca/news/canada/newfoundland-labrador/why-doc-solution-to-doctor-shortage-may-be-more-than-money-1.2929966
3. Ashraf Darwish and Aboul Ella Hassanien, "Wearable and Implantable Wireless Sensor Network Solutions for Healthcare Monitoring," *Sensors*, vol. 11, no. 6, pp. 5561-5595, May 2011.
4. B. H. Calhoun et al., "Body Sensor Networks: A Holistic Approach From Silicon to Users," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 91-106, August 2011.
5. Prasan Kumar Sahoo, "Efficient Security Mechanisms for mHealth Applications Using Wireless Body Sensor Networks," *Sensors*, vol. 12, pp. 12606-12633, September 2012.
6. Siddharth S Rautaray and Anupam Agrawal, "Version based hand gesture recognition for human computer interaction: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1-54, Jan. 2015.
8. Karthik Narayanan Ramalingam, *Modeling error-based Adaptive User Interfaces*. Aimes, USA: Iowa State University, 2011.
7. Wiard Jorritsma, Fokie Cnossen, and Peter van Ooijen, "Adaptive support for user interface customization: a study in radiology," *International Journal of Human-Computer Studies*, vol. 77, no. 1, pp. 1-9, May 2015.
9. Ana L. C. Bazzan, "Beyond Reinforcemnet Learning and Local View in Multiagent Systems," *Künstliche Intelligenz*, vol. 28, no. 3, pp. 179-189, August 2014.
10. Volodymyr Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529-533, Feburary 2015.
11. M. A. Hanson et al., "Body Area Sensor Networks: Challenges and Opportunities," *Computer*, vol. 42, no. 1, pp. 58-65, January 2009.
12. Cho Namjun, Bae Joonsung, and Yoo Hoi-Jun, "A 10.8 mW Body Channel Communications/MICS Dual-Band Transceiver for a Unified Body Sensor Network Controller," *IEEE Journal of Solid-State Circuts*, vol. 44, no. 12, pp. 3459-3468, December 2009.
13. Jörg P. Müller and Klaus Fischer, "Application Impact of Multiagent Systems and Technologys: A survey," in *Agent-Oriented Software Engineering*, Onn Shehory and Arnon Sturm, Eds. Berlin, Heidelberg, Germany: Springer-Verlag, 2014, pp. 27-53.
14. MySQL Corporation. (2015, January) MySQL in Healthcare. [Online]. HYPERLINK "https://www.mysql.com/why-mysql/isv-oem-corner/healthcare/" https://www.mysql.com/why-mysql/isv-oem-corner/healthcare/