Multi-Agent Solution for a Cloud-based e-Health Application

Cosmin Toader
Computer Science Department
University POLITEHNICA of Bucharest
Bucharest, Romania
cosmintg@gmail.com

Nirvana Popescu
Computer Science Department
University POLITEHNICA of Bucharest
Bucharest, Romania
nirvana.popescu@cs.pub.ro

Vlad Ciobanu
Computer Science Department
University POLITEHNICA of Bucharest
Bucharest, Romania
vlad.ciobanu@cs.pub.ro

***Abstract*—Ubiquitous computing for innovative health-care systems, services and applications became more and more connected to Cloud systems and the applications required a scalable, reliable and secure environments, the containerized environments representing suitable solutions. In this paper we deal with the advantage of multi-agent systems and IoT for e-health applications in scalable platforms. We investigated how Cloud-based model can be adopted. Thus, an architecture was designed for a medical data based intelligent system that processes the collected data and takes the right decisions based on them. This work also presents a drill down into the mobile client architecture focusing on the implementation of some of the simple periodic agents and combining them into complex periodic agents. The pulse-oximetry sensors were considered in this research. In this context, a heart rate simple periodic agent and a SpO2 simple periodic agent were developed. Our model considers that the data measured by the sensors will be used to monitor health status of a patient in real time or to discover threshold values for the data that can be further careful be analyzed and interpreted as a medical pre-diagnosis. (*Abstract*)**

Keywords—multi-agent systems; e-health systems; cloud computing; scalability.

I. INTRODUCTION

IoT (Internet of Things) offers great solutions when talking about dealing with the increase of calculation power and computational intelligence in various devices. Their presence is remarked in several applications in e-health, most of them in patient surveillance area [1]. The data is processed in real-time and adjust to changing conditions, not only to collect and stream.

Typically, in multi-agent systems (MAS), a mobile agent is a portion of autonomous software, capable of migrating between nodes of a network [2]. Combining it with IoT, we can get to certain forms of mobility that can physically move the device. E-Health area became a research zone where multi-agent systems have been quickly introduced. Also MAS were involved when the data is pre-processed on the client-side as a model and sent to the server. In this case the client must just update the model [3, 4]. This is a "model-to-data" approach and was successfully used when dealing with latency and processing power. Mobile agents have been also used in telemedicine [5]. This approach involves on one hand medical services and on the other hand it involves computer and communication technologies. In this context, the solution advanced here is represented by the design of a safe agent-based telemedicine and a P2P networking architecture.

In this context, the scope of the research work presented in this project is to combine multi-agent systems and IoT, having together all their advantages. Thus, it was proposed an architecture of a medical data based intelligent system that processes data and takes different decisions accordingly. Different bio-sensors are involved in the hardware part, their role being to collect data from a human being (temperature, ECG, pulse oximetry, etc.). The server side cumulate the data and send it to the certain medical authorities.

There were three modes of transmission for the mobile component: normal data transmission, emergency data transmission and audio/video stream. In the first case, the periodic medical data is taken from the patient and further sent to the cloud by means of MLLP (minimal lower layer protocol). The HL7 protocol [6] is used for serialization. In the second case, just a set of selected data is sent (2G connections). Some parameter thresholds, like minimal O2 level will be relevant to decide switching to this mode. In the last case, the user can initiate this transmission.

The e-health platform consists of different biosensors and raspberry-pi. The data measured by the sensors will be used for real-time monitoring of the patients or to discover threshold values for the data that can be further careful be analyzed and interpreted as a medical pre-diagnosis. To store the medical data, to aggregate and process them, a cloud component is included that will allow the medical personnel to access them.

The paper is structures in seven sections. After Introduction, Section II presents a Cloud server architecture and a Mobile client architecture. In Section III we discussed the communication protocols and the security aspects of such a system. The multi-agent architecture is presented in Section IV. We conducted several experiments and the results are analyzed in Section V. Some advantages and disadvantages were underlined in Section VI. The paper ends with Section VII presenting the conclusions.

978-1-5386-4444-7/18/$31.00 ©2018 IEEE

II. THE MOBILE CLIENT AND THE CLOUD SERVER ARCHITECTURES

The "*Cloud Service*" module is the main one of the architecture, being responsible with getting data from the mobile clients. The server will also process the received data and will decide to store it or to raise an event handler for notifying the appropriate module if it is necessary or to realize the audio/video streaming connection (Fig. 1).

Nevertheless, reliability and manageability are some of the main advantages of cloud computing [7]. These features are very important for service-dedicated platforms since cloud computing is much more reliable and consistent than in-house IT infrastructure. More details can be seen in paper [13].

The second server module is *the web component*, having a classical MVC architecture. The .Net Core technology have also been chosen this time too. The web server works closely with the cloud service and it transmits web content to the medical staff. SignalR is a technology used by the web component that allows different notifications to be sent real time to the web browser just in case of the calls made by the emergency module.

Another component that plays an important role is *the event handler* for handling real-time events when the patient's vital signs reach a critical threshold value, or in the case of accidents with multiple victims needing emergency medical intervention. By sending relevant alerts to specific authorities, the staff will figure out the magnitude of the incident and respond accordingly. A repository design pattern was developed to store data for the web component and the cloud service.

In order to collect data from sensors or from the user, three agents were designed and form the mobile client. These agents negotiate between them to decide which mode to use in order to send the data (Fig. 2):

- Periodic analyzer agent will take data periodically from the sensors and if they are considered relevant, they will be sent them to the cloud.
- Emergency agent has the main role to collect data from the sensors and to decide if there are critical parameters.
- Manual handling agent can override the decisions of the above mentioned two agents and it is able to initiate the audio/video call to the medical staff.

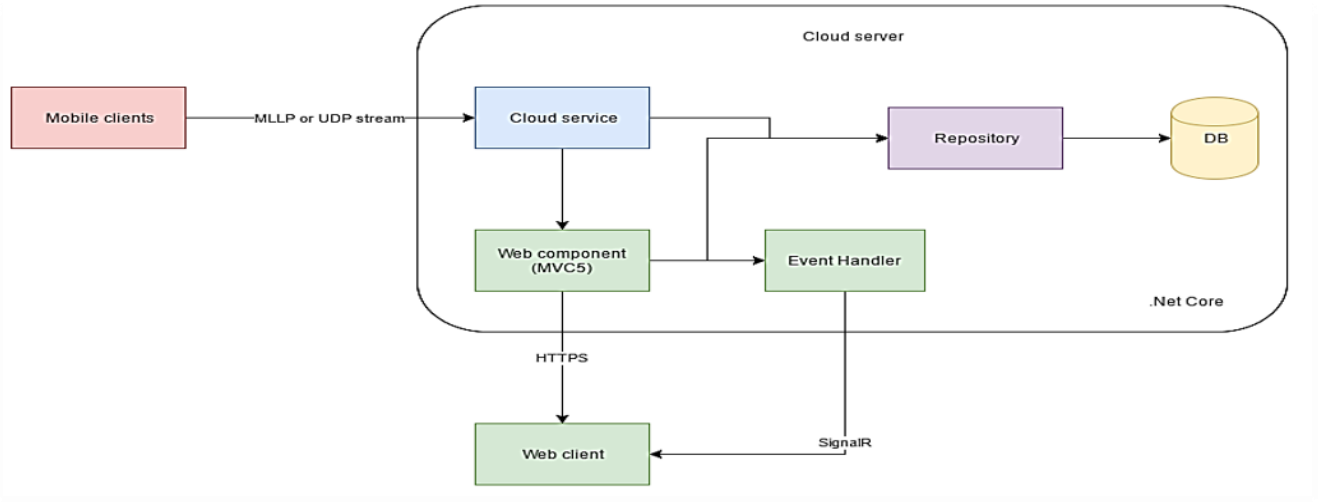

Fig. 1. The cloud service architecture

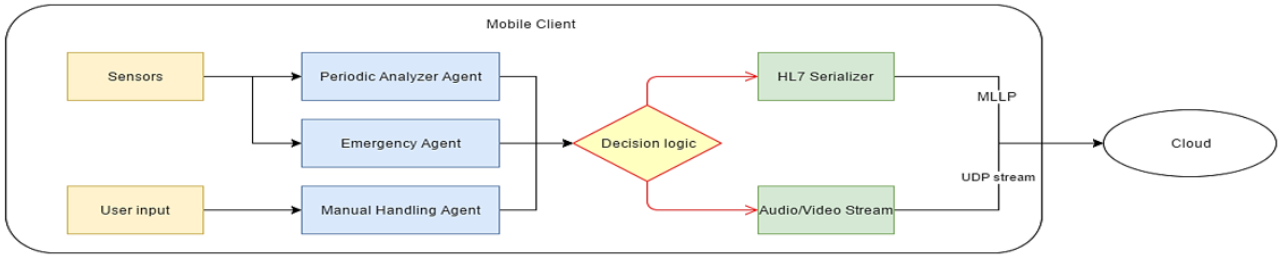

Fig. 2. The mobile client architecture

III. COMMUNICATION PROTOCOLS AND SECURITY ASPECTS

The MLLP protocol is a minimalistic framing protocol in the OSI session layer, over TCP/IP. The purpose of this protocol is to work like an interface between HL7 and the network using a minimal overhead. Although it is not mandatory, this protocol has become a standard for transmitting HL7 messages over TCP/IP. The detection and correction of errors are realized by the lower level protocols (TCP/IP). The HL7 message is wrapped between a leading and a trailing non-printable character that will not appear in the actual content of the HL7 message.

MLLP has several limitations since the block is framed by single byte values. Therefore, all the characters sent throough the MLLP block need to be encoded using an encoding that will not conflict with the delimiter bytes (usually single-byte encodings like UTF-8 are permitted). Also, both the sender and receiver need to implement the same encoding, meaning that the integration could not be possible unless both parties mutually agree on an encoding. Another disadvantage comes from the fact that it doesn't provide any support for encryption, since it is a very light protocol, but additional protocols can be used on top of its layer (however, the sockets can be encrypted using TLS). To ensure security, best practices are to use a VPN or a secure IP connection (IPsec).

HL7 messages can be sent via HTTP as well, as an alternative to the standard MLLP. It has important advantages since it is widely used among developers, well understood and well documented, it is an application layer protocol in the OSI stack allowing authentication and it permits a character encoding to be set in a standardized way.

The security via HTTPS is improved, allowing both transport-level and message-level security. It is widely supported across platforms.

To conclude**,** for both security and commodity reasons, HTTPS is recommended, but however, MLLP brings up a huge advantage: lower bandwidth. After this critical analyze, the recommendation for our project would be to use HTTPS for the normal data transmission, and use MLLP only for emergency data transmission mode, where speed and latency matters.

Going further with our analyze HL7 v3 is an interoperability specification for health and medical transactions. Its aim is to support all healthcare workflows. The development started in 1995 and the initial publication happened in 2005. HL7 v3 is more of a standard than its predecessor, version 2, being a less customizable framework. However, it is not backwards compatible. The HL7 organization came up with a newer standard that incorporates the best of HL7 v3 while solving problems that the version 3 had met, called FHIR (Fast Healthcare Interoperable Resource).

FHIR was built to improve security over a HTTPS layer and offers a strongly defined model with easier customization. It is based on REST which makes it so much easier to implement and use for organizations and developers.

From the advantages of HL7 v3, we can consider the fact that it can be used via MLLP, therefore optimizing the traffic, at the cost of security and it is older on the market and more widely spread.

IV. MULTI-AGENT SYSTEM ARCHITECTURE

This section describes the in-depths for the periodic analyzer, one of the most important components of the architecture. Its purpose is to analyze data from the medical sensors over time, and make decisions to raise a certain warn-level to the server. Besides the warn level, the system will use the "model-to-data" mechanism [4,5] that will help by greatly reducing the server processing power, latency and server disk size.

The main logic of the periodic analyzer agent is to use a pattern detection mechanism to detect irregularities from the medical sensor data. We will use multiple periodic analyzer agents, each one running on a separate thread and communicating with each other via the "blackboard" mechanism. The blackboard represents a shared memory zone where every agent can read or write data using thread synchronization techniques. Those agents need to coordinate, since they share a common goal: detecting irregularities that might represent real-life problems in the patient's medical parameters. Therefore, a communication technique is needed. Further, having collectively motivated agents with common goals, the blackboard system was chosen.

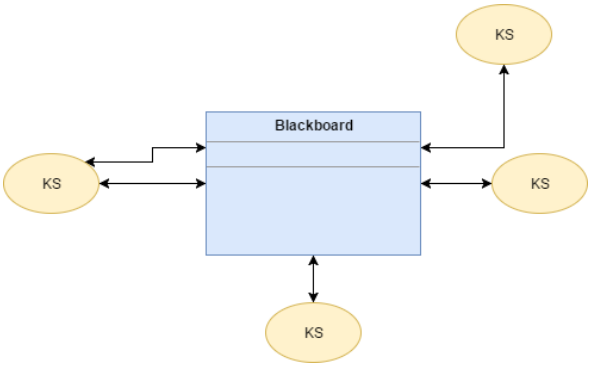

Fig.3 Agent representation (KS = knowledge source)

A. Agent's Own Goal

Before getting to the common goals of the agents, let's start with the purpose of an individual agent. One periodic analyzer agent is responsible, in the first place, with the data gathered from one medical sensor. So, let us use the example of having only one sensor and one agent (let's take the heart rate sensor as a very simple example). The sensor will write a new data input every x amount of time. For the heartrate sensor, for example, let's assume that will tick every X seconds, count the number of beats, then extrapolates the value for 1 minute to get a "fairly" accurate beats-per-minute. That means we get a new value every X seconds that we could store in a queue to have the whole history of how our heartrate changed over time. We will use queue, because it will have a limit depending on both memory size and the computational power. For this example, we will assume the sensor will send a new value every 5

seconds and we have a limit of 720 in our queue. Every time we get our 721st value, we will remove the oldest entry from the queue so we permanently have only 720 values. That means we track the record of the heart-rate for the last hour.

The agent will work with this queue (a linked-list could be used as well, we'll stick with queue just to make the theoretical part easier). Each time the queue updates, the periodic agent's individual purpose is to analyze patterns in this queue and detect irregularities. The agent's logic in detecting patterns from only one sensor will be highly scalable and configurable from an XML file. For this example, a pattern could be a significant raise in the heart-rate every 10 minutes with a one-minute error tolerance (this pattern is used just for a pure theoretical example, it doesn't mean anything, maybe only the fact that the patient is doing some effort every 10 minutes within an hour).

For the detected pattern, the agent could raise a certain warn level, for example setting a score for the pattern's importance from 0 to 100. The agent should only handle one pattern at a time, for one sensor only. The agent computes the pattern every time the sensor ticks, or every x amount of ticks, a configurable variable.

B. *Agent's Common Goal*

The simplest agent model is able to detect one type of pattern from one sensor only and assign a score to it. In order to achieve more from our medical sensors, the agents can communicate with each other in order to detect even more complex patterns. On each agent computation time, that happens every X amount of sensor ticks, a pattern must be computed (that may, or may not be found). Once a pattern is found at T(x) (time of tick X), the pattern data will be written in the blackboard, to share the knowledge with the other agents.

The data will contain the pattern ID and the pattern score (zero if no pattern was found), in a historical array of patterns. This way we can form more complex agents, that are able to detect patterns based on both their own sensor data, and on other agents' already computed patterns from T(x-1). As an example, a complex periodical analyzer agent could search for patterns based on data collected from two other agents, one that is monitoring the heart-rate like the one presented earlier, and another agent that is searching for patterns in the patient's temperature change.

This way, every T(x) we have a heart-rate pattern from T(x-1) and a temperature pattern from T(x-1). As a very simple example, if we detect that the patient's sensor is indicating a significant increase in heart-rate every 10 minutes, we can assume that the patient is just doing some physical effort every 10 minutes.

But if the same pattern occurs in the temperature sensor, and we detect a growth in temperature every 10 minutes, in the same amount of time (with an error tolerance), we can be even more certain that the patient is doing physical activities.

However, detecting heart-rate increases and constantly having a low body temperature, could signal a combined pattern with a higher warn level.

The system will allow multiple pattern configurations and will create an agent for each pattern. In this way, we can create a pattern based on other patterns, and each agent will work to collectively contribute to that complex pattern, forming a "pyramid" of agents (Fig.5).

A. *Scalability*

The periodic agent's architecture is highly configurable via xml files, therefore patterns can be added dynamically to the system. An update mechanism is foreseen to deploy the configuration files from the server-side, allowing to easily update the agents' logic with minimal effort and no compiling will be needed. The complexity of the agents is scalable only depending on the client machine's performance. A technique of computing patterns server-side will be also developed, using the "model-to-data" mechanism.

II. EXPERIMENTAL EVALUATION

For the experiments, the agent implementation has used the pulse-oximetry sensor with the Raspberry-pi hardware. The aim of the implemented system is to gather data from the sensor, analyze it and raise warning levels for each agent accordingly.

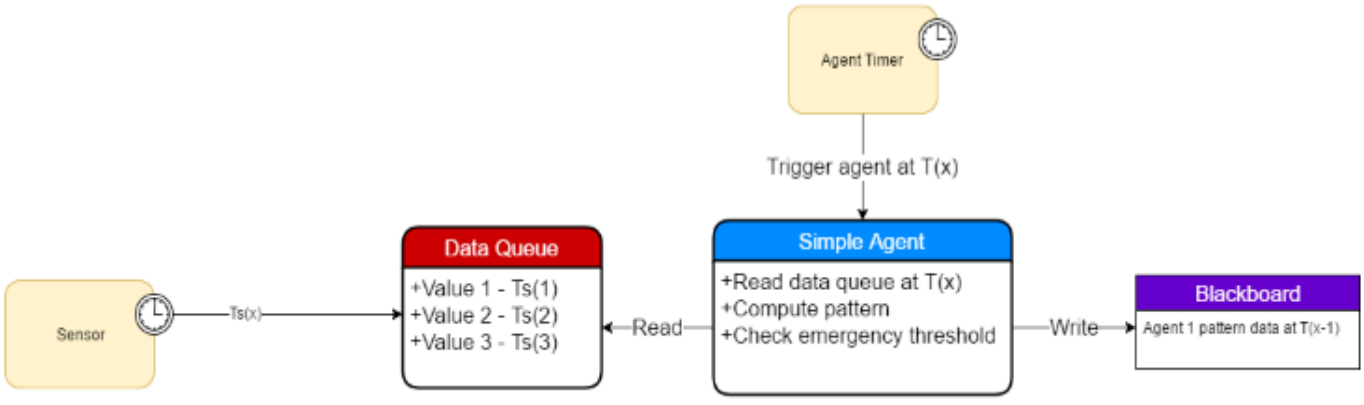

Fig.4. Simple periodic agent architecture

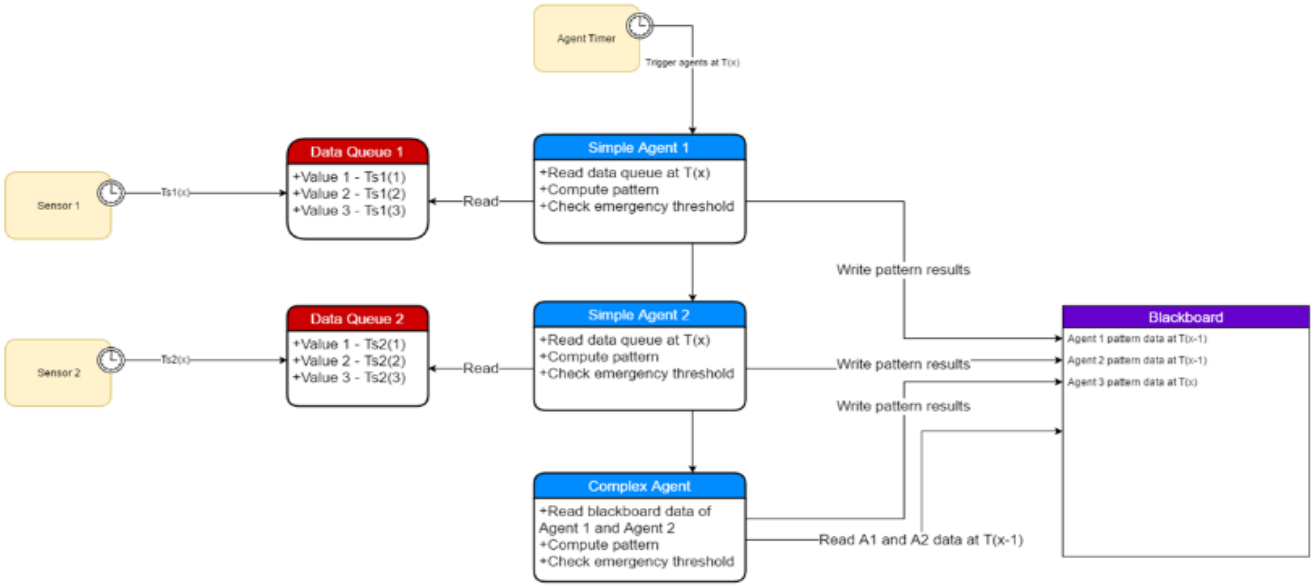

Fig.5. Agent combination example

The warning levels should be transmitted eventually to the cloud and list possible scenarios.

Pulse-oximetry is a noninvasive method that continuously gathers data and gives estimates of the arterial hemoglobin oxygen saturation (SpO2), along with the heartrate that the sensor computes with the plethysmographic signal [8]. The arterial hemoglobin oxygen saturation from this sensor is only an estimate, the standard in getting the SpO2 level consisting in performing an arterial blood gas analysis, which is more accurate, but also invasive, since it requires a drawn sample of arterial blood [9]. The pulse-oximetry sensor is easy to use, being usually placed on a finger tip of the patient. It uses a pair of light emitting diodes, red and infrared. The de-oxygenated hemoglobin will attract more infrared radiation and vice-versa for the oxygenated Hb [10, 11].

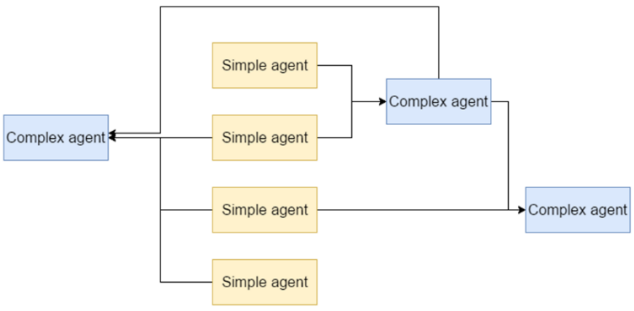

Fig.6. Complex periodic agent architecture

A pulse-oximeter measures: SpO2 – the percentage of blood that is loaded with oxygen and the pulse rate – beats per minute. The pulse-oximetry parameters are:

- A normal pulse is usually between 60 and 80 bpm
- Regular arterial hemoglobin oxygen saturation (SpO2) should be higher than 95%
- A SpO2 level lower than 92% means hypoxemia
- Standard sensor error is around 2% (a value of 82% could mean something between 80% and 84%, the values vary in time)

The accuracy of detecting this only by the increased heartrate and the SpO2 level is low, but it can increase significantly if we add a blood-pressure sensor and an ECG.

Gathering data

The pulse-oximetry sensor feeds data periodically, refreshing both the heartrate value and the SpO2 level. For the heartrate, an integer value is received, representing the BPM (beats per minute). For the SpO2, also an integer value is received, representing a percentage of blood that is loaded with oxygen. Those two data-feeds will be treated separately, and every simple-agent connected to this sensor will receive data from one of those feeds.

Heartrate simple periodic agent

First, we will build a simple periodic agent that will receive data from the heartrate feed only. Its only purpose is to detect patterns in the heartrates and send warning levels accordingly. For each periodic agent, be it simple or complex, we will assign warning levels from 0 to 100, 100 being the highest. The normal BPM value for a healthy person is between 60 and 80. The aim for this agent is to detect anomalies in the heartrate, but the task is not easy. A higher heartrate value could mean an increased warning level, but it could also mean the individual is just practicing some sports, so the BPM will increase for that period.

The next thing that needs to be taken into consideration, is the patient's age. We shall consider receiving the patient's age as a parameter from some input method, when configuring the

device. The first parameter is: age = 30.

TABLE I. RANGE AND WORN LEVELS FOR HEART RATE

Range level	Range (BPM)	Warn level	Possible reason
-5	0-20	0	Sensor error
-4	20-30	70	Very low pulse
-3	30-40	60	Very low pulse
-2	40-50	50	Low pulse
-1	50-60	30	Low pulse
0	60-80	0	Normal pulse
1	80-100	30	Pulse increasing
2	100-120	35	Pulse increasing
3	120-135	40	Pulse increasing
4	135-150	45	Pulse increasing
5	>150	80	Very high pulse

The maximum heartrate is computed by the following formula: max = 220 – age. Therefore, the maximum BPM value for a 30 years old patient is 220 – 30 = 190 BPM. The target heartrate for exercising is somewhere between 50% and 70% of the max heartrate. So, for a person practicing sports, or exercising at a gym, for the example above of 30 years old, the target heart-rate during exercising is 190*50% = 80 and 190*70%=133. Therefore, for a 30 years old person, detecting a heartrate between 80 and 133 BPM could mean exercising, so the warning level for those values will be low.

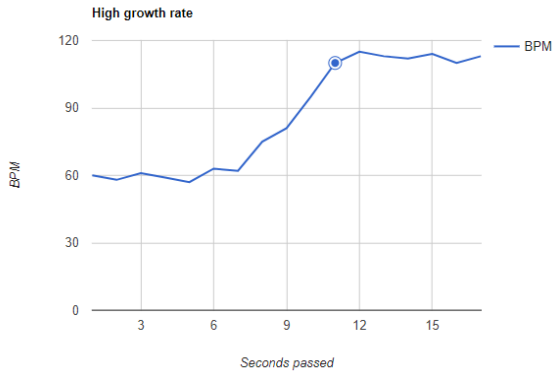

Fig.7. Detecting a growth in heartrate

For the heartrate, we have a collection of data from the feed, viewed as a key-value pair of <timestamp, bpm>. We will define two types of patterns:

- Progressive pattern – linear growth or decrease of the values
- Repeating pattern – regular increases or decreases at the same amount of time

Detecting a growth in the heartrate can tell us something about the patient based on the growth rate. For example, a very high growth rate could tell us the patient has suffered some kind of shock, while a lower growth rate could mean the patient just received some bad news, or started to do some physical activity (see Fig. 7).

A very high growth rate in a heartrate will have a higher warn level than a lower one. Therefore, whenever the high growth rate is detected, we will add 30% to the warn level.

Example 1: normal pulse = 60. Grew to 110 over 2 seconds. The new warn level will be 35 + 30% * 35 = 45.5.

We will consider a high growth for the heartrate any growth from the normal BPM range (range 0) to a 2 or more range levels higher in less than 5 seconds. For a medium growth rate, which means more than 5 seconds until shifting ranges, but less than 1 minute, we will increase the warn level with 10% only. And for low growth rates, we will keep the same warn level according to the table mentioned above. Detecting a decrease in the heartrate, similar to the growth, can mean something. A sudden decrease of the BPM can mean hypovolemia (low blood quantity in body). We will consider a sudden decrease any decrease from the normal BPN range to a range level of -2 or less in under 5 seconds, and we will increase their warn levels by 30% (see Fig. 8).

Example 2: normal pulse = 60. Decreased to 30 over less than 5 seconds. Warn level will be 60 + 30% * 60 = 78.

Example 3: normal pulse = 60. Decreased to 0 over less than 5 seconds. Warn level will still be 0, since this will most likely mean the sensor was disconnected.

For a medium decrease range, we will consider a window from 5 to 60 seconds, and we will increase the warn level by 10%. As for low decrease ranges, we will keep the default warn levels.

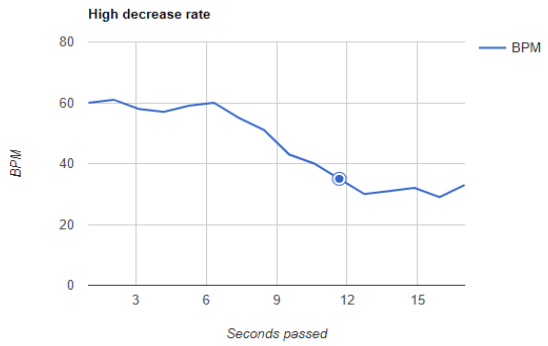

Fig.8. Detecting a decrease in heartrate

SpO2 simple periodic agent

The SpO2 sensor provides integer values within a range from 0 to 100, and an error rate of 2. A normal person has the arterial hemoglobin oxygen saturation level higher than 95%. Anything lower represents a warning sign. Hypoxemia can be called for a saturation level lower than 92%.

For the SpO2, we have a collection of data from the feed, viewed as a key-value pair of < timestamp, saturation_ percentage >. Detecting a decrease in SpO2 level over more than 2 hours could mean either a mixt dysfunction or a respiratory dysfunction. As an example, a patient with a chronic problem like COPD with an usual saturation level between 90 and 95, who developed recently a pneumonia could raise the warning level higher, by hitting a saturation level of 85-90 (or even lower) requiring a non-invasive ventilation (CPAP), the respiratory effort being too high.

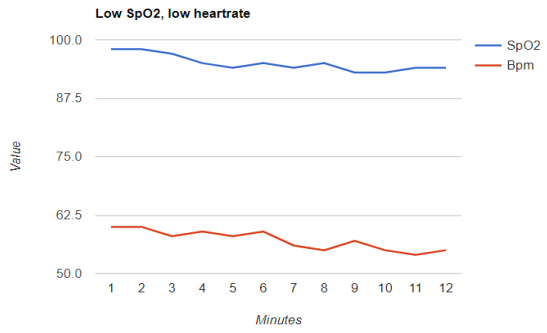

Fig.9. The output of a complex periodic agent

Detecting a sudden decrease in SpO2 level (less than 10 minutes) could raise the warn level significantly. For example, in a myocardial infarction phase, because of vascularization decrease of a myocardial zone, the SpO2 level will decrease suddenly. Depending on the zone and dimension, the SpO2 level can decrease more sudden or slower. We consider a sudden decrease any shift between 2 range levels that occur in less than 10 minutes, with a difference higher than 4 percentages.

Complex periodic agent - BPM and SpO2

This complex agent's purpose is simple: it needs to combine the output of multiple simple agents (2 in this case) and aggregate the result or find a pattern in the combined outputs. For example, if SpO2 agent reports a low arterial hemoglobin oxygen saturation level, and the heartrate agent reports a low heartrate, the patient is most likely suffering from a sleep apnea [12] (see Fig.9).

III. ADVANTAGES AND DISADVANTAGES

The main advantage over this solution is the use of the complex agents. While other works present solutions of monitoring and alerting the parameters from various sensors, the proposed architecture will combine the data from multiple inputs, getting more accurate warning levels.

By just using simple agents, the project acts just like any of the existing systems for sending alerts to a centralized system, but combining them brings a great improvement.

Another advantage is the scalability of the project. More agents can be implemented and each agent can request data computed from any of the other agents.

A disadvantage at this point is the hardware limitation, since some part of the data is computed client-side. Another disadvantage is that computing data on the server side will not be available for the other agents on the client side.

IV. CONCLUSIONS

The paper presented an architecture designed for a medical data based intelligent system that processes the collected data and takes the right decisions based on them. The work emphasized the advantage of multi-agent systems and IoT for e-health applications in scalable platforms. It was also investigated how cloud-based model can be adopted.

By using only the pulse-oximeter sensor, the system can already use three agents of which two simple and one complex, that combined can raise warn levels about a patient based on only 2 data feeds: the heart rate, and the arterial hemoglobin oxygen saturation level. The warn levels are set accordingly, and even if both sensors give a low warn-level, combined, they can result in a much higher warn-level that could help detect critical situations right in time. The complex agent for the pulse-oximetry sensor is still not well defined, some medical research being needed, but the technical part is in place, proving that a complex agent can be used very efficiently by combining at least two simple agents, either by interpreting their warn levels, either by extracting a pattern based on their outputs.

In the future works, the architecture will be amplified with other sensors like the blood-pressure sensor and ECG. Input from those sensors could really improve the warn-level accuracy for multiple diagnostics, the most important so far being the coronary syndrome.

ACKNOWLEDGMENT

This work is supported by GEX 21/2017 grant financed by University Politehnica of Bucharest.

REFERENCES

[1] A. L. A. Lella, “The u.s. mobile app report (whitepaper)” comScore, 2014.

[2] M. Wooldridge – “An introduction to Multiagent systems”, published by John Wiley & Sons, ISBN-10: 0470519460, 2009.

[3] L.C. Wang, X.W. Meng, Y.J. Zhang, “Context-aware recommender systems”, J. Softw., no. 23, 2012, pp. 1-20.

[4] B. Hidasi and D. Tikk, “Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 7524 LNAI, no. PART 2, 2012, pp. 67–82.

[5] W. Hsu and J. Pan, “Secure Mobile Agent for telemedicine based on P2P networks”, Journal of Medical Systems, 2013 Jun; 37(3), DOI: 10.1007/s10916-013-9947-2.

[6] E. M. Rac-Albu, V. Ciobanu, M. Rac-Albu, N. Popescu, „Interoperability of Medical Data Through eHealth service in Romania”, IESS 1.6:

International Conference on Exploring Service Science, Bucharest, May 2016, pp 11-21.

[7] R. Basmadjian, H. De Meer, R. Lent, G. Giovanni "Cloud computing and its interest in saving energy: the use case of a private cloud", J. of Cloud Computing, Vol 1, issue 5, 2012, DOI: 10.1186/2192-113X-1-5

[8] A. Jubran. "Pulse oximetry". *Critical Care*. Springer, 19:272, 2015.

[9] M.W. Wukitsch, M.T. Petterson, D.R. Tobler, J.A. Pologe, "Pulse oximetry: analysis of theory, technology, and practice", *Journal of Clinical Monitoring*, Vol. 4, Issue 4, 1988, pp 290–301.

[10] P.F. Wouters, H. Gehring, G. Meyfroidt, L. Ponz, J. Gil-Rodriguez, C. Hornberger. "Accuracy of pulse oximeters: the European multi-center trial". Anesth Analg., 94(1 Suppl):S13-6, 2002.

[11] R.K. Webb, A.C. Ralston, W.B. Runciman, "Potential errors in pulse oximetry, II. Effects of changes in saturation and signal quality". Anaesthesia. 1991.

[12] E. Kufoy, J.A. Palma, J. Lopez, M. Alegre, E. Urrestarazu, J. Artieda and J. Iriarte, "Changes in the Heart Rate Variability in Patients with Obstructive Sleep Apnea and Its Response to Acute CPAP Treatment". PLOS Medicine Journal, DOI 10.1371/journal.pone.0033769, 2012.

[13] C. G.Toader, "Multi-Agent Based e-Health System", Proc. of 21st IEEE International Journal on Control Systems and Computer Science CSCS21, Bucharest 2017, ISSN: 2379-0482, DOI: 10.1109/CSCS.2017.37, pp. 696 – 700.