Sensor Observation Service based Medical Instrument Integration

Data Capture, Analysis and Rendering

Chirabrata Bhaumik, Amit Kumar Agrawal, Suman Adak, Avik Ghose, Diptesh Das
TCS Innovation Lab
Tata Consultancy Services
Kolkata, India
e-mail: c.bhaumik@tcs.com, amitk.agrawal@tcs.com, suman.adak@tcs.com, avik.ghose@tcs.com, diptesh.das@tcs.com

***Abstract*—With advancement in technology, wireless portable medical instruments are quite common these days. However, these devices come with heterogeneous interfaces, which make them difficult to integrate the data generated in a common repository. Moreover, if a solution is created to integrate a few devices, adding a device with a new interface requires major effort and rework. This paper addresses this problem and suggests a method for integrating various health care instruments for use at home/small clinic on a generic Open Geospatial Consortium (OGC) Sensor Web Enablement (SWE) based Internet of Things (IoT) platform and exposing healthcare as a service. Finally the paper demonstrates how an example portal can be used to render the data on the server to end-users who may be patients, medical technicians or expert medical practitioners. This will enable governments or hospitals to offer a SMART Healthcare Services for citizens.**

Keywords- Smart Health; IoT; SOS; Healthcare; Mobile Application; Sensor networks; Intelligent Healthcare; Wireless Medical Devices; Smart HealthCare Analytics

I. INTRODUCTION

In developing countries like India, the general mass is not aware of the role of medical history in case management. Even if the medical history is kept, it is in the form of paper reports. These reports must be carried along while consulting with a healthcare professional. Also, these reports are at a higher risk of getting mutilated or lost compared to digital records. Hence, there is a need to digitize & store medical data in a central repository, which can be accessed by different parties (patients, home physicians, expert medical practitioners, medical institutions, insurance parties etc.) as and when required with proper access and authentication mechanism in place.

With the advancement in technology, wireless portable medical instruments are becoming reality. These devices are easy to operate, and can be used at home or small clinic with little or no training. These devices have the capability to communicate the medical observations to external world via digital interfaces. However, these devices come with heterogeneous interfaces and the data generated by them is in different formats. So, to store the data generated by these devices in a central repository, either manual intervention is required or a system must be there in place to take care of the idiosyncrasies of the varied interfaces & data.

Numerous relevant studies [3][4][5][7][8][9][10][16] have been found, out of which [3] and [4] are closer to the problem at hand:

Gavin E. Churcher [3] discussed the application of Sensor Web Enablement to a telecare application. It is a monitoring solution, which uses “dbFeeder” to directly insert data to SOS database. The major limitation of this work is not using the standard SOS interface to insert the observations.

M.V.M. Figueredo [4] explains a telemedicine system where the monitoring device communicates the medical data over RS-232 channel to a mobile phone, which in turn posts the data to a server through internet. Server stores the medical data in a custom designed database and presents the data as printed form or xml. The major limitation of this approach is the custom database schema, which necessitates a lot of rework to add a new monitoring device.

To address all these challenges, the authors have designed an Internet of Things (IoT) platform and offer healthcare as a vertical service to it. The IoT platform is generic enough to support other verticals as well. The bare minimum requirements for this IoT platform are as follows:

- Open Source implementation of Sensor Observation Service (SOS)[12] from 52North.org[2],
- Postgresql database with postgis plugin
- Proxy to handle REST requests from the clients and convert them to SOS compliant XML
- Tomcat servlet container

This platform provides a unified XML interface to communicate with the clients, and uses its own schema to store the data irrespective of incoming data formats. It eliminates the need to redesign the schema if data in a new format needs to be stored.

To provide healthcare service as an offering from this IoT platform, we implemented the following modules:

- A User Datagram Protocol (UDP) adapter
- A Bluetooth adapter

These adapters are small modules meant to bridge the gap between device interfaces (UDP and Bluetooth) and IoT platform interface (i.e. XML). We have chosen only these two types of adapters because we could get hold of wireless medical devices belonging to either of these two categories.

Section II describes the architecture of the whole system. In Section III, we describe the application components and data flow. In Section IV, we provide the results and snapshots from our portal. Section V mentions the concluding remarks, and finally, Section VI acknowledges contributions made by various people in the organization to carry out this work.

II. SYSTEM OVERVIEW

Figure 1 illustrates the basic architecture of our proposed system. It is a layered architecture, which comprises of IoT platform, Plugin layer and Device Layer. Now, we shall take a closer look at each of these layers from the architectural perspective.

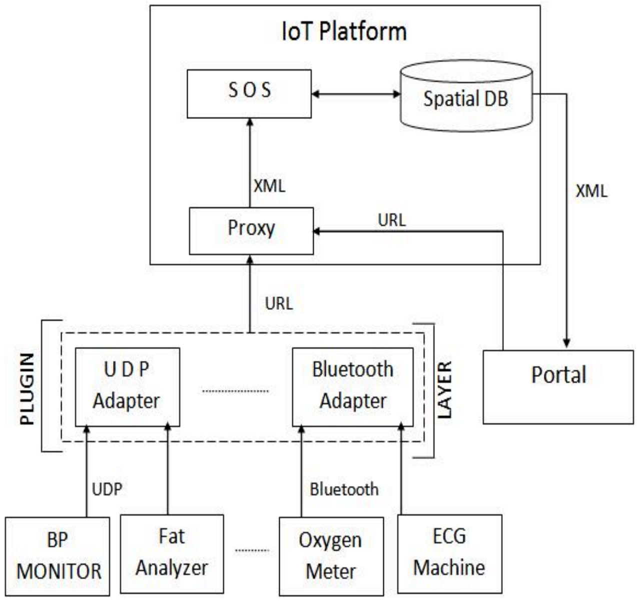


Figure 1. System Architecture

A. *IoT Platform*

The major offering from the IoT Platform is Sensor Observation Service (SOS). SOS is a part of the Sensor Web Enablement (SWE) [1][15] initiative from Open Geospatial Consortium (OGC). OGC standard states that SOS provides a generic model for all sensors and the observations collected by those sensors. It acts as a horizontal service, which can accommodate & store the observations from all types of sensors. It provides access to the sensor data in a standard way irrespective of the sensor type. SOS heavily depends on Observation & Measurement (O&M) standard [13][14] to model sensor observations. The SOS standard is described in full length in [11].

We are using Open Source implementation of SOS from 52North.org. It is a java web application and uses postgresql database to store the observations. It also supports spatial data types by using postgis plug-in from postgres. It accepts the sensor observations as O&M compliant XML document, and provides the response as O&M compliant XML document. This is the unifying ground on which observations from varied sensors flow into a common repository.

As O&M compliant XML documents are quite complex and resource consuming, it is difficult for low power devices to generate this kind of complex documents. Also it will involve network overhead, as only a few parameters change across the observations for a sensor. Hence, we have introduced a Proxy component in the IoT platform, which assumes the responsibility of generating the XML document and provides a parameterized Uniform Resource Locator (URL) as its interface. It simplifies the job of the sensors and reduces network overhead.

B. *Plugin Layer*

Wireless sensors do not follow any particular guidelines to provide an interface or observation format. The interface and data format completely depends on the manufacturer's discretion. The conventions used to send data & data format is significantly different for all types of devices.

So, it becomes necessary to provide a layer, which can listen on an interface supported by the sensor, parse the observation data and send it to the Proxy as a parameterized URL. We name this layer as Plugin layer. Whenever a new medical device is introduced with a different interface or data format, a module needs to be incorporated in this layer to support the interface and send the data to Proxy.

C. *Device Layer*

It is a virtual layer, which comprises of all the medical sensors. The diagram in Figure 1 depicts the devices that we have used in our work. Any new medical sensor will simply be added to this layer. Also, we are seeing another trend of new types of medical devices, which are coming out with a closed protocol. What we are observing that manufacturers of these devices are coming out with a portal of their own, and are also providing webservice Application Programming Interface (API) on request. So, the device layer can be easily plugged in as a module to get the medical data record from these portals as webservice API request.

III. APPLICATION DETAILS

We have worked with a few medical devices from ETComm such as Electrocardiogram (ECG) Machine [HC-201B], a FAT analyzer [HC-301W], a Blood Pressure (BP) monitor [HC-502W] & a Blood oxygen meter [HC801B]. These devices are neither 6LowPAN compliant, nor HL7 certified. So, we could not use these standards to communicate with the devices. All these devices map to the Device Layer as depicted in Figure 1.

Based on the communicative capability, these devices fall into two broad categories. The devices, which belong to the first category, communicate with a gateway over 433 MHz RF band. The gateway provides an interface to configure the IP address and port of UDP server where it can send the data. In our case, BP Monitor and FAT analyzer belongs to this category. The other categories of devices communicate with the external world via Bluetooth. In our case, ECG machine & Blood Oxygen Meter belongs to this category of devices.

The data formats for the observations generated by these devices are quite different. BP monitors and Fat analyzer generates UDP payload where as ECG machine and Blood Oxygen meter generates a file.

To handle these interfaces, we have developed UDP adapter and Bluetooth Adapter modules on an Android phone. The UDP adapter module listens on an agreed upon port, identifies the device (both BP monitor and Fat analyzer), parses the data and sends them to proxy as parameterized URL. The Bluetooth adapter accepts the file, identifies the device, extracts and forwards data to proxy as parameterized URL. If the data consists of some binary field (e.g. image), it is sent as attachment to proxy layer and stored in IoT server either in the database, or in file system keeping a reference in IoT database. For instance, ECG Machine generates binary data, which is sent to proxy as attachment along with URL, whereas text data generated by Blood Oxygen meter is sent to proxy as parameterized URL. These modules map to the Plugin Layer as depicted in Figure 1. The medical devices used in this project do not provide alternate paths of communication. Each medical device provides just one interface and Plugin Layer provides listener for that interface (Bluetooth & UDP). XML is used as SOS interface which is part of the IoT platform. Figure 1 depicts the transmission path and role of these technologies in the whole application.

The parameterized URL contains a request string and a set of name value parameters. These request strings map to the SOS core operations. To store the observations, SOS provides two transaction operations:

- RegisterSensor - This operation registers a sensor with the SOS. It is required to register a sensor before storing any observation for it.
- InsertObservation - This operation is meant to insert an observation in the SOS repository.

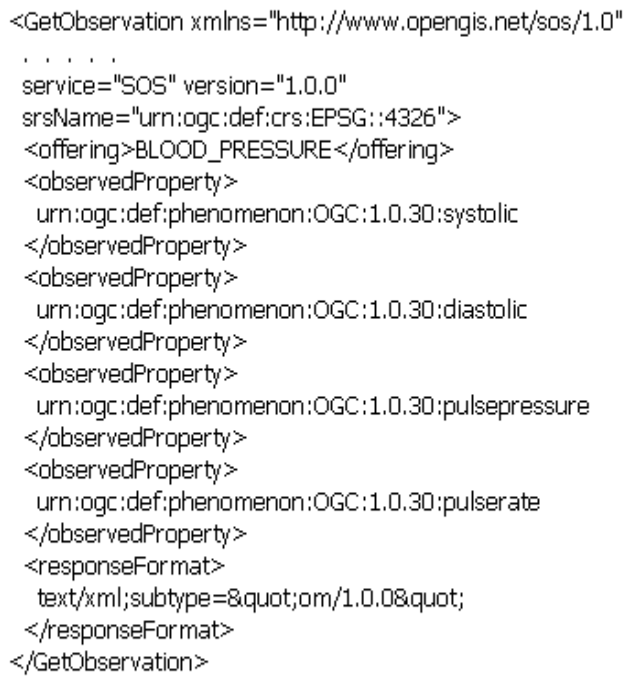
To access the observations, SOS provides following operations:

- GetObservation - It is one of the mandatory core operation provided by SOS. It provides access to sensor observations based on the query. The query can involve both spatial as well as scalar data.

These operations are based on XML interface. The template XML conforming to SOS standard must be created for each combination of device types and SOS operation. All the sensor specific details and observation parameters must be carefully captured and mapped to SOS terminology. The following terms related to SOS demand a brief introduction as they are used extensively in our template XML documents:

- Sampling Time: The time when observation was made.
- Procedure: It is usually a sensor, which generates the observation. We have mapped the device identifier to procedure.
- Observed Property: These are the parameters that constitute the observation.
- Phenomenon: This is a characteristic parameter of the observation.
- Composite Phenomenon: It is sum total of characteristic phenomenon.
- Feature of Interest: It refers to the subject to which the observations relate to.
- Result: It is the value of a parameter created by a procedure.
- Observation: It is a process of observing.

The snapshot in Figure 2 depicts the XML we had designed for GetObservation Service for Blood Pressure.



```
<GetObservation xmlns="http://www.opengis.net/sos/1.0"
 . . . . .
 service="SOS" version="1.0.0"
 srsName="urn:ogc:def:crs:EPSG::4326">
 <offering>BLOOD_PRESSURE</offering>
 <observedProperty>
  urn:ogc:def:phenomenon:OGC:1.0.30:systolic
 </observedProperty>
 <observedProperty>
  urn:ogc:def:phenomenon:OGC:1.0.30:diastolic
 </observedProperty>
 <observedProperty>
  urn:ogc:def:phenomenon:OGC:1.0.30:pulsepressure
 </observedProperty>
 <observedProperty>
  urn:ogc:def:phenomenon:OGC:1.0.30:pulserate
 </observedProperty>
 <responseFormat>
  text/xml;subtype=&quot;om/1.0.0&quot;
 </responseFormat>
</GetObservation>
```

Figure 2. Request XML for BP GetObservation

Now, we take examples of Blood Pressure monitor, and Fat analyzer and illustrate the mapping to XML. As BP Monitor is the sensor, its unique device identifier can be used as procedure (or sensor Id). The four parameters viz. “Systolic Pressure”, “Diastolic Pressure”, “Pulse Rate” and “Pulse Pressure” measured by BP monitor maps to four Phenomena. The group of these four phenomena is mapped to Observed Property & Composite Phenomenon. The observation belongs to the patient, and so we have mapped the patient identifier as Feature of Interest. The result is mapped to the actual value for all the four parameters as reported by the BP monitor. For Blood Pressure, we have mapped the group of parameters as observed property.

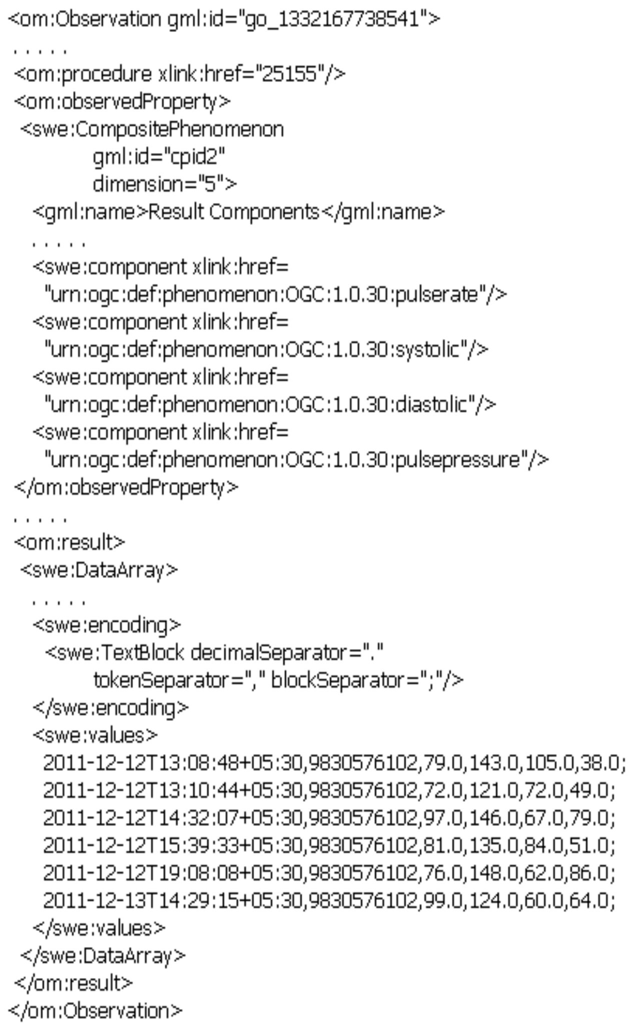
For Fat analyzer, its unique device identifier can be used as procedure (or sensor Id). The Phenomenon will be changed to the five parameters measured by it, and accordingly the Observed Property & Composite Phenomenon will change. As, the observation belongs to the patient, so the mapping of Feature Of Interest will remain same as BP monitor. The Result is mapped to the actual value for all the five parameters as reported by the Fat analyzer. These mappings enable the creation of XML templates.

On receiving the parameterized URL, proxy component picks an XML template meant for the request string and device type, puts the parameters in the placeholders and sends it to SOS via HTTP POST.

The response generated by SOS depends on the SOS operation. On success, RegisterSensor returns AssignedSensorId whereas InsertObservation returns observation id. On failure, an exception or error message is issued.

It is evident from the above discussion that the medical instruments (even though provided by the same vendor) do not follow any guidelines to support uniform interface across all its devices and the same is true for the data format. Hence, we needed the plugin layer to handle these differences.

The URL requests originated from Plugin Layer map to the SOS transaction operations (as the sensors only store the observation, they don't fetch it). The URL requests originated from Portal (as depicted in Figure 1) maps to the GetObservation. The snapshot in Figure 3 depicts a typical XML response from SOS.



```
<om:Observation gml:id="go_1332167738541">
 . . . . .
 <om:procedure xlink:href="25155"/>
 <om:observedProperty>
  <swe:CompositePhenomenon
          gml:id="cpid2"
          dimension="5">
   <gml:name>Result Components</gml:name>
   . . . . .
   <swe:component xlink:href=
    "urn:ogc:def:phenomenon:OGC:1.0.30:pulserate"/>
   <swe:component xlink:href=
    "urn:ogc:def:phenomenon:OGC:1.0.30:systolic"/>
   <swe:component xlink:href=
    "urn:ogc:def:phenomenon:OGC:1.0.30:diastolic"/>
   <swe:component xlink:href=
    "urn:ogc:def:phenomenon:OGC:1.0.30:pulsepressure"/>
 </om:observedProperty>
 . . . . .
 <om:result>
  <swe:DataArray>
   . . . . .
   <swe:encoding>
    <swe:TextBlock decimalSeparator="."
          tokenSeparator="," blockSeparator=";"/>
   </swe:encoding>
   <swe:values>
    2011-12-12T13:08:48+05:30,9830576102,79.0,143.0,105.0,38.0;
    2011-12-12T13:10:44+05:30,9830576102,72.0,121.0,72.0,49.0;
    2011-12-12T14:32:07+05:30,9830576102,97.0,146.0,67.0,79.0;
    2011-12-12T15:39:33+05:30,9830576102,81.0,135.0,84.0,51.0;
    2011-12-12T19:08:08+05:30,9830576102,76.0,148.0,62.0,86.0;
    2011-12-13T14:29:15+05:30,9830576102,99.0,124.0,60.0,64.0;
   </swe:values>
  </swe:DataArray>
 </om:result>
</om:Observation>
```

Figure 3. Response XML from SOS containing BP Data

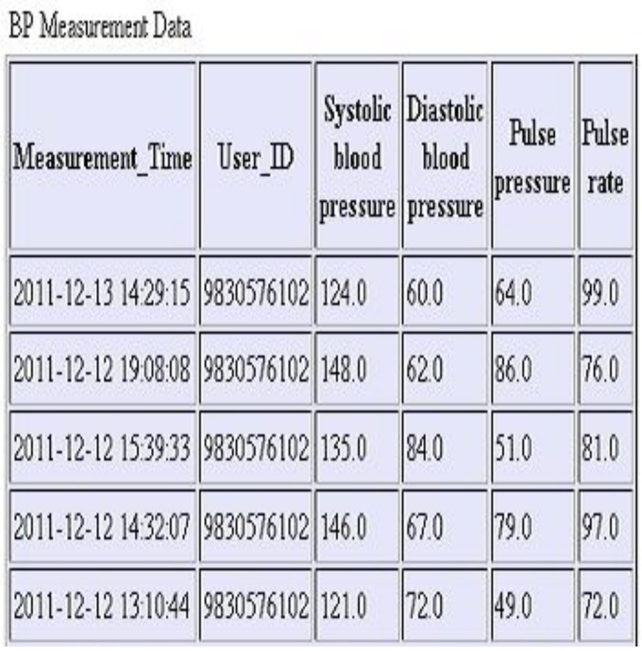
The Portal is the reporting tool that retrieves data from SOS using GetObservation operation. GetObservation provides querying capability based on the parameters like feature of interest (i.e., patient id), procedure (i.e., sensor id), offering, observed property, sampling time etc. To access the observations in SOS, a request is send to the proxy as parameterized URL. Proxy parses the URL and based on the request string, it turns generates the XML and send it to SOS. The response to this request is an XML with all the values meeting the query criteria. This XML is forwarded to the portal, where it gets rendered in tabular format.

Section IV describes the various medical devices we have worked with, the android code we had to develop for getting medical device readings via Bluetooth and the portal view of the data that has been kept in the central database using SOS.

IV. RESULTS

The Fat Analyzer and the BP Monitor belong to first category of devices, which communicate via vendor provided gateway and upload the medical sensor data via SOS using our UDP Adapter. Here, only the gateway needs to be configured for proper server IP and port for data posting. Once the data gets posted, the user can have a view of the medical readings in the portal. The snapshot in Figure 4 is depicting a typical portal view of Blood Pressure data.

However, for the second class of devices, where medical sensor data has to be captured via Bluetooth, we had to develop a mobile application to capture readings via Bluetooth and posting it to central database using SOS. The snapshot in Figure 5 depicts the landing page of our app, which waits for user input to take either ECG reading or Oximeter reading.



BP Measurement Data

Measurement_Time	User_ID	Systolic blood pressure	Diastolic blood pressure	Pulse pressure	Pulse rate
2011-12-13 14:29:15	9830576102	124.0	60.0	64.0	99.0
2011-12-12 19:08:08	9830576102	148.0	62.0	86.0	76.0
2011-12-12 15:39:33	9830576102	135.0	84.0	51.0	81.0
2011-12-12 14:32:07	9830576102	146.0	67.0	79.0	97.0
2011-12-12 13:10:44	9830576102	121.0	72.0	49.0	72.0

Figure 4. Portal View of BP Data

In the series of snapshots in Figure 8, we have depicted the steps one need to perform in our android app for taking the ECG data from the device, and getting it posted to the central database using SOS.

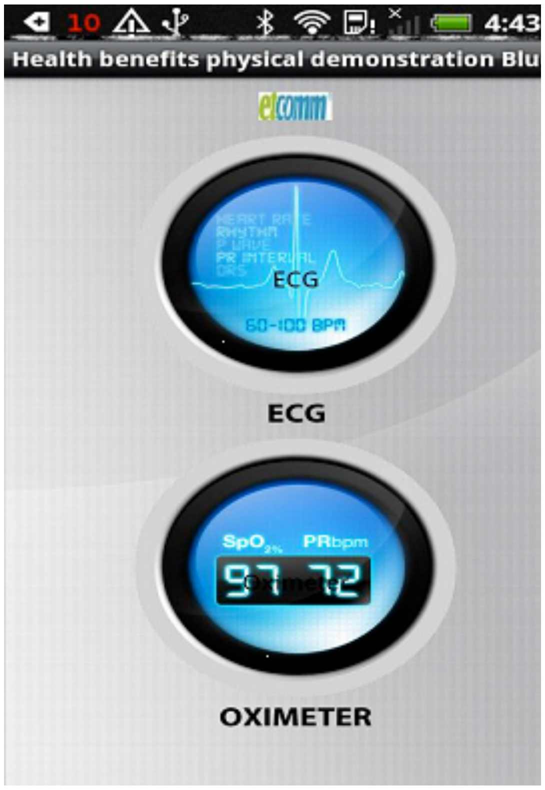


Figure 5. Landing Screen for the Android App

The second class of devices also need to have a local viewer application in the mobile as the user can check the readings and then push it to database once verified. The snapshot in Figure 6 depicts how we have implemented the ECG viewing scheme in our mobile application on Android. Once the data gets posted in database, Figure 7 and Figure 9 show the snapshot views from our Portal instance.

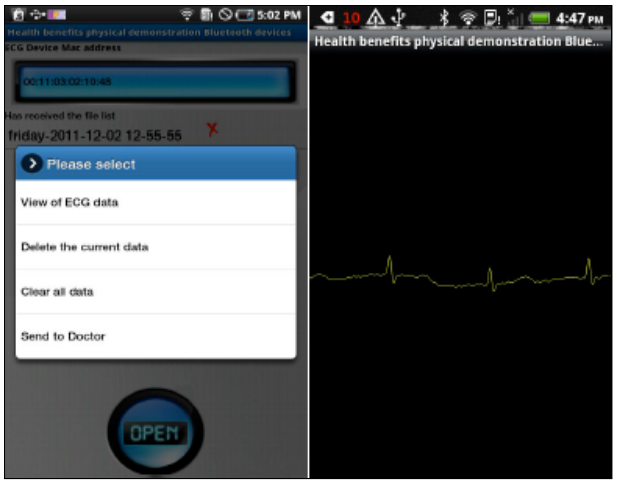
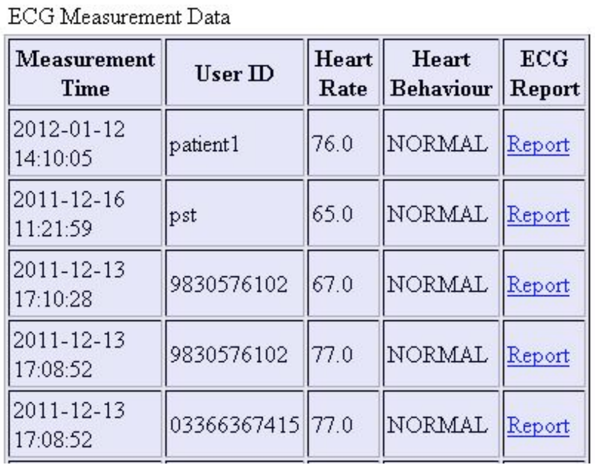

5:02 PM
Health benefits physical demonstration Bluetooth devices
ECG Device Mac address
00:11:03:02:10:48
Has received the file list
friday-2011-12-02 12-55-55
Please select
View of ECG data
Delete the current data
Clear all data
Send to Doctor
OPEN
10
4:47 PM
Health benefits physical demonstration Blue...

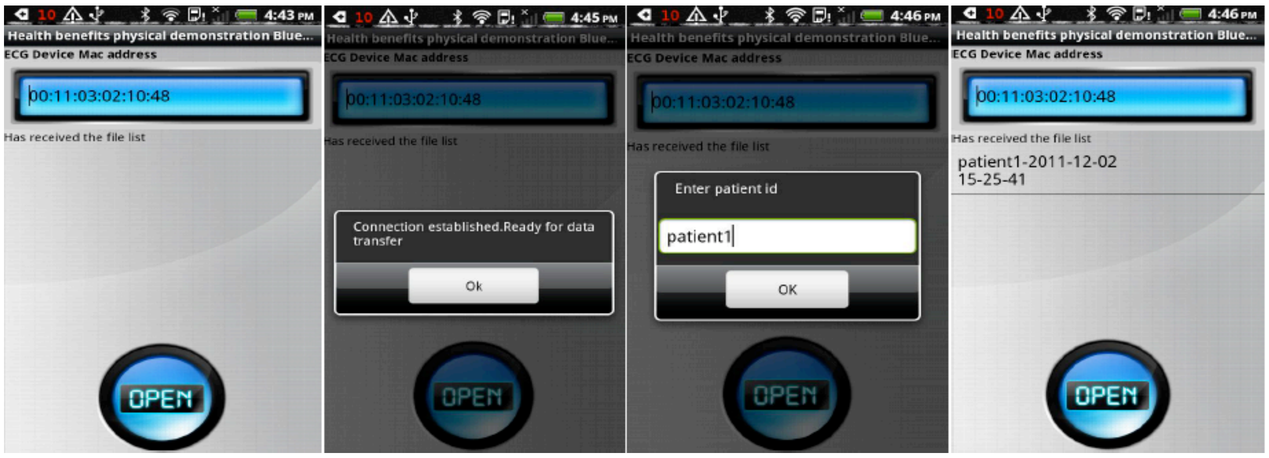
Figure 6. Local Viewing of ECG Data via Android App



ECG Measurement Data

Measurement Time	User ID	Heart Rate	Heart Behaviour	ECG Report
2012-01-12 14:10:05	patient1	76.0	NORMAL	Report
2011-12-16 11:21:59	pst	65.0	NORMAL	Report
2011-12-13 17:10:28	9830576102	67.0	NORMAL	Report
2011-12-13 17:08:52	9830576102	77.0	NORMAL	Report
2011-12-13 17:08:52	03366367415	77.0	NORMAL	Report

Figure 7. ECG Data Portal View


10
4:43 PM
Health benefits physical demonstration Blue...
ECG Device Mac address
00:11:03:02:10:48
Has received the file list
OPEN
10
4:45 PM
Health benefits physical demonstration Blue...
ECG Device Mac address
00:11:03:02:10:48
Has received the file list
Ok
OPEN
10
4:46 PM
Health benefits physical demonstration Blue...
ECG Device Mac address
00:11:03:02:10:48
Has received the file list
OK
OPEN
10
4:46 PM
Health benefits physical demonstration Blue...
ECG Device Mac address
00:11:03:02:10:48
Has received the file list
OPEN

Connection established.Ready for data transfer

Enter patient id

patient1

patient1-2011-12-02 15-25-41

Figure 8. Steps for taking ECG via Android App

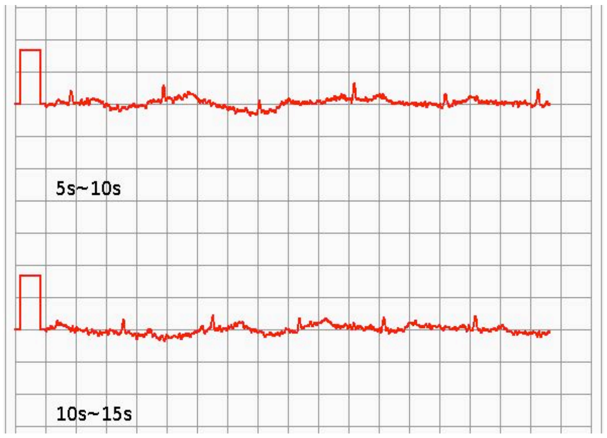


Figure 9. ECG Graph View from Portal

V. Conclusion and Future Work

Our proposed application demonstrates healthcare as a vertical service to IoT platform. It also demonstrates the use of medical instruments with heterogeneous interfaces as web-enabled sensors, which contribute to a common SOS-enabled repository. The SOS layer enables the seamless integration of sensors by providing them a common ground, i.e. xml interface and SOS schema. Following are a few challenges and shortcomings of the system where improvement can be made:

- There exists devices which are closed for interaction and upload data directly to vendor's website for a portal view. Some of these vendors provide APIs to access this data from portal. The adapter, required at Plugin layer to incorporate these medical devices, needs to poll and gets these data as sensor observation into the IoT platform.
- The Healthcare service can really be given as a SMART service once we can bring SMART Analytics on the healthcare data. Currently we have implemented only the portal view of the healthcare data, but analytics services need to be run over this data to give more meaningful services to various stakeholders. So, the platform needs to be adapted for more analytics services to run over it.
- Mapping for medical streaming data to SOS compliant xml is not attempted in this application and can be taken as a potential future work.
- REST interface is not available in SOS implementations. While developing our application, we experienced that instead of XML interfaces, REST interface will be more efficient.
- Communication between Plugin Layer & Proxy as well as Proxy & Portal can be standardized using HL7.
- Available open source SOS implementations do not allow removal / editing of erroneous/corrupt data from SOS repository.
- The privacy and security of such sensitive data has not been taken care yet. Here, anybody with a credential can view anybody's medical data. This needs to be taken care in future version of the platform.

Acknowledgment

The authors would like to sincerely thank Mr. Arpan Pal and Mr. Prateep Misra for reviewing the paper. The authors would also like to thank Mr. Debnarayan Kar, Mr. Kallol Ghosh, Ms. Priyanka Sinha, Mr. Prasun Bhattercherjee and Mr. Joydip Das, all members of TCS Innovation Lab under the theme of Ubiquitous Computing for helping in implementation of the whole scheme for successful deployment of the project.

References

[1] Botts M., Percivall G., Reed C., and Davidson J., "OGC Sensor Web Enablement: Overview and High Level Architecture," OGC 07-165, Version 3 Dec. 28, 2007, [retrieved: Apr., 2012], http://portal.opengeospatial.org/files/?artifact_id=25562

[2] Open Source SOS Software from 52North, [retrieved: Apr. , 2012], http://52north.org/communities/sensorweb/sos/

[3] Churcher G. E., Bilchev G., Foley J., Gedge R., and Mizutani T., "Experiences applying Sensor Web Enablement to a practical telecare application," Wireless Pervasive Computing, 2008. ISWPC 2008. 3rd International Symposium on, pp. 138-142, 7-9 May 2008

[4] Figueredo M. V. M. and Dias J. S., "Mobile Telemedicine System for Home Care and Patient Monitoring," Engineering in Medicine and Biology Society, 2004. IEMBS '04. 26th Annual International Conference of the IEEE, vol. 2, pp. 3387- 3390, 1-5 Sept. 2004

[5] Sufi F., Fang Q., Mahmoud S. S., and Cosic I., "A Mobile Phone Based Intelligent Telemonitoring Platform," Medical Devices and Biosensors, 2006. 3rd IEEE/EMBS International Summer School on, pp. 101-104, 4-6 Sept. 2006

[6] Jirka S. and Bröring A., "Applying OGC Sensor Web Enablement to risk monitoring and disaster management," GSDI 11 World Conference, 2009, [retrieved: Apr., 2012] http://www.gsdi.org/gsdiconf/gsdi11/papers/pdf/96.pdf

[7] Varshney U., "Pervasive healthcare and wireless health monitoring," Mobile Networks and Applications, vol. 12, no. 2-3 (March 2007), pp. 113-127, DOI=10.1007/s11036-007-0017-1, http://dx.doi.org/10.1007/s11036-007-0017-1

[8] Sashima A., Inoue Y., Ikeda T., Yamashita T., Ohta M., and Kurumatani K., "Design and Implementation of Wireless Mobile Sensing Platform," SenSys'07, Nov. 6–9, 2007, Sydney, Australia, [retrieved: Apr., 2012] http://lecs.cs.ucla.edu/%7Enithya/SensysPSWorkshop/camera_ready/sashima.pdf

[9] Varshney U., "Pervasive Healthcare: Applications Challenges And Wireless Solutions," Communications of the Association for Information Systems, vol. 16(2005), art. 3, [retrieved: Apr., 2012] http://aisel.aisnet.org/cgi/viewcontent.cgi?article=3018&context=cais

[10] Gupta V., Poursohi A., and Udupi P., "Sensor.Network: An open data exchange for the web of things," Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on, pp. 753-755, March 29 2010 - April 2 2010

[11] McFerren G., Hohls D., Fleming G., and Sutton T., "Evaluating Sensor Observation Service implementations," Geoscience and Remote Sensing Symposium,2009 IEEE International, IGARSS 2009, vol. 5, pp. 363 –366, 12-17 July 2009

[12] Arthur N. and Priest M., "Sensor Observation Service," OGC 06-009r6, Oct. 2007, OpenGIS Implementation Standard, [retrieved: Apr., 2012], https://portal.opengeospatial.org/files/?artifact_id=26667

[13] Cox S., "Observations and Measurements – Part 1 - Observation schema," Version 1.0(2007a), OGC 07-022r1, Dec. 2007, OpenGIS Implementation Standard, [retrieved: Apr., 2012], https://portal.opengeospatial.org/files/?artifact_id=22466

[14] Cox S., "Observations and Measurements – Part 2 - Sampling Features," Version 1.0(2007b), OGC 07-002r3, Dec. 2007, OpenGIS

Implementation Standard, [retrieved: Apr., 2012], https://portal.opengeospatial.org/files/?artifact_id=22467

[15] Simonis I., "OGC Sensor Web Enablement Architecture," OGC 06-021r4, Aug. 2008, Version: 0.4.0, OpenGIS Best Practice, [retrieved: Apr., 2012], https://portal.opengeospatial.org/files/?artifact_id=29405

[16] Hashmi N., Myung D., Gaynor M., and Moulton S., "A sensor-based, web service-enabled, emergency medical response system," In Proceedings of the 2005 workshop on End-to-end, sense-and-respond systems, applications and services (EESR '05), USENIX Association, Berkeley, CA, USA, pp. 25-29