# Object-oriented Design for Wireless Sensor Network assisted Global Patient Care Monitoring System

Rajeev Ranjan
Indian Institute of Information Technology, Allahabad

Shirshu Varma
Indian Institute of Information Technology, Allahabad

## ABSTRACT
Health care monitoring system can be characterized as the combination of communication technology and biometric sensor nodes. Patients' care monitoring is a part of telemedicine monitoring, in which HTTP (Hypertext Transfer protocol) is used to access information from devices which have the internet connection. In this paper, we present a design of an object oriented database system for server and client side to monitor patients' applications data. It provides a novel approach of having ipv6 low power wireless personal area network (6lowpan) and internet based connectivity between patients and doctors. In this paper patients' data is extracted from the sensor network and is given to a web based system, however the focus of the doctor is still on the senor networks only.

## Keywords
Object Oriented Database, Wireless Sensor Networks, Health Monitoring System

## 1. INTRODUCTION
The high speeds of data transfer in wireless communication media have unlocked the doors for the development many applications like health care, military, security and disaster management. The wireless data transmit rates combined with web enabled computer system, is opening new fields on Internet services. Under the Internet Engineering Task Force (IETF) working group of 6lowpan; inter connectivity between low powered IEEE802.15.4 devices and wired IPv6 domain work is going on [15].

Wireless sensor networks are made up of large number of sensor nodes. The sensor nodes are densely deployed either inside a phenomenon or very close to it. The location of nodes need not be pre-determined or engineered. Recently developed nodes can be directly connected to the internet. This gives us the advantage to access and examine patients from a remote location. The examined data needs to be stored for later analysis. Object-oriented databases for data storage have significant benefits to the applications that exercise complex object models, high concurrency requirements and large data sets.

Current health care systems is optimized and structured for responding to crisis and managing illness and facing challenges: rapid growth in population and rising spending and awareness on health care. If physiological information about patients can be accessed and examined remotely by a doctor and it also allows a doctor to have better understanding about the patient's current condition. Most of the work done in this field gives attention to hospital management only. We always tend to forget that patient data is also very important.

Our paper presents a novel design to monitor and store patients' data for effective remote health monitoring system. It describes a global health monitoring scheme with the help of object- oriented database and java. Our design tries to make a sincere effort for monitoring patients' data, which will be useful for doctors as well as the patients.

**Paper Organization:** Section 2 describes some of the related work done in this area. Section 3 describes the importance of object-oriented database management system. Section 4 describes the design of our system. Section 5 describes our implementation and results. Finally, we conclude in section 6.

At the end, Appendix A shows the different classes and their mapping.

## 2. SOME PROJECTS AND RELATED WORK
All Many health care projects are under progress in different institutions to offer more assistance to the people. Recent developments have focused on wearable health devices [11]. Health projects [1-3] are under progress to build a system for collecting major body signals and manipulating those signals in remote health care institutes and organizations. The Terva monitoring system [5] have been designed for data related to health like blood pressure, temperature, weight, and so on. The data has been gathered four times a day and saved in the form of a time-of-day (TOD) matrix and examined later. This system becomes practical to be used in a stationary manner. Wireless Wellness Monitor [1] is a response-based self monitoring system to manage obesity. It has been developed using Bluetooth and Jini network to sustain Java dynamic networking. The system comprises computing devices, a base station, mobile terminals and databases connected to the internet**.** The computing devices collect data and store in the server. The data can be retrieved wirelessly and collected through databases from mobile terminals. MobiHealth project [4-6] can monitor vital health signals with the help of tiny biomedical sensors and broadcast them to health care professionals with the help of highly powerful wireless system**.** GPRS and UMTS have been used for broadcasting signal on the fly and Body Area Network (BAN) has been applied in signal monitoring. Wireless Wearable Body Area Network [7] has been developed which consists of stationary sensors which is only communicate with the central control unit. Wellness Assistant (WA) uses inexpensive handheld devices for example PDAs, cell phones and wrist watches comprise of short range wireless abilities to monitor obesity, diabetes etc. [10] explains various pervasive health care applications and requirements. Assisted cognition systems use ubiquitous computing and artificial intelligence technology to restore some of the memory and problem-solving abilities that have been lost by an Alzheimer patient [8].

The database approach is generally used for WSNs. It offers an easy-to-use interface. It allows the user to query to the sensors to take out the data of interest. Madden et al. [12] have implemented an inquisitional query processing system

for sensor networks. This approach provides a interface for the data collection and in-network aggregation. Yao and Gehrke proposed a layered architecture consisting of application, routing and query proxy layers.

## 3. OBJECT ORIENTED DATABASE MANAGEMENT SYSTEM

The main goal of the development of the object-oriented databases was to overcome the borders of RDBMS and rise of Internet and the Web. An object-oriented database management system (OODBMS) is a database management system (DBMS) that supports the modeling as well as creation of data as objects.

With the help of this database technology, user can combine object-oriented programming with existing database technology, which offers an integrated application development system. OOPS programming has following main characteristics:

- *Data encapsulation:* Data encapsulation is a process to manage data and methods by hiding the implementation status of the objects. The encapsulated objects are assessed only by the methods rather than their internal states.
- *Inheritance:* Inheritance is a way to categorize and reuse code by making objects on the basis of earlier defined objects.
- *Object identity:* Object identity permits objects of the database to be not dependent of each other. It defines the characteristic of an object that identifies them from other objects.
- *Polymorphism:* Dynamic binding and polymorphism and allow sharing the specification of the operation or action with other objects which is defined for one object. This allows users or programmers to create objects to give solutions without writing codes which is precise to each object.

Data definition and manipulation language (DDML) is the important language to Object Oriented DBMS. With the use of DDML the persistent data are created, updated, retrieved or deleted. An OODBMS requires a computational against a relational language to avoid impedance mismatch. The maintenance and query of a database is made with the help of DDMLs.

The OODBMS has lots of advantages. Object-oriented provides more natural way of thinking. The defined actions of such systems are free from a particular database**.** The data types defined in object-oriented databases are extendable to support complex data, for example, multi-media operations. The reusability, reliability and stability are the other key advantages of OODBMS. It allows defining user's own methods to represent or manipulate data. One of the key benefits of OODBMS is that these types of databases have expanded into fields not known by the RDBMS. Object-oriented databases are used in a few new fields such that medicine, multimedia, and high-energy physics. The benefits to the object-oriented databases to problems regarding information management that is showed by the requirement to manage:

- Different data types,
- Relationships between different type of objects, and
- Objects with complex actions.

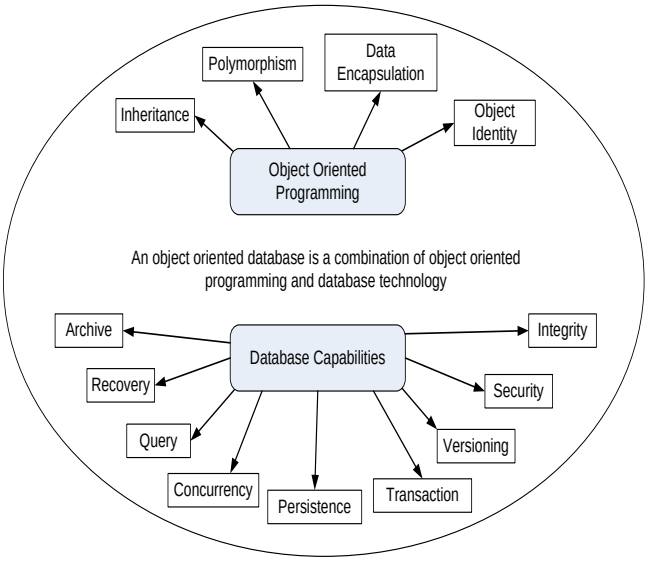


**Figure 1: Features of ODBMS**

## 4. SYSTEM DESIGN

The paper aims to design and develop a new concept for healthcare parameter analysis. We use the 6lowpan based Internet Protocol (IP) enabled sensor motes. These sensor motes are able to extract data from biomedical sensors, used to monitor health condition. The biomedical sensor network system incorporates heterogeneous devices, some of them wearable on the patient and some of them are placed inside the living room. Data is collected, pre-processed, aggregated, stored, monitored and worked upon using a range of sensors and devices. The 6lowpan enabled nodes have the wireless link to the internet supported server or gateway through router. Physicians or non-technical staff can directly monitor the health and other biomedical parameter with the help of internet. There is a single mote associated with every patient. All the motes have unique MAC Id to identify the mote and hence the patient. The mote can handle almost 16 different data related to the patient. The data is being stored in database and the required information can be retrieved in the care monitoring system. In order to simplify the system we have divided the whole data into category i.e. offline data and online data. The online data are those types of data which can be monitored continuously with sensor motes, for example, temperature, pressure, heartbeat etc and offline data refers to those data which can be measured as different test results such as blood test, urine test etc. The offline data is stored into a database with time and date to evaluate later. We combine both online and offline data to make a final report to the medical staff.

Figure 2 shows the pictorial representation of the online biomedical application. Bio-medical sensors and accumulators are attached to the object-oriented database and Internet through systems. The offline data are stored in the object-oriented database while online data is directly available on the internet. The medical and paramedical staff can check the data on system attached to Internet and databases. Figure 3 is the logical representation of the system.

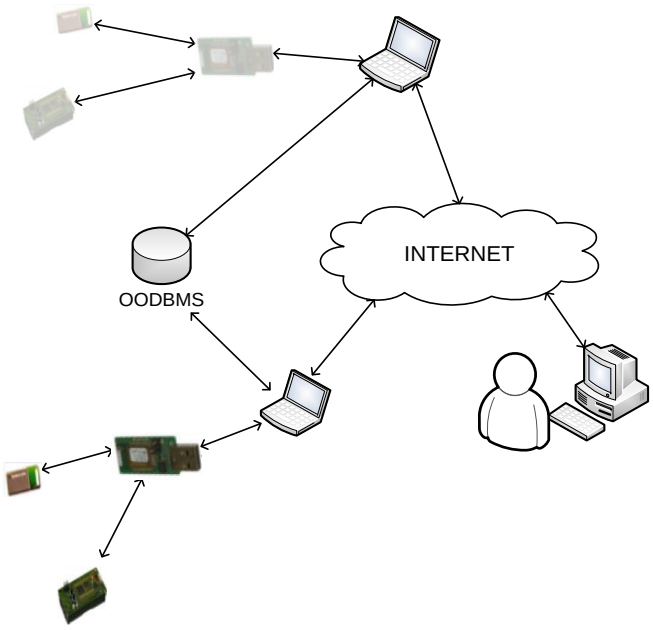


**Figure 2: System Design**

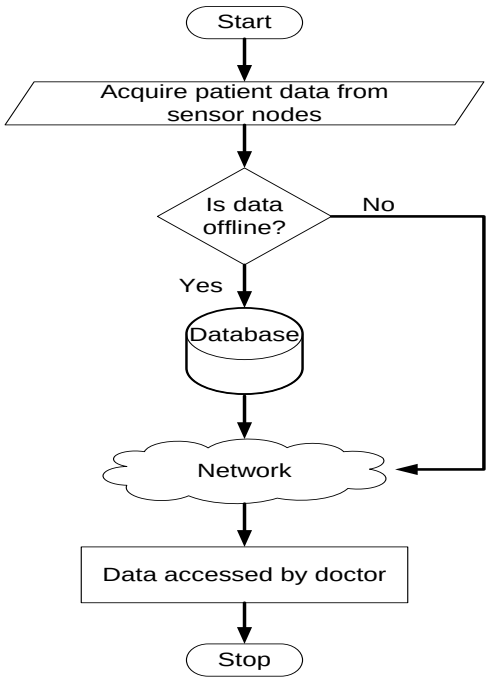


**Figure 3: Flow chart**

## 4.1 Database Design

The entire monitoring system at host end mainly divided into two parts: the database management server and the system management server. System management server provides interface to the users. Database server and databases combines to form the database management server. This server interacts between a range of modules and databases. The database stores data for further analysis. Here, in this paper, we use the top-down approach construct the system. In the case of health monitoring system following are the different steps of top-down approaches [17]:

I. **Analyze the Environment:** A hospital provides facilities like consolation, diagnosis, treatment facilities, immunization etc. The primary step during designing a database to analyze and identify with the user's environment and requirement.

II. **Develop Object-Oriented Data Model:** These days object oriented approach becomes popular in many fields. Nowadays it is very common practice to exercise object-oriented ideas in data modeling. We, in this paper, employ object oriented concepts to data modeling.

Data model is a logical union of the real world objects and entities. It also defines restrictions and the relationships among these entities. A database language is a tangible syntax for a data model. A database system employs a data model**.** An object-oriented data model comprises of the following object-oriented theory [18]:

**a)** *Object and object identifier:* Any entity which belongs to real world is modeled as an object. All objects should be unique. A unique Identity is given to each of the object. This Identity is the property of an object and it differentiates it from all other object. In object-oriented databases, this identity is applied by an object identifier (OID). OID is used by the system to recognize every object uniquely and to generate and manage inter-object references. Main properties of OID are it is immutable and is used only once.

**b)** *Attributes and methods:* Every object has some state and a behavior. These state and behavior, which are encapsulated in an object, are accessed from outside only through explicit message passing.

**c)** *Class:* Class is a set of the entities which have same set of attributes and methods. An object must be a member of a class as an instance of that class. A class is like to an abstract data type. A class may also be primitive, e.g. Boolean, integer, string.

**d)** *Class hierarchy and inheritance:* Inheritances derive a new class which is called subclass from an existing class, called superclass. The subclass inherits all the attributes and methods of the superclass, also having additional attributes and methods.

# 5. IMPLEMENTATION

We divide the application into two parts: the server side implementation and client side implementation. JSP and servlets are used have been used for user interface. Tomcat apache 6.0 is used to make the profile server. We have used eyeDB for the database and SMTP protocol is used to send the prescription of physician to the concerned caretaker. Table 1 show the different types of sensors used and data and parameters associated to them for applications. Netbeans is used for the interface between the sensor motes and the system.

**Table 1: Sensors and their parameters**

| SENSORS | PARAMETER |
|---|---|
| **Electrocardiograph** | **Heart Rate** |
| | **ECG(Signals)** |
| **Pulse Oximeter** | **Oxygen Saturation in Blood** |
| **Blood Pressure Meter** | **Systemic arterial pressure** |
| | **Diastolic arterial pressure** |
| | **Average arterial pressure** |
| **Glucometer** | **Glucose** |
| **Scale** | **Weight** |
| **Passive InfraR (PIR)** | **Presence** |
| **InfraRed** | **Pass through** |
| **Door opening** | **Doors or windows opening / closing** |

## 5.1 Client Side Implementation

Http browser is used for the client side implementation. In the implementation of client side, web pages are generated. We provide database queries to perform specific task like INSERT, UPDATE and DELETE. Figure 4 shows different classes and their attributes which is used for the application.
Figure 5 represents the graphical view of the classes and their mapping. The classes are related with a relation name with different classes. The square boxes represent the classes .The **dark solid line** shows the EXTENDS behavior of a class towards the class where **arrow** indicates. The **thin solid line** represents a relationship between classes. The both side **single arrow** shows the 1:1 relation whereas one side single arrow and other side **double arrow** shows the 1: M relation. The both side double arrow sign show the M: N relation between classes. E.g. the class **patient** extends both **patientoff** and **patienton** classes. The class **patient** has a 1: N relation with **caretaker** class and M: N relation with **sensor** class. Every relation has given a relation name, as shown.

## 5.2 Server Side Implementation

We used Java programming with HTML pages for dynamic pages. Due to this the extension has been changed to .jsp. The implementation of the work mainly shows the data flow in a particular module. The web pages consist of homepage, login page, select hospital, select department and patient, parameter display and prescription page. The address which has shoed the home page is **http://172.17.15.27:8888/patient/index.jsp** where 172.17.15.27 is the IP address of the server, 8888 is the port, patient is the folder and login.jsp is the home page. Servlets are used to prepare JSP pages and send them to HTTP browser.

We start the application with doctor homepage. A doctor enters his username and password to the space provided on the homepage. If the username and password is correct, the doctor login to the website and enters to his member area otherwise the url is redirected to the homepage with a message "Enter correct Username/Password".

After successful login to the site, he can select a hospital from a number of hospital list listed. After selection to the hospital, he will able to select department associated with the hospital. After that he can select a patient and can view his offline and online data. On the basis of the data, he can either write a prescription or instruct directly to the caretaker for necessary step or medicine. Figure 6 shows the flow chart of the application.

# 6. EXPERIMENTAL RESULTS

The different tests like running time and memory usage were carried out on a Dell XPS 14 (L401X) notebook PC, equipped with 2.93 Ghz Intel i5-480M Processor CPU, 4GB RAM and Microsoft Windows XP O.S. After the initial experiments, we have evaluated the results attained from our smart algorithm. Result Arrival Rate (RAR) is used to present the time intervals between the two consecutive pages. RAR is measured in terms of bits per second (bps) or pages per second (pps). A reasonable RAR can improve the system resource utilization without degrading the service quality provided for users.

A number of queries were tested to get the response time of different result file sizes. Table 8 shows the Query file size versus Database response time.

Server monitors can be configured to monitor the resource utilization such as, CPU and memory usage of the web servers. The Table below indicates the resource utilization during the test execution.

**Table 2: Query files size versus Database response time**

| **Query File Size (in kilo Byte)** | **Response Time (in millisecond)** | |
|---|---|---|
| | SQL time to receive the first byte | SQL time to receive the Last byte |
| 5 | 4595 | 5100 |
| 10 | 4596 | 5105 |
| 15 | 4690 | 5258 |
| 20 | 4700 | 5300 |
| 25 | 4800 | 5352 |
| 30 | 4800 | 5322 |

**Table 3: Server CPU vs Elapsed time**

| **Elapsed Time (sec)** | **CPU Usages (%)** |
|---|---|
| 10 | 58 |
| 20 | 67 |
| 30 | 43 |
| 40 | 25 |
| 50 | 22 |
| 60 | 17 |

**Table 4: Server Memory vs Elapsed time**

| **Elapsed Time (sec)** | **Memory Usages (%)** |
|---|---|
| 10 | 1 |
| 20 | 1 |
| 30 | 1 |
| 40 | 1.25 |
| 50 | 1.5 |
| 60 | 1.25 |

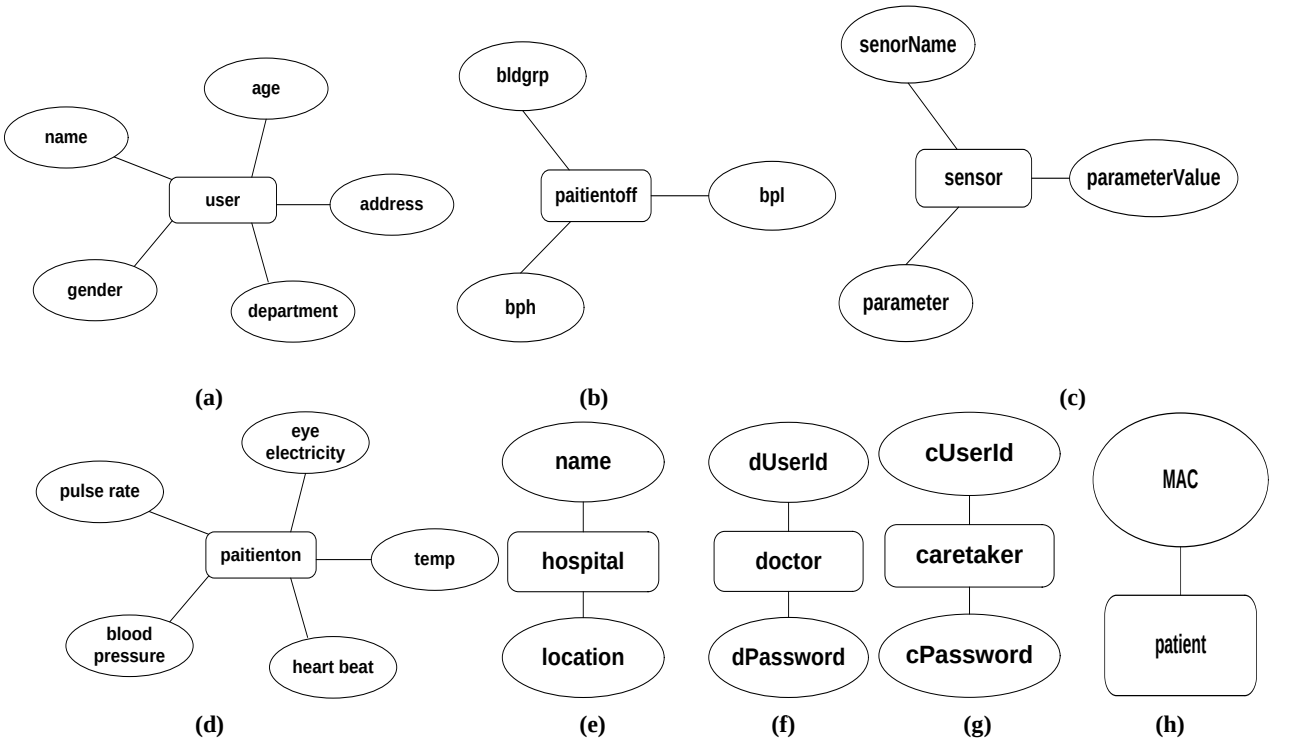


**Figure 4: Different Class and their Attributes**

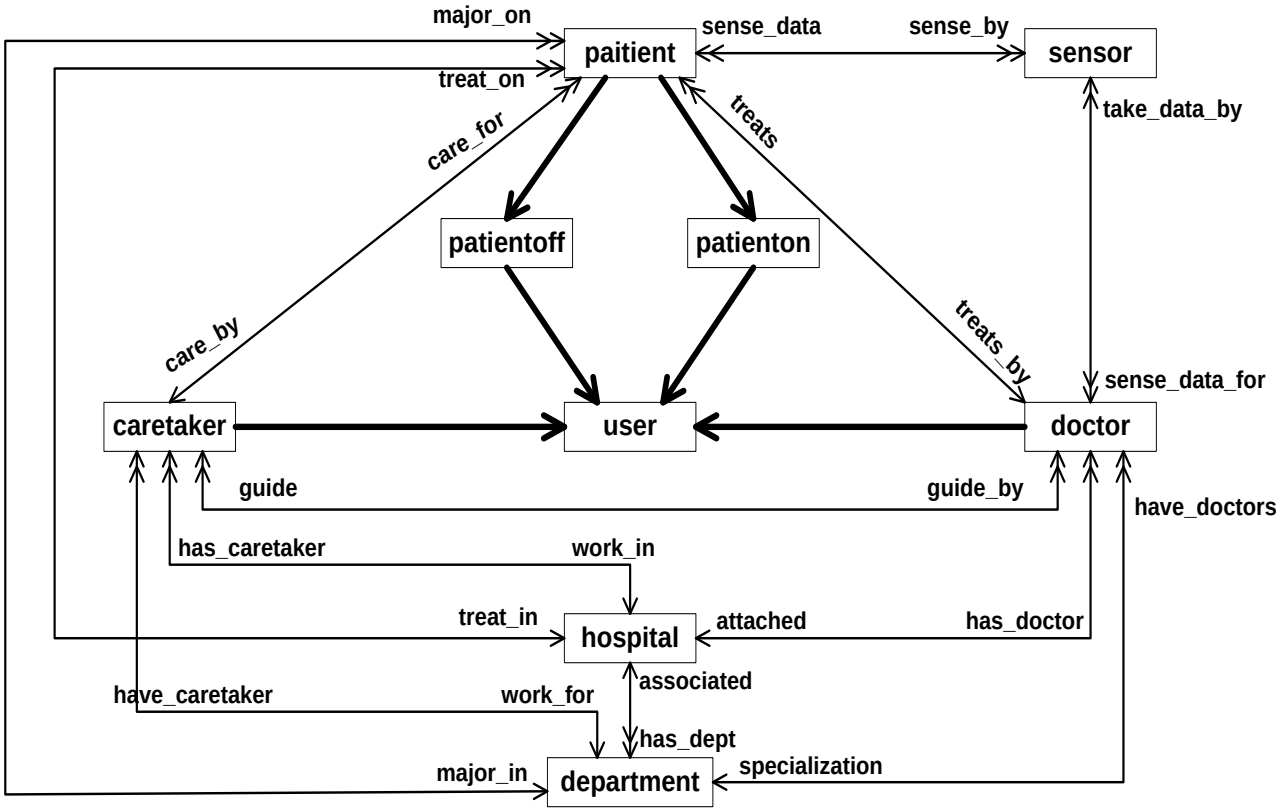


**Figure 5: Classes and Their Relations**

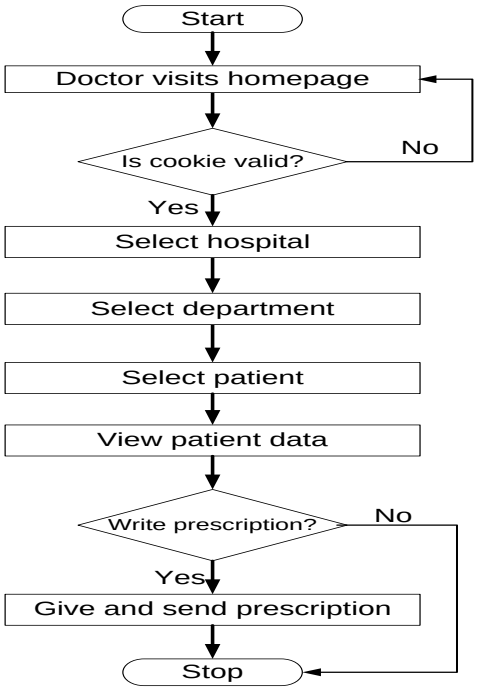


**Figure 6: Flow Chart of the Application**

## 7. CONCLUSION

Health care monitoring system can be defined as the combination of communication technology and biometric sensor nodes. Patients' care monitoring is a part of tele-medicine monitoring, in which HTTP (Hypertext Transfer protocol) is used to access information from devices which have the internet connection. In this paper, we present a design of an object oriented database system for server and client side to monitor patients' applications data. It provides a novel approach of having ipv6 low power wireless personal area network (6lowpan) and internet based connectivity between patients and doctors. In this paper patients'' data is extracted from the sensor network and is given to a web based system, however the focus of the doctor is still on the senor networks only.

## 8. REFERENCES

[1] D. M. Fraser. Biosensors: Making sense of them. Medical Device Technology, 5(8):38-41, Feb 1994.

[2] Janani, K.; Dhulipala, V.R.S.; Chandrasekaran, R.M.; , "A WSN Based Framework for Human Health Monitoring," Devices and Communications (ICDeCom), 2011 International Conference on , vol., no., pp.1-5, 24-25 Feb. 2011.

[3] D. Konstantas, A. van Halteren, R. Bults, K. Wac, I. Widya, N. Dokovsky, G. Koprinkov, V. Jones, and R. Herzog. Mobile patient monitoring: the mobihealth system. Stud Health Technol Inform, 103:307-314, 2004.

[4] Korhonen, R. Lappalainen, T. Tuomisto, T. Koobi, V. Pentikainen, M. Tuomisto, and V. Turjanmaa. Terva: wellness monitoring system. Engineering in Medicine and Biology Society, 20th Annual International Conference of the IEEE, 4(29):1988-1991, Oct 1998.

[5] A. Milenkovic, C. Otto, and E. Jovanov. Wireless sensor networks for personal health monitoring: Issues and an implementation. Computer Communications (Special issue: Wireless Sensor Networks: Performance, Reliability, Security, and Beyond), Elsevier, 29(13-14):2521-2533, Oct 2006.

[6] J. Parkka, M. van Gils, T. Tuomisto, R. Lappalainen, and I. Korhonen. Wireless wellness monitor for personal weight management. Information Technology Applications in Biomedicine, IEEE EMBS International Conference, pages 83-88, Nov 2000.

[7] N. Saranummi, I. Korhonen, M. van Gils, and S. Kivisaari. Barriers limiting the diffusion of ict for proactive and pervasive health care. Proceedings of the IX MEDICON, Pula, Croatia, 4(29):1988-1991, Oct 2001.

[8] D. F. Henry Kautz, Oren Etzioni and D. Weld. Foundations of assisted cognition systems. Technical report, University ofWashington, Computer Science Department, Technical Report, 2003.

[9] A. van Halteren, D. Konstantas, R. Bults, K. Wac, N. Dokovsky, G. Koprinkov, V. Jones, and I. Widya. Mobihealth: ambulant patient monitoring over next generation public wireless networks. Stud Health Technol Inform, 106:107-122, 2004.

[10] U. Varshney. Pervasive healthcare and wireless health monitoring. Journal on Mobile Networks and Applications (Special Issue on Pervasive Healthcare), Springer, 12(2-3):111-228, June 2007.

[11] Wolf L. and Saadaoui S., "Architecture Concept of a Wireless Body Area Sensor Network for Health Monitoring of Elderly People," in Proceedings of Consumer Communications and Networking Conference 4th IEEE, pp. 722-726, 2007.

[12] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: an acquisitional query processing system for sensor networks. ACM Transactions on Database Systems (TODS), 30(1):122–173, 2005.

[13] Y. Yao and J. E. Gehrke. Query processing in sensor networks. In Proceedings of the International Conference on Innovative Data Systems Research (CIDR), January 2003.

[14] Jibing Gong; Shilong Lu; Rui Wang; Li Cui; , "PDhms: Pulse Diagnosis via Wearable Healthcare Sensor Network," Communications (ICC), 2011 IEEE International Conference on , vol., no., pp.1-5, 5-9 June 2011.

[15] N. Kushal Nagar, G. Montenegro, C. Schumacher, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks" RFC 4944, September 2007, pp-1-30

[16] P. Bonnet , J. Gehrke and P. Seshadri , "Towards Sensor Database Systems,"Proc. 2nd Int'l Conf. Mobile Data Management (MDM 01), 2001, pp 314-810.

[17] Rastogi, A.; Kumar, A.; Varma, S.; , "Database design for sensor network based global patient care monitoring system," Wireless Communication and Sensor Networks (WCSN), 2010 Sixth International Conference on , vol., no., pp.1-5, 15-19 Dec. 2010

[18] http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter8/node3.html

## APPANDIX A: Different Classes and mapping

This shows a way of mapping part of the PATIENT database. Entity types are mapped into Object Definition Language (ODL) classes.

```
class  sensor (
            extent Sensor
            key     sensorId)
           {
            attribute      int              sensorId ;
            attribute      string
        sensorName ;
            attribute      string           parameter
;
            attribute      string
        parameter_value;
            relationship   set<patient>
        sense_data inverse patient :: sense_by;
            relationship   set<doctor>
        sense_data_for  inverse doctor:: take_data_by;
            string get_parameter(string MAC);
            void   set_parameter(string MAC);
           };
```

```
class  department  (
            extent Department
             key     departmentId )
           {
            attribute      int
        departmentId ;
            attribute      string
        departmentName;
            relationship   set<patient>          treats_on
inverse patient :: treats_in;
            relationship   doctor
        has_doctor inverse doctor :: work_for;
            relationship   set<department>       has_dept
inverse department ::
associated;
             relationship  set<caretaker>    has_
caretaker inverse hospital ::  work_with  ;
           };
```

```
class  user (
            extent users
            key     userId)
           {
            attribute int  patientId ;
            attribute struct uname{string fname, string
lname} name;
            attribute string gender ;
            attribute string department;
            attribute struct Address{short no, string
aptNo, string city, string state, short zip} address;
           }:
```

```
class  patienton extends user (
            extent Patienton
            key     patientonId )
           {
            attribute int  patientonId ;
            attribute string temp;
            attribute string heartbeat;
            attribute string bloodpressure;
            attribute string pulserate;
            attribute string eyeelectricity;
            attribute date datei;
           };
```

```
class  patientoff extends user
          (
            extent Patientoff
            key     patientoffId )
           {
            attribute int  patientoffId ;
            attribute string bldgrp;
            attribute string bph;
            attribute string bpl;
            attribute date dateo;
           };
```

```
class  patient extends patienton ,patientoff(
                extent Patient
                 key     patientId )
               {
                attribute      int                    patientId;
                attribute      string                 MAC;
                relationship   doctor                 treats_by
inverse doctor :: treats ;
                relationship   hospital               treats_in
inverse hospital :: treat_on;
                relationship   department             major_in
inverse department ::
major_on;
                relationship    caretaker    care_by inverse
caretaker :: care_for;
                relationship    set< sensor>
            of_parameter  inverse sensor:: has_parameter;
                string get_sense_data();
                void set_priscription(string dUserId);
               };
```

```
class  doctor extends user(
                extent Doctor
                 key     doctorId )
               {
                attribute      int                    doctorId ;
                attribute      string                 dUserId;
                attribute      string
            dPassword;
                relationship  set<patient>            treats
inverse patient :: treats_by ;
                relationship   hospital               work_in
inverse hospital :: has_doctor;
                relationship   department             work_for
inverse department ::
have_doctors;
                relationship    set< caretaker>       guide
inverse caretaker :: guide_by;
               relationship    set< sensor>
            check_data inverse sensor:: data_by;
                void set_sense_data(string MAC);
                string get_priscription(string MAC);
               };
```

```
class  caretaker extends user(
                extent Caretaker
                 key     caretakerId )
               {
                attribute      int
            caretakerId ;
                attribute      string                 cUserId;
                attribute      string
            cPassword;
                relationship    set<patient>          care_for
inverse patient :: care_by;
                relationship     doctor               guide_by
inverse doctor :: guide;
                relationship   hospital               work_in
inverse hospital :: has_doctor;
                relationship   department             work_for
inverse department ::
have_doctors;

                void set_guide(string dUserId);
                string get_guide(string MAC);

               };
```