DESIGN OF A PATIENT MONITORING SYSTEM USING 3D ACCELEROMETER SENSORS

by

DEVI SHRAVANTHI KALLEM

B.E, OSMANIA UNIVERSITY, INDIA, 2007

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

DEPARTMENT OF COMPUTING AND INFORMATION SCIENCES
COLLEGE OF ENGINEERING

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2010

Approved by:

Major Professor
Dr. Gurdip Singh

Abstract

The Patient Monitoring System is a wireless sensor network application used for dynamically tracking a patient’s physical activity using 3D Accelerometer Sensors in the Sun Small Programmable Object Technology (SPOT) platform. The system is able to detect different postures of a person and recognize high-level actions performed by a patient by monitoring different pattern of postures. This activity can be monitored remotely from a nurse station or a handheld device. The monitoring system can be used for alerting the nurse station in a hospital, if a patient performs some abnormal action.

In the proposed system, the Sun SPOTs are affixed on a person's chest, thigh, leg and arm. The application determines the posture of a person by sensing the acceleration and tilt values of the SPOT in the direction of X, Y and Z axis. Based on these values the application can determine the postures of a person such as Lying Down, Sitting, Standing, Walking, Bending, and Arm Moving. We provide user mechanisms to define high level actions such as “attempting to get up from Lying Down position”, in terms of patterns of lower-level posture sequences. The system detects these patterns from the posture sequences reported by the Sun SPOTs, and reports them at desired locations.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my Major Professor Dr. Gurdip Singh for his constant help, encouragement and guidance throughout the project.

I would also like to thank Dr. Torben Amtoft and Dr. Mitchell Neilsen for serving in my committee and for their valuable cooperation during the project.

Finally, I wish to thank my family and friends for all their support and encouragement.

CHAPTER 1 - INTRODUCTION

1.1 Wireless Sensor Networks

"A **Wireless Sensor Network** (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, pressure, light, motion or pollutants, at different locations."[**1**]

1.1.1 Wireless Sensor Networks in Healthcare

Recent advances in networking technologies and wireless communication have opened new opportunities in managing healthcare systems, and are being used in conjunction with existing medical technologies in many healthcare applications. The need for applications allowing nurses to work remotely is being observed. A wearable health monitoring system is one such application which will allow monitoring an individual closely, and obtain their physical activity details. These details can help in evaluating a patient's physical activity and generate feedback on medication. It can also help in keeping track of any mandatory physical activities required to be done by the patients. These evaluations and feedback can help in maintaining a better health status of the patient and providing improved health care.

1.1.2 SUN SPOT Wireless Sensors

A Sun SPOT (Sun Small Programmable Object Technology) is a wireless sensor network mote developed by Sun Microsystems. It has a 3D accelerometer sensor, a light sensor and a temperature sensor on-board. The device is built upon the IEEE 802.15.4. Standard, and has a platform that runs JAVA with no operating system. It uses the Squawk Java Virtual Machine. [**10**]

1.1.2.1 3D Accelerometer Sensor

A 3D Accelerometer is a three-axis linear accelerometer that can detect the motion of the Sun SPOT along X, Y and Z axis. It also measures the orientation of the SPOT with respect to gravity.

The Z-axis is perpendicular to the Sun SPOT boards. The X-axis is parallel to the row of LEDs on the sensor board. The Y-axis is parallel to the long edge of the sensor board.

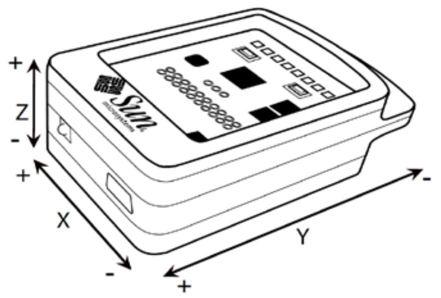


Figure 1.1.1 3D Accelerometer X, Y and Z axes
Image Source [**12**]

When a linear acceleration is applied to the Micro-Electro-Mechanical System (MEMS) sensor element, contained in the accelerometer sensor, it gets displaced from its nominal position and causes an electrical imbalance that is read via the sensor board’s analog-to-digital converter. [**12**]

1.1.2.2 Light Sensor

The Light sensor on the Sun SPOT detects the luminous light intensity of the surrounding light. The light intensity is returned as an integer that ranges from 0 to 750, with 0 being "very dark". [**21**]

Sun SPOT is a good platform to work with WSN technologies partially due to the fact that it allows programming in Java.

1.2 Patient Monitoring System

1.2.1 Background and Motivation

In this report, we describe the physical activity monitoring system that we have developed using Sun SPOTs. A physical activity monitoring system can be useful for surveillance and evaluation of a patient's physical activity. Such systems can be used to improve the quality of a medication program by evaluating previous physical activity records. It allows us to monitor a patient's physical activity remotely and thus, reduce the time and the effort for doing the same task at a nurse station.

1.2.2 Overview of the System

The system uses the 3D Accelerometer Sensor of the Sun SPOT which monitors the body movements of a person. A person is outfitted with four Sun SPOTs, one each on chest, thigh, leg and arm. These Sun SPOTs senses the acceleration and tilt values of each SPOT in the direction of X, Y and Z axis at each body location and communicate these values wirelessly to a basestation, which is connected to a Host Application (PC, PDA, laptop etc.,). Depending on the data from the various Sun SPOTs, the system can identify various postures which include Lying Down, Sitting, Standing, Walking, Bending, Arm Movement and other patterns such as trying to sit, trying to stand, trying to lie down, etc. Using different patterns based on these postures, the system can identify high-level actions performed by a patient. The application can also detect whether a patient has entered a dark room by using the light sensor values. All this information is stored in a database and displayed using a GUI.

This activity can be monitored remotely by reading the information from the database and by broadcasting the information using TCP/IP connection to another host.

The system uses Sun SPOT 3D Accelerometer Sensors to detect the acceleration and tilt values and determine the posture of a person. The system is programmed using JAVA [**17**] with Netbeans 6.8 IDE [**20**] and uses MySQL [**19**] as the database to record the physical activity of a patient and a Graphical User Interface (GUI) has been developed using JAVA Swings [**18**].

CHAPTER 2 - SUN SPOT

The Sun Small Programmable Object Technology or Sun SPOT is a battery-powered, wireless computing device that is being used in sensor networks. Multiple Sun SPOTs can be connected together to form a Wireless Sensor Network.

2.1 Hardware

The Sun Small Programmable Object Technology Hardware Configuration is as follows, The Sun SPOT is based o a 32-bit 180 MHz ARM920T core, and has 512K of RAM and 4MB of flash memory. The Radio part is a CC2420 802.15.4 radio with an effective range of about 80 meters. The radio antenna is a trace on the processor board.

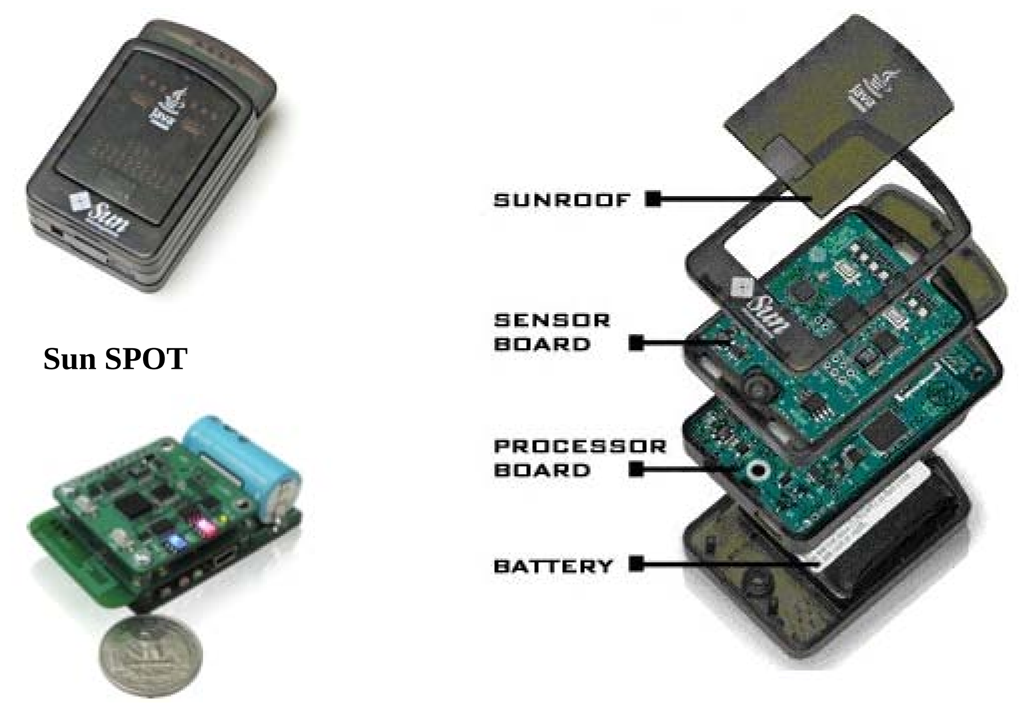


Figure 2.2.1 Sun SPOT Hardware

*Image Source [**13**] and [**16**]*

2.1.1 Processing

The processing configuration of a Sun SPOT contains a 180 MHz 32 bit ARM920T core processor with 512K RAM and 4M Flash. It has a 2.4 GHz IEEE 802.15.4 radio with integrated antenna, an AT91 timer chip and a USB interface. [**13**]

2.1.2 Sensor Board

The sensor board comprises of a 2G/6G three-axis accelerometer, a Temperature sensor, Light sensor, 8 tri-color LEDs, 6 analog inputs, 2 momentary switches and 5 general purpose I/O pins and 4 high current output pins. [**13**]

2.1.3 Battery

The battery is a 3.7V rechargeable 750 mAh lithium-ion. It also has a 30 uA deep sleep mode. Also, provides software for automatic battery management. [**13**]

2.1.4 Protocol

The motes communicate using the IEEE 802.15.4 standard. The SPOT supports the IEEE 802.15.4 MAC layer, on top of which communication protocol like Zigbee can be built. Sun SPOTs communicate wirelessly to one another via the 802.15.4 “Zigbee” Protocol. The base-station also uses the IEEE 801.15.4 standard. [**13**]

2.2 Software

2.2.1 Squawk Virtual Machine

The Sun SPOT system uses the “Squawk VM,” a small J2METM virtual machine (VM). It can host multiple applications concurrently and run wireless transducer application without requiring an underlying operation system. This saves a lot of overhead and improves the performance. The Squawk VM has been programmed almost entirely in JAVA.

It has a fully capable J2ME-level Java VM with OS functionality and currently has an 80K RAM for VM. It can directly execute out of flash memory. The device drivers for the Sun SPOTS are written in JAVA. [**9**][**13**]

2.2.2 Developer Tools

Standard IDEs such as NetBeanTM can be used to create Java code for Sun SPOT applications. An ant based scripted build and deploy process is used.

The deployment options comprises of wired serial or USB connection to battery board or using a wired USB connection to Test board. An Over-the-air deployment of Java code on to Sun SPOTs is also possible through the basestation. A Sun SPOT wired via USB or Serial to a computer can act as a basestation. [**13**]

2.2.3 Standard Libraries

Some of the standard libraries are CLDC 1.0 libraries, 802.15.4 compliant MAC layers and hardware and sensor integration/control libraries. [**13**]

2.3 Basestation

The basestation acts as an interface between the other Sun SPOTs and the host workstation by providing a radio gateway. It does not have any sensors in it. It does not contain a battery and the power is supplied by a USB connection to the host workstation.

The Sun SPOT basestation can be used to remotely deploy and debug code on free range SPOTS using Over the Air Command Server, by communicating with the Sun SPOTS.

The basestation in connection with a development machine (host application) will allow the host application to write applications that make use of the basestation’s radio to communicate remotely with the Sun SPOTs. Note that a full Sun SPOT can also be used as a basestation, though in doing so its sensor board would not be used.

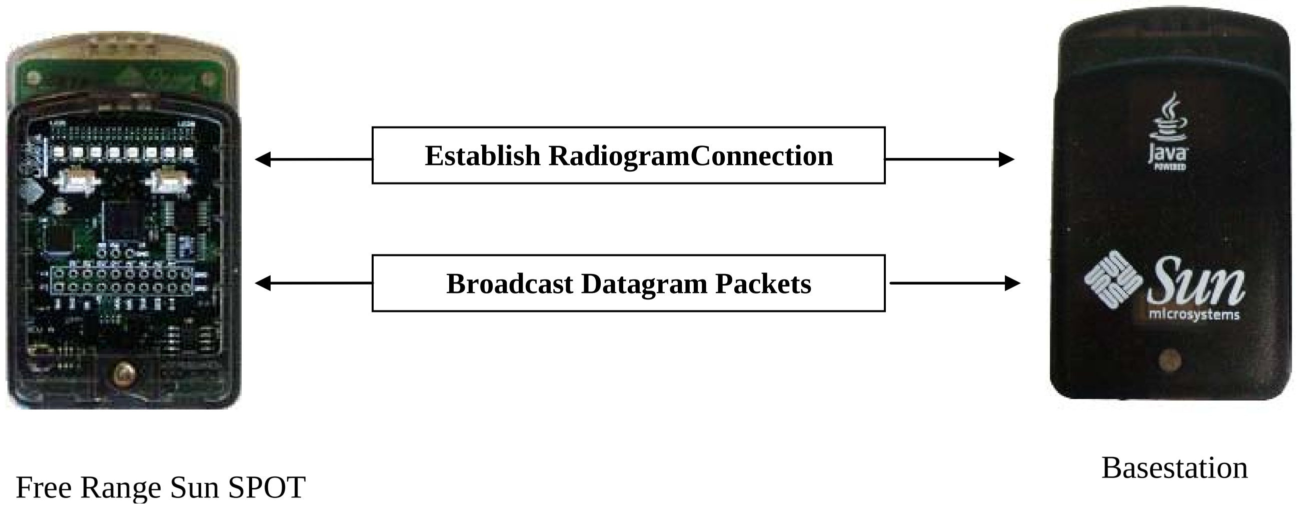


Figure 2.2.2 Radiogram Communication

2.4 Radiogram Protocol

The Sun SPOTs and basestation use the radiogram protocol to do wireless communication within and with each other. Radiogram Protocol is a client-server protocol that provides datagram-based IO between two devices.

Server Connection is opened using:

RadiogramConnection conn = (RadiogramConnection)
Connector.open("radiogram://:<portNo>");
where portNo is a port number in the range 0 to 255 that identifies this particular connection.

The client connection is opened using

RadiogramConnection conn =
(RadiogramConnection)Connector.open("radiogram://<serveraddr>:<portNo>");
where serverAddr is the 64bit unique IEEE Address of the radio of the server, and portNo is a port number in the range 0 to 255 that identifies this particular connection.
Both the port numbers of the Client and Server must match. Datagram packets are sent between client and server to communicate data.

A datagram is sent by calling the send command on the connection:
conn.send(dg);
A datagram is received by calling the received command on the connection:
conn.receive(dg);

The datagrams can only be used for their respective connections and cannot be obtained by other connections. An attempt to send datagrams to a different address than the one used in the connection opened will result in an exception.

It is permitted to open a server connection and a client connection on the same machine using the same port numbers. All incoming datagrams for that port will be routed to the server connection. [**15**]

CHAPTER 3 - Design and Implementation

The proposed system uses 4 Sun SPOT sensors, 1 basestation, and one Host Application. One sensor each is affixed on a person's chest, thigh, leg and arm as shown in the Figure 3.1. The application uses a basestation for wireless communication between the host and the sensors and is connected to the Host Application through a USB port. The Host Application requires a database setup to record patient's physical activity information detected by the application. This information in the database can be viewed at the nurse station and if something goes wrong, the nurse station can be alerted to attend the patient on time.

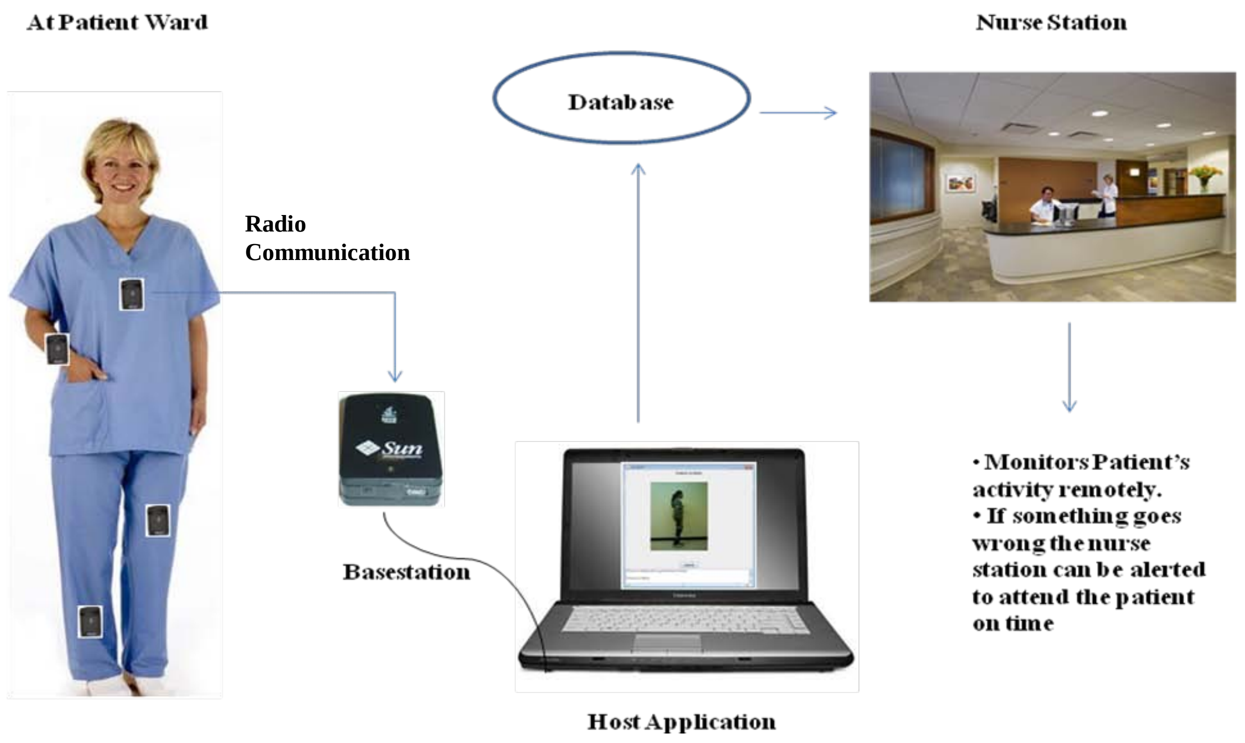

At Patient Ward
Nurse Station
Database
Radio Communication
Basestation
Host Application
• Monitors Patient's activity remotely.
• If something goes wrong the nurse station can be alerted to attend the patient on time

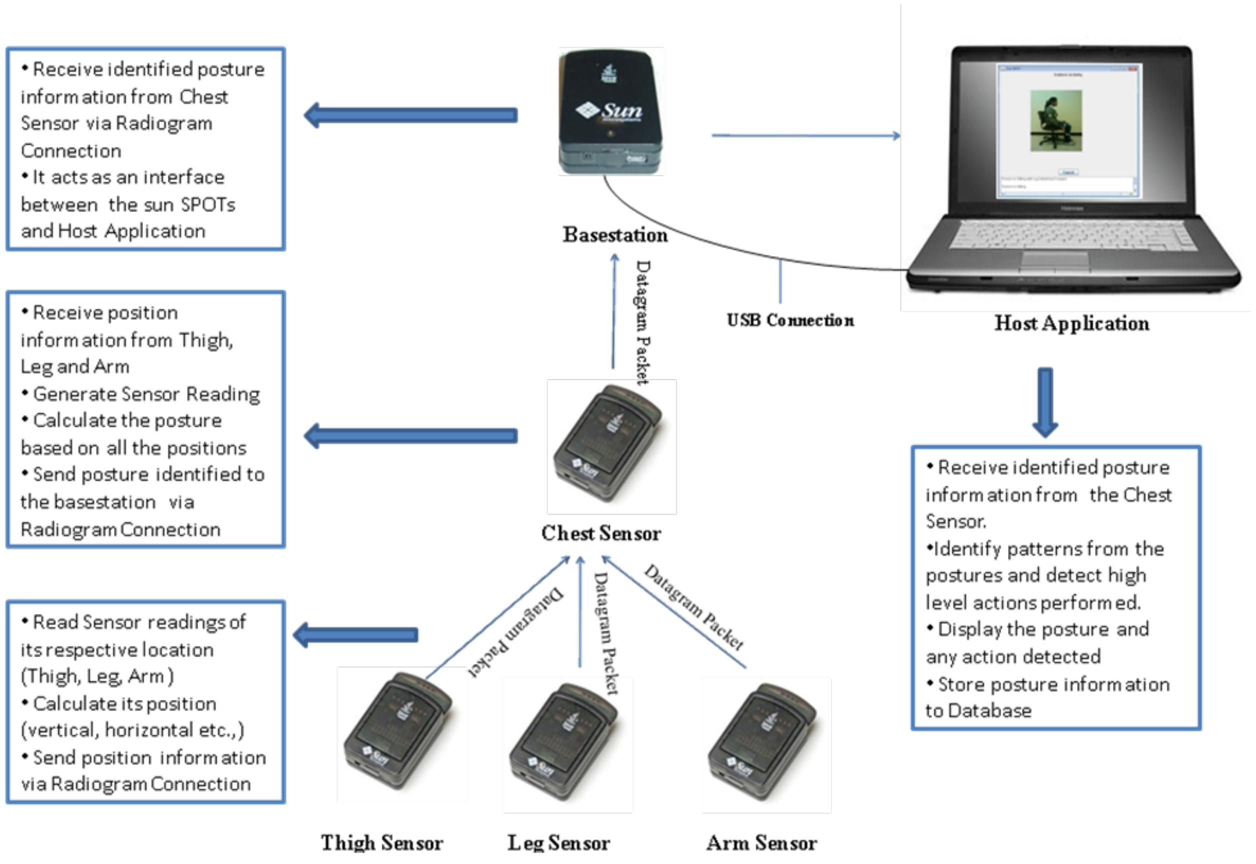
Figure 3.1 Application Deployed in a Hospital

Image source [3], [22] and [24]

3.1 Architecture

3.1.1 Hardware Architecture

Figure 3.2 shows the hardware architecture of the system. A JAVA application is deployed on each Sun SPOT which calculates the position value of its respective location i.e., the location of the chest, thigh, leg, and arm. The thigh, leg and arm Sun SPOTS send their position information to the chest Sun SPOT in the form of datagram packets via RadiogramConnection. The chest Sun SPOT receives these datagram packets, and from the received position values and the position values generated by the chest sensor, it calculates the posture of the person. The detected posture is sent to the basestation using a datagram via RadiogramConnection



• Receive identified posture information from Chest Sensor via Radiogram Connection
• It acts as an interface between the sun SPOTs and Host Application
Basestation
USB Connection
Host Application
Datagram Packet
• Receive position information from Thigh, Leg and Arm
• Generate Sensor Reading
• Calculate the posture based on all the positions
• Send posture identified to the basestation via Radiogram Connection
Chest Sensor
• Receive identified posture information from the Chest Sensor.
•Identify patterns from the postures and detect high level actions performed.
• Display the posture and any action detected
• Store posture information to Database
• Read Sensor readings of its respective location (Thigh, Leg, Arm)
• Calculate its position (vertical, horizontal etc.,)
• Send position information via Radiogram Connection
Datagram Packet
Datagram Packet
Datagram Packet
Thigh Sensor
Leg Sensor
Arm Sensor

Figure 3.2 System Hardware Architecture

The basestation receives the information and transmits it to the Host Application. In the host application the basestation gathers patient information from the sensor attached on the Patient's chest, which in turn acts like a basestation for the other sensors on the thigh, leg and arm of that patient. The host application displays the current posture of the patient on a GUI from the gathered information.

3.1.2 Software Architecture

The system has three applications running:

- Host Application: A host application running on a host connected to the basestation.
- GUIApplication: A Java application running on a PC at the nurse station.
- SensorApplication: A Sun SPOT application deployed on the Sun SPOT sensors.

Host Application:

The application takes the following input from the GUI:

- Sensor IEEE Address
- Each sensor's position on a patient's body.
- Patient ID, i.e., to which patient the sensors are attached to.

The application sends a Radiogram message to all the free range Sun SPOTs informing them about their location on a patient's body. Then, it waits to receive the patient's posture from the chest sensor. Upon receiving the patient's posture, it stores into the database the patient's posture with the current timestamp. It then looks for patterns in the postures to identify any action.

Sensor Application:

This application is deployed on to the Sun SPOT to be positioned on the arm, thigh, leg and chest. It waits for a Radiogram message from the Host Application informing about its position and the chest sensor IEEE Address of that patient. After reading its position, it performs its respective functionality which includes generating accelerometer sensor values and the following tasks depending on their locations:

- Thigh Sensor: Calculate the position of thigh and send it to the chest sensor.
- Leg Sensor: Calculate the position of leg and send it to the chest sensor.

- Arm Sensor: Calculate the position of arm, generate light sensor values and send a message to chest sensor if it is dark as well as send its position.
- Chest Sensor:
  - Receives position values from Thigh, Leg and arm Sensor
  - Calculate the complete posture of the person based on the position values of chest, thigh, leg and arm sensors
  - Sends the person's posture to the Host Application

GUIApplication:

This application takes as input the Patient's Id and retrieves the patient's current posture from the database and displays it on the GUI pictorially.

3.1.2.1 State Machine of the System

The System uses a State Machine Diagram to represent the reachable postures from each posture. All the nodes in the state machine are postures detected by the system and the paths from one posture to another posture determines the reachability of the posture. The System uses this state machine diagram as a base to determine valid reachable postures, to define high level actions and to detect abnormality in physical activity. Figure 3.3 displays the State Machine Diagram of the system. All the arrows in the state machine are bidirectional. Hence, if a posture is shown as reachable from one posture then it can return back to the previous posture as well.

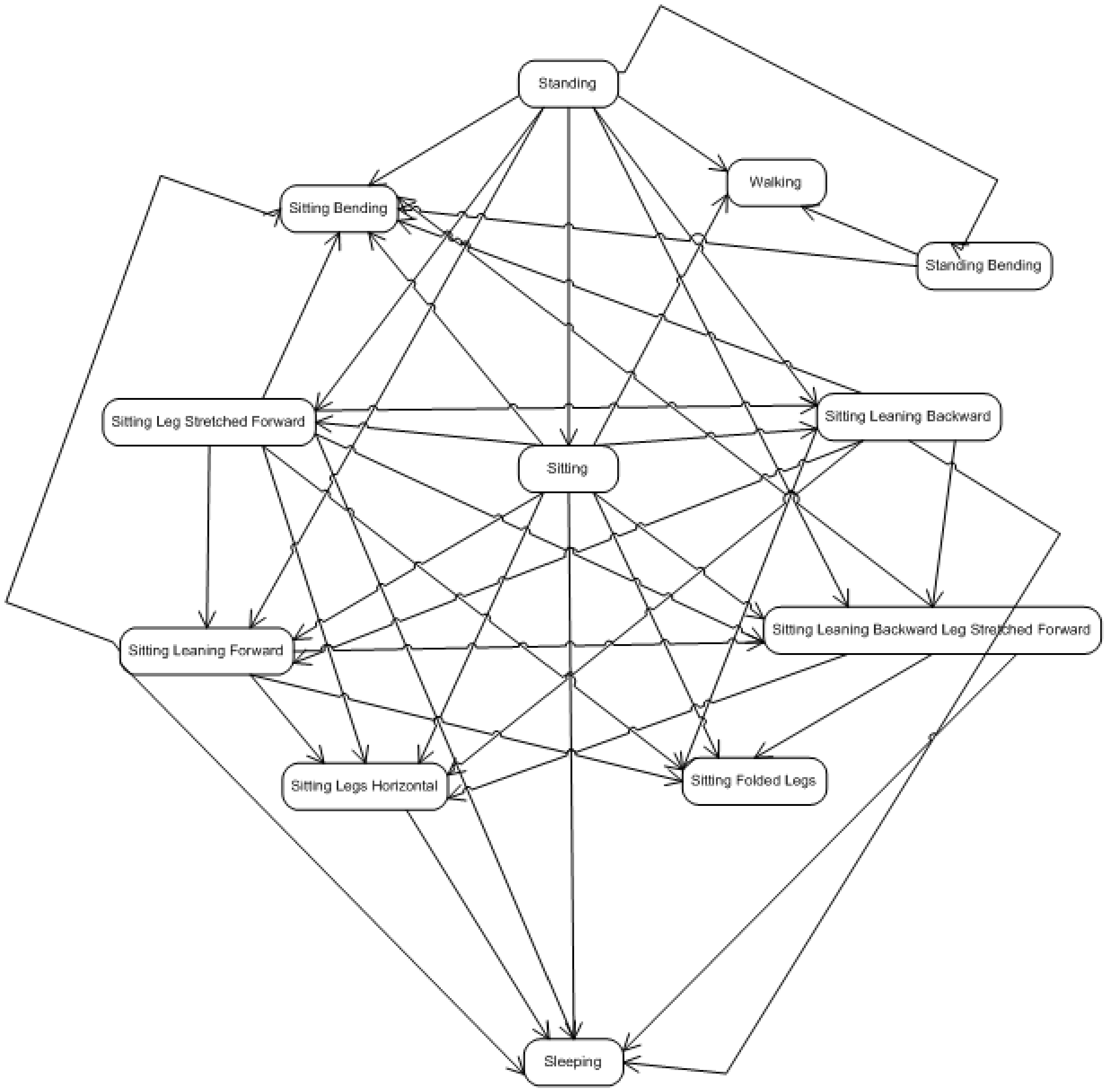


Figure 3.4 State Machine Representing Valid Paths from each Posture

Note: The Paths in the State Machine are bi directional

3.1.2.2 Algorithms

A posture is calculated based on the position values of thigh, leg and chest sensors.
Each sensor gets the following values about its position,

- Tilt in X axis direction
- Tilt in Y axis direction
- Tilt in Z axis direction
- Acceleration in X axis direction
- Acceleration in Y axis direction
- Acceleration in Z axis direction
- Total acceleration

It calculates postures based on the average of the values obtained.
For example: To calculate the Sitting posture of a person, the following should be satisfied

Posture: Sitting

	Chest	**Thigh**	**Leg**
Position	Vertical	Horizontal	Vertical
X axis Tilt	0	0	0
Y axis Tilt	-90	0	-90
Z axis Tilt	0	90	0

Table 3.1 Sitting Posture Calculation Values

The system defines a Sitting posture when the chest is positioned vertical, thigh is positioned horizontal and the leg is positioned Vertical. The arm sensor readings do not contribute to this calculation as the readings from chest, thigh and leg are sufficient to determine the Sitting posture. As observed in Table 3.1 there is a change in Y axis and Z axis when there is a change from Vertical to Horizontal position.

The SensorApplication deployed on each Sun SPOT waits to receive information from the basestation about which body location it is being placed at. Once, it learns about its position the Sun SPOT executes its respective functionality. The getLegPosition(), sendLegPosition() functions run on the Sun SPOT placed on the leg. Each Sun SPOT on the chest, thigh, leg and arm run their respective getPosition() and sendPosition() functions defined in the SensorApplication. This following algorithm shows the functionality of the Sun SPOT on the leg

```
/* Posture Detection*/
/*get the current position of leg */
String getLegPosition ()
{
        if(avgTiltX around 0 && avgTiltY around -90 && avgTiltZ around 0)
                return "Vertical";
        if(avgTiltX around 0 && avgTiltY around 0 and && avgTiltZ around 90)
                return "Horizontal";
         ….  Other position conditions continue
}
/* Send the current position of the leg to the chest Sun SPOT via Radiogram Communication*/
void sendLegPosition ()
{
         RadiogramConnection conn = (RadiogramConnection)
        Connector.open("radiogram://"+chestSunSPOTIEEEAddress+":66");
        Datagram dg = conn.newDatagram(conn.getMaximumLength());
        While (true)
        {
                If (getLegPosition ().equals ("Vertical"))
                {
                 /* Send leg position information to chest Sun SPOT in a datagram packet*/
                   dg.writeUTF("Vertical”);
                conn.send(dg);
            }
        …. conditions to detect other positions
        }
}
```

The CheckPosture() function is executed on the chest Sun SPOT. After receiving position information from the thigh, leg and arm Sun SPOT it checks to see if they fall under the definition of a posture.

```
CheckPosture ()
{
        /* From the position information received from the Sun SPOTs it continuously checks to
        see if the readings fall under any one of the defined range for a posture */
        RadiogramConnection conn = (RadiogramConnection)
        Connector.open("radiogram://"+basestationIEEEAddress+":67");
        Datagram dg = conn.newDatagram(conn.getMaximumLength());
        While (true)
        {
                If (thigh ("Horizontal") && leg ("Vertical") && chest ("Vertical"))
                {
                        dg.writeUTF("Sitting");
                        conn.send(dg);
                }
                If (thigh ("Vertical") && leg ("Vertical") && chest ("Vertical")
                {
                        dg.writeUTF("Standing");
                        conn.send(dg);
                }
                 …. conditions to detect other postures
        }
}
```

We define an *Action* as a sequence of valid postures which a person has gone through. An action sequence can be one or more possible paths from a posture as defined in Figure 3.4. Therefore, an Action is any of the possible paths from an initial posture to a final posture in the System's State Machine Diagram (Figure 3.4) that renders a meaningful sequence in real time environment. For each action a pattern is defined using a State Machine Diagram.

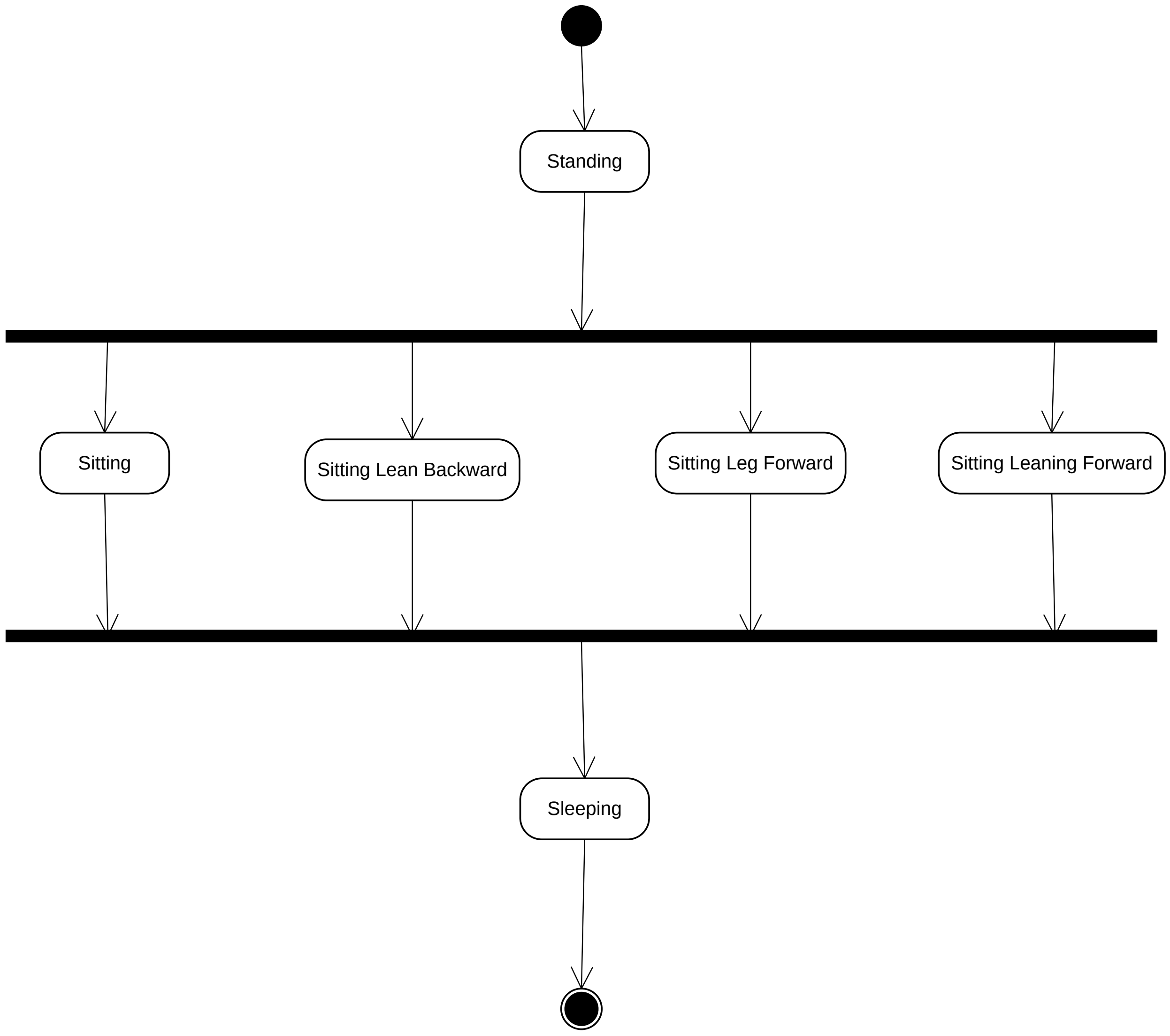


Figure 3.3 A State Machine Representing an Action
Action: "Was standing and went and lied down"
Note: The State Machine has been constructed by extracting a pattern from the System's State Machine Diagram in Figure 3.4.

An algorithm for detecting high level actions by identifying patterns from the detected postures is defined as follows:

The Host Application maintains an ActionPath String to record the sequence of postures detected; Every time a new posture is detected, it validates if the posture is a valid reachable posture from the previous posture by checking the state machine as shown in Figure 3.4. A valid posture is appended to the Action Path.

The ActionPath constantly checks for a pattern that matches an action. This pattern of postures from which an Action is defined is in turn defined in a state machine diagram i.e., each Action has a state machine diagram defined for it. A sample state machine diagram for an Action is show in Figure 3.5. This state machine diagram for an action is derived from the system's state machine diagram by extracting all the possible paths between postures that defines a particular Action.

The sequence is verified based on the timestamp recorded for each posture. If there are a set of postures that were performed in an order of timeline, which is one of the possible paths in the State Machine defined for a particular action, then an Action is said to be performed.

```
/* Action Detection*/
CheckAction ()
{
    /* Validate if the current posture is reachable from the previous posture based on the
    State Machine Diagram of Figure 4.4*/

    If (Validate (posture))
        ActionPath = ActionPath + posture
    CheckPattern ();
}
```

```
CheckPattern ()
{
    /* Path of Postures[] contains a sequence postures that define a particular action*/

    /* Exercise Action*/
    If (ActionPath.Contains(path of Postures[])))
    {
        exerciseCount++;
        If (exerciseCount >=3)
        {
            ActionPath.Empty();
            Print "Exercise Action: Action Name"
        }
    }

    /* General Action*/
    If (ActionPath.Contains(path of Postures[])))
    {
        /* Checks the order of occurrence of postures based on the timestamp values*/
        If(TimeStampInSequence(path of Postures[]))
        {
            ActionPath.empty();
            Print "Action";
        }
    }
}
```

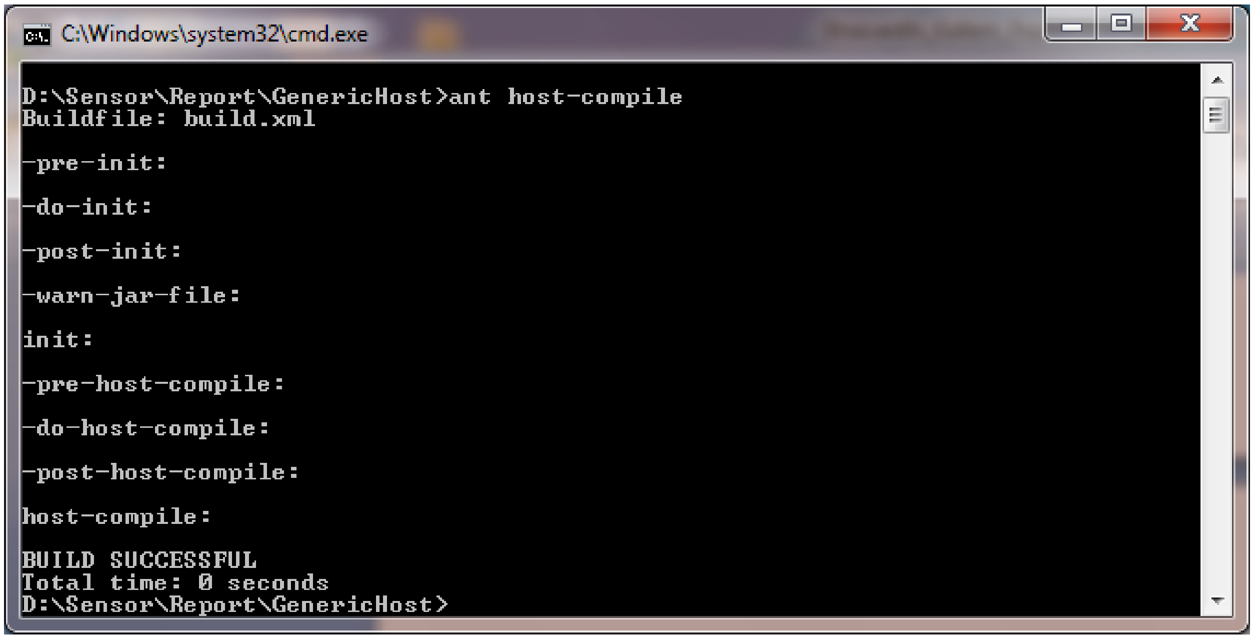
CHAPTER 4 - Deployment

4.1. System in Execution

To run the system we require the following software to be installed: Sun JDK, Sun Java Runtime Environment, Apache Ant Build environment, Sun SPOT SDK and Sun SPOT device driver to connect the basestation to the host application. [**11**]

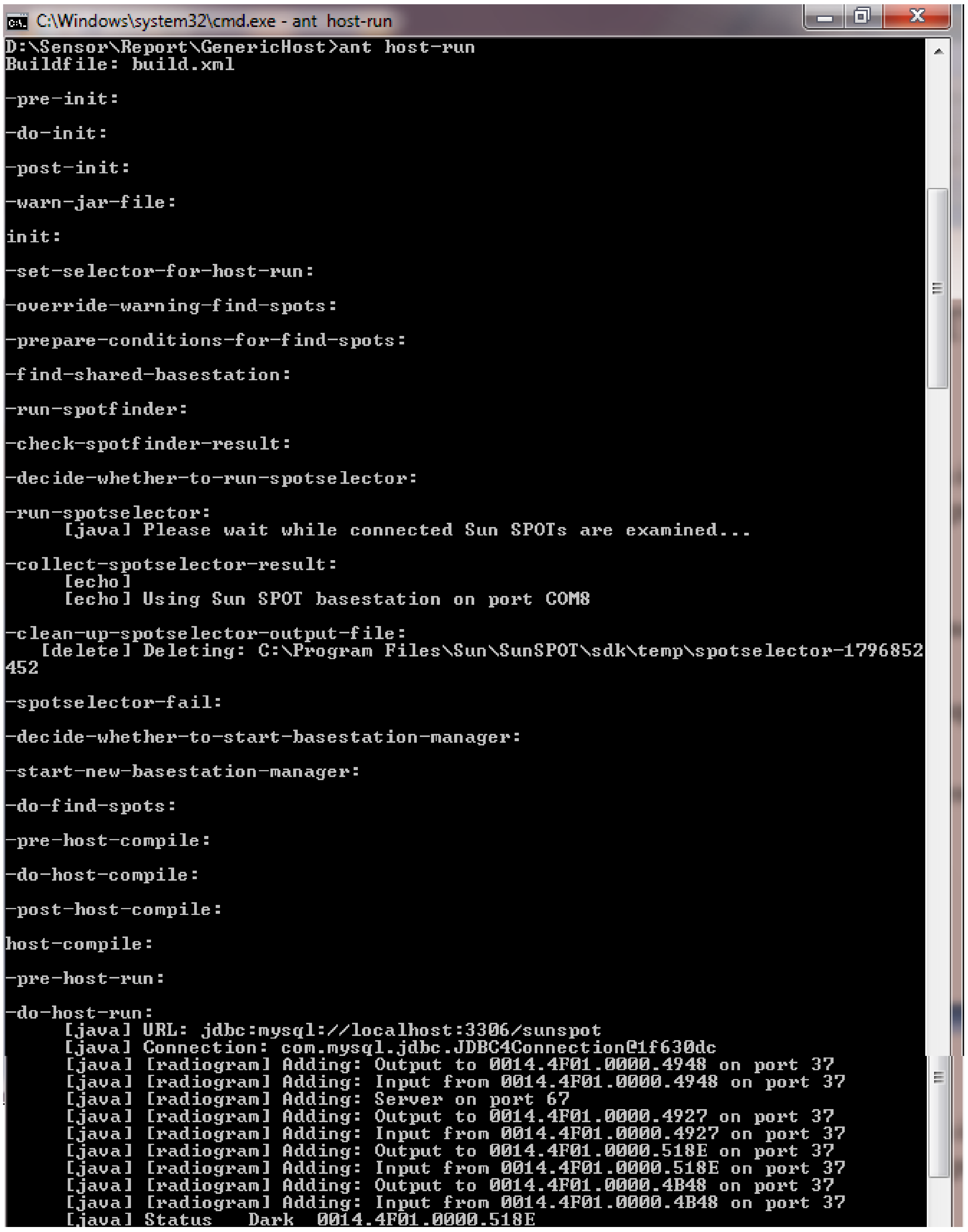
Using command prompt the Sensor_Appliction can be deployed on to the free range Sun SPOTs by executing the commands *ant jar-app* and *ant jar-deploy*. Once the Sun SPOT is turned on by pressing the power button, it starts executing the deployed application.

The application can be executed by compiling the Host Application using *ant host-compile* and run using the *ant host-run* commands. The user interface in Fig 4.1 will prompt and will ask for identification (unique IEEE address) of Sun SPOT.



```
D:\Sensor\Report\GenericHost>ant host-compile
Buildfile: build.xml

-pre-init:

-do-init:

-post-init:

-warn-jar-file:

init:

-pre-host-compile:

-do-host-compile:

-post-host-compile:

host-compile:

BUILD SUCCESSFUL
Total time: 0 seconds
D:\Sensor\Report\GenericHost>
```

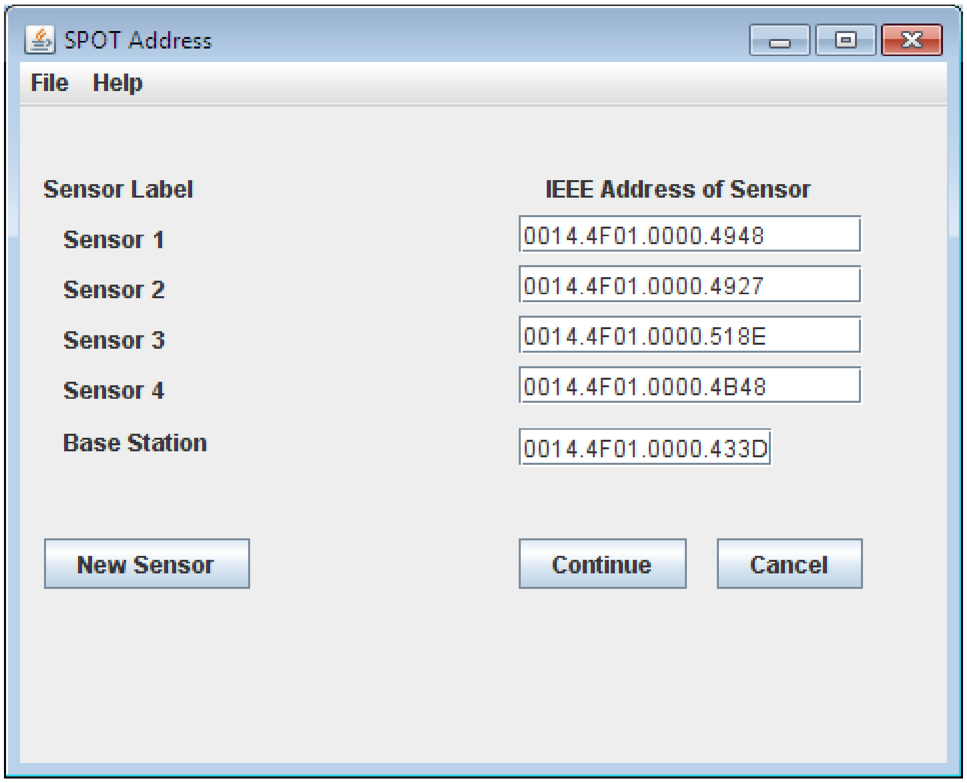
Figure 4.1 Host Application Compile at Command Prompt



C:\Windows\system32\cmd.exe - ant host-run

```
D:\Sensor\Report\GenericHost>ant host-run
Buildfile: build.xml

-pre-init:

-do-init:

-post-init:

-warn-jar-file:

init:

-set-selector-for-host-run:

-override-warning-find-spots:

-prepare-conditions-for-find-spots:

-find-shared-basestation:

-run-spotfinder:

-check-spotfinder-result:

-decide-whether-to-run-spotselector:

-run-spotselector:
     [java] Please wait while connected Sun SPOTs are examined...

-collect-spotselector-result:
     [echo]
     [echo] Using Sun SPOT basestation on port COM8

-clean-up-spotselector-output-file:
   [delete] Deleting: C:\Program Files\Sun\SunSPOT\sdk\temp\spotselector-1796852
452

-spotselector-fail:

-decide-whether-to-start-basestation-manager:

-start-new-basestation-manager:

-do-find-spots:

-pre-host-compile:

-do-host-compile:

-post-host-compile:

host-compile:

-pre-host-run:

-do-host-run:
     [java] URL: jdbc:mysql://localhost:3306/sunspot
     [java] Connection: com.mysql.jdbc.JDBC4Connection@1f630dc
     [java] [radiogram] Adding: Output to 0014.4F01.0000.4948 on port 37
     [java] [radiogram] Adding: Input from 0014.4F01.0000.4948 on port 37
     [java] [radiogram] Adding: Server on port 67
     [java] [radiogram] Adding: Output to 0014.4F01.0000.4927 on port 37
     [java] [radiogram] Adding: Input from 0014.4F01.0000.4927 on port 37
     [java] [radiogram] Adding: Output to 0014.4F01.0000.518E on port 37
     [java] [radiogram] Adding: Input from 0014.4F01.0000.518E on port 37
     [java] [radiogram] Adding: Output to 0014.4F01.0000.4B48 on port 37
     [java] [radiogram] Adding: Input from 0014.4F01.0000.4B48 on port 37
     [java] Status    Dark  0014.4F01.0000.518E
```

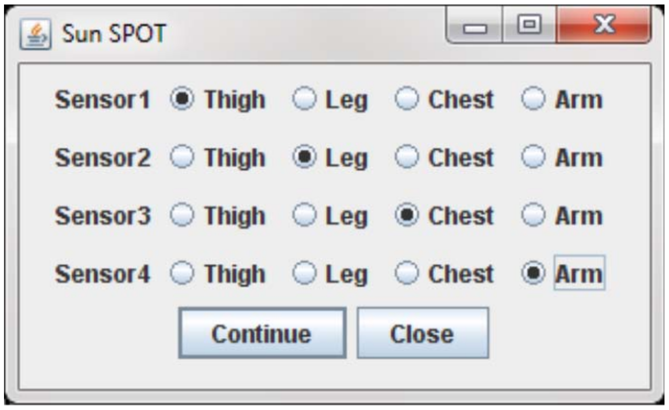
Figure 4.2 Host Application Run from Command Prompt



Sensor Label	IEEE Address of Sensor
Sensor 1	0014.4F01.0000.4948
Sensor 2	0014.4F01.0000.4927
Sensor 3	0014.4F01.0000.518E
Sensor 4	0014.4F01.0000.4B48
Base Station	0014.4F01.0000.433D

Figure 4.3 Sun SPOT Address Information

After entering all the sensor identification information, it will ask for all the SPOT's positioning information as shown in Figure 4.2.



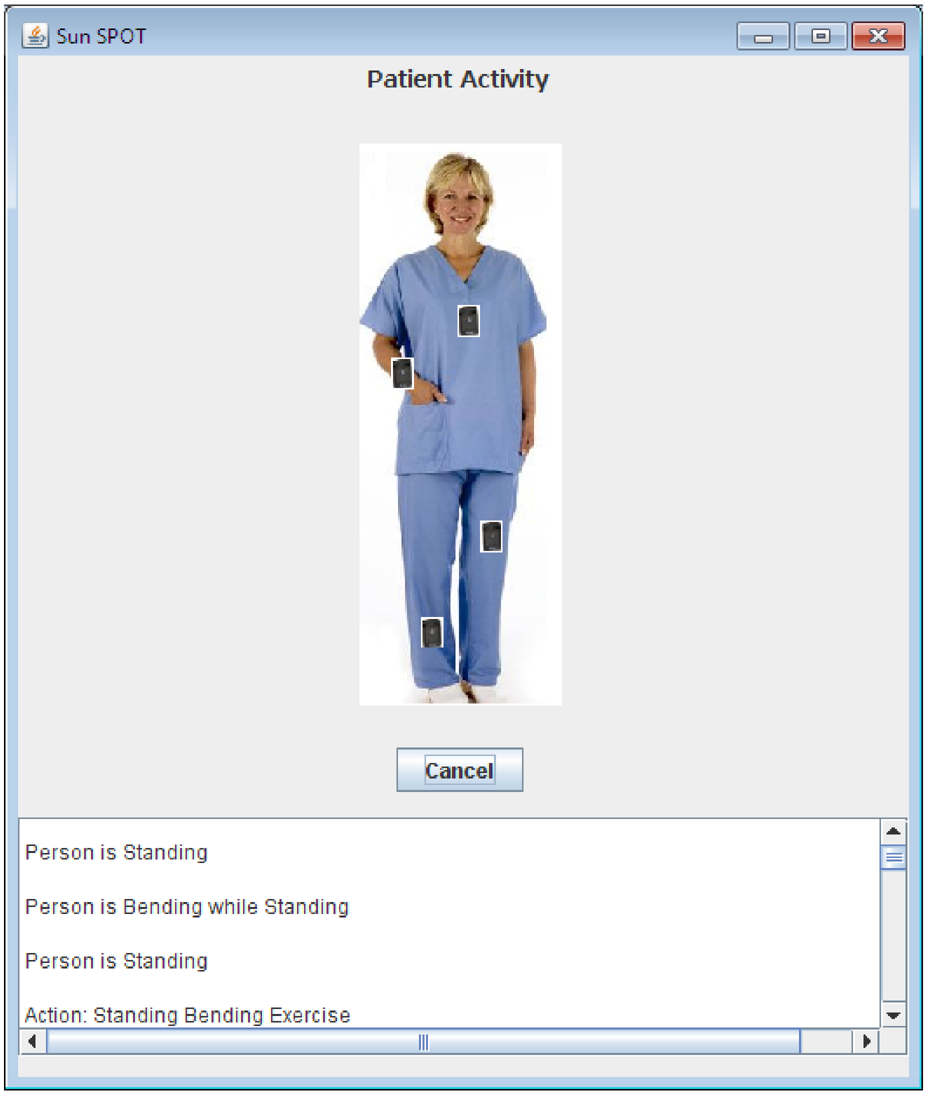
Sensor1 [✓ Thigh] Leg Chest Arm
Sensor2 Thigh [✓ Leg] Chest Arm
Sensor3 Thigh Leg [✓ Chest] Arm
Sensor4 Thigh Leg Chest [✓ Arm]

Figure 4.4 Sun SPOT Positioning Information

After getting the entire free range SPOT's and basestation details, radiogram communication is established between the basestation and free range Sun SPOTs notifying them about the location they are going to operate on.

The Thigh, Leg and Arm SPOTs transmit their positioning information of the respective location to the Chest SPOT which does the necessary calculation to identify the posture of the person and transmits the position information to the host application via the basestation.

The basestation displays the posture information on the GUI as shown in Figure 4.6. This application then looks for patterns in the identified postures and displays any action detected as shown in Figure 4.6.



```
Person is Standing

Person is Bending while Standing

Person is Standing

Action: Standing Bending Exercise
```

Figure 4.5 Posture Information Window

Image source [3]

4.2 Actions

4.2.2 Action 1: Was lying down then started walking

"Was Lying Down started walking" action performance can have the many possible different paths as represented in the State Machine diagram of Fig 4.5. Considering one possible path of Lying Down -> Sitting -> Standing -> Walking the action would be detected as follows:

4.2.2.1 State Machine

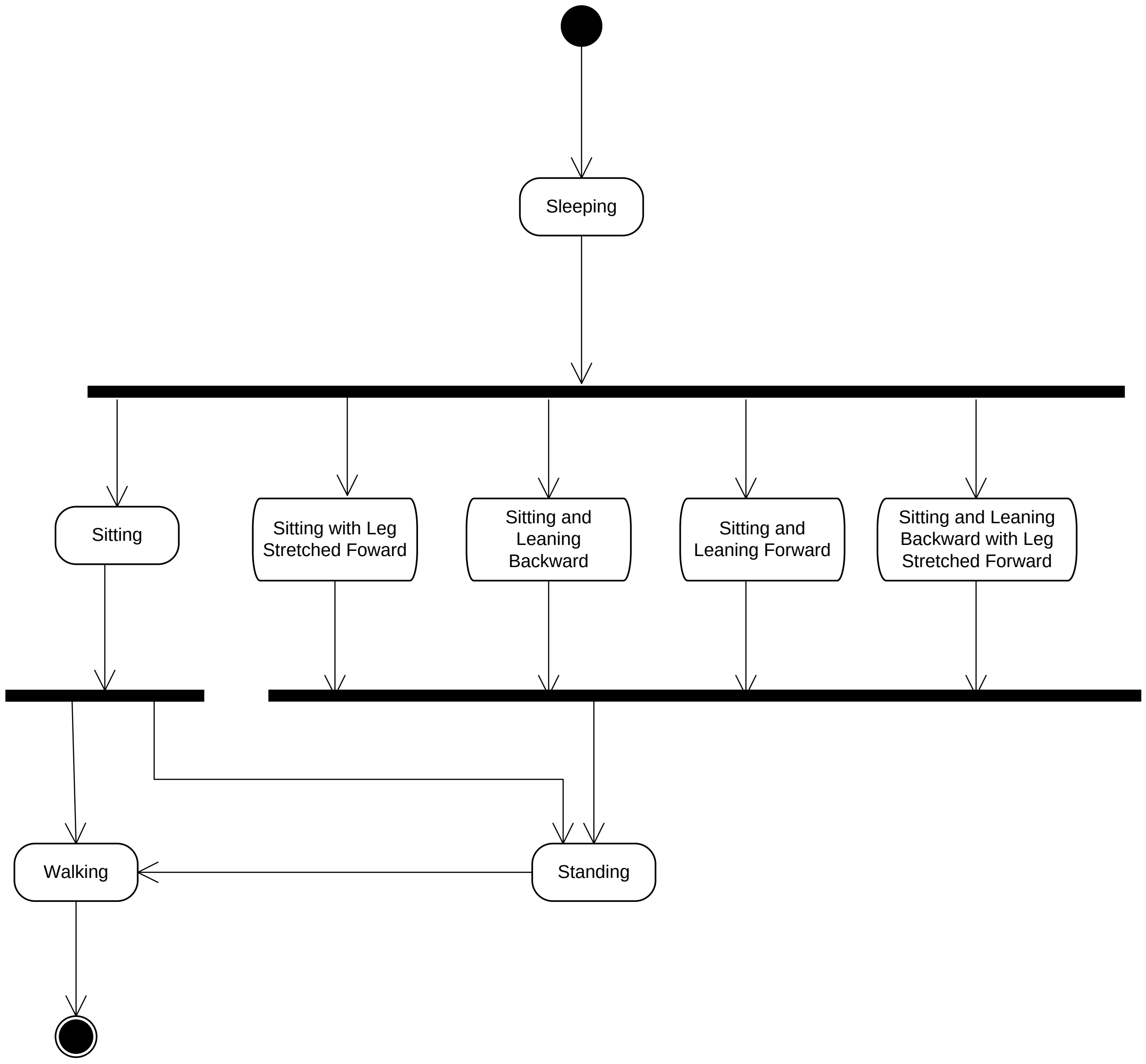
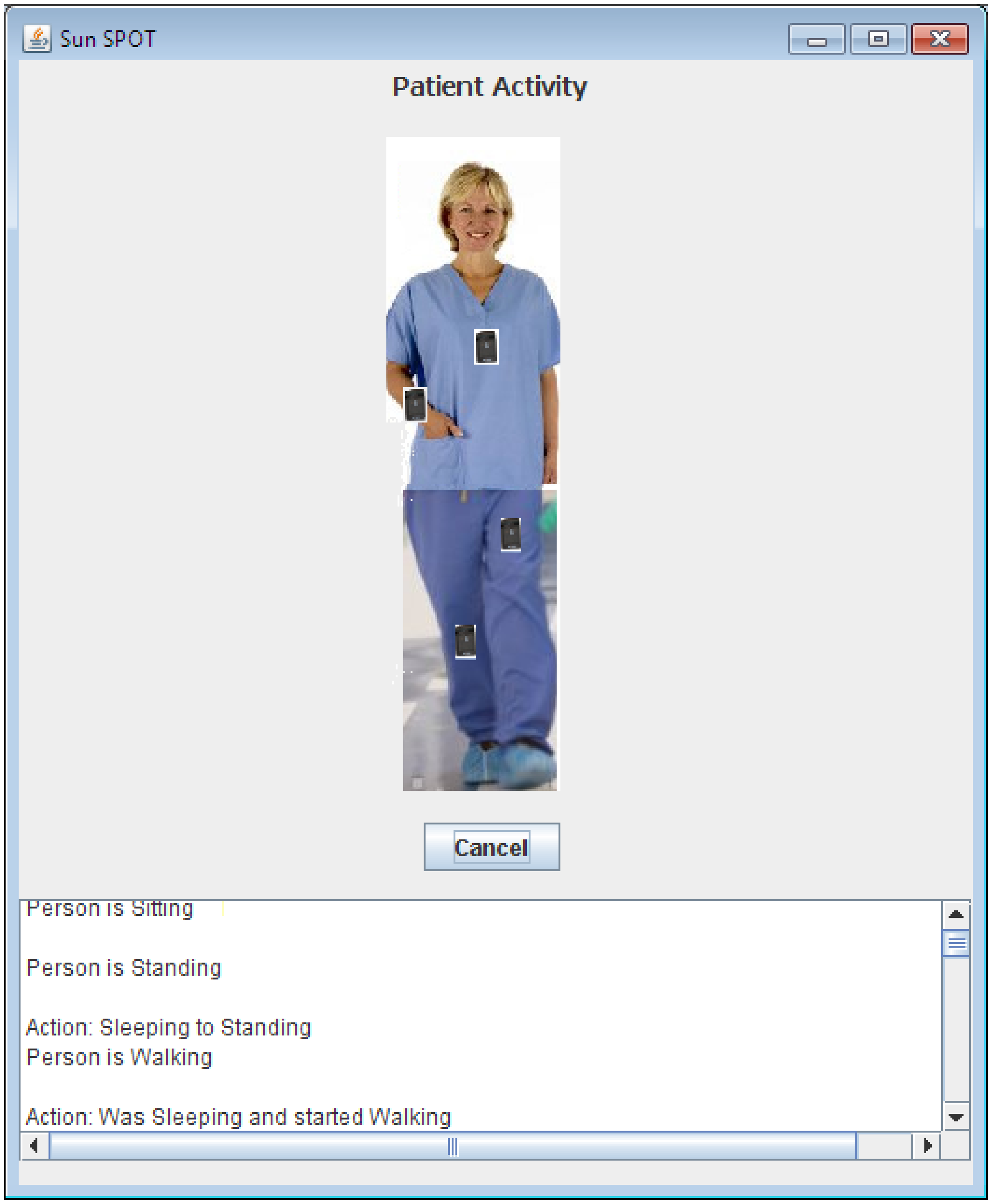

Sleeping
Sitting
Sitting with Leg Stretched Foward
Sitting and Leaning Backward
Sitting and Leaning Forward
Sitting and Leaning Backward with Leg Stretched Forward
Walking
Standing

Figure 4.6 Action 1 State Machine Diagram

Action: Was lying down then started walking

4.2.2.2 Execution



```
Person is Sitting

Person is Standing

Action: Sleeping to Standing
Person is Walking

Action: Was Sleeping and started Walking
```

Figure 4.7 Output: Was lying down then started walking

Image source [4]

4.2.3 Action 2: Bending Exercise

“Standing to Bending Exercise“ action performance will have the path as represented in the State Machine diagram of Figure 4.7. Considering the path of Standing -> Standing Bending occurs at least 3 times, the action would be detected as follows:

4.2.3.1 State Machine

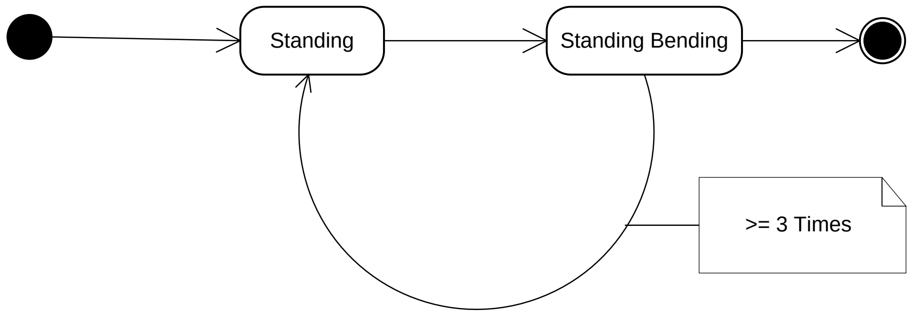
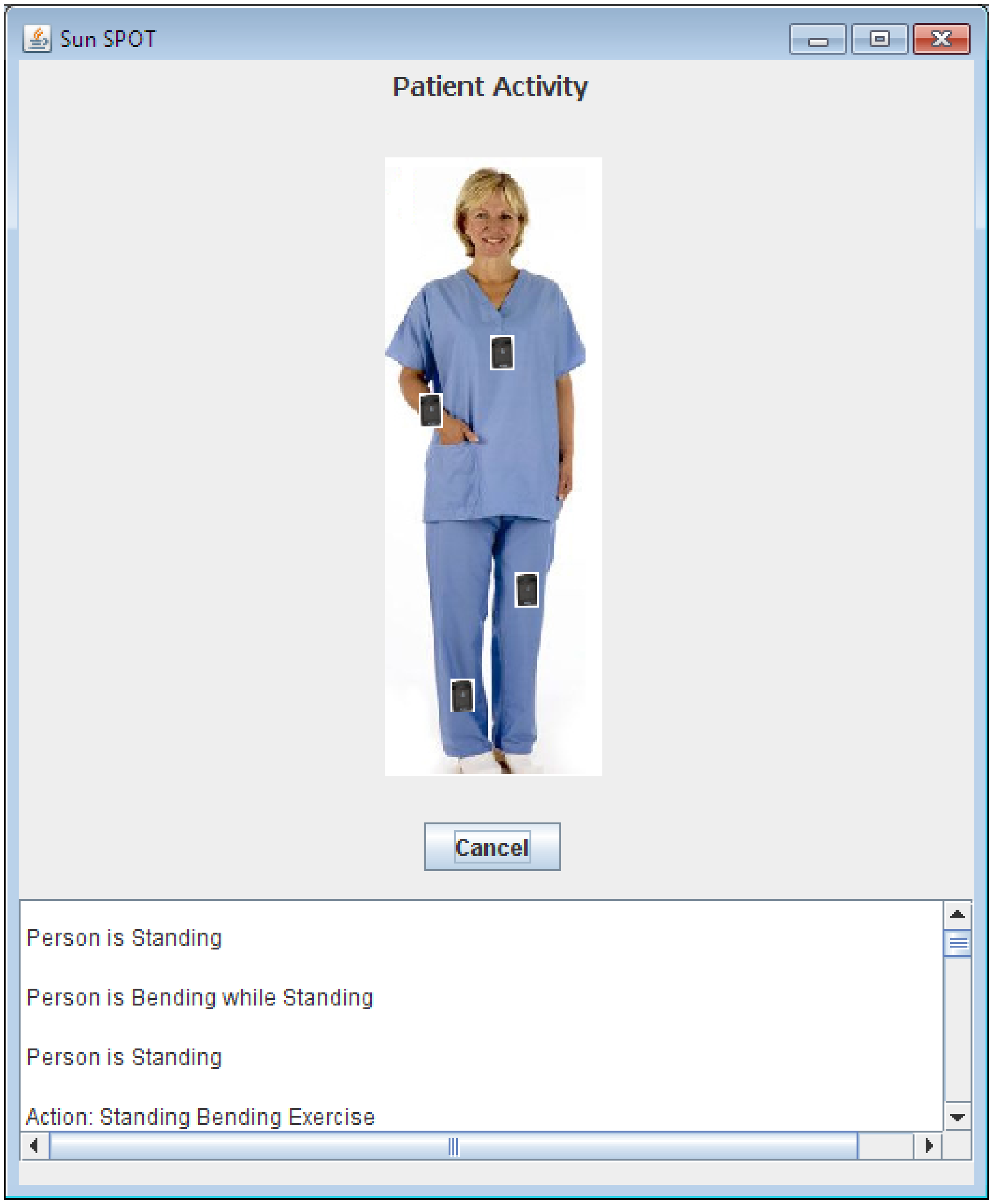


Figure 4.8 Action 2 State Machine Diagram

Action: Standing to Bending Exercise

4.2.3.2 Execution



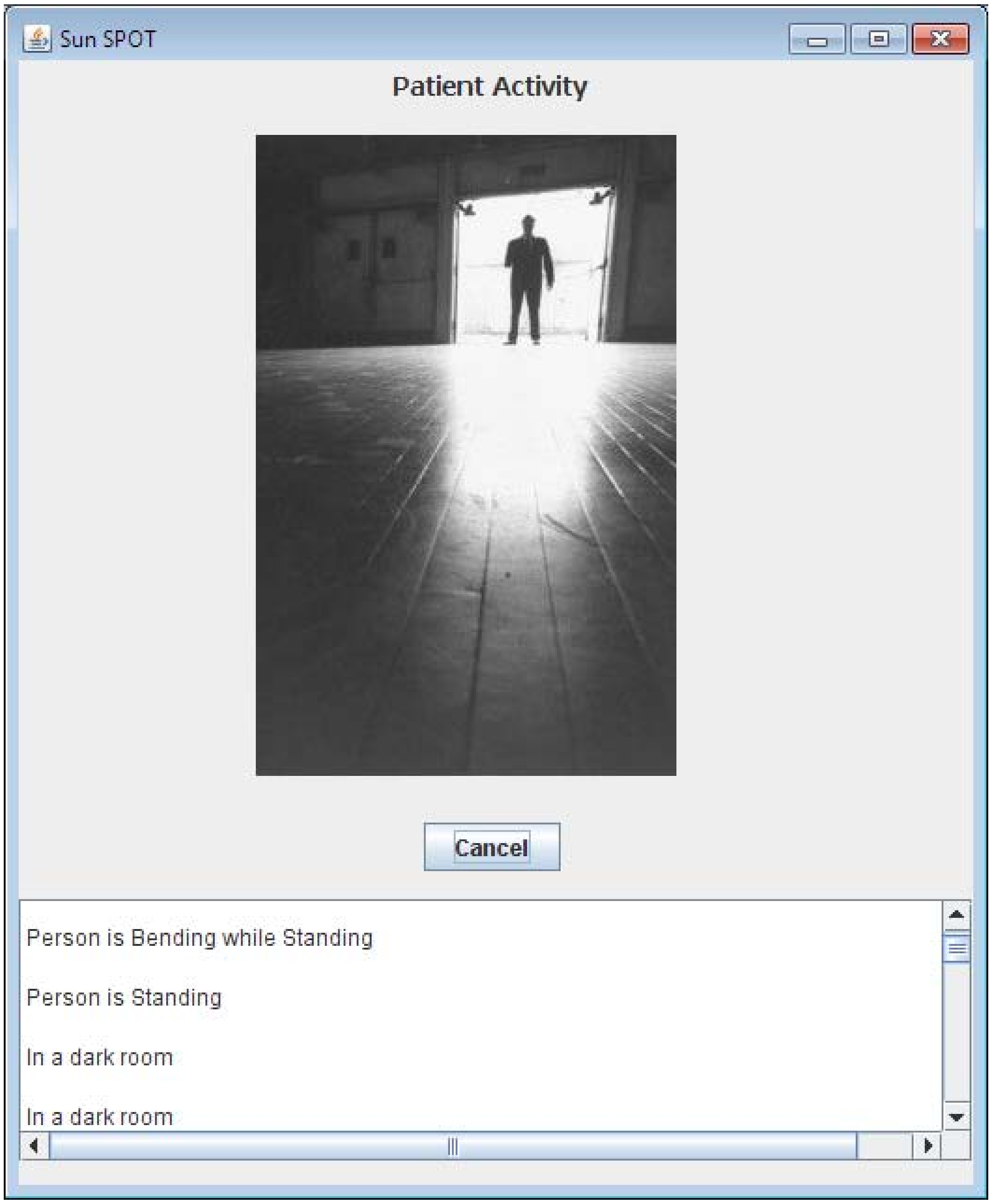
```
Person is Standing

Person is Bending while Standing

Person is Standing

Action: Standing Bending Exercise
```

Figure 4.9 Output: Standing Bending Exercise

Image Source [3]

4.2.4 Patient Entered a Dark Room

The light sensor reads the light intensity values and if falls below the definition of brightness it alters that a Patient has entered a dark room.



```
Person is Bending while Standing

Person is Standing

In a dark room

In a dark room
```

Figure 4.10 Patient Entered a Dark Room

Image Source [23]

4.2.5 List of Postures Detected by the System

- Person is Standing
- Person is Sitting
- Person is Sitting with Leg Stretched Forward
- Person is Sitting and Leaning Backward
- Person is Sitting and Leaning Forward
- Person is Sitting Leaning Backward with Leg Stretched Forward
- Person is Sitting with Legs Horizontal
- Person is Sitting with Folded Legs
- Person is Lying Down
- Person is Walking
- Person is Bending while Sitting
- Person is Bending while Standing
- Arm is Moving

4.2.6 List of Patterned Postures Detected by the System

- Trying to Sit
- Trying to Stand

4.2.7 List of Actions Detected by the System

Exercise Actions

- Standing and Bending Exercise
- Leg Exercise
- Sit up Exercise
- Sitting and Bending Exercise

General Actions

- Was lying down and started walking
- Was lying down woke up and stood
- Walked and went back to lying down
- Walked and sat down
- Was standing and went back to lying down
- Was sitting then started walking

4.2.8 Other Detections by the System

- Person has entered a Dark Room

CHAPTER 5 - Limitations

5.1 Posture Detection

The proposed system can only detect a set of defined postures which can be identified by the Sun SPOTs on the chest, thigh, leg and arm of a patient. It does not allow the flexibility of detecting free fall movements. To achieve these functionalities improvements can be made to the existing system by adding some more sensors on different body parts and detecting the relative change in acceleration values in X, Y and Z axis.

5.2 Range of Communication

Applications using wireless sensor networks can face issues with data packets loss and have limitations with range of communication.

Using Sun SPOT in WSN there can be limitations with communications due to data packets loss at increasing distance of communication between the basestation and the Sun SPOT.

Under perfect conditions the maximum range of communication between a basestation and a Sun SPOT is expected to be above 50meters [2]. But with the increase in number of SPOTS using radio communication and obstacles in the surroundings, the communication cannot be established at greater distances.

For the current application with 4 Sun SPOTS using radio communication the maximum range of communication between the basestation and the Sun SPOTS, with obstacles in surroundings will be approximately 15meters. This decrease in the range of communication is because of the excessive radio communication in the surroundings by other Sun SPOTS and partly because of the surrounding environmental conditions.

This distance may vary depending on the surrounding environmental conditions and also based on the output power of the Sun SPOT.

CHAPTER 6 - Related Work

Sun SPOT technology has also been used to develop numerous other healthcare applications.

A TwitterJacek[5] application, can monitor the physical activity and the health status of a person. The physical activity is measured using Sun SPOT's 3D accelerometer sensor and the health status is measured from ECG readings. The application uses a decision tree classifier to detect activities and the Sun SPOTs transmit the data to mobile devices through wireless communication.

There are various other physical activity monitoring systems developed using different sensors. Integration of Heterogeneous Motion Sensors and GPS in Healthcare Oriented Body Sensor[6] uses the triaxial accelerometer sensor in Telos motes. Apart from the movement monitoring the application also does location tracking through GPS.

Low-Cost Accelerometry-Based Posture Monitoring System for Stroke Survivors[8], uses 10 accelerometer sensors to detect the posture of a person and signal an alert when a bad posture is detected.

Apart from the physical activity monitoring applications the Sun SPOT technology has been used to implement a disaster recovery application.[7]

The Patient Monitoring System primarily focuses on the physical activity of a person by placing 4 Sun SPOTs on the patient. There is also a sensor placed on the chest to calculate the posture accurately and closely monitor the complete physical activity of a person. The Sun SPOT can be programmed in JAVA and does not require any underlying operating system to run the application unlike other motes like the Telos motes which uses the TinyOS operating system.

CHAPTER 7 - Conclusion

Using the 3D accelerometer sensor in Sun SPOT technology a patient monitoring system has been developed to monitor the physical activity of patients. The system can be used to detect the postures and high level actions of a person and can be used to remotely monitor the physical activity of a person. All the physical activity details are stored in a database and can be used as a feedback to improve the quality of medical programs.

7.1Application in Healthcare

The application can be used for various purposes in integration with existing healthcare systems. It can be used to remotely monitor the physical activity of person; this need is especially observed for aging people.

The application can also be used to evaluate and observe the effect of medication on patient's physical activity. For example, A Paralysis patient may be given some medication that might require observation of the reaction of medication on his physical activity.

Some medication requires mandatory physical activity to be performed by the patient and needs to be kept track of. For example, a patient's regimen might include walking daily for a certain period of time and doing physical exercises.

The application can also be used for surveillance purpose in hospitals.

7.2Future Scope for Improvement

The proposed system can only detect a set of defined postures which can be identified by the sensors on the Chest, Thigh, Leg and Arm movement of a patient. It does allow the flexibility of detected free fall movements.

The application can be improved to implement for people with physical disabilities.

References or Bibliography

Wireless Sensor Networks
[1] http://en.wikipedia.org/wiki/Wireless_sensor_network

Sun SPOT range of communication
[2] https://www.sunspotworld.com/forums/viewtopic.php?f=38&t=1580&start=0

[3] Image source: Download from
http://www.cromptons.co.uk/images/flash/flashalt.jpg

[4] Image source: Download from
http://www.cromptons.co.uk/images/flash/flashalt.jpg and
http://www.rwd.com/uploadedImages/Industries/Healthcare/Nurses walking.jpg

[5] TwitterJacket: An automated activity and health monitoring solution for the elderly, Shahriyar Amini, Priya Narasimhan, CyLab, Carnegie Mellon University

[6] Integration of Heterogeneous Motion Sensors and GPS in Healthcare Oriented Body Sensor Networks, Katherine Gilani

[7] ProSense Promote, Mobilize, Reinforce and Integrate Wireless Sensor Networking Research and Researchers: Towards Pervasive Networking of WBC and the EU

[8] Low-Cost Accelerometry-Based Posture Monitoring System for Stroke Survivors Sonia Arteaga, Jessica Chevalier, Andrew Coile, Andrew William Hill, Serdar Sali, Sangheeta Sudhakhrisnan, Sri Kurniawan

[9] SPOTWorld and the Sun SPOT, Randall B. Smith Sun Microsystems Labs

[10] Sun SPOT

http://en.wikipedia.org/wiki/Sun_SPOT

[11] Sun SPOT Software

http://www.Sunspotworld.com/GettingStarted/Vista.html

[12] Sun SPOT Accelerometer

http://www.Sunspotworld.com/docs/AppNotes/AccelerometerAppNote.pdf

[13] Sun SPOT System
http://research.Sun.com/spotlight/SunSPOTSJune30.pdf

[14] Sun SPOT Development forums
https://www.Sunspotworld.com/forums

[15] Sun SPOT Developer's guide
http://www.Sunspotworld.com/docs/Purple/spot-developers-guide.pdf

[16] Sun SPOT Hardware
http://blogs.sun.com/roger/entry/new_sun_spot_hardware_open

[17] Java
http://www.java.com/en/

[18] Java Swings
http://java.sun.com/docs/books/tutorial/uiswing/index.html

[19] MySQL
http://www.mysql.com/

[20] Netbeans IDE
http://netbeans.org/

[21] Sun SPOT light sensor
http://kenai.com/projects/eduni/pages/Tutorial101

[22] Image Source: Download from
http://www.jcrinc.com/Common/Images/custom/Pubs/Case-Study-1-4-Photo-1-.jpg

[23] Image Source: Download from
http://cache3.asset-
cache.net/xc/82497798.jpg?v=1&c=IWSAsset&k=2&d=B5384F3B2A5A9842B39017900AFB4
B028D57DA5FB773D4BDC6E55384B9338FFEE30A760B0D811297

[24] Image Source: Download from
http://www.dealoverflow.com/wp-content/uploads/2008/01/toshiba-satellite-a205-s5814-154-
widescreen-laptop.jpg