SSR (Smart Sleep Recognizer) : Making Decision Based on Probability of User's Intension

Dong-Wook Lee*, Soung-Hun You**, We-Duke Cho**, and Jai-Hoon Kim*

*Graduate School of Information & Communication, Ajou University, Korea,
{dwlee, jaikim}@ajou.ac.kr
** Center of Excellence for Ubiquitous System, Ajou University, Korea,
{sy05804, chowd}@ajou.ac.kr

ABSTRACT

In Ubiquitous environment, the interaction between human and computing devices are minimized or removed. The Smart Sleep Recognizer (SSR) defines current situation based on the probability of human's intention without any direct interaction with computing devices. In this paper, the SSR shows the one of the applicable field that defines the human's sleeping situation based on the sensed data. There are many researches on sleeping of human by implementing REM (Rapid Eye movement) or EEG (Electroencephalogram) technologies. However, these technologies require special devices to make a correct decision. The main advantage of the SSR over these existing technologies is that the SSR does not need any device attached to human body to get information. Instead of using attached devices, the SSR gets current information through deployed sensor network. The SSR analyzes the human's behavior by considering the location, time, movement, or any other items that might affect human's sleep and infers the situations by implementing Bayesian Network (BN). In this paper we shall see how these probability based application is contributed to develop Ubiquitous applications through SSR that does not require direct interactions between devices and human.

Keywords: Ubiquitous System, Sensor Network, Bayesian Network, Sleep Recognizer.

1. INTRODUCTION

The capability of computing has been growing faster than anybody expected and nobody can imagine the human life without computers. The initial purpose of developing computer is simply to help calculation. However, to get help from computers, the human should find out the location of computing devices, approach to the computers and give commands through interaction with computers. This traditional process has been changed since Mark Weiser explained concepts of Ubiquitous computing [1]. Given his explanation or description of Ubiquitous computing, it appears likely to us that the human does not have to figure out where the computers are, what they can do, or how to use them. In counterpoint to the conventional computing, the computers figure out where human is, or what the situation is, and what kind of services they have to provide with minimized or even without interactions in this new computing. In a nut shell, the main purpose of ubiquitous computing is to provide required services before human's demanding. Doubtless, this new paradigm is not far from today's computing technology if we take into account current computing power or network infrastructure for communication between devices. Moreover, this paradigm is already deployed in real world such as home networking or managing the market products area by utilizing sensed data. To provide the appropriate services, the computers or devices should define current situation by gathering contexts or evidences of situation. Schliit [2], Brown [3] and Ryan [4] simply define the context as location, temperature, time but Dey [5] adds emotion or intend of human. However, later, Dey [6] defines broader concept of context as "*Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.*" There are many applications by utilizing the context or situation aware such as Call Forwarding [7], Teleporting [8], Active Map [9], and Shopping Assistant [10]. After defining the situation with found context information, the devices or computers should be configured by itself to provide solution for the found situation. To provide the better services they should be able to have Self-Configurable, Self-Optimizing, Self-Protecting, and Self-Healing capabilities. In addition, the devices or members of service provider should grow intelligently by learning the process or correcting their mistakes without human's contact or help. The SSR adopts these procedures. Initially, the SSR gathers the current situation for defining goal. After defining the goal, the SSR make its own decision based on the probability of human's intention. Finally, the SSR intelligently recognizes whether its decision is correct or not. Whenever it makes decision, the SSR improves its performance by modifying the current information which the SSR has. This paper is organized as follows. The basic concept is explained at section 2. The brief explanation of algorithm and implementation are presented at following section. The section 4 analyses the SSR by implement the real sensors in our test-bed. Future or additional works are stated at section 5 and we will conclude about this paper at section 6.

2. BACKGROUND

There are many researches working on analyzing human's sleep by utilizing REM (Rapid Eye Movement) or EEG (electroencephalogram) technologies. With these technologies, researchers can figure out if human is in what stage of sleeping situation. However, these researches require attaching special devices to human body to get correct information. To provide the best environment for sleeping in Ubiquitous paradigm these technologies are not appropriate. In ubiquitous environment, the computing devices should make their own decision if human is in sleep or not without any contact. However, there is no serious study in making decision of human's sleeping without using attached devices. With consideration of Ubiquitous environment, there should be new technology to define human's current state and provide the best services based on the found state. Notably, the SSR does not require any special devices that attached to human body to define the current situation. The SSR is probability based application and these values are simply associated with human's contexts such as location, movement, fatigue, or age. To find the sleeping probability, the SSR implements Bayesian Network (BN) and assigns current human's state as *Sleep*, *Sleepy*, and *Awake*. Admittedly, there are many reasons that might cause the sleep and many results of sleeping. Figure 1 shows this relationship of sleeping in BN. It is needless to say that time, temperature, light, humidity might be considered as cause of sleeping. In addition, location, snoring sound, and movement might be described as result. If we can define this relation in detail then the final decision of SSR will be more accurate. The basic concept of SSR is that observe human's behaviors or contexts then applies the BN to predicts human's intention such as *Sleep, Sleepy, or Awake* as probability. If the probability of sleep is highest among three states then human is intending to sleep. However, if the human wants to get up from sleeping then the probability of *Awake* should be highest among the states. Based on this probability the SSR deduces or infers the current human's intention and sends this result to service provider. For self-making decision procedure in ubiquitous environment this probability based application might give an excellent direction to work with.

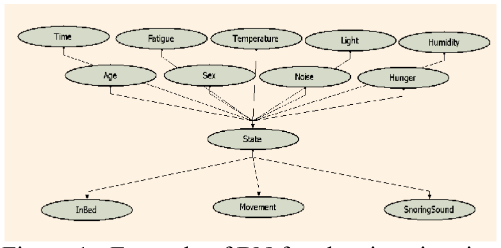

Figure 1 : Example of BN for sleeping situation

3. BASIC ALGORITHM OF THE SSR

3.1 Basic Configuration & Probability Table

Figure 2 shows the simplified version of SSR. This version of SSR considers just one cause of sleeping which is time and two results of sleeping, location and movement. Table 1 to 4 shows the probability tables of the BN in simplified SSR. Since the time is distributed equally for each hour the probability of each time should be the same (1/24 = 0.041). Table 2 shows the probability of *Sleep, Sleepy,* and *Awake* for each hour. The values of this table are based on research at national institute of health [11]. Even though these values are from the survey the sleep probability of each person is different. In other words, when the SSR is deployed there will be many false decisions initially but the SSR will modify these values whenever it figures that the probability of sleep is different than user's actual living pattern. For example, if SSR makes right decision then the probability should be increased but if it makes false decision then the probability is decreased for better result later. Finally, the SSR will find correct probability for user to make a right decision after modifying the values of table. The period of finding correct probability depends on the behavior of user. If the user's life style forms fixed pattern then this period should be short but if not then it would take longer time. The advantage of SSR is that modification of this value is made autonomously without any interaction with human. Since the SSR thinks intelligently, it makes its own decision whether the modification is needed or not without humans help. The table 3 and 4 also shows the conditional probability table (CPT) of in bed and in movement. The tables have the arbitrary numbers for implementation and also these numbers are required to be precise for accurate results.

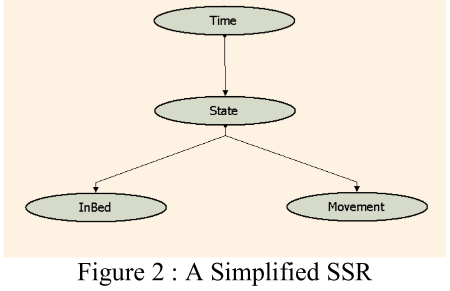

Figure 2 : A Simplified SSR

Table 1 : Probability of Time

Time Node			
Time	01~02	...	23~24
Probability	0.041	...	0.041

Table 2 : Probability of Sleep for each hour

State Node			
Parent Node	Probability		
Time	Sleep	Sleepy	Awake
23~06	0.875	0.063	0.062
06~07	0.750	0.192	0.058
07~08	0.580	0.214	0.206
08~09	0.430	0.214	0.356
09~10	0.225	0.093	0.682
10~11	0.180	0.098	0.722
11~12	0.225	0.093	0.682
12~13	0.312	0.172	0.516
13~14	0.350	0.300	0.350
14~15	0.430	0.216	0.353
15~16	0.475	0.199	0.326
16~17	0.375	0.156	0.469
17~18	0.310	0.172	0.518
18~19	0.180	0.090	0.730
19~20	0.230	0.084	0.686
20~21	0.375	0.237	0.388
21~22	0.625	0.192	0.183
22~23	0.837	0.125	0.037

Table 3 : Probability in bed when sleep

InBed Node		
Parent Node	Probability	
State	In Bed	Out of Bed
Sleep	0.8	0.2
Sleepy	0.5	0.5
Awake	0.2	0.8

Table 4 : Probability in move when sleep

Movement Node		
Parent Node	Probability	
State	Move	No Move
Sleep	0.2	0.8
Sleepy	0.5	0.5
Awake	0.8	0.2

3.2 State Diagram

Figure 3 shows the state diagram of the SSR. In this figure, there are three main states such as *Sleep, Sleepy, Awake* and *intermediate states* between them. The state is changed when the event is occurred and probability is changed. There are many events might be happened and the simplified SSR can not react for every events currently. Suppose the user is moving around at out side of bed at day time. Obviously, the SSR thinks that user is awake in this case because the probability of *Awake* is highest among three states. When the user is moving to the bed and lies abed then the probability of *Awake* is decreased and that of *Sleep* is increased. In this case, the SSR infers that the user intends to get sleep. Since there is event and the probability has been changed the SSR should react for this event. Initially the state was *Awake* in this case but the state should transit to *Sleepy*. However, the SSR does not directly go to the *Sleepy* state. The SSR stays at *intermediate* state between *Sleepy* and *Awake* then this will last for a while for any additional events. The SSR calls this time as *wTime* (waiting Time). If the probability of *Sleep* is considerably high then the *wTime* is relatively short for fast reaction.

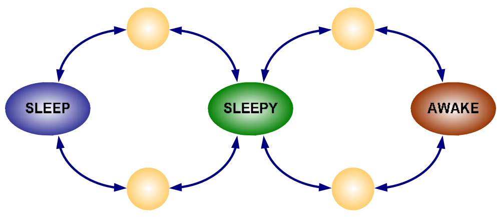

Figure 3 : State Diagram of SSR

However, the *wTime* should be long enough for waiting any event even though the *Sleep* probability is increased but the probability of *Sleep* is not high enough. As a consequence, the SSR expects additional events with lower probability for longer time before going to next state such as *Sleepy* or *Sleep* than higher probability. Clearly the *wTime* depends on the found probability of sleeping by BN. Equation 1 shows how the SSR finds the *wTime*. In this equation, $wTime_0$ refers to the default value of *wTime, r* is for the increasing rate of *wTime*, and P_x means highest probability among three states after event is happened.

$$wTime = r/P_x + wTime_0 \qquad (1)$$

Suppose there are two events. Firstly user is moving around at out side of bed and secondly user is not moving while he or she stays at bed. According to this situation, the second event should return higher probability of sleep. Therefore the *wTime* of second event should be shorter than that of first case for fast reaction. The *wTime* might be chosen by applying different increasing rate *r*. Figure 4 shows the different value of *wTime* when *r* is 10, 20 and 50. According to this figure, the SSR does not wait for long time if the probability is high. However, if the found probability of event is low then the SSR waits additional events for longer time to make a careful decision. For example, if the state of user was *Awake* and he or she moves to bed, then the sleep probability should be increased. If the sleep probability is 40 % then the SSR waits additional events for 30 seconds when *r* is 10 but the SSR would wait for just 16 second if probability of sleep is 90% before sending any message to the service provider. As a result, this procedure returns fast transient for higher probability of user's intension.

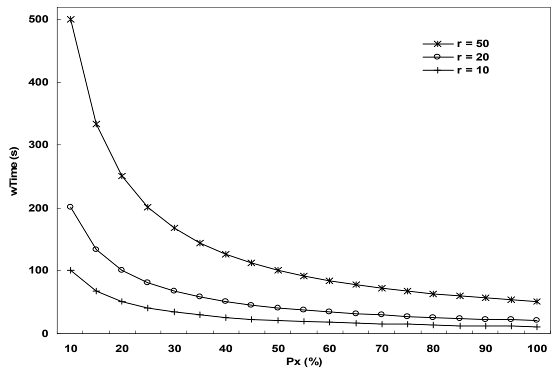

Figure 4 : The values of *wTime* for different value of *r*

3.3 Gathering Data

To decide current state, the SSR needs information about users. This simplified version of SSR only considers location, movement, and time. The SSR can figure out the location of user easily by deploying sensors that senses the current location of user. To find movement of user, the SSR simply compares previous and current location for each second. If the change of location exceeds predefine value or threshold then the SSR thinks that there is movement of user and consider new event is happened. With this simplified SSR, the threshold of this movement is defined as $\sqrt{((\textit{Difference of x's location})^2 + (\textit{Difference of y's location})^2))}$. Simply put, if this threshold is small then the SSR reacts with smaller movement of user. Whenever there is any event then the BN is applied and new probability of user's intend is calculated. Based on this found probability the state will be changed among three states. Consequently, simplified version of SSR considers location, time, and movement of user as contexts to define current situation.

3.4 Finding Wrong Decision

As described earlier, the SSR would make a wrong decision. To figure out this situation the SSR should watch for user's behavior for a while when the state is in *Sleep*. If the user really intends to get sleep then there will be no additional events such as movement or going out of bed for specific time. However, if the user is in bed but does not want to get sleep, for example just simply wants to read a book then the SSR probably makes a wrong decision. The SSR would think the user wanted to get sleep because user is in bed and there is no movement for a while. Since it is *Sleep* state, the light might be off by service provider to provide *Sleep* environment. This is wrong decision. The user might move to turn on the light as soon as lights are off. In this case, the SSR figures out it made a wrong decision because there is event as soon as it made *Sleep* decision. That is to say, to figure out whether it made a wrong decision or not, the SSR watches user's behavior for a while. In *Sleep* state the SSR is waiting for additional events for specific time. This time should be associated with the probability of sleep. If the probability of sleep is high then this time should be longer because the user is really want to get some sleep. However, if the probability of sleep is low then the watching time should be short for weak intend of sleep. For example, the SSR might watch the user for 5 minutes for sleep probability of 90% but might watch user just for 1 minute if the sleep probability is 10 % since with low sleep probability, the user might get up easily. So the SSR simply apply leaner relationship between probability and watching time. Equation 2 shows this relation where w is watching time, s means increasing rate, and Px is simply sleep probability

$$w = s \times P_x(state) \qquad (2)$$

As a result, if there is new event in time w then the SSR thinks that it made wrong decision. However, even though there are events so the state should be move to *Sleepy* from *Sleep* after time w, the SSR defines this is normal situation and does not think it made a mistake.

3.5 Correcting Value in Probability Table

The SSR is probability based decision making algorithm. Therefore the value of probability in BN plays critical role in this algorithm. Initially, there are default values but those do not work well with every person initially. This value should be changed whenever the SSR makes decision whether it is right or wrong. If the SSR made false sleep decision then the probability should be decreased. With decreasing of probability, the *wTime* would be increased and w will be decreased when the same event is happened later. So the SSR will generate more slow reaction than previous. Exponential increasing and decreasing are applied for modifying the probability table depends on the number of consecutive wrong or right decision.

$$\text{Prob}(state, k) = \text{Prob}(state) \pm c \times k^2 \qquad (3)$$

Equation 3 shows the concept of increasing and decreasing of new probability. *Prob(state, k)* is modified value after k consecutive errors. *Prob(state)* is previous probability or value of probability in BN. c is changing rate and k is the number of consecutive wrong or right decisions. According to this equation the first wrong or right decision dose not affect the current probability much. However, if there are multiple or consecutive wrong or right decisions then the changing rate of probability will be exponentially high. Suppose the default probability of sleep is 90% and actual sleep probability of user is 50% then the figure 5 shows how to find this probability by applying this concept when $c = 2$.

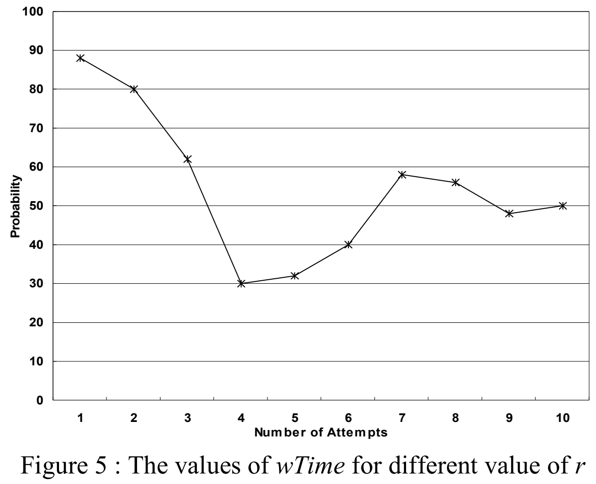

Figure 5 : The values of *wTime* for different value of *r*

Even though the SSR can't find the exact value of probability, the SSR can stop applying this modification if the value of found probability is within specific margin of actual probability. Table 5 shows the process of finding this probability.

Table 5 : Process of finding actual probability of sleep

# of Attempts	Applying Eq. 3.	Probability (%)
1	$90-2\times1^2$	88
2	$88-2\times2^2$	80
3	$80-2\times3^2$	62
4	$62-2\times4^2$	30
5	$30+2\times1^2$	32
6	$32+2\times2^2$	40
7	$40+2\times3^2$	58
8	$58-2\times1^2$	56
9	$56-2\times2^2$	48
10	$48+2\times1^2$	50

As a result, the SSR would find if it makes wrong or right decision and learn what it has to do. To provide appropriate service for user the SSR is growing by itself by modifying the probability of sleep in BN.

4. EVALUATION

We implemented this application on three different scenarios that might be happened in normal life. This evaluation shows how this application work well and how they are growing to provide better services. We deployed this application in our test-bed for real implementation. The SSR receives x, y values from sensors and make decision whether the user is in sleep or not. After making decision, the SSR sends the state information to main server called uMIS (ubiquitous main Information Server). When the uMIS receives the message then the commands for appropriate actions are transfer to DAS (Digital Appliance Server) to provide the best conditions for sleep environment. Table 6 shows the services for each state. For working with this implementation, we set up the basic values for probability table of BN as described earlier and the r is 20 and $wTime_0$ is 5 for *wTime*.

4.1 Goes to Sleep

Description: *The user goes to bed and tries to get sleep at 11 pm.*

In this simulation, for fast result we wanted to have the wTime is about 25 seconds. According to this case, the initial state of SSR is *Awake* since user is out of bed and keeps moving. When the user went to bed but there is still movement then the probability of sleep is increased. The state changes from *Awake* to *Sleepy*. In this case the *wTime* is 27 seconds and SSR is waiting for other events such as moving out of bed. After 27 seconds the SSR thinks that the user intends to get sleep and transients the state to *Sleepy*. At this time the SSR is waiting for another 27 seconds to wait other events. Suppose the user is fall in sleep and there is no more movement. Since this is new event the SSR recalculates another probability for sleep. That's 96%. The state changes to *intermediate state* between *Sleepy* and *Sleep* then waits for another event for 24 seconds. In this simulation, there are no events occurred so after 24 seconds the SSR defines current situation as *Sleep*. However, if user got up when the state is in *intermediate* state then the SSR predicts that user is not willing to get sleep. So the state moves to *Awake* state.

Table 6 : Services for each situation

Event	Playing Music	Light	Curtain	Gas Valve	Fan
Awake $\rightarrow$ Sleepy	10s	50%	-	-	On
Sleepy $\rightarrow$ Sleep	-	0%	Closed	Closed	On
Sleep $\rightarrow$ Sleepy	-	50%	-	Closed	On
Sleepy $\rightarrow$ Awake	10s	100%	Open	-	Off

Table 7 : Procedure for scenario 1

	Event		Current	wTime	Probability (%)		
	Bed	Move			Sleep	Sleepy	Awake
1	No	Yes	Awake	-	38	17	45
2	Yes	Yes	Awake $\rightarrow$ Sleepy	27	84	10	6
3	Yes	Yes	Sleepy	27	84	10	6
4	Yes	No	Sleepy $\rightarrow$ Sleep	24	96	3	1
5	Yes	No	Sleep	-	96	3	1

Table 8 : Procedure for scenario 2

	Event		Current	wTime	Probability (%)		
	Bed	Move			Sleep	Sleepy	Awake
1	Yes	No	Sleep	-	85	12	3
2	Yes	Yes	Sleep $\rightarrow$ Sleepy	17	51	29	18
3	Yes	Yes	Sleepy	6	51	29	18
4	No	Yes	Sleepy $\rightarrow$ Awake	5	11	25	63

Table 9 : Procedure for scenario 3

	Event		Current	wTime	Probability (%)		
	Bed	Move			Sleep	Sleepy	Awake
1	No	Yes	Awake	-	5	18	77
2	Yes	No	Awake $\rightarrow$ Sleepy	4	76	19	5
3	Yes	No	Sleepy	4	76	19	5
4	Yes	No	Sleepy $\rightarrow$ Sleep	4	76	19	5
5	Yes	No	Sleep	-	76	19	5
6	No	Yes	Sleep $\rightarrow$ Awake	-	4	18	78

4.2 Getting Up

Description: *User is getting up 7 am in the morning*
In this scenario, the user is getting up at 7 o'clock in the morning. Initially, the user was sleeping and there is no movement. Obviously, it is *Sleep* state. However, suppose there was little movement while user is in bed. The SSR senses this context information and recalculates the probability of *Awake*. Since the *Awake* probability is increased the direction of state is heading to *Awake*. Therefore the state moves to *intermediate* state between *Sleep* and *Sleepy* then waits for 17 seconds. After 17 seconds, the state moves to the *Sleepy* state and waits for another event. If there is no event then the state is moving back to the *Sleep* because the probability of *Sleep* is highest. However, if the user moves out to the bed then the probability of *Awake* is increased again. Then the SSR believes that user intend to wake up. And state should transient to awake after 5 seconds.

4.3 Wrong Decision

Description: *The SSR defines the user tries to get sleep but user does not intent to sleep*
Suppose the user goes to bed at 8pm and the movement has been disappeared. The SSR thinks that the user is trying to get sleep. So the state is moving to the *Sleep* state and waiting for 228 seconds. Suppose the user is moving out of bed before 228 seconds are elapsed. In this case, the SSR thinks it made wrong decision. So the state goes to the *Awake* state immediately without going through the intermediate state. When anyone compares the probability of 1 and 5 the probability has been decreased slightly. Since this is the first wrong decision, the changing rate is not much but it the consecutive numbers of wrong decision then the decreasing of probability will be increased exponentially.

5. CONCLUSION

In ubiquitous environment, the interfaces between human and computing devices should be minimized or removed. Since there are limited interactions in this environment, the computing devices should infer or predict the user's intention to provide the best service. The probability based situation-aware application is the one of the fine solution in this environment. As we have seen, the SSR shows how the probability based situation-aware application works well without any direct interactions between human and computing devices. Even with small number of contexts, the SSR can make correct decision and grow intelligently. To summarize, the probability based situation-aware application can give concrete solution for inferring human's intention without interactions in Ubiquitous computing environment.

6. ACKNOWLEDGEMENT

This research is supported by the ubiquitous Autonomic Computing and Network Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

REFERENCES

[1] Mark Weiser, "The Computer for the Twenty-First Century," *Scientific American*, pp. 94-104, September 1991
[2] Schilit, B., Theimer, M. Disseminating, "Active Map Information to Mobile Hosts," IEEE Network, 22-32, 1994
[3] Brown, P.J., "The Stick-e Document: a Framework for Creating Context-Aware Applications," Electronic Publishing, 259-172, 1997
[4] Ryan, N., Pascoe, J., Morse, D. "Enhanced Reality Field: the Context-Aware Archaeological Assistant." Gaffney, V., van Leusen, M., Exxon, S. (eds.) "Computer Applications in Archaeology" 1997
[5] Dey, A.K. Context-Aware Computing: The CyberDesk Project. AAAI 1998 Spring Symposium on Intelligent Environments, Technical Report SS-98-02 51-54 1998
[6] Anind K. Dey, Gregory D. Abowd, "Towards a Better Understanding of Context and Context-Awareness," Proc. Of the CHI 2000 Workshop on The What, Who, Where, and How of Context-Awareness, 2004
[7] Roy Want, Andy Hopper, Veronica Falcao, Jonathan Gibbons, "The Active Badge Location System," ACM Transactions on Information System, 91-102, 1992
[8] Frazer Bennett, Tristan Richardson, Andy Harter, "Teleporting – making applications mobile," In proceeding of IEEE Workshop on Mobile Computing Systems and Applications, 82-84, 1994
[9] Roy Want, Bill N. Schilit, Norman I. Adams, Rich Gold, Karin Petersen, David Goldberg, John R. Ellis, Mark Weiser, "An Overview of the PARCTAB Ubiquitous Computing Experiment," IEEE Personal Communications, 28-43, 1995
[10] Abhaya Asthana, Mark Cravatts, Paul Krzyanowski, "An Indoor Wireless System for Personalized Shopping Assistance," In Proceedings of IEEE Workshop on Mobile Computing Systems and Applications, 69-74, 1994
[11] http://science.education.nih.gov/supplements/nih3/sleep/guide/lesson3.htm