Experience-based requirements on data sets for user modeling in the home

Johan Hjelm
Ericsson Research
Nippon Ericsson KK
Tokyo, Japan
johan.hjelm@ericsson.com

Toshikane Oda
Ericsson Research
Nippon Ericsson KK
Tokyo, Japan
toshikane.oda@ericsson.com

Sitorius Timothy Lawrence
Morikawa Laboratory, RCAST
The University of Tokyo
Tokyo, Japan
tsitora@mlab.t.u-tokyo.ac.jp

Hua Si
Morikawa Laboratory, RCAST
The University of Tokyo
Tokyo, Japan
sihua@mlab.t.u-tokyo.ac.jp

Hiroyuki Morikawa
Morikawa Laboratory, RCAST
The University of Tokyo
Tokyo, Japan
mori@mlab.t.u-tokyo.ac.jp

Shunsuke Saurwatari
Morikawa Laboratory, RCAST
The University of Tokyo
Tokyo, Japan
saru@mlab.t.u-tokyo.ac.jp

ABSTRACT
In this paper we describe some of our experiences in creating a user modeling system based on sensor data collection, and the further experiences we have gained in integrating it with a network of consumer devices. We discuss the implications in terms of modeling and the requirements in terms of data formats this imposes. Finally, we provide some requirements in terms of data aggregation and statistics derived from raw data sets, and a stress the fact that the intentions of users need to be mapped to the sensed data, and recorded in an identifiable way.

Author Keywords
Sensor data, analysis, user models, user intentions

ACM Classification Keywords
TBA.

INTRODUCTION
We have built a context-aware service platform called Synapse, which learns users' activity and provides services according to the learning. The Synapse uses Bayesian Networks to build the model of users, and predicts the most relevant services that users will use in the current situation based on the user model. The Synapse is currently under construction at the Morikawa Laboratories in Tokyo University, which logs sensor measurements on a continuous basis as a user interacts with a simulated home environment [1]. To enhance the realism of this system, we are now integrating a system to log user interactions with media resources, assuming these to be performed on DLNA devices. For this, we are using UPnP [3], the technique underpinning DLNA [4].

In the process of developing the Synapse system, and during the integration with UPnP, we have made some learnings about the optimal design of measurement datasets, which we hope to be able to share with the participants in the workshop.

THE SYNAPSE SYSTEM
The Synapse consists of four parts: 1) the sensor event collection part that captures real worlds information, 2) the service control part that provides services, and 3) the Synapse Core. Architecture of Synapse is shown in Figure1.

CHI 2009, April 4–9, 2009, Boston, MA, USA.

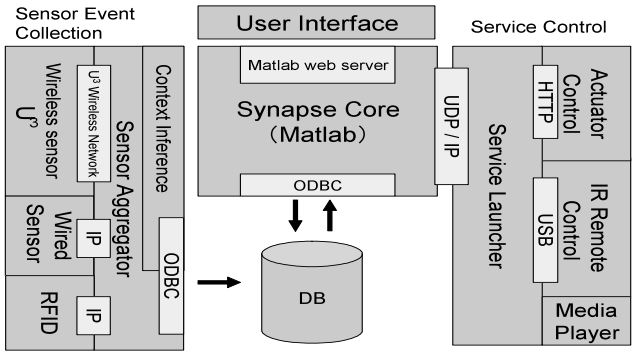

Figure 1: Architecture of Synapse.

Sensor Event Collection Part

The sensor event collection part captures real world information from various sensors, converts raw data into useful context (we call them "sensor events" in Synapse), and records these sensor events in database. Sensor Aggregator fuses the raw sensor data and reduces the noise. For instance, the average temperature in a room is fused from different temperature sensors. Context Inference extracts complex events such as "the user is sleeping" from simple events. On this test bed, 4 kinds of sensors are used simple events. On this test bed, 4 kinds of sensors are used to produce 11 events: RFID is used to identify users, U3 wireless sensor nodes are used to capture the temperature, brightness and human motion, a contact detector detects whether the phone is in use, and an e-calendar detects a day of the week.

All the sensor events are recorded as a time series $\{E_1, E_2 \ldots\}$ in database. Each sensor event is recorded as E (EN, EV, ET), which respectively represents the event ID, the event value and the time at which this event is recorded. We predefine a set of events $\{e_1, e_2 \ldots e_N\}$ (such as e_1 means "temperature", e_2 means "brightness"), and EN $\in \{e_1, e_2 \ldots e_N\}$. Many context inference schemes can be used to recognize events and their values from raw sensor data [10]. However, since event values are generated from different types of sensors (e.g. the temperature is 25°C, and the humidity is 60%), and it is difficult for a general core to process all types of values, we use fuzzy sets [22] approaches to unify all the event values between 0 and 1 as in [10], which means EV $\in$ [0, 1]. Basically, an event will be recorded when the value changes. However, in many scenarios, it is not necessary to record events as frequently as they change, so we can add some requirements to event recording. Events will not be recorded, until they satisfy these requirements. (e.g. one requirement is "e_1 is over 0.7", so e_1 will not be recorded until it is over 0.7.)

Service Control Part

The service control part controls various devices to supply services. All the services are recorded as a time series $\{S_1, S_2 \ldots\}$ in database. Each service is recorded as S (SN, ST), which respectively means the service ID, and the time at which this service is recorded. We predefine a set of services $\{s_1, s_2 \ldots s_M\}$ (such as s_1 means "turn on light", s_2 means "mute music"), and SN $\in \{s_1, s_2 \ldots s_M\}$.

Synapse Core

We apply HMM to model the relationship between the sensor events and the services. Figure 2 shows one cycle of Synapse model. There are two basic components in HMM: the hidden state X_t and the observation of state Y_t. In Synapse, each hidden state X_t corresponds to a service S_t (*not lowercase s*), to indicate the situation in which this service is used, and the observation Y_t is a vector of event values $(y_1, y_2 \ldots y_N)$, which are the current values of $\{e_1, e_2 \ldots e_N\}$. There are three parameters in HMM: the prior probability $\pi(i)=P(X_1=i)$ which represents the initial state, the transition matrix $A(i,j)=P(X_t=j|X_{t-1}=i)$ which represents the probability of transfer from $X_{t-1}=i$ to $X_t=j$, and the observation model $P(Y_t|X_t)$ which represents the relation of X_t and Y_t [16]. The learned results in one cycle are used as the initial estimations of the next cycle.

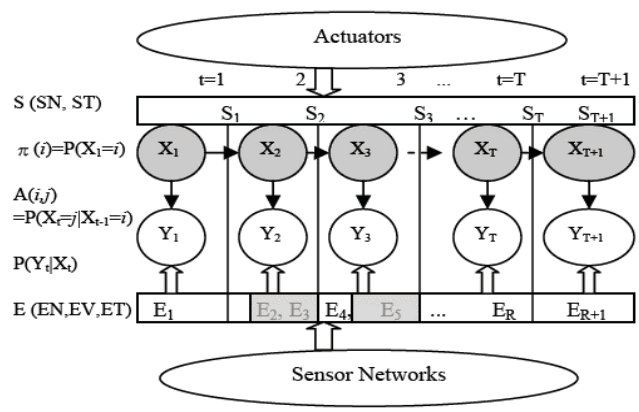

Figure 2: One Cycle of Synapse Model. The grey rectangles indicate a certain interval before S_t.

In the Learning Phase ($1 \le t \le T$), Synapse uses the history records of sensor events and services that happened during t=1, 2…T to compute $A(i,j)=P(X_t=j|X_{t-1}=i)$ and $P(Y_t|X_t)$. We assume that sensor events, which happened in a certain interval before a service, indicate the situation in which this service is used. For instance, in Figure 2, E_2 and E_3 indicate the situation in which S_2 is used. Since X_t cannot be observed directly, we firstly use the forwards-backwards algorithm [17] to infer $X_{1:T}$ from the observation $Y_{1:T}$. In the forwards pass, we recursively compute the filtered estimate $\alpha_t=P(X_t=i|Y_{1:t})$, and in the backwards pass, we recursively compute $\beta_t=P(Y_{t+1:T}|X_t=i)$; then combine them to produce the smoothed estimate $\gamma_t(i)=P(X_t=i|Y_{1:T})$ and the smoothed two-slice estimate $\xi_{t-1,t}|T(i,j)=P(X_{t-1}=i,X_t=j|Y_{1:T})$. After that, we use EM (expectation maximization) algorithm [17] to learn $A(i,j)=P(X_t=j|X_{t-1}=i)$ and $P(Y_t|X_t)$ from $\gamma_t(i)$ and $\xi_{t-1,t}|T(i,j)$.

In the Executing Phase (t>T), Synapse uses the learned transition matrix $A(i,j)=P(X_t=j|X_{t-1}=i)$, observation model $P(Y_t|X_t)$ and the current observation Y_t to compute the occurrence probability of each service. A two-step filtering algorithm is applied: in update step, the probabilities of

Identify applicable sponsor/s here. *(sponsors)*

current state can be gained as we compute $P(X_t|Y_t)$; in predict step, the probabilities of next state can be predicted as we compute $P(X_{t+1}|Y_t)$. As a result, the occurrence probability of each service can be computed as the occurrence probability of each state corresponding to these services. After that, we can sort the services in a descending order of probability. If a probability is higher than a user-defined threshold, the corresponding service will automatically start.

INTEGRATION OF UPNP

A users behavior can only partly be measured by the users physical actions. Media consumption, such as listening to music, can be measured using sensors; but it is easier from a measurement point of view to retrieve this data from the multimedia devices used to consume media.

In a DLNA system, users request media through an (undefined) user interface, which may be equivalent to the user interface on an existing consumer electronics device, such as a television set. Media are, differently from interactions, characterized by the possibility to describe them using a large variety of parameters. For instance, a song can be characterized by the author of the text, the composer of the music, the singer performing the song, the band performing the music, the company owning the copyright, the mode it puts the user in, etc. Some of these parameters have been standardized (e.g. in Dublin Core [5]), and can be included or referenced when the user selects the song. The same goes for other types of media.

Metadata sets describing media are widely available, although not always in standardized formats, one example being the Internet Movie Database (IMDB)[6]. The use of such metadata sets in user modeling is a well researched area, although there is no unified or standardized solution.

USAGE OF THE MEASUREMENTS

User modeling is a matter not only of learning from data sets, but also in relating these data sets to actual behavior. For situated services, the measured data need to be associated with the users situation.

Typically, this type of data is used by recommender engines to select subsequent media which can be presented to the user for consumption (an example is "if you liked Celine Dion, you probably will like Engelbert Humperdinck"). The recommender systems are based on a user model which can either be derived from the user interactions with the data set or externally created. The use of UPnP in the Synapse system will allow us to create a user model by virtue of the users interactions outside the music system, and use it in such a recommender system, together with other user models.

APPLICABLE EXPERIENCES

User modeling is a matter not only of learning from data sets, but also in relating these data sets to actual behaviors, as well as the users intentions with these behaviors.

Precision of the used algorithms

We verified the correctness of prediction with 150 test samples: 30 of which are "Light" scenario test samples, 60 of which are "TV" scenario test samples (30 for Video, 30 for TV_1ch), and 60 of which are "Music" scenario test samples (30 for M_Mute, 30 for M_Loud). We only tested correctness of the first recommendation because of the definitude of result. (The correctness of top 5 predictions will be tested by real inhabitants in the future.) The correctness of prediction is shown on Table 1, which reveals that our methods are practically accurate enough for real life.

Table 1: Correctness of Prediction

Services	Light_On	Video	TV_1ch	M_Mute	M_Loud
Correct	96.7%	93.3%	90.0%	93.3%	90.0%

UPnP integration

The integration of UPnP is ongoing, and will be finalized when the workshop occurs. We will be happy to share our experiences at the workshop.

CONCLUSION

The collection of measurements of user activities is a great way of enabling machine learning, and leads to refinements of models, as we have already shown. However, to find out what the models relate to, and what the activity traces actually relate to, we must associate the data sets with information about the users intentions and activities during the test period. If that is done, the models can both be made more predictive, and the system more responsive.

ACKNOWLEDGMENT

We wish to acknowledge the contribution of our colleagues in Ericsson Research, in particular in the branches in Tokyo and Aachen, as well as our colleagues in Morikawa Labs.

REFERENCES

1. H. Si, Y. Kawahara, H. Morikawa, and T. Aoyama, A Stochastic Approach for Creating Context-Aware Services based on Context Histories in Smart Home, In Proceeding of the 3rd International Conference on Pervasive Computing (Pervasive2005), Exploiting Context Histories in Smart Environments (ECHISE2005), pp. 37-41,Munich,Germany, May 2005.
2. Zellweger, P.T., Bouvin, N.O., Jehøj, H., and Mackinlay, J.D. Fluid Annotations in an Open World. *Proc. Hypertext 2001*, ACM Press (2001), 9-18.
3. UPnP Forum website, http://www.upnp.org/
4. DLNA Consortium website, http://www.dlna.org/home
5. Dublin Core metadata set, http://www.dublincore.org/
6. IMDB, http://www.imdb.com/

Identify applicable sponsor/s here. *(sponsors)*

The columns on the last page should be of approximately equal length.