Learning Key Contexts of Use in the Wild for Driving Plastic User Interfaces Engineering

Vincent Ganneau[1, 2], Gaëlle Calvary[2], Rachel Demumieux[1]

[1] Orange Labs, 2 avenue Pierre Marzin, 22307 Lannion Cedex, France
{Vincent.Ganneau, Rachel.Demumieux}@orange-ftgroup.com
[2] Laboratoire LIG, 385 rue de la Bibliothèque, BP 53, 38041 Grenoble Cedex, France
{Vincent.Ganneau, Gaelle.Calvary}@imag.fr

Abstract. This paper addresses software plasticity, i.e. the ability of interactive systems to adapt to context of use while preserving user-centered properties. In plasticity, a classical approach consists in concentrating design efforts on a set of pre-defined contexts of use that deserve high quality User Interfaces (UIs), and switching from one to another according to variations of context of use at runtime. However, key contexts of use cannot be finely envisioned at design time, especially when dealing with the specific field of mobility. Thus, we propose a designer's partner tool running on the end-user's mobile device to probe key contexts of use in the wild. The underlying principles are data gathering, bayesian learning, and clustering techniques. Probing key contexts of use can save design efforts.

Keywords: Mobility, plasticity, context of use, probing, bayesian network, learning, clustering.

1 Introduction

In ubiquitous computing [11], context-aware adaptation has been widely investigated to cope with the increasing number of platforms, users, and environments, i.e. the diversity of contexts of use. This paper addresses the notion of *plasticity*, i.e. the ability of interactive systems to withstand variations of context of use while preserving user-centered properties [10]. In plasticity, most of the works so far make the implicit hypothesis that the contexts of use to be considered are identified at design time. In practice, this is far from being easy. As known in human-computer interaction, laboratory tests make it possible to observe usability issues with the system [9] but are limited to understand usage and system's impacts in a very few envisioned contexts of use such as home, street, work, etc. [5]. In the specific field of mobility, the number of contexts of use is unpredictable. Limiting them to predefined rough ones may result in not fully meeting the user's expectations. As a result, there is a need for partner tools that help the designers in identifying the *key* contexts of use in the wild on mobile devices such as cell phones.

Recent works in the field of end-user development underline the need for monitoring the end-user's environment (task, place, time, etc.) in order to provide context-aware adaptivity [6]. In addition, experience shows that users tend to have distinct contexts of use when in mobility. In this paper, we propose a Windows Mobile embedded tool that collects objective data from user's actions in the wild and provides algorithms for learning key contexts of use from these observations. The process is based on bayesian user modeling and clustering techniques. The tool aims at separating relevant contexts of use from marginal situations by asking the end-user through a dedicated User Interface (UI). Such a probing task has to be taken in the early phases of the development process to save design efforts. Some frequent or critic contexts may require specific prototyping for ensuring high quality UIs.

2 EMMA: Embedded Manager for Mobile Adaptation

EMMA (Embedded Manager for Mobile Adaptation) is our running system for probing key contexts of use on Windows Mobile devices. EMMA relies on a user model that learns from user's actions gathered in mobility. The overall process is based on the functional decomposition given in Fig. 1. Three steps are identified. The system starts by collecting objective data from the observation of context of use and person-system interaction. Then these data are processed by the user model through learning algorithms. Finally, clustering techniques are performed to discover the best set of key contexts of use given knowledge inferred from the user model.

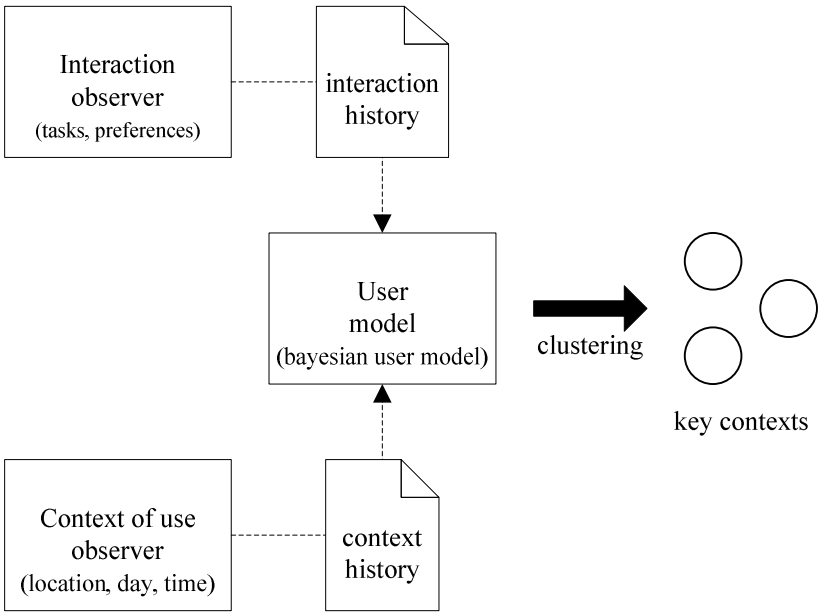

Fig. 1. Overall process for the identification of key contexts of use.

Key context identification may be placed under the end-user's control through a dedicated UI (Fig. 2) that helps in reinforcing system's perception as well as validating the correctness of data. When a new key context of use is detected, the end-user is put in the loop: he/she can customize system's propositions and set the name of the new context (Fig. 3).

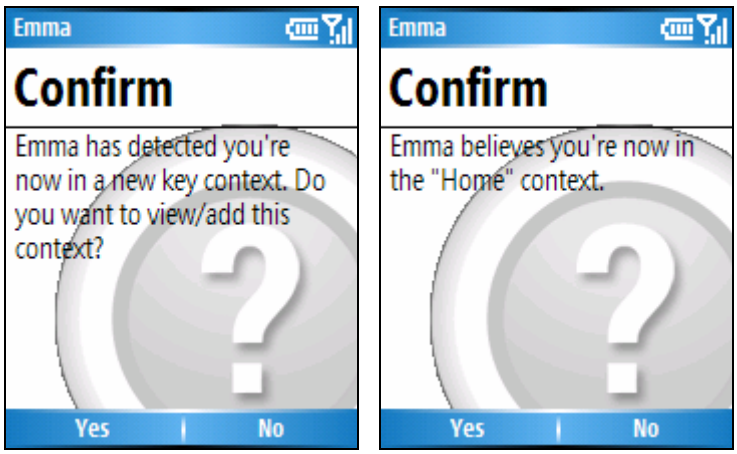

Fig. 2. Key context identification and change may be negotiated with the end-user.

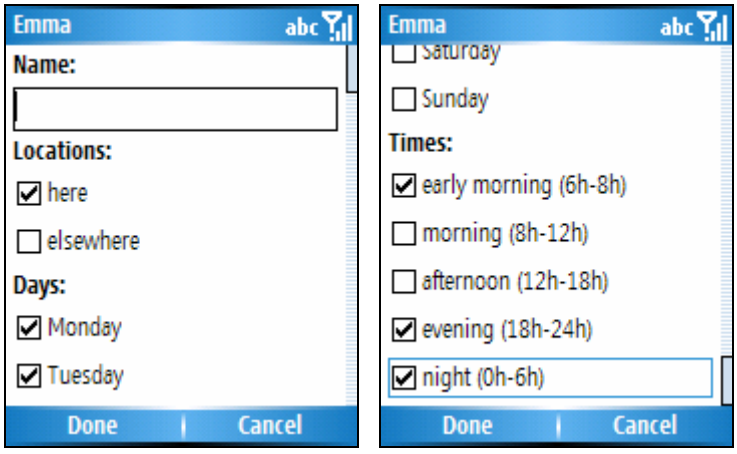

Fig. 3. When adding a new key context, system's propositions may be customized by the end-user. A key context is identified by its name.

3 Bayesian User Modeling

EMMA's user model is based on a bayesian network. Bayesian networks are graphical models that consist in both a qualitative and a quantitative part. The qualitative part is the structure of the network: a directed acyclic graph where vertices are variables and edges denote influences between variables. The quantitative part provides the Conditional Probabilities Tables (CPTs), i.e. the parameters of the network.

Bayesian networks are powerful tools provided with inference and learning algorithms. Inference relies on Bayes' theorem for propagating knowledge along the network. Learning applies for both the structure and the parameters of the network. It can be done from either complete or incomplete raw data. Bayesian networks are usually used for diagnosis, prediction, modeling, and monitoring. A key point is their ability to deal with incompleteness, which argues for their use when dealing with imperfect context information [3]. Bayesian user modeling has been investigated in previous works [4]. On mobile phones, bayesian learning has been used to discover when and how a user changes his/her profile over time [1].

3.1 Structure Building

In practice, bayesian models can be built from expert knowledge and/or automatically from data. As experts, we designed the structure of the user model for daily probing the user's behaviour when interacting with a mobile device in the wild (Fig. 4).

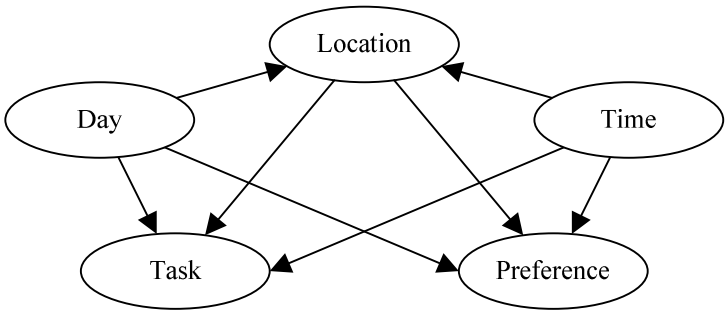

Fig. 4. EMMA's bayesian user model: the impact of context of use (day, time, location) on user's activity (tasks, preferences) is represented by causal links among nodes. User's location is time-context dependent.

In our model, we assumed that user's tasks and preferences may vary according to the context of use. We probed two kinds of context changes: changes in time (day of the week, time of the day) and space (location). Probing might easily be enlarged to additional information. In addition, we assumed that context changes might give rise to repetitive tasks (e.g. switching to silent mode when joining a meeting). Thus, each

node of the model was dedicated to a particular function. We distinguished two kinds of functions depending on whether the node was in charge of sensing and identifying the context of use (i.e. *Day*, *Time*, and *Location* nodes), or tracking user's tasks and preferences (i.e. *Task* and *Preference* nodes). Extending the prototype to other context changes and tasks would be simply done by adding new nodes and causal links in the network. The structure defines the format of the gathered data.

3.2 Data Gathering

Data gathering in mobility has been investigated in previous works [2]. As motivated above, we need to collect two kinds of data to be processed by the user model: context and interaction data. As discussed, time and space contribute to the context identification when processed by the user model. Day and time changes are probed through the corresponding system states. We parse time into five intervals as following: night (0h-6h), early morning (6h-8h), morning (8h-12h), afternoon (12h-18h), and evening (18h-24h). Alike in [7], we use the nearest GSM cell-tower to track changes in the user's location. At any time, the mobile phone is connected to a particular cell-tower unless the user is not in a mobile phone receiving area. Each cell-tower is identified by its Cell IDentifier (CID) and its Location Area Code (LAC). Both CID and LAC are integers. We use the Radio Interface Layer (RIL) provided with Windows Mobile powered devices to catch cell changes. We match each cell change with the day and time within which it occurs.

We probe two kinds of interactions on mobile phones (Fig. 4): applicative tasks – *Task* node – (e.g., messaging, calls, games, etc.) and customization tasks – *Preference* node – (e.g., phone's profile, look and feel, etc.). Every time the active application changes, the interaction observer reports the application the user is interacting with in the interaction history. Observations are matched with day, time, and location (see Table 1). User's preferences are gathered in the same way.

Table 1. Interaction history gathered in mobility.

Location	Day	Time	Application
CID40506LAC4354	Thursday	morning	Settings
CID40511LAC4354	Thursday	morning	Calendar
CID40511LAC4354	Thursday	morning	Contacts
CID58063LAC4354	Thursday	morning	Call History
CID40511LAC4354	Thursday	afternoon	Calendar
CID40506LAC4354	Thursday	afternoon	Messaging
CID40511LAC4354	Thursday	afternoon	Settings
CID58063LAC4354	Thursday	afternoon	Games
CID58063LAC4354	Thursday	evening	Settings
CID64457LAC4354	Thursday	evening	Call History
CID22057LAC4354	Thursday	evening	Messaging

3.3 User Model Implementation

From an implementational point of view, the bayesian user model is developed with Netica™ [8], a software provided by Norsys Software Corp. Netica is a complete software package which includes a graphical editor and an Application Programming Interface (API). The API is available under several operating systems and is accessible within different programming languages. We use a C version specially crafted for Windows Mobile devices. The Netica-C API is a compact Dynamic Link Library (DLL) of ultra-fast C-callable functions.

3.4 Parameters Learning

In order to process bayesian inference, we need to specify the joint probability distribution of each node of the network. As discussed earlier, the structure of the network drives data gathering. In turn, the collected data are processed by a parameter learning algorithm to adapt the CPTs. Netica supports parameter learning from raw case files. Once parameters are learnt, the user model can be used to infer knowledge.

User's tasks and needs evolve over time as user's experience increases. This is an important issue to take into account. Before running the parameter learning algorithm, we therefore fade the CPTs of nodes to indicate greater uncertainty, which accounts for the idea that user's tasks and needs may evolve over time. Thus, what has been recently learned is more strongly weighted than what was learned long ago. The amount of fading to be done is $1 - r^{\Delta t}$, where Δt is the amount of time since the last fading was done, and r is a number less than but close to 1.

4 Clustering

We have experimented clustering techniques for merging atomic contexts of use (days, times, locations) into key contexts of use. Merging is based on past user's actions similarities. For instance, we merge two locations in which the user has set the same phone's profile and used almost the same set of applications. Many clustering methods exist. We use two of them: K-Means and Hierarchical clustering. K-Means clustering is a partitioning method while Hierarchical clustering is an agglomerative one. We use Hierarchical clustering at the beginning when no key context of use has been identified yet. Then we use K-Means to reinforce existing key contexts. Before performing clustering, we first eliminate non-significant variables, i.e. variables of the context (day, time, or location) for which the standard deviations computed for tasks and preferences are close to zero. Standard deviations are computed as follow:

$$\sigma(X) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_i - \bar{x}\right)^2}$$

where X is either *Task* or *Preference*, N is the number of states for node X, and x_i are the conditional probabilities p_i of P(X | *variable*).

Hierarchical clustering starts by putting each data in a separate cluster. Then, at each step, the algorithm chooses the pair of closest clusters and merges them into a new one (Fig. 5). Hierarchical clustering produces clusters for all possible number of clusters. Distances between clusters can be computed from one of single-link, complete-link, average-link, and centroid methods.

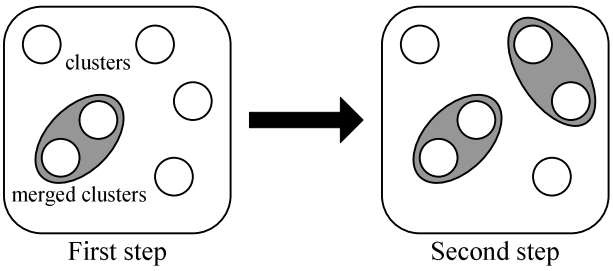

Fig. 5. Two steps of hierarchical clustering.

We use K-Means clustering to reinforce existing key contexts. K-Means assumes a fixed number of clusters, *k*. The goal is to create compact clusters. The classic K-Means algorithm starts by randomly initializing clusters. Here, we start by initializing the first *n* clusters with existing key contexts, and then we randomly initialize the *k* – *n* remaining ones. Then each data is assigned to the nearest cluster based on a similarity measure. Clusters are then recomputed (Fig. 6). The algorithm repeats the last two operations until convergence.

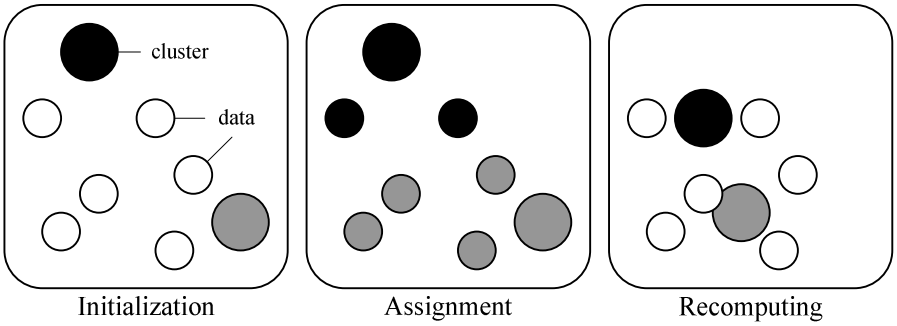

Fig. 6. First step of K-Means clustering.

5 Results and Perspectives

EMMA is still under evaluation. However, the early results based on six people show that users tend to have two key contexts of use at least. This calls for further evaluation to (1) understand whether contexts of use can be matched with user's profiles, (2) measure minimum and maximum numbers of contexts of use, and (3) elaborate a methodology that takes into account this first probing in the wild. In the

near future, EMMA will act as an end-user's tool for managing context-aware adaptation. Envisioned adaptations are phone's profile managing and phone's menu reordering.

References

1. Bridle, R., McCreath, E.: Improving the Mobile Phone Habitat – Learning Changes in User's Profiles. In: AI 2005: Advances in Artificial Intelligence. LNCS, vol. 3809, pp. 970--974. Springer, Heidelberg (2005)
2. Demumieux, R., Losquin, P.: Gather Customer's Real Usage on Mobile Phones. In: Proceedings of the 7th International Conference on Human Computer Interaction with Mobile Devices & Services, Salzburg, Austria, September 19 – 22, 2005. MobileHCI'05, vol. 111, pp. 267--270. ACM, New York, NY (2005)
3. Henricksen, K., Indulska, J. Modelling and Using Imperfect Context Information. In: Workshop on Context Modeling and Reasoning (CoMoRea), Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops, Orlando, FL, March 14 – 17, 2004. PerCom'04, pp. 33--37. IEEE Computer Society, Washington, DC (2004)
4. Horvitz, E., Breese, J., Heckerman, D., Hovel, D., Rommelse, K.: The Lumière Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users. In: Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence, Madison, WI, July 1998. UAI'98, pp. 256--265. Morgan Kaufmann, San Francisco, CA (1998)
5. Kellar, M., Reilly, D., Hawkey, K., Rodgers, M., MacKay, B., Dearman, D., Ha, V., MacInnes, W.J., Nunes, M., Parker, K., Whalen, T., Inkpen, K.M.: It's a Jungle Out There: Practical Considerations for Evaluation in the City. In: CHI'05 Extended Abstracts on Human Factors in Computing Systems, Portland, OR, April 2 – 7, 2005. CHI'05, pp. 1533--1536. ACM, New York, NY (2005)
6. Klann, M., Paternò, F., Wulf, V.: Future Perspectives in End-User Development. In: End-User Development. H. Lieberman, F. Paternò, and V. Wulf Eds. Human-Computer Interaction Series, vol. 9, pp. 475--486 (2006)
7. Laasonen, K.: Where Are You Going? Predicting user movement from cellular data. In: Proceedings of the Proactive Computing Workshop, Helsinki, Finland, November 25 – 26, 2004. PROW'04, pp. 121--124 (2004)
8. Netica, http://www.norsys.com
9. Nielsen, J.: Usability Engineering. In: Morgan Eds. San Francisco (1994)
10. Thevenin, D., Coutaz, J.: Plasticity of User Interfaces: Framework and Research Agenda. In: Proceedings of the 7th IFIP International Conference on Human-Computer Interaction, Edinburgh, Scotland, August 30 – September 3, 1999. INTERACT'99, pp. 110--117. A. Sasse and C. Johnson Eds. IOS Press Publ., Amsterdam, the Netherlands (1999)
11. Weiser, M.: The Computer for the 21st Century. In: Scientific American, 256(3), September 1991, pp. 94--104 (1991)