A framework for ubiquitously capturing users' work and life patterns

Max Van Kleek EMAX@CSAIL.MIT.EDU
MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge MA, 02139 USA

1. Introduction

The field of *user modeling* is concerned with the construction of computational models that describe various aspects of people's lives. While user modeling has witnessed its greatest successes in recommendation systems of online retailers and content providers which amass thousands of opinions from customers, less focus has been paid on individual, *personal* user modeling. Models of a single user, gathered by observing the user's interactions on all of their digital devices collectively, has the potential to capture the user's activities more completely and with greater fidelity than would be possible by, for example, examining the user's interactions with a single web site. Second, these models, taken over a longer term, such as over weeks, months, or years, has the potential to be able to identify time-dependent characteristics of the user, and may potentially open up a way to help identify different moods, situations or tasks in which the user may be engaged.

This paper presents a framework that enables the capture, secure storage, and analysis of a user's activity and environment traces on the user's own computational devices, for the purpose of enabling applications to build predictive models of the user's behavior. Unlike previous approaches such as (Mitchell et al., 2006), which rely on e-mail inboxes, address books and locally kept files for their histories, our system primarily focuses on real-time, transparent capture of the user activities while minimizing user intervention. Unlike systems such as CALO (Cheyer et al., 2005) which require users to use special instrumented versions of common applications, our system leaves the user's applications basically unmodified. Finally, our framework is the first to facilitate integration of user data across all the trusted devices owned by the user, making it easy for applications to gain a unified view of the user's activities.

2. Related Work

Our system differs from TaskTracer (Shen et al., 2006) in several ways; first, as described in section 3, our system abstracts user's low-level actions into a higher-level vocabulary, which allows us to treat e-mails, web pages, and documents uniformly. Second, we incorporate aspects of the user's environment other than the time of day of the event, including the user's location, music they are listening to, and activity level.

3. Design

The system is comprised of individual *observer modules*, which each monitor a single type of user interaction. Observers that have been implemented include the *imap email miner* which connects to the user's mail server, and examines the user's inbox for sent, received and new e-mail; an *http proxy*, which monitors all web traffic generated by the user; *filesystem watcher*, which monitors access/modification times for all files in the user's root filesystem or home directory, and the *Window focus watcher*, which watches for user application switches. Application-specific observers include the *iTunes* and *iChat* observers, which watch for what music the user is playing, and open instant messenger conversations. The *Location monitor* scans for nearby access points for establishing the users "location", and *HIDMonitor* watches user activity level

When each observer module notices a significant event, it tells the framework that something important has happened, using an appropriate term from the action, environment, or system ontology illustrated in figure 1. For example, when the user accesses a document either on the local filesystem or via the web, the appropriate observer module reports an *Access* action, along with a timestamp, the corresponding document path/url, and a uni- and bi-gram word feature vector summarizing the contents of the document. Likewise, when the user sends an email, or chats with another user over an instant messenger, the appropriate observer generates a *Communicate* event, along with the name/e-mail address/handle of the corresponding party, and a uni- and bi-gram word count reflecting the contents of the communication. The user's location, music, idle time, and system start-up and sleep events, are logged under the appropriate categories. The ontology may be arbitrarily extended by adding new observers to the system.

The framework stores all such events in *remembrance logs*, kept in a password-encrypted store saved in the user's home

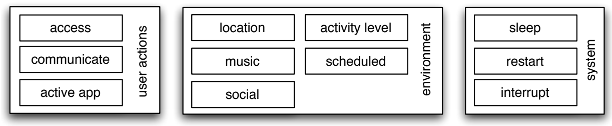

Figure 1. Basic user modeling ontology. Each box represents a type of record written to the user context log.

directory. This password is requested from the user when the system is started up, as well as by any applications needing access to query the contents of the log.

3.1 Access and Aggregation

The framework provides facilities for decrypting and searching for contents within the captured log. Users may query for individual log entries by type, time, or target; additionally, the application may query for *skipping-window summaries* of a particular data type, which summarizes all record instances of a particular type within a particular time interval. For users that own multiple computers, the user can establish their other machines as *context peers*[1], after which point queries from one application on one computer are passed to the other peers (via remote method calls). Results are then temporally collated and merged before being returned to the application.

4. Implementation

The core of the system is implemented entirely in Python; however, many of the observers in the current implementation are designed with MacOS X dependencies; specifically, they interface with MacOS X through AppleScript and command line utilities.

5. Future work

While keeping the system as transparent to the user is desirable to avoid interfering with the user's activities, one disadvantage of being *too* transparent is that the system might become literally invisible to the user. Thus, we will add a graphical indicator that allows the activity and presence of our system to always be known to the user. Another necessary requirement for long-term user adoption is to give the user more control over the capture process, beyond being able to easily stop and re-start the system. An example would be to allow the user to easily prevent to the system from logging sensitive documents that meet a particular criterion, such as those that are a certain type or located in areas of the filesystem. A further feature that might be of use would be a tool that will allow the user to search for and delete sections of their activity log that match some particular criteria, without corrupting the log.

[1]This is currently onerous because the user has always manually update the host ip addresses for their peers if their network location(s) change, and because authentication is done manually by passing the user's passcode on in the query. When the system transitions to using globally connected, persistent, authenticated names using UIA (Ford et al., 2006), the process will become greatly simplified.

6. Conclusion

As the first phase in my thesis work, many questions remain open regarding whether the information captured by this framework will be useful or suitable for various applications of long-term user modeling. The next major phase of this project will surround exploring just that; we will begin by considering strategies for recovering the user's task context, for the purpose of inferring the user's distribution of attentional resources across their tasks. As described in detail in (Van Kleek, 2006), this is part of a larger project aiming to allow users examine their past, to better understand how they are working and living, as well as how certain landmark events, or changes in their physical and social environments, ultimately affected their lives.

References

Cheyer, A., Park, J., & Giuli, R. (2005). Iris: Integrate. relate. infer. share. *Workshop on The Semantic Desktop - Next Generation Personal Information Management and Collaboration Infrastructure at the International Semantic Web Conference (ISWC2005).* Galway, Ireland.

Ford, B., Strauss, J., Lesniewski-Laas, C., Rhea, S., Kaashoek, F., & Morris, R. (2006). User-relative names for globally connected personal devices. *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS06).* Santa Barbara, CA.

Mitchell, T., Wang, S., Huang, Y., & Cheyer, A. (2006). Extracting knowledge about users' activities from raw workstation contents. *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-2006).* Boston, MA.

Shen, J., Li, L., Dietterich, T. G., & Herlocker, J. L. (2006). A hybrid learning system for recognizing user tasks from desktop activities and email messages. *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces* (pp. 86–92). New York, NY, USA: ACM Press.

Van Kleek, M. (2006). Thesis proposal: Proactive support for task and interrupt management.