# Actionable User Intentions for Real-Time Mobile Assistant Applications

Thimios Panagos, Shoshana Loeb, Ben Falchuk

Applied Research, Telcordia Technologies
One Telcordia Drive, Piscataway, New Jersey, 08854 USA
{ thimios, shoshi, bfalchuk }@research.telcordia.com

**Abstract.** We have developed a class of mobile applications that monitor and assist users with their tasks in time and space. The applications provide users with suggestions and reminders based on derived user intent. User intentions are derived from calendar and to-do list entries and are placed in time and space based on user context and availability for the suggested tasks, and the availability of resources in their environment that are critical to task accomplishment.

**Keywords:** Intelligent mobile assistant, user intention, mobile services, LBS, personalization.

## 1 Introduction

Today, the proliferation of mobile applications has greatly increased both the opportunity and challenges of offering personalized services. On one hand, mobile devices allow applications to know more about the user's immediate context (e.g., current location). On the other hand, the time window and user attentiveness present a challenge to applications that offer time sensitive advice and assistance.

It is widely agreed that efforts aiming at providing computer applications with the ability to infer one's intention for the purposes of text or speech understanding, for example, showed that the problem is complex and highly context sensitive. Research in Artificial Intelligence and related areas uncovered the usefulness of constructing hierarchies for organizing commonly used goals, plans scripts, or frames. These knowledge structures provided the context and inference capabilities required for the interpretation of individual actions. User intentions can then be derived from their actions by matching these actions into pre-defined plans and associated goals. Often, when predefined knowledge structure is not known or available, applications attempt to detect behavioral patterns using various machine learning techniques that are then matched with known plans and goals (e.g., [1]).

The need to infer user intentions became more pronounced with the construction of interactive personalized applications where the value of the application is based on offering a unique user experience using knowledge about user intent. In the context of Web applications this requirement translated, for example, to ecommerce sites providing personalized suggestions to their visitors. In most cases, since it was close

to impossible to figure out what the user was looking for, the user was assigned to a cluster of "similar" individuals and the application employed, in one form or another, a statistical collaborative filtering approach.

In this paper, we describe a class of mobile assistant applications that offer users personalized reminders and suggestions. These applications monitor user activities and look for the right time and location to provide useful advice based on user intentions and priorities. To enable these applications, we took the approach that user intentions can be derived from the calendar and to-do list (i.e., productivity) applications users interact with. In addition, the applications also monitor in real time whether or not the user schedule is on track and suppress suggestions if they may cause further schedule disruptions.

## 2 Technologies of Mobile Reminders

In order for a mobile service (and its backend) to effectively exploit the personal contextual and productivity resources associated with a user (e.g., her schedule and to-do items), it is necessary to be able to: i) access this information on an ongoing asynchronous basis, ii) understand it, and iii) trigger effective actions from what is discovered from the resources. Open APIs and Internet-connected services address 'access' and the ability to run mobile applications in the background on some new platforms satisfies the long-running requirement.

Understanding and exploiting information – though it may be easily accessible – is often most difficult. Syntactic formats (e.g., iCalendar) help but free text is a mode used often for the "description" or "action" part calendar and to-do entries. Understanding when it is appropriate to trigger a user interaction is a challenge that requires understanding of the particular user's habits and current context. Thus, there are several broad requirements for productivity application-driven mobile reminders (beyond user location) including:

- Understanding deeply the languages and formats used by the user's calendar and productivity applications;
- Understanding user context to support reminder generation;
- Using the semantics of time and location to build better reminder systems [2].

Compromises are acceptable but tend to decrease value or efficiency in some way or another; thus a cost-benefit analysis is sometimes applied (for example, constraining user selections to a preset vocabulary or style increases system efficacy but decreases user experience).

## 3 High-level Architecture

User intent is deduced by noticing events associated with the user, such as creation of new calendar entries or to-do tasks, examining current user context, and matching noticed events against potential actions. In this section, we describe the platform used for building the mobile assistant applications mentioned on this paper. Our platform employs an event stream approach to matching incoming events to potential actions

based on user context, as shown in Figure 1. Connectors are used (not shown here) for interacting with external information sources and applications (e.g., Google Calendar, Microsoft Outlook, Web portal) for the purpose of retrieving events and injecting these events into the platform.

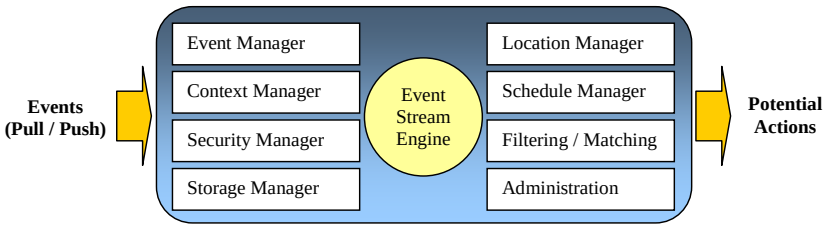


**Figure 1:** High-level platform architecture

In the remainder of this section, we describe some of the main platform components illustrated in Figure 1, "walk through" a to-do entry creation event, and illustrate a Web-based approach to action notifications.

**Event Manager:** Responsible for pre-processing incoming events, such as location updates, and determining whether to forward these events to the context manager, inject them into the event stream engine, or ignore them. The decision to process incoming events is based on user context, and information registered with the schedule and location managers.

**Context Manager:** Responsible for efficient management of user context based on observed context events and user preference and policy enforcement (e.g., privacy rules, notification preferences, profile information). This component is responsible for parsing of the information present in context events (e.g., to-do task description), inferring user intent, and determining how to satisfy the intent (e.g., "buy milk" to-do task is mapped to "grocery store"). This is achieved by identifying the ontology associated with the event and then selecting the appropriate taxonomy within this ontology. Currently, the ontology identification is achieved by parsing the text associated with appointments or tasks.

**Event Stream Engine:** Offers content-based publish-subscriber functionality. Since the intelligence of our platform relies on user context management and efficient pre-processing of incoming events, the event stream engine component can be an existing complex event processing engine (e.g., Esper).

**Location Manager:** Responsible for the efficient representation and management of location information associated with user context entities, such as calendar entries, and location update events. For scalability, a hierarchical representation of location information is being used. Here, non-leaf nodes contain aggregate information about the location information present in the leaves, allowing efficient identification of location events that can be ignored (e.g., traffic accident along a route that does not impact any of the users).

**Schedule Manager:** Responsible for the efficient representation and management of deadlines associated with user context entities, such as calendar entries and to-do tasks, and predictive escalations for cases where deadlines may not be met (e.g., late for appointment).

**Filtering / Matching:** Consumer of notification generated by the event stream engine. It interacts with the context manager for determining how to process these notifications and select those that will be presented to the user.

The following use case illustrates the steps taken when a new to-do entry event is detected, either via explicit registration (e.g., Web portal, SMS) or via a "pull" from the appropriate productivity application (e.g., Microsoft Outlook).

- To-do task connector retrieves new task and forwards it to the event manager;
- Event manager receives new to-do task and follows the steps below:
  - Interacts with security manager to authenticate and authorize data;
  - Interacts with context manager to update user context;
- Context manager follows the steps below upon receipt of new to-do task:
  - Parses the task in order to identify the ontology associated with the information present in it and select an appropriate taxonomy (e.g., grocery store for "buy milk").
  - Interacts with schedule manager in order to register time information associated with the task, if any;
  - Interacts with location manager in order to register location information associated with the task, if any;
  - Registers appropriate event subscriptions (i.e., filters) with the event stream engine.

Figure 2 illustrates two ways in which the platform can inform a user about actions related an observed to-do task (i.e., buy milk), current user location, and user context.


**Figure 2**: Example actions (reminders) generated by the platform; a purely SMS interaction mode (left), and a smartphone application or Web-based portal (right).

## 4 Related Work

Natural Language Processing (NLP) is an ongoing research challenge addressed principally by either knowledge or statistical approaches. Determining parts of speech is a key part of NLP and resolving the ambiguities of language use is particularly difficult. As an example, the mapping of language to intent is very difficult [4] [6].

Microsoft Research [3] and Google, among others, have studied the cost-benefit of mobile reminders in order to effectively choose the best reminder at a given moment. Typically, these approaches rely upon a Bayesian learning phase in which users "train" the system. Other approaches lean heavily on exploiting timing and temporal nature of appointments, tasks and activities. In [5], with the help of profiles, the authors focus on the current time of day to infer users' interests and requirements at the given moment (e.g., for food, travel, etc.).

Startups like Hakia.com allow freely structured search queries and deliver results tailored to the derived meaning of the search terms. Google Calendar and AskSandy-like systems parse limited natural language phrases such as "meet with bob in New York today at 5-7am" while the new mobile devices now ship with onboard applications that assist in productivity in innovative new ways: e.g., Palm Pre and Android G1 can easily enable location-based reminders.

However, greater algorithmic complexity tends to require a server based system; contextual and learning capabilities tend to require a (costly) learning phase in which the system and the user are in a feedback loop. Page-views and ad banner click-through rate (e.g. a metric Google uses) is used to infer interest in products. Text search term auto-complete is a well-used technique that infers search terms by looking at a community's – or a given user's – past searches.

## 5 Conclusions

In this paper, we have presented our work [2][7] in the area of mobile assistant applications that exploit user context information present in calendar and to-do task entries in order to suggest actions at specific times and locations. Our approach to using calendar entries and to-do list tasks for inferring user intent and, thus, offer more relevant and personalized services is unique. Currently, a number of mobile operators are planning to pilot one of the applications we have built on top of the platform outlined in Section 3 to their pre- and post-paid subscribers.

## References


1. Kiefer P. and Stein K.: A Framework for Mobile Intention Recognition in Spatially Structured Environments. Proceedings of the 2nd Workshop on Behavior Monitoring and Interpretation BMI'08, Kaiserslautern, Germany, September 23, 2008.
2. Falchuk B., Loeb S., Panagos T.: Deep-Context Personal Navigation System, Proc. ITS America 15th World Congress on Intelligent Transportation Systems (2008).
3. Kamar E., Horvitz E., Meek C.: Mobile Opportunistic Commerce: Mechanisms, Architecture and Application, Proc. AAMAS 2008, Portugal (2008).
4. Gal. Y et al: The Influence of Task Contexts on the Decision-Making of Humans and Computers, in Modeling and using Context, Springer (2007).
5. Panayiotou C., et al: Time based Personalization for Moving User, Proc. ICMB'05 (2005).
6. Fleischman A., Roy D.: Intentional Context in Situated Natural Language Learning, CoNLL-2005: 9th Conf. on Comp. Natural Language Learning (2005).
7. Loeb S., Falchuk B., Panagos E.: Fabric of Mobile Services, John Wiley & Sons (2009)