Beyond Calendar Mashups: SELFPLANNER 2.0

Ioannis Refanidis[1], Anastasios Alexiadis[1] and Neil Yorke-Smith[2]

[1] University of Macedonia, Dept. of Applied Informatics, Thessaloniki, Greece
{yrefanid, talex}@uom.gr
[2] American University of Beirut, Lebanon and SRI International, US
nysmith@aub.edu.lb

Abstract

Modern electronic calendars offer a variety of functionalities to help a user organize her activities—her tasks and events. However, rarely do these tools support automated scheduling and rescheduling of a user's activities. This demo paper presents SELFPLANNER 2.0, the latest version of a web-based calendar prototype that helps a user to organize her activities by coupling a rich activity model with a scheduling engine. Activities are considered as having temporal domains, utilities, variable durations, and alternative locations; they may be interruptible or periodic; and they may be concurrent. The user is able to express constraints and preferences over the way individual activities or pairs of activities are scheduled. The underlying scheduler seeks to maximize the overall schedule utility using a greedy approach. SELFPLANNER employs Google Calendar for presentation and Google Maps to compute travelling times for temporally adjacent activities scheduled in distant locations.

Introduction

Paper calendars have, for more and more people, given way to electronic calendaring tools. Web-based calendar applications, such as Google Calendar, Yahoo! Calendar, or 30 Boxes, allow the user to connect from a variety of devices without the need for synchronization. They offer intuitive functionality, such as reminders, multiple overlapping calendars, manual meeting arrangement, calendar sharing and publishing, to-do task lists, and so forth. Similar functionalities are provided by client-side calendar applications such as Apple's iCal and Microsoft's Outlook.

Whether web-based or client-based, no popular calendaring applications provide the means for effective automated activity scheduling. By automated scheduling we refer to the use of a scheduler that decides where to put an activity—an event or a task—within a user's calendar, subject to the user's approval. In order for a scheduler to decide when to place an activity, a rich formulation of the scheduling problem is necessary: otherwise the proposed schedule would be rejected by the user as unrealistic. This problem formulation should specify, for each activity: the allowed time intervals (that is, its temporal domain), its duration and whether the duration is fixed or variable, whether the activity is interruptible or not, whether it is periodic or not, where should the user be in order to accomplish it (e.g., the location of the appointment), and how much time is needed in order for the user to move from one location to another. Further, the formulation should reflect the user's preferences, that is the utility gained when an activity is accomplished, as well as additional utility gained by the way an activity is scheduled within its domain (for example, earlier in the day may be preferred to later in the day), and any utility gained by the way pairs of activities are scheduled in relation to each other.

In this demo paper we present SELFPLANNER, a web-based prototype calendar application that couples such a rich activity model with a scheduling engine. Online since 2008, SELFPLANNER has several dozen real users. An earlier version of the system, circa 2008, was evaluated by its users with very promising results (Refanidis and Alexiadis, 2011). The latest version of the system, version 2.0, illustrated in this paper, implements a significantly expanded model (Refanidis and Yorke-Smith, 2010). SELFPLANNER uses Google Calendar for presentation purposes and Google Maps to compute travelling times. The system is not designed to support automated meeting arrangement; the user can use the manual meeting arrangement tools provided by Google Calendar, with the time of the meeting being considered as busy time by SELFPLANNER when scheduling the rest of the activities. The system is accessible through http://selfplanner.uom.gr.

This demo paper illustrates the complexity that an intelligent calendar assistant must address, through a set of motivating examples. We then present the key features of the system, and conclude by anticipating future developments in intelligent calendar applications.

Motivating Examples

The vision behind SELFPLANNER is to consider both events and tasks when providing automated scheduling assistance to support the user's time management. Traditionally, events such as a doctor's appointment are placed in a user's calendar, with a specific time (and potentially location) reference, whereas tasks sit in a to-do list, being characterized perhaps by a deadline. Treating events and tasks differently, with only events having their place into a user's calendar, can result in tasks missing their deadlines. Apple iCal and Microsoft Outlook allow the user to drop a task from the to-do list into the calendar, thus allocating manually some time slots to the task. Google's applications weakly integrate tasks, events, and email; a strong integration is adopted by the open source Chandler Project. With SELFPLANNER, any activity that requires some of the user's time has its place in her calendar. In the following paragraphs we give some motivating examples.

The simplest case of an activity is one that has a designated schedule. For example, attending a lecture. The user has only to decide whether or not to attend the lecture. If she decides to attend, there is no option to negotiate the time when or the location where the lecture will take place. By contrast, activities such as buying food are more flexible: the user can decide between alternative schedules. Consequently, providing automated assistance in scheduling such activities relies on a wealth of information (Krzywicki et. al., 2010), such as:

- A duration estimate (how long?)
- The timetable of the shops (when are they open?)
- The alternative shops which are compatible with the shopping list (which shops sell food?)

Going shopping can also be considered a periodic activity, i.e., one that is repeated, say once a week. While periodic activities do not have to be scheduled in the same time and location within each period, they should be scheduled once within each period, whenever possible. For example, going shopping may take place either on Friday or on Saturday, either in the morning or in the evening, depending on the rest of the activities in the user's calendar within each week. The location for each instance of a periodic activity can also be selected based on the locations of other activities that are scheduled adjacently in time, with the aim to minimize travelling time.

Some activities, such as reading a book, are not usually performed all at once: we call these activities interruptible. The user might estimate an overall duration for this activity, e.g., 30 hours, and the activity will be divided into parts that will be scheduled separately. These parts do not have to share the same duration or location, but their sum should equal the overall duration specified for the activity. The user might specify compatible locations where she should be in order to execute the activity and, potentially, a temporal domain. Note that periodicity and interruptibility are orthogonal properties: an activity can be periodic or interruptible, or both.

Activities may serve as placeholders or reminders (Palen, 1995). It is a common situation to have overlapping activities in a user's calendar. For example, one might define a three-day activity concerning attending ICAPS 2012 in Brazil and several other more specific activities concerning attending specific sessions of the conference.

Busy users often have an overloaded schedule, such that all of their activities cannot fit within their calendars or they cannot all be scheduled in an ideal way. Some decisions have to be taken concerning which activities to schedule and how. Each activity included in the user's calendar presumably yields some utility to the user, which depends on the activity. For instance, doing business may be considered more important by some than spending time for leisure activities. However, whether or not an activity is included in a user's calendar is not the only source of utility. Utility may also result from the way an activity is scheduled, both on its own as well as in conjunction with the rest of them. Hence, it might be preferable to decide not to schedule an activity at all, in order to get a better schedule for the rest of them.

As an illustration, consider sleeping as a typical example of an activity for which alternative schedules result in different degrees of user's satisfaction. The user might prefer to 'execute' this activity at once within each day than executing it in parts. Usually the user prefers to schedule this activity during the night than during the day. She also might prefer to schedule this activity at home than at her office! Finally, sleeping has a variable duration, in the sense that the user can decide how long to sleep (we do not consider stochastic activities, where duration is a random variable). For example, one might want to sleep between 6 and 8 hours daily, with 8 hours resulting in more utility than 6 hours.

Further, utility may also result from the way sets of activities are scheduled in relation to each other. Consider writing a paper and doing housekeeping. Writing the paper is a heavy mental task that is optimized when one has clear head and is concentrated on it. One might prefer to work on writing the paper before doing housekeeping or, alternatively, to leave some minimum temporal distance between the two activities. However, in case it is not possible to schedule these activities in the optimum way, the user is willing to accept any schedule that includes these activities, provided she completes both, even if the schedule violates some of her ideal preferences.

Finally, and not least, constraints can hold between activities. For example, consider a pair of activities concerning going to Crete on holiday by plane, and the return journey. Obviously, going to Crete should precede the return flight (an ordering constraint), whereas we could

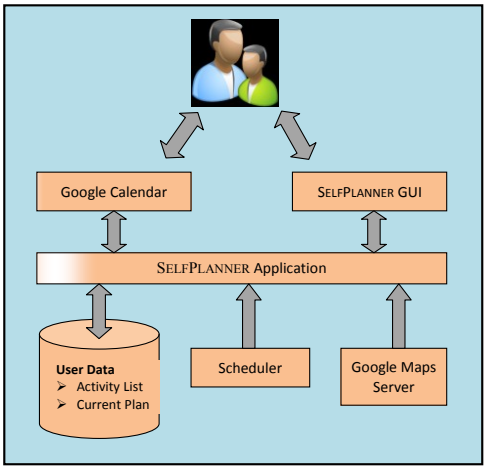

Figure 1. SELFPLANNER architecture.

ask for a minimum and maximum duration for the whole journey (a temporal proximity constraint between these two activities). Other activities to be executed during the vacation should be scheduled between the two flights. Implication constraints might also hold between these activities, imposing for example that there is no reason to schedule one of the two flights—or any other of the activities that are related with the vacation—without scheduling the other.

Scheduling a user's activities is a continuous process. New activities might arrive at any time, causing already scheduled activities to be rescheduled or even abandoned in order to accommodate the new ones within the user's calendar. The need for rescheduling is critical for any intelligent calendar application. In order to minimize the user's disturbance, rescheduling should give less preference in changing the user's short-term schedule.

The System

SELFPLANNER 2.0 supports all the modelling aspects one needs in order to represent the situations described with the motivating examples, and many more. Activities are characterized by a utility value, a duration range (with greater duration resulting in greater utility), a temporal domain, a preference function over the various time slots of the temporal domain, a set of alternative locations, and a utilization parameter, concerning the percentage of a user's attention required when the activity is executed, with the constraint that the sum of the utilizations of all concurrent activities does not exceed 100%. In order for a set of activities to be scheduled concurrently, they need to have the same (or compatible) locations. Temporal preference functions supported are linear descending (the earliest the better), linear ascending (the latest the better), stepwise descending (before some specified time point), and stepwise ascending (after some specified time point). The user is free to specify as little or as much of the information for an activity as she likes.

For interruptible activities the user can specify the minimum and maximum allowable duration for their parts. Further, the user can specify extra constraints—either mandatory hard constraints or flexible soft constraints—concerning the minimum and the maximum temporal distance between pairs of parts of the interruptible activity. Soft constraints, if satisfied, result in extra utility; partial satisfaction is supported. Four types of binary hard and soft constraints are supported: Ordering constraints, minimum distance constraints, maximum distance constraints, and implication constraints. An activity can also be periodic, with each instance of the activity being treated as a separate activity by the scheduler. Daily, weekly and monthly periodic activities are supported.

The scheduling problem that results can be framed as a constraint optimization problem. SELFPLANNER uses a greedy heuristic scheduling engine, based on the Squeaky Wheel Optimization framework, boosted with domain-dependent heuristics, to solve the underlying constraint optimization problem. Experimental results have shown that even problems with dozens of fully featured activities can be solved near optimally in a few seconds (Refanidis and Yorke-Smith, 2010; Refanidis, 2007).

The system's architecture is shown in Figure 1. On the server side, the SELFPLANNER application implements the system's logic; a scheduling engine generates plans for the user's activities; and a database retains users' data, including their activity lists and current plans. The user interface consists of a Java applet running on the client side. The user interacts also with Google Calendar, in order to watch her plan, which is updated by the main SELFPLANNER application. The user can also add new events directly into her Google Calendar, e.g., meetings with other people; these events are considered busy time by the system when scheduling her activities. Finally, the SELFPLANNER application interacts with the Google Maps Server, in order to compute travelling times between the locations of the activities.

Special attention is given to the user interface. While many parameters may be specified for each activity, most of them usually have default values. The most important features of the system (Figure 2) are the following:

- Users indicated that the most time-consuming task concerns the definition of a temporal domain. SELFPLANNER implements an innovative way to define temporal domains, which consists of a combination of template application and manual editing. A template is a user defined pattern of time slot inclusion/exclusion covering a day, a week, or a month. A template can be applied over the entire temporal domain of an activity, or over a part of it, to include or exclude specific time

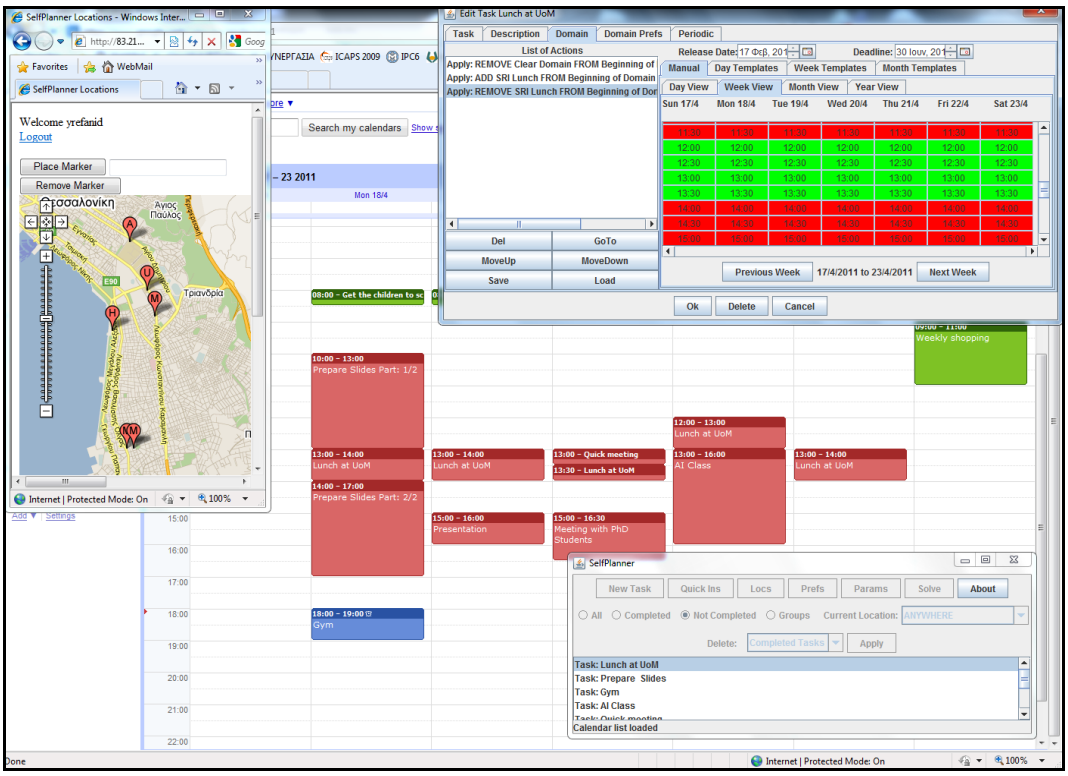

Figure 2. An overview of the SELFPLANNER system. The main application window (lower right) contains the current list of activities. In the upper right corner is the Edit Task dialog box, shown editing the temporal domain of the Lunch at UoM activity. It is interesting to have a look at how this activity has been scheduled. It is a daily periodic activity with a duration that may be either 30 mins or 1 hour (preferred). The restaurant is open from 12:00 to 14:00 daily, and the user prefers to eat as late as possible, i.e., close to 14:00, more strongly than she prefers to have 1 hour for lunch. Seen on the left is the interface for designating relevant locations. Behind the other windows is the Google Calendar interface in which the whole schedule is shown.

slots. Sequences of templates can also be applied. Finally, the user can edit the temporal domain manually. Using templates, the user can define large temporal domains with a few clicks.

- Daylight savings time is taken into account when scheduling, whereas when displaying the resulting plan, the system takes into account the user's timezone. Multiple calendars are supported.
- User defined classes of locations, e.g., malls, are also supported, giving the user the option to assign to an activity large sets of locations with a single choice.
- Specific instances of a periodic activity can be excluded from a user's calendar easily, without the need to resort to the calendar.
- Each time the user logs into the system, she is offered the opportunity to give feedback concerning the status of any activity that should have been accomplished between her last logout and the current time. Hence, the system knows which already scheduled activities were not accomplished and reschedules them.
- The user is able to specify her current location (with the default current location being the location of the last accomplished activity). Hence, there is always enough time to travel to the location of the next activity.
- Activities directly entered into the user's Google Calendar are considered as busy time, so no other activity is scheduled in the same time period. However, these activities are not annotated with a location, so travelling time cannot be computed.
- Since many activities are inflexible, there is an option for rapidly inserting a new non-interruptible activity with specific start time and location. This is equivalent with directly entering the activity into the user's Google Calendar, however doing it through SELFPLANNER allows to specify the activity's location.

Conclusions and Future Work

SELFPLANNER is a continuously evolving prototype application aimed at reducing the effort in the management of a user's electronic calendar. Currently it implements a rich activity model with simple, interruptible and periodic activities, locations, concurrency, and several unary and binary hard and soft constraints, coupled with a powerful best-effort heuristic scheduling engine. An evaluation of an earlier version of the system, since 2008, with real users,

demonstrated its strengths over traditional electronic calendars as well its potential for broader user adoption (Refanidis and Alexiadis, 2011).

Currently, SELFPLANNER is being evolved through its integration in an information system supporting the visitor of an area to enhance his visit with cultural activities. The integrated system, called MYVISITPLANNER, will retain information about cultural activities offered in the area of northern Greece (particularly Macedonia and Thrace), including metadata such as location, working hours, estimated visit duration etc. Coupling this information with the user's calendar and profile, the system will be able to suggest activities to the user and, once accepted, schedule them in his calendar, while taking into account constraints imposed both by the new as well as the existing activities.

SELFPLANNER is a step towards creation of general purpose intelligent user assistants. Such assistant agents aim to helping the user to organize and accomplish her tasks (Maes, 1994; Freed et. al. 2008). One aspect of their duties is to help organize the user's time, i.e., advising what to do and when, or supporting calendar management and meeting coordination (Berry et. al. 2011). However, there are other ways in which SELFPLANNER could be extended in order to provide true intelligent assistance in organizing a user's activities. For example:

- Coordination with other users: Currently SELFPLANNER organizes only the activities of a single user, with meetings with other users being considered as busy time. Assuming that several users are using a system with scheduling capabilities, arranging meetings could involve rescheduling of already scheduled activities for all users (Berry et al. 2011).
- Planning problem: The current activity model could be enhanced with allowing the activities having preconditions and effects. Consequently, the scheduling engine should be replaced by a planning engine, able to add activities into the user's calendar in order to support the open goals.
- Parameter learning: Instead of information about activities being entered by the user, machine learning can be exploited to more intelligent pre-populate fields based on previous activities (Krzywicki et. al. 2010). While the knowledge entry and management has not been a focus of our research, we are aware it is significant for real-life adoption of intelligent agents.
- Information extraction: An intelligent agent should be aware of its user's profile and exploit this knowledge in order to extract information, such as from the internet, concerning activities that might be of interest for its user, e.g., attending ICAPS 2012 (Oh et. al., 2010).
- Web service invocation: Finally, a real intelligent agent should be able to accomplish some activities automatically, be invoking suitable web services if available (Freed et. al. 2008). For example, for an already scheduled activity concerning travelling to Brazil to attend ICAPS 2012, booking the plane tickets and the hotel, or getting information for the weather conditions, might save a lot of the user's time.

Acknowledgements

The first two authors are supported for this work by the European Union and the Greek Ministry of Education, Lifelong Learning and Religions, under the program "Competitiveness and Enterprising" for the areas of Macedonia-Thrace, action "Cooperation 2009", project "A Personalized System to Plan Cultural Paths (myVisitPlanner^{GR})", code 09SYN-62-1129.

References

Berry, P.M., Gervasio, M., Peintner B., and Yorke-Smith, N. 2011. PTIME: Personalized Time Management. *ACM Transactions on Intelligent Systems and Technology* (to appear).

Freed, M., Carbonell, J., Gordon, G., Hayes, J., Myers, B., Siewiorek, D., Smith, S.F., Steinfeld, A. and Tomasic, A. 2008. RADAR: A Personal Assistant that Learns to Reduce Email Overload. Proc. of 23rd AAAI Conference on Artificial Intelligence (AAAI-08), Chicago, IL, USA. AAAI Press.

Krzywicki, A., Wobcke, W. and Wong, A., 2010. An Adaptive Calendar Assistant Using Pattern Mining for User Preference Modelling. Proc. of 14th International Conference on Intelligent User Interfaces (IUI'10), Hong Kong, China, February 2010.

Maes, P., 1994. Agents that Reduce Work and Information Overload. *Journal of ACM*, 37(7).

Oh, J., Meneguzzi, F. and Sycara, K.P., 2010. ANTIPA: An Agent Architecture for Intelligent Information Assistance. Proc. of the 19th European Conference on Artificial Intelligence (ECAI-10), Lisbon, Portugal.

Palen, L., 1999. Social, Individual and Technological Issues for Groupware Calendar Systems. Proc. of CHI 99 Conference on Human Factors in Computing Systems, Pittsburgh, PA, USA.

Refanidis, I. 2007. Managing Personal Tasks with Time Constraints and Preferences. Proc. of 17th International Conference on Automated Planning and Scheduling Systems (ICAPS-07), Providence, RI, USA. AAAI Press.

Refanidis, I. and Alexiadis, A. 2011. Deployment and Evaluation of SELFPLANNER, an Automated Individual Task Management System. *Computational Intelligence*, 27(1), 41-59.

Refanidis, I. and Yorke-Smith, N. 2010. A Constraint Based Programming Approach to Scheduling an Individual's Activities. ACM Transactions on Intelligent Systems and Technologies, 1(2), 12:1--32.