# Towards a User-centred Planning Algorithm for Automated Scheduling in Mobile Calendar Systems

Christoph Halang and Maximilian Schirmer

Mobile Media Group
Bauhaus-Universität Weimar
Bauhausstraße 11
99423 Weimar
christoph.halang@uni-weimar.de, maximilian.schirmer@uni-weimar.de

**Abstract:** Agreeing on appointments and scheduling time slots with multiple participants is a complex task and remains a challenge in our everyday lives. Despite the technological advances in the areas of mobile computing and cloud storage, these planning processes are still tedious. In this paper, we propose a user-centred planning process for scheduling appointments between users. We describe the algorithm derived from the process, and present a first prototype for a mobile calendar application that implements the proposed process.

## 1 Introduction

Allocating time slots and managing appointments is an everyday task that forms the basis for an overwhelming multitude of private and business activities. People arrange and rearrange schedules, they organise and delegate meetings, and everything is based on various calendar systems. Nowadays, mobile calendar systems (e. g., in smartphones) support these planning and coordination processes while people are on the go. Recent mobile calendar systems synchronise with their stationary counterparts and with cloud services. This enables multiple users to interact directly through their calendars, to send invites or notifications for upcoming events, or to share calendar data across several users and devices. But these technological advancements cannot overcome the fact that planning and coordinating appointments still relies on the mutual agreement of several parties. Achieving this agreement can be a complex process. Even more when appointments need to be scheduled along a number of already fixed appointments that block existing time slots. Furthermore, the time required for an appointment actually contains a second part that is required for reaching the appointment's location. With an increasing number of mobile users, this fact quickly results in a planning problem that is close to impossible to solve for most users.

In this paper, we introduce a user-centred planning process for a mobile calendar system that simplifies finding an agreement between users. Spatio-temporal context information is used to provide recommendations for scheduling. The process relies on a planning algorithm that respects travel times between appointments, already scheduled appointments,

and wishes for specific time slots an appointment should take place in. It can be used fully automated, but also allows user intervention (i. e., manual planning) at any time.

The remainder of this paper is structured as follows: Section 2 discusses related work. Section 3 introduces the user-centred planning process. Section 4 presents our mobile calendar system prototype. Section 5 summarises the paper and provides future work.

## 2 Related Work

Due to the shortness of this paper, we can only provide a short overview of relevant fields of research at this point. The research presented in this paper is rooted in the scientific field of operations research [CAA57]. Operations research focusses on modelling and analysing systems that support users in decision-making and planning processes. Typical examples and achievements of operations research are project planning, supply chain management, and all types of scheduling problems. Operations research is an interdisciplinary field of research with a multitude of interfaces to computer science, management, and mathematics.

Furthermore, our work can be ranked into the broad field of context-aware mobile information systems. Specifically, our research concerns the theory and applications in the area of planning and calendaring systems. Existing systems in this area provide users with automated scheduling [GKI07] on mobile devices, context-aware user interfaces for planning tasks [PIF+04], or make use of reasoning algorithms to derive a best-suited schedule [BGPYS11].

Finally, recommender systems [RV97] are also relevant in the context of this paper. They assist users of an information system by presenting recommendations. These recommendations reduce the need for active searching or browsing through large amounts of data and often employ machine learning or data mining techniques to solve this task.

## 3 User-Centered Planning Process

We introduce a user-centred planning process for the automated scheduling of appointments between several involved parties. The algorithm provides recommendations for possible time slots all parties agree on. At any time, user intervention and manual scheduling remains possible. In a typical planning scenario, a single user wants to schedule appointments with several other users. For the course of this paper, we assume a relationship of a single service provider (*user*) with several other users (*customers*) and use this scenario to introduce and explain the details and structure of the process. The planning process involves spatio-temporal context information about the customers' availabilities, the user's availability and locations of appointments. We introduce an algorithm that realises this process and allows the user to generate a scheduling recommendation at any time, with respect to the user's already scheduled appointments.

### 3.1 Structure

We introduce the term *wishpointment* as the agreement to schedule an appointment between the user and a customer. A wishpointment is determined by a single customer, a location where the appointment should take place, the duration, and an expiration date. For every customer, there is an arbitrary number of time intervals we call *time slots*. Time slots are the representation of the dates where the customer is available for scheduling appointments with the user. They are created explicitly by the customer whereas wishpointments are created by the user. The main task of the planning process (cf. Figure 1) is to schedule the actual appointments. Thereby, the process tries to map the wishpointments to the time slots and aims to avoid collisions with other appointments by taking the travel time to the next and from the last appointment into account. The time interval to schedule appointments for (e. g., next week) is chosen by the user. Wishpointments might not be considered for planning because they expire prior to the start of the scheduling interval.

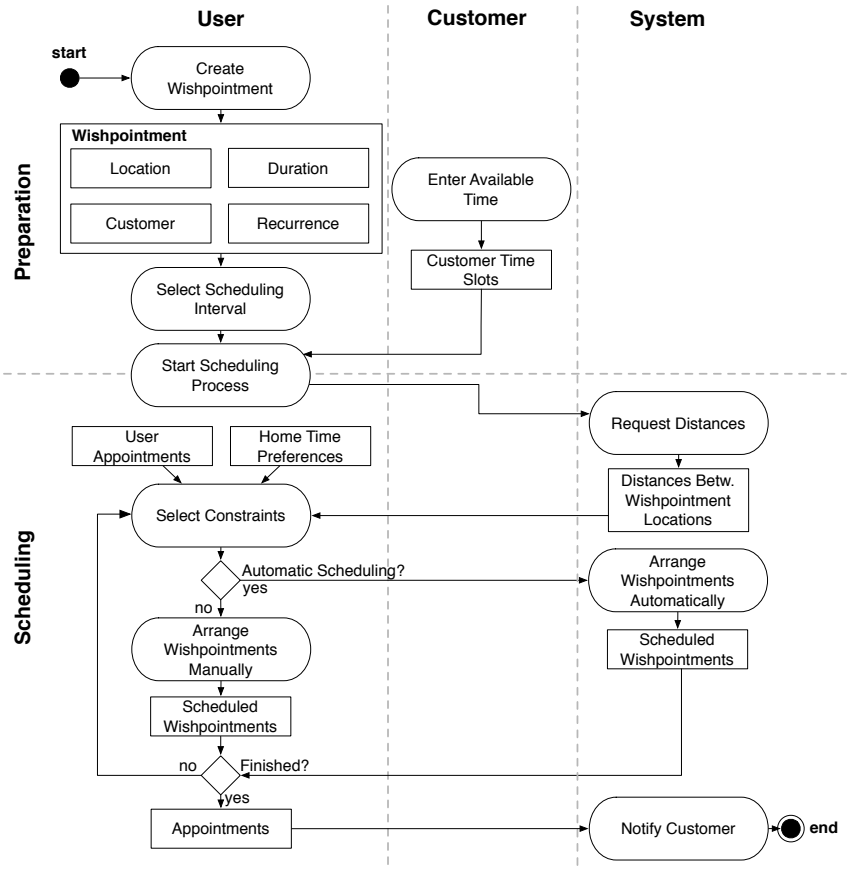


Figure 1: Activity diagram of the planning process. The process can be divided into the involved entities user, customer, and system; as well as the preparation and scheduling phases.

### 3.2 Context information

Context-aware systems gather details about a prevalent context through context elements. At the current state, our proposed user-centred planning process uses location data and personal calendar data as context elements. In the future, we plan to integrate a more complete context model.

The planning process respects the current whereabouts of the user as well as the locations of wishpointments and already scheduled appointments. Driving times between locations are derived from the information where an appointment should take place. This information enables the user and the planning process to assess whether it is possible to reach a specific location, given a start location and maximum travel time interval.

Additional context information is provided by the personal calendar of the user. Naturally, the user's appointments constrain the available time for appointments. Furthermore, the calendar may provide information on the user's location at a given date and time.

### 3.3 Algorithm

The algorithm consists of two phases: (1) *preparation*, and (2) *allocation*. The input of the algorithm is a list of wishpointments, a list of time slots for every wishpointment, and a list of already planned appointments. These appointments are modelled as tuples of slots and wishpointments. For $n$ wishpointments and $m$ possible time slots, a $m \times n$ matrix is created that stores whether a wishpointment can be planned in a specific time slot.

In the *preparation* phase, the number of relevant tuples is reduced to a smaller number of candidates. First, pairs of slots and wishpointments that can't be used for scheduling are identified by checking the following conditions: a) the wishpointment belongs to a different customer than the slot, b) the duration of the wishpointment is longer than the slot, or c) there is a collision with an already planned appointment. Colliding time slots are resized to resolve the collision and allow the slots to be used for further scheduling. At the end of the preparation phase, all relevant candidate pairs of wishpointments and associated time slots are marked in the matrix. In addition, wishpointments are sorted in ascending order by the number of time slots they can be allocated to. Time slots are sorted by their starting date in order to accelerate the search for predecessors and successors.

During the *allocation* phase (see Algorithm 1), the remaining time slots are checked for each wishpointment. First, the algorithm checks for collisions with already planned appointments. Collisions are resolved if possible by resizing or splitting the slot. The next step checks if the time required to arrive at the location would collide with an appointment. Additionally, the time required to arrive at the location of appointment at the next occupied time slot is checked for collision. If the time slot passes all checks, the wishpointment is scheduled in the current slot. When a wishpointment has been scheduled in a specific slot, the slot will not be considered again and is added to the appointments list with all allocated wishpointments and the assigned slot.

The algorithm bases on the assumption that users prefer a larger number of fulfilled wishpointments over travelling shorter distances. Consequently, we focussed on a strategic approach to fulfill as many wishpointments as possible. By looking at every wishpointment only once and starting with those with the fewest possibilities to be scheduled, we end up with a solution suitable for the restricted problem space given by our scenario. The algorithm calculates a single solution, and there is no optimisation for the distance travelled in the given time interval. We expect that in general, there exist very few solutions of the same quality (number of fulfilled wishpointments) for which different round trips can be constructed.

**Algorithm 1** Planning algorithm - Allocation phase

**Input:**

WISHPOINTMENTS // Wishpointments ordered by the number of matching time slots
SLOTS // List of time slots
APPOINTMENTS // List of tuples of wishpointments and slots already scheduled
MATRIX // Matrix of matching wishpointments and slots

**Output:**

APPOINTMENTS // List of tuples of wishpointments and slots

```
01  FOR each W in WISHPOINTMENTS
02    FOR each S in SLOTS
03      IF MATRIX[S][W] == TRUE
04        possible = TRUE
05        FOR each A in APPOINTMENTS
06          IF A.SLOT == S
07            possible = FALSE
08            BREAK
09          IF A.SLOT collides with S AND collision can't be resolved
10            possible = FALSE
11            BREAK
12          IF A.SLOT == S-1
13            IF Approach to S overlaps with A.S
14              IF S.DURATION - Overlap(A.S,Approach(S)) >W.DURATION
15                RESIZE S to remove overlap
16              ELSE
17                possible = FALSE
18                BREAK
19          IF A.SLOT == S+1
20            IF Approach to A.S overlaps with S
21              IF S.DURATION - Overlap(S,Approach(A.S)) >W.DURATION
22                RESIZE S to remove overlap
23              ELSE
24                possible = FALSE
25                BREAK
26        IF possible == TRUE
27          RESIZE S to W.DURATION
28          ADD (W,S) to APPOINTMENTS
29        ELSE
30          MATRIX[S][W] = FALSE
31          RESTORE original duration of S
```

# 4 Prototype

We implemented prototypes for a mobile appointment planning and scheduling application on iOS and Android smartphones and tablets. The application scenario ("Horsesquare") bases on a single user working as a freelancer with an arbitrary number of customers. For every customer, there is an arbitrary number of appointments she wants to schedule with the user. For every appointment to be scheduled, the customer provides a number of time intervals where appointments would be desired.

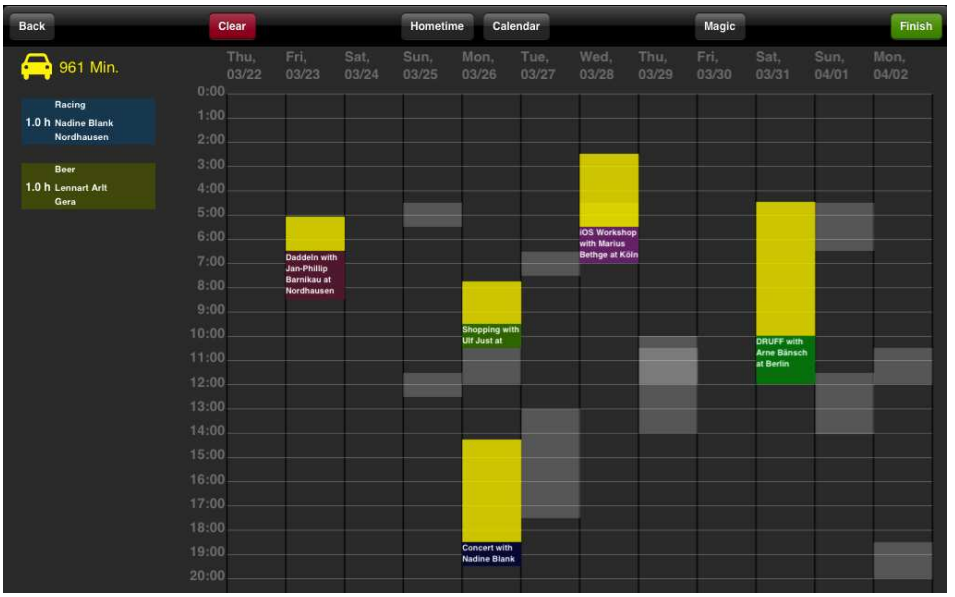


Figure 2: Horsesquare prototype on the iPad - Scheduling process

Figure 2 shows the scheduling view of the iPad prototype. After selecting the time interval for scheduling, the scheduling interface is presented to the user. It is divided into a toolbar and a main view. The main view shows a calendar of the selected scheduling interval that allows the user to arrange the wishpointments for manual scheduling. The wishpointments that are available for scheduling in this interval are shown as rectangles on the left hand side of the main view. Upon touching a wishpointment representation, the calendar view shows the dates available for scheduling the wishpointment, depending on the time slots previously entered by the customers. While being moved over the calendar view, the rectangles change size to reflect the additional time required to arrive in time for this appointment. The additional time results from the location information of the preceding appointment. The total travel time of the current scheduling interval is displayed in the upper left corner in minutes. For scheduled appointments, the time required to approach the location is displayed as a yellow rectangle in the calendar view.

The toolbar offers buttons for clearing all scheduled appointments, adding the time intervals the user wants to spend at home, and selecting the appointments that should be

taken into account during scheduling from the user's calendar. Our prototype bases on the Google Calendar (`http://www.google.com/calendar`) service to exchange the available time of the customers and the date and time of the scheduled appointments. The user shares a calendar with each customer to allow them to enter their available time. The available time for a single wishpointment is entered explicitly as an event in the calendar. By the time the scheduling process is started, the system downloads the available time for each customer. When the scheduling process has finished, the appointments are created in the user's calendar. In addition, an event is created in the calendar shared with the customer, and the customer is invited as an attendee. The driving durations are provided by the Google Maps Directions API (`http://developers.google.com/maps/`) .

## 5 Conclusion and Future Work

The problem we are facing in our scenario is a special case of the Traveling Salesman Problem (TSP) with time windows [Tsi92]. TSP is a typical example for NP-complete discrete optimisation problems. In order to find solutions for such problems, a measure is needed to evaluate the quality of the solution. We presented an approach that strategically schedules appointments for our limited problem space by trying to fulfil as many wishes for appointments as possible. Beyond that, we want to develop an optimisation approach in the future that takes additional criteria (e. g. the distance travelled, the number of scheduled appointments, or the number of free weekdays) into account. Furthermore, the problem of rescheduling existing appointments should be addressed in the future.

**Acknowledgement**

The work presented here is a result from the "Horsesquare" semester project, supervised by Prof. Dr.-Ing. Hagen Höpfner and Maximilian Schirmer, M.Sc. We want to thank the other participants Juliane Reschke, Bastian Karge und Michel Büchner.

## References

[BGPYS11] Pauline M. Berry, Melinda Gervasio, Bart Peintner, and Neil Yorke-Smith. PTIME: Personalized assistance for calendaring. *ACM Transactions on Intelligent Systems and Technology*, 2(4):40:1–40:22, July 2011.

[CAA57] C. West Churchman, Russel L. Ackoff, and E. Leonard Arnoff. *Introduction to Operations Research*. John Wiley & Sons, Inc., New York, NY, USA, 1957.

[GKI07] Georgios Gkekas, Anna Kyrikou, and Nikos Ioannidis. A smart calendar application for mobile environments. In *Proceedings of the 3rd international conference on Mobile multimedia communications*, pages 70:1–70:5, Brussels, BE, 2007. ICST.

[PIF+04] Zachary Pousman, Giovanni Iachello, Rachel Fithian, Jehan Moghazy, and John Stasko. Design iterations for a location-aware event planner. *Personal Ubiquitous Computing*, 8(2):117–125, May 2004.

[RV97] Paul Resnick and Hal R. Varian. Recommender Systems. *Communications of the ACM*, 40(3):56–58, March 1997.

[Tsi92] John N. Tsitsiklis. Special cases of traveling salesman and repairman problems with time windows. *Networks*, 22(3):263–282, 1992.