Custom Soft Keyboards as a Means to Foster Suitability for the Task

Sebastian Müller[1], Carsten Mohs[2], Martin Christof Kindsmüller[1]

Technische Hochschule Brandenburg, HCI Group [1]
timum GmbH, Berlin[2]

Abstract

This paper presents a custom soft keyboard for the Android operating system developed in the context of scheduling available time slots for estate agents. The goal is to determine whether it is feasible to create a keyboard offering additional or different features in comparison to standard keyboards. The resulting keyboard user interface (UI) deviates from standard keyboard UI and the provided functionality was well received by most users.

1 Introduction

This paper presents the analysis of use cases in the context of scheduling available time slots for estate agents and infers which of the use cases should be supported by features of an App and which should be supported by features of a custom soft keyboard. App and keyboard supplement a web solution developed by the *timum GmbH* which operates in the same context.

timum's web solution is optimised for different screen sizes and browsers and is thus already usable on mobile devices. Yet there are use cases where a mobile app could offer an improved or new solution. The development of the keyboard follows a process proposed by Roenspieß and colleagues, (2011) that integrates elements of Feature-Driven Development (Coad et al., 1999), and User-Centered Design (Norman & Draper, 1986). The use cases were gathered by semi-structured interviews (Döring & Bortz, 2016). with experienced real estate agents. Consequently, user feedback was also gathered in all further stages of development. After determining the most important use cases and repeatedly improving the UI designs through user feedback a prototype was implemented in the form of a native application targeted to *Android* smartphones.

Veröffentlicht durch die Gesellschaft für Informatik e. V. 2017 in
M. Burghardt, R. Wimmer, C. Wolff, C. Womser-Hacker (Hrsg.):
Mensch und Computer 2017 – Tagungsband, 10.–13. September 2017, Regensburg.
Copyright (C) 2017 bei den Autoren. https://doi.org/10.18420/muc2017-mci-0362

2 Analysis and Concept

At its core *timum's* web service is a calendar with additional features and data. A first idea was to create an app that provides these additional features and access to said data. It was meant to enhance existing apps. Thus, allowing the users to use the apps they are accustomed to, while also benefitting from the features *timum* provides. The rationale being that this would most likely increase the number of users willing to use the app. Direct interaction of apps is not possible in Android. While it is possible to directly start another app or part of an app from within one's own application and sending necessary data to this application in the process, it is not possible to ensure that the provided data is used in the intended way or at all by the application being started.

The second idea was to create a custom keyboard. It presents the sole exception to directly interact with another application in an even though limited way. Keyboards allow to directly insert data into other applications in form of text or pictures as long as there is an input field to receive the data. They are also allowed to connect to the internet to send network requests.

During a series of semi-structured interviews the following 10 use cases were gathered:

(1) *Support Appointment Creation,* (2) *Share Availabilities,* (3) *Manual Delay Message,* (4) *Automatic Delay Message,* (5) *Manage Unconfirmed Appointments,* (6) *Object -> Interested Users,* (7) *User -> Object of Interest,* (8) *Tough Appointment Creation,* (9) *Customer Ranking* and (10) *Booking Notification.*[1] Out of the ten use cases mentioned above (2) *Share Availabilities* promised to benefit the most from the limited interaction capabilities of a keyboard:

An actor wants to create non-overlapping appointments with a specific set of people. They define a time range in which they can accept appointments. The participants are informed about the event and can pick a time at which they are available. To avoid conflicts, they also see what time slots are already taken by other people of this set but not by whom. The actor can manipulate these appointments.

A keyboard implementing this use case would allow to easily insert a link to the time slots where a certain estate is available into any text field. If no further time slots are known to the system, the keyboard allows the creation of new time slots. It would allow estate agents to immediately answer requests for appointments received over SMS, email or other messaging apps. Instead of copying and pasting or manually typing the links into the text field of the respective application the soft keyboard uses data received from the backend service to create and insert the links directly. Customers who follow such a link can execute the rest of the use case by employing *timum's* UI. The keyboard serves as a universal interface to all applications the user uses regardless whether *timum* planned for the keyboards usage in a particular app. This increases the maintainability of the software. The user benefits from this by having a well-known user interface (UI) available in all applications. Fig. 1 shows the first concepts of such a custom keyboard.

[1] A more thorough analysis of these use cases can be found in Müller (2017)

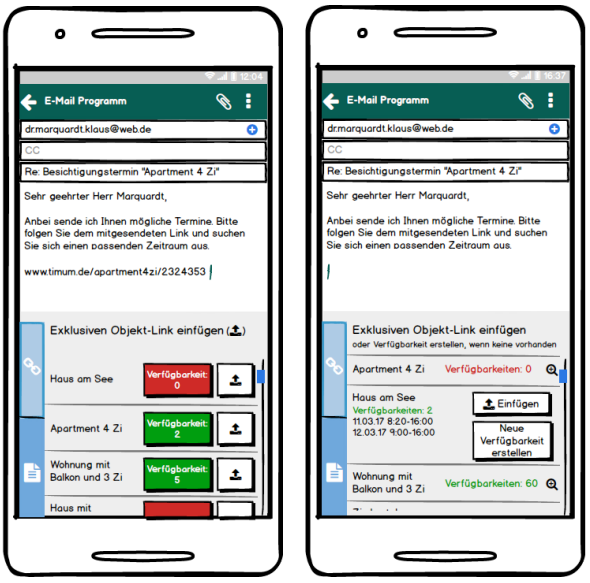

Figure 1: First (left) and second (right) concept of the custom keyboard

3 Results and Outlook

Fig. 2 shows the prototype of the custom keyboard and an example on how it is used in a messaging app. It shows all steps of inserting a link. The keyboard prompts the user to create a new time slot, if no more time slots are available for the chosen estate. As such step 2 is conditional. The UI differs from what is seen in fig. 1 because user feedback revealed multiple flaws with the first versions of the UI. The most severe problem of the first version was that users did not comprehend the meaning of the coloured buttons. Another problem was that estates were not sorted in any way. The absence of any order lead to users trying to discern a meaning from the placement of the list items consequently causing them to make wrong assumptions about the functionality of the keyboard. These issues were fixed in the second version by introducing clearer labels for the buttons as well as sorting the list of estates alphabetically. While the second version was understood, its implementation revealed a lack of overview due to the limited space available for the keyboard. This lead to an effort to simplify the UI as seen in fig. 2. A usability test of the final prototype revealed that all eight participants tested could solve all tasks without prior instruction or training. All participants rated the custom keyboard (as part of the application) on a five-level Likert scale positive: 4 x “very good” and 4 x “good”.

The standard usage patterns of app and custom keyboard show that users only need the custom keyboard in certain circumstances and even then, not for long. One big restriction of the current

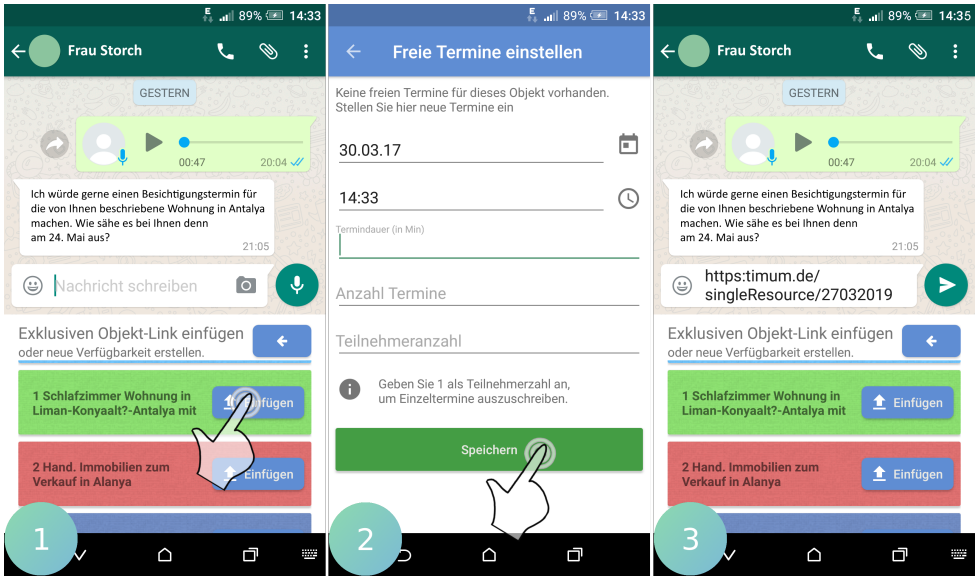

Figure 2: Insert a link. Step 2 is optional.

implementation of the custom keyboard is caused by limitations of the Android operating system: there is no easy way of switching between the standard keyboard and a custom keyboard. One possibility of solving this problem would be to develop an entire keyboard, including the functionalities of a "standard" keyboard. The custom keyboard would be one of multiple parts which can be directly accessed from within this new "standard" keyboard. A single additional button would allow the user to quickly switch to the custom keyboard. As soon as the user presses the paste button and after the paste operation is completed, the keyboard would automatically switch back. This would streamline the workflow of the users by enabling them to complete whatever they were typing immediately.

References

Coad, P., Lefevre, E. & De Luca, J. (1999). *Java Modeling in Color with UML: Enterprise Components and Process*. Upper Saddle River, NJ: Prentice Hall.

Döring, N. & Bortz, J. (2016). *Forschungsmethoden und Evaluation in den Sozial- und Humanwissenschaften*. Berlin: Springer.

Norman, D. & Draper, S. (1986). *User Centered System Design*. Hillsdale, NJ: Lawrence Erlbaum.

Roenspieß, A., Kindsmüller, M.C. & Herczeg, M. (2011). TeaCoMobile: Webbasierte Terminkoordination für Smartphones. In M. Eibl (Hrsg.), *überMEDIEN ÜBERmorgen – Tagungsbericht zur Mensch und Computer 201*. München: Oldenbourg, S. 293-296.

Müller, S. (2017). *User-Centred Design of a Mobile Application for Consensus Scheduling*. Unveröffentlichte Masterarbeit am Fachbereich Informatik und Medien der TH Brandenburg, Brandenburg an der Havel.