Breaking Barriers with Assistive Macros

André Rodrigues
LaSIGE, Faculdade de Ciências,
Universidade de Lisboa
Campo Grande, C6 Piso3, 1749-016
Lisboa, Portugal
afrodrigues@fc.ul.pt

ABSTRACT

People with disabilities rely on assistive technology (AT) software to interact with their mobile device. The overall functionality of AT depends on a set of requirements that are not always fulfilled by application developers, often resulting in cumbersome and slow interactions, or even render content inaccessible. To address these issues we present *Assistive Macros*, an accessibility service developed to enable users to perform a sequence of commands with a single selection. Macros can be created manually by the user or automatically by detecting repeated sequences of interactions. In this paper, we report a preliminary case study with our prototype involving a motor impaired user and his caregivers

Categories and Subject Descriptors

H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous; K4.2 [Computers and Society]: Social Issues – Assistive technologies for persons with disabilities.

General Terms

Performance, Design, Human Factors.

Keywords

Mobile, Multi-impairment, Assistive Technologies, Macros.

1. INTRODUCTION

Since the inclusion of accessibility software in modern touchscreen smartphones, such as screen readers (e.g. *Talkback*[1], *VoiceOver*[2]) and switch access (e.g. *Switch Control*[2], *Switch Access*[1]), they have gained popularity amongst people with visual and motor impairments. Smartphones have become a tool for independence and inclusion. To interact with this technology users often rely on alternative interaction methods that are provided by the accessibility software. Motor impaired users may rely on switches to navigate the interface and select options. In severe cases, navigation is accomplished relying on automatic scanning of the interface elements. Sequential navigation of elements can be extremely time consuming when performing a multiple step operation, even when using row-column or block scanning techniques. This is further exacerbated when interfaces have less relevant information scanned first. Previous research tried to tackle this issue with adaptive solutions that reordered interface items. However, this approach showed to be ineffective due to an increase in cognitive load [1].

ASSETS '15, October 26-28, 2015, Lisbon, Portugal

ACM 978-1-4503-3400-6/15/10.

Motor impaired users can face an insurmountable barrier when applications have options that can only be reached through a specific gestural interaction with no alternative selection method (e.g. app drawers). Since the underlying accessibility service to enable Switch Access also supports Screen Readers, blind and visually impaired (BVI) users can experience similar barriers and challenges when using mobile touchscreen devices.

To address the problems of time consuming navigation and inaccessible content, we developed *Assistive Macros*, an accessibility service that allows the creation of macros giving users the ability to perform a multiple step interaction with a single tap. *Assistive Macros* enables time-consuming or inaccessible interaction sequences to be recorded by the Person with Disability (PwD) or friend / family member. Once a macro has been captured, it can be accessed directly from the device's homescreen or from within the application; for which it provides functionality. Assistive Macros builds on the concepts presented in SWAT [3], a framework that enabled system-wide scanning navigation, leveraging its system-wide capabilities in the creation and reproduction of macros. *Macrodroid*[3] and *Tasker*[4] enable users to create macros based on a set of triggers and perform a restricted subset of device actions. However, these tools rely on developers integrating APIs and software hooks into their apps. Unlike the aforementioned tools, *Assistive Macros* requires no additional integration, and detects common interactions to suggest macros and enable richer macros that cross app boundaries and record actions in every application.

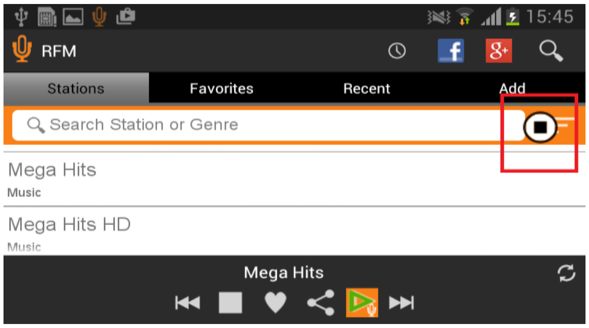

Figure 1- Overlay present over the interface to stop the macro manual recording.

[1] http://www.google.com/accessibility/
[2] https://www.apple.com/accessibility/ios/
[3] http://www.macrodroid.com/
[4] http://tasker.dinglisch.net/

2. ASSISTIVE MACROS

Assistive Macros is an accessibility service designed for Android that can be used with any application, and in conjunction with other accessibility services (e.g. Switch Access, Explore by Touch). Users are able to manually record macros, consisting in a sequence of actions for future reproduction. Moreover, *Assistive Macros* continuously monitors users' interactions to identify repetitive sequences and suggest the generation of new macros.

2.1 Manual Macro Creation

The manual macro creation can record two types of actions, description- or gesture-based. In description-based actions the system stores the selected interactive element's description and its position in the interface hierarchical tree; it relies on the unequivocal description of elements to perform a selection, which enables Assistive Macros to adapt to dynamic interfaces (e.g. select item XPTO in a list). Gesture-based recording requires a higher level of permissions than typical android applications/services. When recording a gesture action, the raw touches are collected, which includes their position and timestamps. Thus gesture-based macros can enable complex gestural interactions, from taps to multi-touch gestures, with possibly otherwise inaccessible interfaces. There is no limit to the amount of actions *Assistive Macros* can chain. Description- and gesture-based actions can be combined within the same macro.

Users, caregivers, and friends can record a macro using *Assistive Macros* bundled app. The recording process starts from the homescreen, where to record, users simply perform the desired chain of actions and press 'stop' to finish (Figure 1). Afterwards they name the macro and a shortcut is created in the homescreen.

2.2 Automatic Macro Generation

When enabled, *Assistive Macros* captures all interactive elements selection sequences that are performed by the user. There are two types of sequences: *session* and *app*. S*ession* sequence begins when the screen is turned on and ends when it goes off. From these, we generate *app* sequences. *App* sequences are created from splitting *session* sequences when the user enters an application. With both types, we apply the Rabin-Karp [2] string-matching algorithm to identify repeated sequences. When a sequence is repeated beyond a configurable threshold, the user receives a pop-up asking if they would like to create a macro for that sequence of actions. *Session* sequences detect common multi-step actions that start from the homescreen and might contain interactions with more than one app, while *App* sequences are restricted to a single app.

2.3 Macro Reproduction

The stored macros possess a sequence of descriptions and/or gestures to be performed during their execution. The reproduction of gestures actions is straightforward - the service injects the same touches with the recorded time intervals in between. When reproducing description actions, *Assistive Macros* maintains an updated list of all interactive items of the screen. When an action is to be performed *Assistive Macros* checks the available items; if the target item is found, it is selected. Macros have two selection modes: Icon-based and Overlay-based. Icon-based reproduction is executed from the homescreen where macros appear as shortcuts. Overlay-based reproduction is triggered when an app is launched and has macros associated.

3. CASE STUDY

We conducted a preliminary study with John, a multi-impaired smartphone user (motor-, visually-, and speech-impaired), currently able to operate a single-button switch. Handling the radio application on his device was a daily struggle and at times impossible; John had to scan eleven options and make four selections, in the best case scenario, to reach the play/stop button of his favorite station. Any mistake could easily double the time taken since one of the interfaces is a list of all stations. Furthermore, the play/stop buttons had no description making it impossible for John to know their location in the scanning sequence. With *Assistive Macros* early version, John's parents, after a five minute explanation, were able to record macros that allowed him to play and stop his favorite channel through a selection of a macro in the home screen. At the end, his mother showed enthusiasm with the possibility of creating personalized macros to help her son live a more independent life.

4. CONCLUSIONS

Manual macro creation allows users and caregivers to have an active role in improving the device accessibility by creating shortcuts that not only convert long interaction sequences in a single selection but also enable users to overcome inaccessible applications. Automatic macro creation allows them to incrementally improve their interaction efficiency by leveraging their routine behaviors. Gesture-based steps allow PwD to perform gestures they would not be able to do otherwise in order to manipulate the interface (e.g. open app drawers).

In the future, we will enrich our data with contextual information. A macro that is adequate in the morning might not be at night. If we are able to detect patterns and give users the opportunity to create context-specific macros we will be able to provide the right macro at the right time further decreasing navigation times. Furthermore, with *Assistive Macros*, we aim to explore the opportunities of crowd-sourcing generated macros for applications to provide immediate support to users upon their first interaction with the application. When an app is installed, users can take advantage of crowd-generated macros, which address some of its inherent accessibility problems. Finally we intend to conduct a user study involving more participants to evaluate additional features of *Assistive Macros*, including the automatic macro generation, and measure its effect on device usability and accessibility.

5. REFERENCES

[1] Horstmann Koester, H. & Levine, S. (1994). Learning and performance of able-bodied individuals using scanning systems with and without word prediction. Assistive Technology, 6, 42—53.

[2] Karp, Richard M.; Rabin, Michael O. (March 1987). "Efficient randomized pattern-matching algorithms". IBM Journal of Research and Development 31 (2): 249–260.

[3] Rodrigues, A., Guerreiro, T. SWAT: Mobile System-Wide Assistive technologies. Proc. of BCS HCI (2014)