# Accelerated Development for Accessible Apps – Model Driven Development of Transportation Apps for Visually Impaired People

Elmar Krainz, Johannes Feiner, Martin Fruhmann

## HAL Id: hal-01647715
## https://inria.hal.science/hal-01647715

Submitted on 24 Nov 2017

# Accelerated Development for Accessible Apps – Model Driven Development of Transportation Apps for Visually Impaired People

Elmar Krainz, Johannes Feiner, and Martin Fruhmann

FH JOANNEUM, Kapfenberg, Austria,
`firstname.lastname@fh-joanneum.at`
`http://www.fh-joanneum.at`

**Abstract.** Implementing usable and accessible user interfaces is a challenge, especially for mobile applications. App developers have to include accessibility in an additional step during the implementation, very often they overlook this extra workload.

There are concepts which combine Model Driven Development (MDD) for apps or semi-automatic support to create accessible software. But helpful tools to support accessibility features for apps during the implementation are hardly discussed in literature.

The aim of this paper is a concept of model-based software development for accessible apps. Within the domain of transportation apps, we provide a model to create an app scaffold with the required elements and accessibility features included from the beginning.

**Keywords:** Model driven Development, Mobile Apps, Accessibility

## 1 Introduction

Mobile applications are one of today's fastest growing software areas[1]. Apps, with simple intuitive user interfaces, are easy to use. The UI is mostly based on graphical representation a touch input.

For visually impaired and blind people this interaction paradigm is a challenge. Accessible web or desktop applications are already widespread, but this know-how cannot always be adopted to mobile apps.

Including accessibility is a challenge for developers and therefore it is handled more like a feature or add-on than a prerequisite. This work shows an approach to use Model-Driven Development (MDD) to create accessible mobile apps.

## 2 Related Work

The topics *Accessibility*, *Mobile Applications* and *Model-Driven Development* are part of current research. This is also visualised in Figure 2, which shows the

---

[1] `http://www.smartinsights.com/mobile-marketing/`
`mobile-marketing-analytics/mobile-marketing-statistics/`

overlapping domains of Model Driven Development (MDD), Accessibility and Mobile Development. There is already some exiting work which covers the outer intersections. For example, Accessibility *and* Mobile Development or Accessibilty *and* Mobile Applications have been covered by previous papers, but seldom all three fields of research are discussed together in the scientific community.
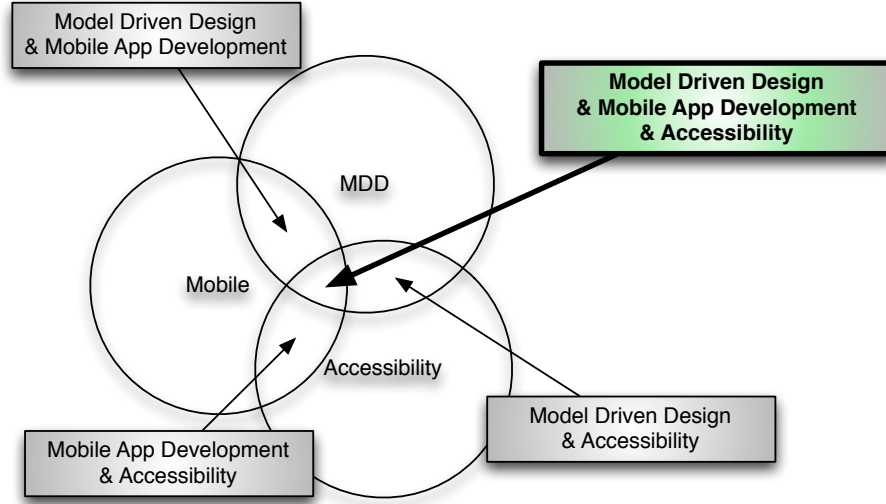


**Fig. 1.** The research focus is in the intersection of Model Driven Development (MDD), Mobile App Development, and Accessibility.

The main contribution of this work lays in the middle of the three circles, namely the intersection of all three domains. Following existing research is related to our approach:

The book Model driven Software Development (MDSD) from Voelter et al. covers the (semi-) automatic generation of software [20], but mentions mobile devices only in one minor example and accessibility is not covered at all. In the fast growing area of mobile applications there is also a need for this agile and effective methods to create software. There are different approaches. Barrnet et al [4] describes a tool to generate data-driven apps based on a Domain Specific Language (DSL, compare [7]), with $MD^2$ form Heitkoetter et al [11] even cross-platform apps are created from a model. The latter tool focuses on business apps where common source code is compiled into native iOS and Android apps, sadly neglecting accessibility topics.

MDD also helps to include accessibility in the development process. The *Johar* framework from Andrews and Hussain [1] uses an interface interpreter to create accessible applications for a range of users with different abilities. González-García et al [9, 8] use a model-based graphical editor tool to design

an accessible web based media player. The UsiXML presented by Vanderdonckt [19] promotes the integration and use of a User Interface Description Language (UIDL).

For accessibility combined with mobile app development, there are several publication which discuss smartphone-based assistive technologies for the blind. For example, Narasimhan et. al. [15] presented in 2009 the mobile phone tool *Trinetra* to assist visually impaired people during grocery shopping. Guidelines to develop for visually impaired and blind people can be found in [17]. Harder [10] discussed the possibilities for general mobility already in 1999, but for several modern approaches of implementation and design of touch-based smartphone interfaces see [12, 13, 4, 14]. These contributions focus on accessible way-finding apps as well as new ways of touchscreen usage for blind people.

## 3  A Concept for Model-Driven-Development for Accessible Apps

This work shows a concept to generate transportation apps for visually impaired and blind people. These people have special needs when it comes to live an independent life. Traveling and moving on their own is a very important factor [10]. Various apps are available to support this user group, but not every app provides an accessible user interface.

The innovative part of the empirical work in this contribution uses MDD to generate accessible apps. With the help of a meta model a concrete model of an transportation app is designed. Several features and workflows for the domain of transportation apps are included in this model. For example, most transportations apps need the functionality to fetch the current GPS position, which is part of the meta-model. Specifying *getposition* (see Section 3.1) in the model adds to appropriate code to the app. With an instance of this model an app scaffold is generated.

Based on best practices for accessible apps (see [12, 13, 4, 14]) and the adoption of common standards (see [5, 18]) required accessibility features are included from the beginning.

App programmers open the generated sources in the development environment (IDE) and can focus on the business logic which is not generated automatically. The resulting app has a simplified workflow and an accessible user interface (see screenshot in in Fig. 2).

### 3.1  Model

Concerning the model, we choose a domain specific approach. Which leads to a very specific and a straightforward model. We choose the domain of mobility and transportation apps. These kinds of apps help visually impaired, and disabled people to be more independent while travelling. For this reason a research on common transportation apps was necessary.

Transportation apps like Google Maps, OSM Routing or apps from different public transport companies have similar features. These are:
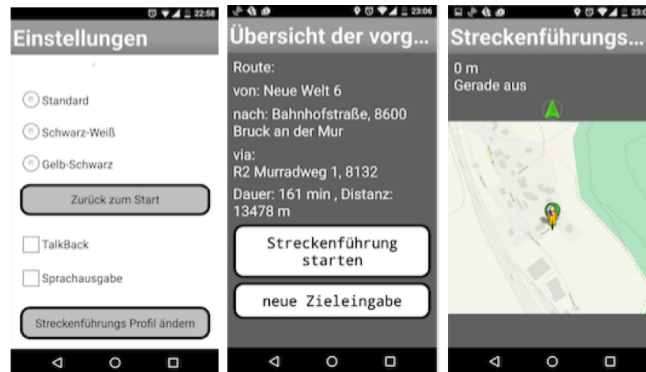
**Fig. 2.** Screenshot of the resulting routing app.

- Fetching your actual position
- Entering a target location
- Searching for a location
- Showing a route
- Display a map
- Receiving infos from public transport

These features and workflows within an app are defined in a model. With a defined notation developers are able to build the the scaffold of their own transportation app.

### 3.2   Modeling Language

Domain Specific Languages (DSL) help you to describe a model in your domain [7]. In this work an internal DSL is used. The language definition and the dedicated models are written in XML. We choose XML, because there are various tools available to create and process XML-based languages. But in the end it is more the less a personal decision, which language you use [6]. For better usability and non programmers a visual editor to create the model based on this DSL would be helpful.

*Example of an GPS position finder app:*

```
<?xml version="1.0" encoding="UTF-8"?>
<app appname="whereami"
        package ="at.fhj.modeling">
   <screen name="startscreen"
              transition="mainscreen">
   </screen>
   <screen name="mainscreen"
```

```
                transition="back" >
      <action function="getposition">
         <input type="button"/>
         <outout type="text"/>
      </action>
   </screen>
</app>
```

### 3.3   App Accessibility

People with special needs are nowadays able to operate standard computers without great difficulties. With the help of, for example, a braille display and build-in features of an operating system a visually restricted person can perform tasks on a PC. Visually impaired and blind people where also used to handle key-based mobile phones with screenreader software as add-on. With the uprise of touch-based smartphones these users had to find new ways of interaction.

On the popular mobile platforms iOS and Android are already some accessibility features included. The VoiceOver[2] system on iPhone allows a blind person to use a smartphone. On the Android platform TalkBack[3] is the integrated accessibility feature to react the user's touch input to voice output. But both systems can only provide an accessible user experience if an app is created properly. Developers have to follow the guidelines for iOS [3] or for Android [2]. Both platforms have some similarities but have also some big differences.

The Web Accessibility Initiative (WAI) of the W3C provides standards to improve accessibility for the Web. The actual standard for web applications is WCAG 2.0 [5] this guideline is based on general principles for accessible development rather then technical solutions. Therefore this standard provides a basis for non web solutions as well. Relevant principles of WCAG 2.0 [5] are:

- Perceivable
- Operable
- Understandable
- Robust

These principles are relevant to modern web applications but can be applied to non-web contents like mobile applications.

Currently a first public draft for a new standard for mobile accessibility [18] is available. This document describes how the principles and guidelines of existing standards like WCAG 2.0 can also be applied to native mobile apps.

In an (semi-) automatic generation process we have to consider platform independent principles and also platform related source code enhancements.

---

[2] http://www.apple.com/accessibility/ios/voiceover/
[3] http://www.google.com/accessibility/on-the-go.html

### 3.4   App Generation

With the model specified in Section 3.1 we are now able to generate an app scaffold for accessible transportation apps. This step, also known as transformation [20], is handled by a chain of tools. The first one in this chain is JAXB [16], whose task is to read the XML file and transform it into Java objects. Having the information stored in objects allows now to use it in combination with the Android command line tools to create an Android project scaffold. After this we are triggering a templating engine which generates the files, which are needed for the structure specified in the model. The process from XML-based model to a final app is illustrated in Fig. 3.4. The previous step also includes the mentioned accessibility features which are noticed in the Section 3.3.
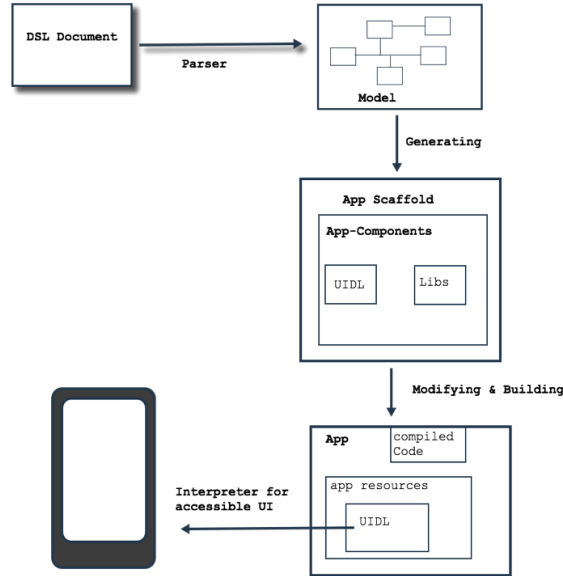
**Fig. 3.** Workflow from Meta-Model to Model to App Scaffold to final App.

The outcome from this transformation process is an app scaffold with accessibility features. All required app components (e.g start-activitiy) and libraries (e.g. routing-lib) are available in the app project. Further product flavours can be integrated in the app's source code. Features like content description for integrated screen reader support or active voice output in selected parts of the app are automatically generated.

## 4   Conclusion

In this paper we describe the combination of the topics accessibility, mobile apps and model-driven development. In the Section related work we looked at the intersection of these domains.

In the empirical part of this work we build a meta-model for the domain of transportation apps. With an instance of this model one can create a app-scaffold for accessible transportation apps. In the model-transformation process all main-screens, workflows and platform related accessibility features are produced. Developers can focus on the app features, because accessibility is build in from the beginning.

Further work will be the improvement of this concept concerning the utility and usability of the model generation and transformation.

## 5   Acknowledgments

## References

1. Andrews, J.H., Hussain, F.: Johar: A framework for developing accessible applications. In: Proc. 11th International ACM SIGACCESS Conference on Computers and Accessibility. pp. 243–244. ASSETS 2009, ACM (Oct 2009)
2. Android: Accessibility - usability - google design guidelines, `http://www.google.com/design/spec/usability/accessibility.html`
3. Apple: Accessibility on ios, `https://developer.apple.com/accessibility/ios/`
4. Barnett, S., Vasa, R., Grundy, J.: Bootstrapping mobile app development. In: Proc. 37th International Conference on Software Engineering - Volume 2. pp. 657–660. ICSE 2015, IEEE Press, Piscataway, NJ, USA (2015)
5. Caldwell, B., Reid, L.G., Cooper, M., Vanderheiden, G.: Web content accessibility guidelines (WCAG) 2.0. W3C recommendation, W3C (Dec 2008), http://www.w3.org/TR/2008/REC-WCAG20-20081211/
6. Fowler, M.: Parserfear (2008), `http://martinfowler.com/bliki/ParserFear.html`
7. Fowler, M.: Domain-specific languages. Pearson Education (2010)
8. González-García, M., Moreno, L., Martínez, P.: A model-based tool to develop an accessible media player. In: Proc. 17th International ACM SIGACCESS Conference on Computers & Accessibility. pp. 415–416. ASSETS 2015, ACM, New York, NY, USA (2015)
9. González-García, M., Moreno, L., Martínez, P., Miñon, R., Abascal, J.: A model-based graphical editor to design accessible media players. J. UCS 19(18), 2656–2676 (2013)
10. Harder, A., Kasten, E., Sabel, B.A.: Möglichkeiten der mobilität blinder menschen. Aktuelle Augenheilkunde 2, 8–13 (1999), `http://www.med.uni-magdeburg.de/~harder/mob1/mob1.html`

11. Heitkötter, H., Majchrzak, T.A., Kuchen, H.: Cross-platform model-driven development of mobile applications with md$^2$. In: Proc 28$^{th}$ Annual ACM Symposium on Applied Computing. pp. 526–533. SAC 2013, ACM (2013)
12. Krajnc, E., Feiner, J., Schmidt, S.: User centered interaction design for mobile applications focused on visually impaired and blind people. HCI in Work and Learning, Life and Leisure pp. 195–202 (2010)
13. Krajnc, E., Knoll, M., Feiner, J., Traar, M.: A touch sensitive user interface approach on smartphones for visually impaired and blind persons. Information Quality in e-Health pp. 585–594 (2011)
14. Mattheiss, E., Krajnc, E.: Route descriptions in advance and turn-by-turn instructions-usability evaluation of a navigational system for visually impaired and blind people in public transport. Human Factors in Computing and Informatics pp. 284–295 (2013)
15. Narasimhan, P., Gandhi, R., Rossi, D.: Smartphone-based assistive technologies for the blind. In: Proc. International Conference on Compilers, Architecture, and Synthesis for Embedded Systems. pp. 223–232. CASES 2009, ACM, New York, NY, USA (2009)
16. Ort, E., Mehta, B.: Java architecture for xml binding (JAXB). Sun Developer Network (Mar 2003), `http://www.oracle.com/technetwork/articles/javase/index-140168.html`
17. Park, K., Goh, T., So, H.J.: Toward accessible mobile application design: Developing mobile application accessibility guidelines for people with visual impairment. In: Proc. of Conference on Human Computer Interaction Korea. pp. 31–38. HCIK 2015, Hanbit Media, Inc., South Korea (dec 2014)
18. Patch, K., Spellman, J., Wahlbin, K.: Mobile accessibility: How wcag 2.0 and other w3c/wai guidelines apply to mobile. Tech. rep., W3C (2015)
19. Vanderdonckt, J., Limbourg, Q., Michotte, B., Bouillon, L., Trevisan, D., Florins, M.: Usixml: a user interface description language for specifying multimodal user interfaces. In: Proceedings of W3C Workshop on Multimodal Interaction WMI. vol. 2004 (2004)
20. Völter, M., Stahl, T., Bettin, J., Haase, A., Helsen, S.: Model-driven software development: technology, engineering, management. John Wiley & Sons (2013)