Rapid creation of sketch-based native Android prototypes with Blended Prototyping

Benjamin Bähr
QU-Labs / T-Labs / TU Berlin
Ernst-Reuter-Pl. 7; 10587 Berlin
baehrben@mailbox.tu-berlin.de

ABSTRACT

Blended Prototyping is a new approach for the development of interface ideas for mobile applications. It adapts concepts of the established Paper Based Prototyping approach and conveys them to an application in the mobile context. In contrast to existing sketch-oriented approaches, Blended Prototyping preserves the collaborative sketching with pen and paper as a central technique of the design process. On the basis of hand made drawings, the system generates mobile applications that show a sketched design, however run on the target devices as apps with their own functionality. To achieve this, the approach establishes a new concept for an agile native programming of Android applications, which could be easily adapted in other solutions. After the concept of Blended Prototyping and its design instruments were discussed in previous publications, this text describes new components that are used for the actual creation and testing of Blended Prototypes. Experiences with the practical system use are discussed and plans for the further system development are displayed.

Author Keywords

Mobile Interface Prototyping; Paper Prototyping; app development; agile app development; app design.

ACM Classification Keywords

H.5.2. [Information interfaces and presentation]: User Interfaces. – Prototyping, User-Centered Design.

General Terms

Human Factors; Design; Languages; Experimentation.

INTRODUCTION

Paper based prototyping

Paper based prototyping is a method, in which user interface prototypes are developed and tested on the base of paper drawings. Paper based prototypes are meant to have a non-perfect look and a reduced sketch like design. This simplicity plays out its advantages for both, the prototype creation and testing. Drafting ideas in team work with the help of pen and paper has proved to be a natural process, which provides a common ground for teams of different disciplines to come together and collaboratively work on creative tasks [4]. The proceeding allows designers to work out and discuss their ideas collaboratively, without the distractions and limitations that typically occur with desktop computer software [9]. Since paper drawings can be created naturally without previous training, team members of different professional background can be well integrated into the design process.

Paper based prototypes are tested in a Wizard-of-Oz procedure, solely on the basis of paper sketches. This allows a quick production and adjustment of prototypes and makes fast test iterations possible, already in earliest design stages. In addition to these advantages for the prototype creation process, the paper based prototyping approach is widely accepted for its positive effects on the prototype testing. Different studies [8,9] show that test users regularly articulate their opinion on unfinished looking prototypes in a more free and honest way, than they do on finished looking products. Moreover, sketches offer designers a way of abstraction that allow them to communicate their questions in a more targeted way [8].

Paper based mobile prototypes

Mobile applications have to meet a number of specific challenges that result from their use in different mobile situations [7]. Therefore they should not solely be tested in stationary laboratory conditions, but be surveyed in their real use contexts [6]. A direct test of paper prototypes as insertion slide-ins in device dummies proved to be problematic [3,10], since neither the tests could be properly surveyed, nor the user-software interaction was properly employed.

To exploit the advantages of paper-based-prototyping in the context of mobile applications, several approaches create sketchy-looking applications that run directly on the target devices. *Axure*, *Balsamiq*, or *Mockflow* are prominent examples for the transformation of this idea into professional software tools.

PID-MAD 2013, Aug 27 2013, Munich, Germany.
(In Conjunction with *MobileHCI '13*, Aug 27-30 2013, Munich, Germany.)

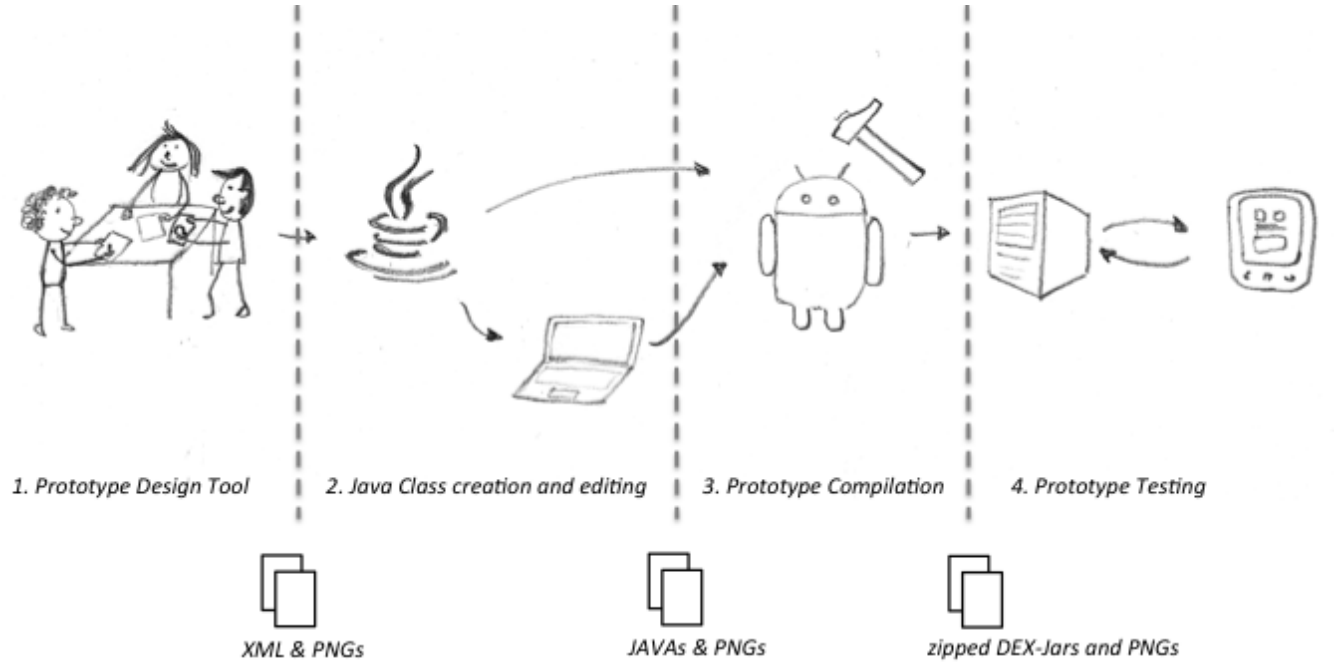

Figure 1: The Blended Prototyping modules and their exchange standards

Here however the prototypes are designed within a desktop software, where the use of mouse and keyboard restrain the collaborative creative work. To better exploit the advantages of the paper based prototyping approach for the design process, the Blended Prototyping platform uses an overhead projection tabletop computing setup. Here, regular paper pen sketches are used as a key element in the design process, providing groups of users a familiar basis to draft and discuss their interface ideas.

To define prototype behavior, existing sketch-like prototyping approaches either use web-technologies like Html5-Javascript combinations, or software specific pseudo code. Such prototypes are run within a prototype player installed on the mobile device, which adapts the code to the specific conditions of the hosting OS. This way, the same prototype can be tested on different platforms without further adjustments.

The technique to use universal programming languages bears however two big disadvantages: *Firstly*, not all system classes can be mapped to unified prototyping language; be it for technical or security reasons. Therefore prototypes created this way are to a certain degree limited in their functionality. *Secondly*, the prototype code developed with unified technologies cannot be used directly in the product development, but must be translated to the domestic language. Such translation can easily bring very burdensome revisions.

Due to these reasons, and in contrast to other existing sketch-like techniques, Blended prototyping follows a native programming approach. Its prototypes are written in Android specific Java code. This way, Blended Prototyping follows a *Throw Away Prototyping* [5] approach in its design, which mainly serves to generate insight out of its testing, however looses the value itself after the testing. At the same time, Blended Prototyping follows an *Evolutionary Prototyping* [5] approach in the prototypes' function development, where the developed code is adjusted and reused in the subsequent iteration cycles, until it more and more resembles the later product.

Blended Prototyping can therefore be applied in the earliest design phases already, to deliver fast and easy click-dummies; the approach is however useful in later development phases as well, where tests with prototypes of more elaborated functionality are needed.

BLENDED PROTOTYPE GENERATION

The Blended Prototyping platform[1] includes tools, which allow a generation of prototypes that can be ran directly on Android devices. The prototypes are written in Android specific Java. Therefore, firstly the code generated in the prototyping process can be reused directly in the later product, and secondly a substitution language does not limit the prototypes' functional capabilities.

As shown above in Figure 1, the Blended Prototyping platform can be structured into four separated modules that are connected with defined interfaces: The prototype design, the Java class creation and editing, the prototype compilation, and the prototype testing. Each of the modules could be principally substituted with alternative solutions,

[1] www.blended-prototpyping.de

as long as those comply with the given Blended Prototyping exchange standards.

Prototype design tool

The design tool, described in detail before in [1,2], uses a tabletop computing environment to allow groups of designers to sketch their interface ideas on paper sheets. With the help of a high-resolution camera the sketches are transformed to digital versions. User control widgets are positioned within the sketched interface with the help of mobile devices that are part of the setup. Depending on the widgets nature, specific parameters are described. Buttons for example can be defined with linkage paths, which are then used to display an interface storyboard on the table surface.

Java class creation and editing

Blended Prototyping uses generated Java classes that allow developers to advance a prototype's functionality with programming code. The XML file created by the design tool serves as a basis for this procedure. Here, the single interface screens that were regarded in the design process are listed, and user widgets that were defined on their surface are explained.

In the class creation process, each screen is translated into its own class, where all user controls are embedded as objects. Depending on the user control's type, different event handlers are generated automatically within the classes. Here *// TODO* tags mark the locations, were code is typically edited. The generated classes are delivered as a part of an Eclipse[2] project, where all necessary dependencies are embedded, so that the code editing can be immediately started. The class code is designed to be as simple and clearly structured as possible, so that little programming knowledge is needed to implement some core functionality.

Principally, the code editing process can be skipped and a prototype is created fully automatically. In this case, the behavior embedded in the prototypes is very simple. Initial click-dummies however can be created in this automated fashion.

Prototype Compilation

After the Java classes are completed they are compiled to binaries by the Android specific DEX compiler. These binaries, and all prototype screen images, are then packed into a zip file that is stored on the server. User studies are organized in a central server database, where user login-keys are created and assigned to specific prototypes.

[2] *Eclipse-IDE: www.eclipse.org*

Prototype Testing

All test users download the same Blended Prototyping Player app. The app itself downloads user specific prototype data from the server. Using the techniques of reflection, this data is dynamically interpreted in the mobile app. Hence the Blended Prototyping Player app is able to consider program code that was created after its creation or installation. This gives developers the freedom to adjust and change their prototype seamlessly, without the users' notice.

The prototype player provides functions to log the user interaction with the prototype automatically. Each interaction with a control is stored with a timestamp in the app's database, which is uploaded in an aggregated form to the server, when the device's available bandwidth is high enough. The usage data accumulated on the server can be aggregated to CSV files that can be easily imported in common statistic software.

BLENDED PROTOTPYING IN PRACTICE

The Blended Prototyping platform was tested in the context of a professional app development project. A team of two programmers, a designer, and a project manager used the system to design initial interface prototypes for an app development project on shared interactive workspaces. The Blended Prototyping environment was used in four design sessions, of which each produced a new prototype version. Unfortunately, the prototypes were not tested in large-scale user tests, but rather in personal use and demos in front of colleagues or advisory board members. After an initial automatically generated click-dummy, the code editing procedures explained above were used in redesign sessions, to produce prototypes with an enriched functionality. These tasks were primarily conducted by the two programmers. However, the designer, who considered herself to be a programming amateur, repeatedly adjusted the code to do some separate design studies in between the bigger collaborative sessions. Unfortunately, the development project was discarded due to reasons that were not related to the prototyping platform. Therefore the development did solely cover early design stages.

Feedback on the Blended Prototyping platform was generated in interviews and presented a generally positive picture. Depending on their professional background, the development team members underlined different positive aspects of the system. The programmers were impressed by the platform's capability to develop native code that could be loaded dynamically into a prototype, without any need of reinstallations on the devices. The designer however was especially excited about the design process and the ways to edit prototype code. She described these as "clear and simple" and stated that they were easier for her to follow than code of regular development projects.

CONCLUSION AND FUTURE WORK

The Blended Prototyping platform aims to provide developers of mobile applications with mechanisms, to test mobile user interfaces as quickly and easily as possible. At the same time, the approach offers tools to implement elaborated behavior within a prototype, so that functional prototype aspects can be reused in later development stages and the final product itself. A 'real world' application of the first platform version generated a generally positive feedback. The tool shows to be well suited to promote collaborative work that invites both, experts as well as less experienced team members to participate equally in a creative discussion. The code editing mechanisms allow a native programming of code, which can be loaded dynamically into already deployed applications. This gives development teams the flexibility to adjust and update their prototypes whenever they like, without the users' notice.

We are currently working on the development of a general prototyping requirements framework that defines objectives and subtasks for mobile interface prototyping approaches. It will be iteratively developed and evaluated in expert reviews. The framework will then be used as an objective measure for the evaluation and comparison of the Blended Prototyping approach towards other developments, and therefore will serve as an important basis for the further system improvement.

The Blended Prototyping system is structured in separate modules that are connected through well-defined standards. Single modules can therefore be easily substituted and adjusted to the certain needs of other design or research projects. We would be happy to provide our tools to other facilities, and to build cooperation projects to further investigate practices and developments in mobile interface prototyping.

REFERENCES

1. Bähr, B., Dix, A., Joost, G., et al. *Prototype! physical, virtual, hybrid, smart: tackling new challenges in design & engineering*. form + zweck, 2012.
2. Bähr, B., Kratz, S., and Rohs, M. A Tabletop System for supporting Paper Prototyping of Mobile Interfaces. 2010.
3. Hendry, D.G., Mackenzie, S., Kurth, A., Spielberg, F., and Larkin, J. Evaluating paper prototypes on the street. *CHI '05 extended abstracts on Human factors in computing systems*, ACM (2005), 1447–1450.
4. Klemmer Scott, Everitt Katherine, and Landay James. Integrating Physical and Digital Interactions on Walls for Fluid Design Collaboration. *Human-Computer Interaction 23*, 2 (2008), 138–213.
5. Kordon, F. and Luqi. An Introduction to Rapid System Prototyping. *IEEE Trans. Softw. Eng. 28*, 9 (2002), 817–821.
6. Nielsen, C.M., Overgaard, M., Pedersen, M.B., Stage, J., and Stenild, S. It's worth the hassle!: the added value of evaluating the usability of mobile systems in the field. *NordiCHI '06*, ACM (2006), 272–280.
7. De Sá, M. and Carriço, L. Defining scenarios for mobile design and evaluation. *CHI '08 extended abstracts on Human factors in computing systems*, ACM (2008), 2847–2852.
8. Schumann, J., Strothotte, T., Laser, S., and Raab, A. Assessing the effect of non-photorealistic rendered images in CAD. *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*, ACM (1996), 35–41.
9. Snyder, C. *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces (The Morgan Kaufmann Series in Interactive Technologies)*. Morgan Kaufmann, 2003.
10. Svanaes, D. and Seland, G. Putting the users center stage: role playing and low-fi prototyping enable end users to design mobile systems. *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (2004), 479–486.