Information Concepts for Cross-device Applications

Michael Nebeling, Christoph Zimmerli, Maria Husmann,
David Simmen and Moira C. Norrie
Institute of Information Systems, ETH Zurich
CH-8092 Zurich, Switzerland
{nebeling|zimmerli|husmann|simmen|norrie}@inf.ethz.ch

ABSTRACT

Cross-device application development poses different requirements ranging from the adaptation and distribution of user interfaces, over the migration of application state to the management of data as it is moved around and shared between devices. In particular, the evaluation of multi-device user interfaces becomes a challenge when they are distributed across a range of interaction resources. We present our ongoing work towards a platform for the design and evaluation of cross-device applications that promotes the central concept of a *cross-device session* useful for both design and evaluation. Our concept enables the tracking and management of the *data* part of a distributed, multi-device application as it is viewed and updated on different devices as well as the *metadata* describing the interactions and how the data evolved as a result of them. We illustrate the benefits of our approach for a first cross-device application and discuss how we plan to address the remaining challenges.

Keywords

Multi-user/multi-device interaction; cross-device development platform; session concept.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Input devices and strategies, Interaction styles*

1. INTRODUCTION

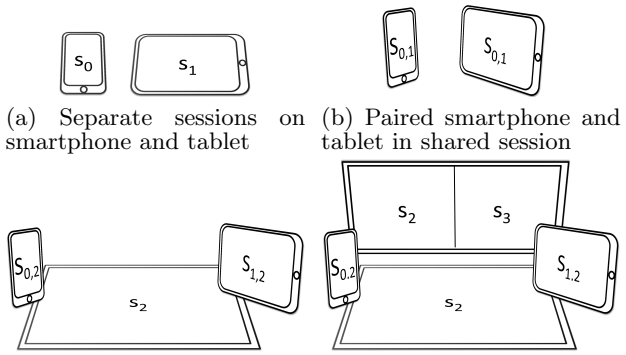
It has become increasingly common that users perform their tasks using various devices, ranging from traditional desktop computers to new types of mobile devices with different multi-modal interaction resources. Yet, most applications today still assume a single device for carrying out the tasks. Recent research has started to focus on settings around multiple devices, developing new forms of interaction in both individual and collaborative usage scenarios [1, 3]. However, this requires new design guidelines and techniques for sharing information or even devices, which is generally not supported by current applications.

Figure 1: Multi-user, multi-device scenario around a tabletop—we present information concepts for the design and evaluation of cross-device applications

Cross-device development requires design decisions and implementation effort on different layers, involving the adaptation, distribution and migration of both user interfaces and data across devices. Several frameworks have been proposed to support multi-device development based on different model-based [6], object-oriented [3] and data-oriented approaches [1]. The common goal to reduce design effort is often anticipated by using different abstraction layers and models to describe the user interface and interactions [9]. However, in our own experience and based on a systematic literature review, we have identified three major problems:

Need for Rapid Prototyping Tools. Existing solutions provide little support for rapid prototyping of multi-device, distributed user interfaces. Rather, the focus is on powerful and expressive models to support systematic multi-device development. However, the increased flexibility of cross-device applications in terms of the use context often requires experimentation with alternative designs. Current GUI builders support the design of user interfaces for a single device, but provide no specific support for user interfaces distributed over multiple devices. There are partial solutions that focus on either adapting interfaces for different devices [4] *or* distributing them across devices [5]. The exceptions are model-based, multi-device authoring environments which however start from an abstract representation rather than concrete interface more common to designers [9].

DUI 2013: 3rd Workshop on Distributed User Interfaces: Models, Methods and Tools. In conjunction with ACM EICS 2013
June 24th, 2013. London, UK.
ISBN-10: 84-616-4792-0
ISBN-13: 978-84-616-4792-7



(a) Separate sessions on smartphone and tablet

(b) Paired smartphone and tablet in shared session

(c) Mobile devices paired with tabletop in shared session

(d) Mobile devices paired with tabletop and wall-size display in parallel session

Figure 2: Different scenarios and session modes

Different Software Architectures and Implementation Methods. There is an increased proliferation of new devices with very different hardware and software characteristics. Developers hence require a wide range of skill sets and programming experience with different languages and software development kits. Many solutions have been built for a fixed environment to support specific interaction techniques [2]. More dynamic support for handling additional devices often requires applications to be built using special middleware and programming models [1, 3]. An exception is the service-oriented architecture in [10] that is however constrained to message-based applications. Web-based approaches to overcome device differences are promising, but often constrained by current browser support [7].

Limited Support for User Studies. Finally, while recent research tends to focus on exploring new cross-device interaction techniques, the current state is to build special instruments for each study [2, 11]. There is insufficient tool support for user studies spanning multiple devices in co-located, remote or mixed usage scenarios.

In this paper, we present our ongoing work towards a cross-device development platform supporting pre-configured physical spaces as well as ad-hoc, dynamic "walk-up and share" mobile scenarios. The goal is to provide flexible support for interactive development on the target devices themselves and dynamic distribution with additional devices. In a first step, this paper presents a novel concept for session management that we have started to exploit for enabling both the design and evaluation of cross-device applications. Our concept facilitates different modes for sharing and managing interfaces as well as data across multiple devices monitoring the data evolution in a central history. We will sketch how it can also be configured for user testing in mixed co-located and asynchronous, multi-device scenarios, and how we plan to support studies based on session playback and analysis.

2. SCENARIOS

There are many possible scenarios in which multiple devices could be used to achieve a task—individually or in collaboration, with one or multiple devices per user [9]. We use the setting in Figure 1 to motivate our session concept.

Figure 2a illustrates a first scenario in which the smartphone and tablet are used individually with no interaction across devices. Here, each device is configured with its own session, S_0 and S_1, manipulating data in isolation.

Figure 2b illustrates a second scenario in which the two mobile devices are paired to participate in a shared Session $S_{0,1}$. As a result, the interactions carried out with one device are no longer independent of the other. The session can be configured so that the interface is distributed across the devices. In this scenario, it is possible that users manipulate their own data or the same data part of the shared session.

Based on this scenario, Figure 2c illustrates another one in which both mobile devices are instead connected to the tabletop and therefore indirectly part of the same Session S_2. Here, the mobile devices effectively function as a remote control and interactions on either device are reflected on the tabletop. However, direct manipulations on the tabletop are not synchronised with Sessions S_0 and S_1. This could be enabled by pairing the tabletop as in the previous scenarios.

Figure 2d extends on the previous scenario in that all manipulations of Session S_2—be it through interactions on the mobile devices or the tabletop itself—are now also reflected on the wall-size display paired with the tabletop. The wall display is also running a separate Session S_3 in parallel.

Likewise, the person sitting with the laptop on the left in Figure 1 may follow the actions performed on the table from her device by configuring Session S_2, and could also do this remotely.

To facilitate user studies, each session could be configured to record the input and performed actions on each device. Additionally, devices such as Kinect for 3D skeletal tracking could be configured as part of Session S_2 and record the user interactions around the tabletop in physical space.

3. MODEL AND ARCHITECTURE

The goal of our platform is to enable the design, distribution and evaluation of multi-device applications—both at the user interface and data level—in a uniform way based on the session concept. Below we present a simple model of the concepts and how they relate to each other. We also present the features of our platform and the architecture of applications. This is followed by an example of how an existing single-device application could be extended for multi-device scenarios based on the concepts.

3.1 Model

As illustrated in Figure 3, our concept of a Session S is defined as $S =< U, D, I >$ linking the concepts of User U, Device D and Information I represented as data and metadata. Data refers to the information managed in a session, while metadata describes the interactions and how the data was manipulated during the session.

Figure 3 depicts the core model. Sessions are created and maintained by one or multiple users for one or multiple devices. A `Session` is by default public and accessible to everyone. The platform supports a `PrivateSession` where the initiating user can directly control which other users get access to it. All interactions with the data of an application are made through sessions. For each interaction, our platform

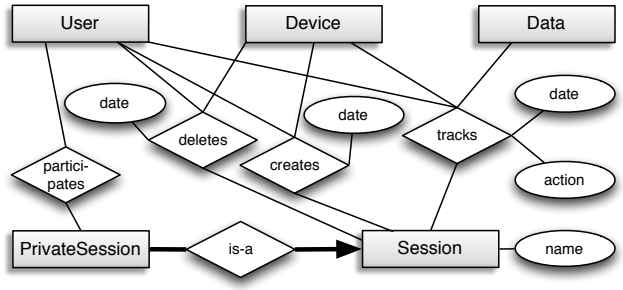


Figure 3: Core model linking the different concepts behind our platform

Action	Description
add	adds data to this session
edit	change the data in this session
copy	add the data to another session
delete	delete the data in this session
move	copy the data and delete it in this session
log	write arbitrary data to the session log

Table 1: Session API

tracks the `User`, `Device` and `Data` with `date` and `action`. The possible actions are summarised in Table 1. The type of data is application-specific and the actual data is not manipulated by our platform. However, the platform enforces a unique identifier for each data instance, while edit actions create new versions of the data. Since the id of a resource stays from creation on over all edit actions and multiple sessions, we are able to trace the evolution of all data in the application over time. Sessions can be managed through the Session API. Note that, in order to keep up the traceability of all actions (e.g. when the session is configured for user studies), the session and its data will never actually be deleted, but only marked as deleted if requested.

3.2 Features

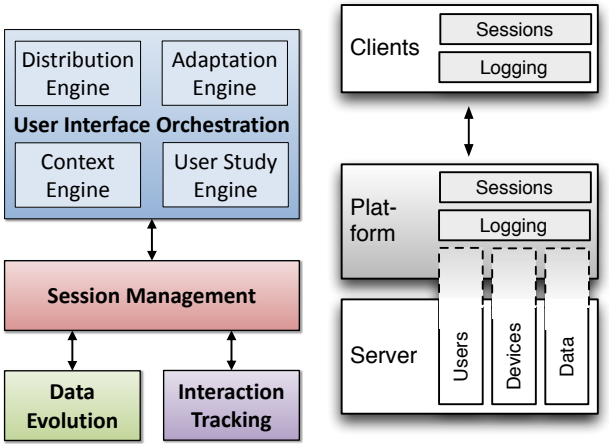
Based on our session concept, our platform provides the following features for cross-device design and evaluation.

Simple User and Device Registration. Our platform enables simple registration of users and devices. For ad-hoc scenarios, automatic device detection is supported based on a device description repository.

Each session involves at least one device, but multiple devices can participate in the same session. As illustrated in Figure 2d, it is also possible to control multiple sessions from a single device, allowing interactions between sessions.

Implicit and Explicit Session Management. By default, session management is implicit, e.g. each device runs its own session. The platform supports explicit session management, e.g. to allow certain users and devices to view and participate in other sessions. This also supports asynchronous and remote collaboration.

Evolution Tracking and History. The metadata collected as a result of the interactions during a session can be used for tracking the data evolution across multiple users and devices. The history is useful for version management and migration of data between both devices and users.



(a) Platform components (b) Extended architecture

Figure 4: Platform and application architecture

Custom Logging and Session Playback. Our platform supports custom logging of actions during a session to track additional events not directly triggered by the application. This can for example be useful when additional devices such as Kinect are used for tracking additional information like the user interaction in physical space.

The platform also supports session playback. This feature was primarily developed to migrate the data between users and devices and for asynchronous and remote scenarios. However, it is also useful for user studies allowing evaluators, not only to trace the data and how it evolved, but also to view the interactions across multiple devices.

Interaction Tracking and Analysis. In related projects, we have started to develop novel interaction tracking and analysis tools that support multi-device development in particular for mobile touch devices [7, 8]. We are currently working on integrating the techniques with our platform.

3.3 Architecture

To support multi-device design and evaluation, our platform consists of the four major components illustrated in Figure 4a. First, **user interface orchestration** is enabled based on a **context engine** for keeping track of users and devices, an **adaptation engine** for adapting interfaces for different form factors and input modalities, a **distribution engine** for rendering (parts of) an interface on different devices and a **user study engine** for conducting user evaluations in multi-device scenarios. Second, the **data evolution** component provides basic functionality for monitoring the application state and keeping a history of the data. Third, the **interaction tracking** component offers techniques for monitoring user behaviour. Finally, the **session management** component at the core of the platform is used to feed the different components with the use context information.

The platform acts as a middleware to track and maintain the data on different devices independent of the application. Figure 4b illustrates the extension of an existing application with our platform. Assumed is a client/server application managing some kind of data. Linking the application to

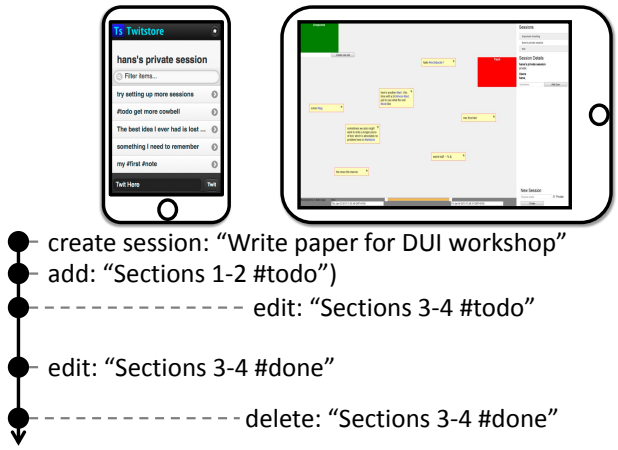


Figure 5: Extended note-taking application with session-based management of notes between users and devices

our platform requires a mapping from the concepts of user, device and data in the platform to available counterparts in the application. If some concept is not present in the application, it is managed solely by the platform. In order to fully profit from the advanced features and logging facilities offered by the platform, clients need to use the Session API.

3.4 Implementation

Our platform is currently implemented based on web technologies using jQuery in combination with jQMultiTouch [7] on the client side and a Java server with an object database backend. The Session API is exposed via a REST interface supporting asynchronous communication using AJAX. MobileESP[1] is used for automatic device detection.

4. FIRST APPLICATION

As a first proof-of-concept, we extended an existing note-taking application developed in another project with the concepts presented in the previous sections. The application already had the notion of users and their private notes with data being stored on a server and two different web-based clients. One client is targeted at mobile devices with smaller screens and shows the notes in list form. The other is intended for desktop and tabletop settings allowing users to organise and group notes spatially on the screen. Both clients and the server required minimal extensions enabling multi-device sessions as illustrated in Figure 5.

So far, we have used the extended application in group meetings and informally tested it with multiple users and different devices such as laptops, tablets, smartphones and a tabletop similar to the scenario in Figure 2c. We have observed how notes are often taken and managed in private sessions before and after meetings, typically while on the go. During the meetings themselves, selected notes were then mostly shared and discussed with others in a shared session on the table. The notes on the table were frequently refined, merged and sometimes also discarded independent of the private copies. Towards the end of meetings, the changes and notes created in the meeting were then copied over from the common to private sessions and merged as necessary. Our platform will enable more formal user studies and allow us to develop and test new multi-device features.

[1]http://blog.mobileesp.com

5. CONCLUSION AND FUTURE WORK

We have presented a cross-device session concept and our first prototype of a platform that supports the design, distribution and evaluation of multi-device applications—both at the user interface and data level. Our approach integrates the different concerns in contrast to existing works that tend to focus only on one of these aspects. For example, [9] only considers aspects related to the user interface, and [3] assumes a single information landscape displayed by all devices. Our first application allows users to manage information separately and arbitrarily connect devices for ad-hoc sharing. Implementing a first multi-device scenario allowed us to overcome many challenges, but several parts remain as future work. We are currently working on a cross-device GUI builder and tools enabling interactive development of new and simple adaptation of existing applications for multi-device environments. Particular attention is paid to the programming model, where we make sessions a first-class concept for static, dynamic and mixed distribution.

6. REFERENCES

[1] T. Gjerlufsen, C. N. Klokmose, J. Eagan, C. Pillias, and M. Beaudouin-Lafon. Shared Substance: Developing Flexible Multi-Surface Applications. In *Proc. CHI*, 2011.

[2] M. Haller, J. Leitner, T. Seifried, J. R. Wallace, S. D. Scott, C. Richter, P. Brandl, A. Gokcezade, and S. E. Hunter. The NICE discussion room: integrating paper and digital media to support co-located group meetings. In *Proc. CHI*, 2010.

[3] H.-C. Jetter, M. Zöllner, J. Gerken, and H. Reiterer. Design and Implementation of Post-WIMP Distributed User Interfaces with ZOIL. *IJHCI*, 28(11), 2012.

[4] J. Lin and J. A. Landay. Employing Patterns and Layers for Early-Stage Design and Prototyping of Cross-Device User Interfaces. In *Proc. CHI*, 2008.

[5] J. Melchior, D. Grolaux, J. Vanderdonckt, and P. V. Roy. A Toolkit for Peer-to-Peer Distributed User Interfaces: Concepts, Implementation, and Applications. In *Proc. EICS*, 2009.

[6] J. Melchior, J. Vanderdonckt, and P. V. Roy. A Model-Based Approach for Distributed User Interfaces. In *Proc. EICS*, 2011.

[7] M. Nebeling and M. C. Norrie. jQMultiTouch: Lightweight Toolkit and Development Framework for Multi-touch/Multi-device Web Interfaces. In *Proc. EICS*, 2012.

[8] M. Nebeling, M. Speicher, and M. C. Norrie. W3Touch: Metrics-based Web Page Adaptation for Touch. In *Proc. CHI*, 2013.

[9] F. Paternò and C. Santoro. A Logical Framework for Multi-Device User Interfaces. In *Proc. EICS*, 2012.

[10] J. S. Pierce and J. Nichols. An Infrastructure for Extending Applications' User Experiences Across Multiple Personal Devices. In *Proc. UIST*, 2008.

[11] T. Seyed, C. Burns, M. C. Sousa, F. Maurer, and A. Tang. Eliciting Usable Gestures for Multi-Display Environments. In *Proc. ITS*, 2012.