INTERNATIONAL JOURNAL OF HUMAN–COMPUTER INTERACTION 2020, VOL. 36, NO. 6, 536–552
https://doi.org/10.1080/10447318.2019.1662667

A Toolkit for Prototyping Tabletop-Centric Cross-Device Interaction

Zheng Huang and Jun Kong

Department of Computer Science, North Dakota State University, Fargo, North Dakota, USA

ABSTRACT

Different types of devices are used in our daily lives, and each type has its own advantages and disadvantages. Cross-device interaction that involves multiple devices can potentially overcome each device's inherent disadvantages. Without a toolkit, it is time consuming to develop cross-device interaction. Focusing on a horizontal large display (called a tabletop), this paper proposes a generic toolkit for prototyping tabletop-centric cross-device applications that involve a large display and multiple smartphones In our toolkit, a user uses a smartphone as a look-through lens for browsing and selecting objects on a tabletop, and remotely manipulates the selected object with multimodal feedback. Based on the above interaction style, our toolkit formalizes the development of cross-device interaction as defining a mobile interface on each tabletop interface object. Since our toolkit encapsulates the interface distribution and synchronization through message passing, interface developers can focus on developing cross-device interaction by designing a mobile interface in response to a selected tabletop interface object. In order to demonstrate the versatility and design space of our toolkit, six design issues, i.e., data transfer, personalization, user interface composition, authentication, localized & private feedback, and input expressiveness, were discussed through a set of sample applications.

1. Introduction

A large screen device is useful in multi-user interaction, such as efficient information sharing (Bradel, Endert, Koch, Andrews, & North, 2013) or collaborative active learning (Martinez-Maldonado, Kalina, & Judy, 2015).When multiple users are interacting with a large screen device, one challenge is to avoid interference among users. For example, a tabletop, a horizontal large display device, is only equipped with one speaker, which inevitably causes interference if multiple users intend to play multimedia contents simultaneously. In addition to the interference, the large screen of a tabletop may potentially impose an accessibility issue on users (Shen, 2006). For example, when users are sitting around a tabletop, the digital content orientation that is suitable for one side is not appropriate for the opposite side. Cross-device interaction, which applies a personal smartphone to supplement a large display, may potentially address the above issues. However, it is challenging to implement cross-device interaction since it involves distributing an interface to different devices and synchronizing their states. Without a tool support, developers have to develop cross-device interaction applications from scratch, including both interface design and underlying communication, which are time-consuming and error-prone. Though toolkits and frameworks have been developed to speed up the development process of cross-device interaction, the majority of those tools either targeted web applications or only supported a single user in a multi-display scenario. Without losing generality, this paper presents a generic toolkit for developing tabletop-centric cross-device interaction applications, which feature with personalizing an interface for each individual user in a multi-user multi-display scenario.

It is natural to place physical objects on top of a surface. Therefore, our toolkit provides this natural gesture of placing and moving a smartphone on top of a tabletop. The smartphone functions like a look-through lens allowing users to browse and select tabletop contents beneath the smartphone. In other words, the screenshot of tabletop contents beneath a smartphone is displayed on the smartphone. So a user can tap the smartphone screen for an object in the screenshot to select the corresponding tabletop object. Once a tabletop object is selected, a personalized mobile interface is rendered on the smartphone for remote manipulation on the tabletop contents. Finally, a user's input on the smartphone is used to update the tabletop interface accordingly. Such interaction style seamlessly integrates a public tabletop with multiple personal smartphones to avoid user interference and address the accessibility issue. In order to evaluate the usability, 32 participants were recruited to complete 5 tasks and then fill in the IBM Post-Study Usability Questionnaire (PSSUQ) (Sauro & Lewis, 2016). The empirical study concluded an overall PSSUQ rating of 1.7 in a 7-likert scale (1 = strongly agree and 7 = strongly disagree), which justified the user-friendliness of the proposed interaction style. Based on the above interaction style, our toolkit formalizes the development of cross-device interaction as defining a mobile interface on each tabletop interface object. In other

CONTACT Jun Kong jun.kong@ndsu.edu North Dakota State University, Fargo, North Dakota, USA.

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/hihc.

words, a mobile interface rendered on the user's smartphone distributes and synchronizes interface states between a tabletop and a smartphone. Thus, our toolkit simplifies the development of cross-device interaction into two steps. First, developers design a standard tabletop interface. Then, cross-device interaction is implemented on a tabletop UI object by designing a mobile interface. The above two-step design process minimizes the gap between a single-device interaction and a cross-device interaction, and thus reduces the learning curve. In addition, our toolkit takes advantage of personal data stored on a smartphone to provide a personalized interface in a multi-user multi-display scenario. In order to demonstrate the expressiveness and versatility of our toolkit, a set of sample applications are developed to address six design issues (i.e., data transfer, personalization, user interface composition, authentication, localized & private feedback, and input expressiveness).

2. Related work

Seminal works were developed to support user-friendly cross-device interaction, such as augmenting a computer with PDAs in the single display groupware (Myers, 2001), exchanging information between a personal device and a public display (Greenberg, Boyle, & Laberge, 1999), or offering a continuous workspace that includes personal computers and information wall/table displays (Rekimoto & Saitoh, 1999). Equipped with diversified sensors, modern smartphones naturally augment a large display for co-located group work in many applications, such as oil and gas exploration (Seyed, Sousa, Maurer, & Tang, 2013), collaborative online shopping (Muta et al., 2014), geospatial exploration (Rodrigues, Carpendale, Seyed, & Maurer, 2014) or emergency response planning (Chokshi, Seyed, Rodrigues, & Maurer, 2014). This section reviews the related work from the perspectives of pairing devices, gestures, and toolkits for cross-device interaction.

2.1. Pairing devices

One central theme in cross-device interaction is to minimize the effort of connecting two or multiple devices in an ad hoc manner so that those connected devices can share and synchronize their UI states. Based on different sensing techniques, various techniques have been proposed to address the pairing issue.

- Touch-based sensing. The stitching approach forges a connection between two devices with the gesture of continuously moving a pen from one device to another one (Hinckley, Ramos, Guimbretiere, Baudisch, & Smith, 2004).
- Radio based sensing. Based on the radio frequency transponder sensing technology, ConnecTables integrates two displays as a larger shared one when two devices move close to each other (Tandler, Prante, Muller-Tomfelde, Streitz, & Steinmetz, 2001).
- Accelerometer. Accelerometer data are useful to derive users' actions, such as bumping two tablets (Hinckley, 2003) or shaking a device (Patel, Pierce, & Abowd, 2004). Accelerometer is often combined with other sensors. For example, Toss-it integrates an accelerometer sensor (i.e., estimating the strength or the trajectory of a user's action) with a camera (i.e., identifying the location of each user based on infrared LED markers) to determine a user's intention (Yatani, Tamura, Hiroki, Sugimoto, & Hashizume, 2005). PhoneTouch synchronizes a touch event detected by an interactive surface with a bump event detected by an accelerometer sensor on the smartphone. The synchronized events are used to connect the smartphone with the surface (Schmidt, Chehimi, Rukzio, & Gellersen, 2010). Hutama, Song, Fu, and Goh (2011) correlates the angle of two touch points with accelerometer data to distinguish different smartphones on a tabletop.
- Acoustic sensing. Point & Connect (Peng, Shen, Zhang, & Lu, 2009) measures the distance change through acoustic signals by pointing the source device to the target device.
- NFC. Touch & Interact uses an NFC phone as a stylus to touch any position in a dynamic display to establish a connection (Hardy & Rukzio, 2008).
- Camera-based sensing. The SpotCodes system augments a display with visual markers, and a user aims his/her phone at a visual tag on the display to select an object (Madhavapeddy, Scott, Sharp, & Upton, 2004). Instead of using built-in cameras, some approaches explore the usage of an additional camera for detecting users' gestures (Lee, Jeong, Lee, Yeom, Shin,& Park, 2008), recognizing infrared blinking signals to establish a connection between a smartphone and a table surface (Wilson & Sarin, 2007) or verifying camera flashes to authenticate users (Schoning, Rohs, & Kruger, 2008) Rather than using a regular RGB camera, Ackad et al. used a depth camera to track users and developed a handshaking procedure based on color detection (Ackad, Clayphan, Maldonado, & Kay, 2012). Consecutive color changes were used to transfer data between devices, such as FlashLight that illuminates the area on the tabletop below a smartphone (Hesselmann, Henze, & Boll, 2010), or C-Blink that produces color blinks from a smartphone (Miyaoku, Higashino, & Tonomura, 2004). HuddleLamp (Radle, Jetter, Marquardt, Reiteret, & Rogers, 2014) combines RGB and depth input to detect and track movements of mobile devices placed on a flat surface, and supports to add or remove displays in an ad-hoc manner. CapCam utilizes the rear camera of a "cam" device to capture color-modulated pairing data rendered on a "Cap" device, and accordingly sets up a wifi connection between the cam and cap devices (Xiao, Hudson, & Harrison, 2016).
- Synchronous action on buttons. SyncTap synchronizes button pressing and releasing events on two devices to connect them together (Rekimoto, Ayatsuka, & Kohno, 2003). The Touch-and-Connect framework (Iwasaki, Kawaguchi, & Inagaki, 2003) connects two devices by first pressing the plug-button in the source device and then pressing the socket-button in the target one.

Our toolkit applies an infrared camera to pair a mobile device with a tabletop through a natural gesture, i.e., placing a mobile device on top of a tabletop.

2.2. Gestures in cross-device interaction

Gesture-based interaction, either bare hand gestures or device-based gestures, is justified to be intuitive and comfortable in a multi-device ecology (Kray, Nesbitt, Dawson, & Rohs, 2010). Various gestures are proposed to support different interaction tasks in cross-device interaction, such as the Scoop-and-Spread gesture (Ayatsuka, Matsushita, & Rekimoto, 2000), the pen-based pick-and-drop gesture (Rekimoto, 1997), the pouring gesture (Seyed et al., 2013), or throw and tilt gesture (Marquardt & Greenberg, 2009). Various studies are conducted to design and evaluate cross-device interaction gestures in different scenarios. Kray et al. (Kray et al., 2010) systematically collected and analyzed a set of user-defined gestures in three scenarios, i.e., phone-to-phone, phone-to-tabletop, and phone to public display. Focusing on tablet-sized devices, Kurdyukova, Redlin, and Andre (2012) investigated three gesture modalities (i.e., multi-touch gestures, spatial gestures and direct contact gestures) in three conditions (i.e., iPad-iPad, iPad-tabletop, and iPad-public display). Radle et al. (2015) concluded that spatially-aware gestures are preferred by users when they are designed with great care.

Among cross-device interaction, one popular scenario is a synergetic usage of a smartphone and a large display, which has been proven to improve the performance (McAdam & Brewster, 2011) and facilitate information exchange in collaborative tasks (Seifert et al., 2012). In such a setting, a smartphone is commonly used as a remote controller to select and manipulate objects in a distant large display, such as sweep and point & shoot (Ballagas, Rohs, & Borchers, 2005), touch projector (Boring, Baur, Butz, Gustafson, & Baudisch, 2010), SnapAndGrab for content sharing (Maunder, Marsden, & Harper, 2008), and remote operations based on display registration (Pears, Jackson, & Oliver, 2009). In addition, it is useful to protect privacy in a public environment by using a personal device to supplement a public display (De Luca & Frauendienst, 2008).

2.3. Toolkits and frameworks

Cross-device interaction toolkits distribute interfaces and synchronize UI states on devices. Panelrama (Yang & Wigdor, 2014) dynamically distributes panels (i.e., groups of controls) in a web page to best-fit devices based on device characteristics (e.g., size and resolution), and only requires minimal changes to existing languages. HydraScope (Hartmann, Beaudouin-Lafon, & Mackay, 2013) transforms an existing web application to run across multiple displays without modifying the source code. The Tandem Browsing Toolkit supports developers to conceptualize, design and implement multi-display web pages through a declarative framework (Heikkinen, Goncalves, Kostakos, Elhart, & Ojala, 2014). Connichiwa is a framework for developing cross-device web applications in a local area network, and supports stitching gesture to explore the spatial relation between two devices (Schreiner, Reiterer, Radle, & Jetter, 2015). PolyChrome is designed for developing synchronous and asynchronous collaboration in visualization (Badam & Elmqvist, 2014). Frosini and Paterno presented a framework that dynamically distributes and/or migrates user interfaces in a multi-device environment (Frosini & Paterno, 2014). XDStudio (Nebeling, Mintsi, & Husmann, 2014) is a GUI builder for interactively developing cross-device interfaces in either a simulated authoring mode or an on-device authoring mode. Our toolkit implemented interface distribution and synchronization through message passing and focused on tabletop-centric multi-user cross-device interaction with the feature of personalization.

With the fast development of hardware, cross-device interaction has been extended from smartphones and large displays to a broader group of devices. WatchConnect (Houben & Marquardt, 2015) supports to quickly prototype smartwatch-centric cross-device applications that take a smartphone as both input and output. Weave allows developers to easily develop cross-device interaction among mobile and wearable devices based on JavaScript (Chi & Li, 2015). XDKinect uses Kinect to capture gesture and speech, and distinguishes proxemics parameters for developing proxemic-aware interaction with a multimodal input (Nebeling, Teunissen, & Husmann, 2014).

Some toolkits were developed to support cross-device interaction in various application domains. Conductor (Hamilton & Wigdor, 2014) enables the construction of cross-device applications, in which a user can easily share information, chain tasks and manage sessions across devices. However, Conductor focused on single-user interaction. By synchronizing a bumping event detected through an accelerometer sensor on a smartphone with a touch event detected through a touch screen on a tabletop, Schmidt, Seifert, Rukzio, and Gellersen (2012) proposed a cross-device interaction style, in which a smartphone was used as a stylus to manipulate contents on a tabletop. However, the smartphone screen is not utilized until a user selects an object from the tabletop. Instead, our toolkit enables to present personalized information on a smartphone during the selection process. Similar to PhoneTouch (Schmidt et al., 2012), the DisplayPointers system (Strohmeier, 2015) uses a mobile device as a tangible pointer and matches the touch information with accelerometer data to recognize different mobile devices. DisplayPointers explores the affordance of manipulating physical mobile objects in multi-device environments. Roudaki, Kong, Walia, and Huang (2014) implemented bimanual cross-device interaction (called MobiSurf), in which the non-dominant hand performed a coarse-grained selection through a physical pointer while the dominant hand held the smartphone for fine-grained interaction. However, it is not practical to carry a physical pointer. SoD-Toolkit explores different multi-device spatially-aware environments that can be augmented with new sensors and device platforms (Seyed, Azazi, Chan, Wang, & Maurer, 2015). The proximity toolkit (Marquardt, Diaz-Marino, Boring, & Greenberg, 2011) supports proxemic interaction, which exploits fine-grained proxemics relationships between people, objects and digital

devices (Ballendat, Marquardt, & Greenberg, 2010). Different from the majority of toolkits that target developers, XDBrowser was designed for conducting a user-driven elicitation study on cross-device interfaces, with the objective to understand user preferences and design requirements (Nebeling & Dey, 2016).

3. A toolkit for cross-device interaction

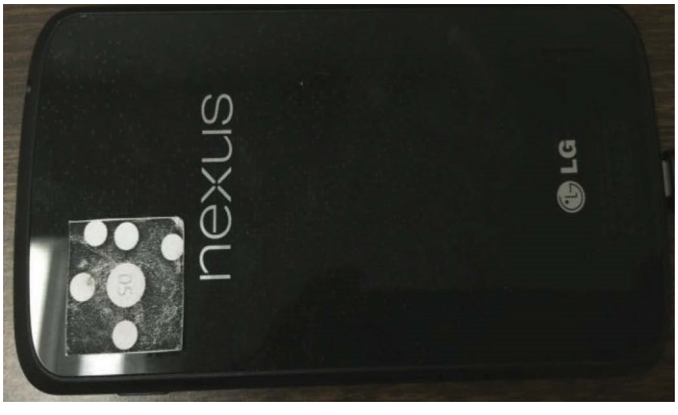
This section presents a toolkit that supports the development of cross-device interaction.

3.1. Overview

In daily lives, people naturally placed objects on top of a table. Accordingly, we introduce this daily natural action to pair a smartphone with a tabletop. Specifically speaking, our approach sticks a unique visual tag to the back of a user'ssmartphone(See Figure 1). When a smartphone is placed on top of a tabletop, the built-in infrared camera of the tabletop reads the visual tag to identify the user. In addition, the visual tag is continuously tracked when the smartphone is moving on top of a tabletop.[1] The location information of a smartphone within a tabletop screen is useful to support spatially-aware interaction, which is favored by users in cross-device interactions (Radle et al., 2015).

Our approach implemented a MagicLens-like design (Bier, Stone, Pier, Buxton, & DeRose, 1993) to display the content beneath a smartphone with personalization for each individual user, since this design has been successfully applied to cross-device interaction, such as augmenting a large display with detail (Baudisch, Good, & Stewart, 2001; Sanneblad & Holmquist, 2006) or distributing different layers of information to different display devices (Rodrigues et al., 2014; Spindler, Stellmach, & Dachselt, 2009). When a user moves his/her smartphone to the target object, he/she taps the target object through the smartphone to trigger the selection process. Our toolkit extends the standard "tap" gesture from single-device interaction to cross-device interaction, which potentially reduces the learning curve. In response to the selected target object, a mobile interface is rendered on the smartphone for private use and a public display that is accessible to all users while a personal smartphone supplements the public display with a private working space. Our toolkit has the following features:

- A daily natural action is used to pair a smartphone and a tabletop.
- Spatially-aware interaction is supported to browse a tabletop with customized contents.
- A traditional "tap" gesture is introduced to cross-device interaction.



3.2. Development of cross-device interaction

In order to demonstrate the expressiveness of our toolkit, we explore a set of applications to address six design issues, i.e., data transfer, personalization, user interface composition, authentication, private feedback,and input expressiveness (See Table 1).

3.2.1. Data transfer

Data transfer between devices is a fundamental issue in cross-device interaction. The data transfer from a tabletop to a mobile device is triggered when a smartphone is placed or moving on top of a tabletop. Conversely, the mobile-totabletop transfer is triggered when a user completes an input on the smartphone. The challenge in the data transfer is to correctly transfer and interpret data across different software and hardware platforms (i.e., Java and Android on the smart-phone, and C# and Windows on the tabletop). To overcome the above challenge, our toolkit implements data transfer through basic socket and stream methods that only send or receive common primitive data types. Those primitive data types are converted to an application-specific object at the receiver.

3.2.2. Personalization

Because smartphones are commonly used as personal devices, they store much personal information, such as contacts, the browsing history, and bookmarks. Taking advantage of

Table 1. Challenges in cross-device interaction.

Category	Description
Data Transfer	Tabletop-to-Mobile Transfer: Transfers data from a tabletop to a smartphone Mobile-to-Tabletop Transfer: Transfers data from a smartphone to a tabletop
Personalization	Autofill: Makes the personal data stored in a smartphone accessible to applications on the tabletop Content Customization: Personalizes information on a smartphone PhoneMap: Overlays personalized digital information on a map through a smartphone
User Interface Composition	Extended Screen: Uses a smartphone as an extended screen to issue commands
Authentication	Access Control: Provides a lightweight access control to lock or unlock tabletop contents Password: Inputs the password on the smartphone LockPattern: Uses a touch-based gesture on a smartphone to authenticate users
Private Feedback	Multimodal feedback: Provides multimodal feedback on smartphones
Input Expressiveness	PhoneGesture: Uses touch-based gestures on a smartphone to issue commands Personalized Gesture: Supports a personalized gesture in cross-device interaction

personal information stored on smartphones, we implemented the autofill technique to minimize the effort of user input. In addition, the tabletop contents can be customized on a user's smartphone to meet the user's personal need.

3.2.2.1. Autofill. Since the tabletop represents a public space, it is not appropriate for storing personal information. Consequently, a user has to manually type in information on the tabletop even if the information is already available on his/her smartphone. However, manual input is error prone and time consuming, particularly when the input is long and complex. Since much personal information is stored in the user's smartphone, our Autofill technique transfers personal information from a smartphone to a tabletop, and thus reduces the effort of data entry. For example, in web browsing, the autofill technique allows a user to choose a web URL from his/her browsing history saved in the smartphone and displays the selected web site on the tabletop. The autofill technique replaces free-style typing with selection, which potentially reduces the error rate (Shneiderman & Plaisant, 2009). Figure 2(a) shows a web browser displaying a website on the tabletop. After a user taps an address bar through the smartphone (See Figure 2(b)), a dropdown list that records the user's browsing history is displayed on the smartphone (See Figure 2(c)). When the user selects a favorable website from the dropdown list, the web browser on the tabletop is updated accordingly (See Figure 2(d)).

3.2.2.2. Content customization. Each user has a unique personal preference due to his/her background, skill level, culture and etc. Our toolkit enables to personalize tabletop contents through a smartphone. For instance, a tabletop is located in the lobby of an international company to introduce the company. When browsing the tabletop, users may have different preferences, such as reading an article in a different language. In order to avoid interference, the personalization is realized on a smartphone, (i.e., translating a selected paragraph from English to a different language). Figure 3(a) presents a tabletop interface displaying the several textual paragraphs. When a user selects a paragraph, a mobile interface (see Figure 3(b)) is displayed on a smartphone. The left area in the mobile interface is used to display texts while the right area contains a list of buttons, which allows a user to choose a language of his/her preference. Instead of displaying identical information to all users, our toolkit provides the flexibility to overlay personalized digital information on a smartphone. By default, when a user moves a smartphone on top of the tabletop, the smartphone simply displays the content beneath the device.Instead,auser canrequest additional information to supplement a default image. For

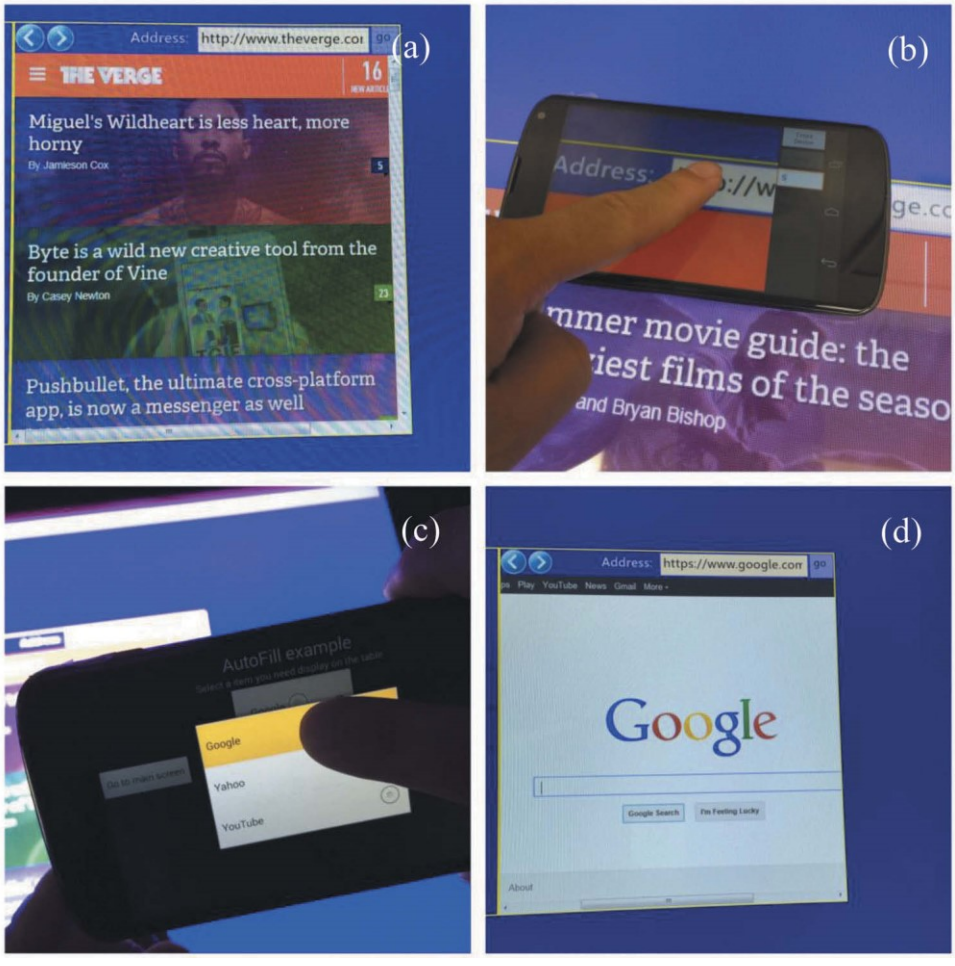


Figure 2. Autofill.

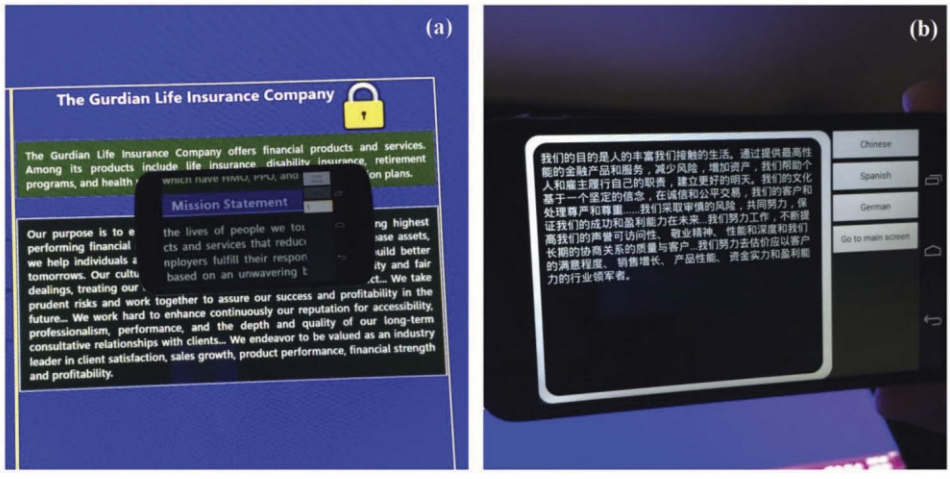


Figure 3. Content customization.

example, Figure 4(a) presents a PhoneMap tabletop interface. The smartphone on the top in Figure 4(b) displays a default map. On the other hand, the smartphone in the bottom in Figure 4(b) presents a personalized map that supplements a default map with a personal point of interest. Our toolkit stores all of map images on the tabletop server. Based on the location and ID of asmartphone,thetabletopservercontinuouslysends to the smartphone a default or user-requested image, which is rendered in a canvas object on the smartphone.

3.2.3. User interface composition

Though a tabletop provides a large screen for sharing information, traditional interfaces are not suitable for multi-user interaction. For example, if a user taps an action menu to issue a command, this action prevents other users from using the same menu simultaneously. Our toolkit extends a tabletop interface by rendering an action menu on a smartphone so that each user can issue a command independently and simultaneously.

3.2.3.1. Extended screen. The extended screen technique supplements a tabletop interface with an action menu on a smart-phone. Figure 5(a) shows that a user selected a paragraph through his/her smartphone. The selection triggers the generation of an action menu on the paired smartphone (See Figure 5(b)). The user can edit the selected paragraph through the mobile action menu, while other users can edit other objects simultaneously without interfering with each other. Based on the user's input, the contents on the tabletop are updated accordingly. In the above example, the mobile interface includes two parts. The left part is an action menu that includes a list of buttons and the right part is a drawing panel that allows a user to use a gesture (See 3.2.6 Input Expressiveness)to issue a command. In summary, the extended screen technique allows multiple users to remotely manipulate tabletop contents simultaneously and avoids interference.

3.2.4. Authentication

Authentication is important in multi-user interaction. Three interaction techniques, i.e.,

Figure 4. PhoneMap.

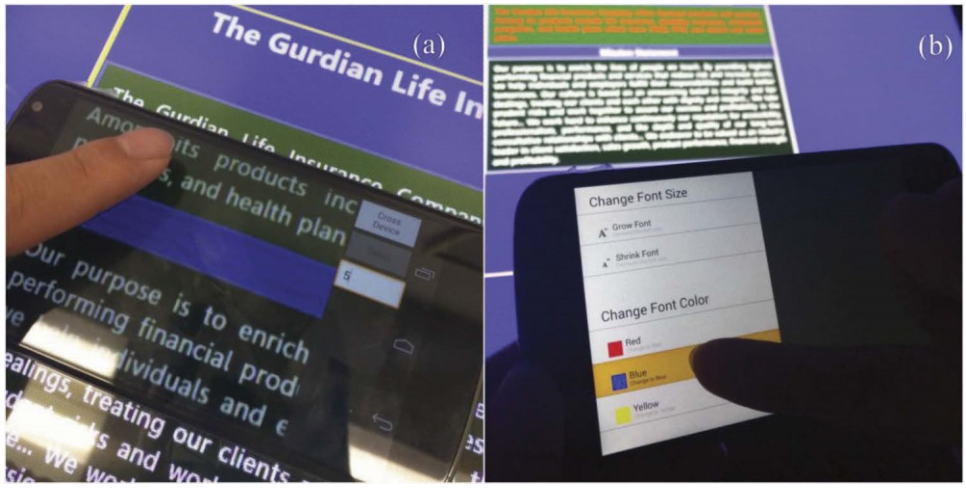


Figure 5. Extended screen.

Access Control, Password, and LockPattern, are discussed to address the authentication issue.

3.2.4.1. Access control. In a multi-user environment, mutual exclusion has to be enforced so that only one user can modify a piece of information at one time. Thus, our toolkit supports a user to lock and unlock a piece of information. Figure 6(a) shows that a user locked a paragraph by tapping the lock icon through a smartphone. Figure 6(b) presents a confirmation message on the smartphone and an updated unlock icon on the tabletop. If another user tries to access a locked paragraph, an error message is displayed (See Figure 6(c)). After the locked paragraph is unlocked by the same user (i.e., tapping the unlock icon), it can be accessible by another user.

3.2.4.2. Password. In a public environment, information displayed on a large screen can be read by any person. Our toolkit enhances the privacy by typing a password through a smartphone. Figure 7(a) shows that a user triggered a login form. Instead of typing the password directly on the tabletop, our toolkit renders a login form on the smartphone so that the user can complete the login process on his/her smartphone to protect privacy, as shown in Figure 7(b). In the above example, the mobile interface includes two parts. The left part is a standard login form and the right part is a drawing panel that allows a user to input a password by drawing a predefined shape (See LockPattern).

3.2.4.3. Lockpattern. Free style typing on a virtual keyboard is inefficient and error-prone. Therefore, alternative authentication techniques have been proposed to replace traditional password, such as fingerprint or Android password pattern. Our toolkit implemented a LockPattern technique, which authenticates a user by drawing a pre-defined shape. After a user triggers a login process, the smartphone displays both a traditional login form and a drawing panel. Therefore, the user can draw a predefined sketch to complete the authentication (See Figure 8).

3.2.5. Private feedback

Private feedback utilizes various output modalities (such as speech, vibration, beep, or dialog alert) on a smartphone to delivery information in a rich presentation format, which meets the need of a broad group of users.

3.2.5.1. Multimodal feedback. Developers can present information in different modalities on a smartphone. For example, in an authentication application (See Figure 7), vibration and beep can be used to supplement a visual login form on the smartphone. Vibration and beep alert a user to switch his/her focus from a tabletop to a smartphone. Multimodal feedback on a smartphone is especially useful for vision impaired users, who do not benefit from visual presentations. By supplementing a visual presentation with speech and vibration on a smartphone, our toolkit enables vision-normal and vision-impaired users to access information.

3.2.6. Input expressiveness

Our toolkit features personalized gestures, which have been proven to improve the memorability and user preference (Nacenta, Kamber, Qiang, & Kristensson, 2013).

3.2.6.1. PhoneGesture. ThePhoneGesturetechniqueallows ausertoissueacommandbydrawing ashapeon thesmartphone In the example of an extended screen, a user can draw in a drawing panel an arrow gesture to increase the font size (See Figure 9).

3.2.6.2. Personalized gesture. Personalized gestures achieved better memorability and higher user preference (Nacenta et al., 2013), and were consistent with users' expectation (Mostafapour & Hancock, 2014; Seyed et al., 2013). Our toolkit introduces personalized gestures to cross-device interaction. Specifically speaking, a user first defines a gesture by drawing a shape on his/her smart-phone. The definition of the personalized gesture is directly stored on the smartphone. At run time, when the user issues a command by drawing a personalized gesture in a drawing panel on the smartphone, the recognition result is sent back to the tabletop server, which accordingly updates the tabletop interface. Unlike menus, gestures are not visible to users and thus need a training to

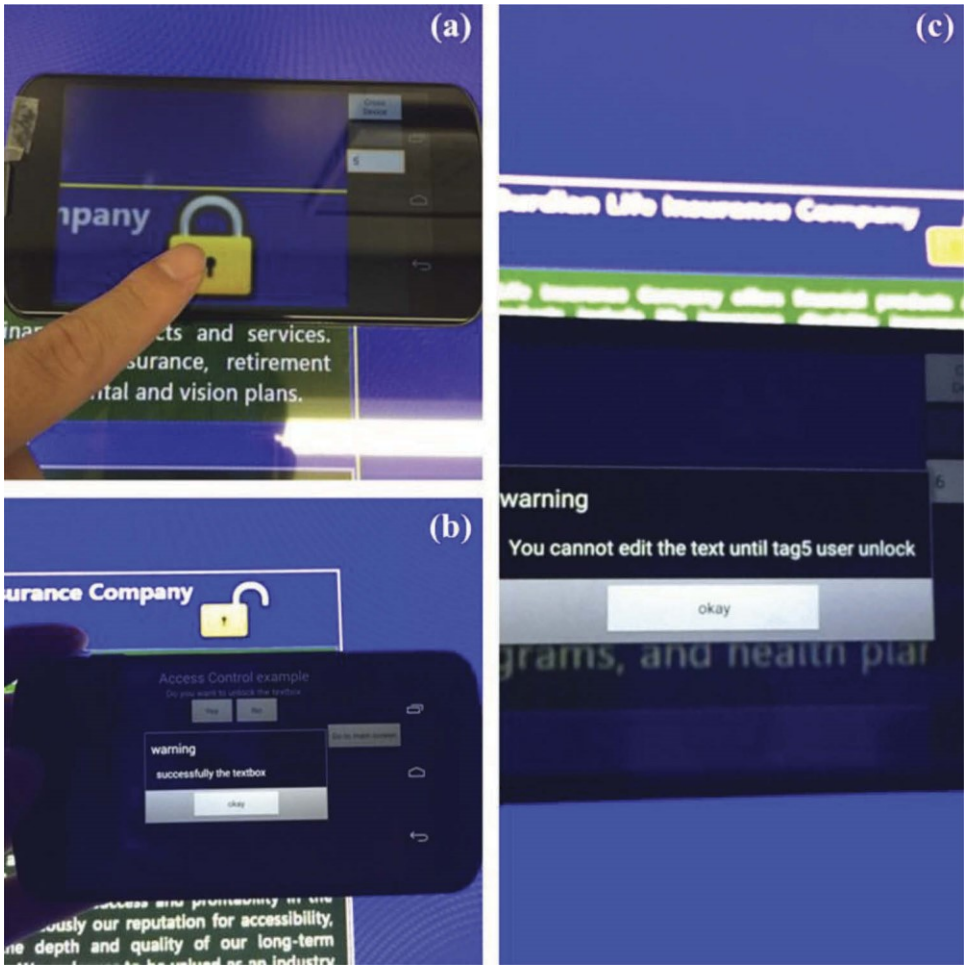


Figure 6. Access control.

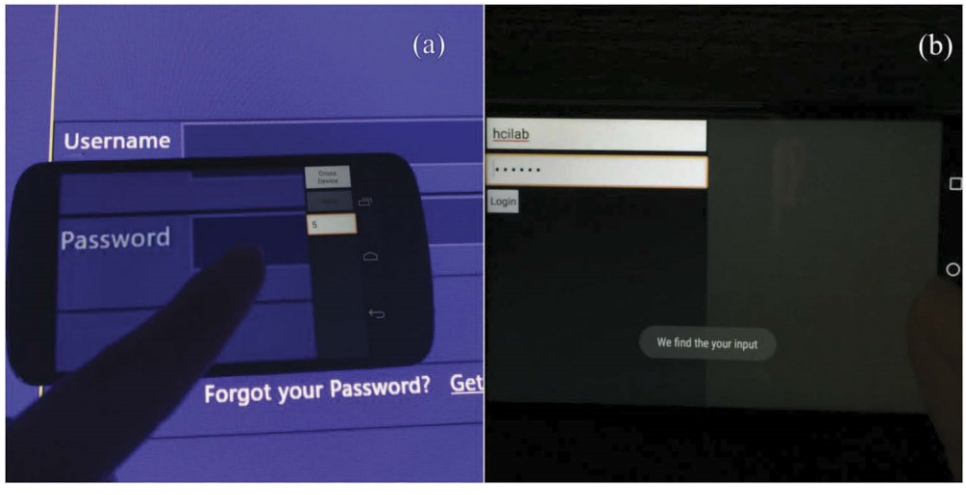


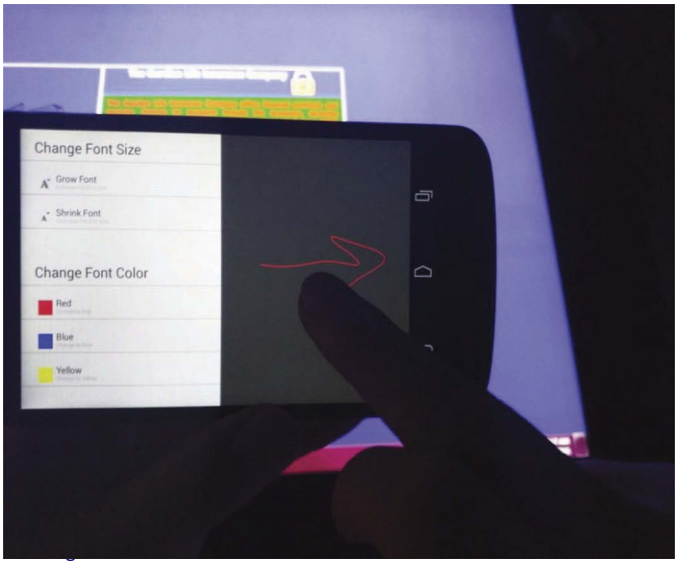
Figure 7. Password.

learn the semantics of each gesture (Seyed et al., 2013). Therefore, our toolkit supports tooltip, which is displayed to a target object, to facilitate a user to memorize personalized gestures. Figure 10(a)/(b) presents the personalized tooltips for each user, respectively.

4. Modeling cross-device interaction

The majority of toolkits focused on technique issues, while the design process of cross-device interaction has not been well explored. Our toolkit introduces a 6-state model (See Figure 11) to guide the design process of cross-device interaction.

Figure 8. LockPattern.



4.1. A 6-state model

A user starts with the OffSurface state, in which a user's smartphone is off the screen of a tabletop. In this state, a user simply uses his/her fingers to interact with the tabletop in a traditional manner. Then, the user places his/her smart-phone on top of the tabletop screen, which changes the state to StaticOnSurface.

Definition 1. Bijective function pair: TAGs → IDs defines an association between a visual tag and a user. Given a tag t, pair (t)= u if and only if visual tag t is paired with user u.

The notations of TAGs and IDs in Definition 1 represent the sets of tags and users, respectively. By placing a smartphone on a tabletop, a smartphone is connected to a tabletop and the user of the smartphone is identified through a visual tag. After paring, a user can either select an object (i.e., transiting from the StaticOnSurface state to the Selected state) or move his/her paired smartphone on top of the tabletop (i.e., transiting from the StaticOnSurface statetothe MovingOnSurface state). During the movement, the smartphone functions as a look-through lens, which displays the contents beneath the smartphone. This design provides the flexibility to personalize contents on a tabletop. In other words, a smartphone can overlay personalized digital information (e.g., restaurant or gas station) over the tabletop contents (e.g., PhoneMap in Figure 4). After a user moves the smartphone to the target object, the user can tap the target on the mobile screen, which transits the user to the Selected state. Definition 2 defines the mapping between the target object and the user. The notion of WIDGETs in Definition 2 indicates the set of tabletop UI objects.

Definition 2. Partial and injective function association: WIDGETs → TAGs defines whether a GUI widget is selected by a user or not. If w∈WIDGETs, then we write association (w) ↓ and say that association (w) is defined to indicate that w is in the domain of association.If w is not in the domain of association, we write association (w) ↑ and say that association (w) is undefined. A GUI widget w is selected by a user if and only if association (w) ↓.

Based on Definition 2, user u is in the Selected state if and only if ∃w∈WIDGETs, pair°association(w)= u. In other words, user u selects tabletop UI object w. Since the association function records the user who selects a tabletop UI object, it supports personalized interaction. Once tabletop UI object w is selected by user u (i.e., pair° association(w)= u), cross-device interaction of object w is defined as a mobile interface on user u's smartphone, as defined in Definition 3.

Definition 3. Total function map: WIDGETs → $2^{MobileUIs}$ defines a mobile interface when tabletop UI object w is selected. Given tabletop UI object w, map(w) = Ø if and only if w does not support cross-device interaction.

For example, a tabletop interface includes a text field, called password, which is designed for password entry. In order to prevent other users from glimpsing the password, interface developers can design a mobile login form

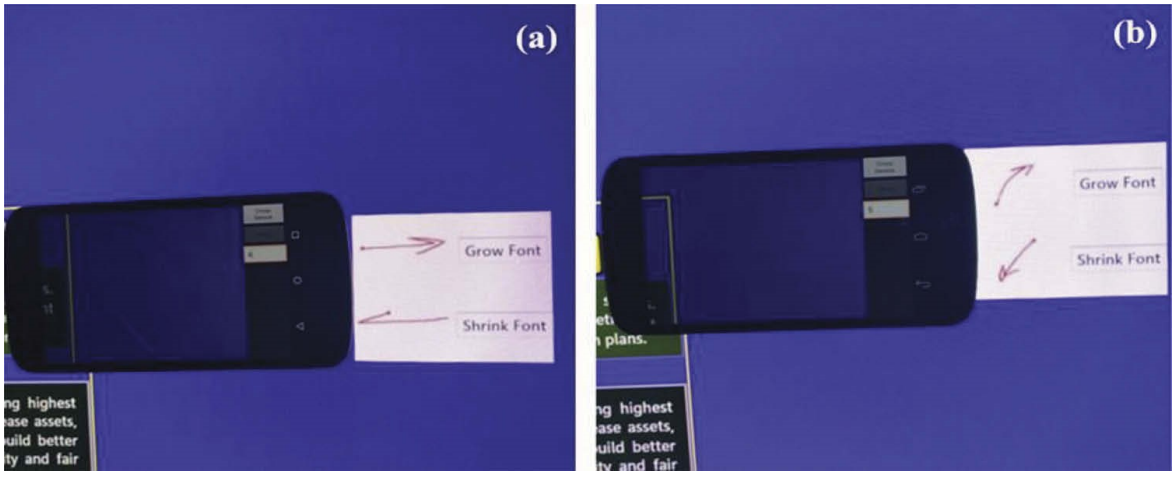


Figure 10. Tooltips

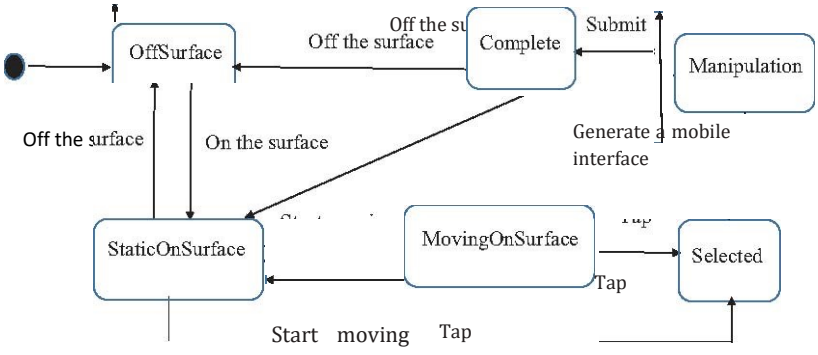


Figure 11. A 6-state model.

supplemented with vibration and speech on the smartphone. Therefore, cross-device interaction of password is defined as a mobile interface with three mobile UI objects, i.e., map (password) = {speech, vibration, edit text}. Since the mobile interface generation is defined on each tabletop UI object, a mobile interface has 4 to 5 visual objects by average. Therefore, our toolkit applies a simple heuristic rule to layout a mobile interface. Specifically speaking, if a mobile interface includes an image, which is displayed in a Canvas object, or a paragraph, which is displayed in a TextView object (See Table 2), the image or paragraph is displayed in the central area and other contents are displayed sequentially from top to bottom at the right side. Otherwise, all interface objects are simply displayed vertically in one column. Though we do not encounter a scenario that includes both an image and a paragraph, an automatic interface generation algorithm, e.g., SUPPLE (Gajos & Weld, 2004), can be integrated to strengthen our toolkit by handling complicated mobile interfaces.

In summary, the above three definitions formalize cross-device interaction of a tabletop UI component as a mobile interface. Table 2 lists all types of mobile UI objects, which can make up of a mobile interface in response to a selected tabletop object, while Table 3 defines cross-device interaction of a tabletop UI component in corresponding to the applications demonstrated in Section 3.

Table 2. Types of mobile UI objects.

Category	Description
Speech Lighting Beep TextView Media Alert Dialogue Web Link Vibration Button Edit Text Canvas Dropdown List	Speak a text Flash the LED light Generate a beep sound Display text contents Play a voice or video file Display a text alert message on smartphone Open a web page Vibrate the smartphone Display a button Display an editable text field Display a drawing and dynamic image panel Display a drop down list

4.2. Guidelines

Based on the above 6-state model, developers follow the following steps to design cross-device interaction.

(1) Interface developers elicit requirements from users and accordingly design a standard tabletop interface without supporting cross-device interaction.

(2) In the standard tabletop interface, interface developers classify two types of tabletop UI objects, i.e., cross-device interaction objects and standard objects. If a cross-device interaction object on a tabletop is selected by a user, it triggers a mobile interface for cross-device interaction; in contrast, a standard object does not support cross-device interaction. For example, a password object in the previous example is a cross-device

Table 3. Definitions of cross-device interaction in different applications.

Applications		Definition
Personalization	Autofill	map(address_bar) = {TextView$_{Title}$, TextView$_{SubTitle}$, DropDown_List}
	Content	map(Paragraph_Text) = {TextView$_{Content}$, Button$_{Chinese}$, Button$_{Spanish}$, Button$_{German}$, Button$_{MainScreen}$}
	Customization	
	PhoneMap	map(map) = {Canvas$_{map}$, Button$_{connect}$, Button$_{setup}$, EditText$_{id}$} map(Paragraph_Text) = {TextView$_{Title}$,
User Interface Composition	Extended Screen	Button$_{GrowFont}$, Button$_{ShrinkFont}$, Button$_{Red}$, TextView$_{Title}$, Button$_{Blue}$, Button$_{Yellow}$, Canvas}
Authentication	Access Control	map(Lock_Icon) = {TextView$_{Title}$, Button$_{Yes}$, Button$_{No}$, Alert Dialogue}
	Password	map(TextField$_{username/password}$) = {EditText$_{username}$, EditText$_{password}$, Button$_{login}$, Canvas}
	LockPattern	map(TextField$_{username/password}$) = {EditText$_{username}$, EditText$_{password}$, Button$_{login}$, Canvas}
Private	Multimodal	map(TextField $_{username/password}$) = {EditText$_{username}$, EditText$_{password}$, Button$_{Login}$, Canvas, Vibration, Beep}

interaction object. On the other hand, a logo, which is designed for display only, is considered a standard object.

(3) Interface developers define a mobile interface for each cross-device interaction object in a tabletop interface. When a cross-device interaction object on a tabletop is selected by a user, the definition of the corresponding mobile interface is coded in a message that is transferred from a tabletop to a smartphone, and a user's input on the smartphone is passed back to the tabletop. In summary, our toolkit distributes a user interface to a smartphone and synchronizes UI states through message passing.

(4) Our toolkit uses the infrared camera to track each user's activity on top of a tabletop and features personalizing the tabletop contents for each individual user. Based on the 6-state model, the personalization can be implemented in the following states: (1) Moving OnSurface or StaticOnSurface, and (2) Manipulation. In the state of MovingOnSurface or StaticOnSurface, according to a detected user identity and the user's request, the server on a tabletop may overlay additional information on a default image (e.g., the screenshot beneath the smartphone) by sending a personalized image to the smartphone. In the state of Manipulation, our toolkit utilizes the personal data that is stored on a smartphone (e.g., recently accessed web sites) to personalize a mobile interface (e.g., contents in a drop down list).

4.3. Message passing

Our toolkit is implemented with a three-tier structure (i.e., communication layer, business layer and presentation layer) deployed on a tabletop and a smartphone. The communication and business layers on a tabletop form the foundation of our toolkit, which extends traditional UI objects with the capacity of cross-device interaction. Specifically speaking, the communication layer applies the TCP/IP protocol to exchange messages between a tabletop and a smartphone. Based on a user'sactivity, the business layer generates and interprets a message. To implement the above three-layer architecture, our toolkit provides a WPF custom control that extends basic UI objects (e.g., Button or TextBox) with the capacity of communications between a tabletop and a smartphone. Specifically, on the tabletop side, a custom control, called ServerControl,supportstwocustomized events, i.e., objectSelect and returnValue events. The objectSelect event of a tabletop object is fired when this object is selected by auser, whoseIDispassedtothe objectSelect event so that the toolkit can deliver a message to the user who selects the object. In the event handler of the objectSelect event, developers define a message that specifies a mobile interface, and the message is delivered the corresponding smartphone to realize cross-device interaction. Each message includes a list of 3-tuple that makes up a mobile interface. Each tuple indicates a UI object in the mobile interface and has three members separated by a semicolon The first member defines the type of a mobile UI object (See Table 2), the second member illustrates the caption, and last one specifies a default value. Except the first member, other two members are optional since some UI objects may not need a default value or a caption, such as a beep. For example, a password in a mobile interface can be defined as {EditText; "password";}. On the other hand, the returnValue event of a tabletop object is fired when an input on this object's mobile interface is returned to the tabletop. Accordingly, in the event handler of the returnValue event, developers need to interpret the return message and update the tabletop interface. On the mobile side, the PairingConnection class provides two-way communications between a tabletop and a smartphone. Once a message is received by the smartphone, the TranslationMessage class is called to interpret the message and accordingly generate a mobile interface.

We use the password example to explain the message passing in our toolkit. When a user interacts with the tabletop and selects the password tabletop object by tapping it through a smartphone, our toolkit fires the objectSelect event, whose event handler records the definition of a mobile interface when password is selected. Then, the definition of the mobile interface is sent to the paired smartphone over a network. When the mobile device receives the message, it unpacks the message and accord-inglyrenders amobileinterface.After theusertypes the required information on the smartphone, the user'sinput is sent back to the tabletop. Accordingly, the receipt of the message fires the returnValue event, whose event handler updates the tabletop interface. In summary, Figure 12 demonstrates a detailed message passing about the pass-word example between a tabletop and a smartphone.

5. Usability evaluation

We conducted an empirical study to evaluate the usability of the cross-device interaction style that is supported in our toolkit.

5.1. Participating subjects

Participants were recruited through emails, and a total of 32 participants successfully completed the study. The participants were college students from a mid-west university, and they were given small extra credits to a class they were taking as incentives to join this study. 84% participants identified themselves as male, with 16% as female. 34% participants used their phones over 18 hours per week, followed by 5 hours or less (31%), 12 to 17 hours (19%), and 6 to 11 hours (16%). All participants owned a smart phone (Android or iPhone). However, almost half of the participants (15 out of 32 participants) rarely used tabletops and only 2 participants used tabletops frequently.

5.2. Apparatus

This study used Samsung SUR40 as an interactive public display and Nexus 4 as a personal device. Samsung SUR40 was featured with a 40-inch full HD LCD, which used the PixelSense technology that can track 50 simultaneous touch points on the surface. On the other hand, Nexus 4 has a 4" display which supplements Samsung SUR40 in cross-device interaction. Our toolkit applied a client-server architecture, where the server was deployed on Samsung SUR40 and the client on Nexus 4. Though the study was conducted on Nexus 4, the proposed toolkit can be running on any type of Android devices.

5.3. Experiment

The experiment began with a pre-study questionnaire, followed by a training session. After completing the experiment, participants filled out a post-study questionnaire to provide their feedback and open comments about using the proposed interaction style.
Step 1: Pre-study questionnaire. In this step, participants provided their background information about reading skill, gender, and experience of using touch screen devices.
Step 2: Training session. In this step, participants were trained to learn cross-device interaction.
Step 3: Experiment. In this step, participants went through the following tasks.

- Task 1 – Personalization (Content Customization). This task focuses on transferring a paragraph from a tabletop to a smartphone and translating it to a different language. Participants were asked to first select a paragraph on the tabletop by tapping the look-through image on the smartphone. Then, a mobile interface (see Figure 3) that included three language buttons was displayed and allowed the participant to translate the paragraph to a language by tapping the corresponding button on the smartphone.
- Task 2 – User Interface Composition (Extended Screen). This task focuses on remotely controlling tabletop contents through a smartphone. After a participant

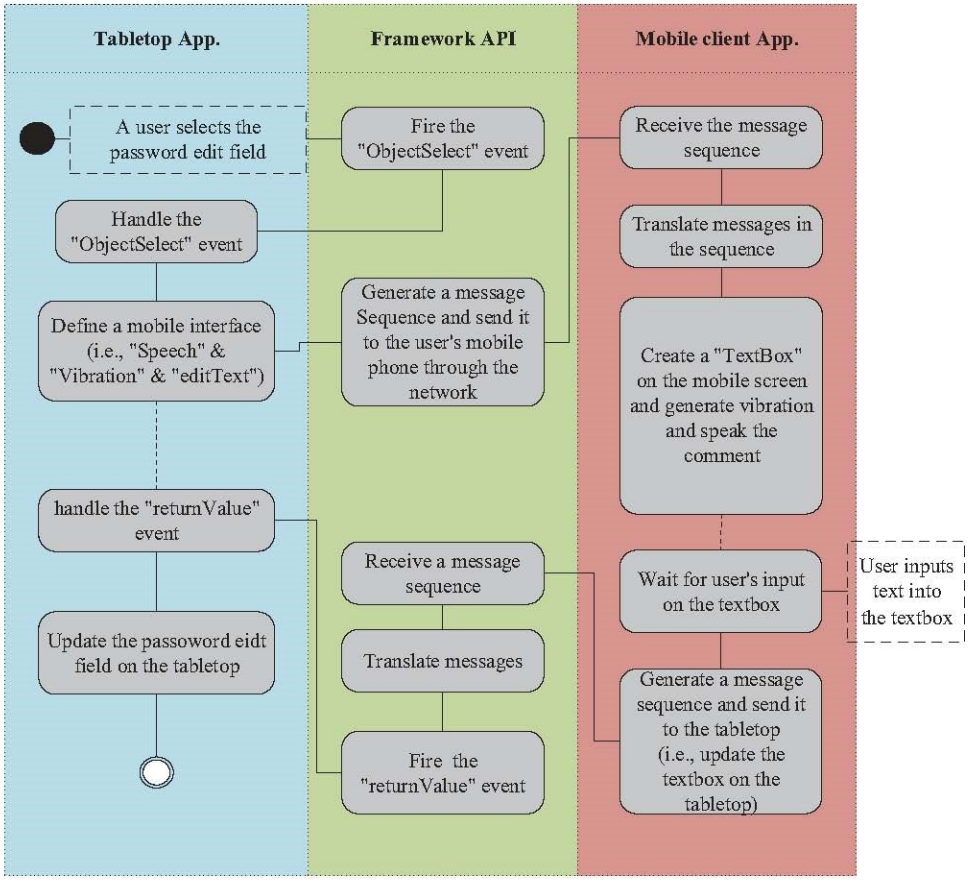


Figure 12. Message passing.

successfully selected the target object through the smart-phone, a mobile interface (see Figure 5) that included a list of buttons and a drawing canvas was displayed on the smartphone. The participant was asked to use two methods (i.e., tapping a button and drawing a predefined gesture in the canvas on the smartphone) to change the format of the target object on the tabletop.

- Task 3 – Authentication (Access Control). This task focuses on locking a paragraph to support mutual exclusion. The experiment observer first locked a paragraph. Then, a participant was asked to select the locked paragraph, which triggered a deny message (see Figure 6(c)) on the participant's smartphone. After the observer unlocked the paragraph, the participant was asked to lock the paragraph by tapping "lock icon" through the smartphone (see Figure 6(a)) and a confirmation message was displayed accordingly (see Figure 6(b)).
- Task 4 – Authentication (Password). This task focuses on completing a login process through the smartphone. After a participant selected the password object, a mobile login interface (see Figure 7) was displayed, and the participant filled in the username and password to complete the login on his/her smartphone.
- Task 5 – Personalization (Autofill). This task focuses on transferring saved information from a smartphone to a tabletop. A participant was asked to select an address bar in a browser. Then, a dropdown list (see Figure 2) that included previously browsed web sites was displayed on the smartphone. The participant can select a website from the dropdown list, and the browser on the tabletop renders the selected website.

Step 4: Post-study questionnaire. After completing the above tasks, participants were asked to fill out an online survey questionnairetoevaluate theirexperience.We usedthe IBMPost-Study Usability Questionnaire (PSSUQ) (Sauro & Lewis, 2016)to measure the usability, since PSSUQ was justified to be highly reliable to evaluate the usability of an interface (Sauro & Lewis, 2016)and hadbeenusedinmanyusability experiments (Buchanan, Farrant, Jones, & Thimbleby, 2001; Noguera, Barranco, Segura, & Martínez, 2012; Sheehan, Lee, Rodriguez, Tiase, & Schnall, 2012). PSSUQ included 16 items, which produces four scores, i.e., System Quality (SysQual, Items 1 through 6), information quality (InfoQual, Items 7 through 12), interface quality (IntQual, Items 13 through 15) and overall (Item 16). All questions are rated from 1 (strongly agree) to 7 points (strongly disagree). After the questionnaire, the observer also collected participants' open comments.

5.4. Results and discussion

5.4.1. System quality

The average score of system usefulness (Items 1 to 6) is 1.64. Our toolkit simulates a context+focus visualization, where the tabletop displays the entire browsing context and a smartphone presents the information of a user's focus. Furthermore, our toolkit applies the popular tap gesture to select an object. The familiar visualization and gesture facilitate users to transfer their previous experience from single-device interaction to cross-device interaction, which makes cross-device interaction easy to learn and use, as evidenced by Item 5. By using a dominant hand to hold a smartphone for browsing, selection and manipulation, our toolkit enables a user to complete the cross-device interaction with one single hand. Such a design reduced the effort of switching focus between different devices and thus improved the efficiency, as evidenced by positive feedback in Items 3, 4, and 6.

5.4.2. Information quality

The average score of information quality (Items 7 to 12) is 1.73. Participants commented that they did like the feature of multimodal feedback, which utilizes a variety of hardware components on a smartphone. The multimodal feedback also provides the flexibility for interface designers to present information in different modalities for each UI object, which is especially useful in a public environment that is featured with diversified interaction contexts.

5.4.3. Interface quality

The average score of interface quality (Items 13 to 15) is 1.94. Over 90% participants agreed that the interface was pleasant and had all the functions they expect. Compared with a large tabletop, smartphones provide a larger number of pixels per inch. Therefore, our approach provides the potential to render a sharper image through a smartphone. The major criticism from the participants is the implementation of a dark background color on the tabletop. This background setting is caused by the technique limitation of infrared-based tracking, which requires a dark background to precisely track a visual marker.

In summary, Table 4 presents the average rating of each item. In order to judge participants' perceptions of usability, we compare our data with the PSSUQ norms (Sauro & Lewis, 2016), as presented in Table 5. The comparison justified the positive feedback on our interaction. One potential improvement is to elaborate the visual feedback on a tabletop. Objects on the tabletop are classified into two categories, i.e., standard objects and cross-device interaction objects. Only a cross-device interaction object can trigger a mobile interface on the smartphone. However, when a smartphone is moving on top of tabletop, our toolkit did not provide a visual hint to differentiate two types of objects. It can cause anxiety for a new user if she/he did not receive any feedback after tapping an object on a smartphone. In the future, we plan to provide visual feedback so that users can tell a standard object from a cross-device interaction object.

6. Design implications

This paper presents a toolkit for prototyping tabletop-centric cross-device interaction. Our toolkit features defining the cross-device interaction of a tabletop UI object as a mobile interface, which distributes and synchronizes interfaces through message passing. By taking advantage of the personal data (such as bookmarks or contacts) on a smartphone, our work enables to personalize interface contents for each individual user in a multi-user multi-display scenario.

Table 4. PSSUQ ratings.

	Standard Deviation	Mean	Cronbach's alpha
1. Overall, I am satisfied with how easy it is to use this system.	0.72	1.72	
2. It was simple to use this system.	0.71	1.53	
3. I was able to complete the tasks and scenarios quickly using this system.	0.62	1.44	
4. I felt comfortable using this system.	0.85	1.72	
5. It was easy to learn to use this system.	0.76	1.47	
6. I believe I could become productive quickly using this system.	1.23	1.97	
7. The system gave error messages that clearly told me how to fix problems.	1.16	2.06	
8. Whenever I made a mistake using the system, I could recover easily and quickly	0.92	1.84	
9. The information (such as on-line help, on-screen messages and other documentation) provided with this system was clear.	0.85	1.72	
10. It was easy to find the information needed.	0.77	1.72	
11. The information was effective in helping me complete the tasks and scenarios.	0.57	1.50	
12. The organization of information on the system screens was clear.	0.67	1.56	
13. The interface of this system was pleasant.	1.12	1.91	
14. I liked using the interface of this system.	1.03	1.81	
15. This system has all the functions and capabilities I expect it to have.	1.12	2.13	
16. Overall, I am satisfied with this system.	0.98	1.75	
			0.906

Table 5. Comparison with PSSUQ norms.

	Our System	PSSUQ norms	Cronbach's alphas	
				p < .001
System Quality (Items 1 to 6)	1.64	2.8		p < .001
Information Quality (Items 7 to 12)	1.73	3.02	0.822	p = .01
Interface				3 p

6.1. Generality

By designing a set of various applications, we demonstrate the versatility and generality of the toolkit. With our toolkit, a user can perform cross-device interaction with one or two hands. Especially, our toolkit is applicable to support two-handed asymmetric tasks. Specifically speaking, once a tabletop object is selected, the area of the selected object on the tabletop is reserved for the selected user. Then, the user uses his/her primary hand to operate the selected object on the tabletop (such as drawing a line), while he/she uses the secondary hand to operate the smartphone to update features (such as changing the color of a line). In addition, smart-phones are equipped with different sensors, which provide alternative interaction methods to supplement the touch screen.

Our toolkit was implemented on a digital tabletop with a built-in infrared camera. However, the 6-state model with an implementation of message passing is not limited to a digital tabletop. Therefore, our toolkit is applicable in amorecommonsetting whereanexternalcamerais installedtocapturemobiledevices that areplacedormoving on a traditional flat surface, instead of a digital tabletop.

6.2. Multi-user interaction

Our toolkit addresses two challenging issues in the multi-user interaction, i.e., user interference and personalization.

- In our approach, a smartphone with a unique marker is placed on top of the tabletop so that a built-in infrared camera can recognize and identify each individual user. Based on the user ID, a personalized interface is distributed to the user's smartphone. Therefore, our toolkit allows multiple users to smoothly interact with different objects on a tabletop. When two users are interested in the same object on the tabletop without modification, they can place their smartphones in such locations that both devices can at least cover a portion of the target object. Thus, both users can tap on their smartphones to select the object at the same time. Once the tabletop object is selected, each user can continue the interaction through his/her own smartphone without interfering with other users. If a task needs to modify an object on the tabletop, a user needs to lock the object to prevent simultaneous modifications through the access control technique (See Section 3.2.4).
- Personalization is desirable in multi-user interaction, which adapts contents to meet each user'spersonal need. Our toolkit supports personalization from two perspectives. First, when a smartphone is on top of a tabletop (i.e., StaticOnSurface or MovingOnSurface), the contents displayed beneath the smartphone can be adjusted based on auser's personal need. Second, after a user selects a tabletop object, both contents (e.g., contents in a drop down list) and interaction methods (e.g., personalized gestures) can be personalized on a smartphone.

6.3. High subjective satisfaction

The interaction style in the toolkit was designed to reduce the gap between traditional single-device interaction and cross-device interaction, which made it easy to learn cross-device interaction. Specifically speaking, the "look-through lens" design simulates the viewfinder of camera, which makes it efficient to find the object beneath the smartphone; and the natural tapping gesture was commonly used on touch screens. Since the look-through lens design and the tapping gesture have been utilized widely in traditional single-device interaction, our interaction design assists users to excess from traditional single-device interaction to cross-device interaction based on their previous experience.

A tap gesture is implemented on the look-through lens to select an object. Though a user can only tap one object on the look-through lens every time, our toolkit provides the flexibility for developers to determine what information to be transferred to a smartphone in response to a selected object. For example, in the password example (See Figure 7), when a user tapped a password object, both the username and password objects were displayed on the smartphone. In summary, when developing an event handler for a selected object

(e.g., a password object), the developers can define to transfer a set of semantically related objects (e.g., both a username object and a password object) to a smartphone for remote manipulation.

Our toolkit supports two methods to select an object. First, a user can move a smartphone on top of a tabletop, and select an object when the smartphone reaches the target. This method is especially useful when additional information is displayed on the smartphone to supplement tabletop contents during the movement of a smartphone. Second, a user can directly drop his/her smartphone on top of the target object, which is more efficient to select an object.

6.4. Empowering tabletop-centric cross-device interaction

The design of cross-device interaction significantly lags behind that of single-device interaction due to the lack of toolkit supports. Since it is challenging and time-consuming to develop prototypes of cross-device interaction, the evaluation mainly depends on Wizard of Oz. Our toolkit reduces the learning curve by extending a standard IDE environment, and thus lowers the threshold of exploring tabletop-centric cross-device interaction. Especially, our approach implements a thin client on the smartphone and leaves all of the application-related logic on the tabletop, which thus reduces the development cost. In addition, the 6-state model in our toolkit simplifies the definition of cross-device interaction as a mobile interface, which supports developers to transfer their previous experience to cross-device interaction.

Spatial information (such as relative location between two devices or the orientation of a device) is valuable to support spatially-aware cross-device interaction. Our toolkit applies a built-in infrared camera on a tabletop to track each smartphone's location and orientation. Accordingly, information can be distributed based on relative locations among smartphones. Therefore, our toolkit is suitable to support spatially-aware cross-device interaction, such as the PhoneMap application. In the future, we will explore spatially-aware interaction in the context of collaboration. For example, when two users are interested in watching a movie, a collaborative browsing can be implemented with our toolkit. Specifically speaking, both smartphones are placed at locations where each smartphone partially covers a movie picture on the tabletop. After each user clicks the movie picture, the smartphone at the left side displays reviews about the movie while the one at the right side shows the play time. In addition, the orientation information of a smartphone can be used to supplement touch-based gestures, such as rotating a smartphone to control information.

7. Conclusion and future work

We implemented a generic toolkit that facilitates interface developers to develop tabletop-centric cross-device interaction. In our toolkit, a smartphone is placed on top of a tabletop, and functions like a look-through lens for browsing tabletop contents. When a user taps an object on the smartphone, a corresponding mobile interface with multimodal feedback is rendered on the smartphone to support cross-device interaction. Based on the above interaction style, we propose a 6-state model that simplifies the development of cross-device interaction as a mobile interface, which reduces the development cost and learning curve. In order to demonstrate the versatility and generality of our toolkit, we develop a set of applications that feature personalization and avoid user interference in a multiuser multi-device environment. The future work includes improving the accessibility for vision-impaired users on tabletop interfaces and evaluating their usability. We will also integrate different interaction techniques (such as one-handed interaction or two-handed interaction) in our toolkit and compare their difference. In this study, participants were recruited from three classes, and may or may not know each other. Since the familiarity between participants may affect the user experience, this user study was limited to single-user interaction in order to control variance. In the future, we will evaluate the usability in the context of multi-user interaction.

Note

1. Without losing generality, we implemented the framework on the Microsoft SUR 40 tabletop and Android smartphones.

Funding

This work is in part supported by NSF under grant #1722913.

References

Ackad, C. J., Clayphan, A., Maldonado, R. M., & Kay, J. (2012). Seamless and continuous user identification for interactive tabletops using personal device handshaking and body tracking. Proc. CHI EA. doi: 10.1094/PDIS-11-11-0999-P

Ayatsuka, Y., Matsushita, N., & Rekimoto, J. (2000). HyperPalette: A hybrid computing environment for small computing devices. Proc. CHI '00 Extended Abstracts on Human Factors in Computing Systems (pp.133–134), The Hague, The Netherlands.

Badam, S. K., & Elmqvist, N. (2014). PolyChrome: A cross-device framework for collaborative web visualization. Proc. ITS'14 (pp.109–118), Dresden, Germany.

Ballagas, R., Rohs, M., & Borchers, J. (2005). Sweep and point & shoot: Phonecam-based interactions for large public displays. Proc. CHI EX. Abstracts, Portland, OR, USA.

Ballendat, T., Marquardt, N., & Greenberg, S. (2010). Proxemic interaction: Designing for a proximity and orientation-aware environment. Proc. ITS'10 (pp. 121–130), Saarbrücken, Germany.

Baudisch, P., Good, N., & Stewart, P. (2001). Focus plus context screens: Combining display technology with visualization techniques. Proc. UIST'01 (pp.31–40). doi: 10.1016/s0021-9290(00)00166-4

Bier, E. A., Stone, M. C., Pier, K., Buxton, W., & DeRose, T. D. (1993). Toolglass and magic lenses: The see-through interface. Proc. SIGGRAPH'93, Anaheim, CA, USA.

Boring, S., Baur, D., Butz, A., Gustafson, S., & Baudisch, P. (2010). Touch Projector: Mobile interaction through video. Proc. CHI'10, Atlanta, Georgia, USA.

Bradel, L., Endert, A., Koch, K., Andrews, C., & North, C. (2013). Large high resolution displays for co-located collaborative sensemaking: Display usage and territoriality. International Journal of Human-computer Studies, 71, 1078–1088. doi:10.1016/j.ijhcs.2013.07.004

Buchanan, G., Farrant, S., Jones, M., & Thimbleby, H. (2001). Improving mobile internet usability. Proc. the 10th international conference on World Wide Web (pp. 673–680), Hong Kong. ACM.

Chi, P. Y., & Li, Y. (2015). Weave: Scripting cross-device wearable interaction. Proc. CHI'15, Seoul, Republic of Korea.

Chokshi, A., Seyed, T., Rodrigues, F. M., & Maurer, F. (2014). ePlan multi-surface: a multi-surface environment for emergency response planning exercises. Proc. ITS'14 (pp.219–228), Dresden, Germany .

Dachselt, R., & Buchholz, R. (2009). Natural throw and tilt interaction between mobile phones and distant displays. Proc. CHI Ext. Abstracts, Boston, MA, USA.

De Luca, A., & Frauendienst, B. (2008). A privacy-respectful input method for public terminals. Proc. NordiCHI, Lund, Sweden.

Frosini, L., & Paterno, F. (2014). User interface distribution in multi-device and multi-user environments with dynamically migrating engines. Proc. EICS'14, Rome, Italy.

Gajos, K., & Weld, D. S. (2004). SUPPLE: Automatically generating user interfaces. Proc. IUI'04 (pp.93–100). doi: 10.1097/01.shk.0000097247.97298.0e

Greenberg, S., Boyle, M., & Laberge, J. (1999). PDAs and shared public displays: Making personal information public, and public information personal. Personal Technologies, 3,54–64.

Hamilton, P., & Wigdor, D. (2014). Conductor: Enabling and understanding cross-device interaction. Proc. CHI'14 (pp.2773–2782), Toronto, Ontario, Canada.

Hardy, R., & Rukzio, E. (2008). Touch & interact: Touch-based interaction of mobile phones with displays. Proc. MobileHCI'08 (pp.245–254), Amsterdam, The Netherlands.

Hartmann, B., Beaudouin-Lafon, M., & Mackay, W. (2013). Hydrascope: creating multi-surface meta-applications through view synchronization and input multiplexing.Proc.PerDis'13, Mountain View, California, USA.

Heikkinen, T., Goncalves, J., Kostakos, V., Elhart, I., & Ojala, T. (2014). Tandem browsing toolkit: Distributed multi-display interfaces with web technologies. Proc. PerDis'14 (pp.142–147). doi: 10.3389/fncir.2014.00142

Hesselmann, T., Henze, N., & Boll, S. (2010). FlashLight: Optical communication between mobile phones and interactive tabletops. Proc. ITS, Saarbrucken, Germany.

Hinckley, K. (2003). Synchronous gestures for multiple persons and computers, Proc. UITS'03, pp.149–158, Vancouver, Canada.

Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., & Smith, M. (2004). Stitching: Pen gestures that span multiple displays. Proc. AVI'04, Gallipoli, Italy.

Houben, S., & Marquardt, N. (2015). WatchConnect: A toolkit for prototyping smartwatch-centric cross-device applications. Proc. CHI'15, Seoul, Republic of Korea (pp.1247–1256).

Hutama, W., Song, P., Fu, C., & Goh, W. (2011). Distinguishing multiple smart-phone interactions on a multi-touch wall display using tilt correlation. Proc. CHI, Vancouver, BC, Canada.

Iwasaki, Y., Kawaguchi, N., & Inagaki, Y. (2003). Touch-and-connect: A connection request framework for Ad-hoc networks and the pervasive computing environment. Proc. PerCom'03, Fort Worth, TX, USA.

Kray, C., Nesbitt, D., Dawson, J., & Rohs, M. (2010). User-defined gestures for connecting mobile phones, public displays, and tabletops Proc. MobileHCI'10 (pp.239–248). doi: 10.1177/1753193409349856

Kurdyukova, E., Redlin, M., & Andre, E. (2012). Studying user-defined ipad gestures for interaction in multi-display environment. Proc. IUI'12 (pp.93–96), Lisbon, Portugal.

Lee, H., Jeong, H., Lee, J., Yeom, K. W., Shin, H. J., & Park, J. H. (2008). Select-and-point: A novel interface for multi-device connection and control based on simple hand gestures. Proc. CHI '08 Extended Abstracts on Human Factors in Computing Systems (pp.3357–3362), Florence, Italy.

Madhavapeddy, A., Scott, D., Sharp, R., & Upton, E. (2004). Using camera-phones to enhance human-computer interaction. Proc. UbiComp'04, Nottingham, England.

Marquardt, N., Diaz-Marino, R., Boring, S., & Greenberg, S. (2011). The proximity toolkit: Prototyping proxemic interactions in ubiquitous computing. Proc. UIST'11, Santa Barbara, California, USA.

Martinez-Maldonado, R., Kalina, Y., & Judy, K. (2015). TSCL: A conceptual model to inform understanding of collaborative learning processes at interactive tabletops. International Journal of Human-computer Studies, 83,62–82. doi:10.1016/j.ijhcs.2015.05.001

Maunder, A. J., Marsden, G., & Harper, R. (2008). SnapAndGrab – Accessing and sharing contextual multi-media content using bluetooth enabled cameraphones and large situated displays. Proc. CHI, Florence, Italy.

McAdam, C., & Brewster, S. (2011). Using mobile phones to interact with tabletop computers. Proc. ITS'11, Kobe, Japan.

Miyaoku, K., Higashino, S., & Tonomura, Y. (2004). C-Blink: A hue-difference-based light signal marker for large screen interaction via any mobile terminal. Proc. UIST, Santa Fe, NM, USA.

Mostafapour, M., & Hancock, M. (2014). Exploring narrative gestures on digital surfaces. Proc. ITS 2014, (pp.5–14), Dresden, Germany.

Muta, M., Mukai, K., Toumoto, R., Okuzono, M., Hoshino, J., Hirano, H., & Masuko, S. (2014). Cyber chamber: Multiuser collaborative assistance system for online shopping. Proc. ITS'14, (pp.289–29,4), Dresden, Germany .

Myers, B. A. (2001). Using handhelds and PCs together. Communications of the ACM, 44,34–41. doi:10.1145/384150.384159

Nacenta, M. A., Kamber, Y., Qiang, Y., & Kristensson, P. O. (2013). Memorability of pre-designed and user-defined gesture sets. Proc. CHI'2013, Paris, France.

Nebeling, M., & Dey, A. K. (2016). XDBrowser: User-defined cross-device web page designs. Proc. CHI'16 (pp.5494–5505), San Jose, CA, USA.

Nebeling, M., Mintsi, T., & Husmann, M. (2014). Interactive development of cross-device user interfaces.Proc. CHI'14, Toronto, Ontario, Canada.

Nebeling, M., Teunissen, E., Husmann, M., & Norrie, M. C. (2014). XDKinect: Development framework for cross-device interaction using kinect. Proc. EICS'14, Rome, Italy.

Noguera, J. M., Barranco, M. J., Segura, R. J., & Martínez, L. (2012). A mobile 3D-GIS hybrid recommender system for tourism. Information Sciences, 215,37–52. doi:10.1016/j.ins.2012.05.010

Patel, S. N., Pierce, J. S., & Abowd, G. D. (2004). A gesture-based authentication scheme for untrusted public terminals. Proc. UIST, Santa Fe, NM, USA.

Pears, N., Jackson, D. G., & Oliver, P. (2009). Smart phone interactions with registered displays. IEEE Pervasive Computing, 8,14–21. doi:10.1109/MPRV.2009.35

Peng, C. Y., Shen, G. B., Zhang, Y. G., & Lu, S. W. (2009). Point&connect: Interaction-based device pairing for mobile phone users. Proc. the 7th international conference on Mobile systems, applications, and services (pp.137–150), Kraków, Poland.

Radle, R., Jetter, H. C., Marquardt, N., Reiteret, H., & Rogers, Y. (2014). HuddleLamp: Spatially-aware mobile displays for Ad-hoc around-the-table collaboration. Proc. ITS'14,(pp.45–54). doi: 10.1007/s00032-013-9609-9

Radle, R., Jetter, H. C., Schreiner, M., Lu, Z., Reiterer, H., & Rogers, Y. (2015). Spatially-aware or spatially-agnostic? Elicitation and evaluation of user-defined cross-device interactions.Proc. CHI'15,Seoul, Republic Korea.

Rekimoto, J. (1997). Pick-and-drop: A direct manipulation technique for multiple computer environments. Proc. UIST'97 (pp.31–39), Banff Alberta, Canada.

Rekimoto, J., Ayatsuka, Y., & Kohno, M. (2003). SyncTap: An interaction technique for mobile networking. Proc. 5th International Symposium on Mobile HCI, LNCS 2795 (pp.104–115), Udine, Italy.

Rekimoto, J., & Saitoh, M. (1999). Augmented surfaces: A spatially continuous work space for hybrid computing environments. Proc. CHI'99 (pp.378–385), Pittsburgh, Pennsylvania, USA .

Rodrigues, F. M., Carpendale, S., Seyed, T., & Maurer, F. (2014). Bancada: Using mobile zoomable lenses for geospatial exploration. Proc. ITS'14 (pp.409–414), Dresden, Germany.

Roudaki, A., Kong, J., Walia, G., & Huang, Z. (2014). A framework for bimanual cross-device interactions. Journal of Visual Languages and Computing, 25(6), 727–737. doi:10.1016/j.jvlc.2014.10.002

Sanneblad, J., & Holmquist, L.E. (2006). Ubiquitous graphics: Combining hand-held and wall-size displays to interact with large images", Proc. AVI'06, (pp.373–377). doi: 10.1016/j. meatsci.2006.04.003

Sauro, J., & Lewis, J. R. (2016). Quantifying the user experience: Practical statistics for user research, Morgan Kaufman.

Schmidt, D., Chehimi, F., Rukzio, E., & Gellersen, H. (2010). PhoneTouch: A technique for direct phone interaction on surfaces. Proc. UIST, pp.13–16. doi: 10.3109/10253890.2010.504789

Schmidt, D., Seifert, J., Rukzio, E., & Gellersen, H. (2012). A cross-device interaction style for mobiles and surfaces. Proc. DIS. doi: 10.1094/ PDIS-11-11-0999-PDN

Schoning, J., Rohs, M., & Kruger, A. (2008). Using mobile phones to spontaneously authenticate and interact with multi-touch surfaces. Proc. Workshop on Designing Multi-Touch Interaction Techniques for Coupled Private and Public Displays, Naples, Italy.

Schreiner, M.,Reiterer, H.,Radle,R., &Jetter, H.-C.(2015). Connichiwa -A framework for cross-device web applications.Proc. CHI'15. extended abstracts, Seoul, Republic of Korea.

Seifert, J., Simeone, A., Schmidt, D., Holleis, P., Reinartz, C., Wagner, M., … Rukzio, E. (2012). MobiSurf: Improving co-located collaboration through integrating smartphones and interactive surfaces. Proc. 2012 ACM international conference on Interactive tabletops and surfaces (pp. 51–60), ACM. doi: 10.1093/ geront/gns070

Seyed, T., Azazi, A., Chan, E., Wang, Y., & Maurer, F. (2015). SoDtoolkit A toolkit for interactively prototyping and developing multi-sensor, multi-device environments. Proc. ITS (pp.172–180), Madeira, Portugal.

Seyed, T., Sousa, M. C., Maurer, F., & Tang, A. (2013). SkyHunter: A multi-surface environment for supporting oil and gas exploration. Proc. ITS'13 (pp.15–22), St. Andrews, Scotland, United Kingdom.

Sheehan, B., Lee, Y., Rodriguez, M., Tiase, V., & Schnall, R. A comparison of usability factors of four smartphones for accessing healthcare information by adolescents. Applied Clinical Informatics, 3 (4), 356–366. doi:10.4338/ACI-2012-06-RA-0021

Shen, C., Ryall, K., Forlines, C., Esenther, A., Vernier, F. D., Everitt, K., … Tse, E. (2006). Informing the design of direct-touch tabletops. IEEE Computer Graphics and Applications, 26(5), 36–46.

Shneiderman, B., & Plaisant, C. (2009). Designing the user interface: Strategies for effective human-computer interaction", Addison Wesley.

Spindler, M., Stellmach, S., & Dachselt, A. R. (2009). PaperLens: Advanced magic lens interaction above the tabletop. Proc. ITS'09, Banff, Alberta, Canada.

Strohmeier, P. (2015). DisplayPointers: Seamless cross-device interactions. Proc. ACE'15. Article No. 4, Iskandar, Malaysia.

Tandler, P., Prante, T., Muller-Tomfelde, C., Streitz, N., & Steinmetz, R. (2001). ConnecTables: Dynamic coupling of displays for the flexible creation of shared workspaces. Proc. UIST'01 (pp.11–20),Orlando, Florida, USA .

Wilson, A., & Sarin, R. (2007). BlueTable: Connecting wireless smart-phones on interactive surfaces using vision-based handshaking. Proc. Graphics Interface 2007 (pp.119–125), Montreal, Canada.

Xiao, R., Hudson, S., & Harrison, C. (2016). CapCam: Enabling quick, Ad-hoc, position-tracked interactions between devices. Proc. ISS'16 (pp.169–178), Niagara Falls, Ontario, Canada.

Yang, J., & Wigdor, D. (2014). Panelrama: Enabling easy specification of cross-device web application. Proc. CHI'14(pp.2783–2792), Toronto, Ontario, Canada.

Yatani, K., Tamura, K., Hiroki, K., Sugimoto, M., & Hashizume, H. (2005). Toss-it: Intuitive information transfer techniques for smartphones. Proc. CHI'05, Portland, OR, USA.

About the Authors

Zheng Huang is a PhD candidate of Software Engineering at Department of Computer Science, North Dakota State University. His research interests include Human Computer Interaction and Mobile Computing. Especially, he aims at designing the intuitive cross-device interface to improve the user experience across different digital devices.

Jun Kong received the B.S. degree from the Huazhong University of Science and Technology in 1998, the M.S. degree from Shanghai Jiao Tong University in 2001, and the Ph.D. degree from the University of Texas at Dallas in 2005, all in computer science. He is a full Professor of computer science with North Dakota State University, Fargo, USA. His research and teaching interests include human–computer interaction, software engineering and brain-computer interface.