Technical Framework Supporting a Cross-Device Drag-and-Drop Technique

Adalberto L. Simeone
Lancaster University
Lancaster, UK
a.simeone@lancaster.ac.uk

Julian Seifert
Ulm University
Ulm, Germany
julian.seifert@uni-ulm.de

Dominik Schmidt
Hasso Plattner Institute
Potsdam, Germany
dominik.schmidt@hpi.uni-potsdam.de

Paul Holleis
DOCOMO Euro Labs
Munich, Germany
holleis@docomo-euro.com

Enrico Rukzio
Ulm University
Ulm, Germany
enrico.rukzio@uni-ulm.de

Hans Gellersen
Lancaster University
Lancaster, UK
hwg@lancaster.ac.uk

ABSTRACT

We present the technical framework supporting a cross-device *Drag-and-Drop* technique designed to facilitate interactions involving multiple touchscreen devices. This technique supports users that need to transfer information received or produced on one device to another device which might be more suited to process it. Furthermore, it does not require any additional instrumentation. The technique is a two-handed gesture where one hand is used to suitably align the mobile phone with the larger screen, while the other is used to select and drag an object from one device to the other where it can be applied directly onto a target application. We describe the implementation of the framework that enables spontaneous data-transfer between a mobile device and a desktop computer.

Categories and Subject Descriptors

H5.2 [**Information interfaces and presentation**]: User Interfaces - Graphical user interfaces

1. INTRODUCTION

Users frequently perform tasks that span across multiple devices. Some devices might be particularly suited for specific sub-tasks, such as using a smartphone to take a picture. However, it might be easier to use a desktop PC to process the image and place it in a presentation. These interactions commonly occur in daily life and include other scenarios such as transferring documents, music, addresses, phone Research has focused on interactions involving mobile and situated devices [3, 4, 5] and also on how to augment the capabilities of smartphone devices [1, 6, 8]. In this paper, however, we focus on a specific type of cross-device interactions. We demonstrate how to enable spontaneous data transfer through touch gestures between a mobile and a situated device and vice versa. These interactions are characterized by two phases: 1) the user *selects* data on the source device that s/he wishes to *apply* on another target device; 2) the user identifies the target location inside the destination device and *applies* the data by performing a drag gesture involving both devices.

Figure 1: A cross-device Drag-and-Drop gesture used to transfer a phone number found on the PC to a smartphone so that it can be used to place a call. The user first highlights a phone number anywhere on the screen (a); then through a drag gesture it is moved towards the edges of the screen and into the *Drag Detector* application which captures the data and sends it to the paired mobile device (b); the smartphone receives the data and allows the user to choose how he wishes to apply the data (c); once dropped on the dialler icon, the associated application is launched (d).

We have implemented a prototype system that demonstrates a cross-device Drag-and-Drop technique in two ways, with a generic interface and with a custom application. The generic interface is realized with a *Bridge* application on the mobile phone that overcomes limitations of sandboxed environments. It allows users to share data from the mobile to the desktop. Users can *share* objects they wish to this application, from which they can be dragged towards a touch-enabled desktop screen. In addition to the Bridge, we have implemented a custom email application to illustrate how

MUM '13, Dec 02-05 2013, Luleå, Sweden

http://dx.doi.org/10.1145/2541831.2541879 ...$15.00.

Drag-and-Drop might be used if it were natively supported. The *Drag Detector Window*, a software component running on the desktop allows the system to capture cross-device Drag-and-Drop touch gestures as they enter/exit the boundaries of the screen. It is also used to allow users to drag items from the desktop system to the mobile. These two applications and user feedback of the deployed systems are described in [7]. In the following sections, the technical implementation of the framework is described.

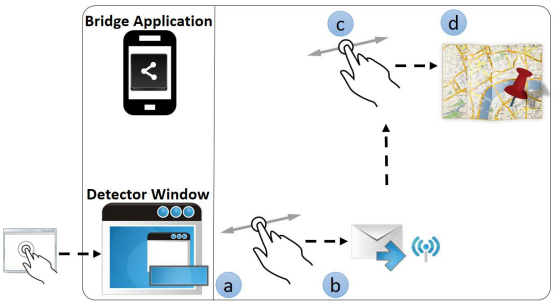

Figure 2: Sequence of events for the transfer of an address found in a webpage: (a) the user highlights the address and drags the data towards the edges of the screen; (b) once the user exits the Drag Detector window, it intercepts this event and sends the data to the Bridge application on the mobile; (c) the user continues the drag gesture inside the mobile and chooses the application class on which to drop the data; (d) once released, the data will be sent to the associated application on the mobile.

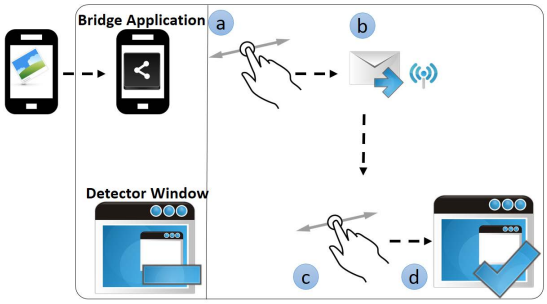

Figure 3: Sequence of events for the transfer of a picture shared to the Bridge Application: (a) the user initiates a drag gesture towards the edges of the mobile; (b) the Bridge Application detects this events and sends the data to the Drag Detector on the desktop; (c) the Drag Detector intercepts the incoming touch gesture and encapsulates the data in a local Drag-and-Drop event; (d) the user continues the drag gesture until s/he reaches the intended drop location.

2. IMPLEMENTATION

On a conceptual level the *Drag-and-Drop* technique allows users to drag data from a mobile device and drop it into a desktop screen and vice versa. It is performed by means of one single uninterrupted touch gesture and hence, it requires that both screens support touch input (see Fig. 1).

Indeed, in order to extend the Drag-and-Drop metaphor to all pre-existing mobile applications, it would be necessary to capture all touch events occurring on the device. As this is not normally possible, the *Bridge application* is a way to overcome the limitations of mobile operating systems that, for security reasons, disallow third-party developers to intercept events at the system level. The application itself is composed of two components: one running on the mobile device and the other running as a desktop application. To demonstrate how an application could be designed to accommodate cross-device Drag-and-Drop gestures, we have also designed a custom mobile email client that can respond to events originating from a different device [7]. Both applications are built on the same framework and are targeted for use on configurations involving a desktop system and a mobile device both belonging to the same user. They communicate by means of XML messages sent through the network that describe the data about to be transferred.

2.1 Bridge Application

In the Bridge application, both desktop and mobile components need to be running for the technique to work. The desktop component triggers the activation of a semi-transparent receiver window (called the *Detector window*, see Fig. 2) on the sides of the screen whenever a dragging gesture is initiated on the screen. When the user's finger leaves the boundaries of the screen the system interprets it as a Drag-and-Drop event whose source is an application running within the system and whose actual drop target is the Detector window. Thus, all content dropped within the Detector window is forwarded to the paired mobile device. By the time the user's finger moves to the mobile device from the desktop screen, metadata describing the transferred item will already be available on the target. Thus, the application running on the mobile device can use it to allow users to continue the cross-device gesture.

In the other direction, drag gestures originate from within the mobile application (see Fig. 3). When the user leaves the boundaries of the screen of the mobile device, it will be interpreted by the system as a touch release event occurring in proximity of the screen edges. This will let the mobile application notify the desktop system to trigger the activation of the detector window. If the user continues the Drag-and-Drop gesture by passing through the boundaries of the Detector window displayed on the desktop screen, the system will encapsulate the data in a Drag-and-Drop event that from the system's perspective is originating within the desktop itself (otherwise the transfer will be cancelled). Users can continue the gesture and drop the dragged data in any application that can respond to regular Drag-and-Drop events. The actual outcome depends on the receiving application.

2.2 Email Application

The custom Email application uses the framework we have built to support cross-device drag gestures. All elements of an email can be dragged outside the mobile and into the paired desktop system (see Fig. 4). In order to activate these features, the user has to perform a *long touch* on the element s/he wishes to transfer (e.g.: the subject, to/from addresses and any part of the body). Once a long touch is detected, a preview of the data being dragged appears in proximity of the user's finger. Once the user leaves the boundaries of the mobile, the preview disappears and the data is sent to the desktop system. When the user enters the desktop screen, the data is captured by the Drag Detector window and made available to be dropped on any application accepting text data. In the other direction, data can be dragged by the desktop and dropped into any field of a new email.

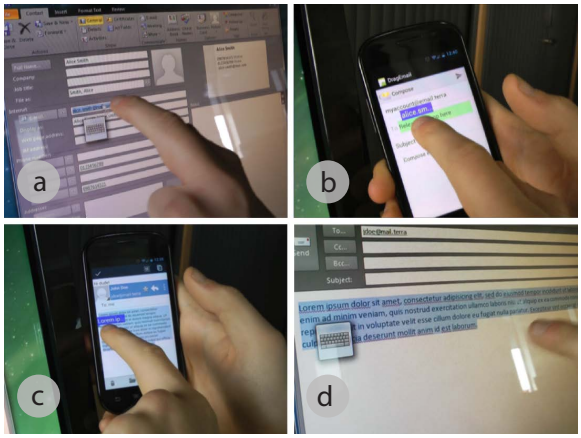

Figure 4: Two scenarios involving the custom Email application: 1) the user first selects an email address anywhere on the desktop and then drags it towards the mobile (a); once there, it can be dropped in any field accepting an email address (b); 2) in other direction, the user can select part or all of the body (c); once dragged in the desktop, it can be dropped in any application accepting text (d).

Listing 1: The following is an example of the messages exchanged by the two devices. The 'Type' attribute identifies the message type and is sent as soon as the user drags data outside the borders of the device's screen. If the data dragged is of textual nature (i.e. an address phone number or URL) the actual data is sent alongside the message otherwise it asks the receiver to enter in a binary transmission mode.

```
<?xml version="1.0" encoding="utf-8"?>
<Message Type="DragRequest" IP="127.0.0.1">
   <Resource Type="Address">
      <Value>
         Lorem Ipsum Way
      </Value>
   </Resource>
</Message>
```

3. METADATA STRUCTURE

Whenever a cross-device Drag-and-Drop gesture is initiated, the source system sends a network message to the paired system. This *message* contains metadata about the request type and on the incoming item such as its format specification, size and origin (see Listing 1). If the data can be transferred instantaneously (such as text-based data), then the information is carried within the message. Alternatively, the target system responds by means of an acknowledgement *confirmation* message to the source device so that it can return to the default state. Otherwise, it requests the target system to enter a point to point file-transfer mode.

On the desktop, if the transfer requires more time, the Drag Detector is transparently extended to cover the whole screen so that it may capture the intended drop location. Once the item is fully received, the system will simulate another drop event on the captured location. Analogously, by dropping the item within the Bridge Application on the mobile, the system will trigger the transfer and the corresponding action associated to the drop location. In both circumstances, if the data is still being transferred, the system will wait until its completion (while informing the user on the progress) and then trigger the associated action.

4. LIMITATIONS

The primary limitation of this work consists in the necessity of pairing the two devices. However, this was beyond the scope of our work as we mainly focused on extending interactions across devices. It is conceivable that existing network discovery protocols may be employed in order to streamline the pairing process [2].

Our system was originally implemented for Windows 7. The rising popularity of bevel-gestures could affect the usability of our technique. In order to work correctly, our technique does not require the user to literally drag their finger over the bevel, as long as their finger lands inside the sensible area of the receiver device. Indeed during informal evaluation, we observed how users that were more technologically experienced quickly understood that it was not an essential part of the interaction .

5. CONCLUSION

We have described the technical implementation of the framework supporting the cross-device Drag-and-Drop technique. Due to restrictions placed on mobile operating systems, users can employ the technique by means of an intermediate Bridge application that informs the target device of the incoming data through network messages . The framework can also be used to add built-in support for cross-device Drag-and-Drop gestures.

6. ACKNOWLEDGEMENTS

This work was supported by the MULTITAG project funded by DOCOMO Euro-Labs. The first author was supported by a Marie Curie Intra European Fellowship within the 7th European Community Framework Programme.

7. REFERENCES

[1] R. Hardy and E. Rukzio. Touch & Interact: Touch-based interaction of mobile phones with displays. In *Proc. MobileHCI '08*, pages 245–254. ACM, 2008.

[2] E. Meshkova, J. Riihijärvi, M. Petrova, and P. Mähönen. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Computer Networks*, 52(11):2097–2128, 2008.

[3] B. A. Myers. Using handhelds and pcs together. *Communications of the ACM*, 44(11):34–41, 2001.

[4] J. Rekimoto. Pick-and-drop: a direct manipulation technique for multiple computer environments. In *Proc. UIST '97*, pages 31–39. ACM, 1997.

[5] J. Rekimoto and M. Saitoh. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. In *Proc. CHI '99*, pages 378–385. ACM, 1999.

[6] D. Schmidt, F. Chehimi, E. Rukzio, and H. Gellersen. PhoneTouch: A Technique for Direct Phone Interaction on Surfaces. In *Proc. UIST '10*, pages 13–16. ACM, 2010.

[7] A. L. Simeone, J. Seifert, D. Schmidt, P. Holleis, E. Rukzio, and H. Gellersen. A cross-device drag-and-drop technique. In *Proc. MUM '13*. ACM, 2013.

[8] A. D. Wilson and R. Sarin. BlueTable: Connecting wireless mobile devices on interactive surfaces using vision-based handshaking. In *Proc. GI '07*, pages 119–125. CIPS, Canadian Human-Computer Commnication Society, 2007.