Welcome!

This site provides more information on the tutorial, "Cross-Device Interfaces: Existing Research, Current Tools, Outlook" by Michael Nebeling, to be held at the 2016 ACM International Conference on Interactive Surfaces and Spaces, taking place from November 6 to 9 in Niagara Falls, Canada.

Download the Slides

Click here to download the slides for the tutorial
Who Am I?

Assistant Professor
University of Michigan
School of Information
Who are you?

(Name, affiliation, your most recent paper 😊)
Tutorial Overview

It’s a journey from single-device, to multi-device, to cross-device interfaces

Mix of research and practical lessons on designing, programming, and testing cross-device interfaces

Goal: understand the requirements and enabling techniques to support cross-device interfaces

- Challenges and requirements
- Design dimensions
- Concepts and basic implementation
Why do we want cross-device interfaces?

(Applications, scenarios, settings, ... )
Applications / Scenarios / Settings

meeting rooms

interactive rooms

classrooms

hospitals

musuems

retail spaces

on the go...
Multi-Device Environments

http://idea-garden.org/
Multi-Device Environments
Multi-Device Environments ...

**Rooms** (office, home, ...)

- Equipped with various devices and sensors to track users and their activities to support tasks
- E.g., Aura (CMU), iRoom (Stanford), WILD Room (INRIA), NiCE Discussion Room (Hagenberg)
- Supported techniques are scalable with more sensors

**The User**

- Nowadays carrying and wearing multiple devices
- E.g., smartwatch, smartphone, tablet
- Needs new techniques for tracking users and their activities on the devices themselves
- Existing techniques relying on external devices and sensors don’t apply
Multi-Device Use

Serial Use

Parallel Use

Resource Lending

Related Parallel Use

Unrelated Parallel Use

Tero Jokela, Jarno Ojala, Thomas Olsson: A Diary Study on Combining Multiple Information Devices in Everyday Activities and Tasks (CHI’15)
Challenge 1: Multi-Device Interfaces
Challenge 2: Cross-Device Interaction
What are cross-device interfaces?

(How do they differ from other kinds of interfaces? Characteristics, requirements, ...)

Multi-Device Interfaces

adapted to device in use

E.g., layout based on viewing size

a popular technique is responsive design using

- fluid grid layout
- flexible images
- media queries to determine device characteristics
Multi-Device Design Approaches

Interface A → Interface B

- Graceful Degradation

Interface A ← Interface B

- Mobile-first / Progressive Enhancement

Interface A → Interface B

- Responsive / Adaptive Design
Cross-Device Interfaces

distributed and synchronized between multiple devices

support multiple interactive devices used sequentially or in parallel to support the same task

support multiple users working on the same task
Cross-Device Interaction

Cross-device interaction can be useful for...

1) combining devices for more screen real estate
2) making better use of devices with limited input/output
3) using one device to control the others remotely

...
Issues

Nowadays, many activities span multiple devices

**But:** multi-device workflows have to be planned

Two primary design issues

1) information and actions are tied to the device rather than the user

2) devices are not aware of other devices and their role in the user's larger task
Cross-Device Interfaces: Existing Research, Current Tools, Outlook

Michael Nebeling
nebeling@umich.edu
http://mi2lab.com

Cross-device
Adaptation
Multi-device
Distribution
What can be adapted?

**Presentation**

the perceivable aspects (including media and interaction techniques choice, layout, graphical attributes, ...)

**Navigation (or "dynamic behavior")**

including navigation structure, dynamic activation and deactivation of interaction techniques, ...

**Content**

including texts, labels, images

Various combinations of adaptation strategies are possible

conservation, rearrangement, increase, reduction, simplification, magnification, ...
What can be distributed?

**Input**
redirection of keyboard, mouse, touch, gesture, and voice input

**Output**
display or content redirection of text, graphics, audio, video

**Platform**
interface execution is distributed across different platforms (i.e., architectures, operating systems, networks, etc.)

**Space**
interface is distributed physically, or geographically, e.g., co-located or remote interactive spaces

**Time**
interface elements execute simultaneously (synchronously), or distributed in time (asynchronously)

N. Elmqvist: Distributed user interfaces: State of the art. Distributed User Interfaces 2011
What can be synchronized?

Input Events

- clicks (keypress, mouse, tap)
- input data changes (text fields, checkboxes, ...)
- scrolling, zooming, ...

Output Events

- interface element focus
- interface manipulation (e.g., new element)
- interface redirection (different screen)

Other Events

- device connections
- shared variables
...
How do you program cross-device interfaces?

(Frameworks, languages, tools you employ)
Frameworks, Languages, Tools

- "own"
- XD MVC (tool support)
- XD family of tools
- Commichawa
- Webstrates.net
- "Uedge" (-UsiXML)
- Chord distr?aster?
- T6D0MVC
- benchmarks.js
- Cross-device
  - models + sockets.js
  - Phantom.js
How We Program Interfaces
How We **Will** Program Interfaces
A Logical Framework for Multi-Device User Interfaces

Fabio Paternò, Carmen Santoro
CNR-ISTI, HIIS Laboratory
Via G. Moruzzi 1, 56124 Pisa
Italy
{fabio.paterno, carmen.santoro}@isti.cnr.it

ABSTRACT
In this paper, we present a framework for describing various design dimensions that can help in better understanding the features provided by tools and applications for multi-device environments. We indicate the possible options for each dimension, and also discuss how various research proposals in the area are located in our framework. The final discussion also points out important areas for future research.

Keywords: Multi-device User Interfaces; Logical Framework; Distributed and Migratory User Interfaces.

ACM Classification Keywords
H.5 Information Interfaces and Presentation,
H.5.2 User Interfaces

General Terms: Design, Human Factors.

INTRODUCTION
Nowadays, it is extremely common to see users performing their tasks using various devices ranging from the traditional stationary desktop platform to mobile devices with various multimodal interaction resources. However, by now, users’ expectations have not yet been adequately fulfilled. Often all this technological offering is not exploited as it could be, and when users perform cross-device service access they encounter various usability issues: poor adaptation to the context of use, lack of coordination among tasks performed through different devices, inadequate support for seamless cross-device task performance. For example, one potential source of found out that users already employ a variety of techniques for accessing and managing information across devices. However, there is still room for improvements, especially from the user experience viewpoint: participants in the study reported that managing information across devices is the most challenging aspect of using multiple devices. To this end, various approaches are possible. In distributed User Interfaces (UIs) we have solutions that allow users to exploit user interfaces distributed across multiple devices at a given time to access their applications. In migratory UIs users can change device and still access the application with some level of continuity, which means that at least some parts of the original user interface preserve their state after changing device. More generally, it is important to reach a better understanding of how we can design tools and applications exploiting multi-device UIs, which is the main goal of the framework that we propose.

In this paper, after discussing some relevant work in the area, we first suggest a logical framework in order to describe the range of possibilities that multi-device UIs offer, by identifying ten dimensions that have been judged relevant for such systems based on our analysis of the state of art and experience in designing multi-device environments. Then, we summarise the main points of the framework by also providing a table supporting an analysis of some proposals in this field. Finally, we conclude with some summary remarks and future directions of research currently underexplored and

STATE OF THE ART
Design Dimensions/Challenges

**Distribution**, e.g. static vs. dynamic

**Migration**, e.g. cross-platform transitions, task migration, data synchronization when switching between devices

**Granularity**: manipulation of single/groups/all interface elements across various devices to support partial/total migration

**Trigger**: system vs. user (or mixed); push vs. pull, depends on whether local or remote device is trigger

**Sharing**: single vs. multi-user, sharing by moving vs. sharing by interaction

Paternò and Santoro, "A Logical Framework for Multi-Device User Interfaces" (EICS'12)
Design Dimensions/Challenges ...

Timing, e.g. immediate vs. deferred effect

Modalities, i.e. mono vs. trans vs. multi-modality

Generation, i.e. design-time vs. run-time

Adaptation, e.g. scaling, transducing or transforming

Architecture, e.g. client/server vs. peer-to-peer

Paternò and Santoro, "A Logical Framework for Multi-Device User Interfaces" (EICS'12)
Cross-Device Programming

User Interface Design
(Panelrama, XDStudio, Weave, XDBrowser, etc.)

Application Development
(HydraScope, MultiMasher, WatchConnect, etc.)

Data Management
(Firebase/Meteor/PubNub, XDSession, etc.)
What are hot topics in cross-device research?

(Papers you’ve read, or papers you’d like to write 😊)
Research Topics

* Collaboration in multi-surface envs
* Instrumental interaction
* IoT scenarios/technologies
* Tracking users' contributions
* Smart watches/eye-wear
* Compare architecture - diversity of users
Overview of XD Research

On Cross-Device Working
Need for sequential and parallel use of devices

On Cross-Device Interaction Techniques
Design space of cross-device interfaces is large

On Cross-Device Interface Programming
Many tools for prototyping, development, and testing
Cross-Device Working

Cross-Device Techniques

Programming

Anything Else

Oulasvirta & Sumari (CHI’07)
Dearman & Pierce (CHI’08)
Santosa & Wigdor (UbiComp’13)
Kane et al. (INTERACT’09)
Karlsson et al. (CHI’10)
Jokela et al. (CHI’15)

GroupTogether (UIST’12)
Schmidt et al. (DIS’12)
Seyed (ITS’12)
Chen et al. (CHI’13)
Duet (CHI’14)
Conductor (CHI’15)
Raedle et al. (CHI’15)

XDStudio (CHI’14)
Panelrama (CHI’14)
XDKinect (EICS’14)
Fisher et al. (IJHCS’14)
Frosini & Paterno (EICS’14)
Weave (CHI’15)
WatchConnect (CHI’15)

Damask (CHI’08)
Gummy (AVI’08)
Highlight (UIST’08)
HuddleLamp (ITS’14)
MultiMasher (WISE’14)
XDBrowser (CHI’16)
Audrey Sanctorum, Beat Signer: “Towards User-defined Cross-Device Interaction” (DUI’16)
Audrey Sanctorum, Beat Signer: “Towards User-defined Cross-Device Interaction” (DUI’16)
Family of Cross-Device Systems

XD

Studio [CHI’14]
Kinect [EICS’14]
Session [EICS’15]
Browser [CHI’16]
Family of Cross-Device Systems

XD

Studio [CHI’14]
Kinect [EICS’14]
Session [EICS’15]
Browser [CHI’16]
Recent Frameworks and Tools

Interactive Development of Cross-Device User Interfaces
Michael Nebeling
Department of Computer Science, ETH Zurich
nebeling@inf.ethz.ch

Theano Mintsi
School of Computer and Communication Sciences, EPFL
theano.mintsi@epfl.ch

Maria Husmann
Moira C. Norrie
Department of Computer Science, ETH Zurich
[husmann,norrie]@inf.ethz.ch

ABSTRACT
Current GUI builders provide a design environment for user interfaces that target either a single type or fixed set of devices, and provide little support for scenarios in which the user interface, or parts of it, are distributed across multiple devices.

This paper presents XDStudio—a new, web-based design environment for developing cross-device user interfaces based on two authoring modes. In the first, a simulated mode, one device is used as a central authoring device, while target devices are simply simulated. In the second, a building mode, individual devices are connected to the authoring device to produce a distributed application.

Panelrama: Enabling Easy Specification of Cross-Device Web Applications
Jishuo Yang, Daniel Vigdor
University of Toronto
{jishuo.yang,daniel}@gup.toronto.edu

ABSTRACT
Panelrama is a web-based framework for the specification of web applications using distributed user interfaces. Our implementation provides developers with higher-level abstractions for the specification of user interfaces, including minimal changes to programming languages. Additionally, we describe a solution to ensure the liveliness and consistency of UI states across devices. We illustrate the use of Panelrama through three sample applications which use its support for known interaction methods.

Tandem Browsing Toolkit: Distributed Multi-Display Interfaces with Web Technologies
Tommi Heikkinen1, Jorge Goncalves1, Vassilis Kostakos1, Ivan Elhart1, Timo Ojala1
1Department of Computer Science and Engineering
University of Oulu, Finland
{firstname.lastname}@ee.oulu.fi

ABSTRACT
We present the Tandem Browsing toolkit that allows developers to build multi-display and multi-user applications for pervasive displays with web technologies. Existing tools for this purpose either focus on user needs, rather than developer needs, or do not rely on open web standards. Our research-based toolkit allows users to specify and control the configurations of web applications.

User Interface Distribution in Multi-Device and Multi-User Environments with Dynamically Migrating Engines
Luca Frosini
HIIS Laboratory – ISTI-CNR
Via G. Moruzzi, 1
56124 Pisa (Italy)
luca.frosini@isti.cnr.it
+39 050 621 2602

Fabio Paternò
HIIS Laboratory – ISTI-CNR
Via G. Moruzzi, 1
56124 Pisa (Italy)
fabio.paterno@isti.cnr.it
+39 050 621 3066

ABSTRACT
User interface distribution is a challenging problem in multi-device and multi-user environments. Existing solutions for distributing user interfaces across devices do not consider the dynamic migration of engines and their interdependencies. Our solution is based on the concept of engines, which are lightweight and highly portable components that can be dynamically loaded and unloaded from devices.

WATCHCONNECT: A Toolkit for Prototyping Smartwatch-Centric Cross-Device Applications
Steven Honben1,2
1Pervasive Interaction Technology Laboratory
2University College London
Rue d’Uccle 7, Copenhagen, DK
s.honben@ucl.ac.uk

ABSTRACT
People increasingly use smartwatches in tandem with other devices such as smartphones, laptops or tablets. This allows for novel cross-device applications that use the watch as both input device and output display. However, despite the increasing availability of smartwatches, prototyping cross-device smartwatch-centric applications remains a challenging task: developers are limited in the tools they can employ, and available tools provide only limited access to different types of input sensors for cross-device interactions.

This paper introduces WatchConnect, a toolkit for rapidly prototyping cross-device applications and interaction scenarios.

Weave: Scripting Cross-Device Wearable Interaction
Pei-Yu (Peggy) Chi
Computer Science Division, UC Berkeley
peggychi@cs.berkeley.edu

Yang Li
Google Inc.
yangli@acm.org

ABSTRACT
We present Weave, a framework for developers to create cross-device wearable interactions by scripting. Weave provides a set of wearable APIs, based on JavaScript, for developers to easily distribute UI output and combine sensing events and output across mobile and wearable devices. Weave allows developers to focus on their target interaction behaviors and manipulate devices according to their preferences and affordances, rather than low-level specifications. Weave also contributes an integrated and a watch [4]. Existing commercial products have begun to support basic cross-device behaviors, such as browsing an image gallery on the phone with a watch remote [14] or navigating media on the TV with a phone [12]. However, programming cross-device wearable interaction remains challenging. To create an interaction behavior that spans multiple wearable and mobile devices, developers have to design and implement both the varying input and output resources and preferred interaction styles of each device.

Multi-Device Interaction

Smartwatch Interaction

Panelrama: Enabling Easy Specification of Cross-Device Web Applications
Jishuo Yang, Daniel Vigdor
University of Toronto
{jishuo.yang,daniel}@gup.toronto.edu

ABSTRACT
Panelrama is a web-based framework for the specification of web applications using distributed user interfaces. Our implementation provides developers with higher-level abstractions for the specification of user interfaces, including minimal changes to programming languages. Additionally, we describe a solution to ensure the liveliness and consistency of UI states across devices. We illustrate the use of Panelrama through three sample applications which use its support for known interaction methods.

Tandem Browsing Toolkit: Distributed Multi-Display Interfaces with Web Technologies
Tommi Heikkinen1, Jorge Goncalves1, Vassilis Kostakos1, Ivan Elhart1, Timo Ojala1
1Department of Computer Science and Engineering
University of Oulu, Finland
{firstname.lastname}@ee.oulu.fi

ABSTRACT
We present the Tandem Browsing toolkit that allows developers to build multi-display and multi-user applications for pervasive displays with web technologies. Existing tools for this purpose either focus on user needs, rather than developer needs, or do not rely on open web standards. Our research-based toolkit allows users to specify and control the configurations of web applications.

User Interface Distribution in Multi-Device and Multi-User Environments with Dynamically Migrating Engines
Luca Frosini
HIIS Laboratory – ISTI-CNR
Via G. Moruzzi, 1
56124 Pisa (Italy)
luca.frosini@isti.cnr.it
+39 050 621 2602

Fabio Paternò
HIIS Laboratory – ISTI-CNR
Via G. Moruzzi, 1
56124 Pisa (Italy)
fabio.paterno@isti.cnr.it
+39 050 621 3066

ABSTRACT
User interface distribution is a challenging problem in multi-device and multi-user environments. Existing solutions for distributing user interfaces across devices do not consider the dynamic migration of engines and their interdependencies. Our solution is based on the concept of engines, which are lightweight and highly portable components that can be dynamically loaded and unloaded from devices.

WATCHCONNECT: A Toolkit for Prototyping Smartwatch-Centric Cross-Device Applications
Steven Honben1,2
1Pervasive Interaction Technology Laboratory
2University College London
Rue d’Uccle 7, Copenhagen, DK
s.honben@ucl.ac.uk

ABSTRACT
People increasingly use smartwatches in tandem with other devices such as smartphones, laptops or tablets. This allows for novel cross-device applications that use the watch as both input device and output display. However, despite the increasing availability of smartwatches, prototyping cross-device smartwatch-centric applications remains a challenging task: developers are limited in the tools they can employ, and available tools provide only limited access to different types of input sensors for cross-device interactions.

This paper introduces WatchConnect, a toolkit for rapidly prototyping cross-device applications and interaction scenarios.

Weave: Scripting Cross-Device Wearable Interaction
Pei-Yu (Peggy) Chi
Computer Science Division, UC Berkeley
peggychi@cs.berkeley.edu

Yang Li
Google Inc.
yangli@acm.org

ABSTRACT
We present Weave, a framework for developers to create cross-device wearable interactions by scripting. Weave provides a set of wearable APIs, based on JavaScript, for developers to easily distribute UI output and combine sensing events and output across mobile and wearable devices. Weave allows developers to focus on their target interaction behaviors and manipulate devices according to their preferences and affordances, rather than low-level specifications. Weave also contributes an integrated and a watch [4]. Existing commercial products have begun to support basic cross-device behaviors, such as browsing an image gallery on the phone with a watch remote [14] or navigating media on the TV with a phone [12]. However, programming cross-device wearable interaction remains challenging. To create an interaction behavior that spans multiple wearable and mobile devices, developers have to design and implement both the varying input and output resources and preferred interaction styles of each device.
XDBrowser

Demo
Interfaces and Tasks

15 non-technical end-users
144 desirable multi-device designs
(1) optimize for screen space
(2) optimize for input
(3) minimize device switching
6 Cross-Device Interface Patterns

- (a) Mail (Overview+Detail)
- (b) Mail (View+Input)
- (c) Media (Remote-Control)
- (d) Maps (Split)
- (e) Article (Overview+Detail)
- (f) Slides (Extend)

Async Patterns
Move and Sync Patterns
Copy and Sync Patterns
XDBrowser 2.0
New (Forms of) Devices
(smartwatches/eye-wear, hybrid vs. shapeshifting vs. modular devices)

Better Native Support
(prototyping, programming, debugging, etc.)

More Conductors
(more cross-device designers & developers)
Carrying on with the musical analogy, design has typically been preoccupied with creating new instruments. However wonderful any one of those instruments might be, the true potential is only realized when they play well together—essentially as one. It is the creativity and skill of the conductor that is essential to that happening.

The next "big thing" is not a thing. It is a change in the relationship amongst the things. Without the Conductor’s input, we are on a fast path to hitting the complexity barrier, since the cumulative complexity of a bunch of simple things—regardless of how delightful, simple and desirable they may be—will soon exceed the ability of humans to cope. It is the Conductor who carries the responsibility for the design of those relationships and ensuring that their collective value significantly exceeds the sum of their individual values, and their cumulative complexity is significantly less than the sum of their individual complexities.
What is the future of cross-device interfaces?

(What are your takeaways from this tutorial? Where do you think we need to go from here?)