Planning Adaptive Mobile Experiences When Wireframing

Qian Yang **John Zimmerman** **Aaron Steinfeld** **Anthony Tomasic**
School of Computer Science, Carnegie Mellon University, PA, USA
{qyang1, johnz, astein, tomasic}@cs.cmu.edu

ABSTRACT
Machine learning improves mobile user experience. Interestingly, envisioning apps with adaptive interfaces that reduce navigation and selection effort is not standard UX practice. When implementing an adaptive UI for our mobile transit app, we encountered a number of problems. Our original design did not log necessary information nor did it induce users to provide good labels. On reflection, we realized UX designers should identify and refine UI adaptions when sketching wireframes. To advance on this insight, we reviewed the interfaces of popular apps and extracted six design patterns where UI adaptation can improve in-app navigation. Next, we designed an exemplar set of wireframes, illustrating how UX designers might annotate their interaction flows to communicate planned adaptation and note the information (logs and labels) needed to make the desired inferences.

Author Keywords
Design patterns; interaction design; mobile interfaces.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous

INTRODUCTION

Machine learning (ML) has become an attractive and effective tool for improving mobile user experience (UX). Many of the most popular mobile apps regularly mine user behavior and context data in order to make personalized recommendations, filter out spam, predict travel time, and log behaviors like walking or sleeping. The increasing number of people who interact with these smart systems now expect more natural and intelligent interfaces [40].

The idea of improving UX with adaptive interfaces has been a topic of technical advancement. Mobile UI particularly benefits from adaptations that reduce navigation and selection efforts because (1) mobile tasks and contexts vary greatly, (2) small screens limit interaction and content space, and (3) the on-the-go usage often limits attention making every second count [4,8,10]. For example, the Starbucks app might learn a user pays with their phone and land the user on the payment screen when launched inside a Starbucks' store. Adaptive interfaces also benefit users with disabilities. These users pay a higher cost to navigate UIs in terms of cognitive and physical effort [28]. Recently, some interaction design experts have begun to claim that, "AI is the new UI"[6]. They view adaptation as the most important path forward for improving UX [6,20].

If AI really is the new UI, it seems planning interfaces and interaction flows that adapt should be a part of UX practice. Interestingly, this is extremely rare. Practice-focused design patterns as well as books and articles on wireframing do not discuss how to design for adaptation or how to collect the information needed for ML [7,15,33,39,42]. Most interaction design programs do not teach students to look for adaptation opportunities and build them into their designs. Finally, current mobile UI prototyping tools do not support the design of adaptive mobile UIs that change as they learn user preferences (see Cooper's great list of prototyping tools used by practitioners [38]).

We attempted to enhance our mobile transit app, Tiramisu, with an adaptive interface in order to reduce navigation and selection efforts. We encountered two problems that made this impossible. First, we had not logged the information needed to infer what users most likely wanted to do. Second, we had not properly motivated users to provide good labels that would support adaptation. In reflecting on these breakdowns, we realized that we should have planned for an adaptive UI when we first sketched our wireframes. By planning for an adaptive UI, we could have designed the logging systems and tailored the interactions to collect the required information.

Although challenging, recognizing learning opportunities when wireframing will be an important goal for future IxD practices. To advance on this challenge, we reviewed the interfaces of many popular mobile apps. We extracted six UI design patterns that define where and how adaptation might be applied. In addition, we generated an exemplar of a wireframed interaction flow. This flow shows both the pre-adapted and adapted interaction paths and documents the data needed to make an inference, a recovery method in the case of an inference error, and an inference quality needed to trigger adaptation. This flow functions as a boundary object, aiding designers in discussions with their development team.

This paper makes three contributions. First, we discover and promote the lack of planning for adaptive UI in current UX practice. Second, we provide six mobile design patterns showing where and how adaptations might be added. Third, we provide an exemplar wireframed interaction flow that allows design teams to capture and communicate their plans for an adaptive mobile UI.

RELATED WORK

Adaptive User Interfaces

HCI refers to learning systems that reduce UI navigation and selection as adaptive UIs. Adaptive UIs automatically adjust content, layout, or visual presentation based on detected changes in device, context, task, or demands [2,21]. They typically employ a user profile and set of user models to personalize user experiences and to address heterogeneous user needs [23].

Online personalization benefits both users and service providers [24]. Successful personalization anticipates users' intentions and reduces their navigation (number of screens) and selection efforts (number of taps and inputs). For example, Netflix's movie recommendations reduce the effort needed to find a movie. Service provider benefits include improved customer satisfaction, persuasion, and loyalty. The service design and research community has recognized personalization as an important part of building and maintaining long-term customer relationships with a view toward creating *customer lifetime value* [37].

HCI researchers have investigated adaptive UIs both for personalization and for context-aware applications. Familiar examples include automating display of desktop interfaces on mobile devices (done prior to responsive design) (i.e. SUPPLE [12]) and reducing navigation by reconfiguring menu items and toolbars based on frequency of use (i.e., Microsoft's Smart Menu and the Windows XP Start Menu [13]). More recently, interfaces now also intelligently respond to use contexts [5]. They surface information or services users likely want by monitoring contextual clues such as the user's location (i.e., Cyberguide [1]), current behavior (i.e. Sensay [41]), or previous behavior to automate task completion (i.e. VIO [45]).

The ML community describes adaptive interfaces as a special class of ML systems [23]. To build such a system, developers collect examples, extract features and labels (the values they wish to predict), train a prediction model, and test the accuracy of the predictions [32]. Developers must reformulate the adaptation they want into a problem that ML methods can directly address, engineer an adequate set of the descriptors, and work with interaction designers to develop interfaces that accurately capture the required data to generate high quality output [22]. Some researchers address this challenge by developing the ML system and interface simultaneously, thereby allowing negotiation between the ML input requirements and the actions required for users to complete their tasks (e.g., [45]).

Adaptation and User experience

Past work notes that adding adaptation can be dangerous [28]. Adaptive UIs confuse users who perceive changes in system behavior as negatively inconsistent instead of positively personal, thus increasing cognitive workload [45] and decreasing their sense of control [16,44]. In response, researchers designed visual metaphors to make adaptations more predictable (vividly exemplified by eight adaptive interface strategies applied to pull-down menus [9]).

When to adapt comes down to a tradeoff between the quality of the inferring system and the risk/reward users associate with a successful or failed adaptive action. Working with developers, designers should engage in discussions about a systems inference performance because user satisfaction does not correlate directly with high accuracy [27]. The notion of "fail-soft" [25], inference errors that still move users closer to their goal, provides a great example of how to address the tradeoff between performance and user experience. For example, Internet search engines return a list of ranked links (moving the user closer) instead of taking them to a specific page.

High-performance ML requires carefully selected set of data descriptors that can accurately and directly characterize user intention [21,22,29]. ML often needs *ground truth* labels for good and/or bad examples that can be hard to lift from implicit user actions. ML developers also seek a minimal set of descriptors because use of less relevant and irrelevant features can confound learning [23]. Likewise, being careless with samples can lead to *overtraining* and biasing the results. Therefore, interfaces must often induce users to provide explicit labels in addition to implicitly inferring features [2,13]. UX teams planning for adaptation should carefully weigh the additional effort to log implicit behaviors and induce good explicit labels, the risk of inference errors, and the opportunity for improved user experience [17,30].

Rogers argues that context-aware technologies should NOT do things for people, but they should instead engage people more actively in their current activity [35]. More discussion on better positioning of automation in interfaces as a means of optimizing costs and benefits can be found in mixed-initiative interface research (e.g., [17]). Billsus et al. [2] generated a set of design recommendations for adaptive interfaces for news readers. Below we list the ones that are most likely to generalize to mobile UIs:

- Create an effective interface that works well before the personalization is running
- Adapt quickly to what the user does/wants and monitor for changes in taste/desire
- Support multiple modes of information access (such as providing both recommended set of items and an alpha-ordered set of items)
- Make recovery from a mistaken user-action easy to execute
- Respect the user's privacy

The idea of improving UX with ML is not new, and the HCI research community has done great work to demonstrate this functionality. The maturity of this area catalyzed the concept of *AI is the new UI* [6]. Our research attempts to bridge the prior technical advances on adaptive UIs with current UX design practices that largely ignore this opportunity and the challenges it brings.

A CASE STUDY: ADAPTIVE UI FOR TIRAMISU

We deployed a transit information app, *Tiramisu*, in 2011, and since its release, thousands of users have used this app [43]. We wanted to redesign the app to simplify interaction flows in response to issues users identified. Specifically, we wanted to collapse the navigation of several screens and eliminate the need for several selection actions by inferring what information users most likely wanted.

Tiramisu has an unusual interaction flow for a transit information app. It crowdsources real-time arrival information by encouraging users to share location traces from their phones as they commute. Upon launching, the app lands on a home screen that lets users choose to see *Nearby* stops and previously created *Favorite* stops (Figure 1a). When selecting *Nearby*, users transition to a map. They select a stop to reveal its name (Figure 1b) and tap the name flag to transition to a list of the bus routes that use the stop (Figure 1c). Here they see arrival times for upcoming buses. Selecting an arriving bus launches a popup (Figure 1d) where users indicate if they plan to board this bus (*Getting on*). This transitions them to a destination screen (Figure 1e) showing a list of downstream stops. Tiramisu collects the destination so it can stop sharing the location from the user's phone as they arrive at their destination. Selecting a

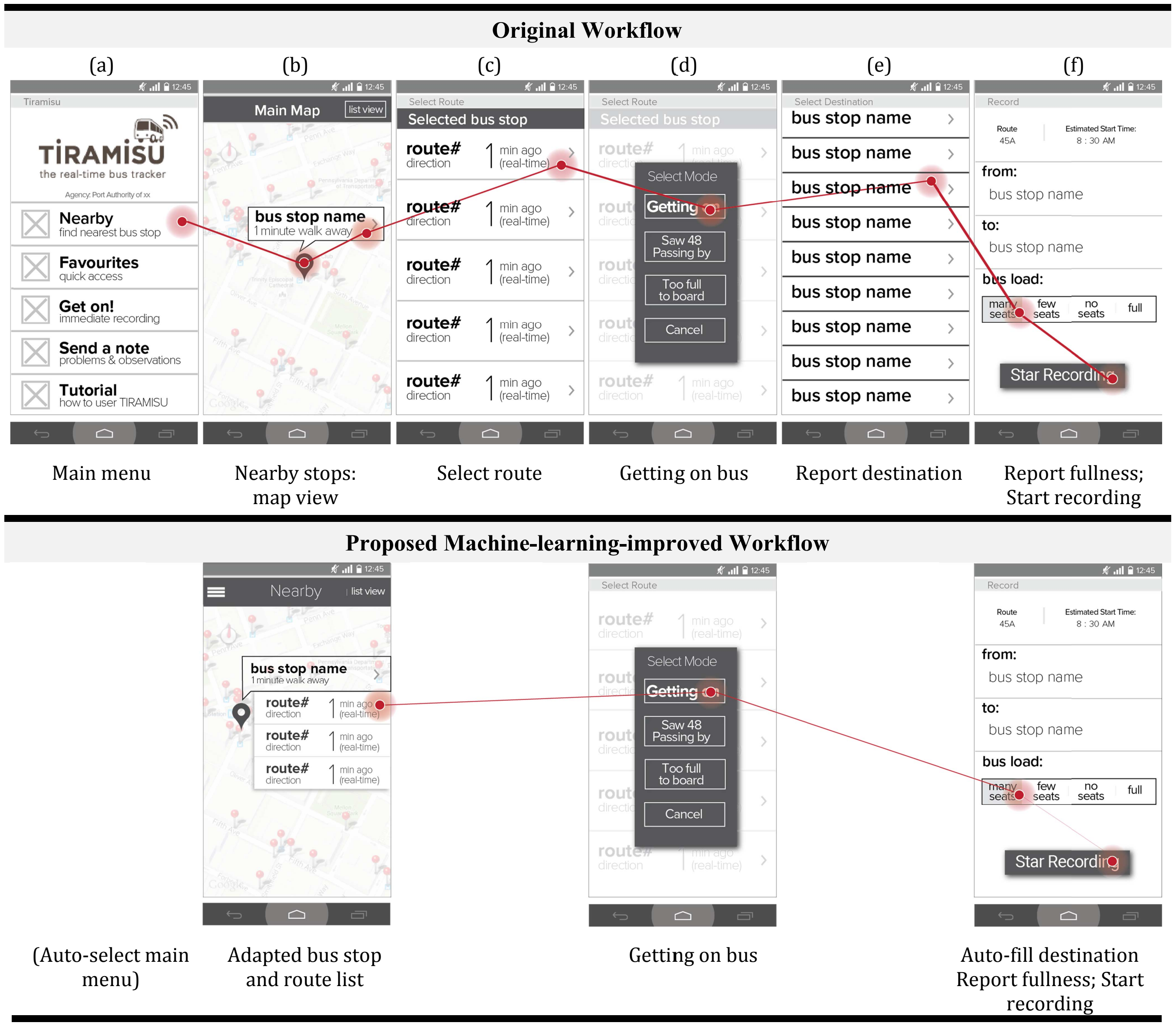
Original Workflow
(a)
(b)
(c)
(d)
(e)
(f)
Tiramisu
TIRAMISU the real-time bus tracker
Agency: Port Authority of xx
Nearby find nearest bus stop
Favourites quick access
Get on! immediate recording
Send a note problems & observations
Tutorial how to user TIRAMISU
Main Map
list view
bus stop name 1 minute walk away
Select Route
Selected bus stop
route# direction 1 min ago (real-time)
Select Mode
Getting on
Saw 48 Passing by
Too full to board
Cancel
Select Destination
bus stop name
Record
Route 45A
Estimated Start Time: 8 : 30 AM
from:
to:
bus load:
many seats
few seats
no seats
full
Star Recording
Main menu
Nearby stops: map view
Select route
Getting on bus
Report destination
Report fullness; Start recording
Proposed Machine-learning-improved Workflow
Nearby
(Auto-select main menu)
Adapted bus stop and route list
Getting on bus
Auto-fill destination Report fullness; Start recording

Figure 1. Tiramisu interface wireframes (Fonts resized for readability).

destination transitions the user to the tracing screen (Figure 1f). Here they must indicate vehicle fullness and then they can share a location trace by tapping *Start Recording*.

To see arrival times, users navigate three screens, make three selections by tapping, and scroll one list. To share a location trace needed to generate the real-time arrival information, users navigate six screens and a popup menu, make eight selections by tapping, and scroll two lists. Several users complained that it took too many screen and taps to share a trace. In addition, some complained about entering a destination. This level of effort was particularly onerous for blind users who used screen readers.

When planning the redesign, we wanted to collapse this flow to minimize navigation and selection effort. We selected three specific places to do this:

1. Infer the tab a user wanted, eliminating the home screen (Figure 1a);
2. If they wanted *Nearby*, then infer the stop, eliminating the map and stop selection and taking them right to the arrival times (Figure 1b);
3. When sharing a trace, infer the destination, skipping the destination screen (from 1e);

We assumed frequent users typically take the same trips at the same times, such as a daily commute to work. So based on time of day and location, we could infer origin and destination bus stops. Previous studies successfully predicted bus-riding patterns by mining time-location pairs from a history rides [26,31,34]. In an ideal case, the app would deliver the user to a list of desired arrival times when they launched the app. To share a trace, users would need to navigate two screens and a popup, while scrolling one list and making two tap selections. The adaptive UI would be significantly faster and require much less effort.

We encountered two problems that prevented the design improvements. First, our system did not log all the information we needed. For example, we could not tell which selection a user had made on the home screen. Our system logged calls to the database, not onscreen actions. Database calls to send arrival times could originate from (1) selecting a stop on the map, (2) looking up a stop from *Favorites*, or (3) refreshing a list of arrival times. Second, some logs contained poor quality data. Specifically, some of the destination selections users made contained the next stop on the route. We suspect that few took a bus to travel the distance of one stop. We inferred they might be simply selecting the first item on the destination screen in order to dismiss the screen and get to tracing. Both of these problems produced irreparable defects in the data.

Reflections

We reflected on the challenging state of our redesign efforts. We compared it to our previously successful pre-planned ML implementation to see if we could identify a breakdown in our process and improve future design efforts. After much discussion, we concluded that the breakdown happened when we wireframed the interaction flows.

We should have noted adaptation opportunities when sketching our initial wireframes. When wireframing interaction flows, a design team views the interface and task from the users' point of view. This perspective helps designers develop a sense of the users' intentions and identify the sets of actions users will likely repeat. Sketching should produce flow without the adaptation, to show the initial design and also adapted flows that document reduced navigation and selection. A set of flows might include one where the interface makes suggestions and others where better inference performance allows for more automation. After completing these sketches, we should have shared these with our development team so that tasks can be properly framed and plans can be made for logging the required information.

Most of the original design team members had several years of experience designing, developing, and evaluating intelligent products and services that used ML. *This was not our first rodeo*. We speculated on how much more difficult this might be for design teams with little to no experience working with ML systems. In looking at UX practice books that describe wireframing, interaction design classes that teach wireframing, and current practices of UX design teams, we observed that anticipating adaptation in order to reduce navigation and selection is not written about, taught, or a typical part of current UX practice. To help both ourselves and other UX designers with less experience, we decided to identify mobile interaction patterns that would likely benefit from adaptation.

ADAPTIVE UI PATTERNS

To help designers both recognize opportunities, we searched for mobile UI design patterns [3] that might benefit adaptation. We started by analyzing the 25 most popular mobile apps of 2015 [11]. We downloaded these and navigated their interfaces using our phones; collectively discussing and developing consensus for what constituted a pattern. As we used these apps, we also discussed other apps we regularly used as additional sources for patterns. We scoped our search to *in-app navigation* because (1) navigation is a common UI component shared among apps, (2) it is the focus of much of the wireframing stage of UX design, and (3) it does not require any changes to an operating system that most often falls well outside the purview of an apps UX team.

This process resulted in six mobile UI design patterns that signal where designers should consider adding adaption. For each pattern, we noted common UI elements along with approaches for executing the adaptation. Below we describe each pattern with its known uses and variations, the adaptation opportunities it offers, the forms it might take after learning, and any interaction challenges.

1. Landing

Use when: The landing page contains multiple subsections of function or content. Often each tab on a landing page represent a subsection; each tab links to a separate screen. Almost all applications we analyzed landed on a screen with tabs a tab panel in launching screen. Exceptions include apps with a single function, like *Boomerang*.

UI adaptation opportunity: Landing on the tab a user most likely wants based on collective and/or personal history. For *Tiramisu*, the home screen matches this pattern.

Most current content and media consumption apps, such as news, use a recommender to rank content, but they often employ a designer-selected default to land on a desired page. Adaptation would benefit users not served well by this default. Social network apps also personalize the listing of posts; however, they generally do not learn when users want to land on notifications or land on new posts.

Adapted presentations:
The wireframes below illustrate how this adaptation might unfold as predictions improve. Figure 2a shows the initial UI and Figure 2b and 2c show two stages of an adapted landing tab. The first stage reorders tabs, revealing hidden tabs based on prior usage (assuming the design has more tabs than can be displayed). The second stage automatically lands the user on the inferred tab. Automation should appear progressive and steady. The uniformity both in user experience and in the underlying data flow across the set of wireframes can reduce potential confusion.

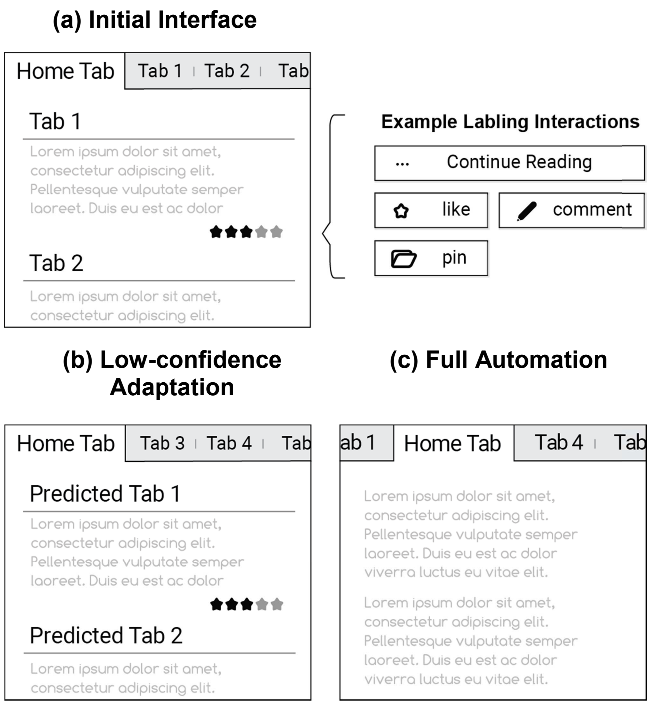
(a) Initial Interface
Home Tab
Tab 1
Tab 2
Tab
Tab 1
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque vulputate semper laoreet. Duis eu est ac dolor
Tab 2
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Example Labling Interactions
Continue Reading
like
comment
pin
(b) Low-confidence Adaptation
(c) Full Automation
Home Tab
Tab 3
Tab 4
Tab
Predicted Tab 1
Predicted Tab 2
ab 1
Home Tab
Tab 4
Tab
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque vulputate semper laoreet. Duis eu est ac dolor viverra luctus eu vitae elit.

Figure 2. Wireframe patterns for automated tab landing.

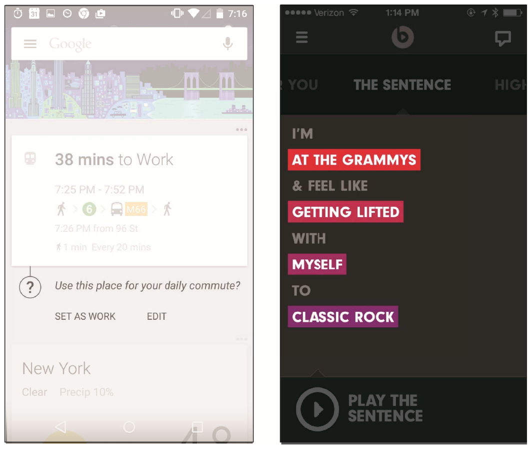

Figure 3. Interface examples that encourage explicit labeling. (a) Google Now Launcher (b) Beats music

Interaction Challenges
1). *Different triggers*: When adapting the landing, design teams should consider different scenarios. Behaviors might need to be different for a fresh launch, for bringing an app back into focus, or when transitioning in from another app.

2). *Quality labels:* On a tabbed interface, it can be difficult to infer if the app landed on the best tab simply by monitoring the time users take to navigate to a different tab. When the initial tab's visit time is short, it is difficult to infer if the user wanted a different tab or if they only needed a small amount of information from the default tab. Interfaces can explicitly ask users for labels. For example, *Google Now* (Figure 3a) prompts users with, "Is this card useful right now?" to validate a recommendation. Such direct labeling prompts should be used infrequently as they have great capacity to annoy users.

2. Single Selection

Use when: Interface contains single-selection lists, pickers (such as date and time), toggles, single checkboxes, list of radio buttons, etc.

UI adaptation opportunity: Reducing navigation or reducing selection effort situations where users must frequently make the same choice. For Tiramisu, the selection of a desired stop on the map screen and the selection of the destination when tracing both follow this pattern.

Adapted presentations: Figure 4 shows several options to reduce selection effort for a single-selection list based on the quality of a prediction. (4a) shows the initial control with a structured single-selection list. (4b) adapts the list by placing a highlight on the most-likely item. (4c) adapts the list by showing likely items at the top of the list followed by a structured list of all items. (4d) adapts the list by making a selection. Users can fix an error by tapping the control and selecting the correct item.

Interaction Challenge:
1) *Determining trigger quality:* Designers should work closely with the developers to select a performance threshold that triggers the adaptation. The design team understands the impact on user experience when the system performs correctly versus when it makes an error. The design and development teams should collectively determine thresholds.

2) *Selecting adaptation forms:* The design team needs to select a form or set of forms based on the type of choice being made. For very long lists, providing a ranked list at the top can speed interaction. When fully automating by making a selection, designers should consider how easily users can recognize an error and make a repair.

3. Form
Use when: Interface contains one or more text fields, a screen with form elements, tagging tasks, and others that have form like behaviors.

UI adaptation opportunity: Auto-completing repeated user inputs. Text input can be auto-completed by mining history. Multi-field forms, systems can monitor the first few inputs and then fill remaining items based on history of form filling. Familiar uses of this pattern include autocompleting multiple email recipients, filling in a user's contact information or address, and automatically tagging an uploaded photo as a specific person.

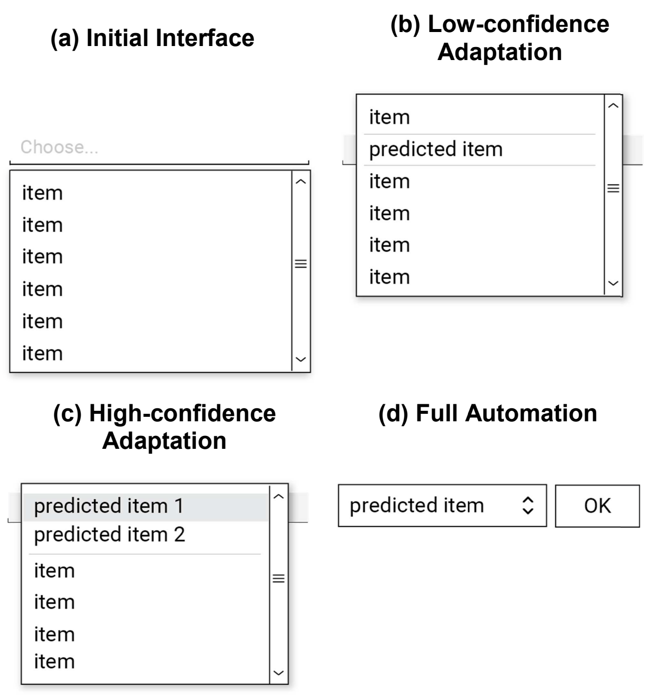

Figure 4. Wireframe patterns for automated list selection.

Interaction challenge:
1) *Recognizing errors:* HCI research shows that users sometimes struggle to recognize form-filling errors [45]. Visually distinguishing automated actions from explicit user actions can help. Also, teams can set very high quality thresholds for trigger adaptation for specific elements where an error might be catastrophic.

2) *Setting adaptation quality thresholds*: Auto-completing a longer form with no "undo" can penalize users who would need to make many individual deletions and corrections.

4. Multi-screen Transactions
Use when: A user task consists of many steps that span more than one screen.

UI adaptation opportunity: Automating the choices on one or more screens to reduce navigation and selection. Our Tiramisu redesign uses this adaption by automatically selecting the landing tab (eliminating the home screen) and automatically selecting the desired stop (eliminated the map and need to select a stop on the map).

Adapted presentations: Collapsing multi-screen transactions will be quite specific to an individual app. Amazon's "Buy-now with 1-click" feature provides a familiar example. The design uses a single selection to collapse payment selection and shipping screens, transitioning the user directly to the confirmation screen showing the item has been ordered. The Spotify music app offers another example. The app can detect the pace of a user's exercise (walk or run) and then automatically select songs that match the pace.

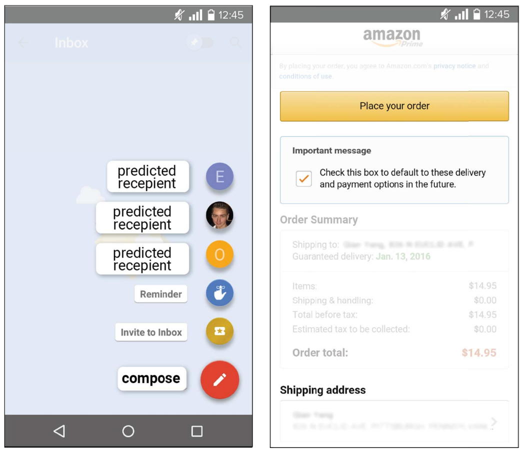
12:45
Inbox
predicted recepient
predicted recepient
predicted recepient
Reminder
Invite to Inbox
compose
12:45
amazon Prime
By placing your order, you agree to Amazon.com's privacy notice and conditions of use.
Place your order
Important message
Check this box to default to these delivery and payment options in the future.
Order Summary
Shipping to:
Guaranteed delivery: Jan. 13, 2016
Items: $14.95
Shipping & handling: $0.00
Total before tax: $14.95
Estimated tax to be collected: $0.00
Order total: $14.95
Shipping address

Figure 5. Shortcut screen transaction UI examples: (a) Gmail app provides a set of recipient shortcuts for composing new emails. (b) Amazon shopping app offers users options to automate ordering process to different extents.

Interaction challenges:
1). *Lack of Consistency:* When apps automate several simultaneous actions, it increases the likelihood of confusing users or of introducing errors that disrupt the user experience. Designers should carefully consider the value of the adaptation versus the potential for disruption.

2). *Recovery:* An adaptive UI that collapses a multi-screen transaction should provide an easy mechanism for users to recover in the event of an inference error. This recovery might be some sort of “undo” or other control that can rollback some or all of the actions taken. With highly accurate prediction models, such as spam detection, an even more complex recovery interaction may be required.

3). *Adaptation Threshold:* The inference quality threshold should be high for adaptation that makes several selections. Ways to increase confidence include single selection confirmations users make in order to trigger a multi-screen transaction and explicit user confirmation authorizing the app to automate this set of actions in future transactions.

5. User Generated and Incoming Contents
Use When: The app displays or handles sensor data; user created content such as text, audio, or photos; or content generated by people or machines that the app downloads or displays.

UI adaptation opportunity: Like multi-screen transactions, this pattern and its potential adapted layouts are often quite specific to the underlying task. Adaptation generally involves recognizing entities such as a face, a phone number or date string (as compared to other text), a voice, a user activity, etc. This pattern can be a means to automate selection, form filling, or collapsing multi-screen transactions.

Adapted presentations: Most will be specific to an individual task or recognized entity. Face detect has begun to emerge as a pattern that uses a box around a face next to a textbox showing the inferred name.

Interaction challenges: Determining the quality thresholds for triggering the adaptation. Communicating to users actions the app has taken on the users’ behalf.

6. Attention-catcher
Use when: Interruptions or requests for the user’s immediate attention, often in the form of pop-up windows and banners. Common examples include notifications; popups that encourage users to register, review, or login to the app using a social media account; and reminders to upgrade an app for an improved user experience.

UI adaptation opportunity: This adaptation attempts to determine either the best time to ask a user for this non-required information (when they are most-likely to provide it) or when it should no longer continue to ask the user for this information. The timing can be learned jointly as stepping-stones of a user-app relationship. Discovering the best timings for these requests opens up the design space to intelligently manage long-term user experiences.

Interaction challenges:
1). *Selecting default threshold:* Improperly set thresholds can be extremely annoying, so designers should be conservative, and consider if they should use this

Recognizing patterns in UI elements	Patterns	Entailed Interaction Challenges
• Landing page that contains multiple subsections of function or content, often presented by a panel of tabs;	**Landing**	• Differentiate actions for different landings; • Labeling correct/incorrect adaptations appropriately;
• Single-selection lists, including pickers, toggles and single checkbox radio buttons; • Extensive lists, including tab panel with overflow items, navigation menu used in combination with slide-out menu;	**Single Selection**	• Deliberate adaptation threshold
• Forms; Lists of checkboxes or toggles whose items are correlated; • One or more text fields or tags;	**Form**	• Differentiate actions of the agent and of the user; • Deliberate level of automation;
• A series of screens that constitute a user task;	**Multi-screen Transaction**	• Deliberate automation threshold; • Explain automation; • Ease correction;
• User inputs and uploads, including texts, audios, photos, videos and beyond;	**Incoming Content**	• Ease correction;
• Pop-up windows; • Banners;	**Attention-catcher**	• Deliberate threshold; • Allow dismiss;

Table 1: An overview of adaptive navigation UI patterns

adaptation.

2). *Allowing for easy dismissal:* Interaction flows should aid users pick up where they left off before the interruptions.

3). *Determining value proposition for nags:* Rewarding users with escalated experience with an eye on creating and managing long-term UX values.

IMPLEMENTING PATTERNS WHEN WIREFRAMING

We searched for and documented these six design patterns in order to help UX design teams recognize opportunities to add adaptation to their mobile interfaces. Once a team has identified an opportunity, they must construct interaction flows that communicate both the pre-adaptive and post-adaptive behaviors. These flows then need to be documented in a way that supports an effective exchange between the design team and the development team. To this end, we generated an annotated wireframe exemplar using our *Tiramisu* UI as an example.

We first went through several typical usage scenarios to envision a preliminary task flow. We identified adaptation opportunities: adaptive landing and three single selections (wanted bus stop, destination, and fullness). We then generated a preliminary set of features we felt would most likely be needed to make the inference. We selected these

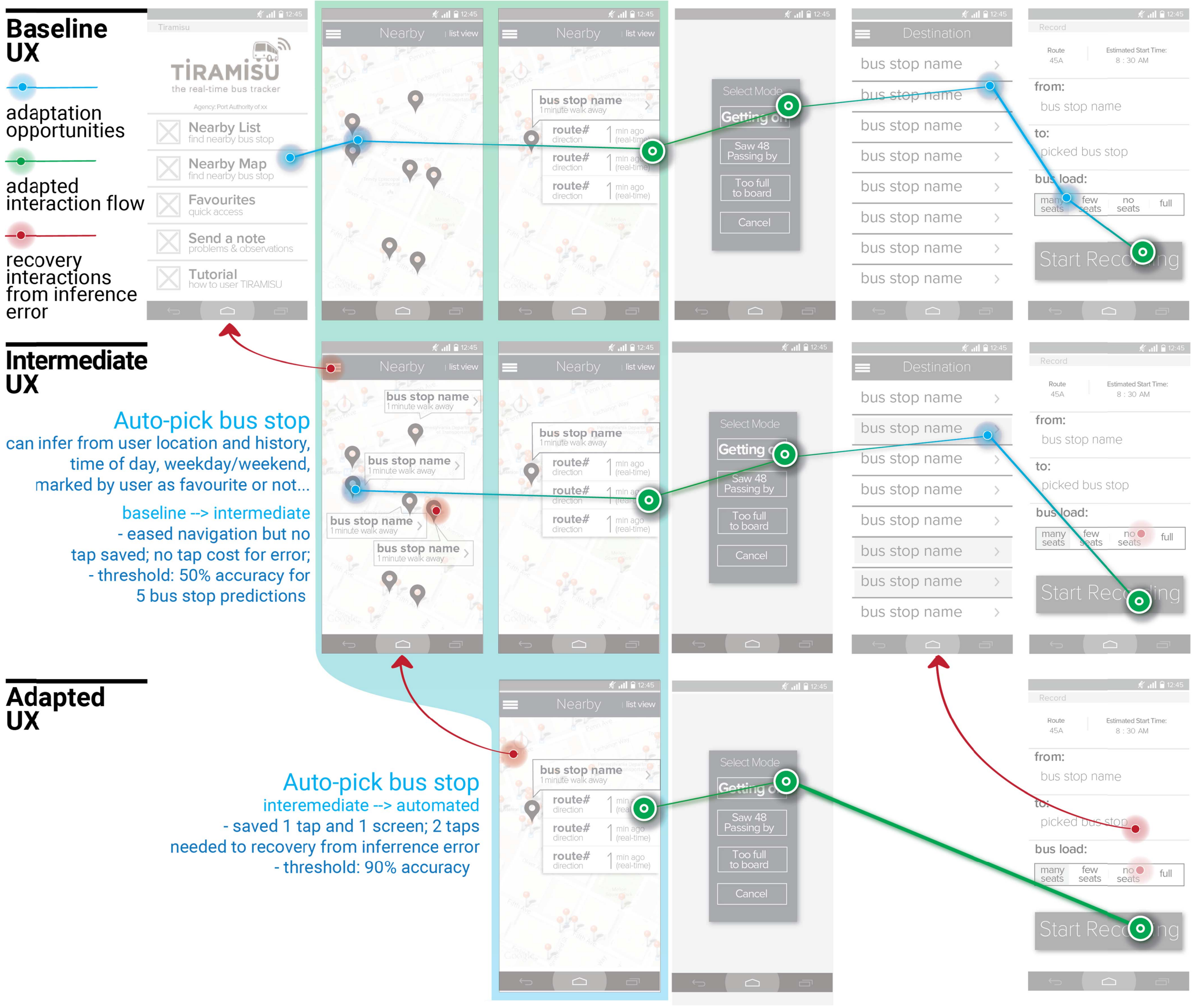

Figure 6: Proposed wireframe revision enabled by machine learning automations. Data collected over the early lifecycle of the app enabled great opportunities for machine learning to simplify in-app navigation. The blue dots represent user actions that we hope to automate; the green traces are the user actions needed when all adaptations are successful; red traces represents page transactions to recover from an inference error.

based on how we imagined most users would engage with the app the majority of the time. We wanted to capture the required features to aid in discussions with the development team on what should be logged. Next, we sketched the pre- and post adaptation interaction flows. When making the post-adaptation wireframes, we focused on users should recognize and recover from inference errors. We planned stages for the adaptation with an eye towards increasing automation as prediction quality increased. We annotated the flow across the wireframes to show the specific user actions (Figure 6 blue lines), and the steps needed to recover from an error (Figure 6 red lines). Finally, we note a rough estimation of the prediction threshold for each level of automation.

Figure 6 shows an annotated flow of wireframes generated through this process. The top shows the pre-adaption flow with highlights indicating the user selections. Below that are the adaptive flows. For each adaptation, we identified a preliminary set of features based on our understanding of user intentions and actions. The automatic selection of the most likely desired bus stop provides a good example; we suspected that the features time of day, day of week, and user location would have strong correlations. Transit rider mobility patterns often show a strong geo-temporal pattern, especially for weekdays. Whether the user has marked a nearby bus stop as a *Favorite* can also help an inference of the stop they most likely want information about. We listed these features on the wireframe.

Each stage of each adaptation, the wireframe notes the potential benefits and costs, and accordingly an estimated accuracy threshold required for triggering the adaptation (Figure 8 blue notes). For example, when accurately predicting the most likely stop a user might be interested performs poorly, the map highlights the five most likely stops (Figure6 – Intermediate UX – screen one). When none of these match the user's intention, they can continue to select the stop they want from the map with no interaction penalty. When successful, the prediction saves navigation efforts and the number of selection actions needed remain the same as the interaction with no adaptation. Considering both the benefit and cost of automation are minimal, we expect the accuracy threshold should be relatively low (i.e. 50%).

Design teams should use these annotated interaction flows as a boundary object to discuss adaptation with their development team. Both teams can use the flow as a common structure for either simulating user experiences or modeling underlying data. They can also find issues in their common interests (such as the user actions to mine for inferences and the accuracy thresholds for interface changes) annotated on the wireframe, offering starting places for both teams to build upon collaboratively.

DISCUSSION

Adaptive UIs

Everyday people use more and more mobile apps that employ ML to enhance user experience. Like others in the HCI and IxD communities, we promote the idea that *ML is the new UI*. Adaptive UIs that proactively improve user experience represent one aspect of this trend, and HCI researchers have produced a strong technical foundation through their development of dynamically adaptive UIs.

The idea of adaptive UIs has not yet become a standard part of UX design and development. One traditional challenge researchers noted with their acceptance comes from users who expect consistency in software behavior [13]. We suggest this might be less of a problem today than in the past. Today, almost all of the mainstream online tools and mobile apps continuously update their interfaces, exposing users to a frequent pace of UI change never seen with traditional desktop applications and operating systems. We suspect today's users have become more robust to interface changes and will thus more easily accept and appreciate adaptive UIs. In addition, with the increasing exposure to ML-enhanced systems, users are becoming more familiar with the possibility that systems make inference errors. Now seems the perfect time to push on the idea to make adaptive UIs a standard part of UX practice.

The software development community has learned over many years that HCI and user experience should be considered early in the development process and not added as an afterthought at the end. These considerations range from usability [18], to usable privacy [14], to accessibility [36] and even sustainability [19]. Similarly, this paper promotes the idea that an adaptive UI that learns from a user's interaction history how to reduce navigation and selection effort should also be considered at the early stages of the design process; specifically, when sketching wireframes and interaction flows. Our attempt to enhance our own app with an adaptive UI revealed an unanticipated barrier. Because we had not planned for adaptation, our initial design did not collect good labels and our system did not log the information needed to make high quality inferences. The lack of planning for adaptive UIs in current UX practice stands as a barrier to operationalizing UI adaptation technology in current products and services.

Creating effective interaction designs that both collect data from users and seed valuable adaptation remains a design challenge. Promises of adapted interactions often come at the expense of usability. In order to make confident predictions, machine-learning systems need to induce more explicit inputs from users. This tradeoff is best demonstrated by comparing a new user's experiences with *Tiramisu* and with *Transit,* a mobile app with similar functionality but no personalization (Figure 7). How can designers induce and regulate interactions in order to collect data and labels? How can interactions externalize users' intentions and capture their behaviors? How can UX teams find harmony between adaptations and user engagement? We have seen many creative solutions employed in other services that intend to produce a ready source of training data. For example, web music player *Beats* invites users to complete a narrative of "I'm (at a party), feeling like (chilling out) with (my friends) to (old school hip-hop)" and the player *plays the sentence* with recommended sound tracks. This example employs an entertaining dialog that motivates users to contribute information; information that can be used as labels for personalization.

UX designers should consider the risk of inference errors. Many open research questions remain around how to calculate the benefit of successful adaptation against the harm caused by inference errors. Much more research is needed on interfaces that make it easier for users to recognize and make repair to these errors.

We see significant opportunities in the design of new tools that support UX designers in utilizing machine learning; tools that help them step beyond the static flows they produce today. Our work provides a set of staged adaptive UIs (pre-adaptation UX, partial adaptation UX, and fully adapted UX). These will not fit into current prototyping tools that assume a single interaction flow. We encourage more design researchers to work on the challenge of integrating adaptation into current wireframing and interface prototyping tools. The prototyping tools should support designers in detailing staged adaptive interaction flows and with collecting rationales for the information that should be logged.,

ML Is the New UX

Our work narrowly investigates generating adaptive UIs, and it promotes the idea of building this into the early stages of the design process. However, adaptive UIs represent only one small way that ML can enhance UX. If interaction designers and HCI practitioners really believe that *ML is the new UX*, more work needs to be done to advance current UX education and practices.

Most of the ML enhancements to user experience, from spam filters to activity and context recognition systems, start as ideas within ML development teams. These ideas then get passed to HCI and UX teams to integrate into an interaction design. UX teams play little to no role in generating the ideas around what could or should be done. In our experience, most UX designers are not prepared to effectively envision new ML-enhanced products and services. They lack a tacit knowledge of ML as a design material. Today UX teams either never consider adding ML to enhance their designs or they think of them too much like magic and generate ideas that cannot be effectively implemented. They need more education on what ML can do, and they need practice tools that support them in investigating and communicating their design ideas.

To deliver of the idea that ML is the new UX, IxD and HCI researchers must develop new means that help UX teams develop a tacit understanding of ML as a design material. They must invent new prototyping tools that allow design teams to effectively communicate their concepts to development teams and to investigate all of the implications these concepts generate, such as the risk of errors or the need to induce users to provide explicit labels for training.

The idea that ML is the new UX opens up a large space for more and more IxD and HCI research that builds off the latest trends in industry and that offers new ideas for how the emerging power of ML can be operationalized in interaction designs. Parallel to these research efforts should be work to integrate ML as a design material into IxD and HCI education. Students coming from these programs should have an understanding of what this technology can do and of how to assess the costs and benefits of adding it into a new design.

LIMITATIONS AND FUTURE WORK

Through this paper we hope to formulate some changes to HCI and interaction design research, practice, and education that will shape how designers envision interactions. We also hope to start a serious discussion around the larger issues related to the idea that ML is the new UX. The presented work is intended to provide another start point for this emerging conversation.

One limitation of the presented work, including both the design patterns and the example of how to capture an adaptive UI in an interaction flow of wireframes, is a lack of evaluation. We generated the patterns and documentation approach to address a specific design challenge we faced with the intention that they might generalize to design teams who wish to make mobile UIs adaptive. We do not know if these patterns will aid design teams in identifying places to add adaptation or with their search for an effective interaction form. We do not know if the annotated interaction flows will generate effective conversations between ML developers and UX design teams. We will continue this work and evaluate and refine our patterns and wireframe exemplar, working with practitioners and researchers. We encourage the IxD and HCI research community to join us, and build new prototyping tools that support and shape adaptive interactions.

ACKNOWLEDGEMENT

The contents of this paper were developed under a grant from the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR grant number 90RE5011-01-00). NIDILRR is a Center within the Administration for Community Living (ACL), Department of Health and Human Services (HHS).

REFERENCES

1. Gregory D. Abowd, Christopher G. Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. 1997. Cyberguide: a mobile context-aware tour guide. *Wireless Networks* 3, 5: 421–433. http://doi.org/10.1023/A:1019194325861

2. Daniel Billsus, Clifford a Brunk, Craig Evans, Brian Gladish, and Michael Pazzani. 2002. Adaptive Interfaces for Ubiquitous Web Access. *Communications of the ACM* 45, 5: 34–38.

3. Jan O. Borchers. 2001. A pattern approach to interaction design. *AI & Society* 15, 4: 359–376. http://doi.org/10.1007/BF01206115

4. Robert Bridle and Eric McCreath. 2006. Inducing shortcuts on a mobile phone interface. *Proceedings of the international conference on Intelligent user interfaces (IUI '06)*, 327. http://doi.org/10.1145/1111449.1111526

5. Dermot Browne. 2014. *Adaptive User Interfaces*. Elsevier.

6. John Brownlee. 2015. Apple Finally Learns AI Is The New UI. Retrieved December 25, 2015 from http://www.fastcodesign.com/3047199/apple-finally-learns-ai-is-the-new-ui

7. Christian Crumlish and Erin Malone. 2009. *Designing Social Interfaces: Principles, Patterns, and Practices for Improving the User Experience*. "O'Reilly Media, Inc."

8. Razak Eun Lee, Young and Benbasat. 2003. Interface Design for Mobile Commerce. *Communications of the ACM* 46, 12: 48–52.

9. Leah Findlater and Krzysztof Z. Gajos. 2009. Design Space and Evaluation Graphical User Interfaces. *AI Magazine*, Höök 2000: 68–73. http://doi.org/10.1609/aimag.v30i4.2268

10. Leah Findlater and Joanna McGrenere. 2008. Impact of screen size on performance, awareness, and user satisfaction with adaptive graphical user interfaces. *Proceedings of the 26th SIGCHI Conference on Human Factors in Computing Systems*: 1247–1256. http://doi.org/10.1145/1357054.1357249

11. Dan Frommer. 2015. These are the 25 most popular mobile apps in America. Retrieved January 17, 2016 from http://qz.com/481245/these-are-the-25-most-popular-2015-mobile-apps-in-america/

12. Krzysztof Z. Gajos, Daniel S. Weld, and Jacob O. Wobbrock. 2010. Automatically generating personalized user interfaces with Supple. *Artificial Intelligence* 174, 12-13: 910–950. http://doi.org/10.1016/j.artint.2010.05.005

13. Kz Gajos, Mary Czerwinski, Ds Tan, and Ds Weld. 2006. Exploring the design space for adaptive graphical user interfaces. *Proceedings of Advanced visual interfaces*: 201–208. http://doi.org/10.1145/1133265.1133306

14. Jason Hong. 2009. Usable Privacy and Security : A Grand Challenge for HCI Usable Privacy and Security : A Grand Challenge for HCI.

15. Steven Hoober and Eric Berkman. 2011. *Designing Mobile Interfaces*. "O'Reilly Media, Inc."

16. K. Höök. 2000. Steps to take before intelligent user interfaces become real. *Interacting with Computers* 12, 4: 409–426. http://doi.org/10.1016/S0953-5438(99)00006-5

17. Eric Horvitz. 1999. Principles of Mixed-Initiative User Interfaces. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. http://doi.org/10.1145/302979.303030

18. Natalia Juristo and Ana Moreno. 2008. Moving usability forward to the beginning of the software development process. *Human Computer Interaction*, Bass 1999.

19. Azam Khan, Lyn Bartram, Eli Blevis, Carl DiSalvo, Jon Froehlich, and Gordon Kurtenbach. 2011. CHI 2011 sustainability community invited panel. *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems - CHI EA '11*, ACM Press, 73. http://doi.org/10.1145/1979742.1979484

20. Cliff Kuang. 2013. Why a New Golden Age for UI Design Is Around the Corner. *Wired Magazine 21.09*. Retrieved December 28, 2015 from http://www.wired.com/2013/08/design-and-the-digital-world/

21. Miroslav Kubat, Robert C. Holte, and Stan Matwin. Machine Learning for the Detection of Oil Spills in Satellite Radar Images. *Machine Learning* 30, 2-3: 195–215. http://doi.org/10.1023/A:1007452223027

22. Pat Langley. 1997. Machine Learning for Adaptive User Interfaces. In *KI-97: Advances in Artificial Intelligence*, Gerhard Brewka, Christopher Habel and Bernhard Nebel (eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 53–62. http://doi.org/10.1007/3-540-63493-2

23. Pat Langley. 1999. User Modeling in Adaptive Interfaces. *UM '99 Proceedings of the seventh international conference on User modeling*, Springer-Verlag New York, Inc., 357–370. Retrieved from http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.8715\nhttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.29.8715&amp;rep=rep1&amp;type=pdf

24. Min Kyung Lee. 2013. Designing personalization in technology-based services. Retrieved January 9, 2016 from http://dl.acm.org/citation.cfm?id=2575012

25. Henry Lieberman, Hugo Liu, Push Singh, and Barbara Barry. 2004. Beating Commons Sense into Interactive Applications. *AI Magazine* 25: 63–76.

26. Xiaolei Ma, Yao-Jan Wu, Yinhai Wang, Feng Chen, and Jianfeng Liu. 2013. Mining smart card data for transit riders' travel patterns. *Transportation Research Part C: Emerging Technologies* 36, January 2016: 1–12. http://doi.org/10.1016/j.trc.2013.07.010

27. Sean M. McNee, John Riedl, and Joseph A. Konstan. 2006. Being accurate is not enough: How Accuracy Metrics have hurt Recommender Systems. *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06*, ACM Press, 1097. http://doi.org/10.1145/1125451.1125659

28. Kyle Montague, Vicki L Hanson, and Andy Cobley. 2011. Adaptive Interfaces : A Little Learning is a Dangerous Thing. In *Universal Access in Human-Computer Interaction. Design for All and eInclusion*. 391–399. http://doi.org/10.1007/978-3-642-21672-5_43

29. Issam El Naqa and Martin J. Murphy. 2015. What Is Machine Learning? In *Machine Learning in Radiation Oncology*, Issam El Naqa, Ruijiang Li and Martin J. Murphy (eds.). Springer International Publishing, Cham, 3–11. http://doi.org/10.1007/978-3-319-18305-3

30. D. Narayanan, J. Flinn, and M. Satyanarayanan. 2000. Using history to improve mobile application adaptation. *Proceedings Third IEEE Workshop on Mobile Computing Systems and Applications*, December: 31–40. http://doi.org/10.1109/MCSA.2000.895379

31. Le T Nguyen, Heng-tze Cheng, Pang Wu, Senaka Buthpitiya, and Ying Zhang. 2012. *PnLUM : System for Prediction of Next Location for Users with Mobility*.

32. Kayur Patel. 2010. Lowering the Barrier to Applying Machine Learning. *Adjunct Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*: 355–358. http://doi.org/10.1145/1866218.1866222

33. Jenny Preece, Helen Sharp, and Yvonne Rogers. 2015. *Interaction Design: Beyond Human-Computer Interaction*. John Wiley & Sons.

34. Rudy Raymond, Takamitsu Sugiura, and Kota Tsubouchi. 2011. Location recommendation based on location history and spatio-temporal correlations for an on-demand bus system. *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '11*, ACM Press, 377. http://doi.org/10.1145/2093973.2094027

35. Yvonne Rogers. 2006. Moving on from Weiser's Vision of Calm Computing: engaging UbiComp experiences. *Ubicomp '06*: 404–421. http://doi.org/10.1007/11853565_24

36. Brian J. Rosmaita. 2006. Accessibility first!: a new approach to web design. *Proceedings of the 37th SIGCSE technical symposium on Computer science education - SIGCSE '06*, ACM Press, 270. http://doi.org/10.1145/1121341.1121426

37. Roland T. Rust and Tuck Siong Chung. 2006. Marketing Models of Service and Relationships. *Marketing Science* 25, 6: 560–580. http://doi.org/10.1287/mksc.1050.0139

38. Emily Schwartzman. Designer's Toolkit: Prototyping Tools, Reviews and rankings to help you find the best prototyping tool for the job. Retrieved January 7, 2016 from http://www.cooper.com/prototyping-tools

39. Ahmed Seffah and Homa Javahery. 2005. *Multiple User Interfaces: Cross-Platform Applications and Context-Aware Interfaces*. John Wiley & Sons.

40. Ling Shao, Caifeng Shan, Jiebo Luo, and Minoru Etoh. 2010. Multimedia Interaction and Intelligent User Interfaces: Principles, Methods and Applications. Retrieved January 13, 2016 from http://dl.acm.org/citation.cfm?id=1869926

41. Daniel Siewiorek, Asim Smailagic, Junichi Furukawa, et al. 2003. SenSay: A Context-Aware Mobile Phone. 248. Retrieved January 15, 2016 from http://dl.acm.org/citation.cfm?id=946249.946884

42. Jenifer Tidwell. 2010. *Designing Interfaces*. "O'Reilly Media, Inc."

43. Anthony Tomasic, John Zimmerman, Aaron Steinfeld, and Yun Huang. 2014. Motivating Contribution in a Participatory Sensing System via Quid-pro-quo. *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing.*: 979–988. http://doi.org/10.1145/2531602.2531705

44. Daniel S. Weld, Corin Anderson, Pedro Domingos, et al. 2003. Automatically personalizing user interfaces. *IJCAI'03 Proceedings of the 18th international joint conference on Artificial intelligence*, Morgan Kaufmann Publishers Inc., 1613–1619. Retrieved December 26, 2015 from http://dl.acm.org/citation.cfm?id=1630659.1630944

45. John Zimmerman, Anthony Tomasic, Isaac Simmons, et al. 2007. Vio: a mixed-initiative approach to learning and automating procedural update tasks. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*, ACM Press, 1445. http://doi.org/10.1145/1240624.1240843