TapGlance: Designing a Unified Smartphone Interface

Daniel C. Robbins Bongshin Lee Roland Fernandez
Microsoft Research
One Microsoft Way
Redmond, WA 98052
dcr@microsoft.com bongshin@microsoft.com rfernand@microsoft.com

ABSTRACT

The difference between using one mobile phone and another can feel like learning a new language based on our extensive experience designing mobile applications for spatial data navigation, faceted search, and glanceable information, we have developed design principles for unifying the various aspects of the internet connected mobile phone ("smartphone") user experience.

This paper presents TapGlance, a design proposal for a novel mobile phone user interface. TapGlance adapts its presentation to different levels of user attention, provides ubiquitous faceted search, and uses a zooming metaphor to unite inter- and intra-application navigation. Because our interface relies on a spatial metaphor it can also be adapted to non-textual representations and thus useful to broader populations. This paper describes our design goals, design process, and the resulting TapGlance design.

General Terms

Design, Human Factors

Author Keywords

Mobile devices, smartphones, faceted metadata, search interfaces, visual interaction, zoomable user interfaces, peripheral displays

ACM Classification Keywords

H5.m Information interfaces and presentation (e.g., HCI): Miscellaneous.

1. INTRODUCTION

Mobile phones are practically ubiquitous. Today there are 2.5 billion mobile phones and already 80% of the world's population has network coverage. It is estimated that by 2015 there will be 5 billion active mobile phones [5]. Short-message-systems (SMS) are used to keep track of crop prices in India, friends send each other photos, people retrieve movie times from web services, people watch current television shows while on the move, and yes, people do still call each other with their phones.

Given phone adoption rates in the developing world, it is very possible that many of these emerging markets will embrace internet connectivity via a mobile phone faster than they do via a desktop PC, similar to how voice enabled mobile phones leaped-frogged land-lines in many parts of the world. Whether this happens in a low-bandwidth fashion via chained SMS or a direct connection, it is happening. For many people in the world, a smartphone may become their only means of connecting to the internet. Because of this and other market pressures, both the commercial and research communities are working hard to imbue internet connected mobile devices with the rich capabilities of desktop computers, otherwise known as "smartphones." As to be expected, many current smartphone applications have adopted user interface elements from the desktop experience such as scrollbars, longs lists, arrays of check-boxes, and tiny fonts. On a traditional mobile phone the mobile operator (carrier) has near complete control over the entire user experience. On most smartphones, just as with the desktop PC, users and other third parties can install many different applications. The advantage is a plurality of capabilities; the tradeoff is that we are left with a jumble of different interaction styles and visual presentations. While users can take the time to manage this variety on the desktop, the demands of the mobile environment make this very aggravating

(a) (b) (c)

Figure 1: Overview of the TapGlance UI: (a) 9 top level "Glance" tiles show salient information from most useful feeds, (b) Interacting with the music application, and (c) showing songs that are shared amongst friends.

User interaction challenges caused by these rapidly proliferating mobile interfaces motivated us to develop a unifying visualization and interaction paradigm called TapGlance (Figure 1). We are motivated by previous research in three main areas: spatial data navigation, faceted search and, most recently, glanceable information displays. In this paper we present a glanceable, searchable, and navigable interface that can unify the most common smartphone interactions. Our contributions are a set of guidelines for unifying the smartphone user experience, a candidate design that meets our criteria, and observations about the tradeoffs in this kind of holistic design. This work is primarily based on the zoomable mobile interfaces from the ZoneZoom and FaThumb projects which we will discuss along with other related work. We also present a detailed walkthrough of our candidate interface in the context of a common mobile scenario.

2. RELATED WORK

Previous work that served as the building blocks for this project comes from the following high-level areas: Mobile phone search interfaces, mobile phone information navigation

DIS 2008, February 25-27, 2008, Cape Town, South Africa.

interfaces, generalized faceted search interfaces, and peripheral awareness displays.

Faceted search involves the use of top level categories to filter large sets of structured information. Marti Hearst has many useful recommendations for the design of faceted search interfaces [8], [21]. She suggests that, when possible, only provide those facets which apply to the most number of items in the dataset. Users have shown themselves to be adept at understanding the context of a sub-facet, so the entire hierarchy need not be displayed at all times. Her work also suggests that keyword searching be applied first across the facets themselves, then the metadata, and lastly the content itself. The Phlat project [4] used categories common to a user's own data as a front-end to a desktop search system. mSpace Mobile [22] is an extension of the desktop mSpace faceted search interface, geared especially for mobile devices. Users are presented with "fish-eyed" tiled panes, each pane returning information from a particular facet or view. mSpace Mobile currently relies on touch screen devices with fairly high resolution displays.

The FaThumb project [12] applied used faceted search interface to search across one particular database. FaThumb used a taxonomy of facets that was directly tied to the typical number keypad of a mobile phone. Zooming and animation imparted valuable perceptual feedback for navigation through the facet hierarchy. The current Live Search Mobile application [14] presents hierarchical facets although only in successively arranged lists which do not take advantage of spatial memory.

Rather than using facets, the K-Menu system [13] uses a free-text based interface as a general front-end to the entirety of a mobile phone's menu system. As users type free-text into a search box, all matching phone functions are presented in a linear list. While very general, this kind of interface takes a user's full attention – possibly not well suited to mobile environments.

AppLens and LaunchTile [11], along with FaThumb, used variations of the segmented navigation techniques developed in the ZoneZoom project [17]. ZoneZoom divided the mobile phone display into 9 tiles, each recursively zoomable via the number keys on a mobile phone. ZoneZoom only addressed navigation within a spatial dataset and did not investigate search or dynamic information displays. AppLens and LaunchTile use larger sets of tiles that are accessed either by gesture on a touch enabled device or by taps of the phone's directional keys and have now been incorporated into the commercial Zumobi UI framework [23]. Because Zumobi has more tiles than hardware keys, a user can build an association between a particular number key and a particular tile. Several systems have used various visual techniques to enable navigation amongst large and/or dense spatial datasets on a mobile device. Halo [1] uses sections of circles, centered on data-points, as indications of off-screen data locations. "Jump and Refine" [7], TouchGrid [9], Rosenbaum et al [18] and "Hopping" from Irani et al [10] use variations of the segmented navigation system to successively refine selection. Both Burigat et al [2] suggests a good framework for evaluating these systems.

In terms of glanceable interfaces – displays that can be apprehended with a minimum of attention, Pousman and Stasko [16] provide a good overview of desktop systems. Matthews et al has gone into depth on the tradeoffs between high-fidelity and abstraction in the design of peripheral displays [15]. The Scope project [20] used a very abstract set of cues to represent dynamic information sources ("feeds") on the desktop but its compete reliance on iconography limited its usability. Sideshow was a precursor to the many gadget or widget based desktop notification systems. Sideshow placed a user selected collection of highly salient information feeds into the display periphery [3]. Most of the existing research has focused on glanceable displays that lie in the user's periphery on a desktop PC. These interfaces need to get a user's attention with "just enough" prominence, while at the same time not distracting the user from highly focused tasks. Our focus, though, is on the mobile phone where the device (and display) are for short periods, front-and-center.

3. DESIGN GOALS

Upfront, we decided that TapGlance had to support the most common mobile scenarios. In our conception of this scenario, users have a standard 12-key device and the phone is only one of many stimuli clamoring for their limited attention. Additionally, users have only a limited time to devote to learning any given application. Because of this we wanted to provide an interface paradigm where users can easily navigate from application to application and within each application via a small set of shared interactions. In order to best work in a mobile setting, these interactions should take advantage of spatial memory and accommodate themselves to the amount of attention a user has to devote at any one time.

3.1 Design for Emerging Markets

There is a sweet-spot where a certain level of hardware capability (processor, memory, display, and network connectivity) becomes widely attainable by a significant portion of the population. Our first design goal is to design for this sweet-spot. When the mobile phone can couple data access with non-voice communication (text or pictographic) and when that phone can also manage a user's personal information we call that device a "smartphone." It is less the network speed that is important and more what the mobile phone can do with data, once that data hits the phone. Because of this, it is very important to us that we design an interface that while optimized for medium resolution (320x240) color displays can also gracefully scale to low resolution (128x96) gray-scale devices. This "scalability" should extend not only to hardware capability but also to user ability: the interface should still be usable the user is not fully literate. Our interface should still work if icons are substituted for text, speech input should be an alternate method for selecting options, and voice output should reinforce on-screen information.

3.2 Respect Attention

Most smartphone applications assume that a user is giving the device and the current application her full attention. The problem with this assumption is that mobile users, surprisingly enough, often use phones in mobile situations! Whether actually physically moving through the environment or engaging in a conversation with another person in the same room, mobile phone users have many attentional demands placed on them. We cannot assume that users can tune out the rest of the world in order to scroll through long lists, visually compare subtle icon differences, and keep previous application states in their short-term memory. Instead many people are using mobile phones in the context of "continuous partial attention [19]."

The takeaway is that a mobile interface should only require a "loose feedback loop." A user should be able to initiate a command, give limited attention to the device or even look away from the device, and still easily understand how the display has changed. Scrolling tends to violate this principle because it requires a tight interaction feedback loop. A user must continuously look at the display until the desired navigation goal has been reached. We also need to be careful in how we apply other technologies. If done badly, speech interfaces for mobile phones can require continuous attention on

the part of the user and this contradicts our focus on supporting short bursts of user attention. Likewise, the lack of tactile affordances and feedback on touch-screens make it difficult for the user to acquire and reliably use on-screen buttons.

3.3 Be Glanceable

As with any other user interface, a smartphone presents lists of choices to users. But a user's fleeting attention may be compromised if she is required to serially interrogate multiple items in order to confidently make a choice. It is very common for desktop applications to encourage users to serially hover over individual items, one after another, in order to ascertain indentifying information. This kind of serial interrogation – even when hover is replaced with tab-based selection – is cumbersome in a mobile setting and forces the user to rely on her short-term memory for making informed comparisons.

The goal of glanceability can be overshot in an attempt to simplify visual presentation [15]. In the peripheral notifications display of the Scope project items were only shown as a composite icon representing their most salient metadata [20]. The Scope display was glanceable in that users could easily determine if there were new items of interest and get an overview of how many items needed to be attended to. The problem, though, is that the lack of text identifiers meant that users still had to serially interrogate items in order to gain a sense of what an item actually was. It was not enough to know about an item, users wanted to see the unique identifiers for each item.

3.4 Use a Consistent Interface Metaphor

As called out previously the proliferation of mobile applications, each with their own interaction metaphor, presents learning challenges to the typical user. We can't (and shouldn't) force application authors to adhere in lock-step to a strict set of UI guidelines. But there are a base set of operations that users engage in across multiple application types. Users need to find, view, browse, edit, tag, create, and send sets of items. If we can figure out how to map each of these general operations back to application specific data and then provide a consistent means of accessing these features, we are much closer to the goal of interface unification.

3.5 Use Facets to Reunite Search and Browse

Web-based search interfaces (and their desktop descendents) have become so popular that we have almost forgotten that "browsing" is just as important a piece of sense-making as is search [8]. Faceted Hierarchical Search interfaces seek to reunite search and browse capabilities. Facets are different dimensions or property types for items in a dataset and they can be used to filter the dataset. In the previously described Phlat system, as well as in other faceted search systems, a user can interactively build a query, not just from free-text terms, but also by selecting parameter values from a set of canonical property types [4]. To see a list of files modified in the last week that were related to a particular person, a user of a faceted search system would choose the "Last Week" parameter from the date facet and the appropriate person's name from the list in the People facet. Based on the user studies in the FaThumb project [12], facets appear to be well suited for retrieving information on a mobile phone because they allow the user to avoid the cumbersome process of typing free-text on the phone to construct queries.

4. THE DESIGN PROCESS

The TapGlance project did not begin as an interface for mobile devices. Instead it developed out of various desktop information retrieval systems that combined faceted metadata browsing with full-text search [4]. Our interest in faceted search led to a series of thought and design exercises where we tried to adapt these faceted interfaces to more general configurations. We hoped that facets could be used not only for information retrieval but also as a means of issuing commands in the UI. While we quickly realized the challenges in trying to make a universal interface for all tasks on a typical desktop computer, we found that the more constrained application on a mobile device to be well suited for using facets as a central interface metaphor.

Once we moved the TapGlance project to the mobile phone, it became much easier to derive several narrative scenarios. Not only did these scenarios help us generate initial concepts but they also served as test cases once we had a detailed design. Summaries of a few scenarios are:

1. *Mike wants to see a list of all emails related to Project Beta that include Brad*
2. *Doug needs to quickly get a list of nearby supply stores that are open late in the day*
3. *Pat takes pictures of a bunch of products at a supply showroom and quickly later tags her favorites*
4. *Joe is at the park and wants to know if any of his friends are nearby so he can invite them over to play Frisbee*
5. *Linda is waiting for the bus and wants to hear music that she shares in common with Felicia*
6. *Tim wants to get a list of highly rated restaurants that are nearby*
7. *Jack shows a friend digital images of the two of them*
8. *Susan glances at her phone to see if there is anything urgent to attend to. She notices that an appointment later in the day is about Project Alpha. Wanting to gain some context, Susan retrieves a list of people who are involved with Project Alpha. Susan changes the view to show these people on a map so that she can see whose office is closest to hers. Finally she calls one of those people to arrange an informal get-together before the important meeting.*

This last complex scenario generated many interface ideas including a method to pivot from a list view of meeting attendees to a map view with the attendees plotted on the map. While this is certainly a powerful feature, it relies on deep infrastructure changes. Because of this we eventually scaled back the key scenario to center on showing an appointment location on a map. This is highlighted in the "Detailed UI walk-through" section.

Before we jumped into sketching an interface we first used these types of scenarios to derive several conceptual sub-tasks: 1) the user is presented with an overview of her most urgent information, 2) within this view she views a ranked display of calendar events, 3) then she dives into a particular appointment, 4) she extracts a piece of metadata (project name) from the appointment, 5) she filters another dataset (corporate address book) based on the project name, 6) "casts" these results into a geographic projection, and then 7) issues a command using the data (phone number) from one of the contacts.

As we began the visual design stage we examined various metaphors: tabbed interfaces, speed dependent zooming, purely graphical, and purely spatial. Our initial idea was to use a high level faceted search interface as the top-level interface. We explored how the generalized facets of people, place, time, file type, and source could be used to access all of the smartphone's

functionality. A user would retrieve different sets of information by navigating through the facet hierarchy. The problem was precisely that: a user had to navigate through a facet hierarchy just see when their next appointment was. This complex interaction contradicted one of our primary design goals, that of glanceability. Our extremely general facets were effective for creating queries, but not so effective for the primary phone tasks: making phone calls and seeing overviews of urgent information with no interaction. The need to resolve this key conflict, the conflict between generality and phone specific tasks, was at the heart of our design process. We eventually resolved this conflict by designing a way in which a general faceted search interface could exist within a targeted phone interface – all of this using a spatial zooming metaphor.

5. THE TAPGLANCE INTERFACE

5.1 User Interface Overview

The TapGlance interface is made up of a collection of panes. By default, the topmost pane, the TopBar, has keypad focus because this is where phone numbers are entered. At the very bottom of the display, as with many phone interfaces, labels indicate the functioning of the left and right soft-buttons. The user shifts focus amongst the panes by repeatedly tapping the right soft-button. Below the TopBar is a pane that contains the default set of 9 Feed Tiles (Figure 2). Each of these tiles displays a glanceable representation of the most urgent information from sources most relevant to mobile phone users. Our initial set of proposed Feed Tiles is: Search, Applications, RSS, Inbox, Calendar, Media, Scratchpad, and two tiles reserved for various dynamic alerts. For the most part, each Feed Tile's contents are populated by standing queries. For example, the Calendar Feed Tile might display the time and subject of the user's next meeting. Every tile is associated with the spatially related number key on the keypad. The two "alert" tiles are dynamically populated with information that is most relevant and urgent to the user. Typically this would be people-centric updates such as new messages or profile changes related to people that are important to the user. These two tiles would display information akin to the "Newsfeeds" feature in social networking sites such as Facebook. The dynamic nature of the alert tiles acknowledges that the idea of what is "important" is very context dependent. While the content in the two alert tiles changes over time, their spatial location is stable, thus encouraging muscle memory.

If the user needs to quickly learn more about the particular information displayed in a Feed Tile, she need merely press-and-hold the corresponding hardware key to temporarily zoom and "peek" into that tile. The "peek" view of the calendar tile would show detailed information about the next appointment along with a summary of the subsequent appointment. Upon releasing the key, the view animates back to the default set of Feed Tiles. When there is more time to devote to the phone, the user can activate any application associated with a Feed Tile by tapping the appropriate number key. Opening the Calendar application shows an overview of the day's appointments with each appointment individually selectable for more interaction.

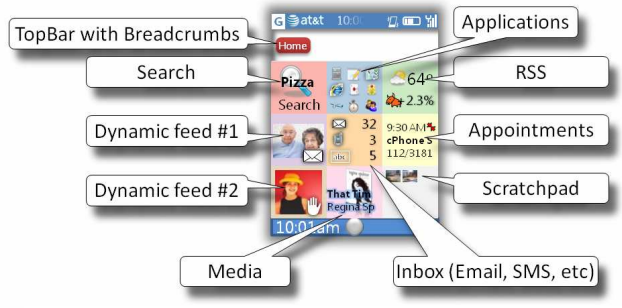


Figure 2: Home screen TapGlance UI elements

If the user wants to filter these items she can activate the Faceted Search interface from a Central Menu. When opened, the Faceted Search interface fills the lower portion of the display with 9 tiles, each an access point to a canonical category of metadata (Figure 1(c)). As different filters are applied to the current collection of items the view style within the Feed Tile changes accordingly. If the user filtered the list of meetings by location, the timeline view of the meetings would automatically transition into a map view.

Commands such as open, edit, and share are accessed via a Central Menu. This menu can be activated at any time by pressing the key at the center of the d-pad. As much as possible, the items on the menu remain constant across the UI.

Because TapGlance is targeted at a 12-key smartphone, every displayed item can be selected or activated via a series of numerical key presses. These selections are scoped to the currently focused pane.

5.2 Addressing Our Design Goals

In this section we discuss how the TapGlance UI addresses the design goals stated in section 3.To reiterate, we want our interface to: 1) take advantage of the most common hardware platform, 2) respect the user's attention by throttling the amount of information to match a user's degree of attention, 3) design the information displays to be glanceable, and 4) provide a consistent user interface paradigm across and within the base set of applications.

Our common hardware platform has nine number keys arranged in a 3x3 grid. Likewise, the primary components of our interface, the Feed Tiles, the search facets, and menu options are arranged in a 3x3 grid. This serves two purposes. The spatially stabile location of sub-region content within the grid leverages spatial memory, enhances glanceability, and ensures keyboard accessibility. And as in the ZoneZoom project, spatially arranged views, such as maps and photo-grids, can be recursively navigated via subdivision into 9 sub-regions [17]. Tapping a number key animates a zoom into the desired sub-region. Pressing-and-holding that same key previews the zoom and releasing the view zooms back to the initial view.

There are cases where the user may have the time and attention to interact with larger lists of choices. In these cases we do provide a few special case views that contain larger numbers of items. Even in these cases, our TapGlance design proposal overlays numerical access to the most commonly desired choices. Other systems are also using a zoomable tile metaphor [11]. These systems, though, have more tiles at a given zoom level than there are hardware keys on the mobile device. Because of that, we fear that users will be hampered in building up a mental model of the data space.

5.3 Abstraction in the User Interface

As stated in our goals, respecting and accommodating the user's degree of attention is primary to our design. Our first goal is to support the most ephemeral interaction – pulling a phone out of one's pocket to see if there is anything important to attend to. For this "glance" level of interaction each Feed Tile shows only one or two items in a very stylized manner (Figure 3(a)). This representation is optimized for the particular type of information. For communications (the "Inbox" tile) we show just the name of the person who sent the most recent urgent email and a numeric count of unread communications. The calendar tile shows just the name of the next appointment and the media "glance" tile shows the name of the currently playing

(a) (b) (c)

Figure 3: Three levels of abstraction for the media tile: (a) "Glance" view just shows basic information for the current song, (b) "Peek" view shows much more information about the current song, and (c) the "Application" view shows the current playlist.

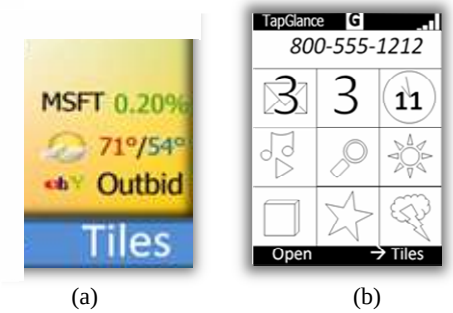
song. The design of these tiles borrows from the rich set of work already underway in glanceable peripheral displays including the Scope [20], and Sideshow [3].

When a user wants to see details about the information presented in a particular feed, the user presses-and-holds a spatially associated key to temporarily "peek" in on a particular facet. When "peeking" into the media tile, the user is shown the artist's name, song length, genre, album art, and what the next song to play will be (Figure 3(b)). When the user releases her finger, the screen zooms back out to the Feeds home page. This spring-loaded interaction supports users who have a limited amount of attention to devote to the phone but want a little more detail about the most urgent items.

When the user has the time to devote more attention to the smartphone she taps the number key associated with a particular feed. This causes the interface to switch and zoom into that feed's related application. For instance, zooming into the media tile activates the media application where the user can inspect and interact with the current playlist (Figure 3(c)). Both peeking and inspecting are carried out with quick and fluid zooming animations which shift user comprehension of the transition to the perceptual level. This differentiation between pressing-and-holding and tapping was successfully used in the ZoneZoom mobile interface [17].

5.4 Supporting Emerging Markets

We can also use abstraction to adapt TapGlance to hardware and users of varying abilities. If we replace the text in our UI with icons or pictograms, we can both support lower capability displays and users who may have limited literacy (Figure 4(b)).



(a) (b)

Figure 4: TapGlance scales to different audiences: (a) full color with text at 320x240 and (b) black and white with icons at 128x96

The density of information displayed in each Feed Tile decreases but our overall metaphor of segmented zooming is still intact. Speech interfaces can also help to empower currently underserved users. Speech input actually has a chance of being useful for selecting Feed Tiles, facets, results items, and menu options because TapGlance only presents nine direct choices at any time.

5.5 Faceted Search

Throughout the user experience, TapGlance presents users with collections of items. At the glance and peeking level of the interface we show only a very small set of items. But at the application level, when the smartphone has more of the user's attention, the number of items can be arbitrarily large. A calendar stretches infinitely, a media playlist can contain many songs, and a map can have a large number of points-of-interest markers on it. Users need easy ways to filter these sets of items and to find items amongst these large sets.

As stated before, TapGlance borrows from the methods used in the FaThumb system to provide filtering [12]. As a stand-alone application on the smartphone, the FaThumb system was restricted to browsing and searching the Yellow Pages. The TapGlance faceted search design goes beyond FaThumb in several ways. TapGlance simplifies the interactions required to add and subtract filters. The filtering mechanism is accessible from all places in the phone interface via the common Central Menu. The facets we chose are meant to encompass all of the structured data generally accessible via a typical smartphone. Just as with FaThumb, TapGlance uses a 3x3 grid of facet tiles that are tied to the spatial layout of the hardware number keys. A summary of our top-level facets and their usage can be seen in Table 1 and an image of them in context of the UI can be seen in Figure 1(c) and Figure 5(3). In general, we reserve the ninth tile as means of accessing less commonly used sub-facets.

We did consider other layouts for the facet hierarchy. Existing mobile search interfaces (and even the IPod) typically present users with separate cascading screens of hierarchically linked lists. This scheme does maximize the amount of hierarchy seen at one time. But, because the results set is not seen at the same time, users do not get real-time feedback as to how changing query terms affects the results set. In TapGlance we consistently devote a portion of the screen real-estate to search results. As seen in Figure 1 (c) when a user zooms into a sub-facet in the lower Facet pane, the middle Feed pane updates its contents appropriately thus informing the user as to whether or not her search is yielding useful results. In addition visual tokens ("breadcrumbs") appear in the TopBar to reinforce the current query.

Supporting a multi-path filtering process is central to our design. We cannot know a priori how every user is going to approach a given information retrieval task. Some may think of appointments more in terms of time, some by location, and some by people. Because of this rich variability, it is very important to allow a user to apply any filters in any order. Of course, with the right visual cues, we hope to guide users such that they never end up with the dreaded "zero results" view. It is not even that we want multiple paths to lead every user to the same end collection of filters. For some users, as suggested above, a map may be a meaningful visualization. For others, a list, with appropriate metadata in an attached column might be more easily parsed. In any case, our TapGlance design supports many ways of slicing, dicing, and viewing data.

In the previous FaThumb application, applying filters required two steps. First a user would have to navigate into a desired filter (Top level facets → Location sub-facets → "10 Blocks"). At this point, the user would then "pin" the "10 Blocks" filter by explicitly tapping on the left-soft key. In practice, we found this two step process cumbersome. Our original intent in relying on this multi-stage interaction was so that users could explore

Facet Name	Finds:	Example sub-facets
Commands	Common commands across integrated applications	Create, Edit, View
People	Content authors, contacts, IM buddies, and etc.	Friends, Co-workers
Date	Creation, modification, and viewed dates	Today, Last week
Type	Files on device or in cloud based stores	Document, Photo
Tag	Hierarchical tag sets	Travel, Shopping
Location	Items with geo-coded metadata and location based web searches	Neighborhood, City
Property	General meta-data across all items	Size, Status, Rating
Favorites	Previously "pinned" items	Recent, Frequent
More…	Head-room for facets that aren't used as frequently	

Table 1: Default filter facets and their general uses

the facet hierarchy without applying facets. We decided, though, that if it is easy enough to remove a filter, then it is better to support immediate filter application. This is in fact what occurs in other existing (and successful) faceted search systems such as Flamenco [8] and Ebay Express [5]. In TapGlance the parent facet of the current set of sub-facets is always automatically applied as a filter.

The filtering interface also uses the notion of peeking. If instead of tapping on a filter, the user presses-and-holds the associated key, the adjacent list of results is only temporarily filtered. The user can essentially preview the effect that filtering the current result list would have – without having to fully commit to the filter. This spring-loaded filtering, akin to the spring-loaded navigation of ZoneZoom, is a nod toward tentativeness and exploration in the UI.

5.6 Taxonomy Design

An expert in hierarchical faceted search, Marti Hearst, says that even with the state of the art in clustering algorithms, faceted search UIs remain "boutique search interfaces [8]." This is because faceted search is best suited to datasets that are structured and have rich metadata on them. Luckily, data accessible via a smartphone tends to be well structured. Users quickly need to find contacts, appointments, recent communications, and map locations. In the TapGlance design we choose to abstract the specific parameters on each data type into higher level (and more useful) sub-facets. If we were to strictly categorize each item based on its base level of metadata, we would end up with a very deep hierarchy. For example, the date field for a photo taken by the camera might be slightly different than the received date for an email message. Using the same kind of abstraction strategy as in the Phlat system, we decided to roll-up such parameters as "Date" and "People" [4].

Our faceted search interface is meant to be accessed from a mobile device. Because of this, it was very important that every aspect of the UI take the best advantage of spatial memory. Thus, we chose to not follow Hearst's suggestions to only show the most salient sub-facets. If our interface followed such a dynamic presentation, every use of the faceted search UI could possibly present user's with wildly varying sub-facets. This would require a high-cognitive load on the part of the user – not a desirable thing for a mobile UI. Instead, we tend to err on the side of generality. An ad-hoc (at this point) analysis of the kinds of data we need to access and the commands with which to transform the data suggested a few top level categories. This means that in TapGlance there are cases where a user will see sub-facet tiles that have a result count of zero. In our defense, aside from the benefit to spatial stability, even the presence of a "zero count" can help to teach a user about the distribution of data. For example, when a user constructs a query such that the "African" sub-category of the "Restaurant" facet shows a "zero count" in a given neighborhood, they have just learned that they should probably look elsewhere for Injera bread. In addition, the stable presentation of the facet hierarchy can help a user learn the overall taxonomy.

There are many ways, though, in which we gladly follow Hearst's recommendations as these worked well in the FaThumb project. In brief, when there is enough space, our TapGlance design incorporates predictive counts "cuddled" next to the facet. We also fully integrate free-text search: if a user moves the focus to the TopBar, typing on the number key-pad enters free-text query terms via typical mobile phone text-entry method such as T9 or multi-tap. These free text terms will first search across the names of facets and sub-facets. These "facet hits" are shown as visual cues superimposed over appropriate facet tiles. In our design, the search is also extended across the metadata for all data items accessible via the phone and across the full-text content of these items. The division of labor between the client (the phone itself) and a server is gated by network latency, size of the local phone cache, and local processor speed in executing queries.

In general, when we have a choice, we have opted for a looser, rather than tighter, construction of the facet hierarchy. We firmly believe that the benefit of returning too many results outweighs the cost of returning too few or even no results. Our taxonomy, especially the "Tag" facet, explicitly allows for sub-facets living in multiple places in the hierarchy.

5.7 Animation in the UI

The aforementioned interactions results in many state changes in the user interface. Because of this TapGlance relies on animated transitions. An example state change serves to illustrate this. Let us assume that the user is currently viewing a list of people and that she wants to see where these people are located on a map. The list of people could have been generated in multiple ways: favorite contacts, recent phone calls, or attendees at a meeting. To do this, the user opens a Central Menu (via the "Action" key). From this menu they invoke the "View Style" sub-menu and choose "Map." The animation between these states happens as follows: The people list shrinks to sit at the location of the "People" facet, a location meant to reinforce the association tile in Figure 5(1-4). The animation continues with the Facet tiles zooming up so that the "Location" tile fills the screen tile in Figure 5(5-6). As this zoom happens, an actual map display fades in over on top of the map tile and gains focus tile in Figure 5(6-7). Lastly, the individual people items animate to their appropriate locations on the underlying map tile in Figure 5(7-8).

6. DETAILED UI WALK-THROUGH

We now walk through a detailed interaction sequence in the TapGlance UI, again using the previously stated scenario. The key presses indicated assume 1) two "soft-keys", 2) a directional pad of four keys (d-pad), 3) a center "action" key, 4) a dedicated "home" key, 5) a dedicated "back" key, and 6) of course the twelve standard number keys (including "#" and "*"). In this description, we will refer to cells on a 3x3 grid by the corresponding number on the phone keypad. While this walkthrough may seem overly exhaustive, this is the vital to the design process. It is only by looping back through the initial scenario that we can make sure our candidate design is still valid.

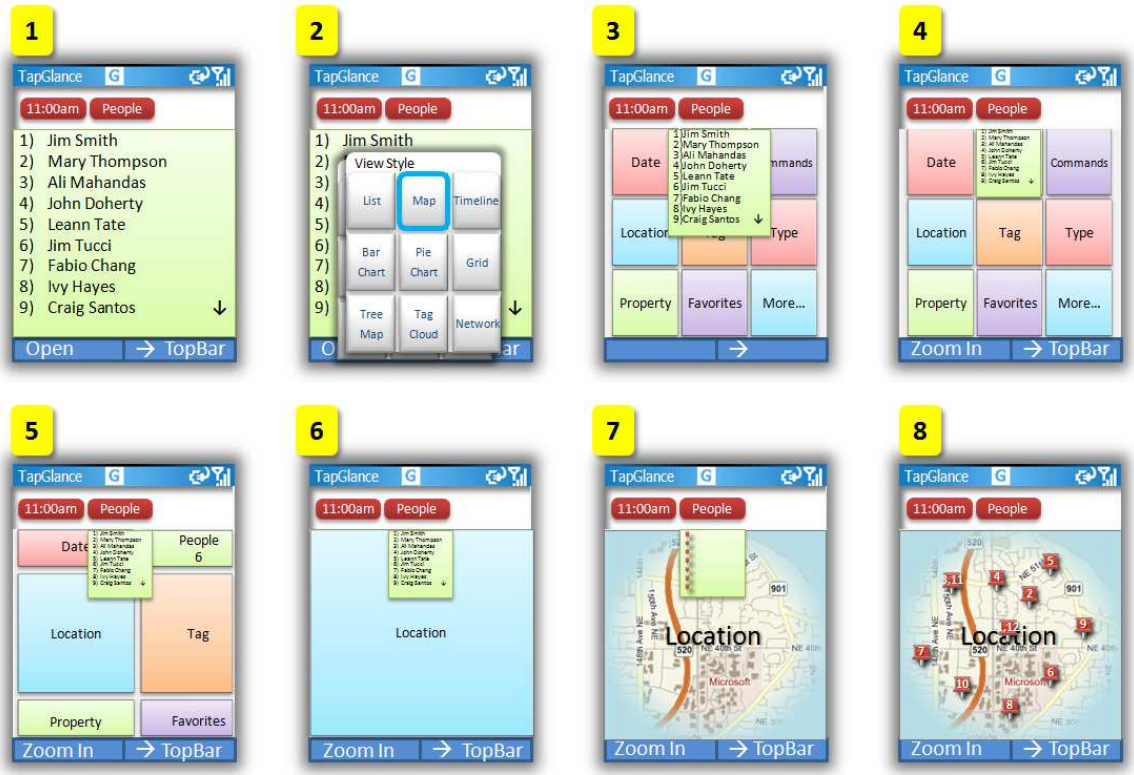


Figure 5: An animation sequence for the transition from a list view to a map view: (1-3) The user chooses the "Map" view style from the Central Menu, (3-4) the people list shrinks to nest inside the people tile of the underlying Facet view, (5-6) the Facet tiles expand to fill the display, and in (7-8) each person animates from its list position to its map position.

Immediately after parking her car at a client's business, Susan pulls the TapGlance enabled smartphone out of her pocket (or handbag) and glances over the home screen. She wants to know what her morning has in store so Susan **presses-and-holds** down the 6 key which spatially corresponds to the calendar tile (Figure 6 (2)). While she holds down the 6 key, the calendar tile temporarily zooms to fill the screen. The temporary display shows the location and attendees for her first appointment and a synopsis of the subsequent appointment.

During this ephemeral interaction, Susan realizes that her second appointment is in another building. To get a better idea of how to get to the next meeting Susan, she **taps** on the 6 key to enter the full calendar application (Figure 6 (4)). The content portion of the display now shows her daily calendar. Susan scrolls down to the second appointment by using the d-pad and taps the left soft key which is currently labeled "Open." The display then zooms so that only the selected appointment fills the screen (Figure 6 (5)).

Susan then opens the Central Menu by tapping the "action" key (Figure 6 (6)). Susan taps the 8 key to call up the sub-menu for "View" and then taps the 8 key again to select the "Map" style option (Figure 6 (7)). Once Susan learns the Central Menu, she can quickly double-tap the 8 key to change to a map view at any time. The map animates in to show both Susan's current location and the location of the selected appointment (Figure 6 (8)).

This scenario highlights the user's needs and how TapGlance meets them. TapGlance supports ephemeral interactions by tying spring-loaded navigation to the press-and-hold action. TapGlance provides graphical cues to aid in context reacquisition. TapGlance supplies a filtering interface that works across every application and in every view. And throughout the entire user experience, TapGlance uses a common navigation mechanism: zooming into sub-regions of the display.

7. OBSERVATIONS AND FUTURE WORK

We have presented TapGlance, a unified smartphone user interface where users can accomplish many mobile information tasks, at various levels of detail, via a common interface. We were surprised that, so far, the design of the navigation system has proven to be the trickiest aspect of the TapGlance project. There are a limited number of hardware keys and many of them, either by convention or by OS constraint, are already reserved. This means that we often had to overload many meanings onto one key.

Our next step with the TapGlance project is to integrate an existing faceted search application with an existing glanceable home screen application. We will also extend our existing faceted search application to handle the broader set of structured data accessible from a smartphone. We also intend to improve the navigation and searching of the facet hierarchy itself. Incorporation of a "similarity engine" would allow users to use synonyms for terms in our own hierarchy.

In a powerful sense TapGlance enables mash-ups of the smartphone user experience. Existing smartphones present disparate silos of information: contacts, calendar, and communications. Our TapGlance design proposes a way in which users can combine and visualize data from across multiple silos. As an example, the user can easily find a set of people associated with an appointment, display these people on a map, and then filter that initial set based on another set of criteria. A hierarchical faceted search interface can be used throughout the TapGlance experience to filter any of the structured information available from the smartphone.

Figure 6: (1-2) press-and-hold the 3 key to temporarily zoom into the calendar tile, (3-4) tap the 3 key zoom into the calendar application and use the D-pad to select the second appointment, (5) open an appointment by tapping the left soft-key, (6) open the Central Menu by tapping the "Action" key, (7) from the View sub-menu, select the "Map" style, and (8) the view changes to show a map the encompasses both the user's current location and the location of the selected appointment.

Commonly used commands can be invoked from a spatially arranged menu system. All of this is consistently accomplished by tapping phone number keys to zoom into and amongst spatially stable sub-regions of the display. Distinguishing press-and-hold interactions allows a user to preview the result of actions such as selection, navigation, and filter application. Our organization of the most salient information into 9 high-level feeds ensures that users need only glance at the TapGlance home-screen to learn what items most need attention. We have applied, in a novel way, segmented spatial zooming to both faceted search and application navigation.

We have presented TapGlance, a unified smartphone user interface where users can accomplish many mobile information tasks, at various levels of detail, via a common interface. TapGlance combines segmented zooming navigation and ubiquitous faceted search. By leveraging spatial memory and adapting to a user's attention, TapGlance is usable by a broad population.

8. ACKNOWLEDGMENTS

This design rests on very fruitful collaborations with Amy Karlson, Ben Bederson, John SanGiovanni, Susan Dumais, Ed Cutrell and the VIBE group. Bringing actual code to life on real smartphones relied almost entirely on the contributions of Eric Rudolph and Raman Sarin. Lastly, thanks family for letting me focus on this paper while life continued around me, and greeting me with open arms when I emerged.

9. REFERENCES

[1] Baudisch, P. and Rosenholtz, R. 2003. Halo: a technique for visualizing off-screen objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, USA, April 05 - 10, 2003). CHI '03. ACM Press, New York, NY, 481-488.

[2] Burigat, S., Chittaro, L., and Gabrielli, S. 2006. Visualizing locations of off-screen objects on mobile devices: a comparative evaluation of three approaches. In *Proceedings of the 8th Conference on Human-Computer interaction with Mobile Devices and Services* (Helsinki, Finland, September 12 - 15, 2006). MobileHCI '06, vol. 159. ACM Press, New York, NY, 239-246.

[3] Cadiz, J. J., Venolia, G., Jancke, G., and Gupta, A. 2002. Designing and deploying an information awareness interface. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work* (New Orleans, Louisiana, USA, November 16 - 20, 2002). CSCW '02. ACM Press, New York, NY, 314-323.

[4] Cutrell, E., Robbins, D., Dumais, S., and Sarin, R. 2006. Fast, flexible filtering with phlat. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada, April 22 - 27, 2006). R. Grinter, T. Rodden, P. Aoki, E. Cutrell, R. Jeffries, and G. Olson, Eds. CHI '06. ACM Press, New York, NY, 261-270.

[5] Ebay Express, *Ebay Express web site*, http:// http://www.ebayexpress.com.

[6] GSM Association, Universal Access: How Mobile can Bring Communications to All, *GSM Association Universal Access Report*, London, UK, 2007.

[7] Hachet, M., Pouderoux, J., Tyndiuk, F., and Guitton, P., "Jump and refine" for rapid pointing on mobile phones. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA, April 28 - May 03, 2007). CHI '07. ACM Press, New York, NY, 167-170.

[8] Hearst, M., Design Recommendations for Hierarchical Faceted Search Interfaces, in the *ACM SIGIR Workshop on Faceted Search*, August, 2006.

[9] Hertzum, M., and Hornbæk, K. (2005). TouchGrid: Touchpad Pointing by Recursively Mapping Taps to Smaller Display Regions. *Behaviour & Information Technology*, 24(5), 337-346.

[10] Irani, P., Gutwin, C., and Yang, X. D. 2006. Improving selection of off-screen targets with hopping. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada, April 22 - 27, 2006). R. Grinter, T. Rodden, P. Aoki, E. Cutrell, R. Jeffries, and G. Olson, Eds. CHI '06. ACM Press, New York, NY, 299-308.

[11] Karlson, A. K., Bederson, B. B., and SanGiovanni, J. 2005. AppLens and launchTile: two designs for one-handed thumb use on small devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Portland, Oregon, USA, April 02 - 07, 2005). CHI '05. ACM Press, New York, NY, 201-210.

[12] Karlson, A. K., Robertson, G. G., Robbins, D. C., Czerwinski, M. P., and Smith, G. R. 2006. FaThumb: a facet-based interface for mobile search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada, April 22 - 27, 2006). R. Grinter, T. Rodden, P. Aoki, E. Cutrell, R. Jeffries, and G. Olson, Eds. CHI '06. ACM Press, New York, NY, 711-720.

[13] Lee, S. E. and Lee, G. 2007. K-menu: a keyword-based dynamic menu interface for small computers. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems* (San Jose, CA, USA, April 28 - May 03, 2007). CHI '07. ACM Press, New York, NY, 2543-2548.

[14] Live Search Mobile smartphone application, http://mobile.search.live.com/about/download/default.aspx , © 2007, Microsoft Corp.

[15] Matthews, T., Blais, D., Shick, A., Mankoff, J., Forlizzi, J., Rohrbach, S., Klatzky, S., Evaluating glanceable visuals for multitasking. EECS Department, University of California, Berkeley, *Technical Report No. EECS-2006-173*, 2006.

[16] Pousman, Z. and Stasko, J. 2006. A taxonomy of ambient information systems: four patterns of design. In *Proceedings of the Working Conference on Advanced Visual interfaces* (Venezia, Italy, May 23 - 26, 2006). AVI '06. ACM Press, New York, NY, 67-74.

[17] Robbins, D. C., Cutrell, E., Sarin, R., and Horvitz, E. 2004. ZoneZoom: map navigation for smartphones with recursive view segmentation. In *Proceedings of the Working Conference on Advanced Visual interfaces* (Gallipoli, Italy, May 25 - 28, 2004). AVI '04. ACM Press, New York, NY, 231-234.

[18] Rosenbaum, R. U., and Schumann, H. Grid-based interaction for effective image browsing on mobile devices. *Proc. SPIE Int. Soc*. Opt. Eng. 5684, 2005, 170--180.

[19] Stone, L., Linda Stone's Thoughts on Attention, *Blog entry*.

[20] Van Dantzich, M., Robbins, D., Horvitz, E., and Czerwinski, M., Scope: Providing Awareness of Multiple Notifications at a Glance. *Proceedings of AVI 2002*. pp. 157--166.

[21] White, R. W., Drucker, S. M., Marchionini, G., Hearst, M., and schraefel, m. c. 2007. Exploratory search and HCI: designing and evaluating interfaces to support exploratory search interaction. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems* (San Jose, CA, USA, April 28 - May 03, 2007). CHI '07. ACM Press, New York, NY, 2877-2880.

[22] Wilson, M., Russell, A., schraefel, m. c., and Smith, D. A. 2006. mSpace mobile: a UI gestalt to support on-the-go info-interaction. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems* (Montréal, Québec, Canada, April 22 - 27, 2006). CHI '06. ACM Press, New York, NY, 247-250.

[23] Zumobi, *Zumobi web site*, http://www.zumobi.com.