TapGlance: Designing a Unified Smartphone interface for Personal Information Management

Daniel C. Robbins
Microsoft Research
One Microsoft Way
Redmond, WA 98052
dcr@microsoft.com

ABSTRACT

Finding a photo on a mobile phone should be as easy as finding a phone number. Comparing the date of one calendar appointment to the due date of a task should be equally easy. And seeing where a set of friends are on a map should be as easy as checking the local weather. All of these are information tasks and all of them are migrating to an emerging class of mobile device we call smartphones. Currently, these types of tasks are restricted in their fluidity by separate applications and strong object typing. We propose ways of re-conceptualizing these constructs so that users can fluidly create, edit, manage, and share personal information.

We are presenting TapGlance, a design proposal for how to support common Personal Information Management (PIM) on a smartphone. This paper presents extensions to our previous design work on TapGlance (to be presented at DIS 2008). TapGlance is a reworking of the entire smartphone user experience (UX). In the initial TapGlance work we focused on adapting the interface to the various levels of attention that a user had, presenting information in a feed style, and coupling all of that with a faceted search system. Our current work focuses on how tasks, tagging, and commands can be woven into the TapGlance UX. Our new design centers on methods for creating, organizing, and disseminating information. This information encompasses many different types, from text to photos to people. User interactions are consistent across information types and independent of origin and storage of the information. This TapGlance design proposal is the first step before we engage in prototyping and user evaluation.

General Terms

Design, Human Factors

Author Keywords

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

ACM Classification Keywords

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

Figure 1: Overview of the TapGlance UI: (1) the locked screen shows glanceable overview information, (2) the default set of HomeTiles, (3) zoomed into the Calendar.

1. INTRODUCTION

Traditionally PIM has been done on the desktop where there is a great deal of screen real-estate, a full keyboard, and guaranteed network connectivity. The performance constraints for mobile phone based PIM are much tighter. Because people exist in divided attention situations the underlying system has to be much more judicious in terms of which information is presented. Organizing data is much harder in a mobile situation because the length of working sessions is much shorter and it's very difficult to provide an overview of a large information corpus on a very small screen. We have spent several years developing a series of smartphone based user interfaces – all with the lofty goal of unifying the user experience across multiple applications and contexts. We have very recently made significant changes to the overall TapGlance interface design as an attempt to better address the needs of personal information management.

Several key principles underlie the design of TapGlance. Firstly, our smartphone based system must respect the user's degree of attention. To accomplish this, the information feeds on the smartphone are presented at various levels of abstraction. Secondly, a user should be able to pivot information retrieval around any and all available dimensions: time, location, person being the primary dimensions. This information retrieval must be available across all information that the phone encounters, whether that be local to the device or accessible over the network. Because the context within which the device is used is very fluid, there must be multiple ways to initiate a given task. And lastly, the interface should optimize itself for the common PIM oriented tasks of information creation, editing, retrieval, and sharing.

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 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 [2]. 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 [1] used categories common to a user's own data as a front-end to a desktop search system. mSpace Mobile [10] 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 [3] 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 idea of segmenting the screen and tying different regions to the number keys is based on the work in ZoneZoom [8]. The idea of a zoomable tile set has also made it into the Zumobi mobile application [11]. The current Live Search Mobile application [4] presents hierarchical facets although only in successively arranged lists which do not take advantage of spatial memory.

In terms of glanceable interfaces – displays that can be apprehended with a minimum of attention, Pousman and Stasko [6] 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 [5]. The Scope project [9] 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 gadgets or widget based desktop notification systems. 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

There are several areas which we are not going to address in detail in the TapGlance design proposal. It is beyond the scope of this paper to provide detailed designs for the classic PIM applications such as Email, Calendar, and Tasks. Instead, this paper discusses how the existing TapGlance design can be adapted to support these kinds of applications and in turn, how these applications can be adapted to fit into the TapGlance framework.

Each of the canonical PIM applications bursts with functions. We are not proposing a new set of manipulation and recall features but are instead concerned with how given set of features – ideally hierarchically arranged in levels of abstraction – can be accessed through a faceted search mechanism.

In addition, any real PIM system must work hand-in-hand with a server or cloud-based storage and retrieval system. We are not going to describe the changes needed in those parts of the infrastructure.

Our design goals for the TapGlance are as follows:

- **Text is king**: if in doubt use text as a representation
- **Always show results**: do not let the user create a query that results in zero results.
- **Stable spatial layout**: keep common options and information items in stable locations in the interface.
- **Short lists of items**: when possible, only show enough items to fill the screen. Provide tools for paging through grouped sets of results.
- **Don't rely on short-term memory**: provide a means of easily squirreling away the results of a work session.
- **Choose really good defaults**: the phone has to work out of the box and the PIM features have to make sense.
- **Optimize for a 12-key smartphone**: this is the phone form-factor that **billions** of people in the world are going to have for the foreseeable future.

Beyond that, a detailed discussion of glanceability and unification in the smartphone UX is discussed in the previous TapGlance work [7].

4. THE TAPGLANCE USER INTEFACE

The original TapGlance work consisted of a design proposal for a unified smartphone user interface. This paper describes ways in which we have adapted the TapGlance design to better suit PIM tasks. We refined the home screen to itself have different modes. Instead of just presenting nine tiles, each with a different information source, we now combine some of the tiles to show fewer tiles and give more real-estate to a few of the more important tiles. In this redesign, we have also made a distinction between the locked and unlocked modes for the phone. In the locked mode, the text-entry field is hidden and a top-most status area is increased in size to make the time and date very easy to read. When someone pulls the phone out of their pocket, the most common thing they want to see is the time. In the previous design, time was relegated to the smallest portion of the topmost status bar.

The TapGlance interface consists of a large central pane that contains a combination of nine information readouts and quick access buttons, called the *HomeTiles* [Figure 1(2)]. Above the HomeTiles is a rich text-entry region called the *TopBar*. A thin, standard cell-phone style status readout is at the very top of the display and the very bottom carries labels for the phone soft-keys. As the user shifts between the different major modes, animation is used to give more or less screen real-estate to each of these sections. We choose the nine tiles because this maps to the number keys on the vast majority of mobile phones. For touch enabled devices we anticipate that restricting the display to nine tiles would ensure that each tile is an easy finger target. For devices with full QWERTY keypads, an alternate home screen would probably be appropriate. But, as said previously, our main focus is on 12-key phones as this is the most prevalent world-wide.

4.1 Scenario driven design

The aforementioned TapGlance user interface is motivated by a set of PIM centered scenarios. These scenarios not only help drive the initial design but also act as "test cases" for the validity of the designs. In this section I will briefly list a few scenarios and then, in the space allotted, relate how two of them could be accomplished with the proposed TapGlance user interface design. As mentioned elsewhere, there intentionally are many ways to accomplish these same tasks; some are better suited for quick interactions and some are better suited to sit-down-and-concentrate on the phone sessions.

4.1.1 Scenario #1

Scott is at the store and remembers that he will need to call the baby-sitter later on in the evening.

Please refer to Figure 7 for a visual guide to this process. After unlocking the phone, Scott would just start typing "call babysitter" into the TopBar, using any of the available text input methods. If Scott single-taps the left soft-key, he is presented with a standard "Save" menu of choices. By double-tapping Scott can take a short-cut past the pop-up menu and select the most common menu choice of "Save to Scratchpad." Scott could leave it that and if he hadn't handled too many other items on the phone in the meantime, the text-note might still be near the top of the scratchpad.

But Scott might want to add more metadata to this item, such as a due date. Before shifting focus away from the TopBar, Scott would open the main menu, select the "Tag" option, then "Date", and then "Tonight." Scott does not have to specify that this is the due-date for the item. What Scott is doing is creating a relationship between the calendar and the text note. Heuristics built into TapGlance would by default create a reminder when an item is given time information. If Scott was in a very exacting mood, he could alternatively navigated from the main menu into the "Property" sub-facet, found the specific "Due date" property, and then added then manually added an exact value. The takeaway from this is that a user should be able to very quickly create information and, when needed, add metadata at various levels of specificity.

4.1.2 Scenario #2

Mike is in a meeting and he quickly wants to gather a list of all emails related to Project Beta that include Brad.

Mike moves the focus from the TopBar to the HomeTiles by tapping the right soft-key once. He then opens the Inbox HomeTile and it zooms up to replace the home set of HomeTiles. Mike immediately realizes that the emails he needs to gather are not displayed on the screen. When Mike chooses "Find" from the main menu, the FacetPane (faceted search interface) slides up to cover the bottom third of the Inbox pane and the focus is shifted to the FacetPane. Mike navigates into the Tag facet and chooses the "Recent" sub-facet. This in turn gives Mike a list of recently used tags, among which is the "Project Beta" tag. Mike then taps the left soft-key to apply the "Project Beta" tag as a filter. A visual token ("breadcrumb") appears in the TopBar to reinforce the current filter. Mike taps the right soft-key which causes the FacetPane to navigate back to its root display of the top level facets. Mike now navigates into the "People" facet then applies the "Co-workers" filter. Tapping the right soft-key brings the focus back to the TopBar. Mike starts to type "Brad" and as he types, the set of emails shown in the Inbox is filtered to just show those emails which have are related to both Brad and Project Beta. At this point Mike could read through the emails or he could save this query via selecting "Save" from the main menu and then "Query" from the "Save" sub-menu. That query would then be available for recall from the "Favorites" facet.

4.1.3 Additional Scenarios

Following are several other PIM related scenarios that the TapGlance design can stretch to accommodate. These are useful in that they point to the broad array of situations in which PIM occurs in the mobile world.

Doug is at a construction work-site and he needs to quickly get a list of nearby supply stores that are open late in the day.

Pat takes pictures of a bunch of products at a supply showroom and quickly tags those items which best meet her criteria.

Tim is getting out of a movie downtown and he wants to get a list of highly rated restaurants that are near the movie theatre.

Jack wants to show a friend digital images of the two of them.

5. LEVELS OF ABSTRACTION

Most smartphone applications arrogantly assume that at all times they deserve the user's full attention. This contradicts how people use phones in the real-world – where the phone is just one of many stimuli vying for our attention. Central to the TapGlance design proposal is an interface that gracefully adjusts its presentation to match the amount of attention that a user wants to devote to the phone. The lightest-weight mode we call *glance-mode* and in this mode only the most important, non-interactive information is shown in a very readable manner. In *inspection-mode*, the user sees information from a wider array of sources. In the *peek-mode*, a user can temporarily get more details about an item from a particular information source. And in *interaction-mode*, a user can fully engage with a tailored application or document.

In glance-mode – when the user just quickly pulls the phone out of their pocket – the top half of the display is taken up by an enlarged status display [Figure 1(1)]. Below that, the visible HomeTiles are devoted to updates from important contacts, information about the user's next appointment, and a readout of any currently playing media (such as music).

When the user unlocks the phone and thus enters the inspection-mode, the upper status readout shrinks to reveal both the TopBar and all of the HomeTiles [Figure 1(2)]. In addition to the previously mentioned information sources, we anticipate that the default set of nine tiles will include information and access to the user's inbox, favorite applications, data-feeds (such as weather), and additional people oriented notifications. For the most part, each HomeTile's contents are populated by standing queries. Initially the TopBar has keypad focus so that phone numbers can be immediately typed. In inspection-mode, the user cycles the focus between the TopBar and the HomeTiles by repeatedly tapping the right soft-key.

The exact choice for default home tiles is not the focus of this paper and is most certainly a matter of great debate. We hope that by the time we roll out our first prototype, we will have picked a reasonable set. In picking these top-level tiles we tried to achieve a balance between dynamically updated information sources (such as "Inbox" and "Weather") and typical mobile computing tasks such as "Search" and "Photo-taking." We wanted the most commonly used tasks to be as few clicks away as possible. While

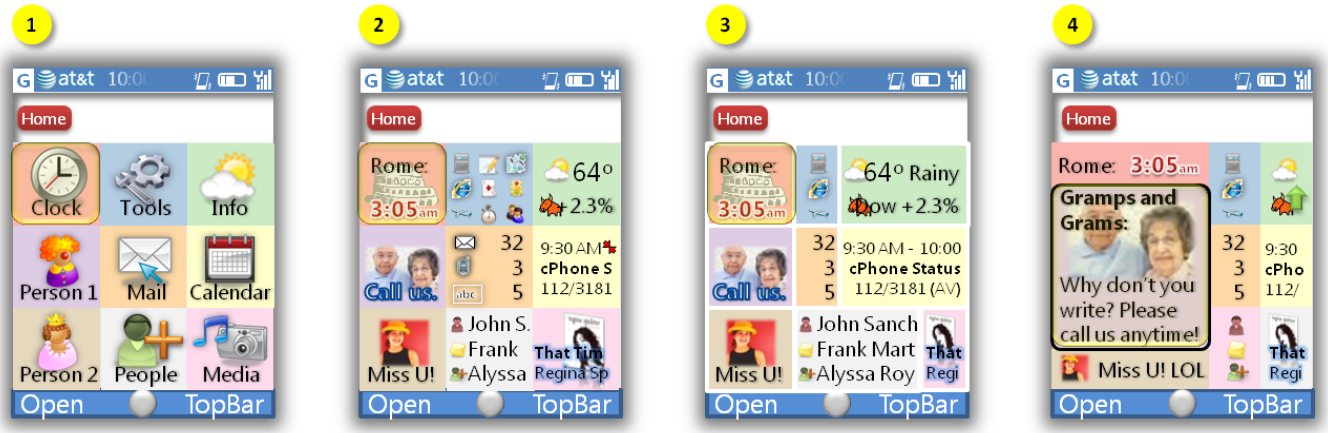


Figure 2: HomeTiles in different configurations: (1) as displayed when the user first runs TapGlance, (2) after repeated use each tile is populated with specific information, (3) each tile is given different amounts of space, and (4) the "Person 1" tile is shown enlarged because it has urgent information in it.

this may seem inconsistent, we really believe that usefulness – having the most common information and tasks readily available – outweighed pure uniformity. Choosing the default tiles and how they each render themselves is critical to the success of this project. Lab-testing will help with our first pass, but we are even more excited to do in-the-field deployments once we have a prototype system up and running.

If the user wants to quickly peek at more detail about an item in one of the HomeTiles the user presses-and-holds the number key that corresponds to the grid location of the desired HomeTile. While holding that number key, the HomeTile zooms to nearly fill the display and shows more information. For example, the small glanceable version of the calendar tile only has room to show the time, and portions of the name and location of the next meeting. But by using peek-mode, the user would see the duration, full name of the appointment, and a list of attendees. When the user releases the number key, the peek view shrinks back down so that all nine tiles are visible.

If the user wants to give the phone more of their attention, they enter inspection-mode. To get an overview of their whole day or schedule a new appointment, the user would tap the number key that corresponds to the calendar tile [Figure 1(3)]. The calendar tile would then zoom to fill the entire screen and the extra screen real-estate would be used to show a broader view of the user's day. We estimate that a zoomed-in tile, on a typical smartphone, could display two lines of text about four appointments. We anticipate that a user would be able to configure the overall font size and contrast ratios to suit their own perceptual abilities.

At first glance the TapGlance screen may appear to very dense. While the fonts are small, they are in-fact the standard text size from a typical smartphone with a 240x320 resolution display. It is almost certain that our font sizes, icon-density, and color choices will be refined once we create a prototype and proceed with user testing. When the user first uses a TapGlance enabled device, each home tile would display a descriptive label, such as "Mail" or "Calendar" [Figure 2(1)]. These labels coupled with the stable location of the default tiles will help users learn the default set. As a user visits these tiles and the tiles become populated with user-specific information, the descriptive titles would be deprecated to make more room for dynamic information [Figure 2(2)]. We can think of this as "progressive densification." Even though we use the nine-grid layout throughout the user-interface, we can differentially adjust the sizes of each tile so that more space is given to more important tiles [Figure 2(3,4)]. Not only does this draw attention to important information, it may also aid in distinguishing different states in the UI. To be sure, a balance needs to be struck between stable didactic information (labels) and dynamic information (updates). Different existing mobile interfaces exhibit different takes on this balance. Some smartphones, such as the Windows Mobile smartphone, show arbitrarily long lists of dynamic information including the details for a user's next appointment. Other smartphones, such as the iPhone and Blackberry, typically show a set of static labeled icons, each representing a different information source or application.

5.1 Intelligent Visualization of Feeds

The richness of each HomeTile lies in the underlying query coupled with an adaptable view style. A view style is a combination of sorting, filtering, grouping, and layout styles. For example, when the media tile is shown in its smallest "glance" mode, only one item that matches the query (the currently playing song) is displayed. This is a very tight filter coupled with a summary layout style. Much of the power of the tiles comes from intelligence built into the various layouts. When a song is currently playing, the most important piece of information is the name of the song and the artist. If there isn't a currently playing song, the most recently "touched" media items are displayed. If those items are photos, then two photos can be displayed in the double-wide media tile on the default home screen.

When the user zooms into the media tile, the default view style shows nine cells to match the hardware number keys on the phone [Figure 6]. Some of these cells are populated with individual items (such as a thumbnail for a photo) but other cells may reference a grouping of items, such as multiple photos associated with a particular event. In essence, in the default summary layout style, each cell progressively shows broader and broader collections of information, usually as cast onto the time axis. In the case of media, the first row of cells shows the three most recently taken photos. Each cell in the second row represents a collection of images that are closely related on the time axis, e.g.

they were taken together. This logarithmic lens also shows up in other dimensions. The location Filter facet (facets are discussed later) is made up of sub-facets that are labeled with successively broader durations of distance: one block, neighborhood, city, state, country, and etc. Likewise for the time facet: today, last week, last month, last year, and before last year.

5.2 Information Creation

A primary aspect of personal information management is that user's also need to both create new atomic items and to create additional information in existing items. Creating information consists of activities such as taking a photo, jotting down a note, and associating a phone number with a person. In a very intentional way, each of these tasks can be accomplished through a variety of workflows. This is a thread that runs throughout TapGlance – that depending on context, a user will conceive of multiple ways to accomplish the same thing. These multiple methods can be seen in Figure 4. Sometimes a user may want to access the phone's camera from a photo-browsing context. At other times the user might want to start with an applications list and then find the camera. And at another time, a user might indicate to the system that they want to create something and then be offered a set of choices such as "create a photo", "create a video", or just "create a note." Part of the reason for having this plethora of entry-points is that it shouldn't take too many navigation steps to get to a specific creation task, no matter what activity the user is currently engaged in.

In the TapGlance design, jotting down a note is an even easier task. In current phones, a user has to navigate to a note taking application, create a new note, then save the note. In TapGlance a user unlocks the phone and just starts typing. The typing, whether it is multi-tap or T9 text entry gets simultaneously interpreted in multiple ways and the possible interpretations are reflected back to the user [Figure 8]. The typed digits are shown in various way: the number itself (so a phone call can be made), matches against numbers in the phone's address book, the T9 or multi-tap text is matched against all items in the phone's database, and the text is also left as free-text. At any point the user can choose to dial the direct phone number by hitting the hardware call button. If one of the other matches is more desirable, then the user moves the focus to the results list and scrolls down to that match entry and hits the left soft button to initiate either a "save" or an "open" (depending on the item type).

5.3 Navigation and Menus

From the scenario explanation and the previous description of how to save a note, it may seem like there is a great deal of navigation in the TapGlance design. But the number of button presses is not a true measure of navigational complexity. Navigation complexity is gated by how much cognitive processing a user has to do when traversing from one context to another. If the steps to get from one "place" to another are predictable, then the cognitive complexity is lowered. If there is a menu that a user uses often and the most commonly used menu choice is already primed for selection, then the button press to activate the desired menu choice can happen very easily. This is akin to reaching out one's hand to where you think a light switch should be (right next to a doorway) and finding the light switch there – we don't have to think about it. The TapGlance main menu system uses a spatially arranged numerically accessed set of nine choices – nine choices consistently laid out in a grid. Each time the user visits a particular menu, the choices are always in the same place. As much as possible, in the TapGlance menu structure, we place the most commonly used sub-menu choice directly under the parent menu position, i.e. if the user pressed the 6 key from the main menu to select the "Send" option, the "Scratchpad" option will also be located in the number 6 slot on the sub-menu. A user who is fairly familiar with the menus will be able to quickly double-tap on the 6 key to send the selected item via email. While this is several key presses, because it does not require moving the finger to different keys, the physical effort is decreased. In some sense the double-tap becomes like double-clicking on a mouse button. Another short-cut past menu navigation is available if the user presses-and-holds the Action button. In this case, the menu doesn't even appear and the default menu-choice is activated.

5.4 The Scratchpad

There are times when a user is not interested in a set of items that is the result of a query. The user may have a set of songs they want to listen which don't really share any distinguishable meta-data. Likewise, a user may want to collect a bunch of emails together that wouldn't otherwise be returned from a query. There may also be times when a user wants to compare items from across multiple queries, e.g. "let me flip back and forth between looking at my calendar for a particular day and the content of a particular email."

In a desktop PIM environment, a user would typically open each desired item in a new window. Then the user carefully arranges the separate windows side by side on a large display. In a typical smartphone based PIM system, this is really not possible. The user has to navigate through the various PIM apps to find a particular piece of information. If they then want to make a comparison across types (or even across time as when comparing two different days) the user has to initiate a great deal of navigation to traverse between the different items.

In the TapGlance design proposal we borrow from desktop photo management and web based mapping applications. As a user is browsing their photo collection, they add individual photos to the scratchpad by several means: dragging the photo into a separate part of the UI, clicking on a pushpin on the photo, or merely shift-selecting several items. Online mapping applications allow, with one click, users to add a point-of-interest to an online collection. As discussed in the scenario section, a scratchpad is primary to the TapGlance interface.

At first glance, it might appear that the "flagging" functionality in many email applications effectively gives a scratchpad behavior. In actuality this is too heavy weight and causes some other problems. The scratchpad is special in that it is specific to a user session. It is generally only useful during one uninterrupted interaction sequence. Relying on a flagging system is problematic because the flags are persistent. If the user flags a few items, changes their context, say from email to calendar, then wishes to see their list of flagged items, they will see every flagged item, not just items flagged from the current interaction session. Likewise, web browsers let users add the current page to a list of "Favorites" or "Bookmarks." Just as with the flag, this mixes together short term with long term lists of items.

In TapGlance we propose that a user would explicitly add items to a scratchpad via a simple menu operation. The user can even choose to have the scratchpad be one of the nine HomeTiles. In this case, each time the user inspects the phone display, they would be greeted with a list of the most recent scratchpad items.

5.5 Faceted Search with a Loose Taxonomy

In most PIM systems items are related via their metadata. The main purpose in maintaining metadata is to aid in information retrieval. If the user knows the exact name or identifier for an item in the database, that's great. But that is not often the case and it is often ambiguous as to what the name of an item is. In an email message, is the name the subject, the name of the sender, the first line of the body of the message, or a GUID (Globally Unique Identifier) assigned arbitrarily by the underlying system? We can't know ahead of time how a user will remember an item or how specific their memory of the item is. Because of this, any PIM system needs to support a rich notion of metadata. A user might remember who sent a photo but not when. Another user might remember that an email message contained a link to something about new display technologies but not the actual text of the message. All of these ways in which a user remembers an item needs to be supported as a means of recall. Thus, metadata, along with full content indexing, are the keys to information retrieval.

5.5.1 Existing Metadata systems

In general, most PIM systems present one of three kinds of organizational structures for relating metadata in their database. In a *hierarchical* system, every item in the database exists within a strict pre-authored tree, typically based on the file-type property. For example, all email items are in one branch of the tree, all calendar items in another, and tasks segregated as well. In a *faceted* system, items are related by how their metadata fits into multiple overlapping trees. In a faceted system, all the property types are composed into a hierarchical tree but items themselves are not placed in a hierarchy. A user retrieves items by browsing the property tree and choosing property values from within the tree. These choices act as filters across the entire item database. In a *tagsonomy* (or Tag cloud), there is no relationship between the property values – there is only a flat "bag of tags." Items are related by sharing tags. This is used in many online photo-sharing and collaboration sites. Users add tags to photos and then can later retrieve photos that share a particular set of tags.

Each of these systems has its own pros and cons. The strict hierarchy has a degree of predictability but its lack of flexibility hinders retrieval when a user may not know enough information about an item. A faceted system allows for many item types and supports fluid browsing. The problem is that in standard faceted systems an attribute can only live in one place in the faceted hierarchy. This means that a user's understanding of how information is organized has to match how the initial author of the taxonomy conceived of the corpus. This is not a big deal when using faceted search on the desktop as a there is sufficient screen real estate to simultaneously show multiple sub-facets at the same time. In effect, users can simultaneously pear deep and broad into the metadata tree without having the engage in much navigation. On a small device, though, there is not the space to show more than one facet a time. The user does not have the opportunity to easily gain an overview of the taxonomy. A tag cloud does not require a user to learn and navigate a taxonometric structure but at the same time, the only way to access the tag cloud is via search or ponderous browsing. The organization of tags (when a hierarchy is present) is usually based on statistical methods. Most tag clouds, though, only apply to user generated free-text categorizations. They do not encompass arbitrary properties that exist on a collection of items – properties such as size, date, and author.

5.5.2 Loose faceted hierarchies of metadata in TapGlance

TapGlance uses a hybrid approach where all properties and property values are stored in one connected graph. "October 21st, 2005" is a value that is related to the "Date" property. For a particular item in the database, such as an email, "October 21st, 2005" is related to that item via the "sent" property. In some sense, the value "October 21st, 2005" is an item itself in the database. Eventually, in this kind of graph structure, every item is in some way connected to every other item via relationships between item metadata.

This is a very flexible system but browsing an arbitrary graph is hard on a desktop system and next to impossible on a smartphone. TapGlance uses several strategies to facilitate metadata navigation. In the TapGlance proposal, the user is presented with a set of pre-authored attribute hierarchies – much like a standard faceted search system. The departure is that the hierarchy is very loose. Again, because we believe that different users (and the same user in different contexts) conceive of a metadata structure in different ways, attributes are distributed in many places in the tree structure. We anticipate that the initial arrangement of metadata in the tree would be generated by a combination of statistical methods and hand-tuning by the application developer (derived from typical PIM tasks). For instance, the property "Creation Date" would live both as a child of the top-level "Property" facet and as a child of the "Date" facet. The sub-facet of "friends" would live both under the "People" facet and under the "Property/Author" facet. A first pass at the TapGlance facet hierarchy can be seen in Figure 3. While a typical user would never edit this loose hierarchy but we have explored ways of enabling customization.

The construction of this tree structure on top of an arbitrary graph is at the heart of TapGlance. Since there are an arbitrary number of properties, hard choices have to be made as to which properties are most salient in the interface. The tree that TapGlance presents is geared toward typical PIM tasks. Status oriented properties such as "Not/Done", "Un/read", and "For Follow-up" are not buried in a property tree, instead "Status" gets its own top-level facet. Likewise, the default people sub-facets include designations that are useful to PIM tasks such as "Co-workers", "Team-mates", and "Family." As we mention in section 7, we look forward to doing user test where we can refine this taxonomy. A proposal for how this might look in TapGlance can be seen in Figure 5.

While there is an initial set of sub-facets that are presented to the user, the user can edit the tree itself. To do this the user selects a "configure" option from the main menu and then proceeds to navigate (or search) for properties that are most important to them. The user then assigns these properties to slots in the hierarchy. A user might decide that for their particular style of working it's much more important to have ready access to the "Bit-rate" property for items rather than the "Version" property. To make this change the user would navigate into the "Properties" facet. After seeing that there wasn't a specific sub-facet labeled "bit-rate," they could select the "More..." sub-facet to get a listing of all the available properties. Alternatively, the user could select "Search" from the main menu when the focus was in the facet pane. Upon doing that, the focus would be temporarily shifted to the TopBar where any text entered would be used to search against the names of properties in the Facet hierarchy. After choosing "Configure Facets" from the "Settings" option on the main-menu, the user would then choose a slot for the new

property. And this is why interactive prototypes are often better than textual descriptions.

5.5.3 Using a faceted hierarchy to access commands

Typical faceted search systems are used to grant access to properties for objects. In the TapGlance proposal, we also use the facets to access commands. A top-level facet labeled "Tasks" gives hierarchical access to any commands that would have bearing on the current set of items in the result pane. If there currently is no query, then a default set of PIM related tasks is shown. These commands are arranged into very high-level, PIM task oriented groupings and are generally independent of item type. The default tasks include (among others) Create, Edit, Share, and Remind. The generality of our hierarchy is illustrated by the common task of a user wanting to print a document from the smartphone to a nearby networked printer. The user could choose "Tasks/Create/Printer Version" or "Share/With Printer." Both are valid ways of conceptualizing the task of printing and the flexible nature of the proposed TapGlance facet hierarchy would allows for this.

5.5.4 Adaptive Refinement of Item Definition

The flexible metadata system proposed in TapGlance allows other interesting functionality as well. It tends to blur the distinctions between different item types. In typical PIM systems items are exposed to the user as belonging to particular types, e.g. only email items can have a "from" attribute. In the proposed TapGlance system, most properties can be added to any item. If a photo was sent via an email or other transport system, it gains a "from" attribute. Sometimes these are direct relationships as when properties are directly written to an item: a text note becomes an appointment when it is assigned a start and end date. In other cases, the properties might not be directly assigned to an item but because of a relationship between two items, that property can be used to find both items. If the aforementioned appointment also has a photo linked to it, the photo gains a degree of relatedness to the time axis. The photo does not itself have a start and end date but because it is related to the appointment, the photo could be plotted on a time axis. Again, this serves to illustrate that in the proposed TapGlance system, every item is related to every other item via their properties.

A user can also take advantage of the loose hierarchical structure of our metadata to iteratively refine the tags on an item. For example, when a user first encounters a set of emails they might want to first just tag them all as pertaining to work. Later on, the user can refine the work tag for individual email items by adding particular "project" tags to specific items. And this works in both directions: an item may have very specific, detailed tags on it but the user may not remember the exact specific tags. In a traditional tag-cloud system, that would be the end of the story. If the user doesn't remember the specific tag, there is no way to recall that set of items. In the TapGlance system, though, all tags, no matter how specific, are cast into multiple places in a loose hierarchical structure. An appointment on the user's calendar may have a very specific tag about a feature review. The user might not remember the name of that feature so in a traditional PIM system, it would be hard to find that item. In the proposed TapGlance system, though, users have the option of specifying how that new tag relates to existing tags. In this case, the particular feature tag could be related to an overall project. At recall time, if the user searched for the name of the project, the appointment for the detailed feature review would be returned as a faceted search result.

6. OBSERVATIONS

There is an emerging breed of new PIM applications that also try to blur the distinctions between strict information types. The new Chandler system is a fair representative. It is useful to make a comparison between our proposed TapGlance user interface and the just recently released first version of the Chandler PIM system. Even though Chandler is not currently a mobile optimized application, we choose it for comparison because of similarities in intent between our systems. Both of our systems believe that the traditional divisions between object types need to be jettisoned. Both systems believe that users should be able to iteratively add more detail to an item's definition, and both believe that there are archetypal workflows that a user engages in during their normal workday to keep on top of their projects, commitments, and communications. Both systems firmly believe in providing a layer of abstraction over the vast array of attributes that a heterogeneous system by necessity has to support.

The differences, though, lie more in how actions are initiated in the two systems. Let us consider the example of emailing out a text note. In Chandler the user first has to add the relevant metadata before the "send" buttons become available. The user has to first add a recipient and then the "send" button becomes active. In our TapGlance proposal, the action is the key itself. The user selects an information item, such as a text note and then from the main menu chooses an action category such as send. Once that has happened, the system asks the user for a recipient's address and or name. This is possible in the TapGlance system because we provide a hierarchical composition of PIM centered actions, actions that are defined from the more general to the more specific.

Another difference is that Chandler gives primacy to just a few of the common facets: Type, Action, Tag, and When. Each of these facets gets prime real-estate in the Chandler desktop application. The other canonical facets, who and where, seem to be relegated to second-class citizen status. In the TapGlance design, all of these canonical facets have equal prominence and dedicated view types for visualizing information from each of these dimensions.

7. CONCLUSIONS AND FUTURE WORK

As this has only been a proposal, in essence a design on paper, our next obvious steps are to prototype these designs and start user testing. We will be very curious to see if short training on the general metaphor of TapGlance will enable positive transfer: will users who learn how to use a few applications within TapGlance have an easy time of learning new TapGlance enabled applications?

The designs from TapGlance serve as one of the first steps for the cPhone mobile computing project within Microsoft Research. This project aims to define and prototype a future class of mobile computing and communication device. As part of the cPhone project we are considering how to best use various sensor data to inform the user interface. The information displayed in the home screen TapGlance tiles could be optimized based on GPS, audio, and video sensors and as yet unexplored sensors. This sensor data could also be used to geo-code user actions and semi-suggest appropriate meta-data.

Our TapGlance design proposes a way in which users can combine and visualize data from across multiple silos. A hierarchical faceted search interface can be used throughout the TapGlance experience to filter any of the structured information available from the smartphone. 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. 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 personal information management tasks, at various levels of detail, via a common interface. TapGlance combines segmented zooming navigation and ubiquitous faceted search across common information source feeds.

8. ACKNOWLEDGMENTS

This design rests on very fruitful collaborations with Bongshin Lee and Roland Fernandez, and Ed Cutrell. Lastly, thanks family for letting me focus on this paper while life continued around me, and greeting me with open arms when I emerged, even after many deadline extensions.

9. REFERENCES

[1] 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.

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

[3] 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.

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

[5] 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.

[6] 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.

[7] Robbins, D.C. 2007. TapGlance: Designing a Unified Smartphone Interface. To be published in *Proceedings of Designing Interactive Systems* (Cape Town, South Africa, February 25th – 27th, 2008). DIS 2008. ACM Press, New York, NY.

[8] 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.

[9] 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.

[10] 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.

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

#	Top	Children								
1	People	Me	Family	Friends	Co-Workers	Acquaintances	Services	Team-mates	Sender	Receiver
2	Location	Near Me	Neighborhood	City	State	Country	Near Work	Near Home	Room	Favorites
3	Type	Document	Web	Email	Photos	Videos	Music	Map	Appoint.	Person
4	Tag	Personal	Work	Private	Favorite	Avoid	Custom 1	Custom 2	Recent...	Category ...*
5	Date	Today	Yesterday	Week	Month	Year	Decade	Century	Recent	Frequent
6	Status	Done	Not Done	Read	Unread	Needs Review	Needs Reply	Sent	Opened	Modified
7	Favorites									
8	Property	Size	Priority	Status	Creator	Price	Name	Rating	Source	More...
9	Tasks	Create	Edit	Send	Buy/Sell	Remind	Listen or Watch	Play	Eat	Travel

Figure 3: Initial TapGlance Facet Hierarchy (with loose, overlapping membership)

Favorites Tile

Lists combo of recent, frequent, and pinned items

Collapsed facets for changing algorithm for results

Choose set of pictures → from photos, choose camera app

Apps Tile

Tiled view shows top apps (plus "More...")

Select camera app if in top 9, else, select "More..."

Photo Tile

Recent photos get some tiles

Camera app gets 1 tile

Video app gets 1 tile

Media Tile

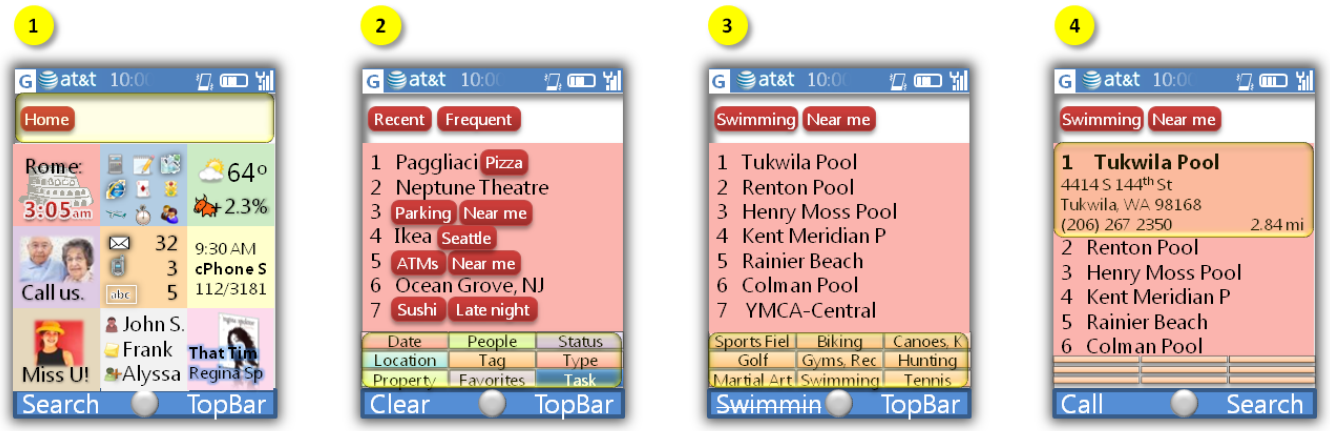
Single Tiled view

"Now playing" music gets main tiles

Recently created/modified/received media gets another tile

Camera gets a tile

Figure 4: Multiple entry points for common application activation (Camera)



From TopBar click the left soft-key ("Search") to open the Search Facets

Facets appear

Tiles zoom into search application

Default query = Recent + Frequent searches

Hit "Clear" to start over

Navigate to "Tags/Categories/Travel, Sports, & Rec"

"Recreation/Swimming"

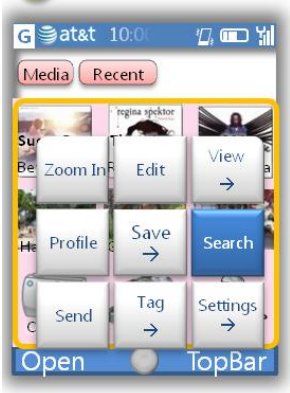
Move focus to results by tapping right soft-key twice

Selected results item shows more meta-data

Facets shrink down

Figure 5: Loose Hierarchical Faceted Web Search

1



Choose “Search” from main menu

2

Each row in results pane **slides** up enough to accomadate Filter Pane

Bottom edge of each item is cropped

Items are **not** scaled down

3

Select "**Type**"

Background color changes to generic

Apps are filtered out

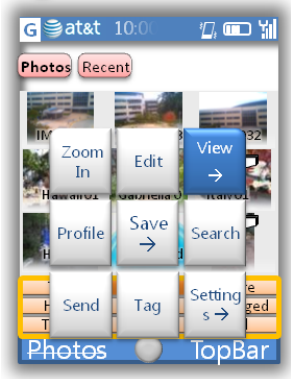
4

Select **Photos**

Apply photos by hitting left soft button

Only photos are shown (videos and music are removed)

5



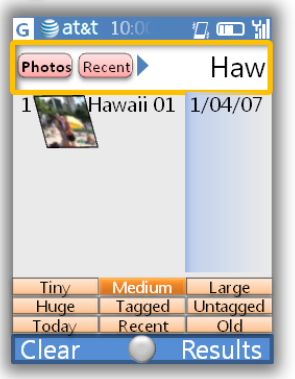
From main menu, change View Style from "Tile Summary" to "Details"

6

Items are listed vertically with meta data in a tabular view

Thumbnails are stacked in psuedo 3D

7



Move focus to TopBar

Start typing text string

sub-focus in TopBar shift to text entry region

Photo set is filtered by search string

Figure 6: Media Filtering

Figure 7: Scenario 1: Calling the babysitter

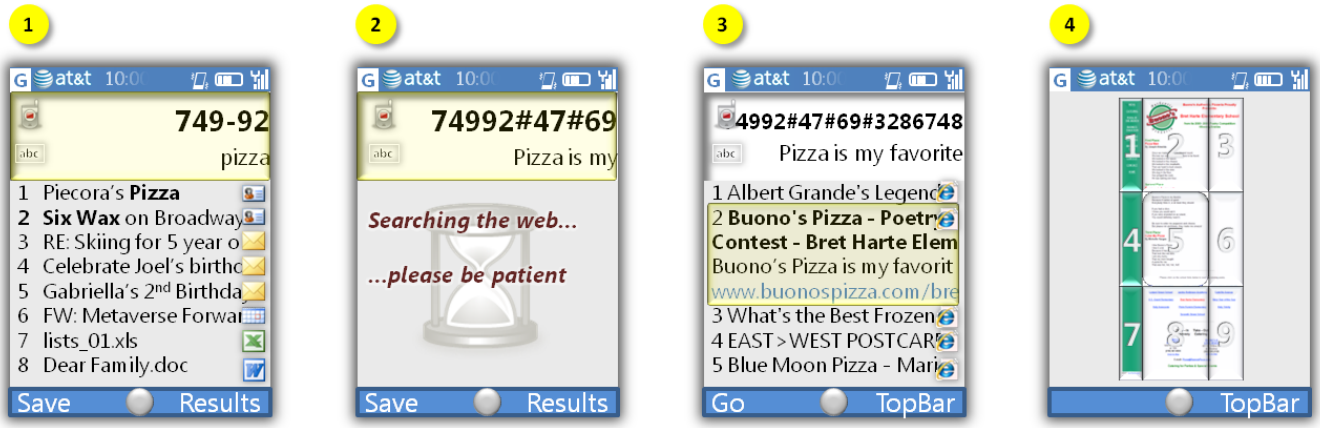


Figure 8: Typing Interpretation