AnyPhone: Mobile Applications for Everyone

Eric Paulos
Intel Research
2150 Shattuck Avenue #1300
Berkeley, CA 94704-1347 USA
eric@paulos.net

Parul Vora
Design School
Human Computer Interaction
Stanford University, CA USA
parutron@stanford.edu

August Joki
Center for Embedded
Networked Sensing
University of California, Los
Angeles, USA
august@cs.ucla.edu

Anthony Burke
Masters of Digital Architecture
University of Technology
Sydney, Australia
anthony.burke@uts.edu.au

Abstract
The mobile phone is one of the most commonly carried pieces of personal, readily accessible digital technologies. Beyond just voice calls, they function as digital cameras, PDAs, internet consoles, and email and instant messaging clients. The demand for improved operating systems and programming languages has given rise to a wide range of hardware and programming APIs. However, the designers of these mobile phone applications are continuously challenged with two inescapable aggravations: (1) how will users locate and download the application to their mobile phone and (2) will the application be compatible with their phone's hardware? We undertook the challenge to discover the design space of mobile phone applications that required no downloading or installation procedure and would operate on any mobile phone regardless of the phone's network, carrier, operating system, age, or hardware. We developed and deployed two such applications – Tree-Map Arrival Information and Group Voting.

Keywords
Mobile Phones, Group Voting, Tree-Map, DTMF

Introduction
Symbian? J2ME? Windows Mobile? PalmSource? BREW? FlashLite? Python? These are just the start of a long list of design questions one undertakes at the onset of the development of a mobile phone application. The decision of the programming language and underlying

operating system inevitably restrict the platform compatibility of the resulting application. Will it run on a Nokia? What about a BlackBerry? Will users need the latest Windows Mobile phone? The competitive marketplace for handset manufactures has driven this competition around hardware and programming APIs. We are in the early stages of the evolution of mobile phones and expect this divergent competitive trend to continue for some time until we reach a convergence. For now the compatibility issue is a real challenge for developers. So much so that an entire cottage industry has emerged with products and services to address these issues [1, 2].

Designing for any phone means allowed for use of the application by a wide range of phones types and generations

Design Challenge

The value of these new hardware features and software advances is undeniable as they enable new capabilities, improved performance, and generate the novel mobile experiences we crave. But is there a design space of mobile phone interactive experiences that will operate on any phone without any download, setup, or installation? Are any such experinces compelling? Useful? Realtime and interactive?

While we often approach the problem of designing mobile experiences by including the most advanced mobile phone technologies (bluetooth, GPS location, touchscreens, video confernecing, audio streaming, etc), how could exploring this challange from the other side (low-tech and universally usable) provide insights into the overall design territory of interactive mobile experiences? To be clear, our motivation was not to move away from the latest mobile technologies but to temporarily constrain our design thinking to uncover a less often explored design space and to more critically focus our designs within a set of very ridged constraints. While there is likely little economic incentive, companies are always developing campaigns and usage models to encourage us to purchase the latest technologies not new uses for our existing hardware, such heavily constrined design methodologies are often extremely valuable in deconstructing a complex design space where there are many choices. We were inspired by the work of others who explicitly explored the alternate side of design themes such as Design Noir [3] and Exurban Noir [4].

We set some ground rules. First our solution would need to be usable on phones without bluetooth, without GPRS, without color screens, without a screen at all in fact as it needed to operate on literaly every mobile phone – including those few still in use from the early 1990's. Secondly, the experience needed to be a shared social experience, interactive, and real-time. This elliminated several existing technologies such as voice messaging since it is not a socially shared experience. It also removed SMS messaging as a tool since such systems (1) require an SMS plan which is not universal and (2) are not real-time due to network latencies of often several minutes.

The challenge seemed extreme, perhaps insurmountable, given the current state of the mobile phone software market. However, the rewards of easy, instantaneously usable interactive mobile experiences by anyone[1] with any mobile phone was tantilizing.

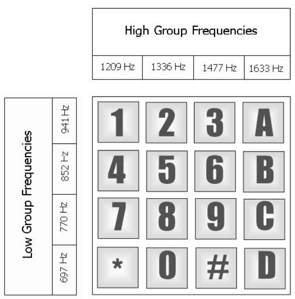
Very soon we realized that the only commonalities across these hardware systems were that they all

[1] Clearly, a caveat to be made clear that we are referring not to anyone but "anyone with any mobile phone".

produced a dialtone, could dial a number, and generate voice and DTMF[2] audio signals.

DTMF as Interface

Literally a technology from the 1950's, DTMF referrers to the unique audio tones generated by each key on the phone keypad (0-9 and * and #). The system is crude but well tested and certainly functional across all mobile phones as they need to generate these same tones to signal and interface to the land based telephone network. Using this mechanism would require no software installation at all or reconfiguration of the phone hardware as dictated by our initial constraints.



DTMF encoding schema

Public Display as Output

The visual output mechanisms across mobile phones is another source of diverse APIs, screen resolutions, *etc*. But with the input coming from the phone could the system interface directly to a shared public display? Such a configuration would indeed generate a shared, interactive experience as required by our initial design constraints. Our design choice was to use a PC with a projected image as the output mechanism for the mobile application. While it limited the accessibility of our application by requiring users to be co-located near such screens or the use of shared screens across mobile locations, it still satisfied our design constraints. The growing deployment of "urban screens" [5] also makes the feasibility of such interactive experiences a very reasonable solution. In fact others have explored various I/O mechanisms of interfacing to large public displays with mobile devices, most notably the work of Paek *et al*. [6]. They describe an architecture for including a wide range of interfaces such as email, instant messaging, SMS text messaging, touch-tone dialing, voice, keyboard, mouse/joystick, camera-based input, *etc*. Their work details only the first two of these interfaces. Our work approaches the problem not in terms of designing for the widest range of I/O but designing for the richest experiences accessible across the least-common-denominator of mobile devices.

[2] Dual-tone multi-frequency (DTMF) signaling is used for telephone signaling over the line in the voice-frequency band to the call switching center. The version of DTMF used for telephone tone dialing is known by the trademarked term Touch-Tone, and is standardized across all carriers.

Asterisk architecture

Asterisk [7] is a complete, open-source PBX[3] in software. It runs on Linux, BSD and MacOSX and provides all of the features you would expect from a PBX and more. Asterisk does voice over IP in many protocols, handles DTMF, and can interoperate with almost all standards-based telephony equipment using relatively inexpensive hardware. In our case we configured a PC as an Asterisk server. Phone calls from a public number were routed to our Asterisk server. Using a python toolkit design to interface with Asterisk, each incoming call was parsed and logged. For each call we were able to get the caller ID, play audio recordings, and listen for the touch tone DTMF signals from the caller's phone. This basic Asterisk configuration was applied to two different applications we developed, deployed, and tested.

[3] A Private Branch eXchange (also called PBX, Private Business eXchange or PABX for Private Automatic Branch eXchange) is a telephone exchange that is owned by a private business, as opposed to one owned by a common carrier or by a telephone company.

Application 1: Tree-Map Arrival Info

At an upcoming public event we were interested in creating a dynamic visualization of the people present in the room and some sense of their distribution across a controversial issue. In our example, individuals were required to register for the event in advance. We used this opportunity to request each person's mobile phone that they would be brining to the event. Each registered participant was also asked to answer a question. In this case the question was somewhat abstract but designed to initiate a categorization and discussion around urban computing technologies. The question was, "What texture is your interactive city?" with the choices being fuzzy, gritty, slippery, and prickly.

Upon arrival at the event, individuals were greeted by a large projection offset to the side of the main presenter's screen. The screen had a phone number shown running down the right hand side. Each person was asked to dial the phone number upon arrival.

Asterisk Configuration

Each call was handled by our Asterisk server which parsed the call. For calls whose caller ID matched phone numbers in the database of expected participants, the person's name and city texture was extracted. If the caller had not previously answer the "texture of your interactive city" question, they were prompted through an interactive voice system to answer the question using their touch-tone keypad. For unidentified callers their phone number was used in place of a name.

Visualization

A dynamic tree-map [8, 9] visualization was used to display the name off each arrived participant and "texture of interactive city" question. More recently arrived (*i.e.* called in) individuals were displayed larger (Figure 1). Users could also call the number again to "logout" (*i.e.* remove themselves from the visualization).

Resulting Experience

The result was a dynamic visualization of who was in the room and the cross sectional makeup of beliefs across an issue. Users found the tool extremely valuable in allowing them to know who had just arrived or left without checking the room visually. It also meant that the shift from a "gritty" crowd to a "slippery" one could be easily noticed.

Figure 1. The Tree-Map Arrival Information Application showing the individuals in the room categorized by their views on a particular issue – in this case the texture of their interactive city.

Application 2: Group Voting

Our second application was also designed for a symposium, workshop, summit, or other group speaker setting. Variants of our solution could be used in less formal settings such as in a public urban location or movie theater where there is a visible shared screen. For our system, we asked each individual speaker to generate two controversial questions on which users would vote. The questions were designed to have two sides such as "Is X good or bad?" or "Should we have more or less of Y". They were asked to avoid open-ended questions such as "what is the meaning of life?"

There is a rich body of work exploring public group voting [10] from as early as Cinematrix (using colored paddles held by audience members) [11] to more recent uses of SMS voting with American Idol and crowd based motion detection with NewsBreaker Live. Our contribution is not in the concept of group voting techniques but its use within our design constraints.

During their presentation each speaker introduced their first question at which point a large projection screen with the two apposing sides of the issue were shown (Figure 2). Audience members were asked to dial the phone number shown at the top of the screen at which point they appeared as a uniquely colored circle along the middle of the screen. Using the "4" and "6" keys on their keypad, they were able to dynamically move their circle to the position where they felt they stood on the issue in question.

Asterisk Configuration

The Asterisk server parsed the caller ID of each call. This meant that if a user hung up they could call back and regain control of the same voting circle on the visualization. The server was able to hold 30 simultaneous calls due mainly to bandwidth limitations (*i.e.* each call is a 128kbs voice connection). Each call was also given 60 seconds to vote during which their touch-tone signals were parsed in real time and handed off to the visualization tool.

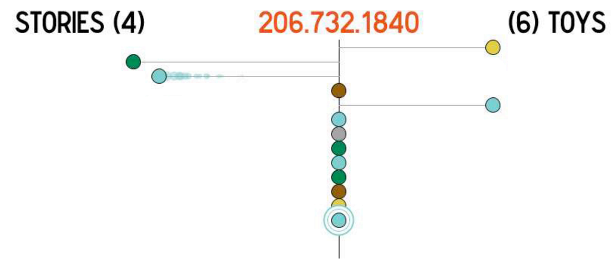


Figure 2. Voter visualization for the question, "Do we need more toys or more stories". The bottom blue caller has just called in.

Visualization

The voting visualization was built with Processing, an open source java-based visualization development tool. Each new caller appeared as a colored circle in the middle of the two voting fields with animated rings around them (Figure 2). The color and style of the caller's icon is deterministic with subsequent calls by the same user resulting in the same visual icon. Callers are plotted vertically as they arrive and can change their vote (using '4' for left and '6' for right on their mobile phones). As their icon moves it leaves an animated trail to highlight the direction of their vote (Figure 3). To more easily identify an individual's icon on the voting screen, a caller could press the '5' key on their phone to send out a rippling animation from their icon. At any time a caller was able to call back in to change their vote. This actually happened quite frequently as the individual reflected on their initial decision.

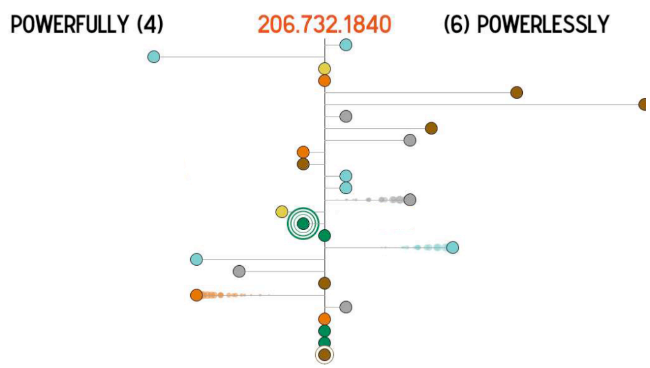
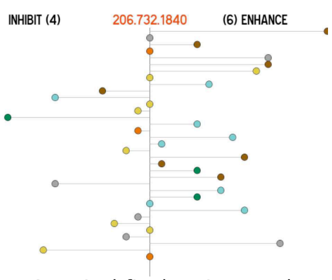


Figure 3. Group Voting in progress with animated trails and ripple highlights shown

Resulting Experience

This public voting system allowed speakers to gauge the feelings and reactions of their audience in real time as they presented on an issue without pausing for discussion. We had asked the speakers to present the first question and simply move on with their presentation, allowing the voting to continue in the background on the side projection screen. This was a novel new addition to the presentation format and warmly received by both speaker and audience alike. Each speaker's final question was presented at the end of the talk allowing for a more public discussion and debate about the issue. Rather than posing a question that only a small number of people would be able to participate in discussing due to the typical time for questions after a talk, the Group Voting tool allowed for an instant snapshot of the audience's view on a topic or issue. Individuals were able to see how their own views were positioned within the scope of the crowd.



A typical final voting result

Call Back Mechanism

One of the more compelling issues is that after discussing the topic for some time it was clear by their votes that certain individuals held opposing views on the issue. Clicking on a vote in the visualization triggered a mechanism that called the individual's phone that cast the vote. Recall that every voter has a caller ID and hence could easily be called back.

Audience members were asked, unlike most public gatherings, to leave their mobile phone in the on/ring position. When a vote from one side of issue was selected, the sound of a ringing mobile phone could be heard in the audience – identifying the person and allowing them to further elaborate on their position. This mechanism had a drawback in that after being used several times, individuals were reluctant to cast extreme votes for fear of being singled out and identified. There were never any guarantees given about the anonymity of the system.

Overall the system was used at three separate events with audiences of up to several hundred in attendance. Overall, the voting application was successfully used 15 times.

Taking a challenging problem of conflicting handsets and mobile phone hardware types, we constrained the design space to uncover applications for mobile phones that avoid the awkward issues of software downloads and compatibility issues. The result is a set of two such compelling applications. We see similar design application that could use our same AnyPhone framework for games and play.

Conclusion

At the outset we set forth a seemingly impossible and rigorous set of design constraints for creating interactive experiences using any mobile phone. We motivated the importance of granting design license to explore a design space that optimizes universal usability and accessibility over performance and technological features. We presented two such solutions that each created compelling new interactive social experiences using any mobile device. The contribution is in the design method and unconventional thinking that demonstrates that expressive and compelling solutions can be found without exploiting the latest technologies and feature sets. We argue that such methodological processes sharpen our design thinking and prevent us from clustering our designs around the various local-minima of a design space.

References

[1] http://www.j2mepolish.org/, "J2ME Polish," 2003.

[2] http://www.trilibis.com/smartpath.php, "SmartPath," 2005.

[3] A. Dunne and F. Raby, *Design Noir: The Secret Life of Electronic Objects*: Birkhauser, 2001.

[4] K. Anderson, E. Paulos, A. Burke, and A. Williams, "Ex-Urban Noir" Workshop at Ubiquitous Computing Conference, 2006.

[5] M. Struppek, "The social potential of Urban Screens," *Visual Communication, Sage Publications*, vol. 5, pp. 173-188, 2006.

[6] T. Paek, M. Agrawala, S. Basu, S. Drucker, T. Kristjansson, R. Logan, K. Toyama, and A. Wilson, *Toward universal mobile interaction for shared displays*. Chicago, Illinois, USA: ACM Press, 2004.

[7] J. Smith, J. V. Meggelen, and L. Madsen, *Asterisk: The Future of Telephony*: O'Reilly Media, 2005.

[8] B. B. Bederson, B. Shneiderman, and M. Wattenberg, "Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies," *ACM Transactions on Graphics (TOG)*, vol. 21, pp. 833-854, 2002.

[9] B. Shneiderman, "Tree visualization with tree-maps: 2-d space-filling approach," *ACM Transactions on Graphics (TOG)* vol. 11, pp. 92-99, 1992.

[10] "CSCW 2002 Workshop on Public, Community and Situated Displays," 2002.

[11] L. Carpenter, "Method and Apparatus for Audience Participation by Electronic Imaging." USA: US Patent #5210604, 1993.