UNIT - IV

Human Computer Interaction – SIT1401

IV MOBILE HCI

UNIT 4 MOBILE HCI

Mobile Ecosystem: Platforms, Application frameworks - Types of Mobile Applications: Widgets, Applications, Games - Mobile Information Architecture, Mobile 2.0, Mobile Design: Elements of Mobile Design, Tools.

4.1 The Mobile Ecosystem

Mobile is an entirely unique ecosystem and, like the Internet, it is made up of many different parts that must all work seamlessly together. With mobile technology, the parts are different, and because you can use mobile devices to access the Internet, that means that not only do you need to understand the facets of the Internet, but you also need to understand the mobile ecosystem.

4.1.1 Operators

The base layer in the mobile ecosystem is the operator. Operators go by many names, depending on what part of the world you happen to be in or who you are talking to. Operators can be referred to as Mobile Network Operators (MNOs); mobile service providers, wireless carriers, or simply carriers; mobile phone operators; or cellular companies. In the mobile community, we officially refer to them as operators, though in the United States, there is a tendency to call them Carriers. Operators are what essentially make the entire mobile ecosystem work. They are the gatekeepers to the kingdom. They install

cellular towers, operate the cellular network, make services (such as the Internet) available for mobile subscribers, and they often maintain relationships with the subscribers, handling billing and support, and offering subsidized device sales and a network of retail stores. The operator‘s role in the ecosystem is to create and maintain a specific set of wireless services over a reliable cellular network. That‘s it. However, to grow the mobile market over the past decade, the operator has been required to take a greater role in the mobile ecosystem, doing much more than just managing the network. For example, they have had to establish trust with subscribers to handle the billing relationship and to offer.

4.1.2 Networks

Operators operate wireless networks. Remember that cellular technology is just a radio that receives a signal from an antenna. The type of radio and antenna determines the capability of the network and the services you can enable on it. You‘ll notice that the vast majority of networks around the world use the GSM standard, using GPRS or GPRS EDGE for 2G data and UMTS or HSDPA for 3G. We also have CDMA (Code Division Multiple Access) and its 2.5G hybrid CDMA2000, which offers greater coverage than its more widely adopted rival. So in places like the United States or China, where people are more spread out, CDMA is a great technology. It uses fewer towers, giving subscribers fewer options as they roam networks.

4.1.3 Devices

What you call phones, the mobile industry calls handsets or terminals. These are terms that I think are becoming outdated with the emergence of wireless devices that rely on operator networks, but do not make phone calls. The number of these ―other‖ devices is a small piece of the overall pie right now, but it‘s growing rapidly. Let‘s focus on the biggest slice of the device pie—mobile phones. As of 2008, there are about 3.6 billion mobile phones currently in use around the world; just more than half the planet‘s population has a mobile phone Most of these devices are feature phones, making up the majority of the marketplace. Smartphones make up a small sliver of worldwide market share and maintain a healthy percentage in the United States and the European Union; smartphone market share is growing with the introduction of the iPhone and devices based on the Android platform. As next- generation devices become a reality, the distinction between feature phones and smartphones will go away. In the next few years, feature phones will largely be located in emerging and developing markets.

4.2 PLATFORMS

A mobile platform_s primary duty is to provide access to the devices. To run

software and services on each of these devices, you need a platform, or a core programming language in which all of your software is written. Like all software platforms, these are split into three categories: licensed, proprietary, and open source.

4.2.1 Licensed

Licensed platforms are sold to device makers for nonexclusive distribution on devices. The goal is to create a common platform of development Application Programming Interfaces (APIs) that work similarly across multiple devices with the least possible effort required to adapt for device differences, although this is hardly reality. Following are the licensed platforms:

Java Micro Edition (Java ME)

Java ME is by far the most predominant software platform of any kind in the mobile ecosystem. It is a licensed subset of the Java platform and provides a collection of Java APIs for the development of software for resource constrained devices such as phones.

Binary Runtime Environment for Wireless (BREW)

BREW is a licensed platform created by Qualcomm for mobile devices, mostly for the U.S. market. It is an interface-independent platform that runs a variety of application frameworks, such as C/C++, Java, and Flash Lite.

Windows Mobile

Windows Mobile is a licensable and compact version of the Windows operating system, combined with a suite of basic applications for mobile devices that is based on the Microsoft Win32 API.

LiMo

LiMo is a Linux-based mobile platform created by the LiMo Foundation. Although Linux is open source, LiMo is a licensed mobile platform used for mobile devices. LiMo includes SDKs for creating Java, native, or mobile web applications using the WebKit browser framework.

4.2.2 Proprietary

Proprietary platforms are designed and developed by device makers for use on their devices. They are not available for use by competing device makers. These include: Palm **Palm** uses three different proprietary platforms. Their first and most recognizable is the Palm OS platform based on the C/C++ programming language; this was initially developed for their Palm Pilot line, but is now used in low-end smartphones such as the Centro line. As Palm moved into higher-end smartphones, they started using the Windows Mobile-based platform for devices like the Treo line. The most recent platform is called webOS, is based

on the WebKit browser framework, and is used in the Prē line.

BlackBerry

Research in Motion maintains their own proprietary Java-based platform, used exclusively by their BlackBerry devices.

iPhone

Apple uses a proprietary version of Mac OS X as a platform for their iPhone and iPod touch line of devices, which is based on Unix.

4.2.3 Open Source

Open source platforms are mobile platforms that are freely available for users to download, alter, and edit. Open source mobile platforms are newer and slightly controversial, but they are increasingly gaining traction with device makers and developers. Android is one of these platforms. It is developed by the Open Handset Alliance, which is spearheaded by Google. The Alliance seeks to develop an open source mobile platform based on the Java programming language.

4.3 APPLICATION FRAMEWORKS

Application frameworks often run on top of operating systems, sharing core services such as communications, messaging, graphics, location, security, authentication, and many others.

Java

Applications written in the Java ME framework can often be deployed across the majority of Java-based devices, but given the diversity of device screen size and processor power, cross-device deployment can be a challenge.

S60

The S60 platform, formerly known as Series 60, is the application platform for devices

that run the Symbian OS. S60 is often associated with Nokia devices—Nokia owns the platform—but it also runs on several non-Nokia devices. S60 is an open source framework. S60 applications can be created in Java, the Symbian C++ framework, or even Flash Lite.

BREW

Applications written in the BREW application framework can be deployed across the majority of BREW-based devices, with slightly less cross-device adaption than other frameworks.

Flash Lite

Adobe Flash Lite is an application framework that uses the Flash Lite and ActionScript frameworks to create vector-based applications. Flash Lite applications can be run within the Flash Lite Player, which is available in a handful of devices around the world.

Flash Lite is a promising and powerful platform, but there has been some difficulty getting it on devices. A distribution service for applications written in Flash Lite is long overdue.

Windows Mobile

Applications written using the Win32 API can be deployed across the majority of Windows Mobile-based devices. Like Java, Windows Mobile applications can be downloaded and installed over the air or loaded via a cable-connected computer.

Cocoa Touch

Cocoa Touch is the API used to create native applications for the iPhone and iPod touch. Cocoa Touch applications must be submitted and certified by Apple before being included in the App Store. Once in the App Store, applications can be purchased, downloaded, and installed over the air or via a cable-connected computer.

Android SDK

The Android SDK allows developers to create native applications for any device that runs the Android platform. By using the Android SDK, developers can write applications in C/C++ or use a Java virtual machine included in the OS that allows the creation of applications with Java, which is more common in the mobile ecosystem.

Web Runtimes (WRTs)

Nokia, Opera, and Yahoo! provide various Web Runtimes, or WRTs. These are meant to be miniframeworks, based on web standards, to create mobile widgets. Both Opera_s and
Nokia_s WRTs meet the W3C-recommended specifications for mobile widgets.

WebKit

WebKit is a browser technology, so applications can be created simply by using web technologies such as HTML, CSS, and JavaScript. WebKit also supports a number of recommended standards not yet implemented in many desktop browsers. Applications can be run and tested in any WebKit browser, desktop, or mobile device.

The Web

The Web is the only application framework that works across virtually all devices and all platforms. Although innovation and usage of the Web as an application framework in

mobile has been lacking for many years, increased demand to offer products and services outside of operator control, together with a desire to support more devices in shorter development cycles, has made the Web one of the most rapidly growing mobile application platforms to date.

4.4 Types of Mobile Applications

4.4.1 Mobile Application Medium Types

The mobile medium type is the type of application framework or mobile technology that presents content or information to the user. It is a technical approach regarding which type of medium to use; this decision is determined by the impact it will have on the user experience.

SMS

The most basic mobile application you can create is an SMS application. Although it might seem odd to consider text messages applications, they are nonetheless a designed experience. Given the ubiquity of devices that support SMS, these applications can be useful tools when integrated with other mobile application types. Typically, the user sends a single keyword to a five-digit short code in order to return information or a link to premium content. For example, sending the keyword ―freebie‖ to a hypothetical short code ―12345‖ might return a text message with a coupon code that could be redeemed at a retail location, or it could include a link to a free ringtone. SMS applications can be both ―free,‖ meaning that there is no additional charge beyond the text message fees an operator charges, and

―premium,‖ meaning that you are charged an additional fee in exchange for access to premium content. The most common uses of SMS applications are mobile content, such ringtones and images, and to interact with actual goods and services. Some vending machines can dispense beverages when you send them an SMS; SMS messages can also be used to purchase time at a parking meter or pay lot. A great example of how SMS adds incredible value would be Twitter, where users can receive SMS alerts from their friends and post to their timeline from any mobile device.

4.4.2 Mobile websites

Mobile website is a website designed specifically for mobile devices, not to be confused with viewing a site made for desktop browsers on a mobile browser. Mobile websites are characterized by their simple ―drill-down‖ architecture, or the simple presentation of navigation links that take you to a page a level deeper.

Mobile websites often have a simple design and are typically informational in nature, offering few—if any—of the interactive elements you might expect from a desktop site. Mobile websites have made up the majority of what we consider the mobile web for the past decade, starting with the early WML-based sites (not much more than a list of links) and moving to today‘s websites, with a richer experience that more closely resembles the visual aesthetic users have come to expect with web content. Though mobile websites are fairly easy to create, they fail to display consistently across multiple mobile browsers—a trait common to all mobile web mediums. The mobile web has been gradually increasing in usage over the years in most major markets, but the limited experience offered little incentive to the user. Many compare the mobile web to a 10-year-old version of the Web: slow, expensive to use, and not much to look at. As better mobile browsers started being introduced to device platforms like the iPhone and Android, the quality of mobile websites began to improve dramatically, and with it, usage improved. For example, in just one year, the U.S. market went from being just barely in the top five consumers of the mobile web to number one, largely due to the impact of the iPhone alone.

Pros

• They are easy to create, maintain, and publish.

• They can use all the same tools and techniques you might already use for desktop sites.

• Nearly all mobile devices can view mobile websites.

Cons

The cons of mobile websites are:

• They can be difficult to support across multiple devices.

• They offer users a limited experience.

• Most mobile websites are simply desktop content reformatted for mobile devices.

4.4.3 Mobile Web Widgets

Largely in response to the poor experience provided by the mobile web over the years, there has been a growing movement to establish mobile widget frameworks and platforms. For years the mobile web user experience was severely underutilized and failed to gain traction in the market, so several operators, device makers, and publishers began creating widget platforms (Figure) to counter the mobile web_s weaknesses.

Figure: An example mobile web widget

I initially saw mobile web widgets as another attempt by the mobile industry to hype a technology that no one wants. I liked to quiz mobile web widget advocates about what makes mobile web widgets different than what we can do with the mobile web.

A component of a user interface that operates in a particular way. The ever-trusty Wikipedia defines a web widget this way:

A portable chunk of code that can be installed and executed within any separate HTML based web page by an end user without requiring additional compilation.

Between these two definitions is a better answer:

A mobile web widget is a standalone chunk of HTML-based code that is executed by the end user in a particular way. Mobile web widgets are small web applications that can_t run by themselves; they need to be executed on top of something else. I think one reason for all the confusion around what is a mobile web widget is that this definition can also encompass any web application that runs in a browser. Opera Widgets, Nokia Web Run Time (WRT), Yahoo! Blueprint, and Adobe Flash Lite are all examples of widget platforms that work on a number of mobile handsets. Using a basic knowledge of HTML (or vector graphics in the case of Flash), you can create compelling user experiences that tap into device features and, in many cases, can run while the device is offline.

Pros

The pros of mobile web widgets are:

• They are easy to create, using basic HTML, CSS, and JavaScript knowledge.

• They can be simple to deploy across multiple handsets.

• They offer an improved user experience and a richer design, tapping into device features and offline use.

Cons

The cons of mobile web widgets are:

• They typically require a compatible widget platform to be installed on the device.

• They cannot run in any mobile web browser.

• They require learning additional proprietary, non-web-standard techniques.

4.4.4 Mobile Web Applications

Mobile web applications are mobile applications that do not need to be installed or compiled on the target device. Using XHTML, CSS, and JavaScript, they are able to provide an application-like experience to the end user while running in any mobile web browser. By Application -like experience, I mean that they do not use the drill-down or page metaphors in which a click equals a refresh of the content in view. Web applications allow users to interact with content in real time, where a click or touch performs an action within the current view.

The history of how mobile web applications came to be so commonplace is interesting, and is one that I think can give us an understanding of how future mobile trends can be assessed and understood. Shortly after the explosion of Web 2.0, web applications like Facebook, Flickr, and Google Reader hit desktop browsers, and there was discussion of how to bring those same web applications to mobile devices. The Web 2.0 movement brought user-centered design principles to the desktop web, and those same principles were sorely needed in the mobile web space as well.

The challenge, as always, was device fragmentation. The mobile browsers were years behind the desktop browsers, making it nearly impossible for a mobile device to render a comparable experience. While XHTML support had become fairly commonplace across devices, the rendering of CSS2 was wildly inconsistent, and support for Java- Script, necessary or simple DHTML, and Ajax was completely nonexistent. To make matters worse, the perceived market demand for mobile web applications was not seen as a priority with many operators and device makers. It was the classic chickenor- the-egg scenario. What had to come first, market demand to drive browser innovation or optimized content to drive the market? With the introduction of the first iPhone, we saw a cataclysmic change

across the board.

Using WebKit, the iPhone could render web applications not optimized for mobile

devices as perfectly usable, including DHTML- and Ajax-powered content. Developers quickly got on board, creating mobile web applications optimized mostly for the iPhone (Figure). The combination of a high-profile device with an incredibly powerful mobile web browser and a quickly increasing catalog of nicely optimized experiences created the perfect storm the community had been waiting for.

Usage of the mobile web exploded with not just users of the iPhone, but users of other handsets, too. Because Web applications being created for the iPhone were based on web standards, they actually worked reasonably well on other devices. Operators and device makers saw that consumers wanted not just the mobile web on their handsets, but the regular Web, too.

Pros:

The pros of mobile web applications are:

• They are easy to create, using basic HTML, CSS, and JavaScript knowledge.

• They are simple to deploy across multiple handsets.

• They offer a better user experience and a rich design, tapping into device features & offline use.

• Content is accessible on any mobile web browser.

Cons:

The cons of mobile web applications are:

• The optimal experience might not be available on all handsets.

• They can be challenging (but not impossible) to support across multiple devices.

• They don_t always support native application features, like offline mode, location lookup, file system access, camera, and so on.

4.4.5 Games

The most popular of all media available to mobile devices. Technically games are really just native applications that use the similar platform SDKs to create immersive experiences (Figure). But I treat them differently from native applications for two reasons: they cannot be easily duplicated with web technologies, and porting them to multiple mobile platforms is a bit easier than typical platform-based applications.

Seeing as how we have yet to see these types of gaming experiences appear on the desktop using standard web technologies, I believe we are still a few years out from seeing them on mobile devices. Adobe_s Flash and the SVG (scalable vector graphics) standard are the only way to do it on the Web now, and will likely be how it is done on mobile devices in the future, the primary obstacle being the performance of the device in dealing with vector graphics. The reason games are relatively easy to port (―relatively‖ being the key word), is that the bulk of the gaming experience is in the graphics and actually uses very little of the device APIs. The game mechanics are the only thing that needs to adapted to the various platforms. Like in console gaming, there are a great number of mobile game porting shops that can quickly take a game written in one language and port it to another.

These differences, in my mind, are what make mobile games stand apart from all other application genres—their capability to be unique and difficult to duplicate in another application type, though the game itself is relatively easy to port. Looking at this model for other application areas—namely, the mobile web—could provide helpful insight into how we create the future of mobile web applications.

Pros: The pros of game applications are:

• They provide a simple and easy way to create an immersive experience.

• They can be ported to multiple devices relatively easily.

Cons: The cons of game applications are:

• They can be costly to develop as an original game title.

• They cannot easily be ported to the mobile web.

4.5 MOBILE INFORMATION ARCHITECTURE

What Is Information Architecture?

The structural design of shared information environments

- The combination of organizations, labelling, search, and navigation systems within websites and intranets
- The art and science of shaping information products and experiences to support usability and find ability
- An emerging discipline and community of practice focused on bringing principles of design and architecture to the digital landscape

Information architecture

The organization of data within an informational space. In other words, how the user will get to information or perform tasks within a website or application.

Interaction design

The design of how the user can participate with the information present, either in a direct or indirect way, meaning how the user will interact with the website of application to create a more meaningful experience and accomplish her goals.

Information design

The visual layout of information or how the user will assess meaning and direction given the information presented to him.

Navigation design

The words used to describe information spaces; the labels or triggers used to tell the users what something is and to establish the expectation of what they will find.

Interface design

The design of the visual paradigms used to create action or understanding.

The role of information architecture is played by a variety of people, from product managers to designers and even developers. To make things more confusing, information architecture can be called many different things throughout the design and development process. Words like intuitive, simple, findable, usable, or the executive favourite easy to-use—all describe the role that information architects play in creating digital experiences.

The visual design of your product, what frameworks you use, and how it is developed are integral to the success of any product, but the information architecture stands apart as being the most crucial element of your product. It is the first line of scrimmage—the user_s first impression of your product. Even if you have the best design, the best code, and the best backend service, if the user cannot figure out how to use it, she will fail and so will

your product.

Mobile Information Architecture

Information architecture has become a common discipline in the web industry; unfortunately, the mobile industry like software has only a handful of specialized mobile information architects. Although mobile information architecture is hardly a discipline in its own right, it certainly ought to be. This is not because it is so dissimilar from its desktop cousin, but because of context, added technical constraints, and needing to display on a smaller screen as much information as we would on a desktop.

The role of a mobile information architect would be to interpret this content to the mobile context. Do you use the same structure, or sections? Do you present the same information above the fold? If so, how should that be prioritized? How does the user navigate to other areas? Do you use the same visual and interaction paradigms, or invent new ones? And if you do start to invent new paradigms, will you lose the visual characteristics of what users expect?

Keeping It Simple

When thinking about your mobile information architecture, you want to keep it as simple as possible.

Support your defined goals

If something doesn_t support the defined goals, lose it. Go back to your user goals and needs, and identify the tasks that map to them. Find those needs and fill them.

Clear, simple labels

Good trigger labels, the words we use to describe each link or action, are crucial in Mobile. Words like products or services aren_t good trigger labels. Users have a much higher threshold of pain when clicking about on a desktop site or application, hunting and pecking for tasty morsels. Mobile performs short, to-the-point, get-it-quick, and get-out types of tasks. What is convenient on the desktop might be a deal breaker on mobile.

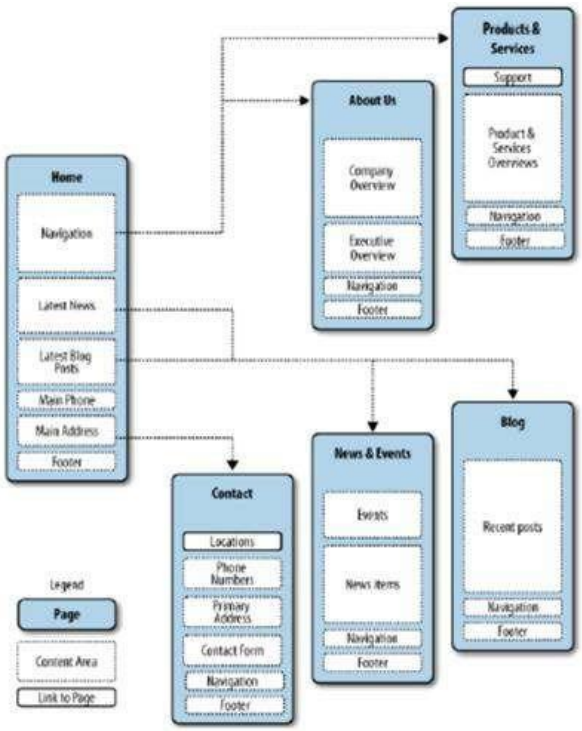
4.5.1 Site Maps

Relationship of content to other content and provide a map for how the user will travel through the informational space Limit opportunities for mistakes In Figure, you can see a poorly designed mobile information architecture that too closely mimics its desktop cousin; it was not designed with the mobile user in mind.

In the mobile context, tasks are short and users have limited time to perform them. And with mobile websites, we can_t assume that the users have access to a reliable broadband connection that allows them to quickly go back to the previous page. In addition, the users more often than not have to pay for each page view in data charges. So not only do they pay cash for viewing the wrong page by mistake, they pay to again download the page they started from: we can_t assume that pages will be cached properly.

Confirm the path by teasing content

Information-heavy sites and applications often employ nested or drill-down architectures, forcing the user to select category after category to get to their target. To reduce risking the user_s time and money, we want to make sure we present enough information for the user to wade through our information architecture successfully. On the Web, we take these risks very lightly, but with mobile, we must give our users a helping hand. We do this by teasing content within each category— that is, providing at least one content item per category.



4.5.2 Clickstreams

Clickstream is a term used for showing the behavior on websites, displaying the order in which users travel through a site_s information architecture, usually based on data gathered from server logs. Clickstreams are usually historical, used to see the flaws in your information architecture, typically using heat-mapping or simple percentages to show where your users are going. I_ve always found them to be a useful tool for researching large websites. The maps the ideal path the user will take to perform common tasks. Being able to visually lay out the path users will take gives you a holistic or bird_s-eye view of your mobile information architecture, just as a road map does. When you can see all the paths next to each other and take a step back, you start to see shortcuts and how you can get users to their goal faster or easier, as shown in Figure. Just create user or process flows, such as the esoteric diagram shown in Figure, which is made up of boxes and diamonds that look more like circuit board diagrams than information architecture. If that is what your team prefers, then by all means, flow away. Personally, I like to present all of my information architecture deliverables from the perspective of the user, using the same metaphors she will use to make her way through my information architecture in this case, either a screen or page metaphor.

4.5.3 Wireframes

The next information architecture tool at our disposal is wireframes. Wireframes are

a way to lay out information on the page, also referred to as information design. Site maps show how our content is organized in our informational space; wireframes show how the user will directly interact with it. Wireframes are like the peanut butter to the site map jelly in our information architecture sandwich. It_s the stuff that sticks. Wireframes like the one in Figure serve to make our information space tangible and useful. Wireframes to be one of the most valuable information deliverables to communicate my vision for how a site or app will work, the challenge is that a diagram on a piece of paper doesn_t go a long way toward describing how the interactions will work. Most common are what I call ―in-place interactions or areas where the user can interact with an element without leaving the page. This can be done with Ajax or a little show/hide JavaScript. These interactions can include copious amounts of annotation, describing each content area in as much length as you can fit in the margins of the page.

4.5.4 Prototyping

Prototypes might sound like a scary (or costly) step in the process. Some view them as redundant or too time-consuming, preferring to jump in and start coding things. But as with wireframes, I_ve found that each product we_ve built out some sort of prototype has saved both time and money.

Paper prototypes

The most basic level we have is paper prototyping: taking our printed-out wireframes or even drawings of our interface, like the one shown in Figure, and putting them in front of people.

Context prototype

The next step is creating a context prototype (Figure). Take a higher-end device that enables you to load full-screen images on it. Take your wireframes or sketches and load them onto the device, sized to fill the device screen. Leave the office. Go for a walk down to your nearest café. Or get on a bus or a train. Pay particular attention to what you are thinking and your physical behavior while you are using your interface and then write it down. If you are brave and don‗t have strict nondisclosure issues, ask the people around you to use it, too. I wouldn‗t bother with timing interactions or sessions, but try to keep an eye on a clock to determine how long the average session is.

4.6 Mobile 2.0

The Web as a platform for the mobile context, this means ―write once, deploy everywhere,‖ moving away from the costly native applications deployed over multiple frameworks and networks.

Harnessing collective intelligence this isn‗t something the mobile community has done much of, but projects like WURFL—an open source repository of device profiles provided by the community—is exactly what mobile needs more of.

Data is the next Intel inside Mobile takes this principle several steps further. It can include the data we seek, the data we create, and the data about or around our physical locations.

End of the software release cycle Long development and testing cycles heavily weigh on mobile projects, decreasing all hopes of profitability. Shorter agile cycles are needed to make mobile development work as a business. Releasing for one device, iterating, improving, and then releasing for another is a great way to ensure profitability in mobile.

Lightweight programming models Because mobile technology is practically built on enterprise Java, the notion of using lightweight models is often viewed with some skepticism. But decreasing the programming overhead required means more innovation occurs faster.

Software above the level of a single device This effectively means that software isn‗t just about computers anymore. We need to approach new software as though the user will demand it work in multiple contexts, from mobile phones to portable gaming consoles and e- book readers.

Rich user experiences a great and rich user experience helps people spend less time with the software and more time living their lives. Mobile design is about enabling users to live their lives better.

4.6.1 The Convergence of the Web and Mobile

It is obvious that in the minds of many, Mobile 2.0 is the Web. At this point, the mobile web has always been viewed as a second-class citizen within the mobile ecosystem, for many reasons, as discussed later. Mobile is already a medium, but the consensus is that by leveraging the power of the Web, integrating web services into the mobile medium is the future of mobile development. When the iPhone exploded onto the scene, it increased the usage of the mobile web by its users to levels never seen before. The spur of new mobile web apps created just for the iPhone doubled the number of mobile websites available in under a year. If Web 2.0 taught us that the Web is the platform, then Mobile 2.0 tells us that mobile will be the primary context in which we leverage the Web in the future.

4.6.2 The Mobile Web Browser

If the future of mobile is the Web, then it only makes sense that the mobile web browser is the next killer app of mobile. Again, this is something we saw confirmed with WebKit in the iPhone and later in Android. However, of particular concern is how device fragmentation factors into mobile browsers. For example, how can we expect developers to support more than 30 different mobile browsers? A fellow panelist from the Mozilla Minimo project offered a potential solution in consolidation—that we will see only a few mobile browsers in the future; specifically, browsers built on Mozilla, Opera, Internet Explorer, and WebKit technologies. At the time, I thought that prediction was too focused on smart phones, but in the years since, the line between smart phone and feature phone seems to be going away, so this prediction is fairly accurate. But the single biggest challenge in mobile remains device fragmentation. The mobile browser enables us to penetrate the problem by not having our content locked so specifically to the device abilities, screen size, and form factor, but device fragmentation still causes old, outdated browsers to remain in the market long after they should be put out to pasture. What appears to be solving browser fragmentation is actually the iPhone. The Mobile Safari browser included with the iPhone provided such an excellent web experience on a mobile device that it drove use of the mobile web to huge levels, which means big profits for the operators. This also means that the mobile web is no longer a secondclass citizen. In the post-iPhone market, all new devices are judged by the quality of their mobile web browser. Operators know it and therefore are demanding better

4.6.3 JavaScript

If you are going to provide mobile web applications, you have to have a mobile web

browser that supports Ajax, or, as it is technically known, XML Http Request. It makes a lot of those cool interactions in your web browser work via the capability to load content asynchronously into your browser view. But it isn‘t just Ajax; it is JavaScript, a web technology that has largely been ignored with most mobile web browsers. Ajax is great, but just being able to do a little show hide or change a style after you click or touch it goes a long way toward improving the user experience. This is probably where mobile web browsers fall behind desktop browsers the most. Because they both support XHTML and CSS relatively well, JavaScript has been a nogo in mobile for years. In order for mobile web apps to rival native applications, you have to support some JavaScript. Modern mobile browsers have made much progress over the last few years, but there is still plenty of work to be done. For example, accessing the device capabilities like the phone book or file system with JavaScript doesn‘t work in a consistent way. These problems still need to be solved in order to truly reap the benefits of the Web.

4.6.4 Mobile User Experience

Traditionally, the user experience available in the mobile web has been like using a website from 1995: mostly text-based, difficult to use, and ugly as sin. This isn‘t to say that the user experience of mobile applications has been much better, but it used to be that if you wanted a good experience, you built a native app. Descriptions within the industry range from the honest ―the mobile user experience is utterly horrid,‖ to the sales pitch of ―look at these cool things you can do,‖ to the optimistic ―the mobile user experience is the future!‖ These polar attitudes toward the mobile user experience are somewhat ironic, given that the mobile user experience was largely ignored for close to a decade. People in mobile treat the user experience like a chicken-and-egg scenario: bad input/output of the user experience prevents adoption, but designing a shiny user experience with bells and whistles will bring them in droves. Device APIs usually force you to use their models of user experience, meaning that you have to work in the constraints of the API. This is good for creating consistent experiences for that platform, but these experiences don‘t translate to others. For example, would you take an iPhone app design and put it on an Android device? The user experience for these devices is similar but still remains different. The beauty of the Web, literally, is that you can design whatever experience you want, for better or worse. You are in control of the presentation and can establish your own visual metaphors. The problem has been that traditionally complex (which often equates to good) user experiences haven‘t been possible on mobile devices. Modern mobile web browsers, as they come closer to their desktop counterparts, remove this distinction, giving us the same canvas on mobile

devices that we have for the desktop. This means that creating mobile experiences just got a whole lot easier. It also means we can have a consistent user experience across multiple mediums.

4.7 MOBILE DESIGN

When building mobile experiences, it is impossible to create a great experience without three ingredients: context, information architecture, and visual design. The visual design of your experience is the direct representation of everything underneath; it is the first impression the user will have. A great design gives the user high expectations of your site or application; a poor design leads to lower expectations. Users‘ expectations translate to value and trust. In the mobile space, where content is often ―free‖ (they still need to pay for data charges), users often have low expectations, due to the limitations of the canvas. Confusing site structures and slow download speeds reinforce those initial low expectations. In the downloadable application space, where application design can be much more robust, users must purchase applications almost sight unseen. For example, they may see just a small screenshot of your application or game. But if the application doesn‘t meet the higher expectations of the design, your application downloads will drop like a stone. The design, that first impression, determines right from the start if the user will spend five seconds, five minutes, or five hours with your product. This leads us to the most significant challenge in mobile design: creativity. You can‘t always be as creative as you want to be. Many devices just can‘t support complex designs for every channel; for example, on many lower-end devices, the mobile web experience may just be a list of links. But every device has the capability to create a bestin- device experience; it just depends on how you take advantage of the application medium and context that you plan to use.

4.7.1 Interpreting Design

Days were spent creating print designs and advertisements, defining precisely what each design would be, down to the picas and points. The method of design is meant for creating a vision for how to communicate information or ideas in his head, and then duplicating that on the printed page. Precise designs might look better, but they can be brutal to implement. More flexible designs might not be much to look at, but they work for the most users, or the lowest common denominator. But more than that, our backgrounds and our training can actually get in the way of creating the best design for the medium. We like to apply the same rules to whatever the problem in front of us might be. In mobile design, you interpret what you know about good design and translate it to this new medium that is both technologically precise and at times incredibly unforgiving, and you provide the design

with the flexibility to present information the way you envision on a number of different devices. In the end, the graphic designer and I scrapped the work, and he provided me his pica perfect designs as giant images, which I turned into a series of massive image maps.

4.7.2 The Mobile Design Tent-Pole

In Hollywood, executives like to use the term ―tent-pole‖ to describe their movies and television shows. The term has dual meanings: one is business, and the other creative. The business goal of a tent-pole production is to support or prop up the losses from other productions. However, to create a tent-pole production, the creators involved must make an artistic work that they know will appeal to the largest possible audience, providing something for everyone. You probably know tent-pole movies as ―blockbusters‖; in television, they are known as ―anchor‖ shows. Trying to reach the widest possible audience poses a problem. Hollywood is learning with great pains that with so many entertainment options and with today‘s audience being so hard to reach through traditional advertising channels, tent-pole productions are failing. As the number of social niches increases, it becomes difficult to satisfy the specific tastes of each social group. What one group finds hysterically funny, several other groups might find offensive. Today, tent-pole productions often come off as bland and half-hearted, failing to appease anyone. One of the most interesting examples of how the tide turned in entertainment is with the animated films of Disney versus those of Pixar. For years, Disney produced tentpole family fare quite successfully. But as competition increased, notably from Pixar, Disney films would spend millions to create stale and dated films, losing audiences and revenue. Meanwhile, Pixar found that their movies could be successful by avoiding the traditional storytelling formats of animated film, which Disney essentially defined. Instead, Pixar based their stories around specific emotional themes and was able to connect with audiences of all ages, in multiple cultures and across multiple niches. In 2006, Disney acquired Pixar, making its top executives the new leaders of all Disney creative projects. Although Disney technically acquired Pixar, I‘ve always looked at it as the other way around. Disney realized that it needed to be more Pixar and less Disney in order to grow and adapt to today‘s changing audiences and niches. This is something that Pixar was doing correctly. Back in the world of mobile design, the de facto strategy is to create tent-pole products. Like the old days of Disney, the strategy is to sink millions into creating tent-pole products, or products that support the largest number of devices that no one will ever use. They are creatively stale, they lack inspiration, and they simply exist with no meaningful purpose to the user. They make the same mistake Disney made, thinking that it could simply put something on the

market that might not be the best quality, but because it carried the Disney name, people would buy it. To have a successful mobile design, you have to think like Pixar. Find that emotional connection, that fundamental need that serves many audiences, many cultures, and many niches and design experiences. Too often, designers simply echo the visual trends of the day, mimicking the inspiration of others—I‘m certainly guilty of it. But with mobile design, once you find that essential thing, that chewy nougat we call ‒context‖ that lives at the center of your product, then you will find ample inspiration of your own to start creating designs that translate not only across multiple devices, but across multiple media. Sure, there are countless examples of poorly designed mobile products that are considered a success. You only need to look as far as the nearest mobile app store to find them. This is because of the sight unseen nature of mobile commerce. Traditionally, you couldn‘t demo—or in some cases even see— screenshots of a game or mobile application before you bought it. You simply had to purchase it and find out if it was any good. Apple‘s App Store quickly changed that. You can clearly see that the best-selling games and applications for the iPhone are the ones with the best designs.

THE ELEMENTS OF MOBILE DESIGN

Good design requires three abilities: the first is a natural gift for being able to see visually how something should look that produces a desired emotion with the target audience. The second is the ability to manifest that vision into something for others to see, use, or participate in. The third knows how to utilize the medium to achieve your design goals.

Six elements of mobile design that you need to consider, starting with the context and layering in visual elements or laying out content to achieve the design goal. Then, you

need to understand how to use the specific tools to create mobile design, and finally, you need to understand the specific design considerations of the mobile medium.

4.8.1 Context

I won_t belabor the point except to say that context is core to the mobile experience. As the designer, it is your job to make sure that the user can figure out how to address context using your app. Make sure you do your homework to answer the following questions:

• Who are the users? What do you know about them? What type of behavior can you assume or predict about the users?

• What is happening? What are the circumstances in which the users will best absorb the content you intend to present?

• When will they interact? Are they at home and have large amounts of time? Are they at work where they have short periods of time? Will they have idle periods of time while waiting for a train, for example?

• Where are the users? Are they in a public space or a private space? Are they inside or outside? Is it day or is it night?

• Why will they use your app? What value will they gain from your content or services in their present situation?

• How are they using their mobile device? Is it held in their hand or in their pocket?

•How are they holding it? Open or closed? Portrait or landscape?

The answers to these questions will greatly affect the course of your design. Treat these questions as a checklist to your design from start to finish.

4.8.2 Message

Message is the overall mental impression you create explicitly through visual design. I like to think of it as the holistic or at times instinctual reaction someone will have to your design. If you take a step back, and look at a design from a distance, what is your impression? Or conversely, look at a design for 30 seconds, and then put it down. What words would you use to describe the experience?

Branding shouldn_t be confused with messaging. Branding is the impression your company name and logo gives—essentially, your reputation. Branding serves to reinforce the message with authority, not deliver it. In mobile, the opportunities for branding are limited, but the need for messaging is great. With such limited real estate, the users don_t care about your brand, but they will care about the messaging, asking themselves questions like, ―What can this do for me? or ―Why is this important to me?

Yahoo!

Yahoo! sort of delivers a message. This app provides a clean interface, putting a focus on search and location, using color to separate it from the news content. But I_m not exactly sure what it is saying. Words you might use to describe the message are crisp, clean, and sharp.

ESPN

The ESPN site clearly is missing a message. It is heavily text-based, trying to put a lot of content above the fold, but doesn_t exactly deliver a message of any kind. If you took out the ESPN logo, you likely would have indifferent expectations of this site; it could be about anything, as the design doesn_t help set expectations for the user in any way. Words you might use to describe the message: bold, cluttered, and content-heavy.

Disney

Disney creates a message with its design. It gives you a lot to look at—probably too much—but it clearly tries to say that the company is about characters for a younger audience. Words you might use to describe the message: bold, busy, and disorienting.

Wikipedia

The Wikipedia design clearly establishes a message. With a prominent search and text-heavy layout featuring an article, you know what you are getting with this design. Words you might use to describe the message: clean, minimal, and text-heavy.

Amazon

Amazon sort of creates a message. Although there are some wasted opportunities above the fold with the odd ad placement, you can see that it is mostly about products (which are improved even more if you scroll down). Words you might use to describe the message: minimal but messy, product-heavy, and disorienting.

4.8.3 Look and Feel

Look and feel is used to describe appearance, as in I want a clean look and feel or I want a usable look and feel. The problem is: as a mobile designer, what does it mean? And how is that different than messaging? I think of look and feel in a literal sense, as something real and tactile that the users can look at, and then feel something they can touch or interact with. Look and feel is used to evoke action how the user will use an interface. Messaging is holistic, as the expectation the users will have about how you will address their context. It is easy to confuse the two, because feel can be interpreted to mean our emotional reaction to design and the role of messaging.

I often find myself explaining the look and feel with the word because, with a cause-and-effect rationale for design decisions, as in The user will press this button because...‖ or The user will go to this screen because… followed by a reason why a button or control is designed a certain way. Establishing a look and feel usually comes from wherever design inspiration comes from. However, your personal inspiration can be a hard thing to justify. Therefore we have design patterns, or documented solutions to design problems, sometimes referred to as style guides. On large mobile projects or in companies with multiple designers, a style guide or pattern library is crucial, maintaining consistency in the look and feel and reducing the need for each design decision to be justified.

4.8.4 Layout

Layout is an important design element, because it is how the user will visually process the page, but the structural and visual components of layout often get merged together, creating confusion and making your design more difficult to produce. The first time layout should rear its head is during information architecture. In fact, I prefer to make about 90 percent of my layout decisions during the information architecture period. I ask myself questions like: where should the navigation go on the page or screen? What kind of navigation type should I use? Should I use tabs or a list? What about a sidebar for larger screens? All of these should be answered when defining the information architecture and before you begin to design. Design is just too subjective of an issue. If you are creating a design for anyone but yourself, chances are good that there will be multiple loosely-based-on- experience opinions that will be offered and debated.

Where the design opinions of the CEO or Chief Marketing Officer (CMO) might

influence a design direction more than, say, the Creative Director or Design Director. By defining design elements like layout prior to actually applying the look and feel, you can separate the discussion. As a self-taught designer, I started out in this business making designs for my own projects. I could just put pen to paper and tweak it to my heart_s content. If I wanted to radically change the layout, I could. When I started my mobile design career with my first mobile company more than a decade ago, I realized that this approach didn_t work. The majority of comments that reviewers would make were about the layout. They focused on the headers, the navigation, the footer, or how content blocks are laid out, and so on. But their feedback got muddied with the look and feel, the colors, and other design elements.

Reviewers do make remarks like I like the navigation list, but can you make it look more raised? Most designers don_t hear that; they hear the navigation isn_t right, do it again. But, with this kind of feedback, there are two important pieces of information about different types of design. First, there is confirmation that the navigation and layout are correct.
Second, there is a question about the look and feel. Because designers hear Do it again, they typically redo the layout, even though it was actually fine.

Creating mobile designs in an environment with multiple reviewers is all about getting the right feedback at the right time. Your job is to create a manifestation of a shared vision. Layout is one of the elements you can present early on and discuss independently. People confuse the quality and fidelity of your deliverables as design. By keeping it basic, you don_t risk having reviewers confuse professionalism with design. The irony is that as I become more adept at defining layouts, I make them of increasingly lower fidelity. For example, when I show my mobile design layouts as wireframes during the information architecture phase, I intentionally present them on blueprint paper, using handwriting fonts for my annotations. It also helps to say that this is not a design; it is a layout, so please give me feedback on the layout.

4.8.5 Color

The fifth design element, color, is hard to talk about in a black-and-white book. Maybe it is fitting, because it wasn_t that long ago that mobile screens were available only

inblack and white well, technically, it was black on a green screen). These days, we have nearly the entire spectrum of colors to choose from for mobile designs.

The most common obstacle you encounter when dealing with color is mobile screens, which come in a number of different color or bit depths, meaning the number of bits (binary digits) used to represent the color of a single pixel in a bitmapped image. When complex designs are displayed on different mobile devices, the limited color depth on one device can cause banding, or unwanted posterization in the image.

For an example of posterization, the technical term for when the gradation of tone is replaced with regions of fewer tones, see in Figure 8-10 how dramatically the color depth can affect the quality of a photo or gradient, producing banding in several parts in the image.

Color palettes

Defining color palettes can be useful for maintaining a consistent use of color in your mobile design. Color palettes typically consist of a predefined number of colors to use throughout the design. Selecting what colors to use varies from designer to designer, each having different techniques and strategies for deciding on the colors. I_ve found that I use three basic ways to define a color palette:

Sequential

In this case, there are primary, secondary, and tertiary colors. Often the primary color is reserved as the ―brand color or the color that most closely resembles the brand_s using a color wheel.

Color	Represents
White	Light, reverence, purity, truth, snow, peace, innocence, cleanliness, simplicity, security, humility, sterility, winter, coldness, surrender, fearfulness, lack of imagination, air, death (in Eastern cultures), life, marriage (in Western cultures), hope, bland
Black	Absence, modernity, power, sophistication, formality, elegance, wealth, mystery, style, evil, death (in Western cultures), fear, seriousness, conventionality, rebellion, anarchism, unity, sorrow, professionalism
Gray	Elegance, humility, respect, reverence, stability, subtlety, wisdom, old age, pessimism, boredom, decay, decrepitude, dullness, pollution, urban sprawl, strong emotions, balance, neutrality, mourning, formality
Yellow	Sunlight, joy, happiness, earth, optimism, intelligence, idealism, wealth (gold), summer, hope, air, liberalism, cowardice, illness (quarantine), fear, hazards, dishonesty, avarice, weakness, greed, decay or aging, femininity, gladness, sociability, friendship
Green	Intelligence, nature, spring, fertility, youth, environment, wealth, money (U.S.), good luck, vigor, generosity, go, grass, aggression, coldness, jealousy, disgrace (China), illness, greed, drug culture, corruption (North Africa), life eternal, air, earth (classical element), sincerity, renewal, natural abundance, growth
Blue	Seas, men, productiveness, interiors, skies, peace, unity, harmony, tranquility, calmness, trust, coolness, confidence, conservatism, water, ice, loyalty, dependability, cleanliness, technology, winter, depression, coldness, idealism, air, wisdom, royalty, nobility, Earth (planet), strength, steadfastness, light, friendliness, peace, truthfulness, love, liberalism (U.S. politics), and conservatism (UK, Canadian, and European politics)
Violet	Nobility, envy, sensuality, spirituality, creativity, wealth, royalty, ceremony, mystery, wisdom, enlightenment, arrogance, flamboyance, gaudiness, mourning, exaggeration, profanity, bisexuality, confusion, pride

Color	Represents
Red	Passion, strength, energy, fire, sex, love, romance, excitement, speed, heat, arrogance, ambition, leadership, masculinity, power, danger, gaudiness, blood, war, anger, revolution, radicalism, aggression, respect, martyrs, conservatism (U.S. politics), Liberalism (Canadian politics), wealth (China), and marriage (India)
Orange	Energy, enthusiasm, balance, happiness, heat, fire, flamboyance, playfulness, aggression, arrogance, gaudiness, overemotion, warning, danger, autumn, desire
Pink	Spring, gratitude, appreciation, admiration, sympathy, socialism, femininity, health, love, romance, marriage, joy, flirtatiousness, innocence and child-like qualities
Brown	Calm, boldness, depth, nature, richness, rustic things, stability, tradition, anachronism, boorishness, dirt, dullness, heaviness, poverty, roughness, earth

Adaptive

An adaptive palette is one in which you leverage the most common colors present in a supporting graphic or image. When creating a design that is meant to look native on the device, I use an adaptive palette to make sure that my colors are consistent with the target mobile platform.

Inspired

This is a design that is created from the great pieces of design you might see online, as shown in Figure below, or offline, in which a picture of the design might inspire you. This could be anything from an old poster in an alley, a business card, or some packaging. When I sit down with a new design, I thumb through some of materials to create an inspired

palette. Like with the adaptive palette, you actually extract the colors from the source image, though you should never ever use the source material in a design.

4.9 MOBILE DESIGN TOOLS

Mobile design requires understanding the design elements and specific tools. The closest thing to a common design tool is Adobe Photoshop, though each framework has a different method of implementing the design into the application. Some frameworks provide a complete interface toolkit, allowing designers or developers to simply piece together the interface, while others leave it to the designer to define from scratch.

Mobile framework	Design tool	Interface toolkits
Java ME	Photoshop, NetBeans	JavaFX, Capuchin
BREW	Photoshop, Flash	BREW UI Toolkit, uiOne, Flash
Flash Lite	Flash	Flash Lite
iPhone	Photoshop, Interface Builder	iPhone SDK

Mobile framework	Design tool	Interface toolkits
Android	Photoshop, XML-based themes	Android SDK
Palm webOS	Photoshop, HTML, CSS, and JavaScript	Mojo SDK
Mobile web	Photoshop, HTML, CSS, and JavaScript	W3C Mobile Web Best Practices
Mobile widgets	Photoshop, HTML, CSS, and JavaScript	Opera Widget SDK, Nokia Web Runtime
Mobile web apps	Photoshop, HTML, CSS, and JavaScript	iUI, jQTouch, W3C Mobile Web App Best Practices

Designing for the Right Device

The truly skilled designer doesn_t create just one product—she translates ideas into experiences. The spirit of your design should be able to be adapted to multiple devices. What device suits this design best? What market niche would appreciate it most? What devices are the most popular within that niche? The days of tent-poles are gone. Focus instead on getting your best possible experience to the market that will appreciate it most. It might not be the largest or best long-term market, but what you will learn from the best possible scenario will tell you volumes about your mobile product_s potential for success or failure.

This knowledge will help you develop your porting and/or adaptation strategy, the most expensive and riskiest part of the mobile equation. iPhone users consume more mobile content and products than the average mobile user. This platform has an easy-to-learn

framework and excellent documentation, for both web and native products, and an excellent display and performance means. Although iPhone users might not be the majority of your market, the ability to create the best possible design and get it in front of those users presents the least expensive product to produce with the lowest risk.

With a successful single device launch, you can start to adapt designs from the best possible experience to the second best possible experience, then the third, and fourth, and so on. The best possible experience is how it should be, so it serves as a reference point for how we will adapt the experience to suit more devices.

Designing for Different Screen Sizes

Mobile devices come in all shapes and sizes. Choice is great for consumers, but bad for design. It can be incredibly difficult to create that best possible experience for a plethora of different screen sizes. For example, your typical feature phone might only be 140 pixels wide, whereas your higher-end smartphone might be three to four times wider. Landscape or portrait? Fixed width or fluid? Do you use one column or two? These are common questions that come up when thinking about your design on multiple screen sizes. The bad news is that there is no simple answer. How you design each screen of content depends on the scope of devices you look to support, your content, and what type of experience you are looking to provide. The good news is that the vast majority of mobile device screens share the same vertical or portrait orientation, even though they vary greatly in dimension.

References

1. Brian Fling, “Mobile Design and Development”, First Edition, O’Reilly Media Inc., 2009