UNIT IV

The Mobile Ecosystem

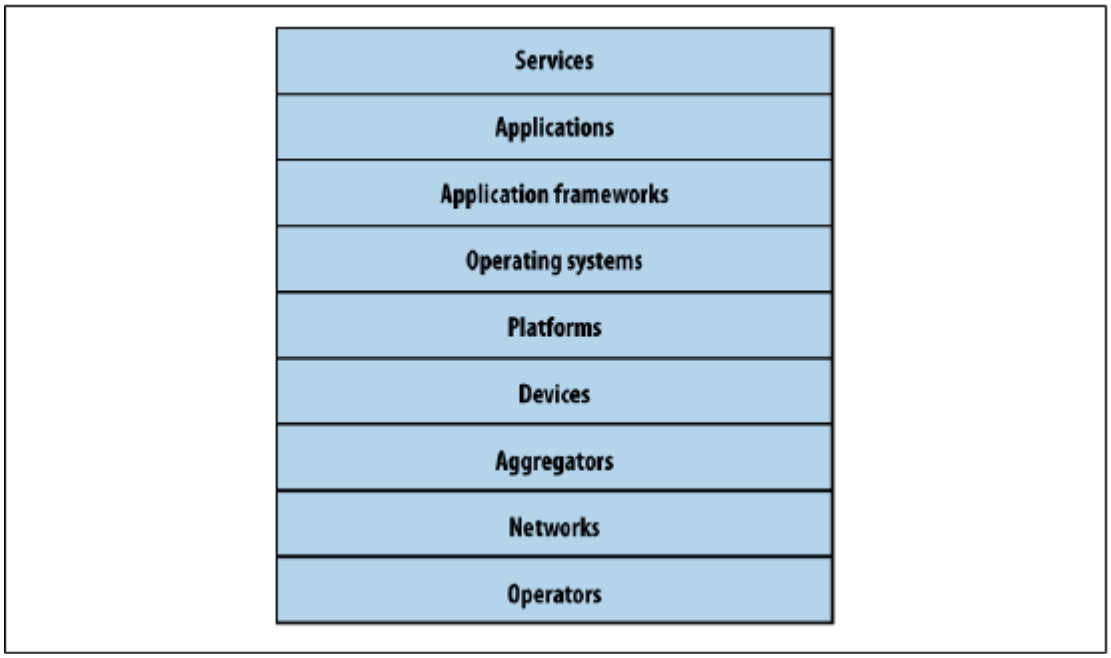


Figure 2-1. The layers of the mobile ecosystem

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.

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.

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.

Devices

What you call phones, the mobile industry calls handsets or terminals. 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.

Operating Systems

Operating systems often have core services or toolkits that enable applications to talk to each other and share data or services.

Examples
Symbian, Windows Mobile, Palm OS, Linux, Mac OS X, Android

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.

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)
Java Micro Edition (Java ME)
Formerly known as J2ME, 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.

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.
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.
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.

Application Frameworks

Often, the first layer the developer can access is the application framework or API released by one of the companies mentioned already. The first layer that you have any control over is the choice of application framework.

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.

Most Java applications are purchased and distributed through the operator, but they can also be downloaded and installed via cable or over the air.

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.

However BREW applications must go through a costly and timely certification process and can be distributed only through an operator.

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

With Palm's introduction of webOS, a mobile platform based on WebKit, and given its predominance as a mobile browser included in mobile platforms like the iPhone, Android, and S60, and that the vast majority of mobile web apps are written specifically for WebKit, I believe we can now refer to WebKit as a mobile framework in its own right.

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.

Applications

Application frameworks are used to create applications, such as a game, a web browser, a camera, or media player. Although the frameworks are well standardized, the devices are not. The largest challenge of deploying applications is knowing the specific device attributes and capabilities.

For example, if you are creating an application using the Java ME application framework, you need to know what version of Java ME the device supports, the screen dimensions, the processor power, the graphics capabilities, the number of buttons it has, and how the buttons are oriented. Multiply that by just a few additional handsets and you have hundreds of variables to consider when building an application. Multiply it by the most popular handsets in a single market and you can easily have a thousand variables, quickly dooming your application's design or development.

Services

Finally, we come to the last layer in the mobile ecosystem: services. Services include tasks such as accessing the Internet, sending a text message, or being able to get a location—basically, anything the user is trying to do.

Types of Mobile Applications

1)Mobile Web Widgets

A mobile web widget is a standalone chunk of HTML-based code that is executed by the end user in a particular way.

Basically, mobile web widgets are small web applications that can't run by themselves; they need to be executed on top of something else.

Opera Widgets, Nokia Web RunTime (WRT), Yahoo! Blueprint, and Adobe Flash Lite are all examples of widget platforms that work on a number of mobile handsets

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.

2)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.

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.

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 and 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, filesystem access, camera, and so on.

Native Applications

The next mobile application medium is the oldest and the most common; it is referred to as native applications, which is actually a misnomer because a mobile web app or mobile web widget can target the native features of the device as well. These applications actually should be called "platform applications," as they have to be developed and compiled for each mobile platform.

These native or platform applications are built specifically for devices that run the platform in question. The most common of all platforms is Java ME (formerly J2ME). In theory, a device written as a Java ME MIDlet should work on the vast majority of feature phones sold around the world

In the smartphone space, the platform SDKs get much more specific. Although many smartphones are also powered by Java, an operating system layer and APIs added

In addition to Java, other smartphone programming languages include versions of C, C++, and Objective-C

Creating a platform application means deciding which devices to target, having a means of testing and certification, and a method to distribute the application to users.

Because platform applications sit on top of the platform layer, they can tap into the majority of the device features, working online or offline

However, if you exclude games, the majority of native applications in use today could be created with a little bit of XHTML, CSS, and JavaScript

Pros

The pros of native applications include:

• They offer a best-in-class user experience, offering a rich design and tapping into device features and offline use.

• They are relatively simple to develop for a single platform.

• You can charge for applications.

Cons

The cons of native applications include:

• They cannot be easily ported to other mobile platforms.

• Developing, testing, and supporting multiple device platforms is incredibly costly.

• They require certification and distribution from a third party that you have no control over.

• They require you to share revenue with the one or more third parties.

3)Games

The final mobile medium is games, the most popular of all media available to mobile devices.

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.

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.

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.

Mobile Information Architecture

What Is Information Architecture?

The mobile information architecture definition outlines the following:

• The structural design of shared information environments

• The combination of organizations, labeling, search, and navigation systems within websites and intranets

• The art and science of shaping information products and experiences to support usability and findability

• An emerging discipline and community of practice focused on bringing principles of design and architecture to the digital landscape

Information architecture, as it is often used as an umbrella term to describe several unique disciplines, including the following:

1. 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.

2. ***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.

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

4. 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.

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

Mobile Information Architecture

For example, if we look at the front page of http://www.nytimes.com as seen from a desktop web browser compared to how it may render in a mobile browser (Figure 7-2), we see a content-heavy site that works well on the desktop, and is designed to present the maximum amount of information above the "fold" or where the screen cuts off the content. However, in the mobile browser, the text is far too small to be useful.

The role of a mobile information architect would be to interpret this content to the mobile context.

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.

Keeping It Simple

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

1. **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.

For example, to get some news and information on a mobile device, you need to first ask what the goal is. What is the need you are trying to fill? Then you need to apply context. Where are your users? What are they doing? Are they waiting for the bus? Do they have only a minute to spare? Or, do they have five minutes to spare? With these answers, you get your information architecture.

2. **Clear, simple labels**

Good trigger labels, the words we use to describe each link or action, are crucial in Mobile. Word s like "products" or "services" aren't good trigger labels. They don't tell us anything about that content or what we can expect. 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.

Keep all your labels short and descriptive, and never try to be clever with the words you use to evoke action. The worst sin is to introduce branding or marketing into your information architecture; this will just serve to confuse and distract your users.

Based on what we know from web design, you should use simple, direct terms for navigating around your pages rather than overly clever terms. That latter typically result in confused visitors who struggle to find the content they are looking for. When that happens, they will go elsewhere to look for the information they want. So, if you apply these same mistakes to a constrained device like mobile, then you end up adding confusion to the user experience at a higher magnitude than the Web.

Site Maps

The first deliverable we use to define mobile information architecture is the site map. Site maps are a classic information architecture deliverable. They visually represent the relationship of content to other content and provide a map for how the user will travel through the informational space.
Mobile site maps aren't that dissimilar from site maps we might use on the Web. But there are a few tips specific to mobile that we want to consider.

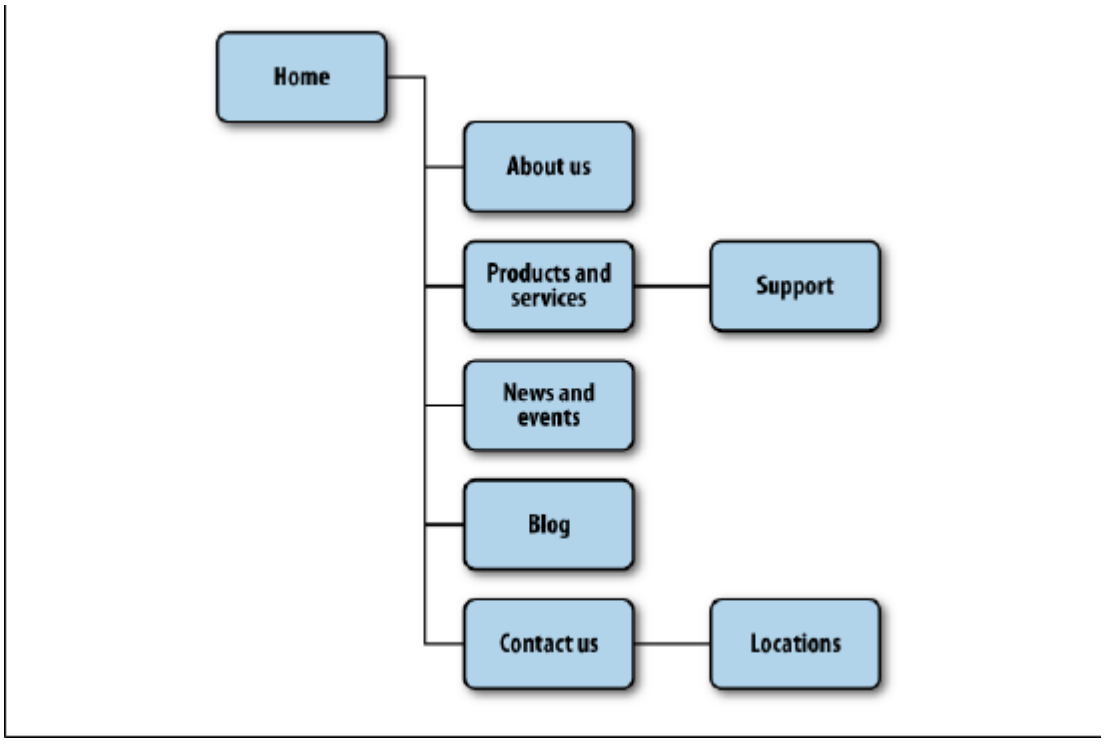


Fig: An example mobile site map

1. **Limit opportunities for mistakes**

Imagine a road with a fork in it. We can go either left or right. The risk that we will make the wrong choice is only 50 percent, meaning that we have a better than good chance that we will get to where we want to go. But imagine three roads. Now our chances have dropped to 33 percent. Four roads drops your chances to 25 percent, and five roads takes you down to 20 percent. Now a 20 percent chance isn't great, but it isn't too bad, either.
Now think of your own website. How many primary navigation areas do you have? Seven? Eight? Ten? Fifteen? What risk is there to the users for making a wrong choice? If they go down the wrong path, they can immediately click back to where they started and go down another path, eliminating the wrong choices to find the right ones. The risks for making the wrong choice are minor.
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.
But in mobile, we cannot make this assumption. In the mobile context, tasks are short and users have limited time to perform them.

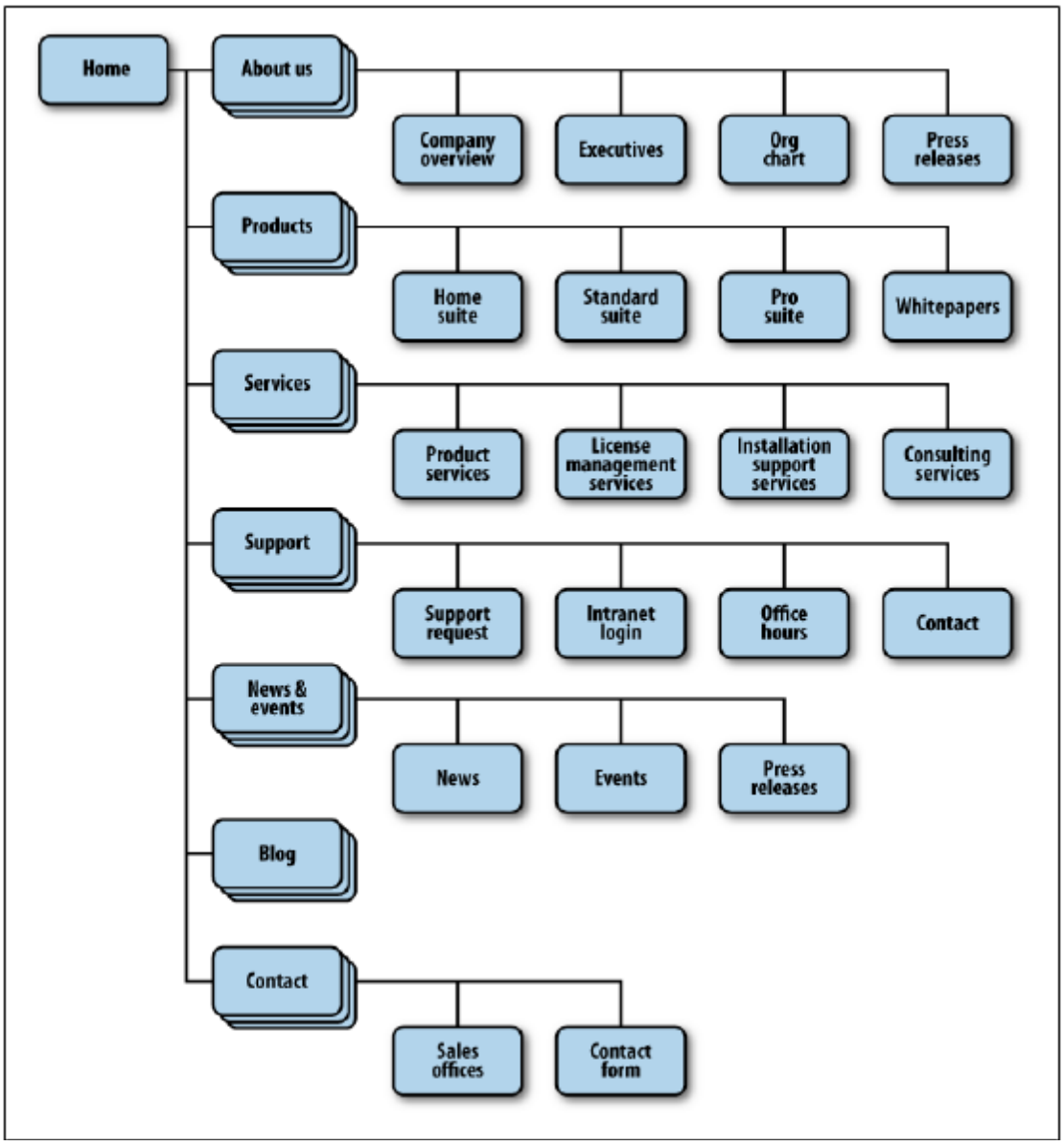


Figure . An example of a bad mobile information architecture that was designed with desktop users in mind rather than mobile users

2. Confirm the path by teasing content

After the users have selected a path, it isn't always clear whether they are getting to where they need to be. 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.

In order to make sense of a vast inventory of content, we have to group, subgroup, and sometimes even subgroup again, creating a drill-down path for the user to browse. Though on paper this might seem like a decent solution, once you populate an application with content, the dreaded "Page 1 of 157" appears. What user would ever sit there with a mobile device and page through 157 pages of ringtones? What user would page through five pages of content?

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 rearchitecting large websites.

A good architect's job is to create a map of user goals, not map out every technical contingency or edge case. Too often, process flows go down a slippery slope of adding every project requirement, bogging down the user experience with unnecessary distractions, rather than focusing on streamlining the experience. Remember, in mobile, our job is to keep it as simple as possible. We need to have an unwavering focus on defining an excellent user experience first and foremost. Anything that distracts us from that goal is just a distraction.

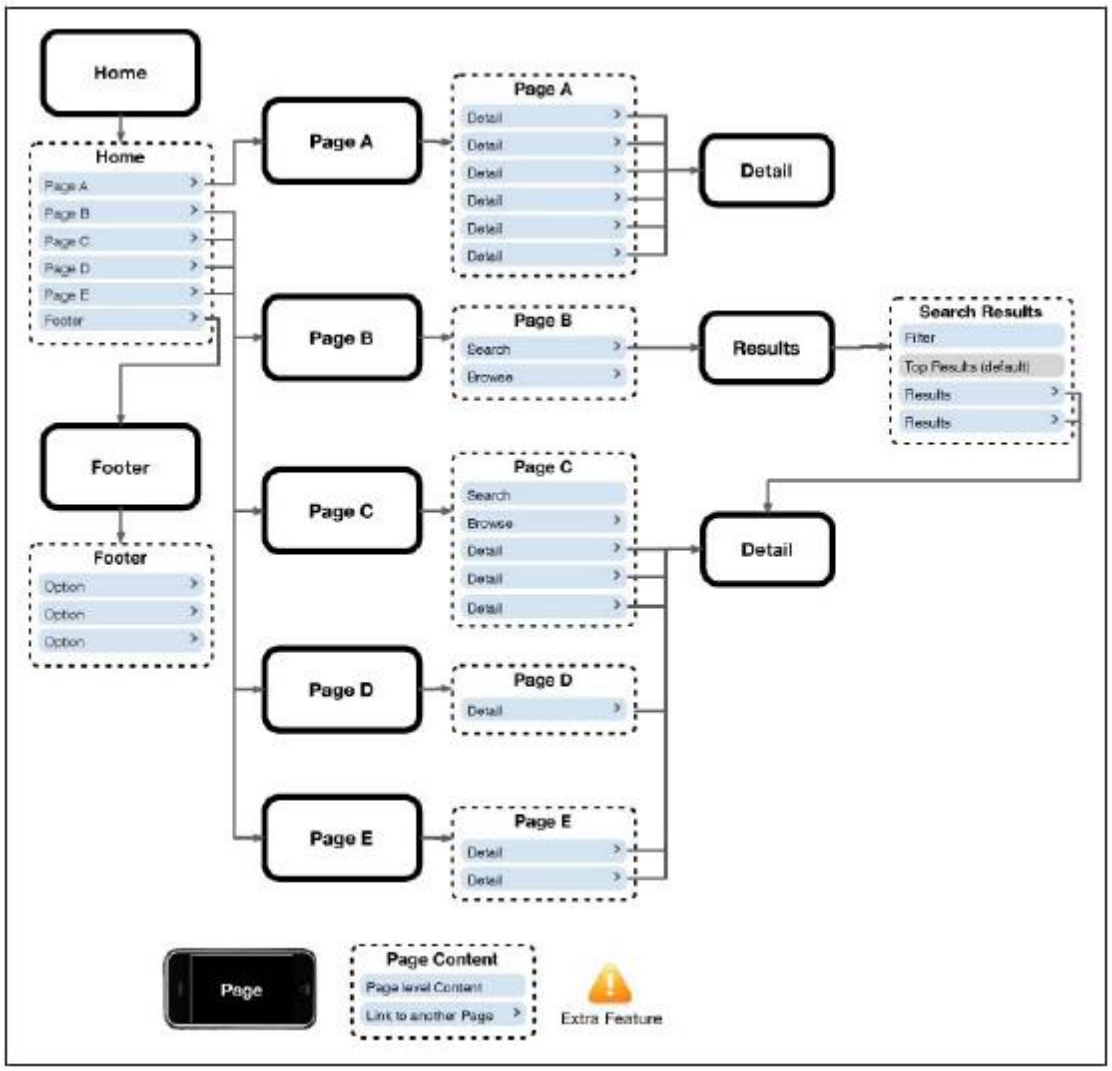


Figure , An example clickstream for an iPhone web application

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.

But the purpose of wireframes is not just to provide a visual for our site map; they also serve to separate layout from visual design, defining how the user will interact with the experience. How do we lay out our navigation? What visual or interaction metaphors will we use to evoke action? What are the best ways to communicate and show information in the assumed context of the user? These questions and many more are answered with wireframes.

Prototyping

The following sections discuss some ways to do some simple and fast mobile prototyping.

1. **Paper prototypes**

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

Create a basic script of tasks (hopefully based on either context or user input) and ask users to perform them, pointing to what they would do.

2. **Context prototype**

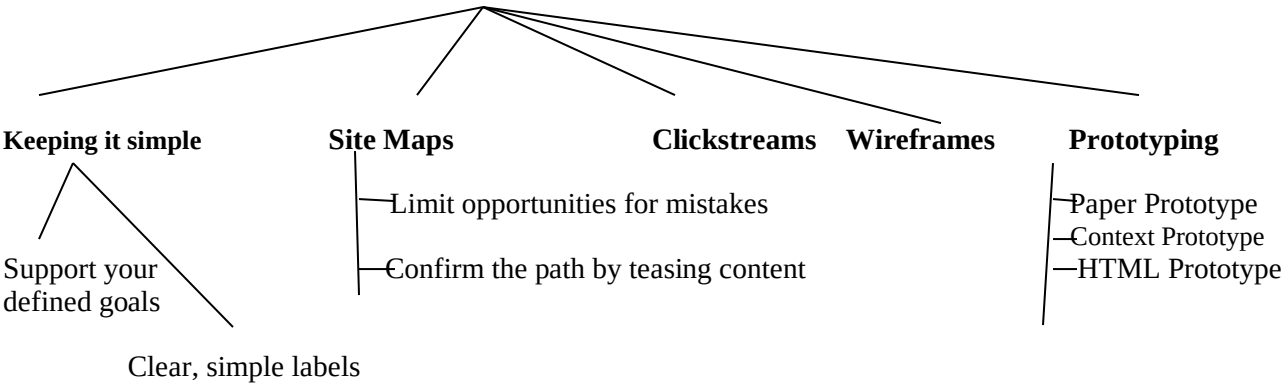
The next step is creating a context prototype (Figure 7-13). Take a higher-end device that enables you to load full-screen images on it. Take your wireframes or sketches and load them onto the device, sized to fill the device screen. Leave the office. Go for a walk down to your nearest café. Or get on a bus or a train. As you are traveling about, pull out your device and start looking your interface in the various contexts you find yourself currently in.

3. **HTML prototypes**

The third step is creating a lightweight, semifunctional static prototype using XHTML, CSS, and JavaScript, if available. This is a prototype that you can actually load onto a device and produce the nearest experience to the final product, but with static dummy content and data (Figure 7-14). It takes a little extra time, but it is worth the effort.

With a static XHTML prototype, you use all the device metaphors of navigation, you see how much content will really be displayed on screen (it is always less than you expect), and you have to deal with slow load times and network latency. In short, you will feel the same pains your user will go through.

Mobile Information Architecture



Mobile Design

The Elements of Mobile Design

1. Context

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.

2. Message

Another design element is your message, or what you are trying to say about your site or application visually. One might also call it the "branding," although I see branding and messaging as two different things. Your 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? Your approach to the design will define that message and create expectations.

For example, hold the book away from you and look at each of the designs in Figure ; try not to focus too heavily on the content. What do each of these designs "say" to you?

Figure 8-4. What is the message for each of these designs?

Which of the following designs provide a message? What do they say to you?

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 is improved even more if you scroll down). Words you might use to describe the message: minimal but messy, product-heavy, and disorienting

3.Look and Feel

The concept of "look and feel" is an odd one, being subjective and hard to define. Typically, 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?

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.
Although a lot of elements go into making Apple's App Store successful, the most important design element is how it looks and feels. Apple includes a robust user interface tool that enables developers to use prebuilt components, supported with detailed Human Interface Guidelines (or HIG) of how to use them, similar to a pattern library. This means that a developer can just sit down and create an iPhone application that
looks like it came from Apple in a matter of minutes. During the App Store submission process, Apple then ensures that the developer uses these tools correctly according to the HIG.

Figure :. Pattern Tap shows a number of user interface patterns that help to establish look and feel

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.
Why define the layout before the mobile 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. There is no right answer—only opinions and gut instincts. Plus, in corporate environments you have internal politics you have to consider, 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.
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.

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.

Different layouts for different devices

The second part of layout design is how to visually represent content. In mobile design, the primary content element you deal with the is navigation. Whether you are designing a site or app, you need to provide users with methods of performing tasks, navigating to other pages, or reading and interacting with content. This can vary, depending on the devices you support.

Figure :. Using a low-fidelity wireframe to define the layout design element before visual design Begins

There are two distinct types of navigation layouts for mobile devices: touch and scroll. With touch, you literally point to where you want to go; therefore, navigation can be anywhere on the screen. But we tend to see most of the primary actions or navigation areas living at the bottom of the screen and secondary actions living at the top of the screen, with the area in between serving as the content area,

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 in black and white. 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.

Figure . An example of different levels of posterization that can occur across multiple device color depths

Different devices have different color depths. In Table 8-1, you can see the supported colors and a few example devices.

Bit depth	Supported colors	Description	Example devices
12-bit	4,096 colors	Used with older phones; dithering artifacts in photos can easily be seen.	Nokia 6800
16-bit	65,536 colors	Also known as HighColor; very common in today's mobile devices. Can cause some banding and dithering artifacts in some designs.	HTC G1, BlackBerry Bold 9000, Nokia 6620
18-bit	262,144 colors	Used in mobile devices to offer Truecolor (see following entry) levels through dithering. Limited banding may be seen.	Samsung Alias, Sony Ericsson TM506
24-bit	16.7 million colors	Also known as Truecolor; supports millions of colors and produces little banding.	iPhone, Palm Pre, Nokia N97

Table : Supported colors and example devices

The psychology of color:

People respond to different colors differently. It is fairly well known that different colors produce different emotions in people, but surprisingly few talk about it outside of art school. Thinking about the emotions that colors evoke in people is an important aspect of mobile design, which is such a personal medium that tends to be used in personal ways. Using the right colors can be useful for delivering the right message and setting
expectations.

For the purposes of reference, Table provides some of the characteristics of various colors that naturally evoke certain emotions in people.

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
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

Table:. Color characteristics

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.

Three basic ways to define a color palette are:

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 meaning. The secondary and tertiary colors are often complementary colors that I select using a color wheel.

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, 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.

6. Typography

The sixth element of mobile design is typography, which in the past would bring to mind the famous statement by Henry Ford:

Any customer can have a car painted any color that he wants so long as it is black.

Traditionally in mobile design, you had only one typeface that you could use , and that was the device font. The only control over the presentation was the size.

Subpixels and pixel density:

There seem to be two basic approaches to how type is rendered on mobile screens: using subpixel-based screens or having a greater pixel density or pixels per inch (PPI).

A subpixel is the division of each pixel into a red, green, and blue (or RGB) unit at a microscopic level, enabling a greater level of antialiasing for each font character or glyph. The addition of these RGB subpixels enables the eye to see greater variations of gray, creating sharper antialiasing and crisp text.

The second approach is to use a great pixel density, or pixels per inch. We often refer to screens by either their actual physical dimensions ("I have a 15.4-inch laptop screen") or their pixel dimensions, or resolution ("The resolution of my laptop is 1440×900 pixels"). The pixel density is determined by dividing the width of the display area in pixels by the width of the display area in inches. So the pixel density for my 15.4-inch laptop would be 110 PPI. In comparison, a 1080p HD television has a PPI of 52.

Readability:

The most important role of typography in mobile design is to provide the user with excellent readability This can be done by following these six simple rules:

1. *Use a high-contrast typeface*
2. *Use the right typeface*
3. *Provide decent leading (rhymes with "heading") or line spacing*
4. *Leave space on the right and left of each line; don't crowd the screen*
5. *Generously utilize headings*
6. *Use short paragraphs*

7. Graphics

The final design element is graphics, or the images that are used to establish or aid a visual experience. Graphics can be used to supplement the look and feel, or as content displayed inline with the text.

Iconography:

The most common form of graphics used in mobile design is icons. Iconography is useful to communicate ideas and actions to users in a constrained visual space.

Photos and images:

Photos and images are used to add meaning to content, often by showing a visual display of a concept, or to add meaning to a design. Using photos and images isn't as common in mobile design as you might think. Because images have a defined height and width, they need to be scaled to the appropriate device size, either by the server, using a content adaptation model, or using the resizing properties of the device.

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.
In below Table , you can see each of the design tools and what interface toolkits are available for it.

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

Table : Design tools and interface toolkits