

Advances in Mobile User Input: Touchscreens and Beyond

Abstract

A wide variety user input technologies are becoming available for mobile phones. Touch screens are becoming prevalent, but also camera, voice, NFC, haptics, sound, and more. Many of these technologies are old, but advances in processing power is making them feasible for use in mass-market mobile phones as input methods. The best user experience will be found by combining input technologies into a context-aware natural user interface.

Early in the technology cycle, these technologies primarily benefit the device manufacturers, who create enhanced experiences for individual devices. Later, they become part of the technology culture and provide opportunities for third party providers, further driving adoption for manufacturers.

If managed correctly, new input technologies can drive device adoption, increase service adoption, and create new business opportunities. The best commercial success will be found by making input methods somewhat standardized and accessible to third party developers.

/Personae Dramatis **Delete for final production**

We're writing for these folks.

- Dr. Inventor. ("I've got this great technology to change the world")
- Immersion Vice President. ("I have a somewhat mature company working on changing the mobile industry.")
- Samsung product manager. ("I've got to get devices to market, quickly, meeting my operator-client's needs.")
- Opera product manager. ("I'm building a platform to give all users a great web experience.")
- Verizon product manager. ("I've got to decide whether to bother with this latest Next Great Thing for my products.")

Table of Contents

/Introduction.....	4
/The Mobile Industry.....	4
/Mobile Phone.....	5
/How to Copy.....	5
/Types of Mobile Input.....	8
/Understanding the Mobile Input Ecosystem.....	9
/Carriers (Operators).....	9
/Manufacturer-Carrier Relationships.....	11
/Chip Makers.....	12
/Technology and Platform Providers.....	13
Input Technology Life Cycle.....	17
Stages.....	17
Accelerators.....	20
/Case Study: Fastap	21
Touch Screens, Evolving.....	24
Technologies beyond touch.....	28
/Text Input.....	28
Speech.....	36

Gestures.....	36
Context Indicators.....	39
Risks and Opportunities in Mobile Input.....	41
Web Site Interaction.....	41
Context Engine (input ecosystem).....	41
Differentiation and Standardization.....	42
Other Risks and Opportunities.....	43
Implications on User Interface Design.....	44
User Interface Paradigms.....	44
Cut Stuff.....	45

/Introduction

For years, mobile phones have lived in the shadow of their computer brethren. Low memory, battery, and processing capabilities meant that only very simple input and output technologies were available. Evolution of these technologies has followed a predictable path of minor improvements, such as adding graphics or color to displays.

Occasionally, disruptive input technologies have been introduced. Predictive text input such as Tegic T9, for example, made triple-tap text entry on 12-key phones a better experience. But there have been far more good inventions that have not made it into devices. Many have been quite good, better than the standard technology. Why haven't these inventions been adopted?

Input techniques go beyond just text input. Cameras, microphones, and GPS receivers input information from the environment. Most implementations, however, use them for single applications: taking pictures, making voice calls, and maps and directions. Why aren't they used more widely?

More importantly, what can an organization that wishes to introduce a new technology, or leverage current technologies in different ways, do to effect change in the mobile ecosystem?

This paper addresses these questions.

/The Mobile Industry

The mobile industry in general has been resistant to input innovations, particularly in the U.S. and Europe where the industry structure is more conservative. Device manufacturers do not sell to consumers, but to conservative mobile operators (known as carriers in parts of the world). Operators, in turn, have to manage the complexity of many types of devices in their network. An inventor must convince a buyer at the mobile operator to then convince the manufacturer to implement a change. This is a very difficult process for anything that is expensive or risky.

In places where the industry structure is less conservative, the innovation story has been different. Japanese operators fully specify the phones in partnership with the manufacturers, thus reducing the risk for the operator.

Now technologies such as global positioning systems (GPS), camera as sensor, Bluetooth for file exchange, radio frequency identification (RFID) chips, and accelerometers are entering the market place. Which ones will succeed? Which ones will be only on a few devices? How do the devices' software, web browser, and applications change? Does it matter? This paper explores these questions in the lens of the evolving landscape of Apple's iPhone, Google's Android, and even Nintendo's Wii.

/Mobile Phone

Throughout this document we use the term “mobile phone” loosely. Certainly the issues in this document apply to all pocket communications devices, those that are carried with the user most of the time.

The ideas in this paper may also be applicable to domains such as portable gaming devices. We do not expect them to be applicable to more specialized information devices like dedicated cameras or music players.

For example, the multifunction iPod Touch is covered, the Sony PlayStation® Portable may be covered, and the Amazon Kindle is not covered.

/How to Copy

There is a danger in imitating any one aspect of any successful product, service, or initiative from one's competitors. Imitations of this nature almost never succeed. Imitating the iPhone's user interface without creating the content ecosystem (iTunes Store) and hardware platform results in a product that falls far short of expectations and merely whets users' appetites for the real thing.

A successful initiative or product is a collection of strategies, processes, execution, and visible artifacts. What will be copied by competitors are the artifacts; the strategies and processes that are not visible to outsiders will not.

Of course the artifacts of a service are not copied directly. Not only is it unethical, but the product team will want differentiation from the competition, and the development team can deliver different things than the competitor's development team.

The best approach is to simulate the processes that created the great experience, and not try to copy the experience itself.

Right now, companies are replicate various aspects of the iPhone. Typically they reproduce parts of the App Store, many of the pretty graphics, and the capacitive touch screen.

The iPhone represents a significant advance in mobile phone design: the multi-touch capacitive screen, excellent graphics processing, accelerometer, good user interface, and integration with iTunes and Apple software products. This device turned the mobile industry upside down.

What should be replicated from the iPhone is not any particular technology, nor its "skateboard" slab form factor, nor its capacitive touch screen, nor its multi-touch technology, nor its App Store. It is instead the fanatical devotion to excellence in user experience. The resulting product may very well have any or all of the above functionality, but they it will be a coherent user experience.

/What End Users Want From Mobile Devices

What do end users want? They want novelty, fashion, and status. They want ruggedness, battery life, and connection. They want tools to help them accomplish their key goals. Increasingly, they will want entertainment. They will want their unique needs met, or they will want a tool to accomplish their absolute most necessary tasks at the lowest possible price.

Key Point: Expect an array of form factors, functionality, and input mechanisms to meet various users' needs.

User needs will shift as technology shifts. As new functionality is incorporated into mobile phones, people will see opportunities to merge devices, change how they currently use devices, or make tasks more convenient.

What form will this take? Don't expect it to stay limited to the current set of form factors. Many manufacturers are experimenting with watch-phones; RIM is experimenting with a watch accessory to a phone. A pen as a phone is in the realm of possibility, especially with a rollable display, projected display, or other accessory.

/Current Segmentation Is Artificial

Will these new forms be smart phones? What does that even mean?

The typical market segmentation is as follows:

- Dumb phones, used for voice and text only. These are the devices users are most likely to leave in the car in case of emergencies.
- Feature phones, used for voice, text, email, camera, music, web, and other features.
- Smart phones, which allow users to add their own programs.

There are several problems with this segmentation. In particular, the distinction between feature-phone and smart-phone moves regularly becomes blurred. What makes a phone smart and why would a customer care? Isn't the ability to take and send good pictures pretty smart?

As of 2009, many feature phones had better Internet browsers than many smart phones. For example, the Obigo, Opera Mini, Bolt, and Openwave browsers all render web pages faster and better than Pocket Internet Explorer.

The definition of "smart phone" is one that is heavily debated within the mobile industry. Blackberries seem to occupy a space between smart and feature phone. When the iPhone was initially launched, it could not add new programs. Nokia has been pushing the Series 60 smart phone operating system further and further down into mass market devices, increasing the official smart phone penetration.

The Garmin-Asus nüvifone G60 has Garmin-quality mapping and directions, music, but has no ability to add applications. Anything (other than more maps) not initially built into the device can not be added. Thus this "high end" device is actually a feature phone; one with a good touch screen, advanced location input, and the form of a smart phone.

The purchase of a smart phone does not mean its "smart" features are used. A huge number of Windows Mobile devices are used exclusively for email, text, and voice. Blackberry is built on the same triad of features.

In short, users do not care about operating systems. They want appealing forms, features that let them accomplish their goals, and a good deal.

New features, including new input technologies, will not necessarily enter the market from the high end, or even from smart phones. The smart phones' very need to be a general computing device reduces their flexibility. Even this may change as new display technologies, such as projector displays, get adopted.

/Phones Like Cars

Many Apple fans believe that all phones will become like the iPhone: a slab device with few or no buttons and a beautiful user interface on a multi-touch display. Certainly most manufacturers copied the basic form factor in the years following the iPhone launch. Nevertheless, other form factors continue to be important.

Consider the automobile market. A "perfect car" might come out; what does it look like? A Toyota Prius? A BMW 3 Series? A Ford Expedition? A Eurovan? Each of these are excellent vehicles, catering to different goals, contexts, and driving behaviors. But I wouldn't want to drive a Prius off-road, nor an Expedition down narrow European streets.

Similarly a slab-phone with a pure touch interface is not for everybody.

/Types of Mobile Input

User input techniques can be categorized by interaction types:

1. Clicking or tapping, including touch and buttons
2. Text input, whether via speech or tapping
3. Decoded speech, both for text input and for command and control
4. Gestures, via touch screen, accelerometer, camera, and more
5. Context, such as location (All of this information is already present. The trick is to get the device to detect it and make sense of it, and then let users interact better.)

/Understanding the Mobile Input Ecosystem

The typical inventor would love to see their wonderful new invention win, simply because it is better. Not only does that not always happen in the wider world (Betamax is technically superior to VHS yet still lost in the market), it especially does not happen in the mobile industry.

To make progress, our inventor and his investors must understand the entire mobile industry. Its structure affects both what is possible and what is practical in adopting new mobile input technologies.

The mobile application value chain includes the user, the device in the user's hand, the technologies on the device, the connectivity to the web, any services enhancing the connection, applications and web sites, and distribution of applications and web sites. Making all this happen are device manufacturers, carriers, technology platform providers, application and content developers, and content distributors. The power structure amongst these entities varies across continents and is frequently problematical.

In addition to the players in this section, organizations such as content providers, software and hardware vendors, government entities, and standards organizations are present. This section outlines key influencers in mobile input.

/Carriers (Operators)

Mobile carriers (or operators) provide the connection to the customer. They purchase phones in large numbers. A operator can order phones customized directly for them. Indeed, companies like Vodafone, Orange, Verizon Wireless, and Sprint all have hundreds of pages of requirements for the types of phones they buy. The greatest operator control over phone design is found in Japan, where the operators have historically designed the entire phone and invited manufacturers to make it or not. Small operators must accept or reject the phones that manufacturers offer, so the range of control varies wildly.

Key Point: Operators are large risk-averse organizations who have massive buying power for mobile phones. The needs of large operators dictate what gets built.

Operators also have a close connection to the user in the form of billing. Most operators have enabled adding a variety of third-party services directly to the monthly bill, ranging from media purchases like ring tones, to sodas from a vending machine, to utility bills. This allows small purchases to be aggregated into one payment, which makes users less sensitive to the overall amount and thus more likely to purchase more. Also using the device's account for purchases reduces the amount of data entry compared to entering credit card data.

The operator's key concerns are simple:

- Any device that connects to its network must not jeopardize the network.
- Any service must not jeopardize the network or the relationship with the operator's customers.
- Churn, a term that represents the percent of an operator's user base who leaves in a given month, needs to shrink (churn rates of 1.5% per month are common).
- Monthly revenue, typically measured in Average Revenue Per User (ARPU), is a key measure.

Operators are likely to have significant inertia in decision making if those decisions do not impact these key concerns.

/Operators and Their Devices

Many carriers subsidize the customers' purchase of new equipment. This subsidization makes their customer acquisition costs quite high, and results in a difference in carrier and manufacturer goals. Thus when a user purchases a subsidized handset, the carrier has already made a significant investment in that user. This fact, coupled with the operators' concern about churn, led to the requirement that users stay with the carrier for a certain number of months or face a penalty.

The carrier then makes revenue based on monthly use, with greater profits for high voice use, high text or data use, and services. Unique services can make the carrier more "sticky", and increase the end users' switching costs. Ease of use and overall user experience have a significant effect on how often the user uses the carrier's services. In contrast, device manufacturers simply want repeat sales. Ease of use is less important to them, but manufacturers like Nokia have recognized that predictability of use will increase repeat sales. As a result, the core Nokia user interface has remained the same for a decade¹.

¹ Expect this to change with their investments in Maemo, Trolltech and QT, and touch.

This set of conflicting revenue streams suggests a much greater motivation on the part of the carrier to have better usability of the device. Indeed, more and more carriers are starting to focus on consistency of user experience across devices - with varying degrees of success.

/Manufacturer-Carrier Relationships

Manufacturers who prefer to take leadership roles tend towards GSM carriers; manufacturers who tend towards a contract manufacturing model tend towards CDMA and Japanese carriers. Smaller GSM operators are faced solely with a "take it or leave it" attitude from large manufacturers like Nokia and Sony Ericsson.

Selling a new technology to a manufacturer frequently involves selling it to the carriers first. A new browser, text input software, or keypad would have to be approved by the carrier and specifications written before the manufacturer would do more than create a test mockup to investigate the challenges associated with the new design.

Manufacturers face the challenge of tight margins. Carriers negotiate low prices per unit, so unit costs become critical. This problem has driven some manufacturers to avoid operating systems like Symbian, which can cost five dollars per unit in license fees. Samsung attempted to write their own browser to avoid browser licensing fees. Few manufacturers will invest in a clock chip, or indeed any piece of hardware that will not increase the sale price of the device.

Carrier power varies across markets. In markets such as Europe, where devices can be used with multiple carriers by a simple card swap, the device manufacturer has a much closer relationship with the end user. In markets such as Japan where the device can only be used with one carrier, the carrier has a stronger relationship with the end user. The result is that European carriers tend to have less influence over device design. Smaller carriers may have no influence other than deciding whether to allow a given device onto their network. Nevertheless, when asked who their mobile operator is, many Europeans will report "Nokia".

The carriers in Japan are extremely powerful: in addition to providing network service, they specify device design, develop network services, and develop new network standards. Device manufacturers can decide whether to make a device specified by the carrier, but have little influence over its design.

The North American market has both models. GSM carriers tend to get phones available in Europe, and have historically had limited influence over their design. CDMA carriers such as Verizon, Telus Mobility, and Sprint had the opportunity for more influence, especially over Japanese and Korean manufacturers' designs. Verizon was extremely slow to take advantage of this, but Telus Mobility and Sprint have a long history of working closely with manufacturers on device design.

/Chip Makers

The brains of any electronic device can be made purely with off-the-shelf resistors, capacitors, inductors, and transistors. However, the resulting device would be enormous and slow.

Chips enable miniaturization and hence speed; functions not possible through hand-wiring become possible on small scale. Chips also radically lower the price per unit and the engineering effort for the device manufacturers.

Key Point: If an input technology can be integrated into the mobile phone's chipset, operators and manufacturers no longer have to worry much about price.

While input technologies can be added to individual devices, the integration process can be complex and require separate pieces of hardware within the device. This is possible, but expensive in both time and cost per unit. Chips bundle this into a single piece of intellectual property.

When the chip makers add in features or support for hardware into the GSM and CDMA chips, the integration process becomes smoother. Code may not have to be written for implementing a feature on the chip, when it did for building a phone from multiple components.

The chip makers face the same problem as input providers: they must sell to manufacturers and operators. Operators have to want the features offered; manufacturers want reliability and fast time to market. A winning chip will pack a lot of features and reduce integration and testing costs.

The mobile phone chip industry is rapidly shifting. No longer is chip manufacturing limited to the likes of QUALCOMM, Texas Instruments, and Nokia; now computer and computer chip companies are entering the fray. Competition is increasing.²

As a result, chip makers are embedding more and more technology into the chips. QUALCOMM bought SnapTrack and embedded their assisted global positioning system (A-GPS) technology directly into the chips that the manufacturers purchase.

/Technology and Platform Providers

Platform providers develop the technologies upon which applications run. These providers are in the challenging position of convincing carriers, device manufacturers, and application developers to simultaneously adopt a technology. Sun and Macromedia (now Adobe) have done this by leveraging developers from the desktop environment. QUALCOMM has done this with its BREW development platform by providing a complete ecosystem and a development environment built directly on the chip set. Google has done this with Android by making their application stack open and by their sheer size. Securing adoption of a platform requires years of effort.

Relevant types of platforms include web browsers, widgets, applications, and operating systems.

/Relevance of Platform Design

If an input method is not available to the browser and to the application environment, it won't be used. For smart phones where much of the value of the device is in the browser and applications, this means the input method essentially isn't there. The value to the end user is reduced; the value to the operator is reduced.

For example, the infrared sensor in the iPhone is used to turn off the screen when answering a call as the device gets close to the phone. The sensor is only available to developers as on/off, which makes it not useful for most interactions. As a result, the infrared sensor is not used in applications and the user doesn't know it is there.

Predictive text input and alternate text input methods suffer from this problem. For years the Java ME (then known as J2ME) platform did not provide methods to switch text

² As of November 2009, QUALCOMM is providing a chip for a Nokia-AT&T phone. This is a first, and suggests they will compete for sales for all technologies, not just CDMA.

input: all typing was triple-tap. On some devices, the text input method was whatever the user had last used elsewhere on the device. On other devices, switching methods could be used, but only by pressing the * key ... something many users will never discover because it is buried in the user manual. Java applications were much harder to use as a result.

Even virtual keyboards had this challenge. Windows Mobile supported alternative keyboards, but Palm and other platforms did not. Keyboard makers had to convince companies to make their somewhat unusual keyboard be the default keyboard.

Handling alternate text input methods is relatively straightforward, and the device manufacturers worked hard to make Java and web browsers work well. But Java ME should have provided a method to handle text input methods from the start.

Many input methods, such as gestures, speech, and context, are much harder to incorporate. Where text input interfaces only with specific form fields, these other methods potentially interact with the entire content.

/Browsers and Web Standards

The browser is relevant not just for itself, but also for the rendering engine it supplies to any widget engine also on the device. Key rendering engines are Webkit (Apple, Android, Symbian, some Blackberries, and others), Presto (Opera), and Gecko (Firefox).

A browser handles two types of user input: actions to manipulate the browser, and actions to manipulate the content. The browser passes through the second type of action to the content itself. The first type of input can be used to zoom, scroll, go back, and other key actions.

Several browsers support gestures for browser actions, but none support gestures, voice, or context beyond location to interact with the actual content. Accelerometers, for example, are not accessible from the browser (as of 2009). GPS is similarly not available. The BONDl effort by the Open Mobile Terminal Project at <http://bondi.omtp.org/> is an effort to expose several device capabilities to web pages, though mostly local storage and context indicators.

The Worldwide Web Consortium (W3C) is working on standardizing gesture and speech interaction. Their efforts tend to trail technology creation, but facilitate widespread adoption. See <http://www.w3.org/standards/webofdevices/> for an overview.

/Application Environments and Operating Systems

Operating systems and application environments do not rely on the W3C to enable new functionality, but they do have the problem of providing a relevant set of functionality to application developers, and how to do this that facilitates both ease of development and a good end user experience.

/Impact on Developers and End Users

Consider the accelerometer, and the user action of shaking the device. Apple could have used one of three approaches when developing the iPhone application platform:

1. Reserve the shake gesture for built-in applications only.
2. Define a shake event, with a very small number of variants. Code would simply “listen” for a shake event the way it might listen for a tap at a certain coordinate.
3. Allow developers more direct access to the accelerometer, and have them define anything they want for detecting device movement.

Apple went with the second and third options, providing developers enormous flexibility. They could define not just a shake event, but any type of event the device could physically detect. Sweeps, flicks, shakes, and tilts are all available.

Going with only the third decision would create greater complexity, as developers have to figure out what all the above means in actual code. Sample code found on the net suggests ten lines of code or so to detect the shake event.

The third option also presents usability issues. The way one application responds to a shake event is not the way another application would. There is a minor problem for developers as well: while emulators can readily trigger a “shake event”, they can not readily emulate a customized gesture with required time, position, and duration. Thus these applications are harder to test.

/Application Environment Support is Good for Business

Good support of an input method in a popular operating system or application environment will go a long way towards making that input method popular.

Consider compass information, which works with the location to determine not just location, but what direction you are pointing. This was a technology used just for built-in navigation applications, and then only lightly. End users would typically consider compass technology irrelevant for a device purchase.

Then a relatively unimportant phone with an important operating system was released with compass technology. The HTC G1 Android, with a good development environment and application distribution system, was enough to get developers experimenting with augmented reality.

When iPhone 3GS was announced, these developers created entire companies. Now both iPhone and Android support interesting augmented reality applications, revealing significant information about the surrounding world. There are applications that reveal the local subway lines and where stations are, superimposed on the little corner grocer you are looking at. Other applications provide recent reviews for the restaurant you are looking at.

The Layar application provides an entire augmented reality platform, with all sorts of data possible.

None of this would have been possible without the operating systems revealing compass data to developers. And the compass technology has become something smart phone purchasers look for in their new phones.

Input Technology Life Cycle

New input technologies follow a predictable path through time, and each stage has its risks, techniques, and opportunities.

(think webcams, touch, multi-touch, camera phones, GPS, compass)

A framework for understanding new UI technologies.

1. Technology innovation
2. Alternatives created
3. Slow adoption
4. Reference commercial implementation
5. Copycat implementations
6. Standardization
7. Mass adoption, movement into web and cross-device platforms

Note operator adoption can be inserted in a few places.

Two entrance paths: new technologies (GPS, accelerometers, eink) and new procedures

Stages

1. Technology Innovation

Three major categories of invention: lone inventor, serial inventor, corporation.

Somebody invents a new technology to revolutionize the mobile industry. In many cases, a lone inventor has genuinely created something new and compelling, and he expects that all he needs to do is to show the device manufacturers his new idea and they will start building it, giving him money.

More successful innovations are paired with a true business model and business.

Strategies

Many lone inventors make the naïve mistake of approaching operators and manufacturers directly, thinking that the companies will be interested in licensing this terrific technology and commercializing it.

Must form a company and develop the idea as a product. Must productize the idea. Manufacturers and chip makers more likely to buy the company than the product.

For software-only inventions, develop the product as an add-on.

For hardware-only, see if you can make an accessory as a proof of concept. Benefit: you get revenues sooner.

2. Alternatives Emerge

See the gesture-on-touch-keyboard, below. Example, Swype. Same idea, a few implementations. They're fighting against each other.

Strategies

Avoid comparing your great input idea to your competition. As far as your customers are concerned, you are competing against do-nothing. “A rising tide raises all ships.”

Create a reference design for your solution.

Build your product and business in conjunction with business development efforts.

Get a trial with a small operator. They may be more flexible!

3. Slow Adoption

Some mild success is had. Maybe operators

Strategies

4. Reference Commercial Implementation

(e.g., iPhone multi-touch)

Strategies

5. Copycat Implementations

with widely varying UX

Strategies

6. Standardization

Strategies

7. Mass Adoption, Movement Into Web and Cross-Device Platforms

(think augmented reality, include Flash & Java)

Strategies

Accelerators

While many things can slow down progress through the life-cycle, some can really accelerate.

Application development ecosystem

insanely popular device

/Case Study: Fastap

Fastap is an alternative keypad that allows a large number of large buttons in a small area. Every letter in the English alphabet, plus all the standard buttons of a 12-key keypad, get their own button. It results in faster typing and greater efficiency than a typical QWERTY keypad, let alone triple-tapping on a 12-key.

Here's Fastap's story.

/Stage 1: Innovation

In 1993 Apple alumni and inventor David Levy³ filed a patent for a new type of mobile phone keyboard. His invention is very efficient, using two levels of buttons and some clever software to enable buttons for every letter in the alphabet, as well as every number, in the space of a normal 12-key pad.



Over the next few years, the patent was awarded and some mobile phone manufacturers became interested. He received some money to work out a fully functioning prototype.

By 1998, things looked bright. The patent had been issued the previous year, and while Ericsson was concerned about the amount of change the keyboard represented, Motorola was paying Levy a fee while they performed due diligence. By 1999, a contract was being negotiated.

1999 saw a setback: Motorola re-organized, and the new manager had not even heard of the keyboard, let alone recognized the partially-negotiated contract.

Levy approached various operators, including Sprint PCS where I met with him. He had started to recognize that many times manufacturers won't add something without the operators' wanting or requesting it. At the time, the keyboard was not ready for production devices; it needed quite a bit more work. As the operator, we asked for some specific improvements to bring it up to professional quality.

³ Read more about Levy's invention story at http://money.cnn.com/magazines/fortune/fortune_archive/1999/03/29/257409/index.htm

To be honest, we weren't thinking about the whole picture. We were suddenly presented with an inventor with an idea; it looked pretty good but we couldn't use it yet. We sent him off to fix it. We did not nurture Levy in any way, as we were too busy. What would have happened if we had the time to invest in these ideas? Perhaps the idea would have taken off. But at the time, we did not have a venture arm nor a strategic sourcing unit. Nevertheless, I was excited about the technology and followed its progress over the next several years.

/Stage 2: Alternatives Emerge

Fastap was not the only technology of its sort. During Motorola's due diligence process, two older, related patents were found. Levy found places where the technology did not overlap, and updated the patent.

Neither of those patents went beyond patent filing. Levy's was the version to go forward, because he (eventually) created a company with business professionals to commercialize his invention.

/Stage 3: Slow Adoption

In the next couple years, Levy started Digit Wireless to develop and deploy his keyboard, called Fastap. Instead of a brilliant inventor trying to sell the technology, business development professionals started working on appropriate demonstrations, deals, testing, and marketing.

The new company understood quite a bit more about commercialization, the process of converting a product idea and prototype into a sellable product. They also understood better how to work with operators and manufacturers. By 2003, Panasonic had released a mobile phone sporting the Fastap keyboard.

/Stage 4: Reference Commercial Implementation

By 2007, two smaller operators, Alltel in the U.S. and Telus Mobility in Canada, had released a Fastap phone by Korean manufacturer LG. The phones were a success, with users sending twice as many text messages⁴ as users of other phones. Things were looking bright, and the company invested in further product development.

⁴ See press release <http://www.highbeam.com/doc/1G1-130208605.html> and original Digit Wireless/Fastap release <http://www.digitwireless.com/flash/pdf/Telus-Data-Results.pdf>

Alas, Fastap's sad story ends here. Investors demanded that the company re-organize. The team who had found the company such success left, and Fastap only ever launched on five phones. QWERTY keyboards, despite being harder to use, are the expected implementation.

Digit Wireless⁵ is now selling TRACE, a virtual keyboard system involving tracing words on the screen. It directly competes with Swype and Shapewriter, and looks nearly indistinguishable.

Motorola and Sony Ericsson, the original manufacturers interested in Fastap, never did release a device. Nor did any major operator.

⁵ <http://www.digitwireless.com/>

Touch Screens, Evolving

Mobile phone companies, both operators and manufacturers, are scrambling to adopt touch technology for their mobile devices. Pundits claim that all smart phones will have touch input within two years.

While this may be true, the results may not be as intended.

Touch technology has largely followed the input technology life cycle. Researchers discovered how to do it in the early 1970's, and was first applied to consumer mobile devices in the 80's. The Apple Newton and the Palm Pilot were the first mass market personal information devices with touch screens, though a few others were available around the same time in the early 1990's.

Key Point: Touch screens are not the apex of mobile device input, nor do they create great products just by existing. Companies will do well to carefully select technologies to create a unified user experience. This may or may not include a touch screen.

Touch Screens' Appeal

- popularity, becoming required
- technology: types of touch: resistive, capacitive, etc.
- implications of design
- drawbacks, in general

Haptics

DEFINE

A key shortcoming of touch screens is that they demand eyes-on interaction. That is, there is no way to tell what is being manipulated without either looking at the screen or some extremely clever design.⁶

The solution to this is some combination of auditory and haptic⁷ feedback. Using haptic is far more natural and does not disturb others.

Haptic feedback is actually not an input method, but rather an output method. However, haptics enable several types of input methods possible or easier to use. For example, haptic feedback combined with a touch screen would make input more reliable.

To date, haptic feedback has largely addressed the problem of touch screens not providing the same tactile and haptic feedback as physical buttons. The physical sensations are transmitted using small vibrator motors in the device **

- haptics (vibrate, field, other; case study Apple haptic feedback for 3GS for blind user)

(3 maybe 4) Haptic feedback - to make touch screens have hardware button-like feel. This is installed on many devices <http://immersion.com/> but is still not standard.

Types

Vibration for edge detection - a mobile device does not just vibrate, but multiple motors work together to make specific spots on the screen feel like buttons. This is accomplished through wave cancellation and addition. To the user, an advanced implementation makes the edges of clickable elements feel clickable.

The Samsung TouchWiz user interface makes extensive use of haptic feedback.

Field effect - the surface creates an electric field, re

⁶ A Google engineer made a clever eyes-free shell for Android devices. The dialer makes the the 5 appear wherever the initial touch takes place, then all other numbers are in the predicted 12-key relative locations. <http://google-opensource.blogspot.com/2009/04/announcing-eyes-free-shell-for-android.html>

⁷ There is a technical distinction between “haptic” and “tactile” but it is irrelevant for this discussion, and largely unknown outside academic circles. We are using the two interchangeably here.

Force feedback and rumble -

Magnetic levitation - still in the very earliest phases and not yet for mobile, these systems currently use a bowl with electrical currents, and a mouse-like controller in the hand. The currents in the bowl provide a three dimensional space for the magnetic controller to navigate through. <http://butterflyhaptics.com/>

Companies

Immersion owns many of the patents in the haptic feedback, force feedback, and rumble; most commercial applications use Immersion technology.

Senseg is pioneering the field effect.

Butterfly Haptics makes magnetic levitation controllers and software.

Pressure Sensitivity: Degrees of Touch

When painting with a paint brush, you can use the entire brush, or theoretically just a single bristle. Indeed, Chinese calligraphy demands exactly this. When sketching with a pencil or pen, different surfaces and pressure can be used to extract different lines, different thicknesses, different character.

Current touch screens use only binary notions of touch: the touch happened at position (x, y). If the screen is multi-touch, then there might be two or three (or fifteen) sets of (x, y) coordinates. Not captured is measure of how hard the user is pressing, nor what part of the finger is being used, nor which finger.

Expect a linear progression of touch screens to incorporate a greater array of human touch expression. Perhaps, for example, the screen will tell the software not just (x, y), but (x, y, d) where d is the diameter. A large surface area could designate a general action, where a small surface area might suggest that a smaller target on the screen is being aimed for.

Lessons from Wacom

Wacom tablets, the pen-and-tablet combinations used by artists and designers to sketch in Photoshop and similar programs, incorporate this notion of multiple levels. Indeed, their tools have either 256 or 512 digitization levels available.

Wacom designers have focused extensively on the feel of the pen moving across the tablet. More advanced tablets advertise a “paper-like feel”. Paper, when drawing, has an intimate feel, a friction

Pressure Sensitivity

Pressure sensitivity is making its way into mobile devices. Nokia recently filed a patent⁸ for three-dimensional touch input; others are undoubtedly exploring the same.

Software Design Implications

⁸ <http://www.unwiredview.com/2009/10/22/nokia-is-exploring-3d-multi-touch-interfaces/>

Technologies beyond touch

All of the technologies in this document can be used in conjunction with touch and multi-touch, at least to some degree.

Touch is good for selecting, moving, and manipulating objects. It is not good for applying many different actions or tools to that selection.

Nor is touch good for eyes-free use. Haptic feedback will help, as will auditory feedback, but these are

This section contains mobile input technologies to extend beyond touch screens.

/Text Input

The challenges of entering text on a small device have been widely known for as long as text messaging has been around. As a result, hundreds of inventors have tried to solve the problem.

There are too many technologies to do them all justice, but Michael Longé of Designer Software and formerly of Tegic, has a wonderful essay titled “Nothing New Under the Thumb⁹” for more information. A note about mobile interaction in general, and text input in particular. Mobile devices can be designed for:

- Single-hand interaction, epitomized by Nokia Series 40/60
- Two-hand interaction, epitomized by the stylus interaction on the Palm, tablet PCs, and phones with side-slide QWERTY¹⁰ keyboards
- One-and-a-half-hand interaction in which most but not all key tasks can be performed with one hand, such as the iPhone and Blackberry
- Two hands and a surface, in which the device needs to be placed on a surface before being comfortably used, such as laptops

⁹ <http://www.designer-software.com/articles.html>

¹⁰ For the purposes of this document, QWERTY and the French AZERTY layouts are identical.

- One hand and a surface, useful for wearable computing, certain industrial contexts, some computing contexts, and many disability contexts. The key value is the remaining hand can be used for other tasks, such as mousing, writing, using a number pad, and so forth.

A well designed device remains consistent in its design. Text input mechanisms should be optimized for each category of use.

Some devices combine categories above. Most notably, several phones have either a 12-key keypad or a virtual keypad when closed, and a full QWERTY slide-out keyboard. These devices are one-handed in one mode, and two-handed in the other. The best of these devices also adjust the rest of the interaction for one-handed versus two-handed use.

/Physical Keyboards

Physical keyboards are clearly in stage 7, mass adoption. Alternative designs of physical keyboards have a lot of established competition, and are languishing mostly in stage 3, slow adoption. Some minor commercial success has occurred.

Physical keyboards have several advantages compared with virtual keyboards on touch screens. In particular, physical buttons' tactile characteristics allow touch typing to emerge. Button edges are obvious, and do not need the help of the eyes to find. Fixed button locations enable muscle memory, reducing the cognitive load of typing. The force feedback of pressing a button lets the fingers know when a button has been pressed enough to type a letter. All sensory input and feedback is missing by default in virtual keyboards.

The 12-key layout for most phones is efficient for one-hand use, and is especially easy for making phone calls. It is not optimal for entering text, requiring triple-tapping of keys to get letters or the assistance of predictive text software.

Alternate physical keyboard layouts have a long history. Originally, the QWERTY layout for typewriters was invented not to slow down typists, but to put frequent letter-pairs somewhat far from each other so the letters striking the page would not jam as often.

The typing speed for any pair of characters is a function of how they are distributed across the hands. The slowest combination of letters is two on the same finger, such as "JU". Next slowest is the same letter in a row, such as "OO". After that is a pair of letters on the same hand but different fingers, such as "LU". Fastest is a pair of letters on opposite hands, such as "TH".

The same action that reduced letter jamming in original typewriters, moving frequent letter combinations far from each other on the keyboard, actually sped up typists. Few keyboard layouts can enable faster typing. Dvorak, for example, is slightly faster for expert typists. But only slightly. In the meantime, the QWERTY layout is well known and has market saturation.

For full-alphabet mobile keyboards, QWERTY still reigns supreme. Not only is it well known, but the same layout that allocates frequent letter combinations to alternate hands in the full size version does the same in the mobile version, though typically only thumbs are used.

Device Use	Dominant Interaction	Promising Contenders
One hand	12-key (1-9, 0, #, *)	None expected. Fastap has best chance.
Two hands	QWERTY, especially slide-out keyboards.	None expected.
One and a half hands	Tiny QWERTY	Multiple input methods per device.
Two hands and a surface	Full QWERTY	None.
One hand and a surface	None; depends on context	Frogpad

Text input alternatives for physical keypads. Speech and touch not included.

In general, the entire alternative physical keyboard industry is languishing at Stage 2: Alternatives Emerge. At the same time, this is the space where most inventors spend their time, or did before the dominance of the iPhone. Now they are spending time on virtual keyboards.

Delta II from Chicago Logic¹¹ is an interesting approach to one-handed QWERTY. Even more than Fastap, however, they lack the business development money and approach to get the keypads on mobile phones. Further, the keypad hinders feature phones' dominant value of easily entering phone numbers; the number buttons are too small and can not easily be found without looking at the keypad. In short, operators' fears about the device are well founded.



Delta II keypad from Chicago Logic. Image copyright Dana Suess.

For one-hand-and-a-surface interactions, Frogpad is a reasonable contender. This keyboard methodology uses neither alphabetic nor QWERTY, but instead uses the same logic of typing speed to create a fast one-handed typing experience.

The value of Frogpad is twofold:

- its buttons are the same size as the buttons on computer keyboards, matching hand size well
- the most frequently used letters and functions are available with a single keypress

Like Fastap, Frogpad is still alive and developing alternatives. For example, a virtual keyboard version of Frogpad is now available, with the same button-size advantages as physical Frogpad.



Frogpad keyboard. Image courtesy <http://www.frogpad.com>

Unlike Fastap, Frogpad is targeted at niche markets. As a result, most people will never see a Frogpad keyboard ... and Frogpad is not targeting mobile operators or phone manufacturers. They will have an easier time building their company.

¹¹ <http://www.chicagologic.com/index.htm>

/Virtual Keyboards and Predictive Text

Windows Mobile and its predecessors have enabled third-party virtual keyboards to be loaded onto their devices since 1997¹². Most other device platforms have limited users to the built-in virtual keyboard, but Google’s Android allows the same type of keyboard substitution.

The fact that two popular platforms allow keyboard swapping has created a cottage industry for virtual keyboards. Despite this seeming wealth of virtual keyboards, these companies typically want to license their input technologies to manufacturers like Apple, Nokia, and Research in Motion.

The cost of entry in this space is quite low, unlike most mobile input methods. In theory, providers could release a product to the public and then license to the manufacturers. In practice, this rarely happens.

Virtual keyboards are obviously in all touch-only devices, and as such are at stage 7, mass adoption. Like physical keyboards, implementations have minor differences. The iPhone’s keyboard, for example, introduced the pop-up key to provide visual feedback that the correct key was selected.

Similarly, predictive text is nearly ubiquitous. The dominant players are T9 (now owned by Nuance) and eZiTap (now owned by Nuance), but several competitors also exist in the space. Older software is not really predictive text, but a program to disambiguate letter entry on a 12-key phone. That is, instead of typing “house” as 44,666,88,7777,33 the program would recognize “house” as the most likely match for 46873 and simply enter it. That’s a savings of 8 keypresses for this single word.

Later programs also include word completion, integration with virtual keyboards such as highlighting most likely letters, and even phrase completion. This is a stage 7 set of technologies, but directly impact the usefulness of virtual keyboards.

Many alternative keyboards exist, and are squarely in stage 3: slow adoption.

Device Use	Dominant Interaction	Promising Contenders
One hand	Virtual 12-key (1-9, 0, #, *) with predictive text	None expected; speech will likely replace this.

¹² See <http://www.hpcfactor.com/support/windowsce/> for an exhaustive history of Windows CE, including screen shots of each version.

Device Use	Dominant Interaction	Promising Contenders
Two hands	Virtual QWERTY	Compact QWERTY, word-drawing keyboards like Swype
One and a half hands	Virtual QWERTY	Compact QWERTY
Two hands and a surface	Full physical QWERTY	i.Tech Virtual Keyboard

Text input alternatives for virtual keypads. Speech and touch not included.

/Single Finger

Some makers have redefined how keyboards are arranged and behave. Two makers have attempted single-hand interaction. Despite the obvious value of this interaction, it has been abandoned by the market. The learning curves for these single finger interactions are high and the speed isn't as fast as QWERTY. We should instead expect to have speech to accelerate single-finger text input.

One single-finger input product, Thumbscript¹³, mapped a gesture language onto the 1-9 numbers, with a simplified alphabet “drawn” on the nine keys. This was extended to use on arbitrary single-finger keyboard surfaces, as long as there is a center point, four edges, and four corners. Thumbscript is a good input mechanism for those with mobility problems, but most likely will remain a niche market mechanism.

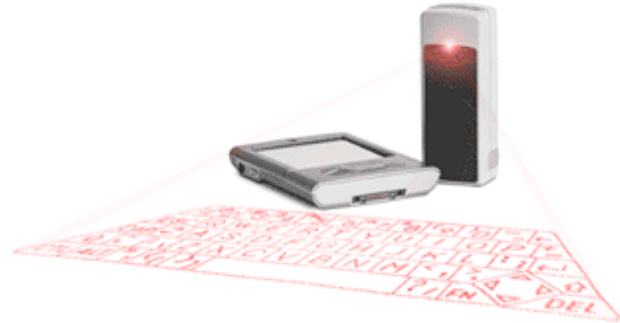
The Fitaly One-Finger Keyboard¹⁴ starts with a letter layout to minimize finger travel, rather than between-finger typing. As such, it is square, and the space key and letters N and E are in the middle. The second row of letters is F-I-T-A-L-Y, giving the product its name.

¹³ <http://www.thumbscript.com>

¹⁴ <http://fitaly.com/fitaly/ofkey.htm>

/Projected Keyboard

One virtual keyboard breaks the boundaries of the mobile device, and has many niche applications. The i.Tech Virtual Keyboard¹⁵ projects a full sized QWERTY keyboard onto physical space and is available as a separate unit. Expect this one to have small scale commercial success, and to compete with Frogpad. It is unlikely to be used while mobile, as it requires two hands and a surface to use.



i.Tech Virtual Keyboard.

/Compact QWERTY

Compact QWERTY keyboards exist in both physical and virtual forms. They represent a compromise between tiny buttons on a full QWERTY layout and the triple-tapping necessary on a 12-key. Both Research in Motion (Blackberry) and Nokia have released phones with compact QWERTY.

The concept is simple: take a QWERTY keyboard, and combine pairs of keys to make a larger key. There is therefore a QW key, a ER key, and a TY key. The resulting buttons are much larger, and easier to tap. Larger keys are faster to find than small keys, easier to learn with muscle memory, and more accurate. Touch typing becomes possible, unlike most virtual keyboards.

Combined with predictive text and word completion, compact QWERTY can be very fast with almost no learning curve. It has the combination of familiarity and device optimization that should make serious text entry easy.

Compact QWERTY virtual keyboards are available as an alternate keyboard for Android and Windows Mobile as the CooTek TouchPal and as an alternative layout for the Better Keyboard from Better Android Apps.

Nokia and RIM have already provided reference commercial implementations of this technology; some of the virtual keyboards are merely copycat implementations. This is

¹⁵ Details at http://www.vkb-support.com/learn_more.php

a technology squarely in stage 5. We anticipate virtual compact QWERTY default installation on some devices as early as 2010¹⁶.

/Word Drawing

The final type of virtual keyboards to watch out for involve eliminating the need to tap on keys entirely. Instead, the user slides the finger from letter to letter in the word, never picking up the finger. Predictive software converts the trace, essentially a drawing, into a word.

This seemingly unique space has at least three current contenders: Swype, Shapewriter, and SlideIT. Swype and Shapewriter are the ones to watch.

Shapewriter and SlideIT are leading with the installable keyboard approach, with Shapewriter far in the lead with Windows Mobile, Android, and even Windows keyboards. Shapewriter even has an iPhone application for taking notes, with the idea that you can take notes quickly and then paste the text into other programs. Shapewriter has created a business model that makes them successful at both the niche and the mass market levels.

Swype was invented by the creator of the original Tegic T9, who is working on replicating the successful T9 business model with the existing manufacturer and operator relationships. They have found some success here: the first Swype-enabled phone, the Samsung Omnia II, was launched in 2009. The company has also announced an Android installable keyboard for 2010, but this may be because they have arranged a deal to supply a default keyboard for an Android manufacturer.

Who has the best technology? The answer is uncertain.

Who has the best chance of mass adoption? Swype. The experience of getting T9 to market domination will be insurmountable. This technology will quickly gain reference implementation status (stage 4), and copycat implementations are already emerging (stage 5). Plus, Swype has the business capability to achieve standardization (stage 6).

From a user perspective, standardization may not happen. Word drawing is fun, but requires the user to change how she enters words. Pausing in the middle of a word will

¹⁶ CooTek launched a beta version of their keyboard technology on the Android market, for free. They then pulled the application, saying that users must wait for the “OEM version.” This suggests a OEM deal is forthcoming.

cause the keyboard to register the wrong word. For now, only the copycat implementations are readily testable, so perhaps Swype has a better user experience.

Speech

Scientists have been working on speech recognition for decades, with AT&T Bell Lab's research starting in the 1930's. The first commercial application didn't appear until the 1980's. By the 1990's, commercial products such as Dragon Naturally Speaking and IBM ViaVoice had enjoyed modest success, particularly in medical transcription and certain mobility-challenged communities.

By the turn of the century, computers had grown powerful enough to have "speaker independent" speech recognition, in which users did not need to spend 40 minutes training a computer to recognize their particular voice. By 2007, handheld devices could have speech recognition, but the technology was not then in high demand.

Gestures

Gestures are a major area of focus in changing human-machine interaction, but they are not tied to any particular technology.

Like speech

Device-recognized gestures differ from environment- or kiosk-recognized gestures in that only one person need be recognized and that person is likely nearby. The device moves. but the person does not move much relative to the device. further, the person is motivated to develop expertise in the system. Thus lessons from kiosk and ubiquitous computing and home environments should be carefully examined before adopting.

Types of Gestures

With Device

Manipulate the device in a set fashion causing the device to perform some action. Note that many of the above examples will not require any learning by the user; the device will just respond to unconscious cues.

Technologies: accelerometer, camera, light sensor, perhaps microphone, RFID, NFC

Examples:

- Move device to ear: answer incoming call, lock screen, and pause media playback
- Shake: clear text input
- Tilt: move to next image
- Upside-down + no light: lock screen and keys

With Device to Environment

Manipulate the device with respect to some object in the environment, such as a bar code or an NFC chip. Or a purse.

Technologies: accelerometer, NFC, camera

Examples:

Tap device to point of sale terminal: pay

Tap devices together: share item on screen

Detect purse RFID chip + no light: lock phone, turn ringer volume up unless on mute

With Device to Device

Manipulate two devices with respect to each other to create a shared experience

Technologies: accelerometer, location, Bluetooth or NFC

Examples:

Tap to share data

Multi-device pong or other game

“Pour” data from one device to another, such as [Siftables](#)

At Device

The purest of gestures, the person communicates with the device non-verbally, with no touching.

Technologies: camera, pressure sensors

Examples:

Peer more closely at device to increase text/image size

Close eyes pauses playback

Shaking fist at device

On Device

Gestures performed on the device include both the prevalent moving the finger(s) on the device through some path. This is currently the vast majority of gestures available, and is mentioned here for completeness.

Note that non-screen device features, such as pressure points or Palm Pre's gesture area, are also "on device". Similarly, Samsung is experimenting with pressure-points and [analyzing the user's hand grip to change UI](#).

Accelerometers

(6) Accelerometers - iPhone, Nokia, Android ... but only for rotating screen and games

gestures!

Sensors

- sound-as-context (e.g., recognize coffee shop based on sound)
- voice

- camera-as-input (e.g., facial recognition, facial expression recognition, bar code reader, Layar)

(3) Light sensors

(2) Voice command - mostly irrelevant voice dial for years, Google Voice and some iPhone stuff becoming more interesting.

(3) Fingerprint reader for device security/unlock. <http://www.gadget.pdamu.com/2009/08/16/sharp-kddi-e06sh-mobile-phone-with-fingerprint-barcode-reader-waterproof/> and <http://www.i4u.com/article1170.html> ... still marginal. Until they get into a smartphone, this is irrelevant. (because it's a corporate security feature as well as security nerd feature; this means smartphone)

Context Indicators

“Context Indicator” is a category, not described elsewhere, of implicit input mechanisms. A user does not directly interact with the device using one of these technologies, but the technology may affect the user experience nonetheless.

One of the earliest context indicators is detecting when the device is in its cradle. Some phones will change their ringer volume and change the default display when this happens.

Some Nokia devices have car mode, which simplifies the function of the entire device, increases display size, and makes the device easy to operate at full arm’s reach when the phone is in the car kit. Newer devices make similar transformations in navigator mode.

As devices become more sophisticated, so may the context available. Some actions may only be available in certain physical contexts, such as at a retail point of sale or in a movie theater. Security settings may be different at home rather than when the device is out.

Personal communications devices could evolve into constantly morphing tools that facilitate relevant actions at the right times, and provide not just-in-time but also just-in-place information.

GPS and Other Location Technologies

compass

gps

tower

Near Field Communications

Incidental Indicators

Finally, devices need not add new hardware to increase their awareness of context.

sound

wifi

Risks and Opportunities in Mobile Input

The reality of mobile phones

Web Site Interaction

Web interaction will lag, at the very time that mobile web is becoming prominent. Standards, sites, and technologies all assume keyboard, mouse or pointer, and a screen. Getting browsers to support a technology only on a small number of devices (e.g., in slow adoption phase) can be nearly impossible, and 99% of web sites will ignore the technology even if it is in the browser.

Mitigate with custom extensions, developer programs, working with W3C & OMA, custom browser, etc.

Key Point: The ...

Context Engine (input ecosystem)

Many of these technologies are good to help understand user context, and change user experience accordingly. Examples include:

- Keyguard turns on when device is upside down and in dark place (pocket), off when removed.
- While you are in a movie theater it would enable vibrate mode.
- In Starbucks the store's music selection becomes available.
- If music is present, now make music recognition available.
- At the airport, flight status could be on the idle screen.
- At bookstore the search defaults to book (and review) search.

Without an engine or platform to coordinate all these information and actions, each new input tech may run in a vacuum and not realize its full potential. The engine will

accelerate the tech curve for all technologies, even before standardization, by providing an API and fallback behaviors.

Differentiation and Standardization

Device companies fiercely compete on intellectual property and differentiation, and input methods are no exception. Apple's gestures, for example, are patented. Palm Pre implemented some of the same, and some different.

To a large extent, standardization between devices is not necessary (NOT SURE I AGREE... "HERE, BORROW MY PHONE" IS NOT THAT UNCOMMON IN THE REAL WORLD). Few people switch from one to another. But core interaction should be consistent. For example, pinch-to-zoom should be standardized to the point that it is present in all multi-touch devices and browsers, as it is not discoverable but it is core.

Early in the life cycle, few devices support a technology and it is hard to develop for. Adoption is limited to that device. If the tech is available to app developers, app development is easy, and the device is sufficiently popular, compelling apps appear. To facilitate this third-party app development, standardized version of a technology are useful. A possible exception if if you have a hugely popular device like an iPhone or RAZR and you also have a good development ecosystem like Andoird or iPhone - then you can skip the standards.

Consider the compass as an input technology. Available in Android and iPhone 3GS devices, third party companies are beginning to provide augmented reality applications like Layar and nearest Tube Station. These apps will drive adoption of Android & iPhone, and the compass will be adopted by other devices. The compass becomes a differentiator for devices and provides service opportunities for operators and developers.

Or, you can be an operator and spend millions of dollars to develop an innovative UI that runs only in your network and then watch your platform wither after 18 months. (Instinct, 3's cradle thingy, Celltop)

Other Risks and Opportunities

- smart phone vs. feature phone and differences in operator involvement & developer/app ecosystem
- multi-platform environments (web, Java, Flash) lag in access to things outside the sandbox
- discoverability - some iPhone users don't know pinch/zoom; more subtle gestures are lost entirely
- poor design or poor implementation sometimes makes entire technology have bad reputation, becomes irrelevant
- new input may require new UI paradigm. Voice UI != scroll and select UI != stylus UI != multi-touch != gesture. OS platforms need to support innovations in interaction without major delay in product launch.
- experiences will be inconsistent.
- new UI paradigms means new classes of system errors that can make it into production because they were not designed, developed, and tested for. For example, accelerometers might drain the battery because they are being triggered while walking.
- battery
- privacy

Implications on User Interface Design

User Interface Paradigms

Many of the above-mentioned technologies challenge well-established user interface paradigms. As a result, they require a high investment across the mobile value chain, from chip manufacturer to operator to

Now that mobile data access such as text messaging, applications, and browsers are becoming prevalent,

WIMPy

For decades, computer user interfaces have been based on a WIMP model: windows, icons, menus, and pointing device. Other systems, such as kiosks, used a

For over a decade, mobile phones have adopted a simple modification of the WIMP: no windows, and instead of a pointing device a highlight area controlled by a scroll button. While nobody called this is an IMP, that's essentially what it was. Single axis control (up and down) became two-axis control, but both were pointing devices and simple evolution. The stylus as an input method is still a pointing device.

Post-WIMP

In general, mobile phones do not use the WIMP paradigm. For example, they use a single screen interface; windows are a bad idea. Regardless, the fundamentals have been the same: use a pointing method to click on a graphic or text on the screen. Perhaps they are IMPs, but the idea is the same. Indeed, mobile phones have been modeled after desktops.

If desktop computers all use the WIMP paradigm,

Softkeys are an example of a non-WIMP

The Nintendo Wii uses

Similarly, text input has been largely some variation of the keyboard since the days of the typewriter. Full QWERTY, small QWERTY, Dvorak, the iPhone virtual keyboard.

Cut Stuff

Lessons form Abroad

Many people in the mobile phone industry are busily learning from the iPhone, and they are learning the wrong things. It is reminiscent of American auto executives in the 1970s visiting Japanese automobile manufacturers and believing the Japanese were hiding the rework portion of the factory. It is also reminiscent of the American mobile operator executives visiting DoCoMo to learn about the success of iMode and bringing home only the cHTML technology.

The Japanese auto manufacturers had re-engineered their entire production process to incorporate quality. Radical changes included giving every worker the power and responsibility to shut down the production line to address a quality problem. But the American executives just saw just-in-time delivery of parts to the factory.

Similarly, a group of American executives visited Japan to understand the success of DoCoMo's iMode, and missed the many cultural factors that made the service a success. Instead, they focused on the product itself. Note that iMode itself has exported only badly; some publishers are exporting their content to other countries with some success.

Similarly,

iPhone Usability

(Re: <http://blogs.oreilly.com/iphone/2008/11/an-interesting-study-on-iphone.html>)

One of the issues with iPhones is something that is an issue for all end users who are picking up a device for the first time:

Research showed that some of the tasks that experienced iPhone users would consider trivially simple were much more complicated for newcomers to the interface. Simple things, like the color of an app's icon when listed on the App Store, had a noticeable

effect on user behavior (red puts some users off, it seems, by making them think that a “red” app icon is some kind of signal to warn them away).

But outside of these basics for the new iPhone user, there are reasons why an iPhone is not for every user, and particularly not designed for specific end user groups.

1) The built-in keyboard for text input -- in either landscape or portrait mode -- is slow.

Writing using a pencil, pen, or stylus is one of the oldest forms of communicating an idea, and the use of Graffiti on the old PalmOS devices and on a newer application designed for the iPhone, Shapewriter, is faster than using the hunt and peck method.

2) Lack of a tactile keyboard makes use more difficult for the visually impaired and those who find keypads more responsive than a flat screen.

There is no raised surface, no keys available with Braille markers that can be custom locked to the device. Also, devices like Blackberry have keyboard shortcuts that are missing on the iPhone.

3) The new user has a steep learning curve about where to find tools which are easily found on other devices.

The new user has to learn the Apple iPhone paradigm from day one, and collect a set of applications to perform functions that are easier to find in other devices by default.

4) Battery life is much shorter on an iPhone.

Battery life is often drained by moving between wifi and 3G networks, and running multiple applications in state. Other devices use power based on one application in an open state in real time, leaving more battery life for the user. This life is improving with the latest 3G iPhone, but it is still less efficient compared to other devices.

Chip Companies Promote Customization For End Users

(Excerpt below taken from:<http://brainstormtech.blogs.fortune.cnn.com/2009/07/16/the-chip-company-that-dares-to-battle-intel/>)

Intel designs and builds all its own chips. It limits its product line to maximize efficiency and protect its profits. And along with Microsoft (MSFT), Intel is famous (or infamous) for using its power to essentially dictate how PCs function.

By contrast, ARM licenses its blueprints to all comers and leaves the manufacturing and sales to its partners. Since ARM doesn't manufacture, it can encourage customers to build whatever they like. That flexibility is especially valuable in the cellphone market, where handset makers often insist on custom chips and software to give them an edge in speed or battery life. "What ARM has done that I think is really good is similar to the strategy we have: It's to find a lot of partners," says QUALCOMM CEO Paul Jacobs, a customer and an admirer. "It's not just you as a company doing it all yourself. You leverage what you do by enabling a lot of other people."