Minimal Yet Integral – Designing a Gestural Interface

Martin Osen

Osen Design, Franz-Lehar-Straße 5,
4563 Micheldorf in Oberösterreich, Austria
martin@osen.at

Abstract. Minimalism and simplicity have become key success factors in the post-PC era. Touchscreens have superseded physical buttons as the dominant user interface of mobile devices. Some of the industry's most successful products tightly integrate hardware, software and services into one convenient solution. All this transformed the setting in which we are designing user experiences today. This paper describes the two-year development of a gestural user interface for a mobile app. Our design process can be broken down into five basic principles: Find a tangible metaphor, understand your hardware, care for your content, reduce it to the essence, and if you feel you can do better, iterate. Finally some yet unsolved issues are described that may impede the design of truly natural interfaces on a fundamental level.

Keywords: Design Philosophy, Minimalism, Mental Model Design, Metaphor Design, Gestural UI, Natural UI, Card-based UI, Smartphone, Tablet, Touchscreen, Casual Reading, Digital Publishing, Case Study.

1 Introduction: Beyond Flat Screens

A classic quote by Alan Kay suggests that people who are really serious about software should make their own hardware.[1] While that was certainly true in the early days of computing, surprisingly it seems even more relevant today. In my own experience as a designer, I frequently enjoyed projects the most where we could at least theoretically consider a holistic solution integrating software and hardware (e.g. IO Concept 2002 [1], Sony Spotlight Navigation 2004 [2], Sony Haptic Chameleon 2004 [3], Volkswagen Concept Study 2006). Recently, Alan Kay's quote has frequently been utilized in the context of big players pitching their vertical business models (e.g. Apple iPhone 2007, Amazon Kindle 2007, Microsoft Surface 2012).

As an interaction designer, the vast majority of my projects involves working on the software layer only. For obvious reasons, tinkering with the hardware is out of reach. It's simply not part of the job. Even in an ideal world it would not be: Time and money constraints aside it would probably not be desirable to reinvent smart phones over and over again every time an app developer is getting serious. I know plenty of interaction designers, software developers, startups and the like who are all really

[1] Alan Kay at a talk at Creative Think seminar, 20 July 1982.

A. Marcus (Ed.): DUXU/HCII 2013, Part III, LNCS 8014, pp. 383–392, 2013.

serious about their software, and yet they usually have to design it for a given hardware, which is always limited and sometimes just plain bad.

So in that typically limited context, how can we still think out of the box, "beyond flat screens"? How can we create an integral result if we cannot control the whole experience?

First and foremost, we need to deeply understand the hardware we are designing for: Their limitations, but even more than that their particular qualities. This understanding is of course not limited to a technological point of view. Social, philosophical and artistic perspectives are just as important. In metaphysical terms, we have to understand not just the body of a technical object, but get a feeling of its soul. It's the same kind of deep understanding that leads a sculptor to an idea of the sculpture already extant inside a raw block of stone. [4] The creative process is then just to remove any excess material.

2 State of the Art: Direct User Interfaces

Touch based user interfaces of smartphones and tablets have considerably changed the public perception of a "computer". Even to the untrained eye it is obvious that simplification and minimalism are no longer side issues discussed in circles of design enthusiasts only. They have become a mainstream factor driving current innovation. Unlike only ten years ago, today simplicity is key to commercial success.

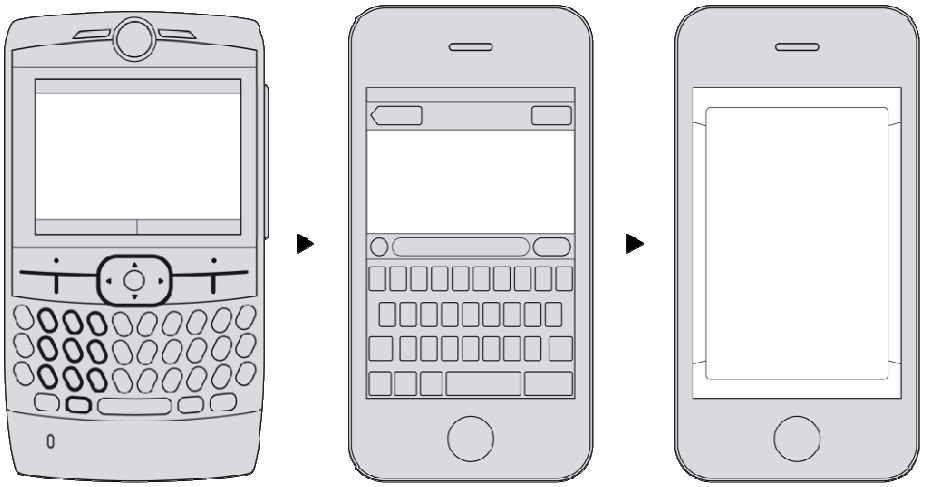

Fig. 1. On previous generation smartphones (e.g. a 2007 Motorola Q, left), casing, ornament and UI elements take up the greater part of the device's surface. In this example, only approx. 20% of it are left for actual presentation of content. The use of touchscreens (e.g. on an iPhone, center) can radically simplify the surface of a device and therefore free up space for content. However, depending on the amount of UI elements that are simulated on screen, the actual signal-to-noise ratio can be even worse. Direct, gestural interfaces (right) that depend less on visual chrome can – in some cases – reveal the true potential of current touch devices.

Touchscreens have replaced hardware buttons as the dominating user interface. In most cases though, the resulting clean and minimal look is only skin deep. The complexity of the user interface has simply moved to another layer: On screen we still interact with simulated buttons, knobs and switches. In terms of usability, a bad solution has arguably been replaced with one that's even worse.

One of the bigger challenges of the near future will be to replace those outdated, out of place paradigms by more natural, direct interaction with the content itself. The entire user experience has to deliver the simplicity that is promised by the minimalistic looks of current hardware.

Natural user interfaces are still in its infancy. Developing such interfaces today means taking part in a highly dynamic and evolutionary process. It should be clear that at the present state, failure and constant iteration are inevitable. The following chapter outlines selected problems that emerged during the two-year development of a gestural, direct user interface for a mobile application and discusses some of the short term and long term solutions we considered.

3 Case Study: CardSkid

CardSkid is a new digital-only publishing format that focuses on “casual reading”, “casual learning” and “casual exploring”. It enables publishers to quickly and efficiently build companion apps for their printed or digital books without any development effort. Content can be created, packaged and deployed using the newly developed standard CML (CardSkid Markup Language).

The end user gets a smartphone app that keeps the essence of a book in their pocket. Favorite quotes, lessons and key facts serve as teaser for a book that hasn't been explored yet. Once the book has been read, CardSkid keeps adding value by serving as an anchor to remember and further connect with the content.

A compelling user experience is crucial to the acceptance of such a publishing format.[2] The industry wide key topics of simplicity and minimalism lie at the very center of CardSkid's focus. It's all about distilling down to the essence. In essence, our design process can be described by five key principles:

3.1 Find a Tangible Metaphor

The metaphor of a deck of cards was the starting point for our design process. This was by far the single most important design decision. The card gave us something tangible, definite, rememberable. It transformed an abstract concept (a digital mobile publishing format for non-fiction literature) into something that could be easily grasped and discussed (a deck of cards in your pocket). From that moment on, every subsequent design decision followed a clear goal: The app should feel like a deck of cards in your hand. That's the soul of our product. Ideally, the user would not even be

[2] While this case study is mainly focusing on the user experience of the player app for iOS, this holds true for all steps of the CardSkid production chain.

aware of the interface or the hardware wrapped around our metaphor, and just be fully immersed in the content. [5]

While the card metaphor has been around since the early days of the GUI [6], we came to value it while fiddling about with real, physical cards. We learned that there is already a market for distributing analog content as cards. For some vertical markets like education, cards are a well-established form of distributing analog content, e.g. flashcards for learning foreign languages. There is ongoing research on how to bring such paradigms into the digital domain. [7]

Since we started development of CardSkid in September 2010, we kept noticing how several new or redesigned GUIs intended to simplify interaction by building on card metaphors: The product view in Amazon's mobile apps (2010), the card stack UI in HP's webOS (2011), or the Chomp-inspired redesign of Apple's App Store (2012), just to name a few. This reinforced our impression that we were on to something.

3.2 Understand Your Hardware

Together with the smartphone app we envisioned the prototype of a tablet app. The requirements for both user interfaces turned out to be completely different. The experience had to be true to the qualities of the respective form factor.

For creating the illusion of holding a real deck of cards, a touchscreen smartphone is the obvious choice, being roughly the same size and thickness. It also fits the bill of casual reading on the go: Take it out, shuffle through some cards, put it back.

On a tablet, due to its different ergonomics, it would have been impossible to recreate a similar feeling of holding a deck. While technically it was trivial to derive a tablet version by simply scaling up the UI to the bigger screen, it broke the whole metaphor, rendering the very foundation of the user experience meaningless. We came up with another paradigm instead: Whereas the smartphone should conceptually *be* a card (instead of *showing* one), the tablet is just that: A viewport to a threedimensional stage with cards floating in space. While the content *per se* always looks the same, the interaction paradigm adapts depending on its context.

Starting point of the tablet experience is again a deck of cards which is fanned out horizontally. A single card on the tablet corresponds in size approximately to a card on the smartphone. Different functionality (e.g. related to content acquisition, organization or annotation) is located in 3D space around the deck at the four cardinal directions. Rotating and tilting the viewport allows to zero in on a particular functional group while still keeping sight of the actual content. As for gestural control, various options have been considered. Basic on-screen swiping gestures to change the viewport appeared to be the most straight-forward solution. However, rotating and tilting the whole device might allow for an even more natural experience. Further user testing will examine whether the latter option also fares better in terms of usability.

3.3 Design Your Content First

It makes a huge difference if you deeply care about your content. In our case, we spent a great deal of the design process designing the look and feel of a card to get out of the content's way. We did this in part by deliberately defining what can not be done. For example, a card can not be scrolled. The boundaries of the screen limit the amount of content that can be shown. Design can be seen as a process of reducing options. What remains is the essence.

Fig. 2. Existing design, taken from printed cards (left) and minimalistic redesign optimized for on-screen and casual, mobile use cases (right)

What, in essence, defines a card? We determined three fundamental rules: A card has a defined size, capable of holding a limited amount of content, a card has a front and a back, and while self-contained, a card is in many cases part of a larger deck.

The main design challenge at this early stage was to digitally recreate as much as possible of an analog deck's user experience. For instance, how does it feel to shuffle cards? Which design parameters (e.g. shape, material, texture, physics, sound) are at hand to evoke that feeling? Most important, how can we ensure to keep the design honest, without reaching the point of adding visual noise and suffering from the negative effects of skeuomorphic design that are widely criticized? [8]

From day one of the design process, we explored our ideas on real content. Two authors allowed us to transform their printed cards into first digital prototypes.[3] Our starting point was a clean and minimalistic design which worked well for printed cards. However, for our purpose it still felt too heavy, cramped and distracting. In order to allow for a truly casual reading and learning experience, we favored an even

[3] Holzer, A.: "CoachingSet" and Ebertz, A.: "Re-Think!"

more minimalistic approach. What remained was pure visual information: Image, text, color. Using gradients and transparency, the design allows each element to take up maximum screen real estate while all elements can coexist in harmony. What is often referred to as "design" goes out of the way to an extent where the cards feel almost "undesigned".

3.4 The Content Is the Interface

In order to minimize visual noise, we initially favored a consistently chromeless user interface. A natural user interface should allow for direct interaction with the content itself. Therefore a set of multi-touch gestures had to be introduced and mapped to the required interaction outcomes. It became clear that there is no such thing as a natural gesture. Every gesture has to be learned, understood, remembered. Specification and standardization of "intuitive" gestures are widely unsolved problems standing in the way of truly natural user interfaces. [9]

Consistency is key to good usability. Unfortunately though, there are instances where any attempt to be consistent is physically impossible. In the realm of mobile computing, typically a number of different apps and services and, in many cases, even different devices and platforms are used concurrently. Even if we can achieve consistency within a single app, we are helpless against other entities that are inconsistent between themselves, be it incidentally or intentionally. This implies that we can only partially build on learned behavior by being consistent. At least at this current state, supporting the user in learning and understanding novel interaction paradigms is central to a usable product.

In order to flatten the learning curve, we opted for a pragmatic hybrid solution: We identified parts of the UI that benefit from novel gestural interaction, and other parts that gain efficiency by employing more established "standard" interaction.

Basic Navigation. Defining element on screen is always a card. A card displays key information on its front, and can be flipped over for additional content. Per default, cards of a deck are laid out horizontally and can be navigated through by performing horizontal swiping gestures. Groups of cards can be organized into stacks. Any card can be pushed up to reveal contextual actions located behind it.[4] Cards can generally be viewed zoomed in or zoomed out. It is of note that those views are non-modal, i.e. any interaction is available in either view. Either view serves a specific purpose: When zoomed in, no visual elements besides the card are visible. Nothing distracts from pure content ("content view"). When zoomed out, visual context like adjacent cards or icons representing core functionality comes into view, easing interaction especially for novice users ("navigation view").

[4] During development, parts of this particular interaction were replaced to be more consistent with standard iOS 6 behavior.

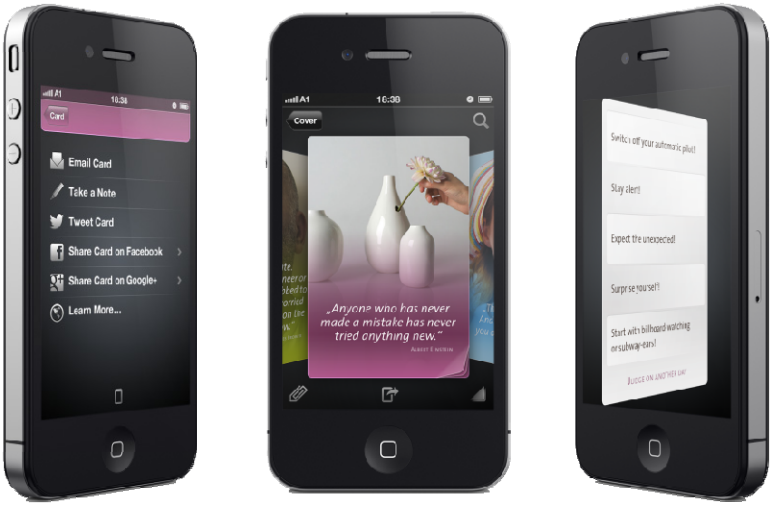

Fig. 3. Early 2012 prototype of CardSkid UI. The device in the center depicts a deck of cards zoomed out into an overview, providing additional visual cues for navigation. On the left screen, a card has been pushed up to reveal a list of contextual actions. On the right screen, a card is being flipped over to show additional content on its back.

Help. In a typical use case, we hardly have the opportunity to explain basic interaction techniques the way we did in the paragraph above. Still, if we rely on non-standardized interaction techniques and can not guarantee consistency with prior user experience, we have to educate the user in one way or another. The crucial question is when and how to provide instructions.

Basically, most current solutions can be subsumed under two strategies: Explain interaction before any issues can occur (often referred to as "tutorial") or provide selective support as needed after interaction came to a standstill (commonly referred to as "help"). In many cases, a combination of both strategies is used.

In a 2011 user study testing our prototype[5], the task of finding help was consistently challenging for users and therefore a suggested focus area for improvement going forward. We tried both aforementioned strategies, separate and combined, in total over a dozen different implementations. Our approaches included but were not limited to:

First, a tutorial explaining basic interaction at first start of the app. While a very common solution, the prominent position of the tutorial interfered with the user's urge to actually try the app and was often described as "pushy". In many cases, users skipped the tutorial entirely.

[5] In November and December 2011, a rapid iterative test of two versions of the CardSkid iPhone app was conducted in San Francisco. A total of nine users completed several exploratory tasks and six specific tasks and gave their feedback about the usability and overall experience of using the app.

Second, a separate stack of “tutorial cards”. This allowed the user to progressively learn how to use the UI by actually using it (“learning by doing”): The initial card would explain how it could be flipped over, the back would then explain how to progress to the next card etc. While this approach might be beneficial to some types of applications, we found no coherent way of connecting all gestures into one story.

Third, we started with another common solution: A button that turns on a modal help overlay. While easily understood, a permanent help button adds visual noise and takes up valuable screen real estate even when not needed. We developed a visually cleaner (and non-modal) alternative where the help overlay would be displayed while performing a “swipe down and hold” gesture. An animated help icon served as visual indicator that was only shown when necessary, that is only when we interpreted user behavior as uncommon or “helpless” (e.g. a rapidly repeating gesture that has no interaction outcome).

Getting back to our minimalistic design philosophy, we dispensed with a dedicated help feature altogether. Instead we introduced visual interface elements where they would improve usability, e.g. a dog-ear visual indicating that a card has a back. Full-screen “content view” still shows barely any chrome that would detract attention from the content. At all times, the zoomed out “navigation view” provides a more self-explaining fallback option.

3.5 Think, Test, Fail, Iterate. Think Again

A fair amount of pragmatism helps to achieve workable results. We found ourselves constantly oscillating between theoretical, pure concepts and hands-on testing. This approach proved to be beneficial, since conceptually perfect solutions tend to dash against an imperfect reality. On the other hand many of the usability issues that manifested during user testing first looked like a matter of fine tuning but actually arose from unsolved conceptual issues. We identified two main areas where gestural user interfaces on current multi-touch hardware fall short on a fundamental level:

Problem of Threshold. Multi-touch hardware allows us to interact via “natural”, analog gestures (e.g. a continuous swipe or rotation). However, many required interaction outcomes in software are still discrete and binary (e.g. opening a dialog or rotating the screen by 90 degrees). An interaction designer has to choose a threshold where a continuous gesture triggers a discrete result. Regardless of whether the dimension of the gesture is distance (e.g. how far to swipe), area (e.g. where to tap) or time (e.g. how long to hold), the threshold will generally be invisible and arbitrary to the user. Fine tuning of thresholds can ease usability issues, but will not solve them. Truly natural user interfaces require applications that are natural from the ground up. Pulling a natural interface over an inapt application can never be honest. Put another way, analog real-world metaphors may not apply well at all to abstract, “digital” topics.

Problem of Reversibility. In our tests users performed quite well in the task of navigating to a card at the lowest hierarchy level. Navigating back to where they left was in contrast found much more difficult. While the analogy of getting *into* a stack of cards by tapping it was easily understood and consistent with user's expectations, there was no clear idea how to reverse that operation.

Reversibility of operations is considered a defining factor of direct user interfaces. [10] While most of today's basic multi-touch gestures are easily reversible, there is one striking exception: The most common and maybe most “intuitive” gesture, a single tap, has no obvious counterpart. While multi-touch gestures are movements along the x/y-axis at the surface of the screen, a tap can conceptually be seen as a movement along the z-axis: Tapping *into* the screen typically goes a level *deeper* (e.g. open a stack of cards, or jump to the next level of a hierarchical list). A natural counterpart like “pulling out” of the screen would be desirable but is not feasible on current multi-touch hardware. Once we understood how this was in essence a hardware limitation, we accepted that we would not be able to solve this problem with a natural gesture. After months of hesitation, we finally introduced a back button.

4 Conclusion

Designing usable and satisfying user experiences today means to strive for concistency in a largely inconsistent environment. In this paper our design methodology is described by five key principles. In addition we identified two fundamental conceptual problems currently standing in the way of designing natural user interfaces for multi-touch hardware.

Acknowledgements. This paper is based on a project supported by the “Pre-Seed program” of the Austrian Wirtschaftsservice Gesellschaft (aws), “Academia plus Business” an initiative of the European Union and “Go-Silicon Valley” a program of the Austrian chamber of commerce. Without the deep appreciation for design driven development that both CardSkid founders David Schwingenschuh and Hannes Schmied brought into the project this paper would not have been worth writing. We further like to thank our content partners and study participants.

References

1. Osen, M.: Model of A Universal Interface For Mobile Communications. Micheldorf (2002)
2. Rapp, S., Michelitsch, G., Osen, M., Williams, J., Barbisch, M., Bohan, R., Valsan, Z., Emele, M.: Spotlight navigation: Interaction with a handheld projection device. In: International Conference on Pervasive Computing, Vienna (2004)
3. Michelitsch, G., Williams, J., Osen, M., Jimenez, B., Rapp, S.: Haptic chameleon: a new concept of shape-changing user interface controls with force feedback. In: CHI 2004 Extended Abstracts on Human Factors in Computing Systems, Vienna (2004)
4. Zöllner, F., Thoenes, C., Pöpper, T.: Michelangelo. Taschen, Cologne (2007)
5. Csíkszentmihályi, M.: Flow: The Psychology of Optimal Experience. Harper and Row, New York (1990)
6. Kahney, L.: HyperCard Forgotten, but Not Gone. Wired (August 14, 2002), `http://www.wired.com/gadgets/mac/commentary/cultofmac/2002/08/54365`

7. Edge, D., Fitchett, S., Whitney, M., Landay, J.: MemReflex: adaptive flashcards for mobile microlearning. In: 14th International Conference on Human-Computer Interaction, San Francisco (2012)
8. Poole, S.: Against Chrome: A Manifesto (2011), http://www.3quarksdaily.com/3quarksdaily/2011/02/against-chrome-a-manifesto.html
9. Ingram, A., Wang, X., Ribarsky, W.: Towards the establishment of a framework for intuitive multi-touch interaction design. In: International Working Conference on Advanced Visual Interfaces, Capri (2012)
10. Shneiderman, B.: Direct manipulation: A step beyond programming languages. In: Joint Conference on Easier and More Productive Use of Computer Systems. (Part II): Human Interface and the User Interface. ACM, New York (1981)