Best of Andrew Chen's Blog

2006 - 2010

Andrew Chen

Best of Andrew Chen's Blog

2006 - 2010

This version was published on 2012-05-24

Tweet This Book!

Please help Andrew Chen by spreading the word about this book on Twitter!

The suggested hashtag for this book is #andrewchen.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

https://twitter.com/search/#andrewchen

Contents

Introduction

Welcome to an experiment in publishing. The volume you are reading right now collects the best of Andrew Chen's writing over the past four years. The essays are drawn verbatim from Andrew's blog Futuristic Play, one of the best entrepreneurship blogs of all time. The selections in this volume were made by the author, but the essays themselves are unedited. As a result, you'll get to see the evolution of Andrew's thought over the years, including his many industry predictions. Most of them turned out to be prescient. Others, not so much. But in both cases, we have all been the beneficiaries of Andrew's lucid thinking and clear writing.

When I first started blogging, there were a handful of people who reached out to me. At that time, I was a complete blogging newbie trespassing on their turf. Some were friendly, some were indifferent. Andrew Chen was awesome. He has been a consistent and passionate supporter.

Over the past four years, Andrew has written about Silicon Valley, startups, game design, business models, virtual goods, interaction design, and much more. For those of us who have made the startup journey, his story will feel eerily familiar. And for those who aspire to do so one day, his writing is full of incredibly useful tips, suggestions, and advice.

Inside, you'll find contemporary analysis of major Internet companies, like YouTube, Yelp, and even AdultFriendFinder, take-downs of business models from dating to eyeballs to viral. And you'll find provocative favorites like "Do hardcore Microsofties suck at startups?", "Your ad-supported Web 2.0 site is actually a B2B enterprise in disguise", and "Does Silicon Valley noise detract from long-term value creation". And most importantly, you'll learn essential tricks of the trade, as in "How to measure if users love your product using cohorts and revisit rates", "How to create a profitable Freemium startup (spreadsheet included!)" and "Minimum desirable product."

Eric Ries
San Francisco
October 11, 2010

Best of 2006

YouTube and Monetizing Social Networking

My buddy Eric Mattson asks my opinion on YouTube:[1]

> Andrew, Do you really view things like YouTube as sustainable businesses? Bandwidth is cheap. Storage is cheap. Why do people need YouTube? Eric

Good question. (BTW, visit his blog at marketingmonger.com[2])

For now, it's too early to tell whether or not YouTube is a sustainable business. They face all the normal problems related to monetizing user-generated content:

1. Low clickthrough rates
2. Brand-unfriendly content
3. Lack of contextual relevance

The first point is driven by the incredibly high frequency of social sites. According to sources like Nielsen, the daily pageviews per user for social sites can be 100+, which is astronomical. This

[1]http://andrewchen.typepad.com/andrew_chens_blog/2006/09/creating_value_.html#comment-23080491

[2]http://marketingmonger.com

makes it difficult for social sites to easily monetize their content through direct-response advertising like AdSense or “spank the money”-type lead generation businesses.

If not direct response, then what about brand? Well, that can work if YouTube is able to carefully isolate “good” versus “bad” parts of their sites. For example, if they took the top 1000 videos and editorially filtered them, they might have some inventory to work with. Otherwise, it’s obviously a bad idea for Ford to potentially show their ads next to Jackass-like car crash antics. And unfortunately for sites like YouTube, one “bad” impression (and subsequent screenshot) is enough to turn off brand advertisers, no matter how many good ads they can show.

And finally, the lack of contextual relevance is a killer. Most social sites end up with a largely homogeneous set of “profile views” or “video views” or “picture views.” The problem there is that it becomes very difficult to tie this vague grouping of pageviews to advertisers where they might perform on either a direct response or branding basis. The easiest sites to monetize are generic sites with categories like “Automotive” or “Finance.” There, artificial scarcity is created around those categories and high CPMs are charged. If you have a undifferentiated pool, it’s much harder to capture the big dollars.

There are ways to address all the problems above, but let’s really not kid ourselves - this is all irrelevant because somewhere out there, a company is preparing a $1B bid for YouTube.

Why is that? Well, the idea of a secular market cap for a distruptive site like YouTube is pretty misleading. Now, it would surely be stupid for fundamentals-driven investor like Warren Buffett to buy YouTube, because the company isn’t profitable and probably won’t be for a while.

That said, if you were a giant media company fearing obsolescence, would you trade 0.5% of your market cap to hedge by buying a huge Internet company? That starts to make a lot more sense. In fact, you might even do it relatively defensively - a not-so-friendly giant in Redmond has been known to do this. You can call this "stupid money" or guys driven by fear rather than greed, but either way, it's logical for them.

So the short of it is, maybe not a sustainable business, but the founders will still get rich.

As for Eric's final point about whether or not people need YouTube because of storage and bandwidth becoming cheap - I think people mostly use YouTube because it has aggregated millions of eyeballs and people like to be seen. That's something that can be kept and sustained over multiple years. I'm sure plenty of competitors will emerge over time, and some will even grow to be as popular as YouTube, but the technology product is not what drives its popularity - it's the overall experience, the community, and the content.

This reminds me of a broader conversation about thinking about products as "user experiences," rather than the bits and bytes of the code - but that's a blog entry for another time.

5 ways I screwed up my first website

Growing up in the time of the dot com bubble was a surreal thing. In 1999, I was 16 or 17 years old, and a junior at the University of Washington. I had started using the Internet early - as a progression from playing with BBSes to using apps like gopher and archie/veronica at the public libraries. Anyway, it seemed like they were handing out money left and right in Silicon Valley, and I figured that my friends and I were pretty much as smart as everyone else - so why not start our own website?

But first, we needed an idea. It turned out that I was about to rent an apartment with one of my best friends, Jake Kreuzter, which turned out to be an enormously annoying experience. We ended up wandering around the UW, calling phone numbers as we walked around, because there was no centralized place to find apartments. There must be an easier way!

So we decided to start Local-Rent, a website where local landlords could post their apartments and people could go rent it. After coming up with this idea, we recruited a couple other friends from the Early Entrance Program that were really smart and reasonably technical. This included our good friends John Richmond, Thor Sletten, and a bunch of other people. This larger group also entertained other random ideas beyond Local-Rent.

This large group, however, did not go well. It wasn't clear what people were going to do, and slowly but surely, people got bored and it was left to 3 of us. (Jake, John, and myself)

We started building the site, but went way overboard. This, in particular, was my fault. (But remember, I was 16 at the time, so sue me) We started coded stuff up in PHP and MySQL, but we came up with a massive schema and lots of features for the website. We had tables for tenants, landlords, apartment complexes, individual apartments, transactions, rental agreements, the whole thing. Note that this was all speculative, since we really didn't understand too much about the renting business anyway, and we also completely focused on the technology.

One thing led to another, and we ended up having a really complex application - in fact, in the living room we had put up a bunch of Post-Its representing the structure of the app, and it was huge. Furthermore, we started by coding up layers within the application. So rather than working on the user interface and all of that, instead, John and I ended up building an ORM (object relational mapper) for database rows to get instantiated as PHP objects. As an aside: It was a neat experience, from a software engineering point of view, since we didn't know ORMs existed as a general concept and it just made sense to us. So now, to see stuff like Hibernate and ActiveRecord is pretty cool.

Anyway, after a couple months of working very hard and getting no where other than the backend infrastructure, we gave up. The feature list seemed huge, and it seemed like we'd never get to the end.

In retrospect, it was a fantastic experience. It was great to understand some of the tendencies that you can get wrapped into - and the entire engineer-as-entrepreneur thing is especially hard because it's easy to get sidetracked by technology rather than users. So even though it was a colossal failure, I'm happy I wasted a summer doing this when I was 17. It was fun in itself, and I became a better person for it.

In the end, we never finished the site, but years later, it would become very exciting to see Craigslist succeed. Although our idea was really a subset of what Craigslist has become, it makes me happy to see a community, locally-focused classified section where random people post sublets, residential housing, and other rental opportunities. It somewhat validates the thinking of a couple eager teenagers with PHP h@x0ring skillz

All in all, I learned a bunch of really important ways, by screwing up:

1) It’s easy to bite off more than you can chew When we first started, we defined way too much stuff to do. It was fun to dream up the “next big thing,” and people with big vision also tend to go overboard with this. But the practical reality is, defining something really big can be demoralizing. It takes too long to build and see results, and you take on a ton of risk by making your invention a big, high-stakes bet instead of a bunch of little bets.

Instead, the right thing to do would probably be to define the vision, make that your North Star, but then step back. You’d look at all the little micro-releases you could do to build up to your North Star, and start to go from there. If you have to integrate some things, do it incrementally, and not all at once at the end.

For Local-Rent, it would have been better to make a little forum-like application for people to post generic listings right away, and try and reach out to rental people. Probably the project would have gone to the next level right away, as we started thinking about the major business hurdles to seed this kind of marketplace.

I see people making this type of mistake even now - you’ll hear startup people that need to be in stealth for one or two years for a consumer media application. It’s just silly. Launch your site already!

2) It’s easy to chase shiny-technology things, if you’re a nerd This applies to me, but maybe not to everyone. It’s often tempting to use each startup opportunity to learn something new, and thus, take on additional technical risk. In our case, we really, really didn’t need an ORM to make the business work. So why did we work on it? Because it was fun 😀 Thats a good reason if the goal of the project is to play with technology. But it's a bad one of you actually want to get stuff done.

Hilariously, I still find myself making this mistake from time to time now. The fundametnal aspect of it is, I do these types of startup projects to learn new technical things anyway. That's an integral part of the goal. But getting too distracted is bad, and I can catch myself easily from that standpoint.

3) More people at the beginning isn't better Although this didn't create too much of a problem in the long run, it seemed like a really good idea to bring in lots of smart people early on. We brought in friends just because we thought they could contribute one way or another. Frankly, it's just distracting. You end up creating busywork that doesn't need to exist in the first place, just to give people stuff to do.

The group eventually evolved into a mix that did balance out. You want everyone to be able to contribute very concretely in the beginning. So no finance people or business people that just muck around with PowerPoint and Excel. That is 1/2 of one fo the guys' jobs. Instead, you grab one or two engineers that are really good at shipping stuff really fast, and you have one business-y technical guy. (In recent cases, I'm the latter). That way, you can focus on making things rather than talking about making things.

4) Focus on the customer (and make it concrete) Another symptom of playing around with technology instead of focusing on the business was not talking to any customers. Although Jake and I, as renters, were part of the consumer market, we really should have spent a lot of time with the other side of the equation (homeowners) as well as potential partners (classifieds, newspapers, etc.).

The right way to go about this is to be very specific about your target audience, and vet ideas and prototypes with them first. I'm a huge fan of thinking through and incorporating user personas[3] into your design process. If you and your developers are arguing about whether or not "Landlord Larry" really uses the site in X or Y way, you're on the right track.

5) Failing ain't so bad And finally, as I mentioned above, trying this project and then giving up was a rewarding experience. I learned about lots of stuff that didn't work, which then implied some hypotheses around what could work. And although your ego can take a hit, at the end of the day, you'll forget it and be happy that you learned so much.

As soon as Local-Rent was over, it only took another year to dream up something else and try it. This time, we had a lot more success getting it off the ground. I'll write more about that project some other time.

Designing for other people versus designing for yourself

I had lunch with a very smart and capable friend of mine, Max[4], today. Max, myself, and a couple other friends co-founded a student group back in college, SEBA[5] (formerly SBE) which we grew from scratch to over 500 students.

Anyway, we had a conversation about starting companies and the difficulty of designing for customers that are dissimilar to

[3]http://www.amazon.com/gp/product/0125662513/
[4]http://www.maxnoy.com/
[5]http://students.washington.edu/seba/

you. In particular, Max was against it😃 He said he'd much rather stick to hobbies and activities where he had direct intuition to make decisions.

It's a very valid point, and a huge hurdle for entrepreneurs that are targeting contrarian demographics. For example, a bunch of 30-year olds targeting the seniors market, or Americans trying to target overseas audiences. I imagine many folks are building social networks that are exploring these different markets, with some level of difficulty.

On a personal level, I'd say that the projects I've worked on to target audiences other than myself have also led to the most humbling outcomes. After thinking you "grok" a target customer audience, when you show them a prototype of your product and ask them what they think, be prepared! In my case, I got a bunch of very humbling feedback that indicated I had a long way to go. Better to get this information early than later, of course.

Accelerating user risk In fact, you could argue that as you get more and more removed from the user, the risk you are taking that your intuition won't match the target audience accelerates. Here's the range of possible scenarios, from best to worst:

- Being the customer
- Having a business partner who's a customer
- Having a good friend/significant other who is the customer
- Talking to the customer intermittently
- Knowing of the customer and learning indirectly

- Don’t talk to the customer

I think as you remove yourself from daily, in-person contact with the target audience, you take on a ton of risk. Mitigating this risk is a big problem, and you’d have to talk to companies like IDEO and such to figure out how to handle this. Most likely, there’s a lot of effort that would have to go into interacting with customers that you could generally skip by building tools for yourself. **A rule of thumb on user risk?** As a corollary to this, I’ve speculated with my friend Dave[6] about how you’d go about funding pre-launch Web 2.0 startups. Ultimately, we agreed that the co-founders would need to represent the target audience for the company - that’s the only way you can reduce some of the risk from initial user behavior.

In fact, I wonder if the personality of an entrepreneur inherently makes it harder to be empathetic to different types of users. After all, part of doing a startup requires a bull-headed exercise in reality-distortion where you ignore a lot of social convention. The ability to do that may be inversely proportional to actually listening to people 😃

Anyway, I’d certainly like to work on applications outside of my direct audience, but clearly I will have to think more about how to regularly listen to and incorporate feedback from a multitude of different people.

[6]http://www.mdv.com/team_feinleib.htm

Best of 2007 Q1

Why build a vertical ad network?

VentureBeat writes an article on a vertical ad network centered around women's interests: Women's online network, Glam, fastest growing on the Web[7].

Pretty interesting stuff, and shows some smart thinking by the folks at Glam. When you're an ad network upstart, by far the hardest problem is the chicken-and-the-egg problem. To have a happily functioning ecosystem, you need publisher ad inventory and advertisers that want to buy it - but neither set of parties will come to the table without the other.

Owning a destination site is good In the case of Google, they already had their own destination site, so they were able to bring their own inventory to the table. Once they had that, they were able to attract advertisers, which they could then tap as they expanded their base of inventory through partner deals.

In fact, you could imagine that since YouTube is such a huge % of videos played on the internet, they might try to do the same thing with video ads. Very smart.

Not controlling your own destiny is bad… In contrast, if you don't have a destination site, you are very much at the whim of your partners. You may be able to do a couple key deals to get publisher inventory or advertisers, but that might not be enough of a critical mass to provide the highest rates. And on top of

[7]http://venturebeat.com/2007/01/09/womens-online-network-glam-fastest-growing-on-the-web/

that, because you are competing with all the other ad networks, you're forced to optimize for RIGHT NOW so that you can retain inventory, at the expense of longer-term experimentation. Bad news for a startup.

Glam's business future For Glam, they can do the same thing as Google. They can aggregate all the mom-and-pop bloggers and websites that have been able to hit on the right consumer nerve, and provide them with the sophistication and expertise to sell to the brand advertisers that drive the higher CPMs.

There are a couple dangers to this business model, in the long run. The first is that advertisers may think that they are buying inventory on Glam-quality sites, and may not approve if the umbrella gets too big or the standards are too lax. So the advertisers will try to figure out what sites are ACTUALLY in the network, and whether or not they should be buying directly from them. The second issue is that within the Glam network, a couple sites may emerge as outsized winners by pageviews or people - over time, these sites will want to build their own direct sales teams and get capture more of the higher CPM dollars. In either case, Glam can probably acquire some of their publisher partners to get bigger and keep those dollars in-house - that might be a smart way to spend their venture dollars and cashflow from their advertising business.

Can you build lock-in with ad networks? An example with MySpace... And finally, one might ask where the vertical ad network thing could go in the long run? Could more large destination sites create loose conglomerates of sites under one ad network? For example, you could imagine that MySpace has various reasons they might want to assert control over all the MySpace-related layouts/backgrounds/icons sites out there.

They could ask ALL of those sites to run under a MySpace ad network, and trade that for preferred access to an API or something similar under a “Premium Partner” program. That way, they can make sure they are getting a slice of all the ad dollars related to MySpace, whether it’s directly on their site or not.

UPDATE: Here’s an example of the bad coverage[8] that can happen if you slap advertisers’ brands on websites you don’t own… (and report the stats as your own)

3 lessons from a web idea that didn’t go anywhere

In my last post[9], I talked about the second bad website I had started. I learned a lot of lessons (some of them very obvious and amateurish) in trying to explore the idea, and here are some of them:

Focus on the core problems, not on “acting” like a business
To a complete startup newbie, there’s a lot of pressure to create legitimacy when you’re a total nobody. This can manifest itself in many forms, and the more corporate/traditional/Microsoft you are, the more likely you are to cave into these pressures. So what are some things that people expect “real” companies to have?

- Business cards

[8]http://www.valleywag.com/tech/how-to/inflate-traffic-pilfer-content-and-bamboozle-investors-227838.php

[9]http://andrewchen.typepad.com/andrew_chens_blog/2007/02/bad_company_ide.html

- Office space
- "Experts"
- A really big, complex idea
- A big team
- Lots of money
- etc…

Ultimately, this insecurity about what a startup is about can really gnaw away at new entrepreneurs, and they focus on the wrong things. In fact, whenever I see a new startup with 4 MBAs, I imagine them running around making financial models, getting office space, and trying to decide on the "brand" behind their business cards when they should probably be trying to prototype their product and verifying consumer/market assumptions.

From the venture capitalist's standpoint, they are betting on the team to execute against a bunch of unknowns, and that involves collecting fundamental data right away. So that should be 90% of your time with 10% being dedicated to whatever minimal things are necessary to support the central tasks.

For us, of course, Jake and I spent a bunch of time creating business cards, trying to convince our friends to join us, architecting for "massive scale," as well as forecasting and re-forecasting financial models. Luckily we didn't waste a huge amount of time and money doing this.

Nerds love to jump right to the code, but you should probably start elsewhere If you get a bunch of engineers together and you have a great idea, there's an instant desire to jump to the code right away, and start developing the product. Let me argue that while this is a great way to get some intuition about the idea, it's probably not the right thing to do at first. (Although you want to get there quick!)

At the end of the day, investing in the development process is a hugely expensive thing to do, by time or money. Once you build out a wonderful dating site, it's hard to turn that into a photo-sharing site. Yet at the same time, these days the development process is not the highest risk thing. Instead, the highest risk thing will be consumer behavior risk, or market risk, or some other issue. Put those two together and you can safely say that it's probably more important to identify the key risks, test for those, and move on.

The key assumption in our business idea was that there was enough value to be collected from these books that it would make it worthwhile. While there are several ways to test that assumption, we jumped into the code very quickly. We later realized that the right approach was to call up a bunch of these charities and used bookstores, and ask them how many books they had. We also figured out that we should actually figure out how many of these books had value, versus being scrap. We ended up creating a pilot program with one of the Goodwill charities to collect data on this, using little more than a barcode scanner and Notepad. Even then, we probably invested too much in our technology system to price things in real-time, post eBay listings in real-time, etc.

Iterate quickly and cheaply (rather than slowly and expensively) Some folks that are reading this might think that going to the code is the cheapest thing to do. Let me beg to differ.

For almost any consumer site out there, the risk on consumer behavior is really the biggest. How do you know that people will like your site? How will you know what audience will love it the most? What features will appeal to them? These questions are impossible to answer.

So I think ultimately, you have to go down the IDEO approach where you pick a general target market, interview a bunch of people, and build a persona for the person you're trying to target. That way you can have constructive conversations about Artsy Anna and how she reacts to new feature X on your photo-sharing site. But after then, you have to be prepared to build 20 different concepts, iterating on each one, until you catch the perfect customer experience.

The key thing that eventually caused us to stop working on the books idea was that Jake had the great thought to grab a dozen books and auction them all off on eBay. Theoretically, a bunch of what we were doing was contingent on eBay being a "perfect" marketplace for books. If Amazon priced a used book at $20, then the eBay closing price would hopefully match that.

Of course, we found that $20 books went unsold, while $0.05 books sold for $10. At the end of the day, our model was broken because there wasn't a good way to figure out what books were monetizable and which ones weren't. While we could have pushed it further by trying to pattern match and creating our own eBay pricing predictions algorithm, instead we laid down our cards - Jake was going to NYU Law School in a month or two anyway, so our little hobby was over.

In retrospect, of course, we should have just done that first. When we came up with the idea, we should have iterated on it by just auctioning off 10 books right away. And if that worked, we could have gone to verify our other assumptions, but we chose to start nerding out right away. Although this could have worked if we were lucky, I think 9 times out of 10 it doesn't end well.

So if you are working on a new website and 10 months into it, there's no traction and you don't know why, you probably could have stepped into it more experimentally. Take the approach as "finding a company" rather than "executing on a plan" and you'll get more mileage from the situation.

How to tell the difference between eyeball companies

I recently wrote on the difference between Eyeball companies versus revenue companies[10].

I've been thinking about the issue of "eyeball companies" AKA companies with lots of users without any revenue. First off, if you have one right now, good for you 🙂 Right now those companies are worth a lot of money, as old media companies are still trying to figure out what to do with this whole Internet thing.

Ultimately, I'd evaluate the future success of eyeball companies according to the following criteria:

First, the basics..

- Have they proven users love it? (100k+ users is good)

[10]http://andrewchen.typepad.com/andrew_chens_blog/2007/02/eyeball_compani.html

- Will additional growth be cheap/viral or expensive? (>50% growth M/M is good)

Then, the interesting strategic questions:

- Are they doing something for free that someone else is doing for $$$?
- Are they taking "attention" from another big, ad-based platform?

The first two I won't address in much detail, except to say that you're not an eyeball company until you have eyeballs 😃 But the second two are interesting questions in the sense that the public markets aren't open right now, so you need someone to step up and pay a LOT of money to take you out, if you're VC-backed.

Because of this, you basically have the big Internet conglomerates (AOL, Google, Yahoo, Microsoft, IAC, and FIM) and old media companies to turn to, in terms of an exit. You ultimately have to look at their businesses and figure out if you're doing something strategically interesting enough to them. So to reiterate, if you are doing something related to internet radio, video, calling, etc., you pose a disruptive innovation to big companies. Other people make a lot of money from that, and they will get nervous if you get really big. They will probably think about buying you if you get really big, either for themselves or to keep you away from a competitor.

However, if you are an eyeball company that has a lot of users that no one cares about - and I don't want to do any finger-pointing but I have my ideas - then even if you get big, no one will get too excited about what you are doing.

Just my quick theory looking back at Skype, MySpace, YouTube, and such.

What's broken with online travel?

Take a moment to think. When you think of the word "travel," what do you imagine?

I've asked a lot of people this, they typically say something like this:

When you ask them to elaborate, they say:

> Aaaahhh. Beach. Relaxation. Escape. Fantasy. "Me time." Or, if you have a significant other, "Our time."

In fact, if you browse the Flickr pool for travel[11], you see a lot of beautiful pictures of exotic cities, breathtaking environments,

[11]http://flickr.com/photos/tags/travel/interesting/show/

etc. (If you have a moment, I encourage you to click and watch... it's really great!)

So taking a page out of my previous analysis on online dating[12] using the MDA framework[13], you could ask the following questions:

- What emotions do you want people to feel, when it comes to online travel?
- What are desirable dynamics and behaviors, when it comes to those emotions?
- And finally, what do you need to build to accomplish these things?

I'd encourage you to read the previous posts on dating and the MDA framework, otherwise this post will make little sense 😃 Let's analyze the current stage of the art.

What emotions do Expedia/Travelocity/etc. convey? Let's face it - Modern travel websites are all about logistics. What flight do you want? Where do you need to go? What dates? They're optimized for business travelers that need to get from Point A to Point B, without a big fuss. So when you think of Expedia[14], you can think about airports, planes, taxi rides, and all the things that get you from Point A to Point B.

[12]http://andrewchen.typepad.com/andrew_chens_blog/2007/03/whats_broken_ab.html

[13]http://andrewchen.typepad.com/andrew_chens_blog/2007/03/game_design_tut.html

[14]http://expedia.com

And in a way, that's advantageous for them - it means that Expedia is about a transaction. The emotions that it conveys are about efficiency, cost effectiveness, and Getting Things Done. Which is great, when you're someone that needs to get from one place to the other.

But how many people, when you ask them about "travel," say they think of: Itineraries, cost effectiveness, efficiency, etc.? Let me argue, though, that for most consumers, travel is not about that. For most people, it's about getting away.

What emotions do you want to amplify, in a travel site? As I mentioned before, my thesis is that most consumer travelers care most about the fantasy of travel, not the logistics. So ideally, you're looking to trigger emotions like:

- Wow, wouldn't it be great if... (Fantasizing)
- Ooooh, that's so pretty! I could just curl up and watch... (Relaxation)
- Honey, it'd be so fun if we... ("Our time")
- God I hate my job, I could really get away... (Escape)

If you had people looking at a series of pictures or imagining what they could be doing, then you have succeeded.

Like all experiences, "travel" starts earlier than when you board the plane. A very smart friend of mine, Kevin Lee of Y!, pointed out that the act of buying magazines and books to do research was part of the experience. Doing those things created the fantasy.

What dynamics do you want to create? Ultimately, if you are succeeding in getting people to experience these emotions, you might argue that they'd do it all the time.

For example, with fantasy, they might create little vacations they probably wouldn't do, just for fun. For example, making an Antarctica trip. Or an African safari that was too expensive, just for the help of it. Or perhaps a girlfriend would plan a romantic getaway for a boyfriend, sending it out of the blue as a substitute for daydreaming.

For relaxation, maybe people would come to the site and make little trips just to soothe them and look at pretty pictures. If they were stressed out from work, maybe they'd come and check out other peoples' vacations, just to keep their mind off things.

if people are making little vacation slideshows and sending it around, or visiting regularly and planning random trips to all over the place, then you know you've really engaged them.

What mechanics drive the dynamics? Now comes the part when you start analyzing the actual features and functionality that drive the dynamics and ultimately the aesthetics of the product. As an aside, it's clear that a lot of nerds start here when they start building products - it's easy to fall in love with technology that way. But when you do that, all you are doing is leaving the aesthetics to chance! They will automatically grow out of whatever features you put in place, and you may or may not understand why your site attracts an audience. (Or doesn't attract an audience)

Based on what I've written so far, I think it's clear that part of what is needed is some sort of immersive travel fantasy process. So you know that some sort of slideshow, or movie, possibly with music, would be great. You might also even place some avatars into the slideshow, to increase the fun. Video would work for this, as well. So you know that this is the end product.

You also need to help people constrain the fantasy, so that they can use it as a practical planning tool. That means when you show a picture of Angkor Wat, you'll want to link to Wikipedia information on it (or whatever). And you'll want to show a visual representation of the itinerary, maybe Indiana Jones style[15]. Similarly, you'll want to take into account information like: how many days are you traveling? What region of the world? What's the budget? What's the schedule?

A skeleton of a travel product idea? Rather than point out a problem and not try to fix it, I wanted to take a crack at the START of a conversation at fixing it. I'm not a travel industry insider, so I don't understand the logistics, but I think the "emotional mismatch" problem is a great one to try to solve. Particularly for a nerd, since it requires thinking about people rather than logistics and technology.

Just brainstorming randomly, perhaps you could have a site where you start out by selecting one picture out of several pairs, similar to what you do on LikeBetter.com[16]. You could use this process to elicit travel preferences, such as:

- Do you like to travel to party? Or to relax?
- Do you prefer tourist traps or authentic holes-in-the-wall?
- Are you an independent traveler, or do you like to go in groups?

[15]http://www.gungeralv.org/dg/archives/000372.php
[16]http://likebetter.com/

- ... etc.

You could then collect some more mundane information like, dates, budget, region of the world, etc. You'd try and make this as visual and engaging as possible, in alignment with the overall feel of the site. So region would be selected through a fancy map with pictures, or whatever.

Based on that information, with preferences and constraints in mind, the system would then generate a series of dreamy slideshows that describe potential itineraries. The user would flip through them, with the option to yay or nay, and the system would adapt. You could regenerate ideas over and over again, to see lots of variations. You could fade the edges of the slideshow, and have them come in and out slowly.

At the same time, you'd need to connect this "toy" with the real world. So under every picture, you'd need to let the user drill in deeper for information. Or combine one travel idea with another. Or understand the pricing differences, or constraints like schedule and so on. And of course, you'd want to have a "Buy" button.

And once you're done traveling, you can upload your pics onto the site and create the ACTUAL slideshow of what you really saw. Perhaps these pictures might get recycled into the entire system, for other people to use.

[Optional pet feature: You might have a little dotted line cutout for an avatar, where you can place you, your friends, boyfriend, or whoever might be suited for being part of the slideshow. Maybe the system could even generate little sayings with justifications like, "Wow this is sooo relaxing!" Ever play with an avatar system? It's fun… make one of you or a friend here[17].]

How do you measure success? If a concept like this were successful, you could imagine that before Spring Break, college kids would be sending out travel slideshows like crazy to each other, trying to come up with the best ideas. Or a couple best friends now living far away might send slideshows to fantasize about relaxing together. Or a boyfriend might surprise his girlfriend by pitching her a vacation idea.

In essence, if people start to use the site for fantasizing rather than pure logistics, you know you were able to capture the aesthetics of travel better than what exists out there today.

[17]http://www.joystiq.com/media/2006/10/mii.swf

Best of 2007 Q4

Technology always changes, but people always stay the same

A couple friends were in town recently, and I went with them to the Mechanical Museum at Fisherman's Wharf[18], where they have lots of different old mechanical arcade machines. The oldest one was from the 1920s, and the average period looked to be 1950s or so.

It reminded me that while technology advances, human nature stays the same.

Love calculators First up, we have a bunch of machines that are sort of like the "How good of a lover are you?" quizzes in trashy women's magazines[19]. In general, you stick in a quarter, put your hand on the pad (or some other interactive action), and then it gives you a score.

Here are pictures of the machines I took on my iPhone - click to see a bigger version:

[18]http://www.museemechanique.org/index.html
[19]http://cosmopolitan.com

[20]

[21]

[22]

The main emotions that are being elicited in these cases are some combination of:

- narcissism
- curiosity
- competition

Extra points to the first (left-most) machine for the tagline:

> "What do your friends call you behind your back?"

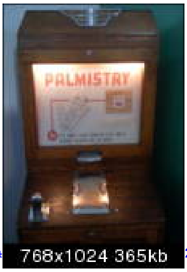
Is this really so different than the various quiz, comparison, and other applications on Facebook? And take a look at a site like this: Best Love Calculator[23]. It even evokes the look and feel of these machines.

Telling the future Also, we have machines like the ones below, which are focused on telling the future. The first machine is the

[20]http://img520.imageshack.us/my.php?image=img0054ll4.jpg
[21]http://img401.imageshack.us/my.php?image=img0059ym3.jpg
[22]http://img152.imageshack.us/my.php?image=img0058ne7.jpg
[23]http://www.bestlovecalculator.com/?gclid=CMHRsPSmuo8CFQgZhgod6W94WQ

most fun - you put your hand into the machines mouth! The second and third ones are both palm reading.

Here are the pictures:

[24]


[25]

Of course, horoscopes are still big these days, and people still inexplicably talk about their "signs" - I would ask "who knows why" but the answer to that is simply "because people are people." I'd boil the emotions down to:

- narcissism
- curiosity
- insecurity?

In the modern world, the entire clairvoyance thing is still around, via numerous websites that you find when you google for "psychic"[26]. Here's a good example of what you get whenyou

[24]http://img180.imageshack.us/my.php?image=img0071ne8.jpg

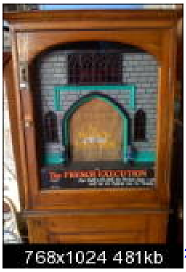
[25]http://img383.imageshack.us/my.php?image=img0040zv0.jpg

[26]http://www.google.com/search?hl=en&safe=off&client=firefox-a&rls=org.mozilla%3Aen-US%3Aofficial&hs=Gc2&q=psychics&btnG=Search

click on one of those ads[27]. And don't even get me started about John Edward's[28] Crossing Over and psychic shows like that.

Jackass and YouTube, oldschool style These next machines are probably the most dated (and hilarious) - basically when you put in a quarter, you're then able to watch some "kinky" (defined by the 50s). The first machine was the "French Execution," which played some music and you got to watch a guillotine chop off a miniature doll's head. The second machine had you grabbing a hand-crank to operate a flip book with pretty boring material in it.

Here are the images:



[29]

[30]

This is obviously Jackass and the YouTube of the 1950s.

Emotionally, this is catering towards:

- novelty
- scarcity

[27]http://asknow.com/LandingA.aspx?GroupName=Psychic
[28]http://en.wikipedia.org/wiki/John_Edward
[29]http://img522.imageshack.us/my.php?image=img0070dh6.jpg
[30]http://img507.imageshack.us/my.php?image=img0042ck8.jpg

- curiosity

Of course, the problem with these machines (unlike the other ones) is that after you've seen it, it doesn't seem so special anymore, and you're unlikely to watch it again. And of course, in a modern society where this type of stuff is available at a much more, ahem, liberal standard, it's really boring. These were fun mainly because they show how dated the place is.

Simulation games While Will Wright is often heralded for creating the Sim games for the PC[31], you can look further back than that to see simulation games. In these machines, we see one which is a helicopter machine and then a crane, both of which are operated using a set of simplistic controls.

Here are the images:

[32]

[33]

Interestingly enough, both simulation games were "directed" rather than undirected. They weren't pure sandbox games (like SecondLife) but rather had a goal structure for the user. This is something that Erik Bethke (of GoPets)[34] and I have talked about in the past.

[31]http://en.wikipedia.org/wiki/Will_Wright_%28game_designer%29

[32]http://img520.imageshack.us/my.php?image=img0044tm6.jpg

[33]http://img99.imageshack.us/my.php?image=img0049uy8.jpg

[34]http://erikbethke.livejournal.com/

Rather than just letting the user fly the helicopter, instead there were lights around the area where you were supposed to hover the chopper. The longer you hover, the more points you score. The lights rotate around the area, and you have to move the chopper there accordingly. The construction crane does the same thing, where you are supposed to grab as much dirt as you can in as little time as possible.

The emotions here are quite different than the other ones:

- aspiration
- fantasy
- competition

And of course, it's obvious that the modern versions of this range from things likeThe Sims[35] to MMOs[36] to any other game that is about role-playing.

Differences with regular websites and Facebook apps Interestingly enough, the machines above are actually quite different than what you would want to build for a modern consumer product.

The reason is that these machines are incented to:

- Have a great hook to draw a user in
- Make them give you a quarter

[35]http://thesims.ea.com
[36]http://worldofwarcraft.com

- Provide some value, but focused less on retention and more on pumping in more quarters

This is misaligned from websites, which share the attribute of drawing users in, but are focused around retention and constant usage, because that's what drives advertising revenue.

Similarly, Facebook apps have a different incentive structure:

- Have a great hook to draw a user in
- ALSO, have a great hook to get the user to pass it along to their friends
- Provide some value, but mostly focused around virality
- Make the structure around frequent usage with continuing value

That's why things like Magic 8 Balls and Fortune cookies and such are gimmicky products that might drive acquisition, but have problems with overall retention and active usage.

Conclusion I often find that studying older historical products like this to be really fascinating. I think you can learn a lot about human psychology by looking at things like:

- card games
- physical architecture
- con artists
- old advertisements

- public speaking
- magic and psychics
- etc.

While many things are not directly applicable to the world, many of these have underlying themes and emotions that might be useful for modern entrepreneurs.

Why your friends list get polluted over time

Best friends forever? I've recently been doing some qualitative research into how people use social networks, and I've learned a great deal of interesting stuff through these interviews. Typically, I'm spending about an hour at a time having folks go through exercises like describing 2-3 items that represent them, drawing out their social network, talking about meeting new people, and a bunch of other random things.

While doing this, I've been paying attention to something that's been bothering me over time as people friend me on Facebook:

> Why does my friend list get so polluted over time with people I don't know at all?

As a great rush of people in SF have gotten on Facebook, I've gotten regular friend requests - mostly legit, but some completely random strangers - and over time, I've collected a pretty large group. However, my group is much larger than the so-calledDunbar number[37], which estimates the largest group size that humans can have social relationships with. (It's 150, by the way)

In fact, this entire issue of "real friends" versus "fake friends" has been an issue in social networks for a long time. First, with Fakesters[38] on Friendster, then talk of "fake friends[39]" on MySpace, and even certifications[40] for folks on dating sites. In the past, it's been said that Facebook actually reflects your "real life" friends because of all the geographical semi-private network

[37]http://en.wikipedia.org/wiki/Dunbar's_number

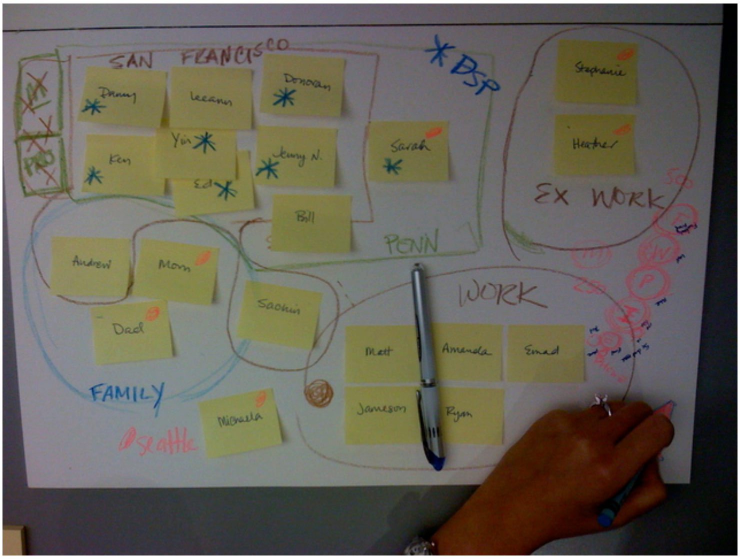
[38]http://www.wired.com/culture/lifestyle/news/2004/07/64156

[39]http://weblog.infoworld.com/techwatch/archives/009121.html

[40]http://www.true.com/magazine/saferdating_prosecute.htm

stuff they do, but over time, I've found my own personal network saturated.

Friendships are complex The first underpinning of this discussion is that friendship networks are actually very complex, and are poorly approximated by the "friends" versus "not friends" paradigm, or even the "friends", "top friends", and then "not friends" paradigm. In fact, you'll see that a lot of social maps look like this:



This is my sister's social map that she drew out for me, and is one of about half a dozen I've seen so far. What you'll see is several overlapping networks based on geographical location (SF versus seattle), organizational affiliation (school/work/etc), suborganizational affiliation (fraternity at school), versus strength of relationship.

And in fact, once you have this social map drawn out, one of the most interesting questions you can ask people is how they figure out in what situations they should:

- call someone
- text someone
- e-mail someone
- poke them
- write on their wall
- write them a message
- meet them in person
- etc

What you'll find, in that discussion, is that there's a steady progression of "commitment" that it takes to go from writing on a wall (the least burdensome thing) versus meeting them in person (the most burdensome thing). In fact, one of the really useful things that social networks provide that e-mail doesn't is a range of expressiveness in your communication such that you can use it for more things than sending notes or data across the wire.

Where does adding a friend go into this? Interestingly enough, if you ask people where "adding a friend" fits into the spectrum of interaction, where do they put it?

That's right: They put it in the very beginning as the EASIEST and LEAST burdensome interaction they have with people. So in fact, if you don't know someone at all, or you are just acquaintances with them, the first thing you do is add them to your friends.

And folks, that is not much of a filter at all 😉

So what you'll find is that as your social network evolves online, you'll end up accumulating more and more acquaintances as a % of your total friends, until your friend list is by far mostly people you don't know (or that you knew in the past), but that you don't really care to see all their pictures and their app installs and all that stuff.

It's very unclear how to come up to a solution for this - you certainly don't want people to need to describe their social networks at the level I asked them to, yet there's enough complexity and detail in your social relationships that you need to capture a lot of detail in order to fix the friends problem.

Is your website a leaky bucket? 4 scenarios for user retention

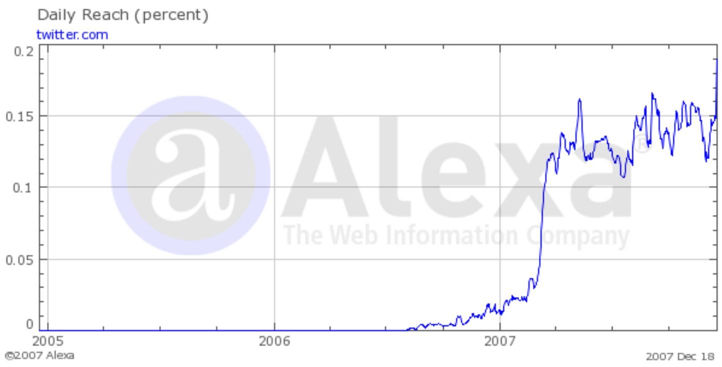
Do you have happy, smiling users? I've previously written a lot about metrics and user acquisition - just look at the left bar of this blog[41] - but have not written much about metrics and user retention. By retention, I mean the process in which you convert new users who don't care about your site into recurring users that are loyal and continually drive pageviews.

In general, I would say that more people care about this than pure user acquisition, which is great, but they are often using aggregate numbers to measure this retention. By aggregate data, I mean looking at an overall Google Analytics number, or looking at an Alexa rank, or some other rolled-up metric which doesn't differentiate between new users that are discovering your site for the first time versus loyal users that are returning to your site.

In fact, in general I think of websites as "leaky buckets" where users are constantly getting poured into the top, and the site is constantly leaking users. In fact, you can imagine that if you pour 1,000 users into any website and then stop additional new users from joining, that 1,000 can only decrease. Over time, some users become loyal and throw off pageviews, but over time, they disappear. The rate at which this happens can be a turned into a metric just like any other number.

Pop quiz: Is Twitter retaining users? First off, take a look at this graph and tell me if you think Twitter is retaining its userbase month over month. What do you think?

[41]http://andrewchen.typepad.com



Think you have any answer?

The growth disambiguation problem And of course, it was a trick question. In fact, it's basically impossible from purely outside data to disambiguate the following scenarios:

1. Pageviews are coming ONLY from new users
2. Pageviews are coming ONLY from one generation of users (like early adopters)
3. Pageviews are coming ONLY from retained users
4. Pageviews are coming from new users and retained users

This should be totally obvious to people, but instead I see people pointing at Alexa graphs and saying that site A or site B is doing well, when in fact they could have a deep systemic problem.

In fact, let me argue the following in this post:

> From aggregate data (like Alexa), you can figure out what sites are doing *poorly* at retention, but not what sites are doing *well*

Let's start with the first scenario:

1. Pageviews are coming ONLY from new users In this first scenario, the retention on your site totally sucks meaning that you lose all your people after the first session. That means that the drop off from a 1,000 users flowing in is 1,000 dropping to 0. Your retention rate is 0% from week 1 to week 2 😃

That said, how could you still get pageviews? First off, you obviously get any pageviews a user might create in the first session, even if they never come back. I think the most common scenarios are the following:

- Users create text content which is SEO'd and placed in the Google index
- Users send invites via e-mail which are then accepted

In either case, they are some form of "viral loop" that attracts new users even if the original user is never retained. In fact, I bet you that a lot of sites out there are buoyed by their search engine traffic, even when they have really terrible retention rates. All that matters is that they do enough work to generate a couple pageviews, and then bring in the next generation.

Using the bucket analogy, this is a bucket that has a firehose filling it, but all the water leaks out almost immediately. With a big enough firehose, the aggregate stats could look good when they are in fact rather shitty.

2. Pageviews are coming ONLY from one generation of users (like early adopters)3. Pageviews are coming ONLY from retained users Similar to the first scenario, you might have a situation where the numbers look great, but it's because the bucket was able to fill well in the first group of users, but after then, the site sucks at retention. Or the inverse, where there's no growth at all, but the retention is great.

In either case, this might hint at a bad systematic condition within the site, but ultimately the aggregate numbers hide the problem. In either case, not being able to acquire and retain brand new users is a problem, and without measuring the groups separately, it seems impossible to assess the true situation.

Back to Twitter for a second So in fact, looking at the Twitter chart, the right answer is "we don't know." A plateau'd chart like that could mean that Twitter is doing fine at retaining some set of users, and it's stalled on new users, or that it's acquiring news users like crazy but not retaining them, or anything in the middle.

That said, given the fact that Twitter pages show up in Google[42], which will provide them with a steady stream of new users, and that the average time on site looks closer to a heavily SEO'd site like Yelp than a social site like MySpace (5min instead of 30min, according to Compete.com), I'd guess that they are actually bleeding users pretty rapidly. Again, it's hard to do an analysis like this without a lot more data to back it up, but that'd be my high-level analysis.

How do you figure out the health of the site then? Measuring

[42]http://www.google.com/search?hl=en&safe=off&client=firefox-a&rls=org.mozilla%3Aen-US%3Aofficial&hs=Wtw&q=site%3Atwitter.com&btnG=Search

"cohorts"[43] In general, the solution to the retention measurement problem lies in separating out NEW users and RETURNING users within the analytics. So at the minimum, you'd have to be able to talk about the following:

- 1 million uniques to the site
- 100,000 new uniques
- 900,000 returning uniques from the month before

That'd give you a sense that the site was actually retaining users well. But to take this further, what you really care about is to carve up your userbase into "cohorts," and measure drop-off rates from time period to time period. Here's the definition of a time-based cohort:

> A cohort is all the users that joined through a particular time period[44]

Only then can you track the retention rate of a SPECIFIC set of users, and then measure other users experiencing an independent scenario. In the "cohort model" you'd end up with a group like:

> **Users that joined in Week 1** week 1 uniques: 100,000 week 2 uniques: 50,000 week 3 uniques: 25,000

[43] http://en.wikipedia.org/wiki/Cohort_(statistics)
[44] http://en.wikipedia.org/wiki/Cohort_(statistics)

In this model, you'd see that 100k users joined in week 1, and if you follow that "cohort" through, you end up with a 50% drop-off rate from week to week.

But then, in week 2, new users joined as well, which creates a week 2 cohort. Of course, in your aggregate metrics, the site would have 100k uniques in week 1, then 125k+50k uniques in week 2.

> **Users that joined in Week 2** week 2 uniques: 125,000 week 3 uniques: 50,000

Note that this cohort only goes through 2 weeks because it starts at week 2 and ends at week 3, whereas the week 1 cohort is able to run 3 weeks.

When you compare to the week 1 to week 2 cohort, you can tell that 1) there was a 25% increase in new users (100k to 125k), and that the retention rate DECREASED to 40% (50k/100k versus 50k/125k). This would be a red flag that your site was sucking, even if your aggregate stats looked good:

> **Total site stats** week 1 uniques: 100,000 week 2 uniques: 175,000 week 3 uniques: N/A* (*since week3 cohort is not defined, 25k+50k+week3 cohort stats)

It's not clear what your time period should be - perhaps weeks, perhaps days, perhaps months. Probably it depends on the average time between your users logging in, or something similar.

Is there a retention coefficient? In fact, one might argue that in analyzing these cohorts that in addition to a "viral coefficient" which is measured in viral marketing, there's in fact a "retention coefficient" that measures how well you are able to keep ahold of users.

This would be true if the cohorts you chose typically lose a constant % from week to week. That would mean that every cohort decays exponentially, which would give you a coefficient. (i.e., f(x) = e\^-ax, where a is the retention coefficient)

Please measure and e-mail me your findings

Best of 2009 H2

Built to Fail: How companies like Google, IDEO, and 37signals build failure-tolerant systems for anything!

Failure is fun, but sometimes only for the people watching - courtesy of GapingVoid[45]

Planning for success, not failureHigh achieving people who have a long history of being successful often plan accordingly - doing so, of course, means that they plan for success in whatever they do. And when you take a successful person

[45]http://www.gapingvoid.com/

and put them in a successful big company that's already making money from their products, there's even more reason to plan for high-achievement outcomes.

But let's say that you put these successful people and put them in environments of great uncertainty, like at a Silicon Valley startup - what happens? That's when realities collide! When you apply the big successful company playbook to startups, you can end up with monolithic planning processes, products that can't find their markets, and lots of money being spent on launches for the wrong products. It's not that these tactics are stupid, it's just that they don't work as well when you're dealing with ill-defined customer problems with unknown solutions.

At the heart of this conversation is - what happens when you take something that's usually assumed to be successful, and you instead say that it's very likely to fail?

In a way, you can think of this as planning to fail, but then building the support structure around the failure in order to create a failure-tolerant system. Let's dive into this.

Planning for failure, not success The title of this blog refers to the fact that companies like Google, IDEO, and 37signals all have the culture of "Failure is OK" built into them.

At Google[46]:

- Google makes money by being always available, ubiquitous, and having a great product
- To deliver their service, they have 100,000s of servers (maybe more?)

[46]http://google.com

- Any one of these servers have a high likelihood of failing at any time
- To create a fault-tolerant system, they have lots of redundancy and lots of sophistication around what happens when an individual box fails
- Contrast this to a big-iron approach that builds all the redundancy into specialized hardware that's designed to never fail

At IDEO[47]:

- Companies hire IDEO to give them fresh designs based on a customer-focused approach
- Part of every project involves lots of brainstorming and coming up with ideas
- However, any specific idea is likely bad (for example, 12 out of 4,000 toy ideas[48] were actually successful = 0.3%)
- Thus, IDEO combines structured brainstorming[49], rapid prototyping[50], and field research to rapidly try out new concepts and get to good products
- Contrast this to a process where the "Great Man[51]" designer thinks about a design problem and then comes up with the right solution spontaneously

[47]http://ideo.com

[48]http://bit.ly/eNgdE

[49]http://www.google.com/search?client=safari&rls=en-us&q=ideo+brainstorming&ie=UTF-8&oe=UTF-8

[50]http://ecorner.stanford.edu/authorMaterialInfo.html?mid=687

[51]http://en.wikipedia.org/wiki/Great_man_theory

At 37signals, in particular Ruby on Rails:

- Rails is framework built for programmers to build websites
- Of course, every web project requires lots of lines of code which can easily break at any moment
- If you assume that programmers will more often write code that is buggy and breaks, then you'll want to make testing and iteration easy - this is at the heart of Agile, TDD, continuous integration, and other related disciplines
- Contrast this to a waterfall engineering approach which assumes the correct design and architecture can be thought out by experienced software engineers

Each one of these examples is similar, yet unique in their own way - but there are similar themes that pervade each one of these approaches.

Characteristics of failure-tolerant systems Each one of these systems takes the central part of a process and assumes failure, and then builds up a support system around it.

This happens by building on a few core principles:

- **Acceptance of failure**: You have to accept that shit happens and failure is commonplace - this needs to be internalized so that failure isn't punished, but rather embraced!
- **Massive redundancy**: Then, it needs to be easy to have lots of redundancy built into the system - for designers,

that means lots of designs get generated. For startups, that means lots of ideas are tested, and for Google, that means lots of servers are used

- **Cheap, easy, fast**: As a side-effect of the redundancy, it needs to be easy, cheap, and fast to have lots of ideas, lots of servers, or write lots of code. The harder it is, harder it will be to create redundancy
- **Iterative, reality-based testing**: Testing these individual components constantly becomes key - you need to force failure on the system to figure out how it reacts from a system-wide level

Building up processes based on the ideas above makes it easier and easier to deal with failure and come out on the other side!

Conclusion and next ideas There are lots of interesting directions that this line of thinking can go.

This area of thinking started out with the hiring process, and the idea that maybe interviews don't work at all - there's a bunch of academic research that implies that, actually. So if how would you build a failure-tolerant system around the hiring process, if you assume that good interview candidates actually have no correlation to successful employees?

For dating, what happens if you assume that people you like to date may not be the kind of person you'd have a successful marriage with? What if people suck at figuring out what kind of guy or gal is the "type you'd bring home to Mom?" I think anyone could attest to the idea that many people suck at figuring out the right person to date, much less the right kind of person to marry. I personally find it crazy that people make a 50+year decision to be married based on a 18-month sample size

For careers, what if it turns out that people have a really bad idea figuring out what they'll actually want to do 40 hours a week, 50 weeks a year, for the rest of their life? How would you figure out the right career faster rather than shorter?

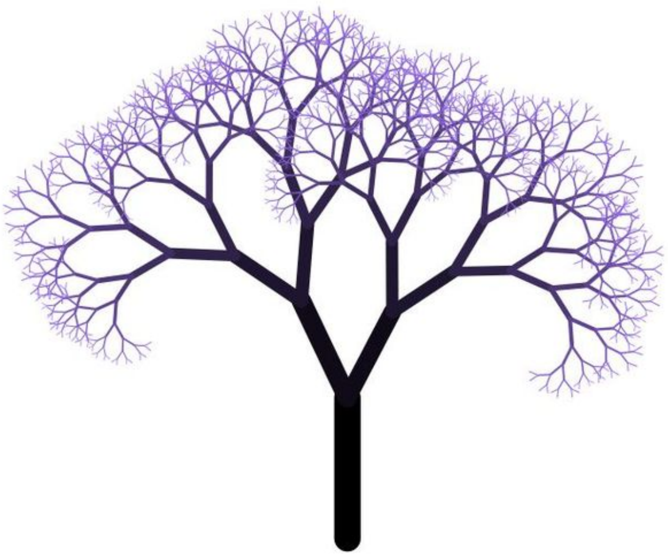
All of these are great thought experiments, I think.

What else am I missing? I'd love to take any suggestions and write up some thought experiments around it.

Want more?If you liked this post,please subscribe[52] or follow me on Twitter[53]. You can also find more essayshere[54].

[52]http://andrewchenblog.com/subscribe/
[53]http://twitter.com/andrew_chen
[54]http://andrewchenblog.com/list-of-essays/

5 crucial stages in designing your viral loop



Designing a viral loop has multiple stagesViral loops have been featured in mainstream media[55] and there's even a book[56] coming out on it - but the step-by-step design of creating a new loop remains obscure, and for good reason. I've come to believe that creating viral loops is akin to

[55]http://www.fastcompany.com/magazine/125/nings-infinite-ambition.html

[56]https://www.amazon.com/dp/1401323499?tag=futuristicplay-20&camp=213381&creative=390973&linkCode=as4&creativeASIN=1401323499&adid=026J7206CA6MG6ANM668&

building a software project - at best, it still comes down to a great team, a strong understanding of the tools available, and relentless iteration. There's no recipe at the heart of it which guarantees a viral process every time, the same way that you can't guarantee that any software project will result in market success.

There are no silver bullets in viral marketing In fact, the core of virality ensures that there will never be a dominant "recipe." If everyone knows how to build a viral loop around social network invites, then everyone will do it, resulting in consumers will become desensitized, which finally leads to lower response rates. Thus this causes the viral loop to unwind, which leads to long-term disaster.

The only way to combat this is to build a viral loop around the core of your product - something that no one will seek to duplicate, unless they are a direct competitor. These viral loops are incredibly effective because they are lasting and sustainable.

I wanted to jot down a couple thoughts on the different stages that viral loop design go through, so that the entreprenurs reading through this can imagine deeply ingrained, user-aligned ways for their products to gain distribution.

Strategize: Stage 1 The first stage of a viral loop is developing the core strategy around the loop. This requires the viral loop designer to think through, step-by-step, how a user will come to find their product and how they will ultimately pass it along to their friends. If you're lazy, there are lots of recipes to follow from the Facebook ecosystem like quizzes, "find your friends," and gifts. As discussed above, these opportunities are already becoming less effective every day.

Even if you decide to use an existing recipe, here are some higher-level strategy questions that should be answered before proceeding:

- How does this viral loop fit into your core product?
- What is the fundamental value proposition you are presenting to your users?
- If your loop is successful, will users transition to your core product or will they bounce when reaching the switchover point?

As you might imagine, most of the discussion here is qualitative and there's very little A/B testing involved.

Implement: Stage 2 The next stage is the rapid development of the core viral loop. This part should hopefully take days or weeks, not months. It will also certainly be wrong. The best advice I can give here is to follow agile development models and to build the smallest number of features and pages to create the initial flow of pages.

As mentioned before, the best implementations are strongly tied to the core product - as a result, if you're a video site, it's best if you can somehow involve videos. If you're a dating site, you probably want to involve dating.

The other implementation advice I'll give is to treat the viral loop code as an iterative, protoyping process. So copy and paste all you need, keep it in a separate codebase, and make it easy to refactor. You'll need to do a lot of messy stuff like changing the order of pages or page elements later, and once you develop your own recipe, it's easy to rewrite it in the "right way."

Launch: Stage 3 The next step is to beg, borrow, or steal traffic 😃 The easiest way is often to pay for it, $50/day or so, just so you have a trickle of traffic coming in.

Optimize: Stage 4 As you get a flow of incoming traffic, this allows you to deeply optimize the experience. This will involve building out some basic infrastructure to do A/B testing, or using Google Web Optimizer, and otherwise. The key thing here, of course, is to measure whether or not the $50/day you're spending results in traffic above and beyond what you're paying for - the more the better, and eventually you'll cross the threshold where traffic scales infinitely.

In this stage, there are a lot of common fixes that you'll want to consider:

- Shortening the flow of pages (can you shrink a 5 page funnel down to 2?)
- Rearranging UI elements to emphasize next steps
- Testing different value propositions for going through the flow
- Increasing the # of people invited

This optimization stage creates great conflict for product and customer-oriented people. Oftentimes, to get a number to move from 10% to 30%, there's temptation to do things that users may not be happy with. This might include things like asking for invites multiple times throughout the initial session, presenting an opt-out process for selecting friends, etc. These are all bad and need to be fixed in order to create a long-term sustainable viral loop.

This optimization step can take a very long time (months is not uncommon) as you zero in on the dozens of small and large changes needed to create a viral loop.

After months of work, two outcomes can result:

- You don't reach your goal, and you're stuck on traffic
- You reach your goal, and your traffic is going bananas!

If you don't reach your goal, then it's time to stop your optimization process. Often the changes that result are just too small to drive substantial increases in metrics. Instead, you'll have to rework your entire value proposition, which means to either go back to Stage 2 or possibly Stage 1. This means you'll want to stop A/B testing and start building out a deeper featureset.

Refine: **Stage 5** If your optimization step was successful, your work is probably not done. The final step is polishing your viral loop.

This includes figuring out issues like:

- Making your loop as user-aligned as possible
- Building a pleasant user experience and removing unnecessary flows or page elements
- Refactoring the code to move it from prototype to production

- Integrating it into your core product in a way that makes sense

A lot of people are tempted to skip this polish step, but don't do it! Skipping this step means that your initial product experience will suck, or be offensive.

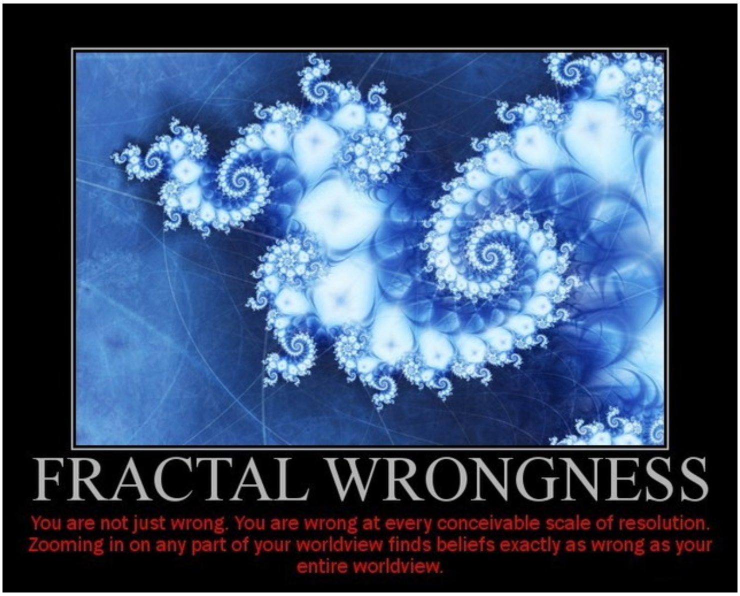
In fact, when there's "excess" virality, that's a great opportunity to make changes to the viral loop that make it nicer or friendlier. In general, if you are getting exponential growth, it'll be great even if it's a slower exponential. What's more important at that point is spendfing your extra growth towards changes that positively impact long-term retention.

On the other hand, if your product is just meant to be short-term mad money, then by all means skip this step 😃

More on viral loops and marketing For those that are interested, I've written more about viral loops and marketing here[57].

Off-topic Also, I found this image while searching for "Fractals" and thought it was funny enough to share:

[57] http://andrewchenblog.com/list-of-essays/#viral


FRACTAL WRONGNESS
You are not just wrong. You are wrong at every conceivable scale of resolution. Zooming in on any part of your worldview finds beliefs exactly as wrong as your entire worldview.

The question that got me to leave Seattle for greener startup pastures

Seattle is a great tech city Since I was 5 years old until 4 years after college, I called Seattle my home, and technology was intertwined with my childhood. As a kid, I found lots of avenues to my formative years in computing, including access to gopher and telnet via Seattle Community Network[58], the pre-web BBS scene[59], and a 5th grade classroom filled with Macs[60].

[58]http://www.scn.org/

[59]http://andrewchenblog.com/2009/08/25/bbs-door-games-social-gaming-innovation-from-the-1980s/

[60]http://andrewchenblog.com/2006/09/27/early-nerd-memories/

As a college student, I got to work at varioustech startups[61] and ended up at a VC firm[62] after I graduated. There's not a lot of cities that have the ecosystem to have given me opportunities like that - maybe half a dozen at the most, and Seattle is certainly high up on the list.

Ultimately though, I left after 2006 - it took a lot of soul searching but ultimately one question got me over the edge. Let me explain what that was.

The question that got me to leave Seattle As I pondered staying or leaving Seattle, I did a lot of thinking about the city from a startup context and what was working and not. Obviously it's great to have companies like Microsoft, Amazon, Real, and others there - it produced a wonderful tech ecosystem that is thriving and growing every day.

But in late-2006, the social networking world had caught fire, and I wondered:

> Post-bubble, when was the last time Seattle produced a world-changing consumer internet company?

And try as I might, I couldn't shake the idea that while the rest of the tech world in California was producing YouTube, MySpace, Facebook, Google, and others, Seattle had Amazon and sort of stopped.

I wasn't sure that I would be able to answer WHY, but I packed my bags and figured I'd figure out a theory at some point. A few

[61]http://cobaltgroup.com/

[62]http://mdv.com

years later, thinking about the question now, I think it has a lot to do with the kinds of companies being built in Seattle.

Different kinds of companies - Commerce versus CommunityMy current hypothesis is that Seattle has a strong history in retail and commerce, which has influenced the kinds of companies that are started there. Obviously you have Amazon, but you also have Eddie Bauer, Blue Nile, Nordstrom, Costco, Starbucks, and numerous other online/offline retail businesses there. There are also lots of transaction-focused startups based in real estate (like Redfin) or travel (Expedia).

These retail and transactionally-focused businesses are great money-makers, but because they target in-market buyers for a particular good or service, it means that you're not really building a huge audience. You end up with the <10% of the general population that is in-market for buying a diamond or plane tickets or a house, not a viral and sticky UGC site you visit every day.

The classic way to build a huge audience is to focus on ad-driven businesses in the world of communication or content publishing, and there just aren't that many of them in Seattle. (Though congrats to the Ben Huh for marching his horde of cats in this direction - the Cheezburger sites have the #1 traffic slot[63] in Seattle right now) If you look at categories like social networking or YouTube or Twitter, these are more like everyday tools that hundreds of millions of people might use every day to communicate or find the content they want. Those are mass audience driven businesses and end up being high-variance outcomes - you end up with huge hits and also

[63]http://www.seattle20.com/startup-index.aspx

big failures because you need more money-losing years to build up the audience necessary to monetize at the rates you want. (just look at Imeem's recent firesale even as they had amassed tens of millions of active users)

Different types of expertise - SEO versus viral/socialSimilarly, the above influence also drives the skillset involved for one of the key startup goals: Driving traffic. My working hypothesis for Seattle is that it's a very strong SEO-oriented community, and you have many of the top experts living and working there. The reason, of course, is that retail and transactional sites are mostly found via Google, and it makes sense to develop a skillset around getting that traffic for free rather than paying the search engine for it.

That's great, but that also closes the door for the all-important knowledge of the viral loop[64] that companies in social gaming are learning now, and what social networks companies learned before them.

For that reason, much of the social gaming and social network action happens down in the Bay Area.

Comments?In short, years later I think I've mostly answered my own question - my hypothesis is that Seattle hasn't produced mass audience consumer products mainly because it's focused on down-to-earth charge-users-for-a-product types of businesses that are more transactional than community. I don't think that's a good or bad thing - just as you'll get more biotech in Boston, there's a specialization in Seattle around commerce/retail. But if you're doing a social UGC thing, the

[64]http://andrewchenblog.com/2007/07/11/whats-your-viral-loop-understanding-the-engine-of-adoption/

Bay area is the best place to be.

Seattle folks (or otherwise): Do you agree or disagree with the above? Let me know in the comments - would enjoy hearing your thoughts.

UPDATE: For all the people who think I'm being a Seattle-hater, here's a similar analysis for the Bay Area: Does Silicon Valley noise detract from long term value creation?[65] It's a related piece and discusses some of what I've noted since being down in SF.

Does every startup need a Steve Jobs?

[65]http://andrewchenblog.com/2009/07/27/does-silicon-valley-noise-detract-from-long-term-value-creation/

What does Steve Jobs really do for Apple? I had a recent conversation on Apple's incredible design culture and what it would take to create that in a startup. In many ways, it seems like an insurmountably difficult challenge to play the role of Steve Jobs, with his god-like sense of product aesthetics and interactions.

And yet, Apple has hundreds of products and experiences - hardware, software, HR materials[66], commercials[67], etc. Steve Jobs certainly doesn't have time to work on the design of every Apple product, and of course has 35,000 employees to manage. So what does Steve Jobs really do, to create the amazing design culture at Apple?

And more importantly, can a startup hope to even start to capture the same kind of culture?

Well, let me give you my best guess 😀

IDEO's product framework for Desirability, Feasibility, and Viability First, let's take a quick detour and talk about IDEO's perspective on new product development - this is documented as part of their 100+ PDF on human centered design[68], but also recounted to me by my patient girlfriend who works there.

The idea is that all products ultimately come from an epic struggle between three perspectives: Desirability, Feasibility, and Viability. IDEO focuses on new products from the desirability side, which means they think about how to make sexy products with clear value propositions, and think technology and business

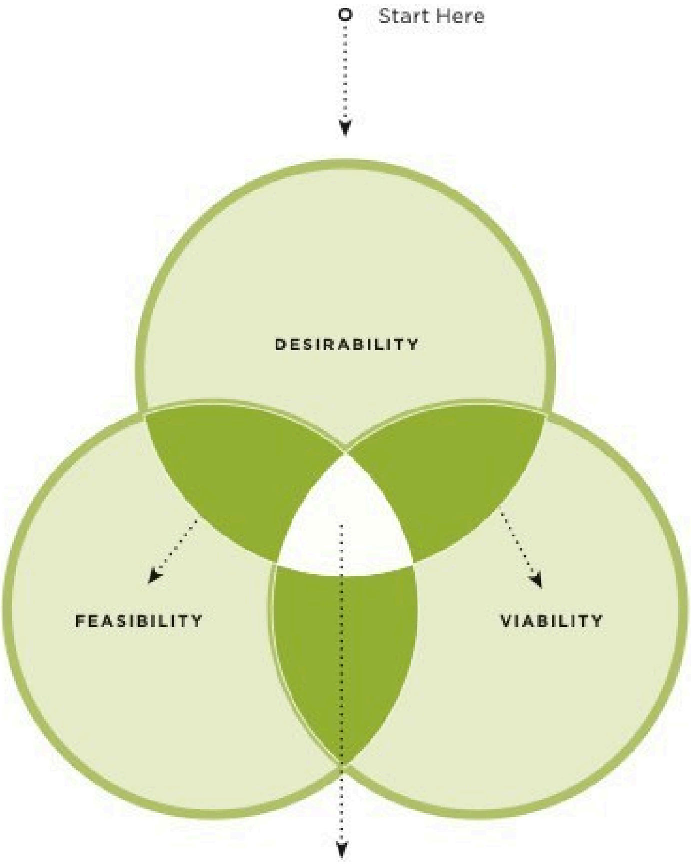
[66]http://www.macrumors.com/2009/10/05/apple-job-offer-unboxing-pictures-posted/

[67]http://www.youtube.com/watch?v=OYecfV3ubP8

[68]http://bit.ly/4B1GBI

goals flow from that. Most of their Fortune 500 clients do not act this way, of course, which is why they have to hire IDEO.

Here's the diagram included in their HCD toolkit:



The solutions that emerge at the end of the Human-Centered Design should hit the overlap of these three lenses; they need to be **Desirable, Feasible, and Viable.**

The way this was retold to me is that these factors map into functional parts of a business:

- Viability = Business focus (marketing, finance)
- Feasibility = Engineering focus (technologies, agile process, etc.)
- Desirability = Design focus (customers, aesthetics, etc.)

Business-focused product perspective: **Viability** For business-oriented products, the focus might be on any of the following:

- “hot markets”
- making money
- funding potential
- distribution
- metrics

The idea there is that you get to a product via one of these first-order items. A business-oriented entrepreneur might identify a market, then try to come up with a product within the market - for example, “wow, Zynga is making $250M/year, and fish games are big. I should come up with a social gaming product too.”

I would also argue that “corporate” thinking (including MBAs and biz plan competitions) fundamentally revolve around this approach - the most important thing becomes the analytical discussion around the business, rather than the core user experience itself. Financial metrics and market sizes become the dominating point of discussion - I would argue also that most venture capitalists fall into this bucket.

The big "religions" in this perspective are frameworks like Built to Last, Crossing the Chasm, Customer Development, Blue Ocean Strategy, even Efficient Market Hypothesis. You might also count Six Sigma, all the stuff in McKinsey quarterlies, etc.

Engineering-focused product perspective: **Feasibility** For technology-oriented products, the focus might be on the following:

- programming language and development stack
- cool technologies or libraries
- engineering processes (agile or otherwise)

For people who use this as a first-order filter, you might end up with a line of thinking like, "BitTorrent is really cool, how do we build a business around it?"

I would also put engineering processes like agile into this, because that can easily become a first-order item in how to build a product as well. Agile won't work for every team, for every product, in every situation, and yet it's viewed as an all-purpose hammer - does that really make sense?

The big "religions" in this perspective are frameworks are agile, scrum, open source, etc. I might also count the "ecosystems" like Rails as a unique culture with its own set of beliefs and conventions. Frameworks like "Lean Startups" ultimately combine both

Business and Engineering goals, via Customer Development plus Agile.

Design-focused product perspective: Desirability For design-focused products, the focus might be on:

- context, culture, and goals
- customer goals and product experience
- design aesthetics and interactions

The first-order filter in this case might be “Sick people go to hospitals and have a terrible experience - how do we improve that?” The tools employed at this initial stage might include user research, development of personas and user goals, and rapid prototyping to explore many product concepts.

The big “religions” here are led by Apple and their aesthetics and standards. And of course folks like IDEO and their “design thinking” ideas.

How business and engineering goals encroach on the desirability of a product Reading through the above, perhaps you have identified yourself as prioritizing one versus the other. And in general, the prioritization of the three different goals drives what kinds of product experiences you can build.

From the perspective of making a sexy, highly desirable product, you’ll find lots of objections from business or engineering:

- “spending money on visual design is too expensive”
- “polishing a product will make the process too slow”

- “this product is boring to implement”
- “can you redesign this product so we can build it in 1 week sprints?”
- “this target user is great, but we want the product to be more powerful and support more audiences”
- “but Zynga doesn’t do this, can you just copy them?”
- “why build so many prototypes that get thrown away? That’s costly and slow”
- “if you added X to this product, it would put us into strategic market Y”
- etc.

How do you handle questions like the above?

All of them are great questions, and of course the right answer means you have to find a balance in the approach. But what is the expense towards the core of your product experience?

Back to Steve Jobs - what does he really do? Long story short, my hypothesis is that Steve Jobs is one of the rare CEOs who is very focused on product desirability. In battles with the business and technology goals, desirability will almost always win out.

So his role isn’t that of a designer, but rather **Chief Design Advocate**. This means:

- he makes it clear that products should be “insanely great”
- he recruits a top design team, and protects them from competing goals

- he is willing to spend money, adjust technology processes, all for the goal of highly desirable products
- he convinces financial analysts, industry pundits, etc. that product design is very important

To me, the amazing part about this is: **Any company can do it**.

Maybe not as good as Jobs, but they can decide to make it a priority - but few companies do. With the pressure of quarterly earnings, what competitors are doing, and employee aspirational desires, the focus moves off of killer experiences for customers - that's no good.

If the above is true, then any of us can be the Steve Jobs of our team. Start by prioritizing design and desirability, and place it on a better footing relative to engineering and business goals. Learn the tools, develop your own religion, and start building great product experiences.

It almost sounds so easy!