Peter Thiel's CS183: Startup - Class 5 Notes Essay

Here is an essay version of my class notes from Class 5 of CS183: Startup. Errors and omissions are mine.

Stephen Cohen, co-founder and Executive VP of Palantir Technologies, and Max Levchin of PayPal and Slide fame joined this class as guest speakers. Credit for good stuff goes to them and Peter. I have tried to be accurate. But note that this is not a transcript of the conversation.

CS183: Startup—Notes Essay—The Mechanics of Mafia

I. Company Cultures

Everybody knows that company culture is important. But it's hard to know exactly what makes for an ideal culture. There are obviously some things that work. Even though they didn't necessarily look like a winning investment at the time, the early Microsoft team clearly got something right.

Then there are some things that don't work so well. A cult is perhaps the paradigmatic version of a culture that doesn't work. Cults are crazy and idealistic in a bad way. Cult members all tend to be fanatically wrong about something big.

And then there is what might be called anti-culture, where you really don't even have a culture at all. Consulting firms are the classic example here. Unfortunately, this is probably the dominant paradigm for companies. Most of the time, they don't even get to the point of having culture. People are mercenaries. People are nihilistic.

Picture a 1-dimensional axis from consultant-nihilism to cultish dogmatism. You want to be somewhere in the middle of that spectrum. To the extent you gravitate towards an extreme, you probably want to be closer to being a cult than being an army of consultants.

Good company culture is more nuanced than simple homogeneity or heterogeneity. On the homogeneity side, everyone being alike isn't enough. A robust company culture is one in which people have something in common that*distinguishes them quite sharply from rest of the world*. If everybody likes ice cream, that probably doesn't matter. If the core people share a relevant and unique philosophy about something important, you're onto something.

Similarly, differences qua differences don't matter much. In strong company cultures, people are different in a way that goes to the core mission. Suppose one key person is on an ice cream only diet. That's quirky. But it's also irrelevant. You want your people to be different in a way that gives the company a strong sense of identity and yet still dovetails with the overall mission. Having different kinds of problem-solvers on a team, for example, can make for a stronger culture.

II. Zero Sum vs. Not

A. To Fight or Not To Fight

Generally speaking, capitalism and competition are better seen as antonyms than as synonyms. To compete isn't what you should set out to do. That doesn't mean you should slack off. To succeed you probably need to work intensely. But you should work on something that others aren't doing. That is, focus on an area that's not zero-sum.

Sometimes, though, you need to compete. Monopoly is the theoretical ideal that you should always pursue. But you won't always find some non-competitive, cornucopian world. You may well find yourself in competitive, zero-sum situations. You must be prepared to handle that competition.

Gandhi is a great historical figure. He had many virtues. But he probably would not have been a great startup advisor. Consider the following quote:

> *"If [Hitler and Mussolini] choose to occupy your homes, you will vacate them. If they do not give you free passage out, you will allow yourselves, man, woman, and child, to be slaughtered, but you will refuse to owe allegiance to them."*

Basically, the message is that you should demonstrate your superiority by allowing yourself to be slaughtered. Do not follow that advice while starting companies. You should try to avoid fighting, but where you must, you should fight and win.

B. Creators or Fighters?

In thinking about building good company culture, it may be helpful to dichotomize two extreme personality types: nerds and athletes. Engineers and STEM people tend to be highly intelligent, good at problem solving, and naturally non zero-sum. Athletes tend to be highly motivated fighters; you only win if the other guy loses. Sports can be seen as classically competitive, antagonistic, zero-sum training. Sometimes, with

martial arts and such, the sport is literally fighting.

Even assuming everyone is technically competent, the problem with company made up of nothing but athletes is that it will be biased towards competing. Athletes like competition because, historically, they've been good at it. So they'll identify areas where there is tons of competition and jump into the fray.

The problem with company made up of nothing but nerds is that it will ignore the fact that there may be situations where you have to fight. So when those situations arise, the nerds will be crushed by their own naiveté.

So you have to strike the right balance between nerds and athletes. Neither extreme is optimal. Consider a 2 x 2 matrix. On the y-axis you have zero-sum people and non zero-sum people. On the x-axis you have warring, competitive environments (think Indian food joints on Castro Street or art galleries in Palo Alto) and then you have peaceful, monopoly/capitalist environments.

Most startups are run by non-zero sum people. They believe world is cornucopian. That's good. But even these people tend to pick competitive, warring fields because they don't know any better. So they get slaughtered. The nerds just don't realize that they've decided to fight a war until it's all over.

The optimal spot on the matrix is monopoly capitalism with some tailored combination of zero-sum and non zero-sum oriented people. You want to pick an environment where you don't have to fight. But you should bring along some good fighters to protect your non zero-sum people and mission, just in case.

C. Investor Heuristics

Founders Fund is a picky VC firm. There are many different types of companies that it doesn't like. The partners have developed maybe 20 or so different dogmas, each taking the form "Never invest in *x*." The "*x*" might be mobile internet, cleantech, etc. Sometimes it seems like there are so many dogmas that it's impossible to invest in anything anymore.

But always up for contrarian thinking, awhile back they made up a new strategy: identify and invest in the best company in or for every particular dogma. It's been more useful as a thought experiment than an actual strategy. But it led them to look at an interesting cleantech company they would've ordinarily skipped over. Though the space is extremely competitive and no one ever really makes any money, this particular company seemed reasonably good. It was run by scientists. It had great engineers and great technology. Everybody was passionate and committed to the mission. Talks of term sheets were in the air.

But then the cap table surfaced. It turned out that the founders and employees owned about 20% of the company. Other VC firms owned 80%. At the time, the company had a $35m valuation, so it was still early stage. The equity breakdown seemed more like a mistake than a red flag. Many versions of the "what the hell happened?!" question were asked. The founders' response was nonchalant: "We are so committed to making the technology work that we didn't care about the equity." That may be a very noble. But it's also pretty bad. The subquestions it raised killed the deal: with such passivity, what are you going to do about your competitors? Can you even build a sales team? If you got run over so hard by*investors*, how are you going to fare against the entire world?

III. A Conversation with Stephen Cohen and Max Levchin

Peter Thiel: You guys have started companies. You've seen what's worked and what hasn't. Talk for a few minutes each. How do you build culture?

Stephen Cohen: Palantir makes analysis platforms aimed at governmental clients. But the founders knew from the outset that they ultimately wanted to make products for enterprise generally as well. Since that would take a long time to pull off, they knew that they needed really brilliant people working together under a shared long-term perspective. They knew that hiring tightly and wisely would be crucial from day one.

That early understanding reflected the three salient properties that inhere in good company culture. First, a company must have very talented people. Second, they must have a long-term time orientation. Third, there must what might be called a generative spirit, where people are constantly creating. With this framework, hiring is more understandable: you just find people who have or contribute to all three properties. Culture is the super-structure to choose and channel people's energies in the right direction.

One error people make is assuming that culture *creates* these three aspects. Take a look at the Netflix company culture slides, for instance. They seem to indicate that you can produce talent from non-talent, or that you can take someone focused on the now and somehow transform them into long-term thinking. But you can't. Culture can always do more harm than good. It can reflect and enhance these three properties. It cannot create them.

From that insight comes the conclusion that hiring is absolutely critical. People you don't hire matter more than people you do hire. You might think that bad hiring decisions won't matter that much, since you can just fire the bad people. But Stalin-esque meritocracy sucks. Yes, you can shoot the bad people in the back of the head. But the problem with that is that you're still shooting people in the back of the head.

Peter Thiel: One early goal at PayPal was never to fire anybody. The founders just hired their friends since they could trust them. But eventually they had to hire more and more people who they knew less and less. They hired a sys admin from outside their network. It was trouble from the beginning; the guy showed up at 6pm on his first day of work. Worse than his tardiness was his lack of hygiene. The near-immediate objections people had were silenced by the founding rule: never fire anybody. A couple of months later, PayPal's systems crashed. The squalid sys admin hadn't made any backups. For a moment it looked like PayPal was done for. Luckily, some engineer went outside his job description and had decided to secretly back up everything everyday. Order was restored and the sys admin was fired. The "no-fire" rule still reflects a good orientation: firing people is like war, and war is bad, so you should try not to do it. But the flipside is that if you wait until it's obvious to everyone that someone should be fired, it's far too late.

Max Levchin: The notion that diversity in an early team is important or good is completely wrong. You should try to make the early team as non-diverse as possible. There are a few reasons for this. The most salient is that, as a startup, you're underfunded and undermanned. It's a big disadvantage; not only are you probably getting into trouble, but you don't even know what trouble that may be. Speed is your only weapon. All you have is speed.

So how do you move fast? If you're alone, you just work really hard and hope it's enough. Since it often isn't, people form teams. But in a team, an *n*-squared communications problem emerges. In a five-person

team, there are something like 25 pairwise relationships to manage and communications to maintain. The more diverse the early group, the harder it is for people to find common ground.

The early PayPal team was four people from the University of Illinois and two from Stanford. There was the obligatory Russian Jew, an Asian kid, and a bunch of white guys. None of that mattered. What mattered was that they were not diverse in any important way. Quite the contrary: They were all nerds. They went to good schools. (The Illinois guys had done the exact same CS curriculum.) They read sci-fi. And they knew how to build stuff. Interesting to note is that they did *not*know how to build stuff the right way. It turned out that scaling up would be very challenging for PayPal because the 26 year-olds who were managing hundreds of thousands of credit cards didn't make all the optimal choices from the beginning. But there was great clarity in the early communications. There was no debate on how to build that first database. And that alone made it possible to build it.

Striving for optimality early on—debating pros and cons of various design decisions in intricate detail—would have doomed PayPal. When systems problems finally caught up to them, their communication was so good that they were able to fix them reasonably quickly. They kept hiring people from Illinois and Stanford. They focused on their network. And things worked out. But only because of a lack of diversity.

PayPal once rejected a candidate who aced all the engineering tests because for fun, the guy said that he liked to play hoops. That single sentence lost him the job. No PayPal people would ever have used the world "hoops." Probably no one even knew how to play "hoops." Basketball would be bad enough. But "hoops?" That guy clearly wouldn't have fit in. He'd have had to explain to the team why he was going to go play hoops on a Thursday night. And no one would have understood him.

PayPal also had a hard time hiring women. An outsider might think that the PayPal guys bought into the stereotype that women don't do CS. But that's not true at all. The truth is that PayPal had trouble hiring women because PayPal was just a bunch of nerds! They never talked to women. So how were they supposed to interact with and hire them?

One good hiring maxim is: whenever there's any doubt, there's no doubt. It's a good heuristic. More often than not, any doubt precluded a hire. But once this very impressive woman came to interview. There were some doubts, since she seemed reluctant to solve a coding problem. But her talk and demeanor—she insisted on being interviewed over a ping-pong game, for instance—indicated that she'd fit into the ubernerd, ubercoder culture. She turned out to be reasonably good at ping-pong. Doubts were suppressed. That was a mistake. She turned out to not know how to code. She was a competent manager but a cultural outsider. PayPal was a place where the younger engineers could and would sometimes wrestle with each other on the floor to solve disputes! If you didn't get the odd mix of nerdiness + alpha maleness, you just stuck out.

Stephen Cohen: Good stuff shows itself. Talent shows itself. It doesn't talk about itself. You must develop a sort of spidey sense to look out for it. Watch what people show you instead of listening to what they're telling you. Seize on any doubt you find. It's never personal. Never let the interview process become personal. But things get personal if you just listen to the other person. Don't ask yourself what you think about what the candidate is saying. Just imagine the person you're interviewing at work. Imagine them in a situation they'd be in if you were to hire them. How does *that* look?

Screening out personal biases is a must. A lot of programmers are dogmatic about syntax. Things have got to be laid out this particular way. Maybe they don't like using factoring methods or something. But that's a personal bias. It has nothing to do with being a good engineer. So those are the wrong questions to focus on. The right question is how badass they are. Smooth appearances are irrelevant to being good. The most talented folks are almost always quirky. Watch for the quirks and embrace them. Nothing is stranger than watching a quirky entrepreneur harshly criticize another quirky entrepreneur for being too quirky.

A specific application of this is the anti-fashion bias. You shouldn't judge people by the stylishness of their clothing; quality people often do not have quality clothing. Which leads to a general observation: Great engineers don't wear designer jeans. So if you're interviewing an engineer, look at his jeans. There are always exceptions, of course. But it's a surprisingly good heuristic.

Max Levchin: The management team at PayPal was very frequently incompatible. Management meetings were not harmonious. Board meetings were even worse. They were certainly productive meetings. Decisions were made and things got done. But people got called idiots if they deserved it.

The next time around, at Slide, we tried to create a nicer environment. The idea of having meetings where people really liked one another seemed great. That was folly. The mistake was to conflate anger with a lack of respect. People who are smart and energetic are often angry. Not at each other, usually. Rather, they're angry that we're "not there yet," i.e. that they have to solve *x* when they should be working on some greater problem *y*. Disharmony at PayPal was actually a side effect of very healthy dynamics.

If people complain about people behind each other's backs, you have a problem. If people don't trust each other to do good work, you have a problem. But if people know that their teammates are going to deliver, you're good. Even if they are all calling each other idiots.

The danger is that you get soft. It's hard not to get soft as you train niceties. Pretty soon you spend more time thinking about how nice everyone is than you do about how qualified they are. That is death. If you think that an A- or B++++ person becomes an A person if they have a good personality, you are an idiot. The rest of the organization has already figured out that you're just being soft. They won't respect the non-A player. And they certainly won't respect you.

Even though people would physically fight on the engineering room floor, if you ever asked PayPal people if they respected each other, the answer was obvious. For a very long time, everyone believed in everyone else. That was not true at Slide. There, the subtle passive-aggressive lack of respect was allowed to develop too long. It proved very costly. At some point, there had to be a relatively significant bloodletting. It was stressful. The victims of the purge were so nice. It was easy to like them. It felt like a very bad, mean thing to do. But it was a good decision. Subsequently the company was run and performed much better. Yes, the love dissipated. But you knew that whatever remained was rooted in respect.

Peter Thiel: It is incredibly important to surface issues quickly. Ideally everyone in an organization is rowing in same direction. Ideally there's a strong, shared vision of the company's future. But at the micro level, details matter a lot. People will disagree about them often. When that happens, it simply must surface. Concealing disagreements because people feel uncomfortable makes for disaster. It doesn't fix things. They just sit undealt with, doing damage. Even in best of startups, a lot of chaotic things happen. If disagreements aren't surfacing, it's not because there are none. Key things are being covered up.

Everyone moving together in lockstep is a big red flag, not an ideal.

The standard view is that companies get destroyed by external competition. Maybe that's true in the long run. But in the short run—and most that fail fail in the short run—they get destroyed internally. Even the best companies have ups and downs. If destructive relationships unravel and wreck havoc during a down, the whole ship can blow up. Companies are not simple unitary entities in larger competitive ecosystems. They are complex entities with complex dynamics. Usually those dynamics blow up before some predator from the wider ecosystem strikes.

Stephen Cohen: You need to avoid people who are likely to blow things up. One key question to ask is: how does this person see themselves? One trendy answer that people seem to have is: I see myself as Steve jobs. Absent context, someone seeing himself as the next Steve Jobs is neither bad nor good. It just is. But in context, it might be a disaster. If you have a team of 10 people trying to build product consensus, imagine what happens if all of them are Steve Jobs. It'd be a nightmare. At best you'd have nine pissed off people and one very insecure guy who got his way.

It's often telling to ask someone why they made the major decisions they did in the past. You can tell if they've processed the emotions behind those decisions. Someone who gets flustered or can't explain a job change may be carrying a lot of baggage. Someone who doesn't take responsibility for past moves will probably not change course and take responsibility in the future.

Max Levchin: Another good interviewing heuristic is to be very wary of salary negotiators. That you should run away from anyone who just wants salary instead of equity is entirely obvious. There is some nuance here, since a lot of people got burned on options during that last boom/bust cycle. But generally you want people to want stock. The best hires don't seem to care too much about money at all. They might ask whether a certain salary is market or not. That is reasonable; no one wants to get screwed. But you want people to care far more about equity. And best hires aren't wooed by an offer of a large number of shares. The best hires say "That's the numerator. What's the denominator?" The best people are the ones who care to ask: How much of the company is mine?

Some companies are sales-driven. You need to hire good salespeople. But that's hard to do, since those people are trained to sell. When they walk in the door, you're getting overwhelmed by phenomenal sales skills. It's hard to know what's real and what's not. So what should you do? The same thing people do for engineers: give them technical questions. Break them. Watch what happens when they break. You'll use lateral-thinking problems instead of algorithms questions, of course. But good sales people are just as smart as engineers, so you shouldn't give them a free pass. You need to build a team that has a lot of raw intelligence. So never slack on interviewing intensity just because the job isn't a technical one.

Peter Thiel: A good thing to do when hiring sales people is to see how much they've sold in the past. But you have to apply some sort of discount rate because they don't always tell the exact truth. Scott Bannister of IronPort just asked sales people to submit their W-2s. Those with proclivity for exaggerating couldn't stump simple test of how much they'd made in commissions in the past.

Question from audience: Suppose you found a great engineer that's a good cultural fit. They are in

high demand. How do make a compensation package that ensures you get that person?

Stephen Cohen: There's a crazy phenomenon with engineers. There is probably some sum of money you could pay to any engineer to work at Palantir and give it their all for one year. But there is no sum of money that you could pay any engineer to go all-out for ten years. Humans can't muster that amount of sustained focus and energy if they don't love what they're doing. The folks who fall in love aren't asking details about salary, trying to extract every penny. The ones who fall in love are just running. So insofar as money is an issue, you should get at exactly why. What does some particular compensation detail mean to the person?

Question: What if engineers are in love with something else, but you think they'd fall in love with your company if they were to join you?

Stephen Cohen: Reframe that question in a marriage context. Don't you think that would make for a higher than normal rate of divorce?

Peter Thiel: One thing that's undervalued in the engineering world is over how long a term most of the value is built. When eBay bought PayPal, all the PayPal engineers left. eBay had to hire them all back as consultants at something like 3x their old salaries because it couldn't manage the codebase without them. The engineers had acquired a tremendous amount of knowledge of PayPal's systems. Even really smart engineers couldn't replace them. So it's worth targeting people who will be around a long time.

The surest way to blow up a company is with a nuclear bomb: send out an e-mail to everybody that lists what each person is getting paid. You should not actually try this experiment. But it's worth doing it as a thought experiment. People will always be upset when they see what others are getting paid. That's a given. But *how upset* will they be? Will they be extremely upset? Would that be justified? Or could they be persuaded that things are quite reasonable?

Engineering compensation is difficult right now. You're competing with Google's prestige and money. The first step is to avoid competing on purely financial terms, where you're likely to lose. You have to have that compelling monopoly narrative that we discussed last week.

Max Levchin: Engineers are very cynical people. They're trained to be. And they can afford to be, given the large number of companies that are trying to recruit them in Silicon Valley right now. Since engineers think any new idea is dumb, they will tend to think that your new idea is dumb. They get paid a lot at Google doing some pretty cool stuff. Why stop indexing the world to go do your dumb thing?

So the way to compete against the giants is not with money. Google will outbid you. They have oil derrick that spits out $30bn in search revenue every year. To win, you need to tell a story about cogs. At Google, you're a cog. Whereas with me, you're an instrumental piece of this great thing that we'll build together. Articulate the vision. Don't even try to pay well. Meet people's cash flow needs. Pay them so they can cover their rent and go out every once in awhile. It's not about cash. It's about breaking through the wall of cynicism. It's about making 1% of this new thing way more exciting than a couple hundred grand and a cubicle at Google.

Stephen Cohen: We tend to massively underestimate the compounding returns of intelligence. As humans, we need to solve big problems. If you graduate Stanford at 22 and Google recruits you, you'll work a 9-to-5. It's probably more like an 11-to-3 in terms of hard work. They'll pay well. It's relaxing. But what they are actually doing is paying you to accept a much lower intellectual growth rate. When you recognize that intelligence is compounding, the cost of that missing long-term compounding is enormous. *They're not giving you the best opportunity of your life*. Then a scary thing can happen: You might realize one day that you've lost your competitive edge. You won't be the best anymore. You won't be able to fall in love with new stuff. Things are cushy where you are. You get complacent and stall. So, run your prospective engineering hires through that narrative. Then show them the alternative: working at your startup.

Question: How does one preserve diversity of opinions in a startup?

Max Levchin: Sometimes diversity of opinion is valuable. Sometimes it's not. Some stuff needs to be off limits. There is some set of things that the founding team should decree is stupid to argue about. PayPal chose C++ early on. It's kind of crappy language. There's plenty to complain about. But the founding engineers never argued about it. Anyone that did want to argue about it wouldn't have fit in. Arguing would have impeded progress.

But arguing about smart marketing moves or different approaches to solving tactical or strategic problems is fundamental. These are the decisions that actually matter. Avoid groupthink in these areas is key. A good rule of thumb is that diversity of opinion is essential anytime you don't know anything about something important. But if there's a strong sense of what's right already, don't argue about it.

Peter Thiel: The relevant Keynes line here is "When the facts change, I change my mind. What do you do?" But you actually don't want to let every new fact call what you're doing into question. You're searching for a great business. What does that search space look like? Is it broad but shallow? Are you looking at every possible business you could do? Or are you focused on one area and drilling down on that?

The super broad, horizontal search is perhaps okay when you're thinking about starting a company initially. But returning to it at later stages is counterproductive. An internet company talking about being a cleantech company is lost. People tend to overrate the value of horizontal search and underestimate the sheer size of the search space. Far better is to understand how to do vertical search and to value depth over breadth.