The 18 Mistakes That Kill Startups

October 2006In the Q & A period after a recent talk, someone asked what made  
startups fail. After standing there gaping for a few seconds I  
realized this was kind of a trick question. It's equivalent to  
asking how to make a startup succeed — if you avoid every cause of  
failure, you succeed — and that's too big a question to answer on  
the fly.Afterwards I realized it could be helpful to look at the problem  
from this direction. If you have a list of all the things you  
shouldn't do, you can turn that into a recipe for succeeding just  
by negating. And this form of list may be more useful in practice.  
It's easier to catch yourself doing something you shouldn't than  
always to remember to do something you should.  
[1]In a sense there's just one mistake that kills startups: not making  
something users want. If you make something users want, you'll  
probably be fine, whatever else you do or don't do. And if you  
don't make something users want, then you're dead, whatever else  
you do or don't do. So really this is a list of 18 things that  
cause startups not to make something users want. Nearly all failure  
funnels through that.1. Single FounderHave you ever noticed how few successful startups were founded by  
just one person? Even companies you think of as having one founder,  
like Oracle, usually turn out to have more. It seems unlikely this  
is a coincidence.What's wrong with having one founder? To start with, it's a vote  
of no confidence. It probably means the founder couldn't talk any  
of his friends into starting the company with him. That's pretty  
alarming, because his friends are the ones who know him best.But even if the founder's friends were all wrong and the company  
is a good bet, he's still at a disadvantage. Starting a startup  
is too hard for one person. Even if you could do all the work  
yourself, you need colleagues to brainstorm with, to talk you out  
of stupid decisions, and to cheer you up when things go wrong.The last one might be the most important. The low points in a  
startup are so low that few could bear them alone. When you have  
multiple founders, esprit de corps binds them together in a way  
that seems to violate conservation laws. Each thinks "I can't let  
my friends down." This is one of the most powerful forces in human  
nature, and it's missing when there's just one founder.2. Bad LocationStartups prosper in some places and not others. Silicon Valley  
dominates, then Boston, then Seattle, Austin, Denver, and New York. After  
that there's not much. Even in New York the number of startups per  
capita is probably a 20th of what it is in Silicon Valley. In towns  
like Houston and Chicago and Detroit it's too small to measure.Why is the falloff so sharp? Probably for the same reason it is  
in other industries. What's the sixth largest fashion center in  
the US? The sixth largest center for oil, or finance, or publishing?  
Whatever they are they're probably so far from the top that it would  
be misleading even to call them centers.It's an interesting question why cities   
become startup hubs, but  
the reason startups prosper in them is probably the same as it is  
for any industry: that's where the experts are. Standards are  
higher; people are more sympathetic to what you're doing; the kind  
of people you want to hire want to live there; supporting industries  
are there; the people you run into in chance meetings are in the  
same business. Who knows exactly how these factors combine to boost  
startups in Silicon Valley and squish them in Detroit, but it's  
clear they do from the number of startups per capita in each.3. Marginal NicheMost of the groups that apply to Y Combinator suffer from a common  
problem: choosing a small, obscure niche in the hope of avoiding  
competition.If you watch little kids playing sports, you notice that below a  
certain age they're afraid of the ball. When the ball comes near  
them their instinct is to avoid it. I didn't make a lot of catches  
as an eight year old outfielder, because whenever a fly ball came  
my way, I used to close my eyes and hold my glove up more for  
protection than in the hope of catching it.Choosing a marginal project is the startup equivalent of my eight  
year old strategy for dealing with fly balls. If you make anything  
good, you're going to have competitors, so you may as well face  
that. You can only avoid competition by avoiding good ideas.I think this shrinking from big problems is mostly unconscious.  
It's not that people think of grand ideas but decide to pursue  
smaller ones because they seem safer. Your unconscious won't even  
let you think of grand ideas. So the solution may be to think about  
ideas without involving yourself. What would be a great idea for  
someone else to do as a startup?4. Derivative IdeaMany of the applications we get are imitations of some existing  
company. That's one source of ideas, but not the best. If you  
look at the origins of successful startups, few were started in  
imitation of some other startup. Where did they get their ideas?  
Usually from some specific, unsolved problem the founders identified.Our startup made software for making online stores. When we started  
it, there wasn't any; the few sites you could order from were  
hand-made at great expense by web consultants. We knew that if  
online shopping ever took off, these sites would have to be generated  
by software, so we wrote some. Pretty straightforward.It seems like the best problems to solve are ones that affect you  
personally. Apple happened because Steve Wozniak wanted a computer,  
Google because Larry and Sergey couldn't find stuff online, Hotmail  
because Sabeer Bhatia and Jack Smith couldn't exchange email at  
work.So instead of copying the Facebook, with some variation that the  
Facebook rightly ignored, look for ideas from the other direction.  
Instead of starting from companies and working back to the problems  
they solved, look for problems and imagine the company that might  
solve them.   
[2]  
What do people complain about? What do you wish there was?5. ObstinacyIn some fields the way to succeed is to have a vision of what you  
want to achieve, and to hold true to it no matter what setbacks you  
encounter. Starting startups is not one of them. The stick-to-your-vision  
approach works for something like winning an Olympic gold medal,  
where the problem is well-defined. Startups are more like science,  
where you need to follow the trail wherever it leads.So don't get too attached to your original plan, because it's  
probably wrong. Most successful startups end up doing something  
different than they originally intended — often so different that  
it doesn't even seem like the same company. You have to be prepared  
to see the better idea when it arrives. And the hardest part of  
that is often discarding your old idea.But openness to new ideas has to be tuned just right. Switching  
to a new idea every week will be equally fatal. Is there some kind  
of external test you can use? One is to ask whether the ideas  
represent some kind of progression. If in each new idea you're  
able to re-use most of what you built for the previous ones, then  
you're probably in a process that converges. Whereas if you keep  
restarting from scratch, that's a bad sign.Fortunately there's someone you can ask for advice: your users. If  
you're thinking about turning in some new direction and your users  
seem excited about it, it's probably a good bet.6. Hiring Bad ProgrammersI forgot to include this in the early versions of the list,  
because nearly all the founders I know are programmers. This is  
not a serious problem for them. They might accidentally hire someone  
bad, but it's not going to kill the company. In a pinch they can  
do whatever's required themselves.But when I think about what killed most of the startups in the  
e-commerce business back in the 90s, it was bad programmers. A lot  
of those companies were started by business guys who thought the  
way startups worked was that you had some clever idea and then hired  
programmers to implement it. That's actually much harder than it  
sounds — almost impossibly hard in fact — because business guys  
can't tell which are the good programmers. They don't even get a  
shot at the best ones, because no one really good wants a job  
implementing the vision of a business guy.In practice what happens is that the business guys choose people  
they think are good programmers (it says here on his resume that  
he's a Microsoft Certified Developer) but who aren't. Then they're  
mystified to find that their startup lumbers along like a World War  
II bomber while their competitors scream past like jet fighters.  
This kind of startup is in the same position as a big company,  
but without the advantages.So how do you pick good programmers if you're not a programmer? I  
don't think there's an answer. I was about to say you'd have to  
find a good programmer to help you hire people. But if you can't  
recognize good programmers, how would you even do that?7. Choosing the Wrong PlatformA related problem (since it tends to be done by bad programmers)  
is choosing the wrong platform. For example, I think a lot of  
startups during the Bubble killed themselves by deciding to build  
server-based applications on Windows. Hotmail was still running  
on FreeBSD for years after Microsoft bought it, presumably because  
Windows couldn't handle the load. If Hotmail's founders  
had chosen to use Windows, they would have been swamped.PayPal only just dodged this bullet. After they merged with X.com,  
the new CEO wanted to switch to Windows — even after PayPal cofounder  
Max Levchin showed that their software scaled only 1% as well on  
Windows as Unix. Fortunately for PayPal they switched CEOs instead.Platform is a vague word. It could mean an operating system, or a  
programming language, or a "framework" built on top of a programming  
language. It implies something that both supports and limits, like  
the foundation of a house.The scary thing about platforms is that there are always some that  
seem to outsiders to be fine, responsible choices and yet, like  
Windows in the 90s, will destroy you if you choose them. Java  
applets were probably the most spectacular example. This was  
supposed to be the new way of delivering applications. Presumably  
it killed just about 100% of the startups who believed that.How do you pick the right platforms? The usual way is to hire good  
programmers and let them choose. But there is a trick you could  
use if you're not a programmer: visit a top computer science  
department and see what they use in research projects.8. Slowness in LaunchingCompanies of all sizes have a hard time getting software done. It's  
intrinsic to the medium; software is always 85% done. It takes an  
effort of will to push through this and get something released to  
users.  
[3]Startups make all kinds of excuses for delaying their launch. Most  
are equivalent to the ones people use for procrastinating in everyday  
life. There's something that needs to happen first. Maybe. But  
if the software were 100% finished and ready to launch at the push  
of a button, would they still be waiting?One reason to launch quickly is that it forces you to actually  
finish some quantum of work. Nothing is truly finished till it's  
released; you can see that from the rush of work that's always  
involved in releasing anything, no matter how finished you thought  
it was. The other reason you need to launch is that it's only by  
bouncing your idea off users that you fully understand it.Several distinct problems manifest themselves as delays in launching:  
working too slowly; not truly understanding the problem; fear of  
having to deal with users; fear of being judged; working on too  
many different things; excessive perfectionism. Fortunately you  
can combat all of them by the simple expedient of forcing yourself  
to launch something fairly quickly.9. Launching Too EarlyLaunching too slowly has probably killed a hundred times more  
startups than launching too fast, but it is possible to launch too  
fast. The danger here is that you ruin your reputation. You launch  
something, the early adopters try it out, and if it's no good they  
may never come back.So what's the minimum you need to launch? We suggest startups think  
about what they plan to do, identify a core that's both (a) useful  
on its own and (b) something that can be incrementally expanded  
into the whole project, and then get that done as soon as possible.This is the same approach I (and many other programmers) use for  
writing software. Think about the overall goal, then start by  
writing the smallest subset of it that does anything useful. If  
it's a subset, you'll have to write it anyway, so in the worst case  
you won't be wasting your time. But more likely you'll find that  
implementing a working subset is both good for morale and helps you  
see more clearly what the rest should do.The early adopters you need to impress are fairly tolerant. They  
don't expect a newly launched product to do everything; it just has  
to do something.10. Having No Specific User in MindYou can't build things users like without understanding them. I  
mentioned earlier that the most successful startups seem to have  
begun by trying to solve a problem their founders had. Perhaps  
there's a rule here: perhaps you create wealth in proportion to how  
well you understand the problem you're solving, and the problems  
you understand best are your own.   
[4]That's just a theory. What's not a theory is the converse: if  
you're trying to solve problems you don't understand, you're hosed.And yet a surprising number of founders seem willing to  
assume that someone, they're not sure exactly who, will want what  
they're building. Do the founders want it? No, they're not the  
target market. Who is? Teenagers. People interested in local  
events (that one is a perennial tarpit). Or "business" users. What  
business users? Gas stations? Movie studios? Defense contractors?You can of course build something for users other than yourself.  
We did. But you should realize you're stepping into dangerous  
territory. You're flying on instruments, in effect, so you should  
(a) consciously shift gears, instead of assuming you can rely on  
your intuitions as you ordinarily would, and (b) look at the  
instruments.In this case the instruments are the users. When designing for  
other people you have to be empirical. You can no longer guess  
what will work; you have to find users and measure their responses.  
So if you're going to make something for teenagers or "business"  
users or some other group that doesn't include you, you have to be  
able to talk some specific ones into using what you're making. If  
you can't, you're on the wrong track.11. Raising Too Little MoneyMost successful startups take funding at some point. Like having  
more than one founder, it seems a good bet statistically. How much  
should you take, though?Startup funding is measured in time. Every startup that isn't  
profitable (meaning nearly all of them, initially) has a certain  
amount of time left before the money runs out and they have to stop.  
This is sometimes referred to as runway, as in "How much runway do  
you have left?" It's a good metaphor because it reminds you that  
when the money runs out you're going to be airborne or dead.Too little money means not enough to get airborne. What airborne  
means depends on the situation. Usually you have to advance to a  
visibly higher level: if all you have is an idea, a working prototype;  
if you have a prototype, launching; if you're launched, significant  
growth. It depends on investors, because until you're profitable  
that's who you have to convince.So if you take money from investors, you have to take enough to get  
to the next step, whatever that is.  
[5]  
Fortunately you have some  
control over both how much you spend and what the next step is. We  
advise startups to set both low, initially: spend practically  
nothing, and make your initial goal simply to build a solid prototype.  
This gives you maximum flexibility.12. Spending Too MuchIt's hard to distinguish spending too much from raising too little.  
If you run out of money, you could say either was the cause. The  
only way to decide which to call it is by comparison with other  
startups. If you raised five million and ran out of money, you  
probably spent too much.Burning through too much money is not as common as it used to be.  
Founders seem to have learned that lesson. Plus it keeps getting  
cheaper to start a startup. So as of this writing few startups  
spend too much. None of the ones we've funded have. (And not just  
because we make small investments; many have gone on to raise further  
rounds.)The classic way to burn through cash is by hiring a lot of people.  
This bites you twice: in addition to increasing your costs, it slows  
you down—so money that's getting consumed faster has to last  
longer. Most hackers understand why that happens; Fred Brooks  
explained it in The Mythical Man-Month.We have three general suggestions about hiring: (a) don't do it if  
you can avoid it, (b) pay people with equity rather than salary,  
not just to save money, but because you want the kind of people who  
are committed enough to prefer that, and (c) only hire people who  
are either going to write code or go out and get users, because  
those are the only things you need at first.13. Raising Too Much MoneyIt's obvious how too little money could kill you, but is there such  
a thing as having too much?Yes and no. The problem is not so much the money itself as what  
comes with it. As one VC who spoke at Y Combinator said, "Once you  
take several million dollars of my money, the clock is ticking."  
If VCs fund you, they're not going to let you just put the money  
in the bank and keep operating as two guys living on ramen. They  
want that money to go to work.   
[6]  
At the very least you'll move  
into proper office space and hire more people. That will change  
the atmosphere, and not entirely for the better. Now most of your  
people will be employees rather than founders. They won't be as  
committed; they'll need to be told what to do; they'll start to  
engage in office politics.When you raise a lot of money, your company moves to the suburbs  
and has kids.Perhaps more dangerously, once you take a lot of money it gets  
harder to change direction. Suppose your initial plan was to sell  
something to companies. After taking VC money you hire a sales  
force to do that. What happens now if you realize you should be  
making this for consumers instead of businesses? That's a completely  
different kind of selling. What happens, in practice, is that you  
don't realize that. The more people you have, the more you stay  
pointed in the same direction.Another drawback of large investments is the time they take. The  
time required to raise money grows with the amount.  
[7]  
When the  
amount rises into the millions, investors get very cautious. VCs  
never quite say yes or no; they just engage you in an apparently  
endless conversation. Raising VC scale investments is thus a huge  
time sink — more work, probably, than the startup itself. And you  
don't want to be spending all your time talking to investors while  
your competitors are spending theirs building things.We advise founders who go on to seek VC money to take the first  
reasonable deal they get. If you get an offer from a reputable  
firm at a reasonable valuation with no unusually onerous terms,  
just take it and get on with building the company.  
[8]  
Who cares  
if you could get a 30% better deal elsewhere? Economically, startups  
are an all-or-nothing game. Bargain-hunting among investors is a  
waste of time.14. Poor Investor ManagementAs a founder, you have to manage your investors. You shouldn't  
ignore them, because they may have useful insights. But neither  
should you let them run the company. That's supposed to be your  
job. If investors had sufficient vision to run the companies  
they fund, why didn't they start them?Pissing off investors by ignoring them is probably less dangerous  
than caving in to them. In our startup, we erred on the ignoring  
side. A lot of our energy got drained  
away in disputes with investors instead of going into the product.  
But this was less costly than giving in, which would probably have  
destroyed the company. If the founders know what they're doing,  
it's better to have half their attention focused on the product  
than the full attention of investors who don't.How hard you have to work on managing investors usually depends on  
how much money you've taken. When you raise VC-scale money, the  
investors get a great deal of control. If they have a board majority,  
they're literally your bosses. In the more common case, where  
founders and investors are equally represented and the deciding  
vote is cast by neutral outside directors, all the investors have  
to do is convince the outside directors and they control the company.If things go well, this shouldn't matter. So long as you seem to  
be advancing rapidly, most investors will leave you alone. But  
things don't always go smoothly in startups. Investors have made  
trouble even for the most successful companies. One of the most  
famous examples is Apple, whose board made a nearly fatal blunder  
in firing Steve Jobs. Apparently even Google got a lot of grief  
from their investors early on.15. Sacrificing Users to (Supposed) ProfitWhen I said at the beginning that if you make something users want,  
you'll be fine, you may have noticed I didn't mention anything about  
having the right business model. That's not because making money  
is unimportant. I'm not suggesting that founders start companies  
with no chance of making money in the hope of unloading them before  
they tank. The reason we tell founders not to worry about the  
business model initially is that making something people want is  
so much harder.I don't know why it's so hard to make something people want. It  
seems like it should be straightforward. But you can tell it must  
be hard by how few startups do it.Because making something people want is so much harder than making  
money from it, you should leave business models for later, just as  
you'd leave some trivial but messy feature for version 2. In version  
1, solve the core problem. And the core problem in a startup is  
how to create wealth   
(= how much people want something x the number  
who want it), not how to convert that wealth into money.The companies that win are the ones that put users first. Google,  
for example. They made search work, then worried about how to make  
money from it. And yet some startup founders still think it's  
irresponsible not to focus on the business model from the beginning.  
They're often encouraged in this by investors whose experience comes  
from less malleable industries.It is irresponsible not to think about business models. It's  
just ten times more irresponsible not to think about the product.16. Not Wanting to Get Your Hands DirtyNearly all programmers would rather spend their time writing code  
and have someone else handle the messy business of extracting money  
from it. And not just the lazy ones. Larry and Sergey apparently  
felt this way too at first. After developing their new search  
algorithm, the first thing they tried was to get some other company  
to buy it.Start a company? Yech. Most hackers would rather just have ideas.  
But as Larry and Sergey found, there's not much of a market for  
ideas. No one trusts an idea till you embody it in a product and  
use that to grow a user base. Then they'll pay big time.Maybe this will change, but I doubt it will change much. There's  
nothing like users for convincing acquirers. It's not just that  
the risk is decreased. The acquirers are human, and they have a  
hard time paying a bunch of young guys millions of dollars just for  
being clever. When the idea is embodied in a company with a lot  
of users, they can tell themselves they're buying the users rather  
than the cleverness, and this is easier for them to swallow.  
[9]If you're going to attract users, you'll probably have to get up  
from your computer and go find some. It's unpleasant work, but if  
you can make yourself do it you have a much greater chance of  
succeeding. In the first batch of startups we funded, in the summer  
of 2005, most of the founders spent all their time building their  
applications. But there was one who was away half the time talking  
to executives at cell phone companies, trying to arrange deals.  
Can you imagine anything more painful for a hacker?  
[10]  
But it  
paid off, because this startup seems the most successful of that  
group by an order of magnitude.If you want to start a startup, you have to face the fact that you  
can't just hack. At least one hacker will have to spend some of  
the time doing business stuff.17. Fights Between FoundersFights between founders are surprisingly common. About 20% of the  
startups we've funded have had a founder leave. It happens so often  
that we've reversed our attitude to vesting. We still don't require  
it, but now we advise founders to vest so there will be an orderly  
way for people to quit.A founder leaving doesn't necessarily kill a startup, though. Plenty  
of successful startups have had that happen.   
[11]  
Fortunately it's  
usually the least committed founder who leaves. If there are three  
founders and one who was lukewarm leaves, big deal. If you have  
two and one leaves, or a guy with critical technical skills leaves,  
that's more of a problem. But even that is survivable. Blogger  
got down to one person, and they bounced back.Most of the disputes I've seen between founders could have been  
avoided if they'd been more careful about who they started a company  
with. Most disputes are not due to the situation but the people.  
Which means they're inevitable. And most founders who've been  
burned by such disputes probably had misgivings, which they suppressed,  
when they started the company. Don't suppress misgivings. It's  
much easier to fix problems before the company is started than  
after. So don't include your housemate in your startup because  
he'd feel left out otherwise. Don't start a company with someone  
you dislike because they have some skill you need and you worry you  
won't find anyone else. The people are the most important ingredient  
in a startup, so don't compromise there.18. A Half-Hearted EffortThe failed startups you hear most about are the spectacular  
flameouts. Those are actually the elite of failures. The most  
common type is not the one that makes spectacular mistakes, but the  
one that doesn't do much of anything — the one we never even hear  
about, because it was some project a couple guys started on the  
side while working on their day jobs, but which never got anywhere  
and was gradually abandoned.Statistically, if you want to avoid failure, it would seem like the  
most important thing is to quit your day job. Most founders of  
failed startups don't quit their day jobs, and most founders of  
successful ones do. If startup failure were a disease, the CDC  
would be issuing bulletins warning people to avoid day jobs.Does that mean you should quit your day job? Not necessarily. I'm  
guessing here, but I'd guess that many of these would-be founders  
may not have the kind of determination it takes to start a company,  
and that in the back of their minds, they know it. The reason they  
don't invest more time in their startup is that they know it's a  
bad investment.  
[12]I'd also guess there's some band of people who could have succeeded  
if they'd taken the leap and done it full-time, but didn't. I have  
no idea how wide this band is, but if the winner/borderline/hopeless  
progression has the sort of distribution you'd expect, the number  
of people who could have made it, if they'd quit their day job, is  
probably an order of magnitude larger than the number who do make  
it.  
[13]If that's true, most startups that could succeed fail because the  
founders don't devote their whole efforts to them. That certainly  
accords with what I see out in the world. Most startups fail because  
they don't make something people want, and the reason most don't  
is that they don't try hard enough.In other words, starting startups is just like everything else.  
The biggest mistake you can make is not to try hard enough. To the  
extent there's a secret to success, it's not to be in denial about  
that.  
Notes[1]  
This is not a complete list of the causes of failure,  
just those you can control. There are also several you can't,  
notably ineptitude and bad luck.[2]  
Ironically, one variant of the Facebook that might work is a  
facebook exclusively for college students.[3]  
Steve Jobs tried to motivate people by saying "Real artists  
ship." This is a fine sentence, but unfortunately not true. Many  
famous works of art are unfinished. It's true in fields that have  
hard deadlines, like architecture and filmmaking, but even there  
people tend to be tweaking stuff till it's yanked out of their  
hands.[4]  
There's probably also a second factor: startup founders tend  
to be at the leading edge of technology, so problems they face are  
probably especially valuable.[5]  
You should take more than you think you'll need, maybe 50% to  
100% more, because software takes longer to write and deals longer  
to close than you expect.[6]  
Since people sometimes call us VCs, I should add that we're  
not. VCs invest large amounts of other people's money. We invest  
small amounts of our own, like angel investors.[7]  
Not linearly of course, or it would take forever to raise five  
million dollars. In practice it just feels like it takes forever.Though if you include the cases where VCs don't invest, it would  
literally take forever in the median case. And maybe we should,  
because the danger of chasing large investments is not just that  
they take a long time. That's the best case. The real danger  
is that you'll expend a lot of time and get nothing.[8]  
Some VCs will offer you an artificially low valuation to see  
if you have the balls to ask for more. It's lame that VCs play  
such games, but some do. If you're dealing with one of those you  
should push back on the valuation a bit.[9]  
Suppose YouTube's founders had gone to Google in 2005 and told  
them "Google Video is badly designed. Give us $10 million and we'll  
tell you all the mistakes you made." They would have gotten  
the royal raspberry. Eighteen months later Google paid $1.6 billion  
for the same lesson, partly because they could then tell themselves  
that they were buying a phenomenon, or a community, or some vague  
thing like that.I don't mean to be hard on Google. They did better than their  
competitors, who may have now missed the video boat entirely.[10]  
Yes, actually: dealing with the government. But phone companies  
are up there.[11]  
Many more than most people realize, because companies don't advertise  
this. Did you know Apple originally had three founders?[12]  
I'm not dissing these people. I don't have the determination  
myself. I've twice come close to starting startups since Viaweb,  
and both times I bailed because I realized that without the spur  
of poverty I just wasn't willing to endure the stress of a startup.[13]  
So how do you know whether you're in the category of people  
who should quit their day job, or the presumably larger one who  
shouldn't? I got to the point of saying that this was hard to judge  
for yourself and that you should seek outside advice, before realizing  
that that's what we do. We think of ourselves as investors, but  
viewed from the other direction Y Combinator is a service for  
advising people whether or not to quit their day job. We could be  
mistaken, and no doubt often are, but we do at least bet money on  
our conclusions.Thanks to Sam Altman, Jessica Livingston, Greg McAdoo, and Robert Morris   
for reading drafts of this.