Do Things that Don't Scale

July 2013One of the most common types of advice we give at Y Combinator is  
to do things that don't scale. A lot of would-be founders believe  
that startups either take off or don't. You build something, make  
it available, and if you've made a better mousetrap, people beat a  
path to your door as promised. Or they don't, in which case the  
market must not exist.  
[1]Actually startups take off because the founders make them take off.  
There may be a handful that just grew by themselves, but usually  
it takes some sort of push to get them going. A good metaphor would  
be the cranks that car engines had before they got electric starters.  
Once the engine was going, it would keep going, but there was a  
separate and laborious process to get it going.RecruitThe most common unscalable thing founders have to do at the start  
is to recruit users manually. Nearly all startups have to. You  
can't wait for users to come to you. You have to go out and get  
them.Stripe is one of the most successful startups we've funded, and the  
problem they solved was an urgent one. If anyone could have sat  
back and waited for users, it was Stripe. But in fact they're  
famous within YC for aggressive early user acquisition.Startups building things for other startups have a big pool of  
potential users in the other companies we've funded, and none took  
better advantage of it than Stripe. At YC we use the term "Collison  
installation" for the technique they invented. More diffident  
founders ask "Will you try our beta?" and if the answer is yes,  
they say "Great, we'll send you a link." But the Collison brothers  
weren't going to wait. When anyone agreed to try Stripe they'd say  
"Right then, give me your laptop" and set them up on the spot.There are two reasons founders resist going out and recruiting users  
individually. One is a combination of shyness and laziness. They'd  
rather sit at home writing code than go out and talk to a bunch of  
strangers and probably be rejected by most of them. But for a  
startup to succeed, at least one founder (usually the CEO) will  
have to spend a lot of time on sales and marketing.  
[2]The other reason founders ignore this path is that the absolute  
numbers seem so small at first. This can't be how the big, famous  
startups got started, they think. The mistake they make is to  
underestimate the power of compound growth. We encourage every  
startup to measure their progress by weekly growth  
rate. If you have 100 users, you need to get 10 more next week  
to grow 10% a week. And while 110 may not seem much better than  
100, if you keep growing at 10% a week you'll be surprised how big  
the numbers get. After a year you'll have 14,000 users, and after  
2 years you'll have 2 million.You'll be doing different things when you're acquiring users a  
thousand at a time, and growth has to slow down eventually. But  
if the market exists you can usually start by recruiting users  
manually and then gradually switch to less manual methods.   
[3]Airbnb is a classic example of this technique. Marketplaces are  
so hard to get rolling that you should expect to take heroic measures  
at first. In Airbnb's case, these consisted of going door to door  
in New York, recruiting new users and helping existing ones improve  
their listings. When I remember the Airbnbs during YC, I picture  
them with rolly bags, because when they showed up for tuesday dinners  
they'd always just flown back from somewhere.FragileAirbnb now seems like an unstoppable juggernaut, but early on it  
was so fragile that about 30 days of going out and engaging in  
person with users made the difference between success and failure.That initial fragility was not a unique feature of Airbnb. Almost  
all startups are fragile initially. And that's one of the biggest  
things inexperienced founders and investors (and reporters and  
know-it-alls on forums) get wrong about them. They unconsciously  
judge larval startups by the standards of established ones. They're  
like someone looking at a newborn baby and concluding "there's no  
way this tiny creature could ever accomplish anything."It's harmless if reporters and know-it-alls dismiss your startup.  
They always get things wrong. It's even ok if investors dismiss  
your startup; they'll change their minds when they see growth. The  
big danger is that you'll dismiss your startup yourself. I've seen  
it happen. I often have to encourage founders who don't see the  
full potential of what they're building. Even Bill Gates made that  
mistake. He returned to Harvard for the fall semester after starting  
Microsoft. He didn't stay long, but he wouldn't have returned at  
all if he'd realized Microsoft was going to be even a fraction of  
the size it turned out to be.   
[4]The question to ask about an early stage startup is not "is this  
company taking over the world?" but "how big could this company  
get if the founders did the right things?" And the right things  
often seem both laborious and inconsequential at the time. Microsoft  
can't have seemed very impressive when it was just a couple guys  
in Albuquerque writing Basic interpreters for a market of a few  
thousand hobbyists (as they were then called), but in retrospect  
that was the optimal path to dominating microcomputer software.  
And I know Brian Chesky and Joe Gebbia didn't feel like they were  
en route to the big time as they were taking "professional" photos  
of their first hosts' apartments. They were just trying to survive.  
But in retrospect that too was the optimal path to dominating a big  
market.How do you find users to recruit manually? If you build something  
to solve your own problems, then  
you only have to find your peers, which is usually straightforward.  
Otherwise you'll have to make a more deliberate effort to locate  
the most promising vein of users. The usual way to do that is to  
get some initial set of users by doing a comparatively untargeted  
launch, and then to observe which kind seem most enthusiastic, and  
seek out more like them. For example, Ben Silbermann noticed that  
a lot of the earliest Pinterest users were interested in design,  
so he went to a conference of design bloggers to recruit users, and  
that worked well.   
[5]DelightYou should take extraordinary measures not just to acquire users,  
but also to make them happy. For as long as they could (which  
turned out to be surprisingly long), Wufoo sent each new user a  
hand-written thank you note. Your first users should feel that  
signing up with you was one of the best choices they ever made.  
And you in turn should be racking your brains to think of new ways  
to delight them.Why do we have to teach startups this? Why is it counterintuitive  
for founders? Three reasons, I think.One is that a lot of startup founders are trained as engineers,  
and customer service is not part of the training of engineers.  
You're supposed to build things that are robust and elegant, not  
be slavishly attentive to individual users like some kind of  
salesperson. Ironically, part of the reason engineering is  
traditionally averse to handholding is that its traditions date  
from a time when engineers were less powerful — when they were  
only in charge of their narrow domain of building things, rather  
than running the whole show. You can be ornery when you're Scotty,  
but not when you're Kirk.Another reason founders don't focus enough on individual customers  
is that they worry it won't scale. But when founders of larval  
startups worry about this, I point out that in their current state  
they have nothing to lose. Maybe if they go out of their way to  
make existing users super happy, they'll one day have too many to  
do so much for. That would be a great problem to have. See if you  
can make it happen. And incidentally, when it does, you'll find  
that delighting customers scales better than you expected. Partly  
because you can usually find ways to make anything scale more than  
you would have predicted, and partly because delighting customers  
will by then have permeated your culture.I have never once seen a startup lured down a blind alley by trying  
too hard to make their initial users happy.But perhaps the biggest thing preventing founders from realizing  
how attentive they could be to their users is that they've never  
experienced such attention themselves. Their standards for customer  
service have been set by the companies they've been customers of,  
which are mostly big ones. Tim Cook doesn't send you a hand-written  
note after you buy a laptop. He can't. But you can. That's one  
advantage of being small: you can provide a level of service no big  
company can.   
[6]Once you realize that existing conventions are not the upper bound  
on user experience, it's interesting in a very pleasant way to think  
about how far you could go to delight your users.ExperienceI was trying to think of a phrase to convey how extreme your attention  
to users should be, and I realized Steve Jobs had already done it:  
insanely great. Steve wasn't just using "insanely" as a synonym  
for "very." He meant it more literally — that one should focus  
on quality of execution to a degree that in everyday life would be  
considered pathological.All the most successful startups we've funded have, and that probably  
doesn't surprise would-be founders. What novice founders don't get  
is what insanely great translates to in a larval startup. When  
Steve Jobs started using that phrase, Apple was already an established  
company. He meant the Mac (and its documentation and even  
packaging — such is the nature of obsession) should be insanely  
well designed and manufactured. That's not hard for engineers to  
grasp. It's just a more extreme version of designing a robust and  
elegant product.What founders have a hard time grasping (and Steve himself might  
have had a hard time grasping) is what insanely great morphs into  
as you roll the time slider back to the first couple months of a  
startup's life. It's not the product that should be insanely great,  
but the experience of being your user. The product is just one  
component of that. For a big company it's necessarily the dominant  
one. But you can and should give users an insanely great experience  
with an early, incomplete, buggy product, if you make up the  
difference with attentiveness.Can, perhaps, but should? Yes. Over-engaging with early users is  
not just a permissible technique for getting growth rolling. For  
most successful startups it's a necessary part of the feedback loop  
that makes the product good. Making a better mousetrap is not an  
atomic operation. Even if you start the way most successful startups  
have, by building something you yourself need, the first thing you  
build is never quite right. And except in domains with big penalties  
for making mistakes, it's often better not to aim for perfection  
initially. In software, especially, it usually works best to get  
something in front of users as soon as it has a quantum of utility,  
and then see what they do with it. Perfectionism is often an excuse  
for procrastination, and in any case your initial model of users  
is always inaccurate, even if you're one of them.   
[7]The feedback you get from engaging directly with your earliest users  
will be the best you ever get. When you're so big you have to  
resort to focus groups, you'll wish you could go over to your users'  
homes and offices and watch them use your stuff like you did when  
there were only a handful of them.FireSometimes the right unscalable trick is to focus on a deliberately  
narrow market. It's like keeping a fire contained at first to get  
it really hot before adding more logs.That's what Facebook did. At first it was just for Harvard students.  
In that form it only had a potential market of a few thousand people,  
but because they felt it was really for them, a critical mass of  
them signed up. After Facebook stopped being for Harvard students,  
it remained for students at specific colleges for quite a while.  
When I interviewed Mark Zuckerberg at Startup School, he said that  
while it was a lot of work creating course lists for each school,  
doing that made students feel the site was their natural home.Any startup that could be described as a marketplace usually has  
to start in a subset of the market, but this can work for other  
startups as well. It's always worth asking if there's a subset of  
the market in which you can get a critical mass of users quickly.  
[8]Most startups that use the contained fire strategy do it unconsciously.  
They build something for themselves and their friends, who happen  
to be the early adopters, and only realize later that they could  
offer it to a broader market. The strategy works just as well if  
you do it unconsciously. The biggest danger of not being consciously  
aware of this pattern is for those who naively discard part of it.  
E.g. if you don't build something for yourself and your friends,  
or even if you do, but you come from the corporate world and your  
friends are not early adopters, you'll no longer have a perfect  
initial market handed to you on a platter.Among companies, the best early adopters are usually other startups.  
They're more open to new things both by nature and because, having  
just been started, they haven't made all their choices yet. Plus  
when they succeed they grow fast, and you with them. It was one  
of many unforeseen advantages of the YC model (and specifically of  
making YC big) that B2B startups now have an instant market of  
hundreds of other startups ready at hand.MerakiFor hardware startups there's a variant of  
doing things that don't scale that we call "pulling a Meraki."  
Although we didn't fund Meraki, the founders were Robert Morris's  
grad students, so we know their history. They got started by doing  
something that really doesn't scale: assembling their routers  
themselves.Hardware startups face an obstacle that software startups don't.  
The minimum order for a factory production run is usually several  
hundred thousand dollars. Which can put you in a catch-22: without  
a product you can't generate the growth you need to raise the money  
to manufacture your product. Back when hardware startups had to  
rely on investors for money, you had to be pretty convincing to  
overcome this. The arrival of crowdfunding (or more precisely,  
preorders) has helped a lot. But even so I'd advise startups to  
pull a Meraki initially if they can. That's what Pebble did. The  
Pebbles   
assembled  
 the first several hundred watches themselves. If  
they hadn't gone through that phase, they probably wouldn't have  
sold $10 million worth of watches when they did go on Kickstarter.Like paying excessive attention to early customers, fabricating  
things yourself turns out to be valuable for hardware startups.  
You can tweak the design faster when you're the factory, and you  
learn things you'd never have known otherwise. Eric Migicovsky of  
Pebble said one of the things he learned was "how valuable it was to  
source good screws." Who knew?ConsultSometimes we advise founders of B2B startups to take over-engagement  
to an extreme, and to pick a single user and act as if they were  
consultants building something just for that one user. The initial  
user serves as the form for your mold; keep tweaking till you fit  
their needs perfectly, and you'll usually find you've made something  
other users want too. Even if there aren't many of them, there are  
probably adjacent territories that have more. As long as you can  
find just one user who really needs something and can act on that  
need, you've got a toehold in making something people want, and  
that's as much as any startup needs initially.   
[9]Consulting is the canonical example of work that doesn't scale.  
But (like other ways of bestowing one's favors liberally) it's safe  
to do it so long as you're not being paid to. That's where companies  
cross the line. So long as you're a product company that's merely  
being extra attentive to a customer, they're very grateful even if  
you don't solve all their problems. But when they start paying you  
specifically for that attentiveness — when they start paying  
you by the hour — they expect you to do everything.Another consulting-like technique for recruiting initially lukewarm  
users is to use your software yourselves on their behalf. We  
did that at Viaweb. When we approached merchants asking if they  
wanted to use our software to make online stores, some said no, but  
they'd let us make one for them. Since we would do anything to get  
users, we did. We felt pretty lame at the time. Instead of  
organizing big strategic e-commerce partnerships, we were trying  
to sell luggage and pens and men's shirts. But in retrospect it  
was exactly the right thing to do, because it taught us how it would  
feel to merchants to use our software. Sometimes the feedback loop  
was near instantaneous: in the middle of building some merchant's  
site I'd find I needed a feature we didn't have, so I'd spend a  
couple hours implementing it and then resume building the site.ManualThere's a more extreme variant where you don't just use your software,  
but are your software. When you only have a small number of users,  
you can sometimes get away with doing by hand things that you plan  
to automate later. This lets you launch faster, and when you do  
finally automate yourself out of the loop, you'll know exactly what  
to build because you'll have muscle memory from doing it yourself.When manual components look to the user like software, this technique  
starts to have aspects of a practical joke. For example, the way  
Stripe delivered "instant" merchant accounts to its first users was  
that the founders manually signed them up for traditional merchant  
accounts behind the scenes.Some startups could be entirely manual at first. If you can find  
someone with a problem that needs solving and you can solve it  
manually, go ahead and do that for as long as you can, and then  
gradually automate the bottlenecks. It would be a little frightening  
to be solving users' problems in a way that wasn't yet automatic,  
but less frightening than the far more common case of having something  
automatic that doesn't yet solve anyone's problems.BigI should mention one sort of initial tactic that usually doesn't  
work: the Big Launch. I occasionally meet founders who seem to  
believe startups are projectiles rather than powered aircraft, and  
that they'll make it big if and only if they're launched with  
sufficient initial velocity. They want to launch simultaneously  
in 8 different publications, with embargoes. And on a tuesday, of  
course, since they read somewhere that's the optimum day to launch  
something.It's easy to see how little launches matter. Think of some successful  
startups. How many of their launches do you remember?  
All you need from a launch is some initial core of users. How well  
you're doing a few months later will depend more on how happy you  
made those users than how many there were of them.  
[10]So why do founders think launches matter? A combination of solipsism  
and laziness. They think what they're building is so great that  
everyone who hears about it will immediately sign up. Plus it would  
be so much less work if you could get users merely by broadcasting  
your existence, rather than recruiting them one at a time. But  
even if what you're building really is great, getting users will  
always be a gradual process — partly because great things  
are usually also novel, but mainly because users have other things  
to think about.Partnerships too usually don't work. They don't work for startups  
in general, but they especially don't work as a way to get growth  
started. It's a common mistake among inexperienced founders to  
believe that a partnership with a big company will be their big  
break. Six months later they're all saying the same thing: that  
was way more work than we expected, and we ended up getting practically  
nothing out of it.   
[11]It's not enough just to do something extraordinary initially. You  
have to make an extraordinary effort initially. Any strategy  
that omits the effort — whether it's expecting a big launch to  
get you users, or a big partner — is ipso facto suspect.VectorThe need to do something unscalably laborious to get started is so  
nearly universal that it might be a good idea to stop thinking of  
startup ideas as scalars. Instead we should try thinking of them  
as pairs of what you're going to build, plus the unscalable thing(s)  
you're going to do initially to get the company going.It could be interesting to start viewing startup ideas this way,  
because now that there are two components you can try to be imaginative  
about the second as well as the first. But in most cases the second  
component will be what it usually is — recruit users manually  
and give them an overwhelmingly good experience — and the main  
benefit of treating startups as vectors will be to remind founders  
they need to work hard in two dimensions.  
[12]In the best case, both components of the vector contribute to your  
company's DNA: the unscalable things you have to do to get started  
are not merely a necessary evil, but change the company permanently  
for the better. If you have to be aggressive about user acquisition  
when you're small, you'll probably still be aggressive when you're  
big. If you have to manufacture your own hardware, or use your  
software on users's behalf, you'll learn things you couldn't have  
learned otherwise. And most importantly, if you have to work hard  
to delight users when you only have a handful of them, you'll keep  
doing it when you have a lot.Notes[1]  
Actually Emerson never mentioned mousetraps specifically. He  
wrote "If a man has good corn or wood, or boards, or pigs, to sell,  
or can make better chairs or knives, crucibles or church organs,  
than anybody else, you will find a broad hard-beaten road to his  
house, though it be in the woods."[2]  
Thanks to Sam Altman for suggesting I make this explicit.  
And no, you can't avoid doing sales by hiring someone to do it for  
you. You have to do sales yourself initially. Later you can hire  
a real salesperson to replace you.[3]  
The reason this works is that as you get bigger, your size  
helps you grow. Patrick Collison wrote "At some point, there was  
a very noticeable change in how Stripe felt. It tipped from being  
this boulder we had to push to being a train car that in fact had  
its own momentum."[4]  
One of the more subtle ways in which YC can help founders  
is by calibrating their ambitions, because we know exactly how a  
lot of successful startups looked when they were just getting  
started.[5]  
If you're building something for which you can't easily get  
a small set of users to observe — e.g. enterprise software — and  
in a domain where you have no connections, you'll have to rely on  
cold calls and introductions. But should you even be working on  
such an idea?[6]  
Garry Tan pointed out an interesting trap founders fall into  
in the beginning. They want so much to seem big that they imitate  
even the flaws of big companies, like indifference to individual  
users. This seems to them more "professional." Actually it's  
better to embrace the fact that you're small and use whatever  
advantages that brings.[7]  
Your user model almost couldn't be perfectly accurate, because  
users' needs often change in response to what you build for them.  
Build them a microcomputer, and suddenly they need to run spreadsheets  
on it, because the arrival of your new microcomputer causes someone  
to invent the spreadsheet.[8]  
If you have to choose between the subset that will sign up  
quickest and those that will pay the most, it's usually best to  
pick the former, because those are probably the early adopters.  
They'll have a better influence on your product, and they won't  
make you expend as much effort on sales. And though they have less  
money, you don't need that much to maintain your target growth rate  
early on.[9]  
Yes, I can imagine cases where you could end up making  
something that was really only useful for one user. But those are  
usually obvious, even to inexperienced founders. So if it's not  
obvious you'd be making something for a market of one, don't worry  
about that danger.[10]  
There may even be an inverse correlation between launch  
magnitude and success. The only launches I remember are famous  
flops like the Segway and Google Wave. Wave is a particularly  
alarming example, because I think it was actually a great idea that  
was killed partly by its overdone launch.[11]  
Google grew big on the back of Yahoo, but that wasn't a  
partnership. Yahoo was their customer.[12]  
It will also remind founders that an idea where the second  
component is empty — an idea where there is nothing you can do  
to get going, e.g. because you have no way to find users to recruit  
manually — is probably a bad idea, at least for those founders.Thanks to Sam Altman, Paul Buchheit, Patrick Collison, Kevin  
Hale, Steven Levy, Jessica Livingston, Geoff Ralston, and Garry Tan for reading  
drafts of this.