

How to plan?

Oct 8, 2022

How to plan? How hard could it be? 4k words scribbled down on a sunny October afternoon for people in tech observing the Season's Traditional Annual Planning Process, inspired by a recent interview question (and 25 years of variously painful planning processes).



Kyle 🏳️‍🌈
@kyleve



you on your first time through annual planning caring a lot // me on my 7th go realizing it doesnt matter



I was in a job interview the other day when I got asked about planning, and so I decided I'd turn the question into another "plucked from the interviews" blog post. (see also [Interview Question: Why do leaders keep disempowering their teams?](#)) To be clear this is a significantly expanded answer than the one I rattled off from the top of my head on Zoom, but I think the themes are roughly the same.

And it's worth noting that because there isn't one true way to plan this post does not try to tackle that impossible topic. Instead these are some things to think about when addressing why planning is often so awful and so often useless.

(Also I'll note that this blog post, like all my blog posts, is very much in draft, Early Access, etc, and feedback and even [pull requests](#) are always welcome)

Interview Question:

Have you been involved in annual planning? How did you handle it? Do you prefer top down or bottom up? Is thinking about trade-offs something you've done? Do you have a framework for that?

This interview was for one of those jobs leading product and engineering which are coming back into vogue, sometimes called a GM, or CPTO, or just a VP of something or other. As an industry we've gotten pretty clear on the downsides of matrix management and the pendulum has started to swing the other way. Though to paraphrase [Andy Grove](#), "*When deciding between organizing functionally and organizing along business units the right answer is always the other one.*" (post for another day)

Let's also set aside the question of trade-offs for the moment, except to say, trade-offs are the very heart of engineering and engineering leadership. Software as a medium is so malleable that functionally anything can be made to work. Any language, architecture, approach, style, process, etc can be made to work given sufficient effort. Meaning we find ourselves not asked to choose between right and wrong answers, but between trade-offs.

Planning is a bit like software in that with enough effort anything can be made to work. (*Anything! I've done planning using left over Halloween candy spread over the floor of the CEO's office, and planning where everything was mapped to an Ancient Greek Olympic sport*) Consequently we aren't looking for the Right Way to Plan, as much as trying to solve for a set of constraints and things we've learned (the hard way) about how humans work. (Also like software: planning is way easier the smaller your team is)

The first challenge we need to navigate about planning is we're usually using a single process (or small set of processes) to serve a bunch of different goals.

Some of the Goals of Planning

- Communicate a shared goal and vision for everyone to pull towards. Our vast and endless sea to yearn for.
- List the projects that teams are going to work on, and the metrics they plan to move.
- Equip the business to estimate whether the projects and impacts are sufficient to meet the organization's targets (growth, revenue, capabilities, etc)
- Solicit insights from the organization on what are the critical and high impact projects to undertake.
- Give leadership a venue to provide feedback, set direction and course correct.
- Choose what we aren't going to do, and what projects we are going to wind down.
- Project plan, tile work, and evaluate our utilization.
- Identify and manage dependencies.
- Headcount planning and allocation.
- And a dozen other goals in any given instance.

As you can see we're a long way from [Do One Thing Well](#).

So here are a few rules of thumb for making planning suck less.

0. [Do fewer things.](#)
1. [Bottom up processes don't work.](#)
2. [Planning is the wrong time to introduce anything new.](#)
3. [You must provide frameworks and constraints.](#)
4. [Project planning has an inflection point.](#)
5. [Don't wait to kill bad ideas.](#)
6. [Minimize dependencies.](#)
7. [Headcount planning won't map to your plans.](#)
8. [What if money is no object?](#)

Some rules of thumb for making planning suck less.

0. Do fewer things.

A reoccurring theme in the suggestions below is that planning is so hard, because we've taken a bunch of really complex and stressful tasks that require a high degree of coordination across our company and tried to do them all at once. Nobody has the time to approach the work thoughtfully, nobody has the time to meet with you to give you feedback, and by the end of it no one has the spoons left to finish strong. Find ways to decouple the goals of planning, and reduce the high stakes nature of it.

1. Bottom up processes don't work.

There are a ton of critical insights we need from the people on the frontlines doing the work, but bottom up planning isn't how to get them. Bottom up planning starts at the squad/team level, the leaves of the hierarchy, and aggregates and synthesizes the plans at each level as they move up the hierarchy. We see a few predictable failure modes with bottom up:

- Individual teams, almost definitionally, aren't in possession of the information and perspective to understand what is most important for the whole organization, and perforce will make locally optimal choices based on the information they do have.
- Individual teams during planning will answer the question of what it would take *them* to accomplish a given goal, without awareness of what other teams will accomplish.

The above two tendencies are a recipe for an organization that grows larger every year, but is largely maintaining the status quo. Ben Horowitz has written about this in [How to Ruin Your Company with One Bad Process](#). His take isn't most charitable, but even people with the best intentions will walk into these traps.

Finally

- Bottom up processes fall victim to anchoring. In a bottom up process you ask people to spend time and energy advocating for what they think the right thing to do is, and what resources they need to do it. They naturally emotionally invest in that outcome. This now becomes the floor for their expectations.

2. Planning is the wrong time to introduce anything new.

Most organizations have a formal period of time annually (or quarterly or every 13 weeks) when they "Do Planning". This period is usually called Planning. The implicit or explicit ask of Planning is often, *"Tell us about all your exciting new ideas. We need your creativity to achieve our goals. Swing for the fences! Throw big rocks!"* This is a mistake.

Look at that list of goals from earlier in the post that we are already trying to solve with Planning. Does this look like a good time to also get creative? Similarly leadership often feels like Planning is their opportunity to introduce the new direction they'd like to see the company move in. All this does is create a scramble, shallow thinking, and low quality plans.

In most companies and organizations doing something new and meaningful requires us to coordinate with people across the organization (certainly in companies that have outgrown the project planning phase we discuss below in ["Project planning has an inflection point."](#))

To pick the canonical example, if we build and launch a new product without planning how it's going to be sold and marketed, and what infrastructure will be needed to support it, and without doing the research to understand how many times previously our company tried to launch this exact product then we'll fail. Which is something we see tech companies do over and over. This kind of coordination is hard at the best of times. Planning is not the best of times. Planning is when everyone is too busy to have these conversations. (and too emotionally keyed up as they contemplate their budget for the coming year and what that means for their team.)

Instead, start working on new things when you aren't Planning. Document what you want to do/build/etc. Share the proposal with people. Get their explicit buy-in to support the new thing. Test your assumptions and estimates with people in your organization with expertise that are different from yours. Get feedback from leadership about whether your idea is aligned with the direction of the organization (or if they're willing to change directions). Tell people how much it's going to cost: in terms of people assigned to it, time to iterate on it, to break through the noise in the market and educate your target customers, etc. Solicit internal interest for people who would be interested to join the effort if it gets approved. This is a Funding Proposal, and it's a great process to run outside of Planning.

Funding Proposals can seem like a heavy process. It can take weeks or months. Weeks or months is what it takes to make a good plan to do something new and meaningful. It isn't something you can do within the constraints of a company's annual Planning process.

The other option that also works well for a large class of ideas: just try them. Don't ask for resources, don't ask for support from other teams, do the quick and dirty test to validate or disprove your idea, build a thing knowing that you'll probably fail and be ready to back out your changes and throw it away. That's a great alternative to doing a real plan. It's the middle zone, where we pretend to plan, but do it badly, that gets us into trouble.

Not signing up to do new things during a hectic time of the year when everyone is running around crazy doesn't just benefit cross functional and rarely exercised relationships. It is also a benefit for partners who usually do (or should) be working closely together, like engineering and product. Something like a funding proposal makes keeping these close relationships in sync simpler, and avoids the inevitable blows up that follow post planning.

Sometimes Planning is taking place against an inescapable backdrop of chaos. For example frequent executive changes, or other forms of executive chaos, can mean frequent radical changes in strategic direction and emphasis. We see teams react either by tearing up their plans and scrambling to create new ones, or by becoming rigid, waiting out the current regime. Having a regular practice that teams use to undertake important new work throughout the year can help to moderate consequences of either reaction, allowing

change with resilience.

3. You must provide frameworks and constraints

Providing frameworks and constraints is how you do the top down planning without micromanaging. [Ben's post](#) touches on the importance of telling teams what the budget they have to operate in is, and holding back a buffer (20%) to encourage teams to be creative and ambitious, and to preserve optionality. Those are useful constraints, but there are so many others that teams need. This approach is explicitly more work for leadership in return for better outcomes.

Metrics (and by extension OKRs) are perennially popular constraints. Revenue, churn, sign ups, availability, etc are the bread and butter of many companies' planning processes. Consider pairing them with metrics that shape a narrative (storytelling metrics) about what you think is important: customers who used our product both on desktop and mobile, % of the company who adopts a product once it's available, etc.

Frameworks are a useful way to introduce constraints.

- "Even overs" can be a useful framework. *Our goal is to be a multi-product company which means prefer net dollar retention even over new customers. Ship new features even over multi-platform consistency.*
- An investment portfolio approach is a framework. *20% of our effort should be going to brand new efforts that are high risk and high reward. The number of people it takes to operate our mature products should decrease 20% YoY even as usage grows.*
- Capabilities models are a framework. *We need to be able to X so we can unlock Y opportunity.*

For similar reasons to what we discussed in points 1 & 2, these constraints can't be uncovered by the planning process, they need to be constraints people know going into Planning.

As an example, a popular constraint is: we'd like to do a marketing event to celebrate our new features this year. It can be tempting to let the teams tell you when they'll be ready and then plan the event, but that's circular. They can only tell you what will be ready when they know when the event is. (and knowing when the event is informs if this is a quick and dirty or a thorough and thoughtful implementation we'll be building)

The most important constraint that frequently gets lost in planning is: what is the cost of the work we've already committed to?

- What is our KTLO?
- What is needed to finish the migrations we have underway?
- What did we already promise customers?
- What is it going to cost to deprecate the pieces of the system that aren't working?
- In general, what does it cost us to maintain the status quo?

These are things you should be tracking throughout the year, but if you aren't, setting aside some time before planning to get these answers is important.

As a leader top down planning comes with not only real cost, but real risk. There's a good chance that you're wrong about at least a few things. You need to be comfortable finding that out, and you need people who are willing to tell you that.

4. Project planning has an inflection point.

There is a time in the life of many companies in which they are single threaded, or close to it. Either everyone is working towards The One Important Thing, or everyone is working either on The One Important Thing or they're working on something unimportant. At this stage most of the key coordination problems you're trying to solve with Planning take care of themselves. Instead Planning will focus on the steps necessary to achieve The One Important Thing: milestones, blockers, risks, launch plans, have we broken the big thing down into smaller things that deliver value incrementally, etc. This is Project Planning and it's perfectly reasonable to do at this stage of a company's life.

As the complexity of what you're trying to accomplish increases and Planning becomes more about coordination and communication, Project Planning is no longer appropriate to do at the company level. It's too fine grained. More importantly it leads you to make overly detailed commitments about *how* you solve a problem to people who are outside your team. You should be committing to the impact of solving the problem. Once you have people outside your team monitoring your approach to solving a problem it becomes much harder to change your plan as you learn new things. (Marty Cagan talks about this frequently, e.g. [in this post on The Problem with Product Roadmaps](#)) Also if you aren't learning new things when you're building software, and especially products, you're doing something terribly wrong, but that is another post.

Many organizations miss this critical inflection point and go sailing off into Strategic Planning with the expectations of Project Planning still firmly in place. Companies operating in this mode are the ones whose frustration with not getting the strategic results they were hoping for leads them to push for more and more fine grained planning as the antidote: an attempt to achieve greatness by optimizing for predictability.

5. Don't wait to kill bad ideas.

Just about every planning process pairs the question, *"What are we going to start doing?"* with the question, *"What are we going to stop doing? Focus! Put more wood behind fewer arrows!"*. And yet somehow the plan for what we're going to stop doing never gets as much time and attention as the plan for what we're going to start doing. Which makes sense.

Stopping doing things is by default way less fun than starting things. Stopping means we tried something and it didn't work. Something that in recent memory everyone agreed was the most important thing we could be doing. Something that was so important that maybe we stopped doing something else in order to do this. (and as a manager, stopping means that when I told you this was the most important thing for you to work on, I was probably wrong, and I might look kind of dumb).

Also, we're in a hurry to start this new thing. It's exciting, it's important, we've already baked its success into next year's financial plan. We gotta get going. Cleaning up and deprecating the old thing, is going to have to wait.

Run this cycle a few times and you'll see your technical debt explode ([there is no such thing as technical debt](#), but sufficient for the moment), and your team's cynicism along with it.

Instead emphasize with each team that you believe that their plans are hypotheses, and that you see their job as testing those hypotheses. Setup regular conversations to ask how that testing process is going. What have they learned? What have they disproven? Where have they changed their mind? I would like to say the goal is to "Make Killing Bad Ideas Easy", but I'm afraid that isn't possible. You see 50% of all of our ideas are going to fail, and the other 49.9% will require multiple iterations before they succeed, and telling the difference between those two conditions is hard work. So make it *easier* to kill bad ideas, and make it easy to talk about killing bad ideas *sooner*.

6. Minimize dependencies

Dependencies are another area where Planning's attempt to serve many goals leads to problems. In Planning, as we've said, we are often being asked to think big, take risks, make big impactful bets. We're also being asked to contribute to a financial plan that needs to be reasonably accurate. These two conflicting desires meet at Dependencies..

I can make a reasonable case for launching a new multi-billion dollar business next year, and all I will need is for everyone in the company to stop what they're doing, and work on my plan (aka agree to my dependencies), plus the additional thousand or so people I've put

in my plan that we'll hire (*that reminds me, let's write down a dependency on recruiting*). I may even be able to convince (bully?) some of those teams into agreeing to work on one of my dependencies. (or maybe I don't even ask and just write it down in the plan?) This all sounds ridiculous and yet you see it over and over again in Planning.

The problem with there being a significant number of dependencies between teams, especially those that are critical to strategic plans, is you are tightly coupling those teams to the point where they are functionally the same team. Either they *should* be the same team, at least for the span of this project/plan/initiative/etc or you should minimize the dependencies on the critical path.

As the person designing Planning, encourage people to build their plans on the foundation of things that are already working and shipped. New capabilities, and building support and excitement for them should be done via Funding Proposals outside this process (see ["Planning is the wrong time to introduce anything new."](#))

7. Headcount planning won't map to your plans

It seems logical that Planning would inform Headcount Planning and often they are tightly coupled. After all, once we know what the highest impact work is next year, we should align what is probably our most expensive resource, people, against delivering that work. This approach inevitably raises the stakes. People who want to grow their teams are often portrayed as empire builders, but growth and hiring are part of any healthy team. We are now motivated to navigate/game Planning to keep our team healthy.

We can start shifting the high stakes nature of Planning by moving towards off cycle funding proposals as discussed. Team growth, and growth projects for the people on the team have then already been handled, and we can engage in Planning with a shared goal of producing the best holistic plan for the company.

Additionally as mentioned in point 3 (Constraints and Frameworks) you should be quantifying the cost of the work we're already committed to. This avoids the common pressure to generate new projects just to keep a team funded. Similarly reducing dependencies simplifies the sometimes 3-card monte of who is asking for headcount to fund the dependency and how that will be allocated (*and are those headcount of people currently working for the company, or the people we'll hire in 6 months?*)

Finally headcount is an area where you just need to be comfortable with ambiguity.

Planning processes that try to be precise will backfire creating thrash and anxiety. I was once at a company that was very concerned about in-year cash flow, and rolled out a headcount planning process that closely modeled in what weeks we could hire people

(because if someone starts in week 26 of the year, only half the salary cost lands in year). There was a spreadsheet produced that showed you what percentage of a headcount had vested for your team, and allowed you to borrow against future headcount. The formula for calculating the cost of borrowing was sufficiently complex that it required a Windows version of Excel, and so all senior engineering leadership had to get a 2nd laptop just to hire. It is understandable to try to exert control through precision over this expensive resource, which is why comfort with ambiguity shows up over and over again in definitions of what it means to be a senior leader.

After you've removed as much of the headcount budgeting pressure from your planning process, your second goal should be to roll out something simple, knowing it will always be fuzzy and inexact. Why will the process be fuzzy and inexact? People. Most companies struggle to have an accurate view of how people map to teams, how many people have been hired but not yet started, interns, etc. A simple process might be something like give each team an end of year "Not to Exceed" number (i.e. have this many people on your team or less by the end of the year). This gives in year flexibility to hire without stop-start interruptions as you try to hit some complex intra-year target with predicted attrition curves, etc. (stop-start hiring is problematic because a good hiring process requires a healthy pipeline of candidates, and a pipeline takes time to build, and dries up when you stop hiring).

But how do we solve for that need of healthy teams to grow separate from what they might have committed to during Planning and why do we care?

First we care, because new people joining a team creates potential for new ideas, new skills, new backgrounds, new opportunities for mentorship, fresh energy, and team continuity when someone (or a set of someones) leaves. To enable a team to see growth without pressuring them to game Planning, it's important to keep a significant percentage of your budget in reserve, say 20%, that teams can propose to use off cycle. Holding back a significant percentage can be painful if you're looking at a sea of bottom up proposals that already add up to 200% of your allocated budget. However if you've implemented some of the planning rules of thumb we're discussing you now have options. We might have as many as 3 different pathways to get funding for a team's growth as appropriate for different types of work: a funding proposal for net new work, a planning process for run of the mill growth, and an exceptions process to access the reserve for opportunistic and organic hiring that keeps a team healthy.

8. What if money is no object?

In tech there are companies for whom money is no object. The resources aren't constrained (for the moment). There is no need to evaluate the plans for their odds of

success. You talk to people at these companies and they say things like, *"I was at (hypergrowth) for 14 months and my team grew by 100% every quarter" (until the layoffs)*. Bottom up approaches are common in these companies.

My experience being at hypergrowth companies, and also being at companies that were trying to be successful post hypergrowth is that it is important to remember that all code, all products, all marketing material, every piece of work product comes with long term costs, even if it's just the cost of deleting it (something most people find surprisingly difficult). Especially in a hypergrowth environment or when money and headcount are no object, you need to be better at planning, because the only constraints are the quality of your thinking.

So why do we even plan?

If we go into Planning already knowing many of the new projects we'll be tackling, what we think the impact of that work will be, with a majority of our budgets already allocated, and with the goal of introducing nothing new, why do we plan at all?



Planning serves two key purposes in this model.

The first is it acts as a point in time to synchronize our multi-threaded company. All year we operate with different teams and functions loosely coupled. Having a moment in time when we all agree we want to collate as comprehensive a view of what we're doing as possible is an important investment to keep us from drifting too far apart. That's what Planning is, a time to all get on the same page before we all start running fast again.

The second is, it's a forcing function. We're all busy and sometimes things just don't get done without a deadline and a person asking for them.

Planning is important, and it doesn't have to suck as much as it does. To get there you need to reduce the complexity of your planning process, try to do fewer things, and take into account the humans who are the hardware you're running the process on top of.

Kellan Elliott-McCrea
kellan@gmail.com

 [kellan](#)
 [kellan](#)
[Personal blog](#)

Hi. I think a lot about leading engineering teams. These are [some notes on engineering leadership](#) from doing the work, teaching the work, and coaching the work. Previously: VP Eng at Dropbox, SVP at Blink Health, CTO at Etsy, Flickr

Architect, etc.