Sprinting to show value

How to survive launching an experimentation program

Rommil Santiago

Sprinting to show value

How to survive launching an experimentation program

Rommil Santiago

To Rose, Dahlia, and Kai. Always.

You should get another book first

If you are looking to launch an experimentation program, and this is the first book you've picked up, I apologize. This is not to say that I don't think this book will be valuable to you, but rather that this book is, in a sense, incomplete.

I say this because this is not a book that walks you through the entire process of standing up an experimentation program. It doesn't tell you how to design experiments, what tools to pick, or what statistical approach to take. There are plenty of books and websites out there that cover these topics quite thoroughly.

What this book is, however, is a companion guide to all that other material which covers all the things that are often overlooked when standing up an experimentation program - particularly the politics of it. Throughout this book, I uncover the dirty underbelly of getting experimentation programs off the ground that no one talks about because it's messy and complicated. This book is a summary of all the lessons I've learned from having done this over and over during my career for various companies in diverse verticals. Now, before you think that I'm the world's expert at this - stop.

While I am quite knowledgeable about this area, this is not to say that I've never made mistakes. I have messed up countless times. I've been dressed-down in front of senior staff. I've even been fired. I don't want that to happen to you. I've written this book to make sure of it.

Furthermore, as it should be quite obvious, this book is also not very long. I don't subscribe to the school of thought that a book has to be lengthy to be worth reading. In fact, I pride myself that this book is short and to the point. Nothing would make me happier than if you could polish off this book during the time it takes to travel from Montreal to Ottawa by plane and feel like you've learned a thing or two.

Let us begin.

Chapter 1: Introduction to the Permission Cycle

If you had a childhood like mine, you probably grew up asking for permission a lot (and if you didn’t, you must have been a handful). I used to ask for permission to go to my friend’s place, to go bike riding by myself, you know, stuff like that. If you were anything like me, you dreamt about the day you didn’t have to ask for permission as much and simply do what you felt was right - or at the very least - what you felt like doing. For most, this day usually came when you either moved out and got your first job. You no longer had to ask to borrow money. You no longer had to ask if you could go out late at night. It was freedom at last! But what does this have to do with launching an experimentation program?

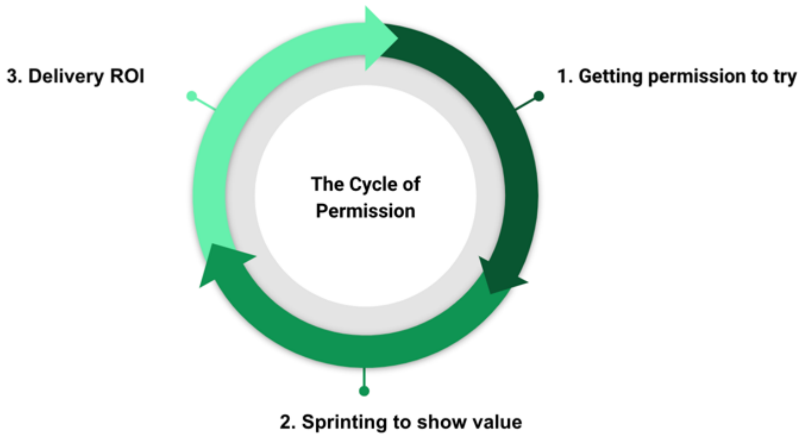
The mildly depressing reality is that launching an experimentation program is a lot like going back in time to when you had to constantly get permission. In fact, at most workplaces - and this is true for large companies as it is for small ones - unless you’re the CEO (and even that's debatable) you're in a constant cycle of seeking permission. It's never phrased like that, of course. But in a way it's very true.

Anytime you want to start a new project, work on something that's in your job description, or say, launch a new practice - you must seek permission from your boss, or your boss's boss, or your boss's boss's boss. Just like when you were younger, you have to do your best to make your case so that you get permission to proceed. Sometimes this permission is called, "getting the green light", or "go-ahead", or "approval", but at the end of the day, it's all the same thing. It's getting the permission to ***try***. And getting this permission to try is merely the beginning of what I call the **Permission Cycle**.

When launching a new program, particularly one like an experimentation program, folks have to complete this cycle on an

annual basis otherwise they risk having their program not getting any more resources, being defunded or potentially cut.

To make things even more complicated, unless your organization lives and breathes experimentation, and if it did you wouldn't be reading this book, there is a good chance that your program will be seen as a nice-to-have. Because of this, your nascent program will perpetually be on the list of things that could be cut - especially during its early stages. So it is in your best interest to keep this cycle going.



The Permission Cycle is simple. It starts with Getting the permission to try, followed by Sprinting to Show Value, followed by Delivering ROI, and finally Getting permission to grow - which is essentially Getting the permission to try again.

The time and the effort spent at each stage of this cycle will depend heavily on your organization's culture and budget. And while I understand that getting through this cycle, especially in your first year, can feel very exhausting, do your best not skip any stages. Doing so will just make your life harder later on.

Now let’s look at each stage in more detail.

Chapter 2: Getting permission to try

Everyone makes sense in their head

Getting permission to try should not be confused with making a business case. While getting permission to try comprises of many things – including making a business case – the two are not the same thing. Business cases are largely logical. In theory, they put all the facts on the table and demonstrate to decision-makers that a certain initiative is a good decision based on reason and the company's values and goals. But decisions are almost never made based on pure logic despite what we want to think.

When one is seeking permission to try, not only does one have to bring the appropriate facts, one has to go in with the right mindset. One has to go in with the goal of making decisions ***together*** with stakeholders - as opposed to convincing them. This changing of one's frame of mind, makes discussions less of a zero-sum situation, and more of a collaborative effort. This help you phrase your points in ways that are more palatable to decision-makers – increasing your chances of earning their permission.

However, even with the right frame of mind, you'll still encounter people who disagree with your point of view. There yet another mindset you must adopt.

A long time ago, at one of the many places I've worked, the CEO used to say that, "everyone makes sense in their own head." I've carried that along with me for years and have shared that little tidbit of knowledge with everyone I've mentored since. What that phrase means is that people make decisions and take actions that are in line with their own beliefs, values, and situations. It's common for us to judge others for making choices that seem illogical to us – however, we often do this without truly understanding what others' values are. Furthermore, one's values are never static. They are often very dependent on *context.*

Because of this, it is very common for two people, when presented with identical facts, to may make very different decisions. One only has to look to recent elections to see this play out in practice.

This is why if you've ever presented a business case to a group of decision makers, you will often notice that rarely will every person in the room nod their heads and agree with you from start to finish. It's more common to have that one person in the room who is skeptical of your big idea. This person will often ask you all sorts of tough questions - or at least ones that you weren't prepared for. In the back of your mind, you probably asked yourself, "Were you not listening to what I said? How are you not on board yet?" There were many possible reasons for this, but the most common is that you failed to establish a *common context* with everyone in your audience. When you establish this common context, you, in a sense, see things through their eyes and start to understand their perspective – and ultimately their objections. And this is gold.

To further drill this point home, let's look at an example. Imagine you are a street vendor that needs a big sale to help pay off your credit card bills. Furthermore, you are trying to convince a group of potential customers that are walking by to try your latest salad. In your mind, if you can get them to purchase your food, not only will you earn money to pay off your debts, they get delicious meals to satisfy their hunger. To you, this is a no-brainer because both sides win. In an effort to get them to purchase your wares, you tell them that your salad is fresh, tasty, healthy, and cheap! Logically, who could refuse a proposition like that? To your dismay, only one customer makes a purchase, while all the others pass. You ask yourself, how could they turn you down? You had the perfect pitch!

As you replay the scenario in your mind, you figure that one potential reason that they turned you down could be that they weren't hungry. In fact, they could have been thirsty. You say to yourself, had you known that, you may have changed your pitch. You could have included free bottles of water with your salads.

But the reality of the situation is that you couldn’t have made this offer because you lacked sufficient information about *their* situation. You lacked enough context. To take this a bit further, had this group of potential customers known that by buying your salads they could have helped you pay off your crushing debt, they may have taken pity on you and taken you up on your offer (OK, probably not.)

This simplistic example highlights the importance of establishing a shared context to getting permission to try. When both sides understand each other's situations, you increase the chances of reaching a decision that satisfies both parties.

But what kinds of information should both sides share?

Firstly, both sides need to understand the other's goals and pressures. While your goal is to launch an experimentation program, you need to understand the goals of everyone else. You need to know whether someone is accountable for revenue, or customer satisfaction, or brand awareness, etc. As you can imagine, if your new program doesn't align with their goals, or even worse, is counter to their goals – then you'll be in a pickle. You'll need to figure out a way to address conflict because it is paramount that you find a way to align what you can achieve with your program with *their* goals. Otherwise, your program will be seen as a distraction for the business.

Beyond understanding other's goals, it's important for you to understand what each person's most pressing concerns are. Again, by knowing this, you can highlight what *they can* accomplish if you were to receive permission to try to launch your program.

But how do you know when you don't have enough context? The easiest way to know that you don't have enough context is whether your stakeholders disagree with you. If they are arriving at a different conclusion than you despite seemingly having the same information as you is usually a clear sign you're missing some context. If you find yourself in this situation, to the best of your abilities, **stop everything** that you are doing and seek to get more context. Instead of going further down the road of confusion and frustration, muster up the courage to ask, ”Can you help me understand why you disagree?” - or something to that effect. This takes a lot of practice and bravery at first because it is common for people to shy away from appearing unknowledgeable or disagreeable. With that said, I still encourage you to fight this urge to shy away and ask for clarity. I promise you, that instead of appearing less than capable, you will appear as a true partner and will encourage others to work with you – and openness to working with you, is better than a, "No."

Handling common objections

Even with all the context in the world, it often helps to have some well-rehearsed responses to common objections to experimentation programs. If you have clear and concise responses, you will come off as some who is knowledgeable, which will encourage people to trust you. So while the following objections are far from exhaustive, they are the most common based on my experience.

Efficiency

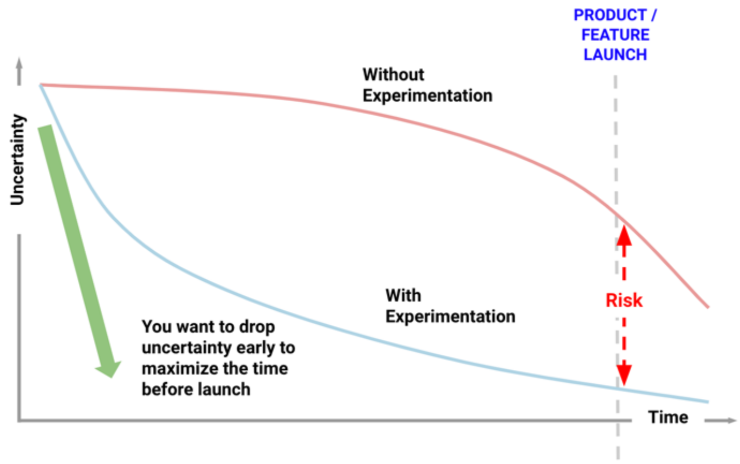
The first common objection, when it comes to experimentation, is that it is often seen as something that is counter to a company's speed and efficiency. This is particularly true at organizations that are delivery-focused rather than outcome focused. The way you can tell if your organization is delivery-focused is whether success is measured by the number of new features that were shipped, regardless of whether they performed well or not. The best way to handle this objection is turn the focus of the conversation towards **reducing uncertainty**.

Whenever you build, or make improvements to something, the goal is always to make it better or at the very least, not to make it worse. This applies to products, marketing campaigns, and workflows. However, whenever you're building something new, or releasing something to a new audience, there are **always** a lot of unknowns. Unknowns like: Will something work? Will people like it? Was our solution the most impactful solution we could have created? Did we invest in the right technology? Did we waste a lot of resources on something that won't do the job?

It must be made crystal clear that experimentation is the way to reduce that uncertainty. By running experiments on real customers, you will understand the relationship between variables, the impact of changes, whether your theories hold true - all with a measurable level of confidence.

To better understand this, refer to this figure.

Experimentation reduces uncertainty and risk



This figure illustrates the level of uncertainty when taking two alternative paths to launching a product or feature. The red path shows that without experimentation the level of uncertainty remains high and steady until it drops drastically around the time of launch. On the other hand, the blue line shows that with experimentation, you will reduce uncertainty very early on in the process. The lead-time that this reduced uncertainty generates allows teams to focus on addressing issues and maximize their resources. As you can see, at launch, there is a delta between the two paths. This delta represents the risks you take on when you don't experiment. This risk will often be realized in terms of lost revenue, market share, or performance. This risk is also materialized in wasted time in terms of reaching an **optimal solution**. It is on this wasted time that you should bring everyone's focus on when handling any objections around the reduction of efficiency. Remember, because experimentation eliminates uncertainty, the time to an optimal product or solution is not lengthened, but rather, it is *shortened* – which will translate to a market advantage.

Existing research

Another common objection you may hear is that the company is certain about something because it has asked countless customers for their opinions. While this research is indeed quite valuable, it isn't as reliable as insights generated from experimentation. This is because every other form of data is prone to **bias**.

For example, there could have been incentives attached to surveys which could have unconsciously influenced those who replied. Respondents may not have been selected randomly – but rather because they looked friendly. There are countless sources of bias that could potentially taint research efforts. Because experiments, namely randomized control trials, are the least likely to be impacted by bias, this is a reason that medicines are proven using experiments rather than focus groups that ask whether a drug made participants feel "better".

Lack of control

Yet another possible objection that decision-makers could raise is claiming that experimentation may hurt the business. Typically, some form of the question, “How do we know we aren’t testing something that will kill sales?” is asked.

From my experience, this kind of objection typically comes from someone who feels they do not have enough control of a situation and are typically accountable to something like sales. Of course, we can sympathize with these concerns. There is nothing more unsettling, at least in a business context, than being accountable for something that others could impact without one’s knowledge. In these situations, it is imperative that you communicate that your program will be transparent to all stakeholders - meaning that anyone can understand the progress and performance of any test at any time. You must also ensure that all major decisions will involve the right decision-makers. You must remember that until you show results, you haven't truly earned anyone's trust or confidence. You're still seeing permission to try. One step at a time.

You will also need to plan for the eventual big mistake. Even with the best monitoring and planning, mistakes *will* happen. You *will* launch a test that will either have some form of critical bug or underperform spectacularly. Anyone who says otherwise, hasn't been experimenting for very long. Without a socialized and agreed-upon contingency plan, any big mistake will undo all the progress you've made. Thus, it is in your best interest to set up thresholds and conditions with your stakeholders under which you will immediately end a test. Doing this sends the message that you care about the needs of your stakeholders and the business. Hopefully this approach will settle some of their nerves.

Chapter 3: Sprinting towards proving value

Once you’ve been given the permission to try, you will begin to feel pressure. It may not be immediately noticeable, but it is inevitable. The same people who gave you the permission to launch your experimentation program, the ones who probably vouched for it to someone important, the ones who probably got you funding, have their name and reputation on the line for you. These people will have understandable doubts in the back of their head about whether they made the right decision to let you try. Thus, it is your primary goal at this stage to eliminate this doubt as you to Sprint towards proving value.

Any win over perfection

The classic mistake that people make when standing up an experimentation program is aiming for perfection out of the gate. On paper, that doesn’t sound like a bad idea. Most people don’t like changing their routines or workflows that don't make sense. This is why people often aim to "get it right the first time". Unfortunately, the reality of the situation is that no one can predict the future – i.e., there is no "right" that works in all situations.

It is with great confidence that I tell you that no matter what you design, eventually it will *not* work. You'll find that either you will end up over-complicating things for the present, or you'll end up under-building things for the future. It could end up that the way you've organized your team isn't robust enough to support the growth of your program. Or perhaps, you will end up over-staffing. Or, you may end up not having enough funding to invest in tools to drive more efficiency. Whatever the case, you'll get it wrong somehow - and unfortunately, getting it wrong may reflect poorly on you.

You should probably come to peace with this now. Go ahead, I can wait. Take all the time you need.

Now that you've accepted this eventuality, what can you do? Beyond managing expectations (which when launching an experimentation program, often falls on deaf ears), you have one option: to sprint. You ust sprint towards showing the value of your program as *fast as you can*. Value in this case is often in some form of a win - be it a revenue lift, aversion of some loss, or an important learning. However, to achieve an experimental win or learning quickly, you must conclude a test quickly. To conclude a test early, you must launch a test quickly. To launch a test quickly, you must set up a test quickly. To set up a test quickly, you must pick a test, you guessed it, quickly.

Your best chance of getting through this gauntlet of obstacles is to pick a low effort and low-ish impact experiment which your *team* has a strong suspicion will win. One that you can launch and analyze quickly. Picking something low impact may feel counterintuitive, especially to product managers, but there is some logic behind this madness.

Of course, picking something that is low effort makes a lot of sense. However, picking something that is low impact has definite benefits that should not be overlooked. You must remember that early on in your program, most people will be skeptical about experimentation as they usually have had little exposure to running experiments. Because of this lack of experience, there is a strong chance that they will make a mistake along the way.

Thus, working on something that is **not critical** gives the team room to relax and maximize their learning from going through the process of launching an experiment. In the case that a team screws up, no worries. Simply end the test and try again. Teams will learn from their mistakes.

On the other hand, this opportunity to learn is less prominent a test is on something very critical to the business. The heightened level of reporting, and micromanagement will add too much baggage to the works.

Furthermore, at most companies, low impact tests require less approvals from stakeholders - which will shorten your time to production. This is not to say you should keep launching sleepers forever. In fact, I'd say after launching 2 tests a team usually has worked out most of the kinks from the workflows.

By getting a test out quickly, under less pressure than usual, on something that has a high confidence in producing win, you will be able realize one of two results:

1. Either you will achieve a win and your team will get to enjoy the sweet nectar of victory
2. Or you will not achieve a win and your team *will generate a learning from the process.*

Either way, you must share your learnings with all your stakeholders, especially those that gave you permission to try. You must do as publicly as possible. You can choose to leverage a company email, a presentation in front of the staff, messages over Slack, video summaries, etc. The sky's the limit. And remember, when you spread the news, be very sure to mention all those that participated in the test. Everyone. Consider CCing them on emails. @-ing them in Slack messages. Whatever you do, just make sure that they are recognized and that they get the credit they deserve.

When you do this, you will notice a few things will start to happen. Firstly, those who were mentioned will enjoy the limelight and will be motivated to continue experimenting. Secondly, you will notice that folks will start reaching out to you trying to get involved. Whatever you do, don't turn them away. Make the time and find a way for them to learn about and contribute to your program. As they launch experiments, give them credit too. This is how experimentation cultures are born.

Now, it's entirely possible that your early experiments don't deliver a win you can share broadly, or a learning that others will find useful (granted the inability to produce a useful learning is usually a symptom of poor experiment design). It's also entirely possible that you may run into all sorts of roadblocks and delays that prevent you from launching or analyzing an experiment quickly. What should you do in this case? Here, you must demonstrate ***progress***.

You must demonstrate progress

There is a saying that it is lonely at the top - and I believe there is some truth to that. The higher at an organization one goes, the harder it is to stay close to the work going on at a company. This is because the more layers between you and the work being done, the closer that communications become a bad game of telephone. Thus, as well intentioned as the folks who may sit between you and senior leadership are, you shouldn't rely on them as the sole means to report progress to senior leadership. When it comes to experimentation, it is very easy for things to be misinterpreted, or simply forgotten along the way. To get around this, what I encourage most people to do is to create a regular digest of work that your program has accomplished and share it widely. Include metrics (good and bad), experiments launched, implemented tech, roadblocks, upcoming plans and meetings, everything.

This accomplishes a few things:

- it acts as a record for you to look back at when trying to run an analysis
- it demonstrates that things are moving forward
- it sparks conversation as folks see all the good work you are doing
- and finally, it is a way for senior leadership to always know the state of your program without having to ask anyone.

Keep doing this regularly, and you'd be surprised how well it is received. You should expect a good deal goodwill as you sprint towards your first win.

Amplify your communications

Unfortunately, digests are only useful if they are noticed or read - so you can't simply put all your eggs into one communications basket. It is in your best interest to not only highlight the good work you are doing, but to encourage others to do so as well. But how can you get others to speak well of your program? Simple. Ensure that the experimentation program is framed as a way to address their needs while not making them change how they do things unnecessarily.

It is natural to instate new processes for new programs. After all, we want things to be done correctly. However, unless you have the clout and authority to make wholesale changes all at once, I encourage folks to play the long game. Yes. Eventually, you will want experimenters to follow a thorough process - but I would suggest doing it in phases, only adding burden to early experimenters if it demonstrates value to them immediately. In fact, as much as possible, you should make the lives of experimenters easier if at all possible. For example, while you may ultimately want experimenters to document experiments, that could be something *you* take care of in the early days. Another example is helping experimenters by handling communication or arranging meetings associated with experiments. Regardless of what tasks you decide to take on, it's critical that whatever you volunteer to do, you deliver on. This accomplishes a few things:

- It helps people associate experimentation with value rather than burden. If they are allowed to see the value of the learnings generated by experiments without being distracted by any noise, early experimenters are more likely to associate experimentation with positive progress (rather than needless process).
- It also helps you build trust with experimenters. By delivering on your commitments, you prove that you are trustworthy - and by extension, so should experimentation. Furthermore, by doing them a favor, you incentivize them to do *you* a favor by experimenting.

- Finally, it demonstrates that you are a *partner* - where their success is your success. Nothing builds trust like knowing the other party also has skin in the game.

As a positive side-effect, by doing all this regularly, you will earn a good reputation. One that other experimenters are likely to share with other potential experimenters as well as their own leaders. Furthermore, do this long enough, and word will generally rise up to senior levels. Eventually, as experimenters' skills improve and they increase in comfort with experimentation, you can start shifting those duties over to them. Little by little, they'll start to adjust the way they work to accommodate experimentation until they are a champion of the program – making your life easier.

Get feedback

Another thing I strongly encourage you to do is to send out surveys and hold meetings (aka retroactives) with your stakeholders to uncover bottlenecks, as well as activities you should continue, start and stop doing. Ensure you invite those that gave you the permission to try to these meetings - they may not attend but they'll know that you're actively working to make the program better. Beyond obviously documenting the findings front these meetings, you must proactively act on the issues raised. Outside of helping the program advance, by acting on issues quickly, you demonstrate that you are committed and are doing all that you can to make the program a success.

With all that said, while I've had a lot of good success from creating digests and quickly acting on feedback, these activities will only buy you so much time. In fact, you should plan to stay in this phase for a very short period (though you should always continue to report progress and act on feedback for as long as the program is active). Eventually you will have move on and focus on delivering ROI.

Chapter 4: Delivering ROI

Experiments are about learning. And this is the message that you should be evangelizing day in and day out. However, eventually, senior leadership, the ones that gave you the permission to try, and to whom you've demonstrated progress to, will eventually demand to see a return on their investment. It is in your best interest to manage the expectations of your stakeholders in terms **how** will you measure the financial impact of experiments and **how** will you measure the investment into the program.

Every company is different, so there's no standard equation to do this. It will also depend on how much visibility you have to your company's books. With that said, here are some things you should consider and agree upon:

- How will you extrapolate proven lifts in revenue over a full year?

- How will you extrapolate proven savings or loss aversion over a full year?
- Will you include platform and tool costs?
- What are your related agency costs?
- Will you include experimentation team costs such as design and development? If so, how?
- How will you include the time when an Experiment is live?

As hinted above, because most tests will not result in a “win”, it’s important to highlight any losses your program has averted. I.e., was the launch or a feature halted because an experiment proved that it would hurt the company.

Saving your company money is just as valuable as generating it. You may want to consider reaching out to your finance department for support. No matter how you choose to calculate ROI, you should include this metric in your regular digest. Nothing makes a program look well managed than increasing revenue numbers.

Don’t sell your program short - think holistically

A common mistake made by those who launch experimentation programs is to only look at the impact of a test on only one part of the business. Every experiment can impact many functions including sales, marketing, customer support, engineering, etc. It is because of this that you must evaluate all your experiments using what is called an **Overall Evaluation Criteria** or OEC. An OEC is a fancy name for a consistent constellation of metrics that you examine for every experiment to understand how the experiment impacts the various angles of the business. Each angle from which you evaluate an experiment represents potential for demonstrating the value of your program in terms of, not just revenue, but also strategic insight and sparking important conversations. By doing this, your experiments will be of interest to more people. And instead of being an afterthought, experiments become ammunition to have debates.

Avoid the temptation of gaming

It is during the Delivering ROI phase, that you need to wean people, including yourself, off any vanity metrics such as the number of experiments launched, and onto those that represent true value for the business.

Of course, constantly running experiments is a good thing - however, it should **not** be your north star. Any metric that cannot be readily interpreted by senior leadership as one that is an indicator of business health, is likely to be a vanity metric. Common vanity metrics for experimentation programs are number of tests run and win percentage. The main problem with vanity metrics, beyond taking the focus away from **learning**, is that they can easily be gamed by launching low quality tests. As you can imagine, one could launch tests on small copy changes – which could game a metric like the number of experiments launched. Similarly, a win percentage metric could incentivize experimenters to never take risks and only test the obvious. Both of these would ultimately lead to a lower ROI for your program.

So which metrics should one measure? Beyond the obvious revenue related metrics such as lift generated, it's important to remember, not all returns are financial. Some of the most important returns for an experimentation program is driving a change of culture. Why would a change in culture be important? Simply put, a culture that embraces experimentation, is data-driven. It is one that measures the impact of its work, it is one that only releases features that have impact, it encourages real thought in terms of delivering value to the customer. One simply has to look at the S&P 500 to see that companies with strong cultures of experimentation like Amazon, Microsoft, and Netflix, outperform those that don't.

A company that embraces an experimentation culture is one where a lot of people are experimenting, and where tests are launched with little effort because experimentation is a **priority**. Thus, two of the metrics I suggest reporting on are the **number of experimenters** actively participating in your program, as well **as the time it takes to arrive at a decision** (aka time to decision) from an experiment. The former will show if the company is truly participating in the program (rather than just having a small handful of enthusiasts), while the later measures the efficiency of your experimentation processes where the assumption is that companies which take a long time to reach a decision from an experiment either is poor at experiment selection, design, development, prioritization, or analysis.

Regardless how you decide to measure your program's ROI, as it grows, so should the ROI, but eventually you ***will*** hit a plateau - or at least see one coming. Some of the signs to watch out for are:

- You aren't seeing much movement in your experiment backlog – i.e., experiments aren't getting launched very frequently. This usually is a sign that you must check whether there is a bottleneck somewhere in the flow such as at engineering, or design.
- You are starting to have a great deal of experiment collision, or in other words, you are running out of real estate to run tests.
- Your experimentation team is starting to lose track of deliverables and/or are missing timelines. You may be starting to see signs of burnout, as well.
- You find yourself saying, "We could run more tests if we just had…"

If you witness one or more of these signs, you may be heading towards a plateau indicating that your resources are or are close to being maxed out – in other words your program is delivering as much as it can. It is at this moment when you should start laying the groundwork to grow your program. But to grow your program, you must get something first. You guessed it. You must get permission.

Chapter 5: Getting permission to grow

Set yourself up for...failure

At this point, let us assume that you have had some success in getting folks at your company to start experimenting, learning about your customers, and realizing returns. Something to keep in mind is that while they may appreciate experimentation, they probably won't be able to envision the possibilities of expanding the program. People simply don't know what they don't know. That is **your job**. As the person in charge of the success of your experimentation program, don't be simply satisfied with merely launching it. Experimentation is not a one and done kind of deal – it is something that must grow with the company in terms of reach, complexity, and impact. As long as companies come up with new ideas, there's always room to increase the impact of experimentation.

Whatever the vision for your program, throughout your journey, it's key to always communicate what a full-blown program can do for the business as well as to highlight the roadblocks in the way from realizing it. Furthermore, this must be done with **foresight**.

I've experienced firsthand, several times, what it's like to launch a successful program only to be bottlenecked later. What often happens is that despite experimentation being bottlenecked, struggling to maintain its now-expected pace, your requests for more resources are stuck in limbo, often ignored, and deprioritized for other parts of the business. But how does this come to be? It's often because you didn't set yourself up for *failure*. A common thing that experimentation program managers do is to never show that anything is wrong – that their team can handle any volume of work flawlessly. I argue, that doing this for too long will be one the most painful mistakes one can make.

Let's explore this.

Every company is constrained for resources. Every. Company. Every company has goals, and the job of its employees is to achieve **all of them** rather than merely a subset of them. Thus, it is natural for leaders to put a lot of focus on the parts of the business that are in trouble or not where everyone thought it should be. Thus, if you have any desire to grow your experimentation program in this kind of environment, you must do these three things effectively:

1. Associate success of the company to the experimentation program
2. Associate the success of the experimentation program to its available resources
3. Associate the success of the company to the resources allocated to your program

The first item is self-evident - especially if you've read this far. The second item is often overlooked by program managers. The third item is where the magic lives.

You must do your job to connect these dots for your leaders as often as you can - particularly if you are starting to see bottlenecks in your future. Your communications and reporting should always seek to connect the fact that the returns generated by your program were delivered by a set of limited resources. You must proactively highlight what could have been achieved if you had more at your disposal. To help illustrate this concept, the following are potential statements one could make in increasing order in terms of its effectiveness in growing an experimentation program:

1. A recent experiment helped generate $1M in revenue.

2. A recent experiment helped generate $1M in revenue, bringing the total generated by the program this year to a grand total of $20M.
3. A recent experiment helped generate $1M in revenue, bringing the total generated by the program this year to a grand total of $20M. **We estimate that if we were to expand this program to other landing pages of the site, we could generate upwards of $30M in revenue - this would require another developer and a strategist to unlock.**

However, to be able to make statements like these with confidence, you must be able to back them up with data - which means you would have had the foresight to instrument these measurements ahead of time. For example, you need to know how much developer time you are currently using. Unless you were actively measuring the time a developer spent towards experimentation, you couldn't come up with any projections. And believe me, without projections, you won’t get approvals (unless you are very charming or are skilled at extorsion).

To tie all these concepts together, you’re trying to demonstrate that your company’s success is tied to the resourcing of your program and that the level of success that you can deliver is getting maxed out and unless you're able to secure more resourcing, you will fail in the near future. If you don’t do this, you’ll eventually find yourself in a situation where your resources are maxed, meaning that you’ll need to spend time collecting the right data with the hopes of getting in front of leadership to plead your case - which will probably result in your situation being sympathized with, but ultimately deprioritized because all the budget is tied up for the quarter or the year. Which means that you’ll have to support the extra load for the time being and hope that nothing else becomes a higher priority than your program along the way.

And once you are finally able to get your hands on more resources, you must have measurements in place to be able to readily demonstrate the return on those resources otherwise you might run into the same situation again down the road.

Chapter 6: The other gotchas

Even if you were to follow all my advice thus far, there are some important gotchas that you should be aware of that didn't quite fit into the other sections of this book.

Starting a new program mid-year is very hard

One of the most enduring patterns I've seen throughout my career is that virtually all programs that are started mid-year struggle during its first six months. Time and time again, the story is pretty consistent - it's tough to get buy-in, folks don't really want to change how they work, they feel experimentation is a waste of time. By the way, this doesn't just apply for experimentation programs. I've heard this in regard to Growth programs as well.

Over the years, what I've come to realize is that this lack of traction is not because of a particular approach, but rather **timing**. If your company is like every other company on the planet (hint: it is), halfway through any given year, there is a good chance that it is behind on some important initiative or goal. At this point, you'll probably see teams having to "double-down" to turn things around. Said in another way, people will be working on achieving goals that don't include your brand-new experimentation program because the goals of your experimentation program did not exist when the company set its goals at the beginning of the year.

What can you do in this case?

I would suggest a few things. I would suggest focusing hard on showing progress - *any* progress. I would also recommend you get ahead of the criticism and tell stakeholders that your program's lack of traction is expected and ask for assistance from above. Then, start preparing for the next fiscal year.

Get on next year's roadmap

Talk to anybody and everybody, including their superiors about getting experimentation onto roadmaps. It's unlikely that you'll run into someone that will think that experimentation is a bad idea, in fact you can expect virtually everyone to say that it makes sense to get experimentation included in next year's plans. But even if you get buy-in, insist that they put it in writing – preferably in a presentation.

Make experimental data mandatory

Another thing you should do is to start talking to leaders at your company and convince them to start asking for experiment results as a mandatory part of **any** business case presented to them. This point is obviously challenging to enforce, but if you are successful, this will have a trickle-down effect where folks will feel that they cannot make a case for any new initiative or show that their work was impactful without an experiment.

Focus on new hires

Sometimes, it's just too difficult to get people to change. In this case, you should explore adding **boilerplate copy to job descriptions** around that candidates must believe in the power of experiments. If you can't change those you work with, you can at least try to influence those who apply. To take this to another level, consider getting running an experiment added to onboarding programs. New employees are always eager to show their worth - so they represent a great opportunity to shape the future.

Training

The beauty and the most challenging thing about experimentation is that it sits at the intersection of many practices. Experimentation involves UX, analytics, statistics, project management, communications, and politics. It is unreasonable to expect others to pick up all those skills overnight. You'll need to be patient as you teach all these concepts to newcomers to the practice. With that said, there are a few topics you shouldn't put too much emphasis on.

Statistics are hard to teach

Avoid trying to get people to understand statistics in depth. Statistics is one of those things that either you get, or you pretend to get. Kidding aside, statistics is hard to grasp for those who haven't been exposed to them much. Having tried to teach many students statistics, I say with great confidence that you should focus on teaching just the high-level concepts and leave the nitty-gritty calculations to either a tool or an analyst. Not only will this ensure that the interpretation of results be more consistent, but this will remove a barrier for folks to experiment.

Workshops (mostly) suck

Another thing that I try to avoid is holding workshops. Workshops are wonderful for the visibility for your program, and they truly do teach people new skills in a friendly environment, but they suffer from a few things:

- If you are trying to train more than a handful of people, scheduling will be very challenging. In fact, as more people become interested in experimentation, you'll have to repeat the training over and over – taking you away from actually running the program.

- It is also hard to accommodate everyone's level of understanding - particularly of statistics. You'll either burn a lot of time leveling-up half the class, or you'll risk boring the other half.
- The squeaky wheel gets the grease. In workshops, a disproportionate amount of attention will be paid to the most outspoken student which risks the needs of the others in the room.

What I recommend is a blend of the following:

- Create a high-level document reviewing the basics of experimentation. Avoid detail. Many students often just want an overview of the process and a list of things they need to know.

- Create training videos. This allows students to refer back to the content easily. It also avoids the need for students to take detailed notes.
- Hold 1:1 meetings to address specific questions after students have read the high-level document and watched the videos. This way you can maximize your time and be more impactful.
- Have live examples and demo sites for people to reference. Seeing things in practice is always helpful to grasp new concepts.
- Share articles that illustrate concepts that you've taught.
- As mentioned earlier, send out surveys to identify gaps in your training and to receive feedback on where things can be improved. What I like to do is to let survey recipients score different aspects of the training to help me dive into problem areas.

In closing

I'd like to thank you for reading this far – in fact, I'm quite shocked that you did. Kidding aside, while not everything I've included in this book will be useful to everyone, I hope you've learned at least one thing that you feel is worth trying at your company.

Launching an experimentation program is challenging, and sometimes it can be hard. But whatever your journey is like, I hope you take the time to look back and do two things:

1. Document everything you've learned. You'll be surprised how much launching an experimentation program will impact other aspects of your life – professionally and personally.
2. Take pride in whatever progress you've made. It's easy to get down on yourself. Remember that not everyone will get as far as you have or will get as far as you will go.

Go get 'em.

Experiment Nation

You're still here? Well, if you'd like to learn even more about experimentation, I strongly suggest you check out ExperimentNation.com. There you will find interviews with experimenters from around the world, links to podcasts about experimentation, online classes, and other useful resources.

See you there.