Chapter 1: I Got Swag, Baby

Imagine playing a game like Super Mario™ with no extra lives—and your entire net worth on the line. This is what it's like to run a start-up. Founders face new challenges constantly, and they can't make a wrong move. In the early days of a company's life, even basic mistakes can be fatal.

There are a million ways to make bad decisions in the process of starting a new company. You can hire the wrong key person, overpay for services, build a solution that doesn't scale well, waste time with the wrong contractor, ignore security and have a leak of personal user data. The list goes on. Because things evolve so quickly in a start-up, there are a massive number of decisions to make. There are forks in the road *everywhere*.

Founders are especially likely to make errors in judgment in areas where they lack expertise, like technology. Non-technical entrepreneurs can make disastrous mistakes when they lack a high-level understanding of the technology on which their start-up relies.

Except, unfortunately, that's *all* of us.

It's nearly impossible to start a company today without integrating *a lot* of technology. Every entrepreneur is, at least partially, in "tech." Every company uses technology for maintaining websites, conducting meetings, doing outreach, storing documents, and much more. Start-ups need tech to build products and scale revenue, but there's no way the founder can become an expert in all these various technologies.

This means entrepreneurs are often forced to manage and oversee the implementation and use of technologies they don't fully understand. They become prone to making simple mistakes. Things can quickly go astray when someone who lacks the necessary expertise is in charge of technology, and it happens all the time in the start-up world.

Thankfully, plenty of evidence shows it's possible to lead a company without having deep technical expertise in every aspect of the business. Many highly successful app founders do not write a single line of code. The CEO of Intel has never built a microchip with his bare hands. The president of a hospital likely couldn't diagram the inner workings of an MRI machine. **The secrets to running a company as a non-technical founder are out there…because plenty of people are doing it.**

Unfortunately, Jeremy Parker and Josh Orbach hadn't discovered any of those secrets when they founded their promotional products company SWAG.COM back in 2016. They were about to discover a hard truth about how costly it can be to not understand your tech.

SWAG.COM Founders Start Strong, But Experience Big Setbacks

Things started out great for Jeremy and Josh. Despite having over 30,000 competitors, the two young men hustled their way through the corporate office buildings of New York City and made $60,000 in sales with nothing but some business cards and a lot of chutzpah. They officially had some revenue. The next step was to build a website. That's where they hit their first major setback.

Jeremy had his heart set on the domain name SWAG.COM, but it wasn't available. Someone else already owned SWAG.COM and they refused to sell it for less than $1.2 million. Of course, the aspiring founders didn't have anything close to that. So, it seemed impossible for them to ever get the domain name SWAG.COM. This is where most normal people would give up. Not these guys.

The entrepreneurs wouldn't settle for anything other than SWAG.COM as their name.

"I had this vision of people seeing our ad in January, and not thinking about it for months," Jeremy said. "And then they are doing a big corporate event in August, and someone suggests getting swag, so they immediately think, 'Hey, I know where to get that.' And we had to be SWAG.COM so people could remember us all that time later."

The domain name was very important.

"It wasn't going to work without that," Jeremy said, shaking his head. "We had to be SWAG.COM. There was no way around it."

With the seller asking for $1.2 million, and Jeremy and Josh sitting on just over $60,000 in assets, the two young founders entered negotiations to purchase SWAG.COM. Gradually, they brought the price down to an even $1 million. Then $800,000. After a few more hours of pleading, it had come down to half a million. By the end of a long and grueling day of talks, the price was just $200,000. It was much cheaper, but still considerably more money than Jeremy and Josh had in their bank account. It seemed the parties would never agree on a sale.

That's when Jeremy came up with a new way to set up the deal.

"I offered to pay back the $200,000 within two years," Jeremy remembers, "and also to give the guy a bit of equity in the company. Worst case scenario, if it didn't go anywhere, and we couldn't pay back the $200k, he'd get the domain back after two years and could do whatever he wanted with it."

The seller agreed to the unusual offer and the clock began ticking. The founders now had two years to get the company operational and earn at least $200,000.

Jeremy and Josh opened a WeWork office space on 42nd Street in New York City and hired a development firm to help them build a website. To keep things simple, they opted to build the initial site using a popular drag-and-drop shopping platform. This way, most of the online store could be made using pre-built options. They only had to build a plug-in that would add a few more advanced, custom features to the site.

They wanted the site to allow users to upload their logo or artwork and position it on a T-shirt, mug, calendar, or other item, then add text, preview a mock-up of the item, and place an order. It was a bit complicated because the site would have to consider factors like inventory, colors, materials, shipping costs, and more to calculate dynamic pricing for the customer in real time. The development team was confident they could handle it all and they started work.

Three months later, the website was 95 percent complete and Jeremy and Josh had spent about $25,000 on the project. They were expecting to receive the final version any day from the developers. That's when they learned some bad news.

"The developer called to explain that he couldn't finish the site," Jeremy says. "I was in shock. Apparently, the platform didn't support the dynamic pricing we needed. And he said his team had only just realized right at the end of the project that it's actually impossible."

So, Jeremy and Josh found themselves back at square one. They needed a website. Except, they had wasted three precious months and half of their money on code that was never going to work, and they weren't any closer to being able to pay back that $200,000.

Jeremy thought about how this catastrophe could have been avoided. *How come the dev team couldn't see this problem farther in advance? Was it a scam? Should I have hired a Chief Technical Officer?* Questions rattled in his head, and he knew he should've asked them back in January.

Just a small way into his entrepreneurial journey, Jeremy was already caught in one of the classic Founder's Dilemmas—he was hiring others to build something he didn't understand. And that's an easy way to make basic, avoidable mistakes.

You may be an expert in pottery. But, what about packaging? Or marketing? Or accounting? Maybe you're a gifted chemical engineer. But, how well do you understand safety regulations? And shipping logistics? No matter what kind of business you're in, and what amazing skills you have, at some point you'll have to hire someone to do something you don't understand. This is a key Founder's Dilemma.

In the case of Jeremy and Josh, they needed to build a website, but didn't understand much about software. This led them to make mistakes early on. Eventually, they realized they had to abandon the plug-in they'd expected to work and start over again with something brand new.

Jeremy and Josh aren't unique. The hurdles they faced in building a tech product, when neither of them knew how to write code, are the same issues that plague millions of entrepreneurs around the world. Their journey is emblematic of a greater struggle in the business world at large. Executives, who may not be technical, need to manage teams of people who are using complex stacks of tech products, manufacturing tools, and chemical processes. A Human Resources specialist may not understand how lasers work, but they need to hire an expert for a new project.

Thankfully, with so many people around the world struggling with these same limitations, some solutions have emerged. For example, in the start-up world, founders are notorious for mentoring each other and sharing ideas that work. In the collaborative culture of Silicon Valley, we love to fix difficult problems (and brag about how brilliant our solutions are). That helps maintain, and accelerate, the pace of innovation.

This book will help you overcome the Founder's Dilemmas in the start-up world. You'll learn how to interpret the rapidly evolving technology landscape so you can avoid the kind of mistakes Jeremy and Josh made. You'll see that, for people building a tech product, a common pitfall is mis-categorizing their objective. They don't understand what *type of product* they are even trying to build. In this book, you'll learn a simple framework for getting this right from the very beginning.

About Me

You might be wondering how I developed this framework and, for that matter, who I am. I'm Stanislav Synko and I've dealt with the Founder's Dilemmas repeatedly during my career as a software engineer, architect, and tech CEO. I've founded two service companies to help start-up founders navigate their technology needs. In 2017, we made our first investment (you guessed it, in SWAG.COM).

Many entrepreneurs are highly skilled in areas like business, sales, and marketing, but not in technology. This is why it's so common for tech companies to have two founders. There's one more business-y person and one technical person. Gates had Allen. Jobs had Wozniak. Zuckerberg had Moskovitz. But, what about founders who are doing it alone? Plenty of people start successful companies by themselves. How can they get reliable technical advice?

This mission became the DNA of my venture studio, Aleph One. We work to help founders understand the different technological options for building their product. We guide them through big decisions of tech strategy, team design, and phasing. Essentially, we function like a CTO and a tech team for companies who don't have a technical co-founder. The cherry on top is our investment piece. I realized there was a need for a company in the service tech industry that has its incentives aligned with entrepreneurs, participating in funding rounds and not just providing high-quality tech service.

When I first met Jeremy and Josh, they were frantic to fix their broken website and I was desperate to find work for my small development company. We worked out a deal to completely rebuild the platform using a custom backend. Within a few months, SWAG.COM was online and orders were rolling in.

Over the following years, I worked closely with the SWAG.COM team to add new features to their site. To solidify the partnership, Aleph One even made an investment in the start-up when they decided to raise capital. As the company grew, they never needed to hire their own in-house development team—Aleph One handled that. Today, more than 35 Aleph One engineers support their tech needs, making SWAG.COM one of the industry leaders.

Things have come a long way for SWAG.COM. Last year, the company earned over $30 million in sales. They were recently acquired by Custom Ink, a major player in the industry. They have long since outgrown their tiny WeWork office space on 42nd Street and now have distribution centers across the country. As for the $200,000 due to the original owner of the SWAG.COM domain? That's been paid in full.

Since the success of SWAG.COM, many founders have approached me about similar partnerships. They want Aleph One to invest in their start-up and to help develop their technology, too. To make things interesting, by investing in our clients, we put some skin in the game so they know we are always going to act in their best interests, rather than trying to find ways to upcharge them for more development work. We want them to be profitable, because when our clients get acquired, we share the upside with founders and are rewarded for the long journey we've taken together.

During my journey with Aleph One, I've constructed a solution to a key Founder's Dilemma—the dilemma of having expertise in areas like business, sales, and marketing, but not in technology.

It’s a practical solution, but it will require some work on your part. It requires analyzing your project in depth to understand your tech needs. Sometimes you might actually be able to build the product yourself if the right tools exist. Other times, you might need to hire a full-time technical genius to oversee development. Often, you can get away with an external tech team, if you can keep priorities aligned.

So, how do you know which type of product you’re building? How can you assess your own tech needs? How can you determine when you need to outsource and when you really need to hire a new in-house employee, or learn for yourself? What’s the first step to take when you find yourself stuck in this key Founder’s Dilemma?

Let’s dive in…

The Three Types of Tech Products

I’ve spent the better part of my career advising start-ups on their technology implementation. In this time, I’ve found it extremely useful to divide all tech products into three main categories, based on the nature of their technology needs: Simple, Custom, and Wild West.

I’ve noticed these are the three distinct tracks a product can take to maturity, depending on how the technology is used to solve its customers’ problems. Heading down the wrong path is a costly mistake that many small start-ups never recover from. When founders don’t understand which type of product they are building, they can easily pursue strategies that directly conflict with the product’s true technology needs.

I have conversations with founders every day, often about securing funding and/or technical resources, and I constantly see the same three types of products in start-ups:

> *Simple*
>
> A Simple product is something an average person could potentially make on their own using existing and out-of-the-box tools.
>
> *Custom*
>
> For a Custom product, the company has built a custom software platform, which gives them an edge over their competitors in an established industry.
>
> *Wild West*
>
> There is an extreme level of innovation for these products, to the point where the company is creating a new business model or inventing a new type of technology—hence, I refer to this level of product as Wild.

Just this morning, I spoke with two different entrepreneurs who have two very different levels of technology requirements. The first company built a mental health wellness app for pediatricians and their patients. This is a Custom product because they are innovating within an existing industry, consumer healthcare services, and have built a custom software platform. The second company helps scientists extract meaning from bioinformatics datasets, like genomic data. Their product is deeply innovative and requires technical genius to pull off. They are absolutely a Wild product. This particular founder is non-technical, but he has partnered with someone who has expertise in AI, data science, and natural language models to help build the product. For Wild products, it's critical to partner with an expert in the technology.

While these three types of products may seem obvious, it's amazing how many founders have misconceptions about what type of product they are building, and what technology they need.

A perfect example of this involves me and my good friend Pavel. He was starting up a media company, based in New York City, with content distributed through their website. He hired my team to build a custom editorial platform. We spent 18 months working on a highly advanced editorial functionality for his articles. He had a creative vision to re-imagine the process of content creation. There were different types of editors, and different steps in the process, and a whole lot going on. He wanted software that tied everything together.

First, we started with a basic WordPress site, but Pavel soon flew past the limits of complexity with that platform. So, we built a custom platform that allowed us to implement features like dynamic ad placement, advanced analytic tools, and a custom flow of editing articles sequentially, by many parties. Admittedly, the platform is very creative. It can produce fun content. Still, at the end of the day, it allows people to write and publish articles.

There are many other existing software platforms that support collaboration between groups of writers, an editorial process, and publication to a blog. The technology doesn't really make any extra money. It was not a software company. The fancy process isn't being licensed to other media companies. It's purely for internal use.

Things became more complicated than they had to be. It began as a Custom product, but we really could have kept it Simple. My biggest regret from that time was that I didn't have the experience to push back and disagree with Pavel. If I could do it again, we would approach it differently from the start. I would tell him that I don't think this fancy technology will help the business. I shouldn't have said yes to all of his requests so easily. As a media company, what we really needed was marketing and readership. We needed to get more eyes on his content and get people sharing it across other platforms. The system we spent 18 months working on was way more complicated than it had to be.

When entrepreneurs misjudge the level of technology they will need to implement to bring their product to the market, the results can be swift and cold. **Founders who thoroughly understand their tech requirements can make the right moves at the start and blow past the competition.**

Let's take a closer look at each type of product.

Simple Products

The most technologically basic type of product is one that a typical person would be able to make for themselves without much trouble. There is no rocket science here. This is a situation where you can use pre-built solutions that don't require any coding knowledge. At this level, you don't have any custom features that need to be developed on top of the platform.

The biggest benefit of pursuing a Simple product is that it doesn't cost much to build, so an entrepreneur can theoretically make it without needing to raise capital. The costs to get a Simple product up and running could easily be put on a regular credit card. Mainly, it's just the founder's time that goes into making the initial version.

As technology evolves, more products are becoming Simple. It's actually quite "simple" to mistake a product as Simple because you're familiar with another product that does something similar to what you're planning. "Hey, if those guys are doing that, surely we can do this, right?" Often, it seems like it should be easy to build on top of existing products and make a few tweaks. But, software development doesn't always work like that.

The story of SWAG.COM is a perfect example of company founders who thought they could build a Simple product, but actually needed to develop a custom platform and thus needed to build a Custom product. They wasted three months and $25,000 hoping to salvage their Simple solution, made on a popular e-commerce platform. Instead, they would have been much better off working on a custom platform right from the start. Their product required advanced business logic, warehousing, dynamic pricing and an innovation in the supply chain to deliver branded products to their customers. This easily places their product in the Custom category.

Often, an idea seems like it will be simple to build at first, but then after getting started it becomes apparent that it's going to be a huge challenge. On the other hand, it can be wise to begin with a more simple version of the product, rather than jumping right into full-scale development of a Custom or Wild product. Sometimes this "simple version" could just be a landing page to gauge interest and collect email addresses. Anything that can be built by an average person without much trouble is potentially a Simple product.

Custom Products

When a company gets to the point where it is building its own custom software platform, it has officially crossed over to building a Custom product. These organizations are writing code to develop their own solution to a problem, and their unique software becomes part of their competitive advantage. They are enhancing existing business models and supply chains. Similar things have been done before, but not exactly *this*.

Custom products are based on a proven business model. When deciding whether a product falls into this category, one of the most obvious questions to ask is whether the company that is building it has any competitors. If there are competitors who are successfully turning a profit, there is no reason a new company shouldn't be able to capture some market share and carve out a niche. In this case, the new software platform (i.e., the Custom product) should give the company an edge to compete in the established industry. On the other hand, if the product does something nobody else is doing, and doesn't fit into an existing business model, it's likely Wild—we'll discuss this product type later.

While building Custom products is not as easy as building Simple products, the process is fairly straightforward. A new platform, for example, is similar to other existing products, so engineers can follow the conventions of other successful platforms to design and develop your product. It's much easier to visualize the roadmap to a complete product when developers are making something similar to other projects they've done before.

One big factor that differentiates the three types of products is how much they cost to build. While a Simple product is relatively cheap to assemble—between a few hundred and a few thousand dollars—a Custom product requires a more significant budget. This means the founder will have to either raise capital, make a sizable investment themselves, earn revenue from another form of business, win the lottery, or somehow find a way to cover the development costs. In general, it's going to cost at least $100,000, and could scale quickly, to build this type of product. Companies building Custom products might raise about a quarter million to a million dollars during their first funding round. Companies building Simple products, however, will usually need less than $100,000 (sometimes even under $10,000) to get going, and the founders often cover this themselves, or borrow it from friends and family.

In total, across all rounds of funding before getting acquired, SWAG.COM raised about $2 million. This might seem like a lot of money, but it's actually not that much in the realm of valuable platforms. The company ultimately sold for many times that amount. This is a pretty typical volume of capital to raise for a company that is building a Custom product.

As we mentioned, savvy entrepreneurs will often start with a Simple product to test an idea and see how people respond, before moving on to build what they know is a Custom product. If there is a favorable reaction to the Simple product, the founder can either directly use the profits to build a Custom product, or use the promising results to raise capital from investors, or both.

When Jeremy and Josh came to me in the midst of their problems starting up SWAG.COM, I immediately saw they were putting their product on the wrong level. They were building a Simple product, when they needed a Custom one. There was no way an off-the-shelf platform would stand up to all of the customizations they wanted.

For one thing, they had an insane number of products available for purchase. And, the user was supposed to be able to upload their logo and place it on the product. Their platform also prompted changes to the order flow depending on the specific type of product. For example, ordering custom T-shirts would prompt a different flow than ordering custom hoodies or mugs. The site also had to display live inventory data throughout a convoluted supply chain and production process. The technology was even supposed to automatically coordinate with manufacturers and shipping centers to manage inventory and order fulfillment all the way through to delivery.

As soon as I heard all of this, I knew they needed a fully custom platform. In fact, they even needed their own API. Any time a company has to build their own API it's definitely not a Simple product. It's either Custom or Wild.

Another example of a Custom product is Booking.com. It was certainly possible to book hotels before booking.com was created, but this innovative product shook up the entire travel and hospitality industry with its custom solution that catered not only to travellers but also owners of estates.

Is It Time to Hire a Full-time CTO?

Building a custom platform in any industry is a challenge, but that doesn't mean it requires a full-time CTO right away. In fact, one big mistake many founders make is hiring a CTO too early. This can actually be a huge problem because executives are expensive, and every dollar counts at a start-up. Additionally, a full-time CTO isn't necessary at this point. While there are certainly some important technical decisions to be made in these early stages, there is realistically only an hour or two of actual CTO work in a week, tops. Yes, you need some kind of partner who understands the tech and can advise you, but you don't need a full-time CTO yet.

Wild Products

Products that involve inventing new technology or developing an innovative business model are known as Wild products. This type of product requires significantly more expertise and capital to produce. Even though Wild products involve very complex technology, it's important to keep them as simple as possible.

One obvious sign that a product is Wild is that new technology has been invented. If any aspect of a product is something you would be able to patent, that's a Wild product. You've created something revolutionary. If your product involves AI, augmented reality, virtual reality, robotics, 3D printing, NFTs, or blockchain, it's almost certainly Wild. Another way to think about it is when a company considers a novel technology to be part of its competitive advantage, that's a Wild product.

The same is true from a business perspective, as well. Sometimes a product comes along that introduces an entirely new business model. Think Apple, Microsoft, Dell, Intel, DeepMind, Google. If you're developing a product that genuinely doesn't have any competitors because nobody else is doing what you're doing, that's a Wild product. Look around for examples of other companies that are successfully doing the same thing you're envisioning, or something very similar. If you can't find any, that might mean you're creating an entirely new business model. Welcome to the Wild.

While hiring a full-time CTO is a waste of resources at a company building a Simple product, it's a must at a start-up building a Wild product. **The biggest misconception at this level is that you can outsource your innovation to someone you don't know.** It's a big mistake to hire a development team without a full-time tech genius onboard to keep an eye on the big picture. The team will run around in circles and waste money. If you aren't a tech genius and you have a Wild kind of idea, start looking for a CTO or technical co-founder fast.

Of the three types of products, a Wild product is the most costly to build, by far. For one thing, you'll need to employ one CTO or tech co-founder who can scale the team. And tech geniuses are expensive. Also, companies pursuing Wild products tend to require cutting-edge technology, which can be priced at astronomical levels. To complicate things further, the timeline for building a Wild product is often vague, because nothing quite like this has ever been done before. And, time is money.

I met a guy the other day who is building a new type of blockchain. He is currently raising his first round of funding—$5 million. That's just one round. He'll need multiple rounds to do everything he's planning.

Plus, he won't make any money until the platform is fully functional. That's quite a lot of capital considering SWAG.COM only raised $2 million in total, across all rounds of funding. This is a big difference between Custom and Wild products.

The company Anthropic raised over $700 million to build an AI platform before even having a product out. They have over 10 publications in peer-reviewed AI journals, but they are not anywhere close to completing their product.

A while ago, some founders came to me for help in building a yacht marketplace app where users could rent out their boats. I thought it was a great idea. Then I heard that they wanted to use blockchain technology. The founder got excited as he explained how the company issued tokens to give clients discounts and incentivize customer loyalty and exclusive access. The basic idea of a yacht marketplace app was a pretty straightforward Custom product. But, adding in blockchain technology pushed things into the Wild category.

When they'd finished explaining the idea, I smiled and said, "This sounds great…for someone who doesn't understand technology. But, honestly, from an architecture perspective it makes no sense to incorporate blockchain so deep into this product."

The room got very quiet and they both turned to glare at me. I gulped. The silence was excruciating. Then, after an agonizing moment, they both broke into smiles.

"Hey, I like you!" They laughed. "You're no bullshit."

In this situation, I did what I wasn't able to do all those years ago with Pavel. I pushed back and advocated *against* the client's ideas.

The product ultimately ended up using light integration of a blockchain. The platform is practically the same for all users, with a small marketing tool that engages blockchain users to get discounts if they've got tokens. That's it. This kept the product in the realm of Custom.

Even in the Wild category, it's still important to build a simple product. In fact, focusing on simplicity may be even more critical at this level because when a company pursues highly revolutionary ideas, it's easy for things to get too complicated. Remember what you value most. Don't get pulled away from your core mission. **Make sure you are solving one very specific problem for your customers.**

Don't Get Your Type Wrong

There are two ways a founder can be wrong about the technology required to build a product: overestimating or underestimating. Each of these scenarios can be disastrous. Some founders also overcomplicate their products, adding AI and blockchain in places that don't make sense or inventing new business models unnecessarily. Others oversimplify things, imagining it will be much easier to build features than it turns out to be. In all of these cases, the actions founders take are misaligned with the steps that will propel the company toward success.

Overestimating the Product

Imagine you assume a certain product is Custom (but, it's actually Simple because there is an existing platform that has almost all of the features you're looking for). You hire a development team to build a custom platform for you. And you spend months working with the UX designer on renderings, providing feedback to the software engineers, and testing the software for bugs. Finally, close to a year later, you're ready to launch.

While you were doing all of that, maybe a competitor quickly built a site using a basic no-code platform in two weeks and spent the rest of the year marketing, forming partnerships, and making sales. So, even though you are only just starting out after a year of working on your site, you're really a year *behind* other business owners who correctly assessed their technology needs as Simple right from the beginning. You spent a bunch of money, ramped up a high burn rate, missing a decent piece of equity because of an investment round and have overcomplicated your platform.

Another serious miscalculation can occur when a founder thinks their product is Wild, but it's really just Custom. There are many recent examples of start-ups trying to build their own data science models when a majority of generic machine learning models are solved and available to use via simple APIs. (We'll discuss AI innovation in more detail in Chapter 10.) These start-ups could spend over a million dollars on revolutionary technology. Or, they could use existing APIs that can do most of what they want. The quicker a founder realizes they are overestimating their product's technology needs, the better.

I talked to a founder last week who is raising a few million dollars to build a platform for purchasing reactive agents for biochemistry labs. She's planning to build a huge, custom e-commerce platform. If built from scratch, it's going to cost a fortune and take up a lot of time and energy. She doesn't really need to do all of that. Instead, she could research multi-vendor marketplace software online and likely get a platform running for under $10,000. She thinks she is planning a Custom product, but really it is Simple.

Underestimating the Product

The situation gets equally scary when the tables are turned. When entrepreneurs underestimate the complexity of their product, rather than overestimating it, things tend to go wrong in a different way. Think of how it might play out if you assumed your product was Simple when it was actually Custom. You might go ahead with building it yourself. You'd create a WordPress site and install lots of plug-ins to achieve the many features you envisioned. You'd likely end up piecing together a collection of different technologies to make the site work. As time goes on, you'd realize there are not enough plug-ins or customization features to make all pieces work together. It feels like wearing clothes that are held together with just a single, fraying piece of thread.

Using too many different technologies together in one product is an invitation for things to break. Each additional plug-in or service that gets added to a site, for instance, makes it more fragile and difficult to scale overall. In mission critical environments, these products tend to fail. When this happens, the damage to your reputation or cashflow can be irreparable.

The Problem with Buying Existing Software

Consider this example. One client wanted to build a platform that could compete with MailChimp, a popular software that enables users to send email blasts to large lists of subscribers. He was setting out to create a Simple product when he was actually envisioning a Custom product.

He thought he would be able to buy some generic, bulk email software and hire a developer to make a few tweaks to it. Except, he had a very specific vision of exactly how he wanted every little detail of the platform to work. He also had a lot of ideas, like integrating a custom supply chain, holding giveaways, and allowing users to run different types of ad campaigns.

The big problem with purchasing existing software is that it is resistant to change. There are really two ways to change existing software: hire the same team that built it to change it, or hire someone else to do it for you. Both options are problematic.

Option 1: Hiring the original team.

If a company is selling software cheaply, they are probably selling it “as is,” meaning they would have a high price tag on customization. Also, they are probably customizing their software for many companies like yours. So, they might be quite jammed keeping up with many product versions and clients. I’ve never had a good experience with hiring the original teams behind cheap software.

Option 2: Hiring someone else.

This option is problematic for two reasons: code quality and code stability. You know how you lose a warranty on a car if you try taking it to an unauthorized dealer for a fix while it is still under warranty? Same goes with software—if you change it, you’re putting its stability at risk. Making an engineer responsible for changing a decent chunk of existing software carries not only stability risk, but also inability to get updates after purchase, as well as infrastructure and security risks. Not to mention how changes would impact the scaling of such software.

In summary: Only buy software if you are ready to use it “as is,” and only if it has been proven to be stable by other clients in live products for long period of time.

Overcomplicating the Product

These days, people frequently tell me they want to build their own blockchain. I think some people imagine fad technologies work somewhat like hot sauce. Can’t you just add a dash of blockchain to any app idea? Won't it make everything taste better? Can you sprinkle some machine learning on top and serve it on a bed of artificial neural networks?

This signals a misunderstanding of how big an undertaking it is to build a blockchain. Some imagine this might be something like a Custom project, or even Simple. Shouldn't there be off-the-shelf blockchain solutions by now? In reality, any product that includes blockchain integration is Wild. Integrating a blockchain is such a huge architecture, infrastructure, and software task, it has to be Wild. It involves cryptography, performance computing, distributed computing—not exactly stuff you can figure out by watching a few YouTube videos.

A similar phenomenon plays out with AI as well. Some people just throw AI into their pitch deck, thinking it could only make the idea sound better. What could be sexier than AI? Everything suddenly feels more cutting edge when it has AI integrated. If an idea is good on its own, it could only be better with AI, right? Founders jump from what would have been a very basic Custom product to a difficult, overcomplicated Wild product.

That being said, use of AI is picking up rapidly with emerging no-code tools that allow you to build comprehensive models to integrate into your product. This means a lower bar for entry, but also easier integrations down the line, which would keep Custom products from entering the Wild category.

I had two different people in my office last week with the same problem. Both said they wanted to build their own blockchain (yeah, you'd be surprised at how many of those we get). But, neither of them had technical expertise in blockchain networks. They both had the attitude that it should be easy to find someone to develop the software. Also, both entrepreneurs had already spent over $50,000 on software development.

These two founders didn't realize they were stepping into the world of Wild products. They didn't need to look for a development team, they needed to hire a full-time tech genius to advise them in exchange for a share in their company, or a combination of salary and vested equity if they are hiring someone they don't know. Using freelancers to build a truly revolutionary product doesn't generally work all that well. Freelancers do what they are told. For more abstract projects, you can't tell collaborators exactly what to do. They need to get creative.

Move On Up

A product doesn't have to stay at the same level forever. In fact, some products progress from Simple to Custom, and then gradually up to Wild. A Simple product is much easier to start, but a Wild product leads to a better valuation. So, there is a strong incentive to climb the ladder and push the value of the product higher.

Investors generally calculate the value of a company based on a certain multiplier of its revenue. A company with a Simple product would have a lower multiplier than one with a Custom or Wild product. This is because a Simple product is actually less competitive—widely available technology doesn't give you any real edge over the competition. Anyone could easily copy your product tomorrow if it was so simple to make. On the other hand, a company with a custom platform would command a significantly higher revenue multiplier because their software gives them a unique advantage. For Wild products, the size of the multipliers can be sensational.

One good example that shows how company valuations can vary is our client who does small business loans. Many small businesses make money, but don't have access to loans, so this company will lend them money through their app. They built an entire custom platform to manage their operations and perform faster due diligence. They now have automated tools for onboarding new clients, too. They are called Paintbrush (getpaintbrush.com). Reach out to them if you need $50,000 debt financing and tell them I sent you!

SMB lending is a tough space. Here, you don't usually get past conservative multiples of 1.5x-2x revenue. Innovators like Paintbrush, though, who have everything automated may get a true tech company multiple of 7x or more.

Another good example of varying company valuations is SWAG.COM. Conventional branded product companies are old school. They have been around for a long time. These types of businesses typically get low multipliers, in the range of 1.5x-2x revenue. But, SWAG.COM sold for well over 2x. They are now working on developing a brand-new business model within their industry, so they may move into Wild soon.

Most unicorns valued at more than $1 billion are in the Wild category. However, it comes with a price. Most start-up failures occur in this category. It is extremely hard to pull Wild products off the ground. If you're going to be disrupting business models, it's good to start conservative. You can get established, learn the market, make connections, look for sources of cash flow, and gradually build up to Wild. This way, you have a solid foundation on which to build some crazy dreams.

So, what does it take to build Simple, Custom, or Wild products? The following chapters will explore key strategies and tools needed to build each of the product types, moving gradually from somewhat obvious topics like website builders to the wild world of AI innovation. My goal is to inspire you to become a little bit more technical and to be more confident about the areas of your product that you don't know, yet.

Are you ready to get to work?