THE STRATEGY OF STARTING WITH LESS

BY FRESH CONSULTING

freshconsulting.com

TABLE OF CONTENTS

SO YOU HAVE AN AMAZING IDEA...

It does ___, allows customers to achieve ___, and has ___ value. It's huge in scope, but that's the point. The multitude of features will allow you to get out ahead of the pack.

Not so fast. While it's tempting to get caught up in the thinking that more is magnificent, stuffing the first iteration of a product with too many features can have a negative effect. If your app suffers from "stuffocation," the early tribe members – your most important targets – may abandon the product due to being overwhelmed and thus, not being able to latch onto the core value. Without support from your early tribe members, you'll be telling people why your product matters, instead of having a fleet of loyalists who show people why it matters.

An alternative is that you can release something less substantial – a Minimum Viable Product, or MVP. It's slim and has the basic features you're interested in testing. It is, admittedly, not perfect, but that's beside the point. Even though your target users might not love it, you believe that through extensive marketing, you'll be able to articulate the product's value to users. Even though it will be rough going at first, you'll learn a lot, getting the maximum amount of useful information about who your users are and what features are important to them while staking your claim on a unique intellectual property.

But again, you risk telling people why your product matters instead of having a bonafide fleet of loyalists who show people why it matters.

What if you took the long view? What if, in the words of William Faulkner, you "[killed] your darlings" (the features you love and couldn't possibly live without) in order to create something slim, functional, and maybe even fun – something truly loveable – thus enabling your early adopters to articulate its value to other users independent of your marketing efforts?

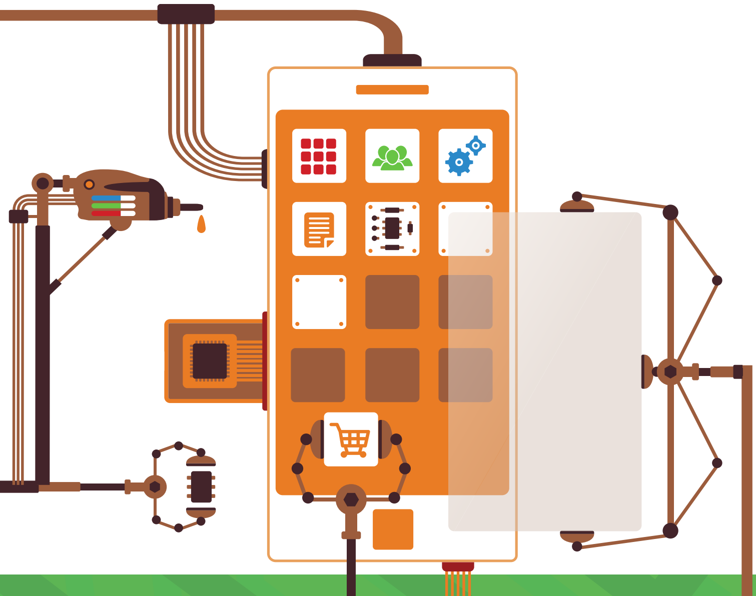
In a perfect world, we could mobilize early adopters to market the product for us. Even though it sounds like a utopian business plan, it's actually completely within the realm of possibility. And it starts with a Minimum Loveable Product (MLP).

THE STRATEGY OF STARTING WITH LESS

A Minimum Loveable Product does one thing really well, instead of focusing on doing too much and losing sight of the central problem you set out to solve.

The ongoing success of your product can be built upon the foundation set by the MLP. If you get people to believe that what you're creating is essential, you afford yourself the capability of continuing to build it out in future iterations. That's where innovation comes in. Once people buy into the value of what you're creating, you can take more risks.

Features aren't bad in and of themselves. The danger lies in not thinking strategically about how quickly you are releasing them – without understanding which features are working well and which are not, you run the risk of releasing features that might diminish the integrity of the user experience without knowing why.



Each additional feature adds an extra layer of complexity, which creates the challenge of rebalancing user flows, visual hierarchy, and information architecture within a finite space. The science and effort required to release a surplus of features and maintain high usability is enormous.

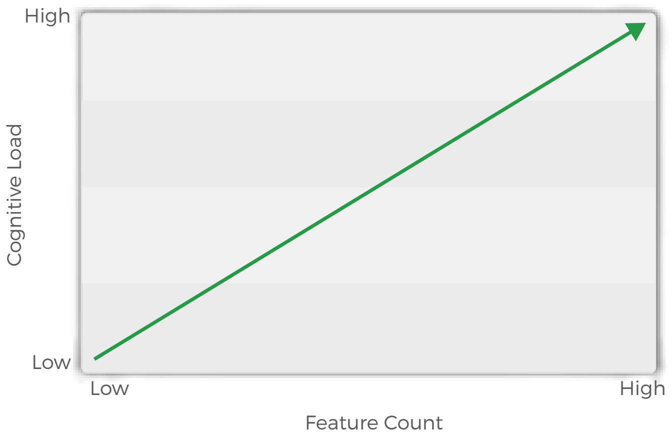
There are instances when the value lies in being comprehensive for niche industries, but it's easy to get carried away with trying to create a product that solves every problem and alleviates every pain point all at one time. Even in that case, a "comprehensive" product may mean 5 key features that do something really well, and not 20.

The strategy of starting with less hinges on showing people why your product matters – via its simple inventiveness, functionality, and self-explanatory value – rather than proselytizing your users to death via marketing campaigns. It is possible to let your product do the talking for you. Differentiating your product can be as simple as creating something that is a pleasure to use, provides a strreamlined acces to value, solves a real world problem, and allows customers to achieve their goal with ease.

The strategy that is utilized at the beginning of building an app is essential, and starting with focus and excellent usability is crucial. Even with less features, your experience can be something truly loveable.

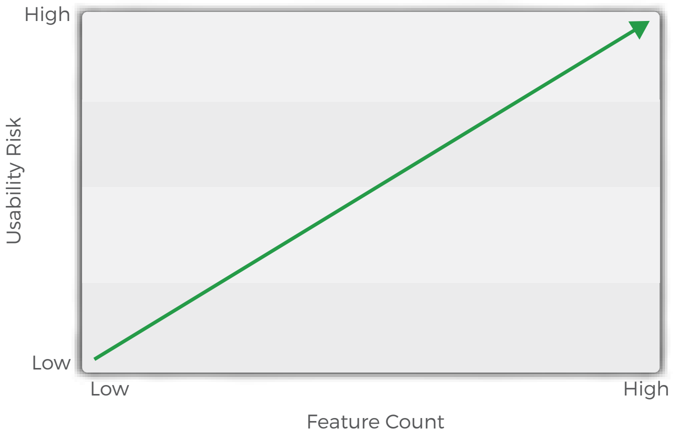
Different challenges surround the release of too many features, which are highlighted in the graphs below.

Features & Cognitive Load



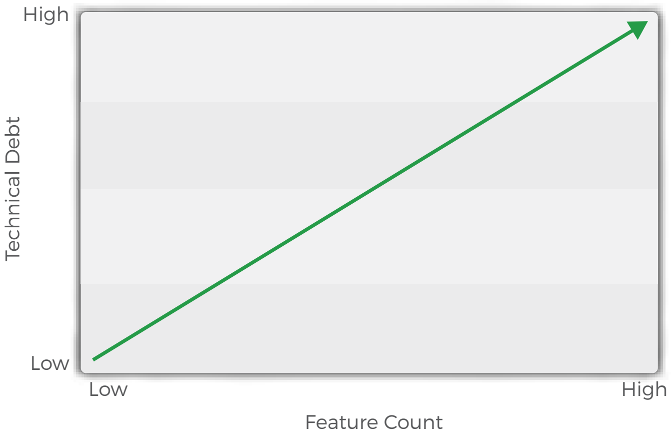
Cognitive Load: Including more features places a mental tax on users, making it more difficult to learn how to use the product

Features & Usability Risk



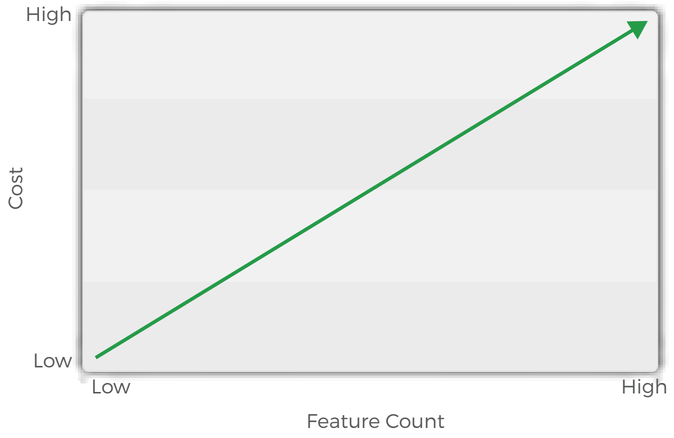
Usability Risk: In an effort to increase feature count, you increase usability risk

Features & Technical Debt



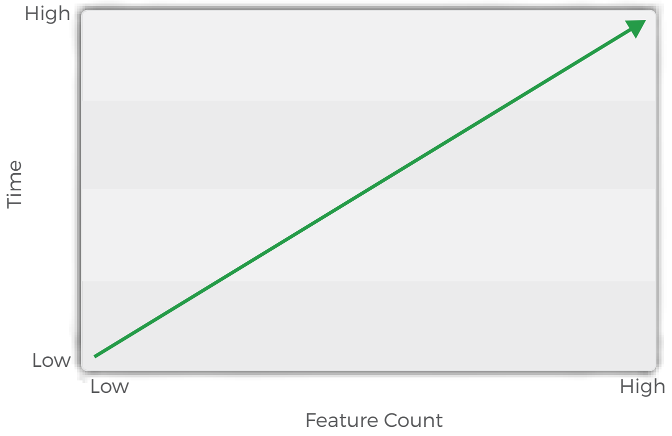
Technical Debt: The more features you develop and release, the harder it will be to change course

Features & Cost



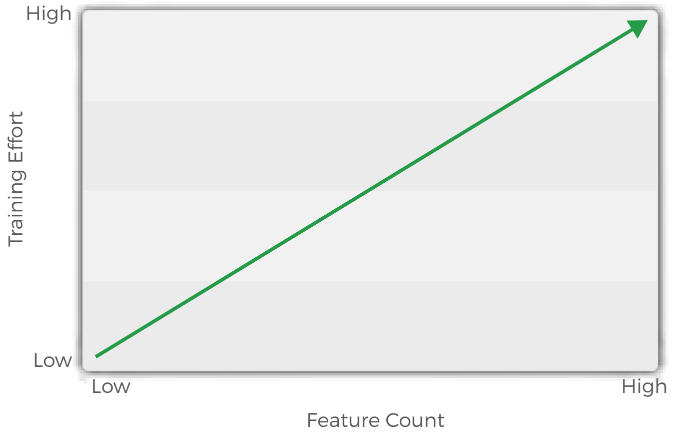
Cost: Developing more features costs money, and having to change the features retroactively costs even more

Features & Time



Time: Developing features takes time, whether it’s spread strategically over a longer period or used on the front end

Features & Training Effort



Training Effort: The more features you include, the more effort and explaining it takes to educate your users

STAGE 1: USE AN MVP FOR EARLY CONTROLLED TESTING

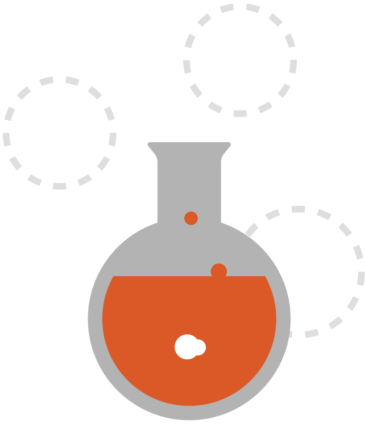
An MVP, or Minimum Viable Product, is the first lightweight version of your product that is stripped down to its most essential features. Creating and releasing an MVP is a strategy that hinges upon getting your product out so that you can actively solicit feedback from its first users (or "early adopters") about what works and what doesn't. This isn't the time to do heavy marketing. Not only does this feedback provide information about the usability of the product's features, but it also provides insight into who your users are, as finding that out definitively in the user research phase is not always realized.

In an article published on Huffington Post, Brian de Haaff, CEO of SaaS Product Roadmap Software company Aha!, argues that MVPs aren't always desirable. He states that "A can of cat food is a Minimum Viable Product (MVP) when you are starving, but it's highly unsatisfying and unlikely to generate a loyal following (of humans)." While an MVP is a useful field testing tool, it won't necessarily make a good first impression, and in the business of making products that people use and creating brands that people become loyal to, making a good first impression is everything.

So, essentially think of an MVP as a field testing tool rather than your go-to-market strategy. As opposed to being used in a sterilized, inauthentic "lab" setting, MVPs are tested by users in their natural "habitats," independent of your instruction to complete certain tasks. Based on tracking the behavior of users and gathering data about what they like and dislike, you can start to hone in on what matters for future iterations.

User Testing is valuable. But user testing and user research are only as valuable as the questions you ask.

You need to drive to answer questions such as "What do users like and dislike about the product?" and "What really matters most to users?" and "What's intuitive and what's not?"



Eric Ries, a pioneer in the Lean Startup Movement, argues in his blog that:

"MVP, despite the name, is not about creating minimal products. If your goal is simply to scratch a clear itch or build something for a quick flip, you really don't need the MVP. In fact, MVP is quite annoying, because it imposes extra overhead. We have to manage to learn something from our first product iteration. In a lot of cases, this requires a lot of energy invested in talking to customers or metrics and analytics."

The key incentive for creating an MVP is to learn – "[to allow] a team to collect the maximum amount of validated learning about customers with the least effort." MVPs are an incredible educational tool. They let you experiment and identify what customers do and don't like about the product.

But to complete Brian de Haaff's analogy, while an MVP might teach you that humans don't like the taste of cat food, at that point you've lost time, money, and the ability to create a loyal following based on the disgusting taste your product leaves in the mouths of your early adopters.

If you do release an MVP, don't market it. Just use it internally to solicit feedback via external testing, without marketing it to others.

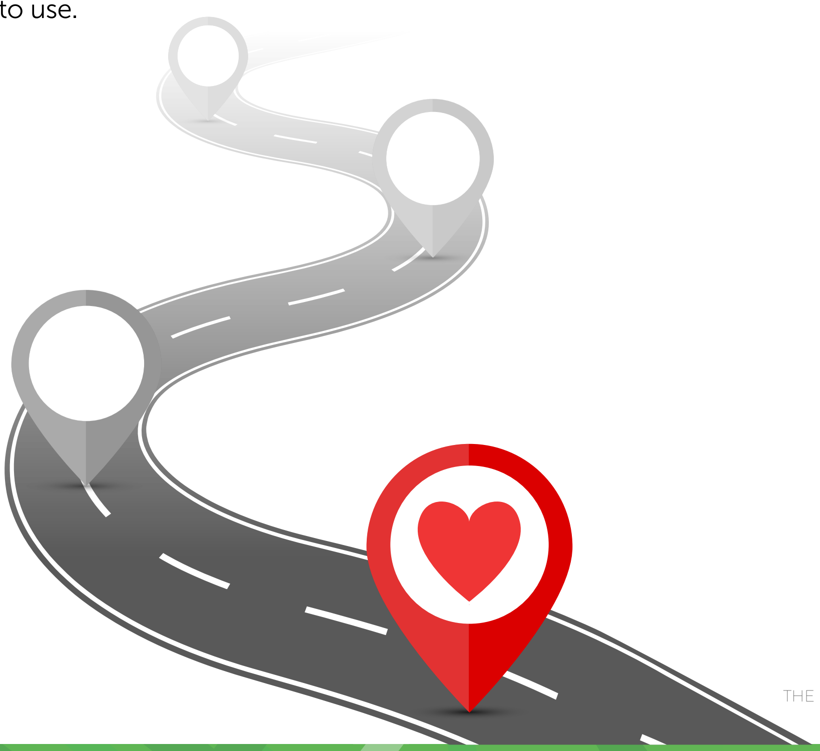
STAGE 2: USE AN MLP AS YOUR FIRST PRODUCT RELEASE

In the design field, there is a fear of over-investing and a danger of under-investing. Many companies perceive that more work upfront won't move the needle, but in reality, starting with more research and less product can allow you to do so. Under-investing creates a risk of missing an opportunity to create a product that people love.

An MLP is similar, in concept, to an MVP. It's a lightweight version of a product. It is the same "stripped down to a core" product that allow(s) you to release something that should accomplish the same purpose as an MVP. It allows the team to learn more about the users and to stake a claim in the intellectual property sphere, but in a way that is drastically different than the MVP. The main difference between the two is that the intention behind an MLP is creating something that is truly delightful to use.

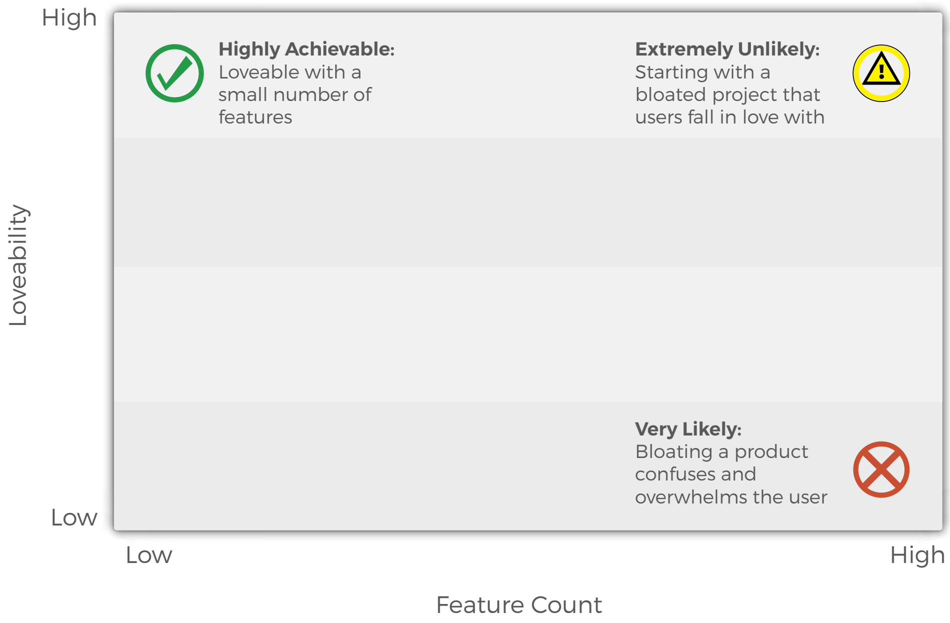
Creating an MLP requires intentional focus and effort. It requires testing with users, which requires extensive design iteration. It also requires paying attention to all the little details that make something elegant or slick.

Do you take the time to develop new features or make something really slick that performs? This is an ongoing debate to consider, and we recommend putting more time into making less features truly delightful to use.



The graphs below show the connection between high loveability and low feature count, in addition to the value that ratio provides. Over time, you can simulatenously increase loveability and feature count as more features are demanded by your users.

Start with High Loveability and Low Feature Count

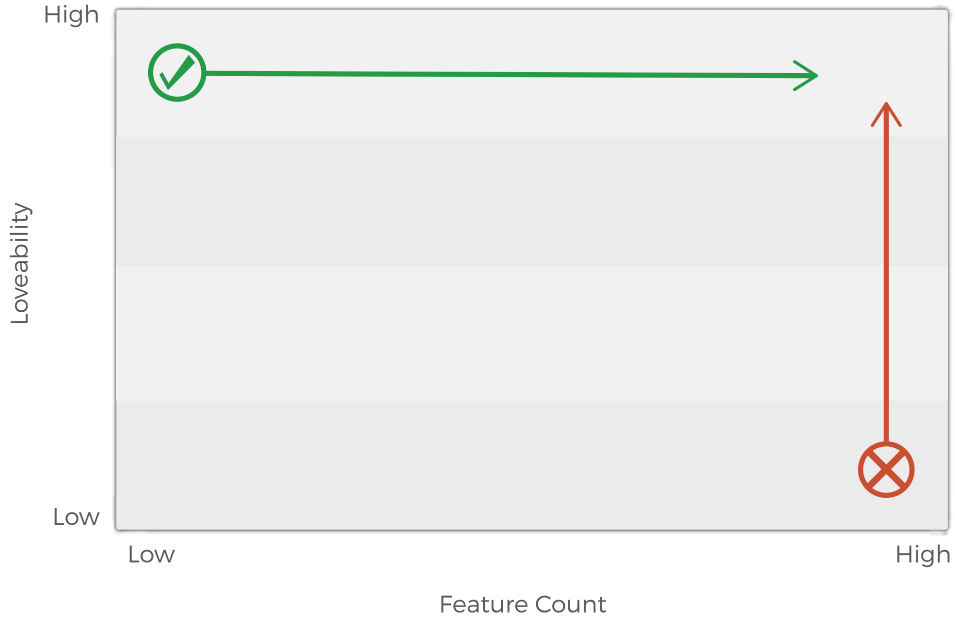

High
Highly Achievable:
Loveable with a small number of features
Extremely Unlikely:
Starting with a bloated project that users fall in love with
Very Likely:
Bloating a product confuses and overwhelms the user
Low
Loveability
Low
High
Feature Count

Loveability vs. Feature Count

High loveability and low feature count is the best strategy. High loveability and high feature count is difficult to execute.

Low loveability and high feature count is a more likely scenario and has the potential to confused users and turn them off of your product.

Moving Horizontally is Easier than Moving Vertically



Increasing Loveability and Feature Count

By moving horizontally and adding features at a rate that doesn't burden users, you increase value as users demand it, maintaining loveability.

Starting from a complicated, feature- rich product and increasing loveability is extremely difficult to pull off: you have the difficult task of educating users on the functionality of the product while simulataneously increasing loveability.

In a research paper titled “Concept Plan for User Testing in Scrum Projects,” three designers from Acato, a design agency in the Netherlands, state:

“Creation of a Minimum Viable Product or MVP [allows] you to ship a product that resonates with early adopters; some of whom will pay you money or give you feedback [. . .] All features, functionalities and wishes are documented as user stories: short descriptions of a feature describing the added value for the user. If certain stories do not have an added value, the team may reconsider if a given feature should be included in the Product.”

Taking this opinion into account, what really distinguishes an MVP from an MLP? Development time? Cost? If a Minimum Viable Product and a Minimum Loveable Product are identically similar in their streamlined nature and intended resonation with early adopters, but the differentiating factor is that an MLP is something people love to use, what does that say about the usability and quality of an MVP? An MVP is a testing tool. In creating an MLP, you will have created a product that people love to use; a product that will carry a unique and compelling value proposition.

An MLP is generally the first iteration of a product, and thus doesn’t carry a promise or prediction of all potential features developed over its lifecycle.

The MLP should immediately convey the magnetic product features that are overwhelmingly valuable to prospective users. In providing these unspoken guarantees – that are experienced by users rather than relayed through marketing efforts – the MLP markets itself.

It’s a prime example of the storytelling principle “Show, don’t tell.” Your product should immediately convey its unique value proposition to your early tribe members, bringing back the maximum amount of love and loyalty from them.

In creating an MLP, your team needs to consider certain questions:

- How do we develop love and loyalty from our early adopters?
- How can we create a unique and compelling value proposition within our constraints?
- How do we ensure that we continue to garner trust, love, and appreciation from our early adopters, instilling in them the belief that the product is worth showing to other potential adopters?

Once again, to Brian de Haaff’s analogy, an MVP does serve the purpose of teaching you what your users like and don’t like, but at the risk of losing their trust and loyalty. An MLP accomplishes the same purpose, but in focusing on creating something that is delightful to use, you can also accomplish the goal of developing more loyalty.

STAGE 3: ALLOW USERS TO GROW WITH THE APP

MLPs may take a slower approach to feature expansion. The key to creating an MLP that conveys a unique and compelling value proposition is to do one thing super well. If the purpose of your app is to allow people to use geolocation services to find mobile businesses, then focus solely on that function. Make it amazing. Make it something that people love.

If you garner support and loyalty from your early adopters, you'll be afforded the opportunity to implement, for example, a feature that serves as a customer feedback mechanism, or a way to share photos and information about favorites, or even the ability for users to broadcast their location to other people within their network so that they can find mobile businesses together. But if you focus on implementing all of those features out of the gate, you risk doing so poorly or being so limited by time constraints and overhead that cause you to miss the opportunity to go-to-market with something unique, inventive, and desirable.

Stuffing your product with an array of features is tempting, but you can design and test specifically for a strong initial feature, avoiding delivering a ton of features that don't add anything to the MLP. Not only can you shoot for a lean and timely go-to-market strategy, but you can also identify what works and what doesn't, avoiding adding features that will add confusion or frustration for users rather than alleviating their pain points.

Dropbox was released with a core functionality: cloud storage and file synchronization. From 2007 to 2011, Dropbox focused on perfecting this basic function before adding additional features such as the ability to upload photos directly, the unveiling of Dropbox Business, and crowdsourced add-ons created by the very users who loved and adopted the original product. Due to the immense popularity of the first MLP, Dropbox was able to start with less and build upon the firm foundation they had created.

Instagram took a similar approach. It started as a photo customization and sharing application, and has since evolved, adding additional features and functionalities as users have grown with the original concept. By starting with a basic feature that users loved, Instagram created a simple roadmap for future success. Features aren't a bad thing, as can be seen in both Dropbox's and Instagram's success. Features just need to be released strategically and at the right moment when your product and users are ready for them. Gmail, Google Docs, Asana, and Basecamp also followed this strategy.

An MLP allows you to find out what people care about. If you go-to-market with one strong feature, you can identify what people love most about it. If you hypothesize that the feature will be valuable and it isn't something they end up loving, you have incredibly useful data with which to reorganize the key elements of the experience. If it is something they love, then you have a foundation to build on. By utilizing qualitative and quantitative data gathered after the initial version of a product is released – through surveys, interviews, or contextual inquiry – you can find what is most important to users. The feedback solicited can be used to inform future iterations of the product.

Focus on solving the most important problems first. This doesn't just need to be done after the product is released, as extensive usability testing should be conducted throughout the entire design process. But instead of focusing on all of the pain points of users, identify the most glaring frustrations and zero in on how to use the features of your product to alleviate them.

"Simple" can be equated with something that is basic, uninteresting, and unimaginative. But it's possible to achieve simplicity that is polished, refined, and stylish. By paying attention to detail you can make your product both slick and straightforward.

Referring back to the earlier example, if you go above and beyond in making reliable and interactive geolocation features, designing for garnering the maximum amount of happiness from early adopters, you'll be able to invest your audience in your product and make them want to use it again and again.

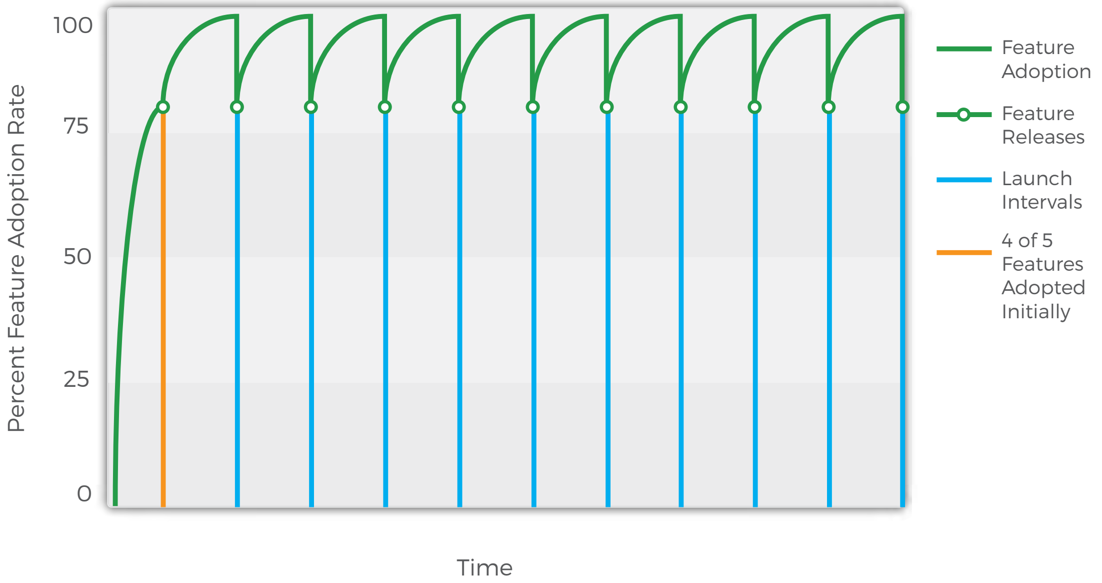
An MLP is a key part of The Strategy of Starting with Less. In order to show the value of your product rather than telling people why it's valuable, you can start with less – but "less" doesn't mean a lower degree of quality. Highly loveable experiences require sophisticated simplicity. You can have a big roadmap of features, with the key being to strategically release features over time.

If you release features in strategic sets over time, you can maintain adoption, creating a positive cycle of learning and mastering features as the product grows. Your aim should be to continually start at a high rate of immediate adoption of new features and push to a theoretical maximum rate of adoption before loading up more features. This ensures that users are not constantly overwhelmed but are comfortable with, ready for, and enthusiastic about new features.

The graphs below show the correlation between adoption rate and initial features released.

The Outcome of Starting with 5 Initial Features

80% Initial Adoption with 10 Incremental Future Releases



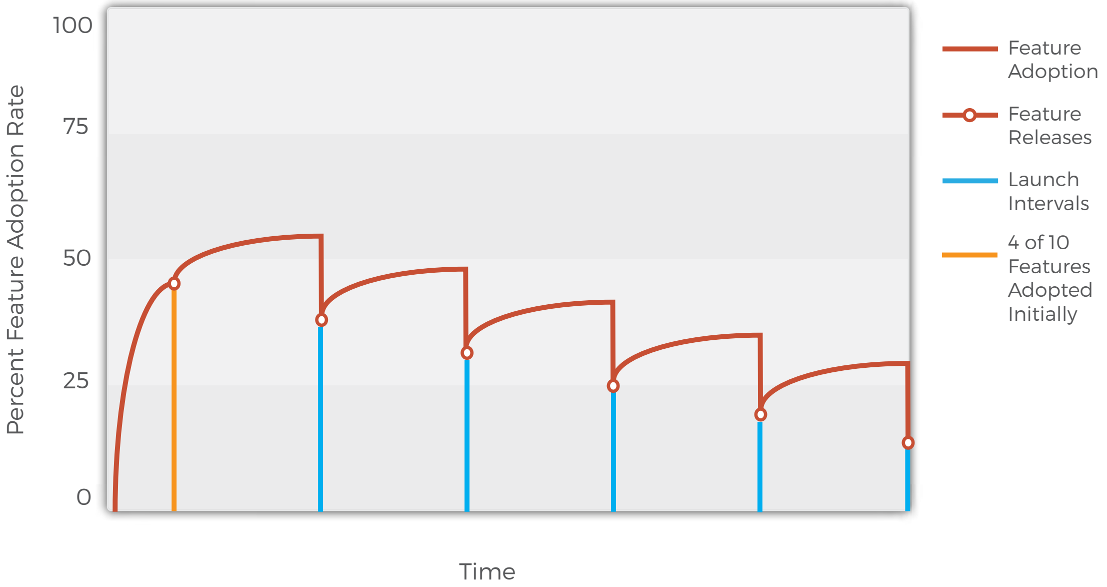
Increased Feature Adoption

If you release features in strategic sets, you can maintain adoption, creating a positive cycle of learning as the product grows. Start at a high immediate adoption of new features and push to a theoretical maximum of adoption before increasing the number of features. This ensures enthusiasm for feature releases and ease of use.

In both charts, the ultimate number of features released is the same (15), but adoption rates differ.

The Outcome of Starting with 10 Initial Features

40% Initial Adoption and 5 Incremental Future Releases



Feature Bloat Leads to Lower Adoption Rates

In contrast, having too many features is harmful. Releasing too many features means that users never grasp the product and feel like they are falling further and further behind. Even if you are equal on paper to your competitor in terms of features and benefits, you might be falling behind, as your value is obscured.

In both charts, the ultimate number of features released is the same, but adoption rates differ.

Users like when features are added that enhance the experience without taxing them cognitively in an overwhelming way.

By initiating a healthy cycle of adoption and implementation of new features as the product grows, you can avoid cognitive complexity that might turn users away.

There's a strategy in the release. If you release 10 new features you may not inspire adoption. The most viable strategy is to release singular or small feature sets, over time. You can always change course if you get too much negative feedback. This is why it is important to truly understand what your customer values and what they do not in order to prioritize which features you release first and in what order.

ALLOW YOUR USERS AND YOUR APPLICATION TO GROW TOGETHER

The Strategy of Starting with Less is a starting point. Your ultimate vision for your application doesn't need to be compromised by starting with less on the front end; rather, the viability of creating a product with a long and fruitful life cycle hinges on having that firm foundation. Starting with less enables you to create something loveable that will allow you to grow exponentially.

Focus on creating a unique and unforgettable experience. Something loveable creates more conversions, and thus, leads to more people buying in. By creating a ton of value, you develop a loyal following that grows with your application.

More isn't better. If you stuff the first iteration of your product with too many features, you run the risk of releasing something disjointed or something that, while amazing in concept, lacks the quality to stand up on its own.

Your early tribe members are your target and without them, you'll be telling people why your product matters, instead of having a contingency of believers who show people why it matters.

Design should be informed by value. Let the strategy of starting with something single and loveable be what differentiates you from the competition. Through inherent simplicity and usability, your product can relay the message, taking the pressure off your team to tell people why it's valuable. Differentiating your product depends on the strategy you start with, and creating something small and truly loveable will allow you to take more risks and become more innovative as your users and your application grow together.