Product Management Handbook

Stop and think

What interests you about Product Management?

Do you want to be in charge of a team?

Do you feel like a generalist who can't really focus in on one specialty?

Do you want to have your hands in everything?

Do you like to stay "high-level"?

Are you interested in switching into tech but don't want to learn anything technical?

If the answer to any of those is "yes," you may wish to reevaluate your choice. Product Managers influence without authority. They have a high degree of specialty in a discipline that isn't often taught: Product Management. They don't actually get to do the work of every discipline, but they step in anywhere that is necessary.

The last in particular is most important. Product Managers need to have deep respect for each of the disciplines they're interested in. To be the best you can be at Product Management, you'll need to want to be a PM. You'll also need to be at least a little curious about what it's like to be a customer service rep, a developer, a designer, and any other role you may work with closely.

So what are some good reasons to get into Product Management?

If you work with Product Managers and want to understand what they're doing. Especially if you'd like to understand what makes a Product Manager particularly good or see ways you might be able to help a less effective PM switch techniques.

If you're interested in shifting into Product Management from an adjacent discipline and want to know which of your responsibilities will stay the same and which will change.

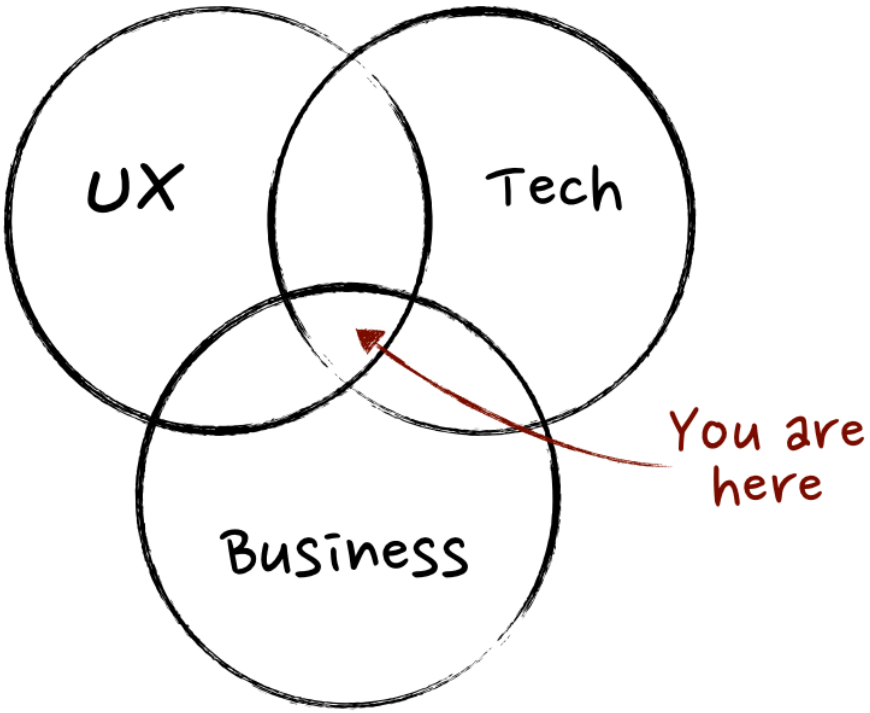
If you shifted into Product Management by mistake and are looking for some quick frameworks to help you evaluate.

If you're an entrepreneur, haven't been a Product Manager, and know you won't be hiring one for a while. This will help give you some ways of thinking to release the first version of your product.

And, in my mind, the best reason—because you're curious. Product Management has been widely discussed recently, and it's always fun to know what's going on.

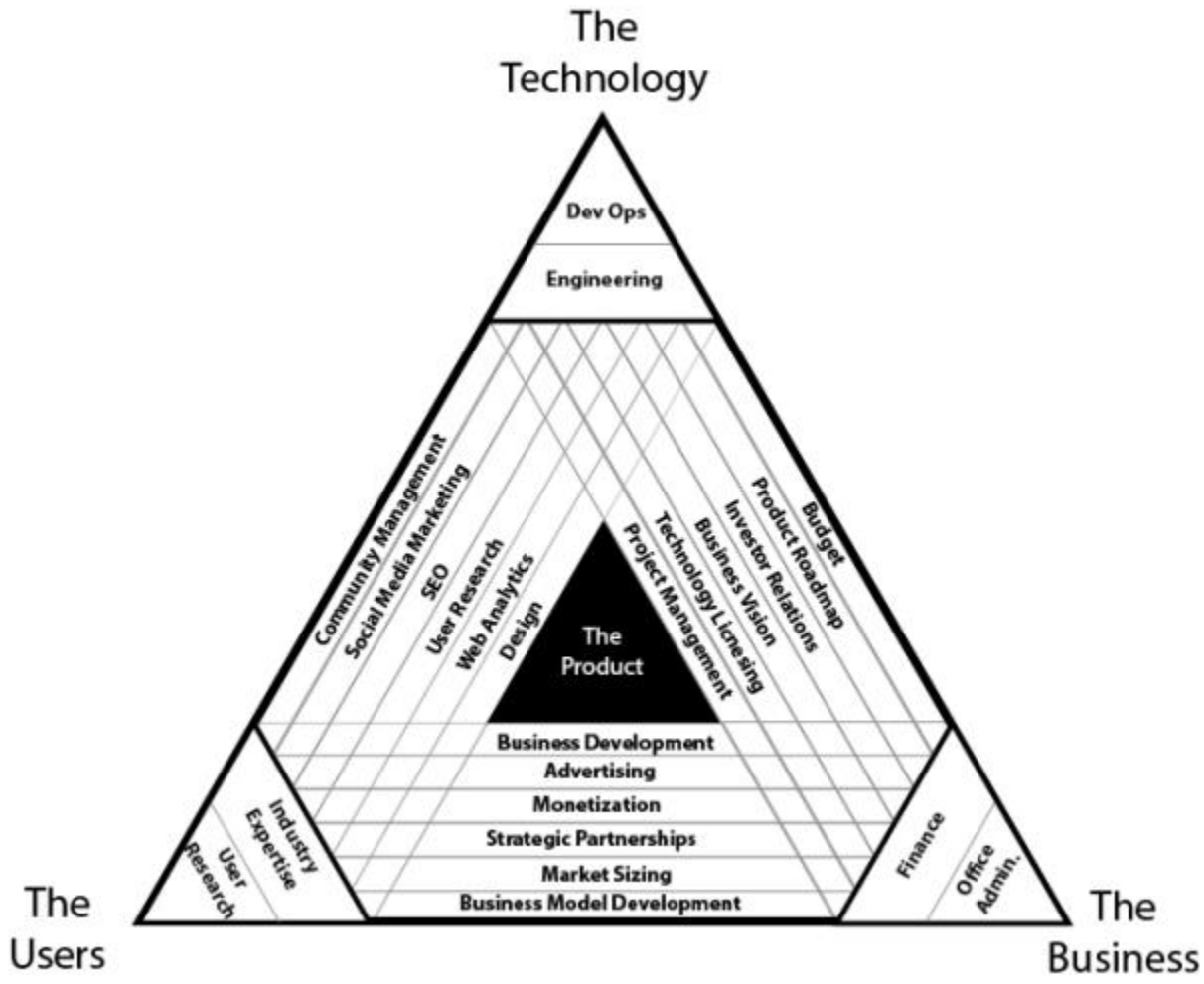
The Three Core Disciplines of Product Management

Now that we've talked a bit about what you shouldn't be hoping to be as a PM, let's talk about what you should. Product Managers balance three key areas in a company: business, UX, and technology.



© 2011 Martin Eriksson | mindtheproduct.com

This doesn't mean that each PM is equally strong in each area. Most PMs rely heavily on a discipline that they worked in before they were PMs. A good explanation of the skills that PMs can bring from each area is:



It's typical to start by being excellent in one of the areas and weak in the other two.

When Product Managers are looking to grow, they have three options:

1) Building skills in the core of their wheelhouse. For example, a UX-focused PM getting even better at UX skills or learning how to do research studies.
2) Building adjacent skills. For instance, a UX-focused PM thinking about using community management techniques to grow the user base.
3) Building upon weaker skills. For instance, a UX-focused PM thinking about SEO.

In many disciplines, individuals focus on building upon their strengths. In Product Management, this can be risky. A PM who can build a lovely product but can't get anyone to buy it won't succeed at the core mission of their job. They need some sense of business skills or an ability to work well with strong business stakeholders. This can work if you have a balanced team of Product Managers who are strong in different disciplines.

A good way to assess what to learn next as a Product Manager (or aspiring PM) is to ask "where is the weak link in my product?" If you're building a lovely product but it isn't growing, time to focus on growth skills. If you're growing like wildfire but have a ton of support tickets saying that people can't find existing features, time to focus on user studies and discoverability.

Opportunity Assessment

Opportunities for new products exist all around us, in every market – even mature markets. This is because what is possible is always changing. New technologies are constantly emerging, competitors come and go, and new people with new talents and new ideas join your company.

The product manager must be able to quickly evaluate opportunities to decide which are promising and which are not; what looks appealing, which should be pursued, which are better left to others, and which ideas are not yet ready for productization.

Ten Fundamental Questions Product Managers should answer:

1. What problem will this solve? (Value Proposition)

2. For whom do we solve that problem? (Target Market)
3. How big is the opportunity? (Market Size)
4. How will we measure success? (metrics/revenue strategy)
5. What alternatives are out there? (competitive landscape)
6. Why are we best suited to pursue this? (our differentiator)
7. Why now? (market window)
8. How will we get this product to market? (go-to-market strategy)
9. What factors are critical to success? (solution requirement)
10. Given the above, what's the recommendation? (go or no-go)

Refer examples for Opportunity Assessment here.

What a product opportunity assessment isn't – An opportunity assessment doesn't aim to provide a solution to a business or customer problem. In contrast, it helps answering the question whether it's worthwhile solving the problem in the first place. Another way to look at this is to look at the value or outcome that you're looking to provide to your business or customer.

Talking to Users

One effective way to get a foothold in the "design" sphere of Product Management is by talking to users. The techniques underlying design come from liberal arts disciplines: ethnography and anthropology. "Users" is just a word to summarize anyone who is using your product and adds some clarity. Some Product Managers use "people" internally as a more human term.

In starting to research your users, you want to get a sense of what they're doing now, why they're doing it, what their pain points are, and how your product could possibly address them.

A good place to start is open-ended questions. This is easiest if you have a consumer-facing product and can practice in day-to-day life. If you're working on travel, for instance, it's easy to say, "What's your next vacation?" If you're working on a product like Highbrow, you can say, "What are you hoping to learn next?" or "What's the most interesting thing you learned recently?"

When having these conversations, try to avoid directing the user. The worst types of questions to ask are closed-ended or leading questions—"How much would you pay for this?" "Would you use X?"

A good rule of thumb is that after about 10 conversations, you'll start to see common themes come up. Those are likely the biggest areas to focus on as a Product Manager. It might even start to happen after as few as six conversations.

It's easy to do this if you're at the very beginning of a product's development and aren't committed to anything yet. That might lead you to an insight like "people are interested in learning in small chunks, rather than investing in a giant class," which would foster a product like Highbrow. It's more challenging if you're already working with an established product and are trying to add new features. In this case, the research can still be valuable to get up to speed and start to build underlying intuition and empathy with your users.

Market Research

Why Research?

There are different reasons to perform market research, which can change the kind of research needed.

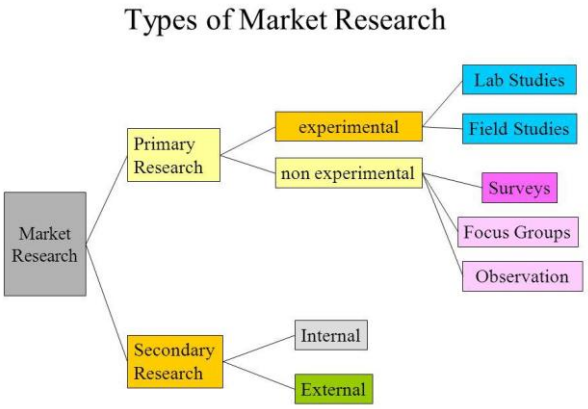
- **New product development**—here we need to investigate the validity of a market problem and to identify the market that has that problem. Many new products are developed from great ideas. That does not need to be a bad thing. But if the idea is not first validated against a measurable market problem and a defined target market then the risk is you over invest too early only to find out you have a solution looking for a problem. Sound expensive? You are right.

- **Incremental improvement**—many product professionals are working on existing products that have an existing customer base. So, no need to validate the value proposition of the product. However, it is tough out there—competition is fierce, customer needs are changing, margin pressure is a constant. To remain competitive, you need to strive for differentiated

value. A great way to do that is keep your finger on the pulse of what your evolving customer pain points are and develop solutions to them that are as unique as possible. Differentiated value commands a price premium because your value is, well..... rare.

- **Usability Refinement**—for some products, the user experience is paramount. It can be a source of competitive advantage. No time for guessing here. You need to engage with your customers, specifically your users, to get direct feedback on how they feel and react to new design options.

Types:



Primary Market Research is a kind of research which is done by company, without using any information that is already made available through other sources.

Secondary market research focuses on using information, or data, that was already gathered and analyzed. Usually information is taken from different free or paid sources.

Tools:

Customer Surveys	Site Analytics	Data Mining
Site Visits	Personas	Usability Testing
Competitive Analysis		

These tools help answer following questions:

1. Do you understand who your users really are?
2. How are users using your product?
3. Can users figure out how to use your product? Where do they stumble?
4. Why do users use your product?
5. What do users like about your product?
6. What do users want added to or changed in your product?

Limitations:

1. Notice that while these questions are critically important, they do not directly address the fundamental question for most product people**:** *What product to build?*
2. Winning products come from the deep understanding of the user's needs combined with an equally deep understanding of what's just not possible.
3. Use market research tools to help refine your product and make it as good as it can possibly be. Just don't expect the techniques to produce the idea for the next Facebook, Flickr, or YouTube.

References:

Use Customer Research to Create Products by Intuit Product Manager

---

Competitive Analysis

Competitive research involves capturing information on competitors' metrics that matter most to your own business. Product managers should begin by identifying who and what the product is up against in market. This gives them a strong understanding of how unique their idea truly is. This is also an important first step towards understanding whether their idea warrants more of their valuable time and effort.

Competitive research can help answer these core questions:

- "Are there other companies doing exactly what I (want to) do?"
- "Are my potential customers getting a product or service at the level that they want or need?"

To answer these questions, product managers must differentiate between direct and indirect competitors:

A direct competitor is a company that offers (more or less) the same good or service within the same market. For example, Coke and Pepsi are direct competitors with each other.

An indirect competitor is a company that offers a different type of product to serve the same need. For example, Sprite and Pepsi are indirect competitors with each other. And water is an indirect competitor to both Sprite and Pepsi.

Product managers should be sure to research both direct and indirect competitors, since both types of analysis are immensely valuable. Understanding the pain points of their competitors' customers can help product leaders discover problems that are theirs to solve. This can offer them a huge market advantage to use in their pitch.

Why does it matter?

Product managers must prove they are building critical components that matter to their business and potential customers. After all, products should never be built in a vacuum — they should solve specific needs for the business and customer. As the product's CEO, they must define these two things before motivating the product team to build, market, sell, and support the solution.

Product managers need quantifiable metrics to:

- Compare their work to the competition
- See where they can fill gaps in customer competition

What information is needed?

Product managers may understand why competitive research matters, but not know what to look for during their research. They should look for certain core information to start making a meaningful competitive research plan. An example from Aha! is included below. PowerPoint and Excel are also popular ways to capture this information.

Vision

Product managers should determine their competitors' end goals for their products — their visions for where these products are headed and what they aim to achieve in market. To achieve this, they should spend time on competitors' websites to assess how they present themselves to customers and prospects. They should ask: "Why do these products exist?" "Which problems do they purport to solve?" Most importantly, they should ask if there are any problems that these products do not seem to solve. Those are potential market gaps to fill.

Strengths

Product managers should try to understand what drives their competitors to do what they do. Many products are built out of deep, personal passion and necessity. It is smart for product managers to spend time researching the founders' backgrounds on LinkedIn, company websites, and other online profiles. Then, they should ask: "What does the competition excel at? What insight and experience do the founders bring to the table?"

Customer Challenge

What do competitors' users struggle with? Which aspects of these products are lacking? To answer these questions, product managers should invest time in taking competitors' product tours to understand all aspects of their offerings. Then, they can search for relevant online forums to read reviews and insights on these products. Quora, Product Hunt, and LinkedIn are three potential platforms that offer unbiased opinions on what customers and prospects think.

Personas

All great products have relevant personas — profiles of that product's ideal customer demographics. These personas are fictional, but should reflect real customer groups. While researching competitors, product managers should ask who these products are intended for, and revisit customer reviews to look for patterns. They can infer which personas competitors are targeting by noticing similarities between titles, industries, year of experience, etc.

Positioning

How do competitors market their products? What language do they use to describe what they offer in market? Which core problems do these products solve? Product managers should review their competitors' marketing messages — on their company websites as well as external websites. This helps them assess where they see their own role within this shared market.

Market

It is essential to define the market landscape when conducting competitive research. A thorough market analysis confirms customer needs, industry changes, and fiscal opportunity. To understand competitors, product managers must also know where they fit within the broader market they share.

Product managers should collect this information for each direct and indirect competitor, and then store it in a central place accessible to their team. This helps them analyze all competitors at a high level. Once they see where their competitors excel and fall short, they can find ways for their product to stand out.

Self-check

Throughout each stage of the competitive analysis, product managers should check in with themselves to benchmark their progress. They should ask if this process is helping them understand their core market — and how their product or feature will solve a specific problem. Ultimately, they want to answer this question:

> *"What's your single biggest advantage that you can offer customers in your target market?"*

The first step towards finding out this answer is conducting competitive research.

What Can You “Sell”?

Once you feel like you have something people want or need based on research, a Product Manager needs to assess what will be viable.

There are two primary ways this works.

(1) In most business-to-business software, the Product Manager evaluates what people will pay for. If you can sell a spreadsheet prototype of your product to a customer, chances are you're onto something. Some consumer scenarios work basically the same way. While consumers are unlikely to pay for a spreadsheet, there's lots of consumer software that people are willing to pay for.

The simplest way to test this is to actually sell something. If you can build something simple and customers are already willing to pay, that's great. Another option is to test if customers would be willing to pay later. Hemingway App, a writing tool, added a button for "$5 to buy the desktop version." Enough users on the free website version tried to download that it was worth investing in the desktop product. Oftentimes, when you see a product that's "out of stock" indefinitely, it's a PM testing if people are willing to pay.

The next level of this is making sure customers are willing to pay enough. A PM should be thinking about how much a customer costs to acquire (CAC) vs. how much money the customer will generate for the company over their lifetime (Lifetime Value, LTV).

(2) The other option is in consumer software that isn't focused on direct monetization. This is based on generating a large user base that's valuable to advertise to.

In this case, you should be more concerned with the metrics around getting more users, not metrics around payment. The most common set of metrics used for this is "AARRR"— Acquisition, Activation, Retention, Referral, and Revenue, as coined by Dave McClure.

Keeping track of users as they go through this process is considered funnel analysis. A PM will focus on where and why people are dropping off. If you have lots of users visiting your site but no one signs up, your value proposition might not be coming through. If you see lots of users sign up, try the product once, and leave, that's another problem.

No matter which model you're using, a PM figures out what's happening to users as they sign up and pay for (or just use) the product. If the company isn't meeting its goals, the PM figures out where the problem is and how to resolve it.

Tradeoffs

The crux of the Product Management role is about making tradeoffs. It's rare that the next action is obvious. There isn't one good formula for how to do this. Over time, as they see more situations, Product Managers build intuition for what tradeoffs will be effective.

The first type of tradeoff is "which is most important?" This comes up when different angles of the PM's job suggest different actions. For instance, what do you do when the business goals and user testing suggest different paths of action?

In this case, the PM is likely to align to the vision of the company. If the founders of a company have said that monetization is the most important element for the year, the business goal is likely to take priority.

Beyond aligning to vision, the PM is also likely to consider "what have we done recently?" If the company has constantly been investing in making great features for users without attempting to monetize, that may be a good shift. On the other hand, if the company has been focused on

growth and users are starting to churn out (leave) the product because it isn't good enough, it may be time to focus on retention and user functionality.

The second type of tradeoff is "how much is enough?" or "how much of a feature do we build?"

For instance, a Product Manager may want to have an internal metrics dashboard. The "best" possible case could be building a dashboard that can do anything and keeps track of everything about the product. That said, it would take a long time to build something like that at full functionality. The "easiest" way might be to have someone analyze the data manually and update a static webpage.

The right answer is usually somewhere between the two extremes. If this is the first version of a product and there are lots of other engineering tasks, the manual version may be the way to go. If you've been running a manual version for a year and it keeps getting more complicated, it might be time to invest in a better dashboard.

These two points end up playing off of each other. The PM needs to figure out how to invest all available time and how to explain to the company why these were the right allotments.

Product Strategy

Steps:

1) Go talk to your prospects:

> Before you get too far down the path of creating a product strategy and building a product roadmap based on your gut, your executive team's thoughts, or even market

research, go out and talk—and more important still, *listen*—to what your users tell you they want.

2) Develop a high-level product vision before mapping out your product strategy.

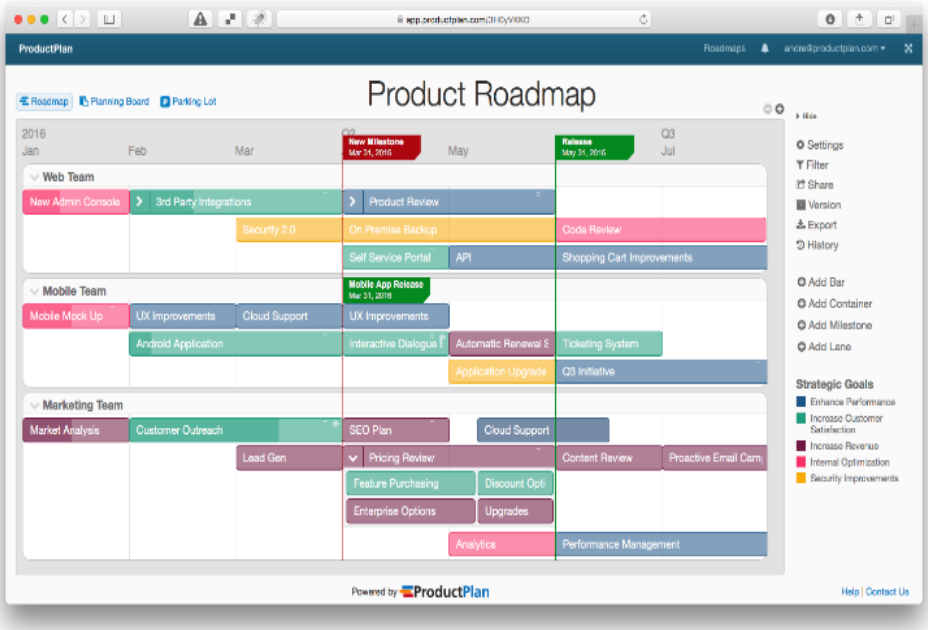
3) Define your product's goals

Goals are measurable, time-bound objectives that have clearly defined success metrics associated with them. They help you set what you want to achieve in the next quarter, year, or 18 months. Here are a few examples: Increase revenue by 30%, Expand into 5 new countries, increase mobile adoption by 100%, reduce the number of support tickets by 15%

"Don't just start throwing feature ideas on the whiteboard. First, define your product's goals."

4) Develop a high-level product vision before mapping out your product strategy.

<u>Roadmapping</u>:



A roadmap represents your product strategy, so it should include the topics you would generally discuss in a product strategy meeting:

- Business goals and objectives
- Product areas
- Order of priorities
- Scope

Prioritization:

Good prioritization requires balancing quick fixes that you expect will have an immediate impact on your customers, versus product evolutions, which will help you lead or disrupt the market.

Prioritization is always a balancing trick between finding high impact quick wins and the kinds of product evolutions that help you create or maintain your competitive lead.

5) Check in with your product's vision to confirm your plan is on track

---

Technical Understanding

The third part of the diagram we started from is the "technical" sphere of Product Management. This is the place where Product Managers tend to struggle the most.

The ones who come from an engineering background struggle not to over-specify how things should be build. The solution is to be as hands-off as possible. If you're an engineer transitioning into Product Management, draw yourself a hard line. The Product Manager defines what is being built; the engineer gets to define how to architect it.

The ones who do not come from engineering struggle with how to get respect from their engineering team and effectively understand technical limitations. This solution is more complex.

Here are three simple ways PMs can start to play into this process:

---

1) Work with the same tools that engineers use. Stay away from dictating a solution that "works for you" instead of one that works for the 10+ engineers you may be cooperating with. If the team is all set up to use bugs in Github issues, make a Github account and figure out how to use Github issues. If you're needed to test early builds before they're on a device, figure out how to get the simulator. Don't be afraid to ask for help.

2) Figure out how to make simple changes yourself. If you find yourself constantly needing to ask for changes that seem simple (e.g., marketing wants to update copy on that landing page again), figure out how to do that yourself. This is particularly effective at smaller companies.

3) Ask good technical questions. The most important piece of building technical knowledge and respect within a company is to ask good questions. If something is hard and you don't understand why, ask. If something you thought would be hard is easy, ask why there too. The best thing you can do is to ask questions that will help you build better intuition for understanding the technical limitations of your product.

Product Specification

Requirements of good specification:

1. the spec must describe the full user experience – not just the product requirements but also the user interaction and visual design. By now hopefully everyone recognizes how closely intertwined the requirements are with the design.

2. The spec must accurately represent the behavior of the software – and we need to acknowledge that words and pretty pictures are just too limited in their ability to describe this behavior.

---

3. There are several critical consumers of the spec – engineering, QA, customer service, marketing, site operations, sales; as such, the spec needs to communicate the behavior of the product in a way that all of these groups get what they need.

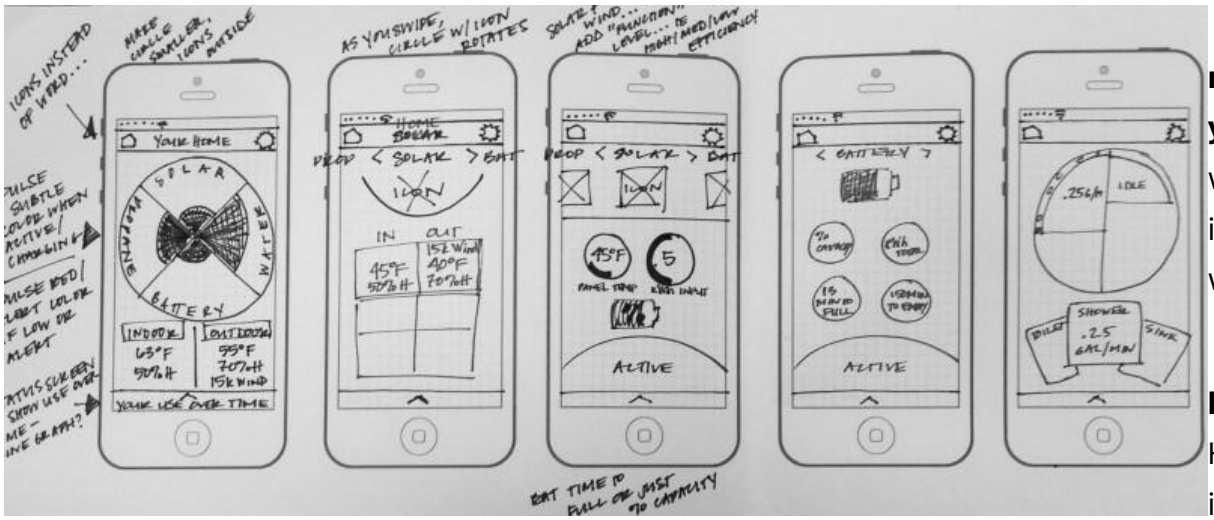
4. The spec will change – the rate of change should slow down dramatically once engineering gets started, but there will be decisions and issues that arise, and the spec should change to reflect the very latest decisions.

5. There are a number of artifacts in the creation of a spec, such as lists of prioritized requirements, wireframes, and mock-ups, but there needs to be a single master representation of the spec, to minimize confusion, ambiguity and version it is.

Wireframing:

Used to convey the following:

- **Structure** – How will the pieces of this site be put together?
- **Content** – What will be displayed on the site?
- **Informational hierarchy** – How is this information organized and displayed?



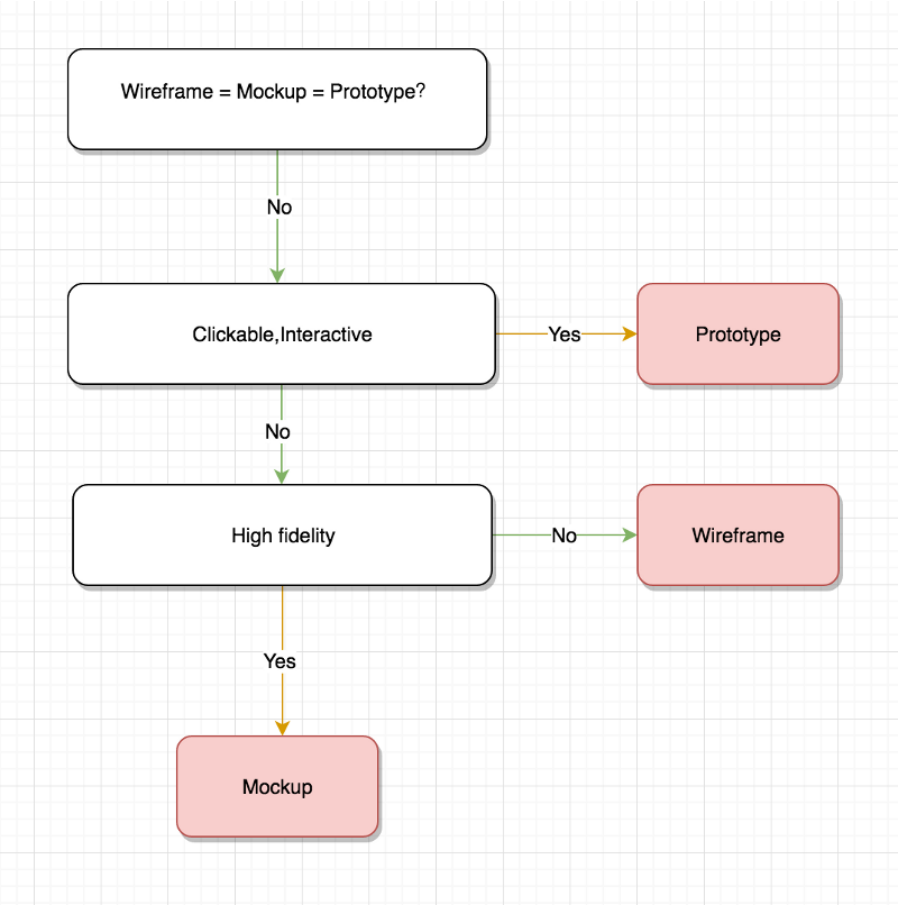
- **Functionality** – How will this interface work?
- **Behavior** – How does it interact with the user? And how does it behave?

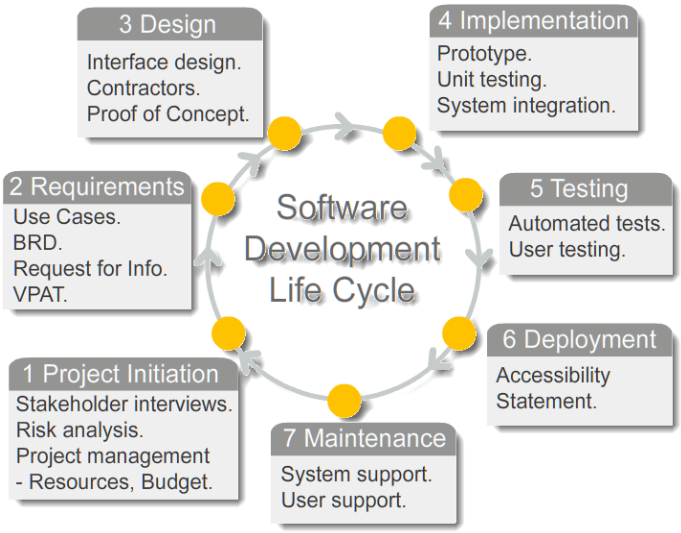
Note: Wireframes are not supposed to represent the visual design, contact graphic elements, or convey the brand or identity.

Users of the wireframe:

1. **Interaction & UX Designers and Information Architects:** use wireframes to show user flows between views or pages.
2. **Graphic Designers:** use wireframes to push the user interface (UI) development process.
3. **Developers:** use wireframes to get a more tangible grasp of the site's functionality.
4. **Business Analysts** use wireframes to visually support the business rules and interaction requirements for a screen.
5. **Internal Business Stakeholders** (ex: Product Managers, Project Managers and Executives): review wireframes to ensure that requirements and objectives are met through the design
6. **External Business Stakeholders** (ex: Partners and Clients): review wireframes to ensure that requirements and objectives are met through the design.

Note: Wireframe vs Mockup vs Prototype





Project Management

Agile Principles:

1. The top priority is to satisfy the customer through early and frequent delivery of valuable software – valuable software early
2. Deliver working software frequently, from a couple of weeks to a couple of months – frequent releases
3. Working software is the primary

measure of progress – software matters more than documents

4. Welcome changing requirements, even late in development – listen and learn rapidly
5. Business people and developers work together daily through the project – intense collaboration
6. Build projects around motivated individuals – give them the environment and support they need and trust them to do their job.
7. The most efficient and effective method of conveying info to and within a development team is through face-to-face conversation
8. The best architectures, requirements and designs emerge from self-organizing teams – agile architectures; good ideas from anywhere
9. Continuous attention to technical excellence and good design enhances agility – refactor frequently
10. Agile processes promote sustainable development – should be able to maintain a constant pace indefinitely – no death marches
11. Simplicity is essential – less is more
12. At regular intervals, the team reflects on how to become more effective, and then adjusts its processes accordingly – post-mortems (Retro!)

Popular Agile Development Processes include Extreme Programming (XP), Crystal, Adaptive, Scrum, and Pragmatic Programming.

PM Concerns:

1. **Unambiguous Specification**

2. **Primitive Notion of "Customer"**
   By working with the product designer (see below) the product manager must ensure that each type of "customer" is represented throughout the product development process.

3. **Scalable, Reliable and Maintainable Product**
   These concerns quickly get into near-religious arguments about the best way to develop and test software, which are largely beyond the scope of the product management role, but the main point relative to the product manager is to ensure the release requirements are clearly defined up front. The engineering organization can then address the concerns of how to manage the risks in the best way they see fit.

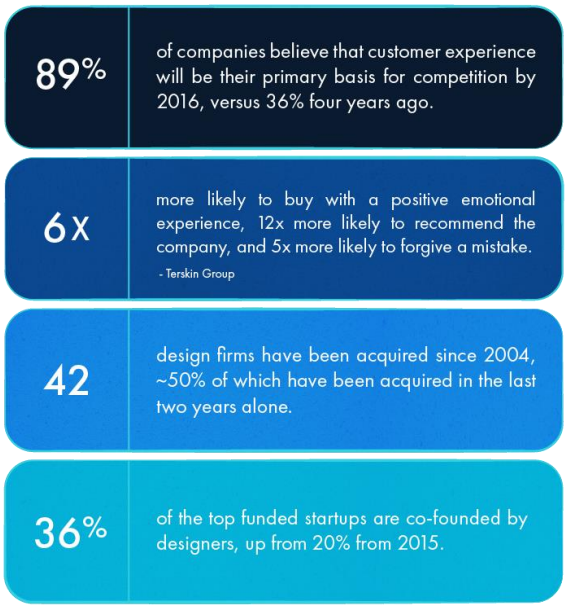
4. **QA**
    - Distinguish between "testing" and "quality assurance."
    - Every member of the team has to be involved in making quality happen.

5. **Project Scheduling**

    One of the challenges of iterative approaches like XP is that while it is much easier to predict the timeframe for a given iteration, it is very hard to predict the number of iterations that will be required to reach a releasable product. If there are time-to market considerations, such as major events where the product is scheduled to debut, then this uncertainty will need to be factored in.

6. **Product Deployment**

Design



Basic Design Principles:

1. Tell users what you think

You should understand your users' workflow so much so that you can determine what they're attempting to do.

2. Minimize cognitive load

Your design should make the user think as little as possible.

3. Three-click Mentality

Every single core action should be achievable within 3 clicks.

4. Reversible Design

Users can get pissed when they can't undo an action.

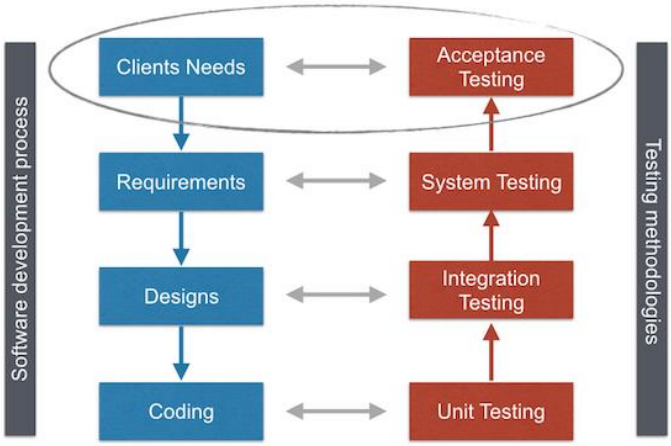
5. Use verbs with context

Narrow down the options and provide context.

6. White space is your friend

Clunky to Slick - eases the emotional experience for users.

QA vs Testing



Distinguish between "testing" and "quality assurance.

- **The worst code requires the most testing:** Having a codebase you can be proud of goes a long way in reducing testing overhead. Invest in making the foundational technologies better. Good technologists also build in lots of automated testing into their code as guard rails to flag issues early and minimize potential bugs.
- **The best technology organizations don't rely on testing to catch issues.** Facebook runs with minimal QA (if at all) because their culture places accountability on the individual engineers to ship bug-free code with complex processes and safe-guards around the scope of any impact (e.g. with phased roll-out to greater and greater groups of users).

- **Bring everyone into the quality process.** Throughout the product development cycle, the entire cross-functional team (up to 30 artists, designers, engineers, and producers) would conduct weekly play tests. Everyone will get into a room for an hour, play the latest version of the game, and recorded all the bugs on a big whiteboard wall. Not only did this guide iterative development, it also gave each team member regular feedback on their own work (e.g. that art asset doesn't look right) and an opportunity to fix issues before the next play test.
- **If you have QA people at all, have them test the product like a user would.** Have a small centralized team of highly-skilled QA generalists that can act as a SWAT team for tactical testing on an as-needed basis rather than always being part of the development cycle. Their job is to try and break the game. However, they play the games as if they were players—without any context of what the requirements are—and pointing out issues as an end0-user would. It's a very effective way to have "fresh eyes" on the product before you ship it.

Deployment

In the spirit of minimizing the disruption caused by change, there are several techniques that can be useful in deploying changes gently.

1. Do everything you can to communicate the changes in advance, through vehicles like newsletters, onsite education, and tutorials. But remember that many people will not have the time or inclination to read what you write, so this technique can only take you so far.

2. If there is any question about the new version of your product working properly, either due to reliability issues, or scale, or performance, then you need to redouble your QA efforts there to try to ensure that you won't have to rollback your updates, which compounds the community angst significantly.

3. If the change is significant, it is important to contain the risk by pursuing some form of gentle deployment – such as parallel, progressive or incremental deployment.

You can do this by deploying a parallel version of your product, and inviting people to "opt-in" and try the new version out when they have the time. Allow those that try to the new version to make it their default if they like it. Once you are confident that the new version is working well, and the majority of your community has converted to using it, you can make it the default and allow customers to "opt-out" and continue to use the old version for a period of time if they don't have the time to learn the changes immediately. Give these people sufficient notice about when support for the old version will be discontinued. For a significant client or service with a large community, expect this process to take on the order of months. Also expect some heat from your engineering and site operations organizations because it is not easy to support parallel versions.

Another gentle deployment approach is to deploy regionally – try the new version out in a limited area of the country or world, and then expand. Or you can deploy the changes incrementally – introduce the changes in bite size pieces over time.

Six Deployment Strategies:

- **Recreate**: Version A is terminated then version B is rolled out.
- **Ramped** (also known as rolling-update or incremental): Version B is slowly rolled out and replacing version A.
- **Blue/Green**: Version B is released alongside version A, then the traffic is switched to version B.
- **Canary**: Version B is released to a subset of users, then proceed to a full rollout.
- **A/B testing**: Version B is released to a subset of users under specific condition.
- **Shadow**: Version B receives real-world traffic alongside version A and doesn't impact the response

---

Launching

Launching is one of the most exciting parts of working on a product. It's the day that everyone finally gets to see what your team has been working on. The Product Manager plays a key role in launching the product.

First is making sure everything that's necessary happens. This varies depending on the company. PMLoop created a 175-item checklist that is a good starting point for you to pull out which items are important for your company. These items range from technical aspects (how do you manage the rollout? Is it to all users at once or just some?) to communications (did someone remember to email the right press?) The PM should be evaluating which items are important and who should be responsible.

Second is making sure that everyone in the company is on the same page. Everyone should be excited about launching. The customer support team will need to be on-hand to support new users. The communications team should be ready to talk to the press. The engineering team will need to be ready to fix bugs. Each of these sets of stakeholders should be feeling excited and involved with the launch process. Since PMs work across these groups, they're best suited to make sure everyone is geared up.

Third is defining what success means for the launch. While launching can feel like a massive success, it's only successful if it actually moves the right metrics. How many users do you want to gain from launch? What percentage are you hoping to retain? What factors might convolute your launch? If you get press in the New York Times on day one, it might impact what you'd previously expected to happen.

While launch day is a big day, the Product Manager knows that launching is just another starting point.

Metrics

The Product Manager is continually looping through the three bubbles we showed at the beginning: UX, business, and technical challenges. We touched before on the beginning of a business—thinking about success and selling your product to obtain fit. But what else do you measure, and when?

There are several schools of thought on key performance indicators. One version is the idea that there is only one metric that matters. In this school of thought, a PM should pick the one metric they want to move and focus in on that. This tends to be a higher-level metric—for instance, Net Promoter Score (NPS). That's what percentage of your customers are enthusiastically recommend your product.

Another school of thought is that the Product Manager should constantly be monitoring whatever is deemed a KPI—a Key Performance Indicator. These are the measurements that show if the product is succeeding or not from a variety of angles. For Highbrow, it might be that you made it to Day 8 of this class and you're still reading! That would mean great retention. But if you're the only user who ever signed up (you aren't), that alone wouldn't indicate success. Highbrow might opt to track both how many users sign up for a course and then how far they get in the course.

At the end of the day, there isn't a huge difference between the one metric vs. small set of metrics schools of thought. Both are ways to make sure that you aren't measuring so many metrics that it's distracting. A PM can easily get swamped thinking about too many things or a metric that's too complicated to discuss within the company. The key is to make sure that the metrics are understandable and useable. These metrics should always tie back to what you decided was important to the business back in Day 4.

Finally, Product Managers use these metrics to do experiments. The right time to do an experiment is when a PM has a hypothesis. If you're changing the homepage copy, a likely hypothesis is that the new copy will lead to more sign-ups. Running the experiment allows you to validate or invalidate that theory. Setting up experiments and looking to learn is more valuable for a PM than combing through random data and trying to decipher value. The latter is more likely to be correlation than causation.

Strategy and Vision

The last piece of the Product Management role spans all three disciplines. The Product Manager maintains and develops an internal vision of the space and product.

This strategy should take into account all of the areas of the product. It should involve a concept of who the customer is and what they need. It should include an understanding of what the business goals are, how to achieve them, and how competitors play in. It should consider technical differentiation and limitations of the product, as well as how to maintain and address them.

The Product Manager does not set the fundamental vision. The founders of the company set the vision at the highest level. The Product Manager is responsible for spending enough time with the founders to internalize their vision. The level of detail provided will vary depending on the founders and their background. It may be that the founders specify what their dream for the company in five years is, and then the PM figures out what products will help the company get there. It might be that the founder has just as specific a vision as the PM does.

If the Product Manager disagrees with the founders about vision, there will be tension within the company. When disagreement occurs, the PM is responsible for understanding why the founders disagree and how their underlying assumptions differ. The PM should start from a point of inquiry;

the worst thing a PM can do is attempt to convince the founders that they are wrong before understanding. Moreover, the Product Manager should never attempt to lead a separate vision within the company.

When the PM does have a vision that aligns with the founder's vision, they're responsible for championing it throughout the company. They write longer-term roadmaps for what needs to be built to align with the vision, in addition to taking into account user and business needs.

They should also be a resource for everyone within the company. The Product Manager should always be well-versed enough to discuss an issue with any employee, even if it requires more research.

Tying it All Together

First, no one tells the Product Manager which of these spaces they should be looking at. Each day, the Product Manager has to decide where to spend their time. Do they need to talk to users to learn? Do they need to see if people are paying? Does everyone internally have the same vision?

Second, none of these pieces are done in isolation. The PM is rarely alone. Instead, they're meeting with users and internal stakeholders to get these pieces accomplished. The PM may not do all the user interviews—they may do some, but then they make time for everyone else in the company to meet users, too.

Both of these skills come with practice. It's hard to learn how to dedicate your time, and who to be talking to, until you've gotten accustomed to the environment. This is why so many Product Managers say it can take up to 90 days to get up to speed in a new job—no matter how much experience you have.

Where you go from here is up to you. There are many books, resources, and classes to learn more about Product Management as a whole and go broader. There are also ways to jump into each skill specifically, diving deeper into one topic area. You could spend time applying these skills in your current working environment. No matter which way you decide to go, best of luck!