How to Foster Design-Driven Development in Your Company

Jess Eddy, January 2014
www.jesseddy.com

What is Design-Driven Development?

Design driven development (DDD) is a process whereby the design of a product, piece of software or app is created to better define requirements, interactions and generally lead the process of development and executing on a better product. Creating a product that solves a problem or that generates revenue is only part of what makes a product successful. The experience the product delivers is the type of value that sets mediocre products apart from great products. Great products are the kind that people want to use all the time and tell their friends about. A design-driven development process is one of the only was to ensure a great experience gets delivered.

Above all else, a good DDD process increases speed, learning and value while at the same time reducing risk.

The Absence of DDD in New Companies

There are many scenarios in which a design-driven development process does not happen. Often times and commonly in startups or new product ideas – design is not accounted for. It's not accounted for maybe because the founders are technical or it's easier to simply build something or there's no money to hire a designer. These scenarios, while understandably unavoidable to a certain extent can cause long-term design legacy problems.

The Absence of DDD in Established Companies

It's not uncommon to see design unaccounted for in large, established companies. While this might sound crazy, this can stem from leaders within the company not understanding the value that design brings to a product, thereby not including design in the development process. This creates a much more dangerous problem since the length of time design goes unaccounted for is much greater. This not only creates design legacy but product debt. The type of debt that eventually will swallow a product whole and inhibit success.

Design Legacy & Product Debt

We talked about how design legacy and product debt is created but what do these things mean exactly? Most of the time we hear the term "legacy" when referring to code in software. Legacy code is code that is either bad or outdated but so embedded in the product that it's too complicated or expensive to replace. Design legacy is very similar. The absence of a DDD process means that many bad experiences have been implemented in a product. Engineers or product managers have likely been put in a position to also design the software. At this stage, it's not a matter of design anymore; it's a deeply embedded product problem. It's a problem so big and a product experience so bad that the product is rendered unusable or unsellable – but the legacy created is so large that it creates debt. Product debt costs a lot of money to fix and usually requires that a product be rebuilt and redesigned from the ground up; this is not an option for many companies.

Concrete Examples

Here are some concrete examples that create design legacy problems in startups and companies.

- Not hiring a designer at all or hiring one in piecemeal
- Hiring a designer, but not utilizing them correctly
- Thinking of design as something applied after the fact, instead of an important part of the overall process
- Not allowing design and user research to answer product questions
- Implementing design late in the product development process
- Contracting design: hiring many different designers *(creating an inconsistent and incoherent product)*
- Not hiring the right designer: one that can drive the process that helps make product decisions

What is the Solution?

Up to this point, I've talked about the problem – because like design, we must understand the problem to create a good solution. The goal, reiterated: to create a design-driven development process, which will allow us to create the best product possible. The first step: *creating a balanced team.*

Balanced Teams

A balanced team is a multi-skilled, collaborative team with a shared vision and the flexibility make decisions and do whatever they believe is best to make the best product possible.

The "collaborative" part of this is very important and where many companies fail. Collaborative in this sense means: each team member is involved in product conversations and decisions; each team member impacts the outcome and is part of the process.

Think of it like you're all co-pilots. You're flying a plane together and the plane can't fly unless you're all sitting in the cockpit.

What a Balanced Team Looks Like

A balanced team (in the context of software products) is typically:

- The product owner/stakeholder
- An engineer (or tech lead)
- A UX designer / designer

**The size of the balanced team depends on the overall size of the project and company and can include more than one type of team member and more types of members.*

What Balanced Teams Do

- Communicate and define the shared vision
- Create a product plan around the shared vision
- Make unified product decisions
- Understand the constraints that impact product decisions

What Balanced Teams Do Not

- Make independent product decisions
- Design and develop features that are too complicated for the given timeline or other constraints
- Have product conversations in silo
- Work on product with no goals or milestones

A goal without a plan is just a wish.

-

- Antoine de Saint-Exupery

Examples of Balanced Team Activities

- Engineers *(or a tech lead)* take part in design reviews
- Designers sketch out solutions to problems with engineer(s) and the product owner
- The team walks through the technical feasibility and viability of features to understand the time impact *(feature scoping)*
- The team plans the product roadmap and decides what type of sprint cycles make sense
- The entire team takes part if customer development/feedback conversations

Capturing & Implementing User Feedback

Another important element of design-driven development is capturing and implementing user feedback via user testing. This is an activity that everyone should embrace and learn how to do. It's an important part of the DDD process because user feedback informs design decisions, which positively impact what we make. Most of the time, it's very easy to get feedback from users, but that's not to say that every product company employs or supports this activity. Some sectors, such as finance, frown upon it due to the lack of transparency in that particular culture and the sheer amount of ego that drives many product decisions. The one thing I've consistently learned about user feedback while working on products is, it's never too early to do it and most of the time it can be done more often.

Tips for Capturing User Feedback

- If time and money is an roadblock, test with people via email or use free tools such as Wufoo to create research forms or test with people via Twitter, Facebook or other open platforms
- To get targeted user feedback, go to where your users are; if you're making a pet focused app – go to Petco
- When decisions can't be made in product meetings it usually means it's time to ask a user
- Do user testing as much as needed until validation is reached
- If your company does not support user testing as an activity, do it in secrecy and bask in the glory when real people love the end product *(do this one at your own risk)*

Product Plan & Project Tracking

These two things are linked and the rule of thumb is: the bigger the team, the more important this activity is and the more time it takes. A basic product plan should include an end date, an understanding of the high-level goals, desired milestones and some idea of what the end product should achieve. For teams that are working on a new idea it's important not to get too obsessed with planning because so much changes so frequently in this type of scenario that planning can actually have a negative impact on progress. The planning that takes place in this type of project has a much shorter view into the future. For example you might be able to plan out the first week but then realize that after talking to a handful of users that your product approach changes completely.

For projects that are more formalized or defined, a little planning can go a long way and a more unified team will produce a better product every time. This type of planning requires a little more time and dedication to setup and manage. It's an up front time investment for a long-term gain in efficiency and overall time savings

Communicate!

One thing your team should be doing not matter what the size, is communicating – A LOT. This does not mean – talk about the product all day. This does mean, communicate what's going on with certain aspects of the project on a daily basis. This is partly an informal way to track a project but the true value is in the face-to-face communication and the transparency it creates.

Thank you!

I hope you enjoyed this high-level overview of how to foster design-driven development in your company. In summary, here are the main tenets I outlined in order to do this:

- Create balanced teams
- Capture and implement user feedback
- Product plan and project tracking
- Communication

Please feel free to email me with any questions!

jess@jesseddy.com