Juggling Priorities: Defects, Tech Debt, Features, and Escalations

Description

In the world of software development, teams often find themselves facing a multitude of demands and pressures, each fighting for attention:

“If we don’t fix this bug, customers are going to leave!”

“If we don’t address the tech debt, our velocity is going to plummet!”

“Without this new feature, we’re not going to get any more new customers!”

“This customer’s unique configuration is causing this feature to not behave as expected!”

Every complaint and concern is valid, and from the perspective of the person raising it, it’s urgent. But how can an engineering organization possibly decide what to tackle first?

Product Panic

It’s all too easy to succumb to product panic. You see a competitor roll out a new feature, and suddenly, it feels like implementing it is a matter of life or death for your product. Or perhaps a customer requests a feature, and you rush to fulfill that request in fear of losing their business.

The problem in both these scenarios lies in the knee-jerk panic reaction. These situations often lack a broader context and can result in rushed decisions that sacrifice quality or lead to wasted effort.

The Defect Trap

Engineers take pride in delivering quality work, and every defect found can feel like a thorn in our side. The temptation to fix every defect immediately is strong. After all, we want to produce a flawless product, right?

But here’s the catch: while you might have a polished piece of work, if it’s offering features that are

three years out of date, are you truly delivering value?

The Triad of Prioritization

To navigate this prioritization maze effectively, every development team should contain a triad of leadership roles, each representing a distinct perspective:

- The Scrum Lead, focused on velocity and therefore aware of the need to address tech debt.
- The Product representative, the voice of the customer, advocating for new features or adjustments.
- The Tech Lead, responsible for the code's quality, performance, and freedom from defects.

Crucially, these three voices must collaborate, listening to each other's concerns and advocating for their perspectives.

Now, let's revisit those earlier complaints and view them through the lens of the Triad:

- **"If we don't fix this bug, customers are going to leave!"**
  - Is this statement data-driven or panic-driven? Analyze the impact. If it affects only a small portion of customers, consider it as part of normal workflow.

- **"If we don't address the tech debt, our velocity is going to plummet!"**
  - When technical debt starts piling up, it's a red flag, signaling the need for action. To help stakeholders grasp the impact of technical debt, consider implementing a focused "two-perspective pointing session." Here's how it works:
    1. **Identify High-Impact Tasks:**
       - Begin by identifying the tasks you believe will require more effort due to existing technical debt.
    2. **Double-Pointing:**
       - For these tasks, assign two sets of points: one based on the anticipated effort under current conditions and another based on the effort required if the technical debt were already addressed.
    3. **Calculate the Difference:**
       - Sum up the total points from both approaches and calculate the difference. For example, if you have 100 points of work, but addressing technical debt would reduce it to 65 points, you're left with 35 points to allocate to other tasks.
    4. **Assess Impact:**
       - Evaluate the points associated with relevant technical debt tasks that impact the selected tickets.

    In this scenario, if the total points related to technical debt are less than the available 35 points, prioritize addressing that technical debt as a prerequisite for other deliverables. Doing so can expedite meeting customer needs and prevent potential financial losses for the company.

    However, if the total technical debt exceeds 35 points, focus on high-impact technical debt items. For example, you might find 15 points of technical debt tasks that, when resolved,

reduce the overall effort by 20 points instead of 35. This still represents a profitable investment in prioritization.

Another opportune time to tackle technical debt is at the end of a sprint. After delivering on committed sprint goals and meeting business expectations, teams can dedicate time to addressing defects and technical debt. This approach maintains momentum while ensuring the long-term health of your codebase.

- **“Without this new feature, we’re not going to get any more new customers!”**
  - As always, where’s the data? How many customers have actually asked for this feature? A customer asking for a feature is extremely different than approaching a customer and saying, “want this?” and they reply, “yes”. Were you expecting a “no”?

    If a notable portion of your customers have raised a concern about a feature missing from the platform, then give it some serious consideration. For every vocal customer, there are many staying silent with the same concern.

    What is the market showing? Is there a trend among competitors to include this feature? Is the feature an appropriate fit for the niche your product is trying to fill? If so, definitely bump it up in priority.

    What are departing customers telling you? Have they said they’re leaving because this is missing? That’s definitely a sign to take it seriously.

    If all the above are true, this is an urgent new feature to get to work on as soon as you can.

- **“This customer’s unique configuration is causing this feature to not behave as expected!”**
  - There are not a lot of scenarios where it makes sense to prioritize this above work that is affecting other customers. There are some things that can bump this up in priority though.
    - Is this a high value client? How many other customers could you afford to lose while catering to this customer’s unique needs?
    - Is this a canary in the coal mine? Could they simply be the first to discover this issue and it stands to reason that it will impact others soon?
    - Is the effort needed to fix their issue small enough that it won’t impact other deliverables?
    - Will addressing any other roadmap items or logged defects address this? Consider using this new variable to bump them up in priority.

The Authority

Don’t hesitate to engage customers directly in prioritization. Ask for their input on key roadmap items, including defect reduction and performance improvements. Building relationships with customers and showing them their feedback matters can lead to stronger product development.

In conclusion, effective prioritization isn’t about choosing defects, tech debt, features, or escalations over each other. It’s about determining what work will benefit your current and future customers, and your business, the most. Data-backed decision-making is your compass in navigating this complex

landscape.

Remember, the Triad’s collaboration and customer engagement are your keys to success in prioritizing technical work.

Category

1. Project Management

Tags

1. defects
2. features
3. priorities
4. product
5. tech debt
6. triad

Date Created
September 15, 2023
Author
steveday