Project Retrospective

Using 3 L's Format

Alan Tobin

2023-05-11

Project: Trustless BTC↔DAI trading solution.

Reason for early termination:

Little buy-in from users on DAI-BTC trading pair, which reduced buy-in from the team.

While the project can't be classed as successful due to the lack of a product being actually used, it brings a number of learnings that will be key to the success of other projects.

Loved

Running software on Mainnet

- It makes you think differently, focusing on critical things.

- Some issues in regtest that we overlooked are actually important in production environment.

Showing something concrete to users

- **User interaction**.

- Getting **feedback** from outside of our bubble
  on what is actually important to users.

Code Ownership & Teamwork

- Really liked **working together**. We know each other's strengths.

- Lots of **trust in the team** allows smooth, efficient running.

- **Clear priorities, small team** allows little communication overhead &
  greater autonomy.

Learned

Code Base

- **Tech debt** we already knew, but realized how important it was
  when started using mainnet.

- Thoughtful **error handling**:
  We can't stop errors but we can provide tools for that.

- Don't abstract too early.

Use case

- **Find a usecase & users first** &
  then design protocol that has properties of the use case,
  not the other way around.

Users

- We found users thanks to our **existing network**.
  People are actually **keen to give feedback**.

- Our **social media game is weak**.

- We question whether we are trying to reach people who are too technical.

Lacked

- Experience dealing with users.
- Knowing what it means to build a product, uncertainty about what happens next.
- Confidence in own skills: finding users, product life cycle.
- Decision making strategies for overcoming the fallacy of "we already built something, now we have to use it".
- Deep domain knowledge (market, finance, trading).
- Strong belief in the product

Code ownership

- Team did not fully own the code
- We were limiting ourselves using a legacy codebase.

Most important learning - ideal formula for how to develop product in our domain

1. Come up with a use case.
2. Explore the domain,
   talk to advisers & experts to challenge the use case.
3. Find or create a **protocol to fit the use case**,
   build PoC, run it on mainnet.
4. Create mocks that work within protocol's limits,
   show it to users, validate that it still fits the use case.
5. Create MVP, get users to use it.
6. Make it a product. Profit! ☺ ☺ ☺