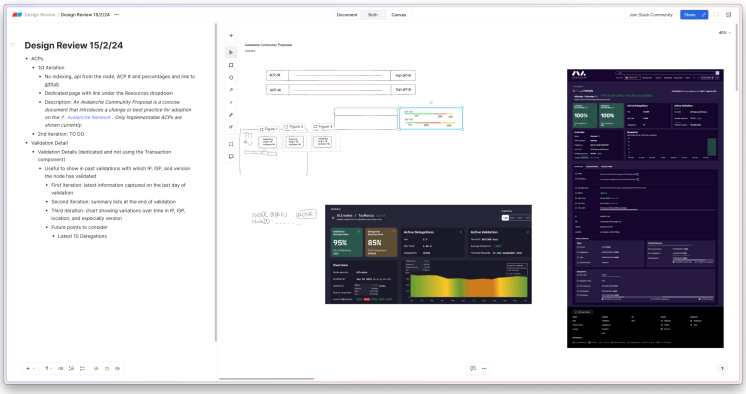
The Importance of Atomic Development



> *Atomic design is a methodology for creating design systems, which involves five distinct levels: atoms, molecules, organisms, templates, and pages. It allows for the construction of robust and scalable interface design systems by breaking down interfaces into smaller components that work together effectively.*

Perplexity

I can't stress enough the importance of making *atomic* changes to established products. That's the single learning I bring with me after 3 years of working on Avascan, a public good for the Avalanche community.

At first, we didn't really have a process: in the first year of R&D, right after we launched, we didn't have a way of tracking different projects other than the very limited milestones tracker offered by Github. I would spend hours every day talking to our designer and trying to be very specific about what I thought should be displayed on each page. That was the period during which we didn't ship a lot of features, just a couple in over 6 months. It was max pain, sacrificing so much in brainstorming new ideas to bring data to use, yet with nothing to show for a long time.

Then, in 2022 the team grew 3x in size, and we took in a Product Manager that was in charge of establishing a very rigid and solid process of product development, mostly through Jira. Stand-up calls every day, 2-weeks sprints and twice-a-week release cycles. Avascan started shipping new things and solidifying the frontend codebase, which changed from vanilla JS to React, using a new and more compact design system, and I moved to other things, trying to find a business model. Then, in 2023 we worked on Routescan as a side project, that quickly grew to be the **third block explorer provider in the EVM industry**, after Etherscan and Blockscout, with still much more room to grow.

The thing is, Routescan has a very clear goal: stay on track with the Etherscan feature set, take inspiration from the UX as much as possible, and make it natively multi-chain in every way possible. This is how we can realistically overtake the industry, and it took me a while to understand it: looking at a competitor's product and adding a Unique Value Proposition that changes design and feature set in a way that's distinguishable and unique. In fact, Routescan and Etherscan are very different in a lot of ways, and even in the feature set now (some things don't make sense for single-chain explorers, while they do make sense for multi-chain ones, and viceversa).

At the same time, Avascan took a great part of 2023 to build features and pages that were long awaited, and that were released just a few days ago: new Staking features, with historical data that many explorers do not have. But the 'issue' with Avascan is that, since it's an Avalanche public good funded (and now paid) by the Foundation, most of the data that we have and can show don't have a history in other products.

Or at least, not a good history.

That's where we can innovate

That's where we can innovate.
That's where we can experiment.

That's where we can push the industry forward.
By thinking about new things, and in new ways, because Routescan is now a business, and Avascan is a public good first, so we can *afford it*.
That's why, starting this month, we're changing the approach with Avascan:

- **Shorter** release cycles (as much as once per day)
- **Shorter** sprints (not more than one week)
- **Faster** prototyping

The last one is a deal-breaker: with faster prototyping, we can focus on one or two mini-projects at a time, and iterating on each one very quickly and **atomically**.
This is the biggest challenge, for me as well: being able to choose the single, simplest and easiest thing to do that can have the highest impact on the product, and work on that alone. Then, starting from that to build on new, extremely small incremental changes that will make the difference in the long run.
After all, *we don't have the slightest idea to know that the path we're choosing is the best*, so we can't take anything for granted. We must put one feet in front of the other and walk slowly, and ask everyone around us if we're doing good.
And we're not doing this out of blue: this year started in a bull market, and everyone's crazy about anything crypto. Trends come and go in a week, and we can't possibly know what's gonna stick beforehand. We do know that some things may have a longer trend cycle than others, but we're talking about months instead of weeks. So the biggest reason we're doing this is because we're somewhat **forced by the market**. We can afford to stay on top of the market and do something that, in one year, can be great, but only if we iterate quickly, as quickly as the market changes idea about anything. Who remembers ERC-404 for example? The narrative has now been shifted to DN404, and both are not even standards! They're just experiments, and who knows if we're still going to talk about them 2 months from now.
There are some major trends, of course, but even those morph quickly, so we need to be careful in modelling after those.
I'm calling this approach **Atomic Development**. I thought I could borrow the term *Atomic Design*, but that's different: for the design, the focus is on breaking down a system into smaller components, until we find the smallest one (atoms) that can be used to compose everything that we need. But for Atomic Development, the focus is on the choice of time vs. impact: **what's the single, easiest change we can make that has the most impact on the product overall?**
This approach has two pros and two cons, the way I see it.
Pros

- **Build every week on the momentum of the week before**: since it's a very easy change, it's going to be released **for sure** and developers will have a boost of endorphins that will get them more involved over time
- **Build with extreme focus, at extreme quality:** being able to only work on one-two projects at a time, and not seeing tens of tasks in a list, will get developers focus on the project at hand and (hopefully) deliver a better product that's more tested.

Cons

- **It's not good for teams that are finding their MVP**, since it's a very cautious approach that will slow down the team
- **It's not good for teams that deal with a lot of projects**, since it builds on the focus you can put in very few things.

Is this the ultimate development workflow?
I honestly don't know. But we'll find out.

Date: 2024-02-15
Words: 1129
Time to read: 5 mins

Newer Older

27th February 2024

Fully onchain ticketing with ZK p...

29th January 2024

Some Etherscan numbers

Tags Archive RSS feed Twitter Instagram GitHub Email QR Code