I was the Blockchain Architect for a dozen projects, and this is what I learnt.

It’s been a hell of a ride.

Alberto Cuesta Cañada

Follow

HackerNoon.com

--

Listen

Share

Success is not final, failure is not fatal: it is the courage to continue that counts.- Winston Churchill

In the year since I started as the Chief Architect forTechHQ I’ve designed at least a dozen blockchain applications.

My first task when I was hired was to write an article abouthow to get started on blockchain as an architect. To date it still is my most popular article but now it is time to write a second part to it. That first article condensed 80 hours of experience, this one more than a thousand.

Please keep on reading to find out the skills that I’ve found most useful as I translated business ideas into technology solutions, in projects from concept stage to implementation.

What I actually do as a Blockchain Architect

There are many different types ofsoftware architecture roles, but I describe my role to others as:

“I’m a bridge between business and technology. I talk to the business side to understand what it is what they need, and to the technology side to know what is possible. Then I come up with proposals to deliver to the business with the technology available and iterate until we find an acceptable solution. During the implementation I refine the idea as challenges appear.”

Some people think that the defining characteristic of an architect is that they know a lot about technology and has some leadership role. Knowing a lot about technology or about business always helps, but what an architect actually does is to understand

and communicate

. My solution design process goes like this:

1. Learn from experts on both sides.
2. Explain what you learn to the stakeholders on the other side.
3. Propose a solution.
4. Go back to 1.

As an architect for TechHQ, I happen to know a lot about blockchain, but for every project I have to learn several new concepts. This month I’ve learnt how the UNDP delivers services to the poor and how to do atomic swaps, for example.

Knowing a lot about technology or about business always helps, but what an architect actually does is to

understand

and

communicate

.

This constant learning and proposing process also means being wrong most of the time

. Only after being wrong a number of times you will have learnt enough to propose a solution that actually fits.

If you want to succeed as an architect, you must focus on being able to learn on a broad range of subjects, understand anyone and make yourself understood by anyone.

What did I learn for the past year

During this past year I learnt a lot about blockchain, but I also learnt how to be a better architect in any new field. If at some point I would decide to become a machine learning architect I would follow the same path. My focus would be five-fold:

- Get a steady stream of challenges.
- Surround yourself with experts in the technology.
- Acquire hands-on knowledge of the technology that is closest to the business.
- Abstract high level patterns from each project.
- Write about everything that you learn.

Get a Steady Stream of Challenges

This might not be your luck, but it was mine. My partners at TechHQ are very busy in the conference and corporate circles and that means a constant flow of projects for me to flesh out. From the far-out complex “these guys want to build their own blockchain platform and consensus algorithms for artificial intelligence modelling” to the clear-cut “these guys want to build an online food market with their own cryptocurrency”.

Many of these projects never went past the concept stage, but all of them were useful for me. I’m quite a productivity nerd and always record the time for each task I do. What took me 24 hours of work with the first project I could do in 4 hours a few months later.

The materials I had to produce were mostly text with a few diagrams and sometimes a code example. The text ranged from academic solution descriptions to marketing manifestos. To say that sometimes I was out of my league is an understatement, but anyway everyone is out of their league most of the time. Even Vitalik. Talented people don’t stay in their comfort zone.

This broad focus was the best thing that happened to my training.

- I learnt the business use cases that make sense in blockchain vs. the ones that don’t.
- I learnt where blockchain technology is mature enough to work and were isn’t.
- I learnt to recognize patterns and use them to propose solutions quickly.
- I learnt when to be persuasive in writing and when to be precise.
- I met many people that I can go to with questions for everything I didn’t learn.

If you are lucky and have a broad stream of challenges coming your way you should just embrace it. If your job is to be responsible for a single product, you will need to make an extra effort to bring in additional sources of knowledge.

Surround yourself with experts in technology

When I joined TechHQ I realised that the blockchain environment was so immature that I could get to the top of the learning curve within a few months. While that is still true, I don’t find it possible anymore to know everything

about blockchain. In any project someone needs to know about these things:

Blockchain infrastructure:

I started by learning a lot about the blockchain infrastructure. Bitcoin, Ethereum, Hyperledger, Quorum, Corda and many other projects are in this space. However, for most of the last year I’ve worked on Ethereum or Quorum, with the occasional Hyperledger project. Right now I think that it is enough to have a passing understanding of other platforms and having people to talk to at the infrastructure level. A lot will happen here in the next few years, so you need to be ready for an eventual major development.

Blockchain Development Tools:

Here I’ve learnt how to work at a basic level with metamask, ganache and truffle, and that is about it.You need much more than that in a project and I’m grateful that I have [Bernardo Vieira

](https://www.linkedin.com/in/obernardovieira/) that seems to know how everything works and never stops delivering cool stuff (solviz, soldoc). If you are going to code you need to fill this gap somehow, and it is a difficult one due to how fast the tools change.

Smart Contract Development:

I learnt to code smart contracts and that allowed me to gain a deeper understanding of blockchain opportunities and development patterns. Software development is closer to the business side than development tools or infrastructure, so if you have to choose which one to learn more of, I'd focus on this one.

Other technology fields:

Cloud Architecture, Networking, DevOps, Testing, Frontend Development, User Experience, Graphic Design. Sometimes you will need some help here, but most of the time you'll be better off letting someone else do the very necessary work. Focus on where you add value.

Other project roles:

If you suck (as I do) at project managing or team leadership, make there is someone else to do those roles. Startups are always strapped of resources and you will find yourself pulled towards these roles by their proximity to the architect role. If I have to help with a non-technology role I will always choose that of business analyst, since it allows me to understand the business better.

Acquire hands-on knowledge

Bernardo asked me six months ago to give him a hand withsome math that was worrying him, and gracefully spent a few hours getting me a blockchain development environment going. He would spend a lot more hours fixing my mistakes while I learnt to use git in a professional environment. I ended furiously coding a fixed point math library for solidity, while I learnt solidity.

A peculiarity of smart contract development is that it forces you to produce simple code, at least in Ethereum. In most blockchain solutions the smart contracts will be less than 10% of the code. What I've found is this makes coding smart contracts easier than common software because once you code a small core the rest is offloaded to the frontend.

This is great for me as an architect, because I can codecompact proofs of conceptthat implement the business ideas of a solution and that stay mostly unchanged in production. Coding is still very time-consuming, so I only do it when I have the time and when the project needs it. As an architect your hourly rate is quite high and you need to be mindful to provide value for it. However, coding is sometimes the best way to show what you mean to the technology side.

Smart contract development led me to understandstaking patterns, tokenization, currency exchanges, payments distribution and access control with deep detail, and those patterns come up again and again when trying to propose solutions to business ideas.

Abstract high-level patterns from everything

I didn't think much of design patterns when I did solution architecture for my last employer. Most of my time was spent negotiating how to get data from A to B against a million conflicting interests. We were instructed to produce patterns but there was little to reuse from a project to the next.

When creating new products is when patterns become useful. "Your company with smart meters wants to do something with blockchain? IoT data collection, a data registry, a payment flow and a control module would allow you to open and close the meters according to payments, we did bits of this for X, Y and Z". "You want a crypto powered food market? You need your own cryptocurrency but then you have options for decentralized and centralized markets, let's have a look".

Breaking down solutions into general patterns allows you to think quickly about what the technology opportunities are for a specific use case and guide your discussions with the business owners. This is very important for the architect, you need to explain to the business side what your proposal is in terms that they understand, and breaking it down is helpful for everyone.

Write about everything that you learn

I used to read in architect role descriptions I needed to show thought leadership. No idea what that actually meant. I also had always wanted to write stuff, but didn't have much to say. It's quite amazing how these two things came together to become a pillar of my job.

As an architect you are perceived as a leader, but this is not true at the start of a project. For the first few iterations you are learning from the business and technology sides so that you can propose solutions. You will need to be a leader when those iterations reach an end without satisfying everyone and you need to broker a compromise between stakeholders..

By writing frequently not only you refine your own thinking and arguments, but those articles also work as some proof that you know what you are talking about. You don't want people to follow you blindly, and you definitely don't want to use the argument that you know your stuff and others should shut up, but first impressions are important, and often what you write are those first impressions.

The other benefit that learning to write has on your job as an architect, is that all my projects are delivered in a written form. There might be code and there are lots and lots of talking, but at the end, there is always some text that is taken to far places. It makes it into the whitepaper, it gets distilled into two-pagers and pitches. I write a lot now, and it becomes easier and easier. It is definitely one of my main skills today.

Conclusion

As an architect for an up and coming software development firm I've designed dozens of blockchain solutions with different degrees of detail and success. It took me 80 hours to feel that I had an idea of blockchain and to start designing solutions. For the thousand next hours the learning pace only accelerated.

If you want my advice on how to become a better blockchain architect, I'd suggest you do the following:

- Make sure you have a steady stream of projects

coming in. Each project is a learning opportunity. Paying clients, open source collaborations and your own projects are all acceptable options. Try not to get stuck in 3 projects a year as it happened to me before.

- Surround yourself with people that are more capable than you

. If you are part of a team, or of many teams, you will be able to specialize on the architecture and to get a broader view of the world.

- If you want to get nerdy, choose smart contract development

over other technical skills. All skillsets are important, but for blockchain the smart contracts are the closest technological component to the business logic, and the most likely to be useful in closing the gap between business plans and technology opportunities.

- Try to make sense of things. I tried my hand at developing aTokenomics Framework. I tried my hand at creating Design Patterns. Every day, try to find new ways of classifying information

and the things that you do. A high level vision is key.

- Write what you learn

, whether anyone reads it or not. You need to be able to express yourself clearly in many different registries, depending on who your audience is. You need to feel comfortable having many pages of text to write.

It's been a hell of a ride. Thanks for reading all this, and please feel free to comment and continue the conversation. I'm always happy to help.