Rafael Castro Gonçalves Silva

Cover Letter

I'm a working computer scientist and a developer.

Prelude

This is a text about myself, so you can know a little bit about me. I did this text in an informal style of writing, so you can imagine me saying what is written here. Also, notice this is being consistently improved and updated

How I'm like

Usually I describe myself as nerd, progressist and cachorrista (a brazilian neologism for dog person). Let me be more detailed about those things:
I enjoy nerd stuff like video games, fantasy movies and technology in general. I know, very typical. Second, by progressist I mean someone that agrees with progressism: the idea that we should embrace changes that can cause improvements in our life quality (e.g., weed legalization). And last: I like dogs a lot.

Of course, I'm not just three things.

Another thing that I'm is playful, and I'm far from the stereotype of a serious adult. Sure I can have serious conversations, but I just find life better to live if I can joke every now and then. Though I enjoy making jokes, I don't like it when it offends someone that did nothing to deserve it. I hate when I see someone being humiliated if the person don't really deserve that. I can't watch *The Office* because I fell like throwing up when I see Michael humiliating his employees that have done nothing to deserve it. Search for *The Office — Try My Cookie Cookie* on YouTube and you will know what I'm talking about.

More on me:

I usually avoid things that cause me stress. If I have the option to spend money so I won't be bothered, I do it. Sometimes I just quit the thing that is causing me pain or anguish. I don't mean that I'm a quitter, I just question myself: do I need this? And depending on the answer I do the benevolent thing to myself. For example, my first job was causing me severe stress because of too much pressure, unrealistic deadlines, toxic culture, so I quitted after just two months working there. It was there right call, so I could move on to the next thing.

I'm eager to learn and learning is quite important to me. This characteristic of mine made me want to avoid to work in the software industry. My vision was partially correct I would say: to a company the best tools are the ones that you can easily find employees to hire for. But, usually, those tools are not really the best ones for the job and are the ones that I don't want to learn.

I learned Ruby for my job, but don't think it was an interesting thing to learn. However, I have learned very interesting things in my current job. So I don't see working in the industry with the same pessimist eyes as before.

But my opinion, in general, stays the same: developers should be eager to learn the best technology for this or that specific problem. I think the problem lives on the fact that programmers most of the time are stuck with the same languages and tools, they should more often look out there and see that languages aren't all the same, and they didn't stop innovating in the 90s. In conclusion about that, I rather work in a place where I'm continuously learning interesting things. There are some very interesting new languages like Rust that I wish to see in the industry more often. However, I understand that you don't see a Rust programmer under every rock like JavaScript programmers.

I'm usually very communicative and I don't fill ashamed of asking people to teach me about something. I don't pretend to know something that I don't. Not only that, but I enjoy working with teams that do *real team work* with members that support each other.

My life in a nutshell

I was born in 1993 in the city of São Paulo (Brazil), but when I was three years old I moved to the State of Santa Catarina (SC), in the south of Brazil and I have been living here since then. I lived in some cities of SC, but at the moment I live in Joinville. In 2012, I started my undergrad in Computer Science, here in Joinville, at the Santa Catarina State University. For the first time in my life I noticed that I was doing something that I enjoyed. I loved the courses of Automata Theory, Compilers, Formal Methods and Computability Theory. I guess I discovered myself in theoretical computer science. After my undergrad, I started my masters at this same university.

Academic me

At the end of my undergrad I discovered that programs can be certified in respect to a specification using proofs. I already knew Curry-Howard correspondence, but when I found proof assistants like Coq and Isabelle, and languages with dependend types like Agda and Idris, I was very surprised to see what people were doing with it. For example, I found out CompCert, a C compiler with a fully verified backend. Come on, this is the coolest piece of software ever made! I started the masters and my advisor had experience with type systems for programming languages, specifically type inference algorithms. So I decided that my masters would be using a proof assistant and would have something related to type inference.

Long story short: I did a Coq certification of algorithm W using Hoare logic using a monad. Google *Monadic W in Coq* and you will find the paper.

After the finishing my masters, I decided to look for a PhD, but things didn't go so far in this matter. In addition, a pandemic started so this plan is delayed.

Professional me

I had just finished my masters, I couldn't start the PhD and I had no professional experience. I decided that the least I could do is to solve the latter. Looking for a job was quite sad because I don't really see many jobs opportunities that I find interesting. I managed to find a job that looked quite convenient for logistic reasons and I decided to take it. As I said, I didn't like it in the first few weeks. The problem were so many, just to get you a glance: it was a startup following this stupid trend of startups that think they are a family. More on that later.

So I quitted that job and started looking for another one. I found a place that some colleagues from the university were working at and they said this was a good place to work. The company is an outsourcing one, they provide their service as teams of programmers. So, I applied and I was hired. This company is called Magrathea. Yes, the planet that makes planets.

I was allocated to a team working in project that is quite interesting. Being brief: it's a distributed system with concurrency scenarios that must process events in whatever order, so it can keep a materialized view correctly updated. At the moment I'm writing this text I'm still working in this project and the experience so far is great. I find it very satisfactory to work with a team of people that have a higher level of abstract thinking and can write non-trivial algorithms. Working with them made me discover how it's like to work with a team that help each other in order to kill a dragon. Now I can say that I can be a good professional.

I and my teammates joke that after this project we don't know what to do because everything else looks quite boring. So, if you are an employer reading this I think you should understand the kind of developer I'm: I don't like trivial problems, I need to be constantly challenged, I like mathematics, I don't like to see technology as a mystical thing as many frameworks make it look like.

I know I don't have much experience as a developer, but now I at least see the possibility of an interesting professional life for me. The experience that I had in my undergrad and masters was amazing, and I could say that this kind of experience also matters in the industry. The abstract thinking that I developed in the academy came in hand in many situations so far. Though I still wish to do a PhD, I don't see this as an issue for my professional life. If you do, you missed the chance of stop reading this text long ago.

The "To whom it may concern" section

If your received this cover letter, this section concerns you.

Some of the most obvious things that job interviewers ask are: "why do you want to leave your current job?", or "why are you looking for a new job?", and variations. Those are very relevant questions that I hope to better answer here than I could possibly to in an interview.

I have a few reasons. And those reasons are true when I'm writing this (May 2021), but I don't know for how long they will be true. However, before I go through them let me explain that I'm not unhappy with current job and I have seen people looking at me with suspicious eyes since I'm looking for a new opportunity anyway. As you will see, my motivations are more personal things than professional. I could achieve great professional success in my current job if I stay here for enough time, but this is not all I want for my life.

The first one is that I decided to take the leap and try to achieve a desire of mine. I don't know if it's kind of a dream, perhaps. I am very curious about other cultures and for years I have been wanting to live in other countries and I just realized that now is the time. The things that were holding me back now seems to be not so important. And I feel that the longer I wait to try this, the harder it will be to that happen because I may find something else holding me here.

Secondly, as any good professional I seek to work in the best company that I can find, meaning in place that can learn the most and achieve the most professionally. I decided to consider (almost) the entire world in this search because, in theory, I would have the biggest chance to find such excellent place to work. I don't have a particular preference regarding the country. But I must confess that I'm seeking to have a good and suitable life, which I already have here in Brazil, and that doesn't

mean I'm expecting an exorbitant salary, just something reasonable. Logically I don't want to walk backwards in this subject.

At last, if I consider the entire world I think there is a higher chance to find a particular place work, that is not only that what I said, but is something that perhaps I will love to work with per many years. Brazil is very limited in software development: we are full of busyness software like CRMs and ERPs, but I rarely see something like the project that I'm working at the moment. I know that you, a person responsible for hiring developers, can't promise the thing you work with is such lovely thing for me. Neither do I can promise that I will have a love at first sight with your company's busyness, but at least we could have a conversation and see if there is a potential for a long-term professional relationship.

Considering what I said, I hope that it's clear that I want to make a big change in my life. And I know this change is something that will take a lot of effort and time. So, let me be clear that I'm aiming to stay a good few years in my next company, and I'm prepared for that.

Things that I like

TODO

The last cool thing that I did as a job

TODO

Some thoughts on what is a good programmer

TODO

Disclaimer on the following rant sections

The following sections have a strong intonation, meaning that I have a very assertive position on those things and that can be seen as arrogant by some people. I personally don't usually take offense on assertive opinions, specially if it is about something technical. I understand that criticism can be taken to the personal level, specially if it is about something loved, but I hope you have the maturity to do not do that on criticism about technology. On the other hand, this kind criticism can be very constructive.

My thoughts on Software Engineering in practice (Rant Section)

TODO

Rage against Object Orientation (Rant Section)

My opinion on Object Orientation (OO) is in opposition with the common belief. OO programming doesn't provide more abstraction, or better organization, or elegance or simplicity, it just provides the illusion of those things. Let's begin stating what OO is: an object is an encapsulated state which you can only interact with by sending messages, in other words OO is the association between data and functions. There are big problems with that:

- States being passed all around. In the original concept the messages in OO should send copies of states, not references of states. But this is not how 99% of OO programming languages are implemented. This is a big problem because it's harder to debug the code and to implement parallelism becomes painful. OO is just global variables with extra steps.

- OO has nothing to do with better organization. One way to implement OO is through classes, which is basically a way to define types. Well, types and module systems allows you to organize your code because you can give names for some specific kinds of data and group them together. Nothing to do with OO.
- OO doesn't provide better abstraction. Abstraction is hiding irrelevant information (one possible definition). When people are presented OO they are showed trivial examples like a class representing something real like the customer, and the teacher says this is how OO allows you to abstract a concrete thing because your customer class is some sort of model of the real life customer which doesn't have the irrelevant information. But in practice OO programs very often have classes that doesn't represent anything real, like services, manager, factories and other doers. Those classes are there to fill a gap and because OO demands you to have the association between data and functions. So at end of the day the promised of abstraction is not fully delivered, and you end up with many words in your program that have nothing to do with the domain in matter.

So what should I do instead of OO? Should I use functional programming? The opposite of OO is non-OO, or what was usually called procedural programming. We can combine procedural programming with imperative and functional programming, as we can also combine these with OO. Procedural programming is just better.

So, in conclusion I must say that I actually write OO code, but not by my choice. Most systems out there are objected oriented and I, as a humble employee, have the job of programming in them. But of course I would like to see more people stop using OO.

Why dynamic typed languages still exists?

Let me be straight here: dynamic typed languages should have died last century and we should only see it on legacy projects. It's like coal power plants, why they still widely used? Both are obsolete technologies that we should be taking the effort to substitute them, or at least avoid more adoption.

I know, there is a - quote and quote - big debate between dynamic and static typing. But this debate shouldn't exist. Every single facility that dynamic typing gives, you can also have with static typing if the person implementing it is not lazy.

- Oh, duck typing? Meh, universal polymorphism is much better.
- Oh, not writing types? Please, are we still living in caves? Haven't you heard of type inference?
- Oh, no compilation step? Well, you can tweak the compiler for compilation speed (in development). Unless you need to recompile everything, this shouldn't be a problem in development. And nowadays we have fast computers that are affordable.

Static typing is just a more advanced technology!
Let's not make a debate about it, ok?

Rage against company's bullshits (Rant Section)