Michał Jarmakowicz

I'm C++ developer with more than 5 years of experience.

During that time I was working with highly complicated and distributed software which drives the most modern radiomodules of the 4G and 5G technology. The job required not only software language skills, but also ability to learn specific domain knowledge which includes the basic principles of signal processing and how it is implemented in the digital (ASICs) and analog side.

But the more interesting fact is the distributed and multi-actor nature of the system which I was developing and maintaining during that years. I strongly believe that skills needed to design solutions with multiple actors and to foresee and prevent problems associated with such systems is crucial not only in the embedded software, but in the every multi-actor and distributed one regardless of used programming language.

Education

Feb 2017	**Bachelor of Engineering** Computer Science / Department of Electronics Wrocław University of Science and Technology
March 2019	**Master of Engineering** Computer Science / Department of Electronics Wrocław University of Science and Technology

Work Experience

Oct 2017 - Oct 2018	**Working Student / C++ Software Developer** NOKIA
Sep 2018 - Aug 2021	**C++ Software Developer** NOKIA

Aug 2021 - Apr 2023 **Technical Leader / C++ Software Developer**

NOKIA

Skills

\

Achievements

Mysterious Filter Issue

In one of the products the very strange hardware issue was found during internal testing. After the specific and unusuall sequences of configurations the filter coefficients went very wrong and signal was damaged. Situation was hard to accidently generate in the real environment and was not a great danger for any customer. Still it was a great blow to the honor and proud of Nokia. Because of it this noble organization sends many of its best programmers and engineers to analyze and solve the issue.

Many of them failed. Finally it was my turn. I was the first men who found conditions which have to be met by configuration to generate issue and then I proposed and implemented solution. Implemented mechanism was detecting that conditions and was applying recovery action in the way which was entirely transparent for the higher layers of software.

Saving The Day for Throughput

This was the time when Nokia took part in the race for 5G. One of the crucial product designed for exceptionally important market needed implementation of the gain control for receiver path. Unfortunatelly team responsible for the product didn't have a free man for that task.

I was delegated to do it. I had to face the constantly changing requirements and the very complicated nature of the hardware, but I succeeded to deliver the working mechanism.

Thanks to that implementation receiver path achieved nearly 100% of expected throughput regardless of physical deviations of hardware and environment temperature.

Childbirth Care

After some time as Technical Leader I was borrowed to the Architecture Team which needed someone to oversee the birth of the new Nokia product.

The new radio module was partially using the same hardware as other products but in the way which was never seen before. My goal was to ensure that code between that products will be common as much as possible and to ensure that development will not go into any pitfall.

I did it. I spot a coming disaster and I raised an alarm that the way chosen by the team for adjusting common software for new requirements will result in the horrible spaghetti code with swarm of the IF statements placed everywhere.

Then together with the rest of Architecture Team we proposed the better solution. I was given a task to prepare detailed software design and to oversee the development. Thanks to it the mechanism of dynamic configuration reloading was implemented and it let us to keep the rest of the code agnostic about specific requirements of that "unusual child".

I Will Not Let You Burn

Power amplifiers are delicate devices and requires procedures in software for preventing burning and performing recovery actions. Designing such procedures to ensure that all actions will be done on time and the recovery action will be transparent for all other software procedures across entire stack is a difficult task.

The challenge was even more challenging because of the "blocking" nature of the system. It prevented the use of the naive approach and forced us to look for a new solution.

I prepared many detailed propositions how to solve the issue. After many discussions decision has been made to create separate service which will take over the control of the all low-level services which controls the power amplifiers or realizes the procedures which depends on the amplifiers. The service processed the messages and modelled the rest of the system in the way which let us to handle all amplifiers faults on time.

When decision has been made I was given a task to prepare detailed software design of this new service with all algorithms and class hierarchy. I did it.

Contact

---

michaljar.github.io

www.linkedin.com/in/michał-jarmakowicz-72126925b