tiagokatcipis

software engineer

contact

+5548991060132
tiagokatcipis@gmail.com
GitHub
LinkedIn
Blog

languages skill

English: advanced

programming languages

Go, C, C++, Python, Lua, Nash, Bash, Javascript

protocols

DBus, RTMP, HTTP, SIP, RTP, AMQP

cloud

Kubernetes, AWS, Azure, Google Cloud, Docker, CoreOS, Terraform

automation

Ansible, Make

dev environment

Vagrant, Docker Compose

monitoring

Prometheus, Grafana, StatsD, Sysdig

education

2006–2011 **Bachelor of Computer Science** UFSC

experience

2017–now **Neoway** Florianópolis, Santa Catarina

Software Engineer - Data Platform Team

Led the migration of the entire data platform from AWS to Azure, which resulted in a fully automated infrastructure. To build this automation an open source project was developed at Neoway to deploy infrastructure at Azure called klb which is a stateless tool to build infrastructure.

After the infrastructure migration I helped to solve an audio captcha, something that had not been done before at Neoway.Doing that involved some researching on current techniques and culminated on a service capable of solving a specific audio captcha with 80% of assertiveness.

The creation of this service involved the integration of different tools and techniques, like sox to remove noise and segment audios, MFC (Mel-Frequency Cepstrum) to extract features from the audio and SVM to train models. This service (and helper tooling) was developed in Go with integration to some open source Python tools for feature extraction.

2015–2017 **Neoway** Florianópolis, Santa Catarina

Lead Software Engineer - Data Capture Team

My main job was to lead the development of a new data capture architecture that would scale better than the current one, both in performance as in development effort required to develop new web scrappers and maintain the current ones.

The work started by substituting the current web scrapper language (a XML) with Python using the Scrappy framework. This enabled us to tackle new challenges like parsing PDF or other complex binary formats (the legacy XML based scrapper lacked this capability). Another small Python framework was also developed to aid the development of scrappers that used Selenium when required. Both frameworks generated metrics using the StatsD protocol.

The next step was to extract logic that resided on the legacy web scrapper engine into separate services, like proxy, captcha, and raw storage services. These services where developed in Go and exported metrics using the StatsD protocol. Another service was also developed to integrate the new web scrappers into the old data pipeline (while the new data pipeline was being developed by another team). This experience involved a lot of designing and coding in Go and Python and also brainstorming with other teams on how to define protocols integrating the data pipeline.

Efforts were made to build an automated development environment that enabled developers to build and debug web scrappers in isolation of each other. To do this we used Docker and docker-compose. The deployment and scheduling of web scrappers was solved using Kubernetes on top of CoreOS at AWS and later at Azure. Kubernetes was essential since each web scrapper is handled as a independent unit of deploy and we had more than 200 of them (and also the services).

On top of that I was coaching the team on this new architecture and how to develop the web scrappers in Python, including the use of automated tests, code reviews and continuous integration to improve the quality of the web scrappers, solving some serious issues with bad quality on the captured data.

The team was responsible for everything, from coding to deployment and operations, including maintaining the Kubernetes cluster healthy. For monitoring we used Sysdig. I also participated on the screening and hiring process of new candidates for the team.

2012–2015 **Dígitro** Florianópolis, Santa Catarina
Lead Software Engineer

Worked on the development of a VoIP phone with touchscreen in a very strict environment (a Blackfin DSP processor with no MMU and 64MB of RAM). The project was built from scratch in C and used DBus to integrate different processes. This project involved learning considerably about SIP and VoIP state machines and also memory issues because of the lack of an MMU, like external memory fragmentation. The development environment was rather complex and involved cross compilation with very specific tools. This build of this development environment was done using Vagrant and Ansible.

Some refined functions on the VoIP phone required integration with our PBX that used a rather complex proprietary protocol. To solve the integration problem we developed a PBX REST service in Lua that integrated with the current legacy protocols exposing them in a more web friendly way. This service was implemented in Lua.

My last project at Dígitro was a solution to web audio playback with very specific audio effects (like silence removal, change in pitch, DTMF/Fax detection) that had to be developed using HTML5. It was the rewrite of an old system that worked using Flash. Some of the components of the old project, like Gstreamer plugins, were reused, but the service itself had to be rewriten using REST and HTTP to do the media streaming (and control). This was developed in Javascript (NodeJS) integrating with the C code for streaming using IPC (Inter Process Communication). The original audio files where encoded in IMA ADPCM and another proprietary CODEC and were exported through SFTP. Part of the problem was also integrating with this legacy infrastructure and transcode it to a more web friendly format.

During these projects I always practiced and advocated TDD (even on embedded systems in C) and helped people on the team to write automated tests for their code. I also coached two new members in the team.

2010–2012 **Dígitro** Florianópolis, Santa Catarina
Software Engineer

I started working on a solution to web audio playback with very specific audio effects (like silence removal, change in pitch) that had to be developed using Flash. To solve that problem I worked with two different open source C++ projects that did reverse engineering of the RTMP protocol to develop our own Flash Media Server. I worked directly with the integration of the server playback logic with Gstreamer and the plugins that enabled the desired effects on playback.

The next project was a solution to biometric identification. This has been solved by building a service around a third party C library that built and scored voice models. I developed a REST service in Lua that integrated with C code that built the voice models and used MongoDB to store the voice models and perform searches on the database.

Then I developed a service to perform searches for specific words in audio files. The service would replace a old one so it had to implement the same textual/proprietary stateful protocol of the old one. The service itself was implemented in Lua with bindings to C code that did the actual search of words.

2008–2010 **Dígitro** Florianópolis, Santa Catarina
Trainee

I started helping in the development of an cross platform (Windows and Linux) audio streaming library for a VoIP softphone, aiming at porting the current application that was Windows only to Linux. The library was C code being cross compiled to Windows using mingw.

After that I got involved developing a prototype of a voice biometrics system. It involved building a small web server in Python that integrated with third party libraries in C that did the build and scored voice models and a cross platform Python application that provided a user friendly GUI to the web server.

2007-2008 **Cyclops / LAPIX** Florianópolis, Santa Catarina
Trainee

Worked on adding new features on the system responsible to integrate medical equipment to the DICOM system. I developed a cross platform application that generated a customized GUI based on a XML description of a form, making it easier to people working at hospitals to manipulate the forms and save them back at XML. Like a domain specific XML editor.

This involved learning C++ and XML parsing, together with developing cross platform GUI applications, on this case using WxWidgets. The code has been tested using CppUnit.

I also automated the setup of the development environment using Python.

open source projects

2017-now **mdtoc** https://github.com/madlambda/mdtoc
A very simple table of contents generator for markdown.

2016-2018 **nash** https://github.com/NeowayLabs/nash
Nash is a shell language focused on simplicity and having a nicer syntax than traditional shells and support to containers. It also strives to be safer than traditional shells.

2016-2018 **klb** https://github.com/NeowayLabs/klb
klb is used to automate infrastructure creation on AWS and Azure. I got involved on designing the support for Azure since this was the tool used to migrate Neoway production infrastructure from AWS to Azure.

2013 **CppUTest** http://cpputest.github.io
CppUTest is a C /C++ based unit xUnit test framework for unit testing and for test-driving code.
In this project I worked both on improving the documentation and at adding new native types to the mock framework (which involved some refactoring).

2012 **GStreamer** http://www.gstreamer.net
GStreamer is a library for constructing graphs of media-handling components. I contributed with a plugin named *removesilence* and some documentation for the GstCheck documentation.

2010-2011 **Pattern detection on H.264** https://github.com/katcipis/h264.pattern.detection
This is my Bachelor's Thesis and it consists of a prototype of a H.264 CODEC that uses OpenCV and H.264 internal algorithms to do pattern detection and object tracking integrated on the encoding process.
Metadata generated on the encoding process is integrated on the video bitstream on conformance with the standard.

2010-2011 **LuaSofia** https://github.com/ppizarro/luasofia
Lua binding for the Sofia-SIP library. Contributed to the project from the start, helping to make decisions about the design of the software and documenting it.

2010 **GPS tracking system** https://github.com/katcipis/gps.tracking
System designed to provide the location of a device at the receive of a position request using SMS.

2010 **LuaNotify** https://github.com/katcipis/luanotify
Lua library that implements a simple Pub/Sub system inspired on glib GSignal API.

presentations

2018 **Object Orientation in Go** The Developers Conference

For people that come from a background on Java or other classic object oriented languages (like C++) there is also some discussion on if Go is actually object oriented.
In this presentation I try to present Go as a language that is more object oriented than these classic languages, at least according to the original foundations of object orientation.
Presentation source can be found here.

2016 **Building Resilient Services in Go** GopherCon Brazil

Resilience is not about never failing, but how do you recover from it. How can you prevent your services from locking down or exhausting all its resources ? How to perform graceful service degradation ? Can this kind of behaviour be tested properly ?

On Go we have some new features, like Contexts, that helps us to model timeouts and cancellation properly.

They can be combined with other useful features as select and channels to model timeouts and resource pools, which can be essential to provide proper service degradation instead of total failure of the system.

On this talk I try to answer this questions using new features available on Go 1.7, direct from production ready software.

Presentation source can be found here.

2016 **Real Life Kubernetes** The Developers Conference

On this presentation we will give a short introduction on Kubernetes and show the experience of learning and using Kubernetes on production for two very distinct systems.

The first one is a data acquisition system, involving running multiple instances of different crawlers, storage services, captcha breaking services, message brokers (like RabbitMQ) and database integration outside the cluster.

The second one is a web application, involving network analysis using graphs with the ultimate goal of fraud prevention. The application is strongly bounded with the microservices architecture and the twelve factor app.

Graph and document databases, cache layers, a message broker and a distributed filesystem are some of the technologies surrounding the application ecosystem.

Presentation source can be found here.