

# Resume: Daniel Owen van Dommelen

- I like working in smaller teams with big ideas.
- I was a travelling developer, now looking for a permanent home (from which to work).
- It's not that I want to work 100% remote only, it's that I actually exist only as an image on a screen.
- My two favorite languages to work in are Go and Ruby.
- I believe if things are not easy, they are not finished.

## Work Experience

### Rocsys : Tech Lead

REMOTE: AUG 2021 - JAN 2022 (however, available from December, I can explain.) DOMAIN: EV Charging Robots

#### Stack

Infrastructure	Backend	Frontend
Argo Suite for GitOps	Prometheus/Grafana for monitoring	Grafana App Plugin
Bare metal, 6 servers	ElasticSearch/Kibana for logs	Grafana Datasource Plugins
AWS S3 + MinIO	Golang Custom Services	Grafana Panel Plugins
Azure AKS	Docker/ContainerD	React
TransIP VPS (CoreOS)	Kubernetes (Lokomotive on Flatcar)	Yarn

#### Remarks

Built infrastructure, backend, and frontend from scratch.

Setup backend process guidelines and CI/CD tooling to keep overhead to a minimum.

Wrote extensive documentation on all elements involved with developing and operating the systems, including architecture design documents and motivations.

### Bit8 : Team Lead (2nd Time)

REMOTE: DEC 2020 - MAY 2021 DOMAIN: Online Gaming

#### Stack

Infrastructure	Backend	Frontend
Virtualized Windows Server	C# .Net 4.7.1	Php 5.40.6
Hyper-V	MySQL 8	Yii 1.1
Virtualized CentOS	Dockerized Redis	Javascript

#### Remarks

Was given back the team of M2 with only 1 remaining "original" member, the rest was only recently put together, which presented some challenges in process. Reintroducing some of the core processes helped a lot.

Spent quite some time upgrading the developer tooling using some more modern techniques, including putting the platform on Docker and into Docker Swarm to allow developers to run the 14.000+ (yes really) integration tests.

Developed a [C# .Net Core project that turns MySQL grammar into execution trees](#) to help pull ancient and massive stored procedures out of the database and turn them into code.

Wrote a tool in Go to canonize configuration tables in the database which would be changed during QA and not changed back, causing `heisenbugs`.

### Orbisk (Zero Foodwaste) : Backend Developer

UTRECHT: OCT 2019 - OCT 2020 DOMAIN: IoT Device for Food Waste Monitoring

Stack

Infrastructure	Backend	Frontend
Google Cloud	Go	Vue
Arduino	Python	Veutify
Raspberrypi	C++	AI annotation tools

Remarks

Introduced Go into the company.

Built a gateway service in Go with various data endpoints, queue based with publishers and observer, and a custom grammar for an ORM to transparently be able to talk to BigQuery (cold-store) and Postgres (hot-store).

Worked on the core data pipeline (ETL).

All this was using many Google Cloud services such as Buckets, Serverless Functions, Dataflow, Pub/Sub, Big Query, Postgres, and Firestore, amongst others.

Worked on camera improvements on the software layer, eventually having to write raw MMAL code to control things at the GPU and VPU level.

Quik Tech Ltd. : Backend Developer

MALTA: NOV 2018 - JUL 2019 DOMAIN: Lottery Games with Hardware Component and Live Video Streaming Layer

Stack

Infrastructure	Backend	Frontend
AWS	Go	Vue
Arduino	Docker	WebSockets
Raspberrypi	RabbitMQ	Hardware Lottery Games

Remarks

Worked on the first lines of code and design of the overal system, including tooling to provision developer environments.

Worked on the real-time video layer in research and implementation capacity, accidentally invented my own [video streaming protocol over WebSockets](#), later implemented a more production-ready streaming service from an external company.

Worked on machine learning model to recognize lotto balls as a redundant system to the hardware's own RFID based setup (which failed often).

Worked on a video monitoring tool that could send a picture of every ball drawn to AWS S3 as well as compressed video of the entire game for audit trailing. Also implemented features to detect balls being stuck in the machines.

Bit8 : Team Lead (1st Time)

MALTA: APR 2018 - OCT 2018 DOMAIN: Online Gaming

Stack

Infrastructure	Backend	Frontend
Virtualized Windows Server	C# .Net 4.7.1	Php 5.40.6
Hyper-V	MySQL 8	Yii 1.1
Virtualized CentOS	Dockerized Redis	Javascript

Remarks

Worked on the rewriting of the "player selections" component.

Worked on a "continuous improvement" effort, my focus was on improving the (Java written) code obfuscator, which I rewrote in Go and to get the 14.000+ (really) integration tests to go all green.

Spend a lot of time on process restructuring within the team, and the usual team management tasks, such as mentoring, one-on-ones, hiring, interviewing, etc.

## QuickBed : CTO

---

PARIS: OCT 2016 - MAR 2018 DOMAIN: Refugee Accomodation

### Stack

Infrastructure	Backend	Frontend
Heroku	Ruby on Rails	Rails Templates
	Python	Javascript
	MongoDB	Google Maps

### Remarks

Wrote most of the initial platform, recruited and trained 5 junior developers, participated (and later mentored) at the TechFugees hackathons, participated at the Data for Good hackathons and invented a new search system that is language agnostic (not only in translation, but also in meaning, slang use, etc.)

## Create50 : Full Stack Developer

---

LONDON: DEC 2014 - OCT 2016 DOMAIN: Online Screenwriting and Filmmaking Competitions

### Stack

Infrastructure	Backend	Frontend
Heroku	Ruby on Rails	Rails templates
Jellastic	MongoDB	Javascript
	Redis	

### Remarks

Wrote the entire platform while being almost immediately in production. We developed features in a way that they could seamlessly integrate with the active production site, which often required complicated data migrations as features shifted around.

Wrote a backoffice system for site management and for judges to be able to provide verdicts and selections of content to proceed to next rounds.

## Gigit : CTO

---

LOS ANGELES: APR 2013 - OCT 2013 DOMAIN: Private Music Events

### Stack

Infrastructure	Backend	Frontend
Heroku	Ruby on Rails	BackboneJS

### Remarks

Wrote the entire platform and booking system quite quickly, maintained it, and implemented new features. Did some interviewing and hired 2 candidates later on. Also, uniquely, supported the arrangements and artists needed for the live events and shows, as we were a very small team, we had to be present for all the events to help in this way.

## Livestation : Full Stack Developer

---

LONDON: SEP 2011 - MAR 2013 DOMAIN: Live News Streaming

Stack

Infrastructure	Backend	Frontend
Heroku	Ruby on Rails	Rails templates
Bare Metal	Custom C++ Video Muxer	Javascript
Samsung Smart TV	Samsung Smart TV SDK	Flash

Remarks

Worked on backend systems and database layer, invented a short text keyword extractor and used it to connect subtitles to YouTube recommendations, did some computer vision to detect context in live unannotated video streams, developed a Samsung Smart TV app client to the streaming platform.

Introduced and implemented Scrum practices and Agile methodologies.

Smartdate : Full Stack Developer

---

PARIS: APR 2011 - AUG 2011 DOMAIN: Online Dating

Stack

Infrastructure	Backend	Frontend
Heroku	Ruby on Rails	Rails templates

Ixly : Full Stack Developer

---

ZEIST: OCT 2010 - APR 2011 DOMAIN: Psychological Testing and Human Resources

Stack

Infrastructure	Backend	Frontend
Heroku	Ruby on Rails	Rails templates

Victor Chandler International : Full Stack Developer

---

GIBRALTAR: APR 2010 - OCT 2010 DOMAIN: Online Gaming

Stack

Infrastructure	Backend	Frontend
Unsure	Ruby on Rails	Rails templates

SellaBand : CTO

---

AMSTERDAM: APR 2009 - APR 2010 DOMAIN: Crowd-Funding

Stack

Infrastructure	Backend	Frontend
----------------	---------	----------

Infrastructure	Backend	Frontend
Bare Metal CentOS	Ruby on Rails	Rails templates

## Noxa : Full Stack Developer

---

HOOFDDORP: DEC 2006 - DEC 2008 DOMAIN: Staffing, Career Development, and Human Resources

### Stack

Infrastructure	Backend	Frontend
Bare Metal CentOS	Ruby on Rails	Rails templates

---

## ADDITIONAL HISTORICAL WORK-EXPERIENCE DATING BACK TO 1998 AVAILABLE ON REQUEST

---

## Active Projects

---

### My Resume

---

#### [Project Link](#)

The document you are reading. Used to be automatically compiled by Gitlab on push to the repository. Moved away from Gitlab because it is bloated and now I am working with Kubernetes the Argo Suite tooling is more appropriate. The resume has not been set up in that system yet. A Dockerfile should be included in this repository with the build steps.

### Wrkspc

---

#### [Project Link](#)

Likely still a hidden repository at the moment, but soon to be released open-source. It is a binary that contains Docker (ContainerD) and a custom Kubernetes distribution so this binary is all that is needed to setup a work environment where every terminal command you are used to using is wrapped in a container and automatically downloaded (pulled) when a command is executed. I call it the last fresh install you will ever need to do. Also it has huge potential in performance, as connecting it to a remote cluster could upgrade anyone's laptop with that compute.

### SPDG

---

- [Interface](#) Secure Private Data Grams (older version)
- [Implementation](#) example using MySQL binlog as an event source (older version).

Secure Private Datagrams were an experiment, but have now successfully been implemented in a real platform (the Rocsys backend) and performed like a charm. Essentially it is a thin wrapper around any other data, using gob encoders and decoders which can marshal any kind of data to bytes, which can then be placed in a Datagram's Payload field. By making all your methods in the backend accept and return only Datagram's (I call this a mono-typed system) you gain enormous flexibility in data movement, and you can transport anything, anywhere. I have successfully transported network connections, over the network, stored them in AWS S3, retrieved and used them.

Currently I am working on additional features around these, chiefly a wrapper around Go channels and WebSockets that can use either of these, transparent to the developer, and makes a Goroutine/channel workflow extendible over a network.

## Languages

---

In order of favoritism (leaving out the ones I can't remember at the moment):

- Go
  - The first I reach for if it needs to be anything more than a one-off script.
  - Love how it makes the design naturally concurrent and some of its opinionated (forced) standards.
  - Hate the trailing comma thing. Too far! Also auto-indentation great, but I choose what is on which line (or I guess not).
- Ruby
  - Love the clean syntax.
  - Love the easy meta-programming features.
  - Eventually too slow and not strongly typed enough to be a one-stop shop for any requirement.
- Erlang
  - Having no loops is great. Takes your designs to next levels.
  - Enjoy the syntax.

- Hate the recursion termination. Not a fan of method (or constructor) overloading.
- Python
  - Reasonable syntax, just compared to Ruby not ideal.
  - Good for machine learning prototypes.
  - One benefit over Ruby is that it shows a few techniques in a way that was easier for me to understand.
- Fortran
  - I like it for its simple syntax but incredible speed working with matrices.
  - The first language I learned when I discovered I am effectively a language collector.
- Basic
  - My first language.
  - A very powerful learning tool to get into programming.
- Rust
  - I wish I liked it more.
  - For pure, raw performance, this would be my choice.
  - But again, I wish I liked it more to dive deeper into it, yet Go always wins by favoritism.
- Php
  - I learned it has more tricks up its sleeve that people give it credit for.
  - Recent improvements to the language have solved some of its canonical issues.
  - I came to mind it a lot less recently.
- Perl
  - Nothing good about it.
- Javascript
  - Not a great language, but still needed in the web developer world.
  - Typescript helped it a great deal though, to be completely fair.
- C
  - No longer needed after Rust, except to support legacy/existing projects.
- Delphi
  - My first official (professional) language.
- C++
  - No longer needed after rust, except to support legacy/existing projects.
- C#
  - Workable syntax.
  - Too verbose at times.
  - Eco-system is bloated and complex.
- PL/SQL
  - Useful extension to standard SQL syntax.
- Foxpro
  - Not much I can remember about it, but it was the first language I had to learn on-the-fly at a job.

I have also implemented some of my own basic languages, mostly based on stack or register virtual machine designs. These have never become very mature, and are just a hobby project.