



# Marc Coiffier

Systems administrator

✉ marc@coiffier.net

☎ 06 59 83 75 00

🎂 June 28th, 1991

🏠 15, allée des Vosges  
38130 Échirolles

🌐 Online résumé (<https://coiffier.net/CV/?lang=en>)

## Skills

Empathy

Curiosity

Adaptability

Teamwork

Initiative

Tenacity

Politeness

## Competences

**Systems administration** (Debian, ArchLinux, NixOS)

**Automation / Deployment** (Docker, Nix, CI/CD, ...)

**Fluent english**

**Functional programming** (Haskell, OCaml)

**Collaborative development** (Git, GitHub, SourceHut, ...)

**Low-level programming** (C, C++, Rust)

**Formal methods** (Coq)

**Web development** (PHP, JS, React, CSS, Tailwind, ...)

## Experience

2016-2020

### Thesis on program proofs at VERIMAG lab

Designing an extension to the Calculus of Constructions that can generate induction principles on Church encodings.

Using Coq to prove coherence and strong normalization properties for this extension.

Using Coq to prove certain algorithms of the OCaml standard library (for example, that its implementation of MergeSort is correct and stable).

2012-2016

### Systems administrator at the UFR IM<sup>2</sup>AG

Managing the lab computers and servers at the UFR IM<sup>2</sup>AG.

- Providing level 1 IT support to the UFR staff, escalating to central IT when necessary
- Deploying Linux/Windows dual-boot images using CloneZilla over PXE
- Orchestrating Linux updates / software installs via a local Debian package server
- Maintaining infrastructure and service documentation in a semi-open wiki site
- Monitoring lab computers using TARSIS, and internally-developped program for infrastructure management

## Personal projects

---

### Curly : a compiler for a simple functional language

Link to the project (<https://coiffier.net/projects/curly/>)

A simple compiler, written in Haskell, that enforces reproducible cross-platform builds.

- A type-system that can infer recursive types (as with the term  $\lambda x. xx$ )
- First-class typeclasses
- Single-pass linking, for just-in-time and static executable assembly code generation
- *Content-addressed* modules to provide reproducible builds by default
- Built-in documentation format, and second-order documentation (no more doc comments)
- Automatic *code-signing* when distributing modules

### Omega : a micro-kernel

Link to the project (<https://coiffier.net/projects/omega/>)

A toy projet that taught me a lot about the way operating systems work internally.

- Multi-process with memory protection
- Multi-thread with inter-process synchronisation
- Complete with basic CLI and a scripting language
- Rudimentary keyboard, and VGA, drivers
- Blazing fast to boot !

### Viz : an interaction network visualizer

Link to the project (<https://coiffier.net/projects/viz/>)

In order to design a beta-optimal evaluator for the Calculus of Construction, that uses interaction nets.

- Create and delete nodes and edges in an SVG canvas
- Apply reduction rules to see how the graph evolves
- Automatically smooth reposition nodes to minimize "internal tension" and make visualization more legible (uses gradient descent)

## Education

---

2016

### Network Engineer

Knowledge acquired by experience after 4 years' working as a systems administrator at the UFR IM<sup>2</sup>AG.

2011

### Maths and Computer Science Licentiate

At the Université Joseph Fourier in Grenoble

2008

### Bachelor's Degree

At the Champollion High School in Grenoble

## Interests

---

Music : piano, violin and guitar

Dance : Lindy Hop with Grenoble Swing (<https://www.grenobleswing.com/>) and traditional dances with the Folk Escapade (<https://escapadefolk.netlib.re/>)

Board games with the Grenoble Games' Den (<https://maisondesjeux-grenoble.org/wordpress/>)