Marco Maida

www.maida.me @ mmaidacs@gmail.com

I am comfortable working on large code bases and designing systems to solve complex problems at scale.
I have extensive experience with **C++**, **Rust**, **C**, **Python**, **C#**, and **Coq** in production enviroments.
I am **outgoing** and I **love working in teams**. I am currently based in **London**.

Software Engineering

Since 2022 — **Software engineer – Robotics and AI.** — Wayve Ltd
I designed, implemented, and maintained several key systems of the on-board software stack of Wayve's autonomous cars. The software collects frames from different sensors; aligns, packages, and uploads enormous amounts of data, and runs a neural network for real-time inference. *(C++, Rust, Python)*.

2016-2019 — **Software Engineer – Videogames.** — 34BigThings
I contributed to five major titles, designing and implementing game infrastructure, AIs, gameplays, and dev tools. I worked on single-player and online-multiplayer games shipped on Steam, PS4, XboxOne, Switch, and mobiles. *(Unity3D, C# and Unreal Engine, C++)*

2015-2016 — **Software Engineer – Videogames, Simulation** — Maserati, Teoresi, Choralia
This experience groups two freelance B2B projects. First, I built an educational game for mobiles and browsers. I led the project end-to-end, managing one artist I hired. Second, I collaborated with an engineer and an artist to create a 3D visualization tool used for product presentation. *(C#, Javascript)*

2013-2016 — **Software Engineer – Industrial Software.** — R.O. srl
I designed and maintained a suite of software solutions for glass processing factories. The software tracks orders and minimises the cutting machines' work, product waste, and logistic delays. I started as an individual contributor and later transitioned to managing a team of 4 engineers *(C, C++, C#, SQL)*

Education and Research

2022 — **Research Internship** — Bloomberg LP
I worked on accelerating SAT solving using GPUs *(C++, CUDA)*.

2019-2022 — **PhD Student.** — Max Planck Institute
I worked at the intersection of formal verification and real-time systems. Additionally, I worked on trace-based schedulability analysis on Linux systems. I mentored three interns and published three papers. *(COQ, C, Rust)*

2019-2022 — **Master in Computer Science.** — Technische Universität Kaiserslautern
I specialised in real-time systems.

2016-2019 — **Bachelor in Computer Science.** — Università degli studi di Torino
I specialised in computability and formal methods.

Publications

2021 — **Foundational Response-Time Analysis as Explainable Evidence of Timeliness.** — Max Planck institute
I developed POET, a tool that yields a formally verified worst-case-scenario timing analysis. I first-authored a publication at ECRTS2022, winning its *outstanding paper* award.
(https://drops.dagstuhl.de/opus/volltexte/2022/16336/pdf/LIPIcs-ECRTS-2022-19.pdf)

2022 — **From Intuition to Coq: A Case Study in Verified Response-Time Analysis, FIFO Scheduling.** — Max Planck institute
We developed a formally verified response-time analysis for FIFO schedulers, challenging the classic pen-and-paper approach. (https://people.mpi-sws.org/~kbedarka/rtss22.pdf)

Private & Open-Source Projects

2022 — **Treecode.** — Personal
I created a novel 2D scannable code that encodes messages as unique trees. (www.maida.me/treecode)

2021 — **POET.** — Max Planck Institute
The tool I created as part of my first academic publication. It is the first-ever implementation of a foundational response-time analysis. (https://gitlab.mpi-sws.org/RT-PROOFS/POET)

2020-2022 — **PROSA.** — Max Planck Institute
I was one of the main contributors of PROSA during my stay at MPI-SWS. PROSA is one of the most influential formally-verified frameworks in the real-time systems' community. (https://gitlab.mpi-sws.org/RT-PROOFS/rt-proofs)

2018 — **Fast Mobile Cycle (FMC) Framework and Toolkit.** — 34BigThings
I developed a Unity3D framework that accelerates the creation of production-ready casual games, paired by a Python toolkit to execute bulk operations on the games. (www.github.com/34OpenThings)