Igor Zhirkov | CS & Engineering

+33 647 215 455 (FR) • igorjirkov@gmail.com • rubber-duck-typing.com

Experience

ITMO University **St.Petersburg, Russia**

Curriculum designer, Teacher *2011–2022*

- In 2020–2022, lead designer of the new 4-year curriculum synthesizing architecture of computer systems, theory of complex systems and programming in a team of ≈ 10.
- Developed a course on low-level programming, program execution, and elements of compilers, over 2000 students in 2013–2019, feedback strongly positive.
- Developed a course on programming languages in computer systems, over 800 students in 2020–22, feedback strongly positive.
- Supervised educational projects including compilers and language VMs e.g.
  - `https://github.com/Tehsapper/yaspcompiler`
  - `https://github.com/merfemor/dwarf-compiler`

Personal projects

- Educational Forth system with objects, GC and library, core written in x64 Assembly. `https://github.com/sayon/forthress`
- Forth compiler in assembly with minimal core and bootstrap `https://github.com/sayon/forthress-2`
- A JIT compiler in C++ for an imperative language with closures. `https://github.com/sayon/simple-jit-cpp`
- Trancert: new take on the automated verified refactoring tool for C based on CompCert semantics, written in Ocaml and Coq `https://github.com/sayon/trancert` (access on demand)

IMT Atlantique **Nantes, France**

Researcher, team Gallinette *2016–2020*

- Developed an automated refactoring tool `refaccert` for C programs using Coq, Ocaml and CompCert, a certified C compiler. The tool is capable of removing and introducing local variables in C programs. It is formally proven that, for all input programs, the set of behaviors of the input C program matches exactly the set of behaviors of refactored program by bisimulation using a complete operational style formal semantics of C.
- Development experience with CompCert, a certified C compiler written in Ocaml and Coq.

JetBrains **St.Petersburg, Russia**

Programmer *2013,2014*

- Kotlin development team, project Kannotator, a static analyzer. Inferring nullability for parameters of generics in Java code i.e. `List<T>` vs `List<T?>` to automatically annotate it for better interoperability with Kotlin. Also contributed to the compiler.
- Verified a filesystem model for JetPad, a collaborative editor, using Coq and SSreflect. The model was a CC model of operational transformations of sorted trees, and the convergence had to be proven and verified.

Competences

Languages: Coq, C, x64 Assembly, Scala, Forth, Java, C#.

Programming languages theory: logic, language design, compilers, functional programming, formal semantics

Low-level programming: programming, disassembly, reverse engineering

Publications

Book: Zhirkov, I. Low-Level Programming: C, Assembly, and Program Execution on Intel© 64 Architecture. Apress, 2017

Technical report: Zhirkov, I., Cohen, J., & Douence, R. Memory bijections: reasoning about exact memory transformations induced by refactorings in CompCert C. 2019

Education

IMT Atlantique, team Gallinette **Nantes, France**

Computer Science, Doctoral studies *2016–2020*

ITMO University **St.Petersburg, Russia**

Software Engineering, graduated with MSc degree *2015–2016*

Academic University of the Russian Academy of Science **St.Petersburg, Russia**

Applied Maths and SE, Compilers, formal semantics, functional programming, virtual machines. *2012–2014*

ITMO University **St.Petersburg, Russia**

Informatics and technology, graduated with BSc degree *2008–2012*