Thomas Durieux

SR. ENGINEER · ENDOR LABS
AI SAST · PROGRAM ANALYSIS · SCA
TOP 2% SCIENTIST · STANFORD 2025

Static analysis at scale — from compiler-grade tooling to production AI SAST

SAST | REACHABILITY | AST | LLM | OPEN · US / EU / REMOTE

loc Rotterdam, NL
mail thomas@durieux.me
web durieux.me
gh github.com/tdurieux
li linkedin.com/in/thomasdurieux

Program-analysis engineer with a decade across academia and industry — now back to shipping product as a Senior IC at Endor Labs. Built static-analysis infrastructure that other people depend on: **Spoon** (Java analysis library, 1,500+ dependent OSS projects), **Anonymous GitHub** (production service, 12.5M requests/month), **docker-parfum** (cross-language AST autofix). Currently leading **AI SAST** backend reliability and contributing to build-less **reachability-based SCA** at Endor; named inventor on **US 12,430,446** (SCA for AI-generated code). 41 peer-reviewed publications, 3,800+ citations.

§01 Experience

10+ YRS

01 2024.09 – Today
Senior Engineer, AI + Program Analysis — Endor Labs, Netherlands
Owns scanner correctness and reliability for the **AI-assisted SAST platform**: scan determinism, cache integrity, scoped & incremental scans, and recovery from interrupted scans across PR and main-branch flows for enterprise customers including **Dropbox** and **Rubrik**. Drove **monorepo and large-repo scanning** reliability for AI initiatives — repository indexing, scan-time forecasting from indexed lines of code, and resumable analysis. Co-defines how AI SAST quality is measured and improved: benchmark harness, public benchmarking, false-positive reduction, detection-confidence scoring, model fine-tuning over curated vulnerability / CVE training data, and customer-feedback loops feeding back into the product. Owns code APIs and customer-facing dashboards. Named inventor on **US 12,430,446** — SCA for AI-generated code via language-aware parsing and cryptographic block hashing. Active research on software supply-chain attacks (xz Utils post-mortem, 2025).

02 2022.06 – 2024.09
Assistant Professor, Software Engineering — TU Delft, SERG group
Connected **program analysis** to **AI-assisted development** ahead of the curve: empirical studies on the security risks of LLM-generated code, container supply-chain attack surface, large-scale evaluation of static-analysis tools. Industry research collaboration with **JetBrains** on developer tools for runtime analysis. Designed and taught the graduate program-analysis curriculum; supervised PhD and MSc students. This research seeded the AI SAST direction Endor is now shipping.

03 2020.02 – 2022.06
Post-doctoral Researcher — KTH, Stockholm, SE
Reachability-based dependency analysis for Java: large-scale measurement of unused transitive dependencies in the Maven ecosystem, and bytecode-level debloating that removes unreachable code while preserving correctness. Designed call-graph-driven reachability that distinguishes *declared* from *actually-used* dependencies — the technical foundation for the dependency-risk reachability features now in production at Endor Labs.

04 2019.02 – 2020.01
Post-doctoral Researcher — INESC-ID Lisbon + CMU Pittsburgh
Pivoted from program repair to **security analysis at scale**: built the largest empirical SAST evaluation of its kind — 9 static-analysis tools benchmarked across 47k smart contracts, plus the open-source SmartBugs framework that runs the pipeline. Also delivered the largest comparative study of automated program-repair tools to date (2,141 real bugs) and CI/CD-integrated patch synthesis driven by static-analysis signals.

§02 Education

01 2015 – 2018
Ph.D., Computer Science · Researcher — INRIA Lille, U. Lille / MSR
Where the program-analysis career began: built static- and dynamic-analysis infrastructure for **automatic patch generation** at runtime for Java and JavaScript. Open-sourced **Nopol**, **NPEFix**, and **Dynamoth** — standard baselines in the research community. Thesis: *From Runtime Failures to Patches*; co-supervised with Microsoft Research. ★ **Best PhD Thesis** (runner-up), French national software engineering research group, 2018.

02 2013 – 2015
M.Sc., Computer Science · *cum laude* — University of Lille
Specialization: Software Engineering and Program Analysis.

03 2012 – 2013
University Exchange — University of Luxembourg
Majoring in Computer Science

04 2010 – 2013
Bachelor, Computer Science · *cum laude* — Institut Paul Lambin, Brussels
Majoring in Computer Science

§03 Open Source

SHIPPED, MAINTAINED, IN PRODUCTION

01 Spoon

JAVA · STATIC ANALYSIS & TRANSFORMATION

Core contributor to a Java parsing/analysis/transformation library — comparable in scope to CodeQL's extensible query model for Java. Used in production at multiple research labs and companies for AST queries, **call-graph and reachability analysis**, and code rewriting.

1.5k+
DEPENDENT OSS PROJECTS

02 Anonymous GitHub

OSS · GITHUB APIS AT SCALE

Production web service that anonymizes GitHub repositories on demand — proxies the GitHub API, rewrites identifying metadata, and serves results through a streaming pipeline. Operates at scale: **213k unique visitors and 12.5M requests per month**, handling rate limits, auth, and large monorepos in production.

12.5M
REQUESTS / MONTH

03 docker-parfum

TREE-SITTER · MULTI-LANGUAGE AUTOFIX

Static analyzer and autofix tool for Dockerfiles. Parses Dockerfile + embedded shell into a unified AST via Tree-sitter, resolves package operations across image layers, and rewrites the AST to fix common security and efficiency anti-patterns automatically.

AST
CROSS-LANGUAGE AUTOFIX

04 EnergiBridge

CROSS-PLATFORM · PERFORMANCE INSTRUMENTATION

Cross-platform measurement library for software energy consumption (CPU, GPU, RAM). Wraps OS-specific performance counters behind a uniform API across Linux, macOS, and Windows. Used as research instrumentation for software-sustainability benchmarking.

36 ★
OPEN SOURCE · MIT

§04 Tooling & Languages

PRODUCTION EXPERIENCE

STATIC ANALYSIS	Tree-sitter · CodeQL · Semgrep · Spoon · call-graph / reachability analysis
SUPPLY CHAIN	Maven · Gradle · npm · PyPI · OSV · SBOM (CycloneDX, SPDX) · reachability-based SCA
CONTAINERS	Docker · Tree-sitter for Dockerfile parsing
PLATFORM	GitHub APIs · GitHub Actions · CI/CD · LLMs (production prompting + eval)
LANGUAGES	**Java** expert · **Python** expert · **TypeScript** expert · **Go** expert · **Rust** working

§05 Related Publications

FULL LIST AT DURIEUX.ME · 41 PAPERS · 3.8K+ CITES

01	ICSE'24	**Breaking the Silence: Threats of Using LLMs in Software Engineering.** Risks of LLM-generated code · 176 cites
02	ICSE'24	**Empirical Study of the Docker Smells Impact on the Image Size.** Container supply-chain attack surface · 22 cites
03	TOSEM'22	**Coverage-Based Debloating for Java Bytecode.** Reachability-based attack-surface reduction · 45 cites
04	FSE'21	**A Longitudinal Analysis of Bloated Java Dependencies.** ★ Best Paper · Foundation for reachability-based SCA · 75 cites
05	TSE'21	**Automated Classification of Overfitting Patches.** ML over static-analysis features · 118 cites
06	ICSE'20	**Empirical Review of Automated Analysis Tools on 47,587 Ethereum Smart Contracts.** Ethereum SAST benchmark · 667 cites
07	FSE'19	**Empirical Review of Java Program Repair Tools on 2,141 Bugs.** ★ Best Paper · Largest comparative repair-tool study · 175 cites
08	TSE'16	**Nopol: Automatic Repair of Conditional Statement Bugs in Java Programs.** Foundational program-repair work · 568 cites