# Eric Walkingshaw

eric@walkingshaw.net
https://eric.walkingshaw.net

## Overview

Compiler engineer applying deep knowledge from 20+ years in software engineering and programming languages research. Expert at programming language design and implementation, including compilers, interpreters, type systems, program analyses, and human factors. Can tackle any language-oriented problem. Track record of building high-quality software in industry and research settings. Excellent writer. Experienced teacher, mentor, and public speaker.

## Experience

**Senior Software Engineer, Elemental Cognition** 2/2023–8/2024

Lead engineer for the first stage of the compiler for Cogent, a knowledge representation language for trustable, explainable hybrid AI systems. The first stage:

- Lexes, parses, and disambiguates (non-trivial for Cogent) concrete syntax.
- Performs static analyses, e.g. type checking, inference, reference resolution.
- Provides language services for the IDE and LLM-based editing assistance agents.
- Generates the intermediate representation used by the second stage.

Wrote and maintained the Cogent language reference.

Collaborated to improve compiler and language integration with LLMs to improve the accuracy of agents that generate and repair Cogent models.

**Software Engineer, Savant Power** 7/2021–11/2022

Developed smart/green energy software systems in Rust and Objective-C.

- Led design and implementation of scriptable simulated energy systems.
- Integrated Tesla Powerwall with the Savant Power host system.

Major contributor to Frame, a DSL based on Harel's statecharts.

- Led design and implementation of Rust code generator and runtime system.

**Assistant Professor, Oregon State University** 9/2014–6/2021

40+ academic publications in programming languages and software engineering.

Won $1.6M, 4.5-year grant to improve robustness of highly configurable systems.

- Designed and implemented the DSL that was our core technical contribution.
- Led a team that delivered all software on-time and passed all evaluations.

Designed and taught courses on PL theory, functional programming, modularity, computer architecture, assembly programming, and data structures. Earned consistently excellent course evaluations.

Organized, chaired, and obtained funding for events at international conferences.

Peer-reviewed and made publishing/funding recommendations for conferences, journals, and government agencies.

## Previous experience

| | |
|---|---|
| Visiting Researcher, University of Marburg, Germany | 7/2013–9/2014 |
| Instructor, Oregon State University | 1/2013–6/2013 |
| Graduate Research Assistant, Oregon State University | 9/2007–6/2013 |
| Software Developer, Institute for Systems Biology, Seattle | 10/2006–6/2007 |
| Software Developer, Teranode, Seattle | 1/2005–6/2006 |
| Student Software Developer, Applied Physics Lab, UW | 5/2003–12/2004 |

## Education

**Ph.D. Computer Science**, 2013
Oregon State University

**M.S. Computer Science**, 2011
Oregon State University

**B.S. Computer Science + minor Applied Math**, 2006
University of Washington

**B.A. English**, 2006
University of Washington

**A.A.**, 2002
Edmonds Community College

## Skills and expertise

### Software engineering

- Expert functional programmer.
- Produce clean, modular, tested, reusable code in any paradigm.
- Working knowledge of dozens of languages in every paradigm. Can learn any language fast.

### Programming languages

- Design and implement DSLs, type systems, compilers, and associated infrastructure.
- Define and prove safety properties of programs/languages.
- Apply human-factors research to improve usability.

### Communication

- Excellent writer.
- Experienced public speaker.
- Effective mentor and manager.
- Cross-disciplinary collaborator.

### Languages and tools

- *Expert:* Rust, Scala, Haskell, Java, LaTeX, Git, Linux.
- *Experienced:* C, Objective-C, relational DBs + SQL, Coq, SML.
- *Knowledgeable:* JS, Python, Ruby, OCaml, many more…