Dr Alex Potanin

Associate Professor, School of Computing
College of Systems and Society, Australian National University
Adjunct Professor, S3D, Carnegie Mellon University
Senior Visiting Fellow, Trustworthy Systems, UNSW Sydney

https://potanin.github.io/ alex.potanin@anu.edu.au

At a Glance

- **Track Record:** 100+ publications across two decades, 6 distinguished/best paper awards, 80+ students supervised at all levels, $1.7M+ in research funding
- **Research Areas:** Cyber Security (Capabilities, Language Security), Software Engineering, Programming Language Design & Implementation, Type Systems (Ownership, Immutability), Software Verification, Quantum Computing Verification, Embedded Systems & Edge AI
- **Leadership:** Chair of SPLASH Steering Committee (2024–2026), Treasurer & Member at Large of ACM SIGPLAN Executive Committee (2025–2027), General Chair of SPLASH/OOPSLA 2022, Research Committee Co-Chair of OOPSLA 2024, PC Chair of APLAS 2025, Permanent Member of IFIP WG 2.4
- **Education:** BSc (Mathematics & Computer Science), BSc Hons, PhD — *Victoria University of Wellington, NZ*
- **Memberships:** Senior Member of the ACM, Member of Engineers Australia, Permanent Member of IFIP 2.4 Working Group on Software Implementation Technology

Research Impact

My ownership type system research directly influenced how the Rust programming language implements its ownership and lifetime system—one of Rust's defining features for memory safety. My foundational papers on generic ownership (OOPSLA 2006, 100+ citations) and generic immutability (FSE 2007, 100+ citations) showed how ownership and immutability guarantees can be incorporated "for free" into existing language designs. The "Performance of Open Source Applications" book cites my research that revolutionised performance evaluations in Talos and similar systems.

After a full-year sabbatical at Carnegie Mellon University, I co-created the Wyvern Programming Language with Professor Jonathan Aldrich—a novel general-purpose language designed with security and usability as primary goals. The CUE configuration language, used widely within Alibaba's cloud, based its module system on Wyvern's design. Scala drew on our Wyvern research both for its path-dependent types and type member mechanisms and, more recently, for its adoption of the capabilities-and-effects combination that Wyvern pioneered. Wyvern produced numerous publications including work on type-specific languages (ECOOP 2014 Distinguished Paper) and decidable typing for type members (POPL 2020).

My **cyber security** research spans capability-based systems, language-level security mechanisms, and vulnerability discovery. My work uncovered critical vulnerabilities in Java-based web servers (Jenkins, Tomcat, JBoss) and .NET equivalents, published at ECOOP 2017 (Distinguished Artifact Award) and supported by $50,000 USD Oracle Labs funding. Wyvern's capability-based module system provides authority-safe security guarantees, and my current research explores CHERI capabilities hardware security and secure-by-construction software development approaches.

Most recently, I have established a significant research programme in **quantum program verification**, applying formal methods expertise from classical software to the rapidly emerging quantum computing domain. This work has already produced publications at top venues: embedding quantum program verification into Dafny (OOPSLA 2025) and validating quantum state preparation programs (ESOP 2026, Distinguished Paper Award). I am supervising multiple students in this area across ANU and Iowa State University, and collaborating with Assistant Professor Liyi Li on

quantum symbolic execution and distributed quantum computation. This represents a natural and impactful extension of my verification and language design expertise into one of computing's most important frontiers.

I am one of the leading experts internationally in the fields of object ownership (restricting access using graph theoretical properties) and immutability (precise definition of when objects are allowed to change). I have an international research track record in secure programming language design and type systems.

In 2008–2012, I led the Unification of Immutability and Ownership project, which resulted in the Royal Society of NZ (equivalent of NSF in the US) Marsden Grant and 1 PhD and 3 Master's thesis students.

My research reputation is acknowledged by a past invitation to the Shonan Workshop and a full membership of the invitation-only IFIP Working Group 2.4, where a small number of leading international researchers present and discuss their latest work in an intensive week of interactions.

I have nearly 100 quality-assured publications as listed on my web page. I supervised over 80 students at different levels in the past, including 17 current ones (1 Postdoc, 11 PhD, 3 Honours, 2 research students), as also listed on my web page.

Selected Award-Winning & High-Impact Publications

1. Liyi Li, Anshu Sharma, Zoukarneini Difaizi Tagba, Sean Frett, and Alex Potanin. *Validating Quantum State Preparation Programs*. ESOP 2026. **ETAPS/ESOP 2026 Distinguished Paper Award.**
2. Amos Robinson, Alex Potanin. *Pipit on the Post: Proving Pre- and Post-Conditions of Reactive Systems*. ECOOP 2024. **ECOOP 2024 Distinguished Artifact Award.**
3. Tobias Runge, Alex Potanin, Thomas Thum, and Ina Schaefer. *Traits: Correctness-by-Construction for Free*. FORTE 2022. **FORTE 2022 Best Paper Award.**
4. Jens Dietrich, Kamil Jezek, Shawn Rasheed, Amjed Tahir, Alex Potanin. *EvilPickles: DoS Attacks Based on Object-Graph Engineering*. ECOOP 2017. **ECOOP 2017 Distinguished Artifact Award.**
5. Cyrus Omar, Darya Melicher, Ligia Nistor, Benjamin Chung, Alex Potanin, and Jonathan Aldrich. *Safely Composable Type-Specific Languages*. ECOOP 2014. **ECOOP 2014 Distinguished Paper Award.**
6. Yoav Zibin, Alex Potanin, Mahmood Ali, Shay Artzi, Adam Kiezun, and Michael D. Ernst. *Object and Reference Immutability using Java Generics*. FSE 2007. **ESEC/FSE 2007 ACM SIGSOFT Distinguished Paper Award.** 100+ citations.
7. Alex Potanin, James Noble, Dave Clarke, and Robert Biddle. *Generic Ownership for Generic Java*. OOPSLA 2006. 100+ citations. Directly influenced the Rust programming language.
8. Julian Mackay, Alex Potanin, Jonathan Aldrich, and Lindsay Groves. *Decidable Subtyping for Path Dependent Types*. POPL 2020. Reusable Artifact badge.
9. Feifei Cheng, Sushen Vangeepuram, Henry Allard, Seyed M.R. Jafari, Alex Potanin, and Liyi Li. *Embedding Quantum Program Verification into Dafny*. OOPSLA 2025.
10. Alex Potanin, Johan Ostlund, Yoav Zibin, Michael D. Ernst. *Immutability*. Book chapter in Aliasing in Object-Oriented Programming, LNCS 7850, Springer, 2013.
    - This is a particularly special piece of work in the prestigious Lecture Notes on Computer Science (LNCS) series by Springer. Both in terms of the review process and the work involved, this book chapter was essentially a typical top journal article. This work surveys the field as well as consolidates four major papers by the authors, which makes it of significant value. I took the lead author role, writing almost all the text. All the chapters in the book underwent a thorough review process by the experts in our field so the quality assurance was both in terms of invitation to write and in terms of the review process. I am very proud of this piece of work, as it is the pinnacle publication summarising my findings

because of the now completed Marsden Grant.

11. Alex Potanin, James Noble, Marcus Frean, and Robert Biddle. *Scale-free Geometry in Object-Oriented Programs*. Communications of the ACM, 2005. 300+ citations.
12. Alex Potanin, Monique Damitio and James Noble. *Are Your Incoming Aliases Really Necessary? Counting the Cost of Object Ownership*. ICSE 2013. Pp 742–751.
    - This paper challenges a common assumption by programmers that efficiency trumps design. We challenged the "Iterator" design pattern to avoid breaking the encapsulation of the collection that iterators traverse. We justify our claim by an extensive and thorough study of the variations of encapsulation-friendly approaches applied to real world. While such advantages were known for the last 10–15 years, we were the first to provide an extensive study that actually demonstrated it.

Research Funding

Period	Funder	Grant	Amount
2026 – 2027	Anthropic	Claude API Credits for Teaching at SOCO (shared with Dr Ben Swift)	$500,000 (in-kind)
2025 – 2026	Skykraft	Research Consulting	$30,000
2024	DVCRI	University Strategic Fund towards Centre of Excellence on Trustworthy Systems	$36,000
2022	ANU	Startup Package Funding	$500,000
2021 – 2023	SHEADI	Faculty Strategic Initiative PhD Scholarship	$100,000
2020 – 2021	Robonomics Network	Research Grant (Nitrate Sensors)	$72,000
2019	Kyoto University	Visiting Professor Scholarship	$20,000
2017 – 2018	Oracle Corporation	Research Grant (Language Security)	$70,000
2013	Carnegie Mellon University	Visiting Researcher Funding (DARPA Lablet)	$25,000
2012	Mozilla Foundation	Research Grant	$15,000
2009 – 2011	Royal Society of NZ	Marsden Fast Start Grant	$300,000
2009	RSNZ	ISAT Grant	$5,420
2007 – 2016	VUW	University Research Fund (multiple rounds)	$47,000
		Total	**$1,720,420**

Students and Mentoring

Over 80 students supervised across two decades at ANU, VUW, CMU, KIT, and Iowa State. My former students and collaborators have gone on to leading positions:

- Radu Muschevici — Assistant Professor, University of Nottingham Malaysia
- Amos Robinson — Computer Scientist, Microsoft
- Justin Lubin — PhD candidate, UC Berkeley
- Darya Melicher — Google
- Paley Li — Postdoc, Czech Technical University (with Jan Vitek)
- Hannes Mehnert — Co-founder, Robur cooperative
- Felix Shi — Senior Security Architect, Xero
- Garming Sam — Security Platform Engineer, Westpac NZ
- Constantine Dymnikov — Solutions Architect, NZ Ministry of Justice

Current Students (17)

Student	Level	Topic
Yan Liu*	Postdoc (ANU)	Research Fellow, 2025–2027
Sasha Pak*	PhD (ANU)	Rust Made Easier (with Ilya Sergey, Fabian Mühlböck)
Abolfazl Sharifi*	PhD (ANU)	Verification of Concurrent Data Structures in Rust using Iris
Edwin Singh*	PhD (ANU, PT)	Why Language Designers Do Modules The Way They Do
Jack Hackshaw*	Honours (ANU)	Pancake in Loom in Lean
Billy Jaffray*	Honours (ANU)	Verification in Rust using Loom
Rishita Sarkar*	BAC R&D (ANU)	Quantum Computing Testing
David Young*	PhD (Kansas, PT)	PCM's for Abstracted Layouts across Multiple Fields: from Heaps to Quantum
Libby Boas*	PhB ASC (ANU)	Quantum Computing Distributed Computation
Haoyu Wu	PhD (ANU)	Nomnom Type Checking (with Fabian Mühlböck)
Carlo Zancanaro	PhD (ANU)	Gradual Typing and Type Polymorphism
Julia Groß	PhD (ANU, PT)	Energy Trading in Blockchains (with Sid Chau, Ramesh Rayudu)
Feifei Cheng	PhD (Iowa State)	Quantum Computing Symbolic Execution (with Liyi Li)
Maximillian Kodetzki	PhD (KIT)	X-by-Construction (with Ina Schaefer)
Jakob Jerebica	PhD (KIT)	Formalisation of Pancake by Construction with Real World Driver Examples (with Ina Schaefer, Michael Norrish)
Alyssa Herald	Honours (ANU)	Security of Capabilities in CHERI
Charlotte Fulham	Eng ASC (ANU)	Fiducia to Microkit Bridge Implementation

* Primary supervisor.

Selected Completed Students

- Iko-Ojo Simon (PhD, ANU, 2023–2025) — Algorithmic Debt → Postdoc, University of São Paulo, Brazil
- Amos Robinson (Postdoc, ANU, 2023–2025) — Higher Level Invariants with Lark → Microsoft
- Tobias Runge (PhD, KIT, 2019–2022) — CbC with Ownership and Traits (with Ina Schaefer)
- Julian Mackay (PhD, VUW, 2015–2019) — Wyvern Type Members (with Jonathan Aldrich)
- Darya Melicher (PhD, CMU, 2013–2019) — Wyvern Modules (with Jonathan Aldrich) → Google
- Ahmed Khalifa (PhD, VUW, 2010–2013) — Ownership and Immutability in the Real World
- Garming Sam (BE Hons, VUW, 2015) — Refactorings in Rust → Westpac NZ
- Radu Muschevici (MSc, VUW, 2007–2008) — Multimethods → Asst. Prof., U. Nottingham Malaysia
- Constantine Dymnikov (MSc, VUW, 2011) — Modular Ownership Inference → NZ MoJ
- Paley Li (BSc Hons, VUW, 2008) — Ownership and Immutability → Postdoc, Czech Technical U.

As Associate Director of HDR at ANU School of Computing (2023–2025), I managed admissions, supervision, and progress for over 200 PhD students, created a monitoring app to proactively track student progress, and revamped the admissions and scholarship process.

Employment Record

2022 – Present	Associate Professor, Australian National University
2023 – Present	Adjunct Professor, S3D, Carnegie Mellon University, USA
2026 – Present	Senior Visiting Fellow, Trustworthy Systems, School of Computer Science and Engineering, UNSW Sydney
2025	Research Consultant, Usability Dynamics Inc, NC, USA
2025	Research Consultant, Skykraft, Canberra, Australia
2023 – 2025	Associate Director, HDR, School of Computing, ANU
2022 – 2025	Adjunct Professor, ECS, VUW, NZ
2022	Deputy Head of School, Victoria University of Wellington
2021 – 2022	Associate Dean (Students), Victoria University of Wellington
2018 – 2022	Associate Professor, Victoria University of Wellington
2019/20 (sabbatical)	Associate Professor, Kyoto University, Japan
2010 – 2017	Senior Lecturer, Victoria University of Wellington
2013 (sabbatical)	Research Scholar, Carnegie Mellon University
2006 – 2009	Lecturer, Victoria University of Wellington
2005 – 2006	Software Engineer, Innaworks Limited
2004	Visiting Researcher, Purdue University
2001 – 2003	Software Engineer, Catalyst Systems Limited (part-time)

Professional Memberships

2024 – 2027	Member at Large (and Treasurer), *ACM SIGPLAN Executive Committee*
2024 – 2026	Chair, *SPLASH Steering Committee*
2023 – Present	Member of Engineers Australia
2002 – Present	Member of the ACM (currently: Senior Member)
2018 – 2022	Member of Engineering New Zealand
2010 – 2014	Member of CORE (CS Departments of Australia/NZ) Executive

Awards

- 2026 – ESOP/ETAPS 2026 Distinguished Paper Award
- 2024 – ECOOP 2024 Distinguished Artifact Award
- 2022 – FORTE 2022 Best Paper Award
- 2017 – ECOOP 2017 Distinguished Artifact Award
- 2014 – ECOOP 2014 Distinguished Paper Award
- 2007 – ESEC/FSE ACM SIGSOFT Distinguished Paper Award
- 2005 – 2nd Prize and Judges Prize at ICFP Programming Contest 2005 and 2003
- 2004 – Claude McCarthy Fellowship; 2003 – J. L. Stewart Scholarship
- 2003 – 2nd Place, ACM International Research Competition (via OOPSLA 2002 SRC)
- 2002 – Freemasons Scholarship; 2001 – Datacom Scholarship in Computer Science
- 1997 – Finalist, George Soros International Mathematics Olympiad
- 1996 – Winner, Russian City Tournament in Mathematics

Community Leadership and Service

Conference Leadership

- Chair, SPLASH Steering Committee, 2024–2026
- Member at Large and Treasurer, ACM SIGPLAN Executive Committee, 2025–2027
- General Chair, SPLASH/OOPSLA 2022 (combined with APLAS 2022), Auckland, NZ
- Research Committee Co-Chair, OOPSLA 2024 (with Bur-Yuh Chang)
- PC Chair, APLAS 2025
- General Chair, APLAS 2018, Wellington, NZ
- Program Co-Chair, APSEC 2016 (with Gail Murphy)
- PC Co-Chair, CATS 2010 and 2011 (with Taso Viglas)
- General Co-Chair, ACSW 2009, Wellington, NZ
- General Chair, 15th PhD Workshop and Doctoral Symposium at ECOOP 2005
- Chair, NOOL Workshop 2015 and 2017
- Workshops Co-Chair, SPLASH 2017 and 2018

- Publications Chair, SPLASH 2015–2017; ICFP 2018
- Virtualisation Co-Chair, SPLASH 2020 and 2021
- SPLASH/OOPSLA Steering Committee Member from 2020 (Chair from 2024)
- SAPLING Steering Committee Member from 2019

Program Committee Member (selected)

PLDI 2025, FSE-IVR 2025, OOPSLA (2021 ERC, 2020, 2019, 2014, 2011 ERC), ECOOP (2023, 2022, 2019, 2016), ESOP 2021, APLAS (2021, 2016), GPCE (2020, 2019), PLDI 2015 (ERC), HotSoS 2019, PLDI SRC 2019, FTfJP 2016, NOOL 2016, IWACO (2024, 2021, 2011, 2007), FormaliSE (2023, 2022), Programming (2023, 2022), Onward! Papers 2023, FOOL 2014, ACSC (2010–2017).

University Service

Since I started at the ANU in June 2022, I have been the program convenor for the Bachelor of Engineering in Software Engineering. From 2023, I also took on the role of the Diversity, Belonging, Inclusion, and Equity Working Group Lead for the School of Computing. Finally, from 2023 to 2025, for 2.5 years, I was the Associate Director, HDR for the School of Computing — looking after our cohort of just over 200 PhD students.

As part of my role as the Associate Director of HDR, I have created a monitoring app to track the progress and conditions of all of our HDR students in the school. The app is being adopted by the core administrative staff and key academics. I have revamped and optimised the processes, establishing a clear admissions and scholarship process that takes diversity into account and clarifies the roles of primary supervisors to ensure continuity for all our HDR students. I ensured that each of the four clusters has regular and consistent HDR monitoring in place every six months to improve the HDR cohort experience proactively.

At the very start of 2022, while still at the VUW, I was employed as Deputy Head of School as an additional 0.4 FTE on top of my other admin role described next to help with the transitional period as we are split into four sub-schools. I have 15 academic staff as direct reports, ranging from Assistant Lecturer to Full Professor, in the areas of Software Engineering and Cyber-Security. I have managed to organise for all of them to conduct annual performance development plans for the year within a week of starting my role. One of the main reasons I have been asked to fill this role was because I managed to both accredit and revise our Bachelor of Engineering in Software Engineering during my tenure as Program Director while keeping all the academics involved both engaged and happy with the proposed changes and getting them through all the necessary academic proposal stages, including VUW Academic Board, and eventual approval by the NZ national CUAP committee, as well as the scheduled accreditation by the Engineering NZ.

In 2021 and 2022 at the VUW, I was employed as 0.4 FTE Associate Dean (Students) at the Faculty of Engineering, looking after approximately 1600 EFTS (or 2000 actual students, as some are part-time) in the School of Engineering and Computer Science and the School of Mathematics and Statistics. One of my first achievements was the creation of the Student Engagement System that allows me to proactively monitor the progress of ∼2000 students in our faculty: picking up those who are asking for extensions, missing deadlines, or not coping with the courses that they are currently enrolled in. With the help of the Faculty Office staff, we can reach out early to them before they become non-engaged and place appropriate support measures in place, ranging from connecting them with counselling services to offering withdrawals and re-weighting of the assessments. Alongside the Student Engagement System (SES), I have also centralised our assessment scheduling, which is now done proactively before each trimester begins because the academic staff saw the clear benefits of the SES in picking up struggling students early. Finally, our extension requests are now integrated into our Assessment System to ensure consistency (especially in times of COVID) and the ability to easily monitor the student's progress. One of the examples of my numerous external engagement achievements was the negotiation in January 2022 of a special pathway from a regional polytechnic (WITT) that allows the students doing their first year locally as BEngTech in the regional NZ to then do our summer papers in COMP remotely in the summer to join alongside our 2nd year students to be able to complete BE at VUW — easing the transition from the regions to the far away city University study.

In 2017–2018, while at the VUW, I served as Program Director and led the Software Group, delivering the SWEN, NWEN, and CYBR majors in BE and the HSWD major in Bachelor of Health, which involves working with over 20 academic and other teaching staff and assisting the Head of School in running these programs. A major part of this role is to enable staff to work productively and constructively together to deliver the best learning experience to students. In 2019, I moved to the Program Director (Science) role to lead the introduction of the AI & ML postgraduate program before leaving on a partial sabbatical in 2020.

As Postgraduate Coordinator from 2012 to 2016 while at the VUW, I led an interdisciplinary postgraduate research program with nearly 100 PhD and Master's students, chaired every PhD proposal meeting, and attended to all matters related to thesis students. I significantly clarified and improved the procedures for induction, reporting, handling of problem cases, PhD progression to full registration, Master's Thesis progress reporting and examinations at the Engineering Faculty during my term, which was my key responsibility. My work earned praise from the FGR and received a commendation from the BE Review Panel in 2016. When I finished, my responsibilities were handed over to three Associate Professors and a professional staff member.

In 2018, I was part of the Engineering New Zealand Accreditation Panel to re-accredit the University of Canterbury's Bachelor of Engineering with nine majors, and I represented the NZ Qualifications Authority to accredit a new Bachelor of Information Technology degree proposed by Whitireia and WelTec Polytechnics. I remained an annual monitor for the "W and W" IT programs at postgraduate and undergraduate levels for 2020, 2021, and 2022.

Research Themes

Cyber Security

Capability-based security, authority-safe module systems, and preventing command injection and other attacks through language design. EvilPickles (ECOOP 2017, Distinguished Artifact) uncovered critical security flaws in Java/JBoss/Jenkins and .NET servers, leading Oracle and Microsoft to fix the vulnerabilities within 6 months—supported by $50,000 USD Oracle Labs funding. Research on CHERI capabilities hardware security (Honours student Alyssa Herald). Secure-by-construction software development through the Wyvern language's capability-based module system (ECOOP 2017, HotSoS 2018) and type-specific languages to fight injection attacks (HotSoS 2014).

Embedded Systems & Edge AI

Trustworthy systems security for embedded and space platforms, including consulting for Skykraft and other industry partners. Exploring secure Edge AI deployment on NVIDIA Jetson Orin.

Quantum Computing

A major new research direction applying formal methods to quantum computing. Developed the first embedding of quantum program verification into the Dafny verification framework (OOPSLA 2025), enabling developers to write and verify quantum programs using familiar classical tools. Our work on validating quantum state preparation programs (ESOP 2026, Distinguished Paper) addresses a critical challenge in ensuring correctness of quantum circuits. Current projects include quantum symbolic execution (PhD student Feifei Cheng at Iowa State), quantum computing testing (Rishita Sarkar), and distributed quantum computation (Libby Boas), with collaborator Liyi Li (Iowa State).

The Wyvern Language

A secure general-purpose programming language using object capabilities and effects, co-developed with Jonathan Aldrich at CMU over 2013–2023. The CUE configuration language based its module system on Wyvern. Scala drew on our Wyvern research for its path-dependent types and type members, and more recently adopted the capabilities-and-effects combination that Wyvern pioneered. Produced 12+ co-authored papers (`http://wyvernlang.github.io/`).

Ownership & Immutability

Pioneered Generic Ownership, showing how type polymorphism provides ownership support. This approach was adopted by the Rust programming language as "lifetime parameters". Led the Marsden-funded project (2008–2012) unifying ownership and immutability in a single mathematical framework.

Software Verification

Proving pre- and post-conditions of reactive systems with the Pipit framework (ECOOP 2024, Distinguished Artifact), and correct-by-construction programming approaches including traits (FORTE 2022 Best Paper) and flexible CbC (LMCS 2023). Research Fellow Yan Liu is working on trustworthy systems verification.

Software Engineering

Empirical studies of software development practices, developer tools and productivity, API usability, and evidence-based approaches to improving software quality and developer experience. Supervised research on reproducibility debt in scientific software (JSS 2025, FSE-IVR 2024, SPLASH 2025 poster), algorithm debt in

ML/DL systems (ESE 2025, ICSME 2024), and automated refactoring in Rust (ACSC 2017, SPLASH 2025 posters). Led the Software Engineering program at VUW (2017–2018) and currently convenes BE Software Engineering at ANU.

Collaborations

- Professor Jonathan Aldrich, CMU, USA (Wyvern, capabilities, type members)
- Professor Ina Schaefer, KIT, Germany (correct-by-construction, information flow)
- Assistant Professor Liyi Li, Iowa State University, USA (quantum program verification, symbolic execution)
- Professor Atsushi Igarashi, Kyoto University, Japan (ownership verification)
- Professor Ilya Sergey, NUS, Singapore (Rust, deductive synthesis)

Referees

- Professor Peter Müller, *ACM Fellow*, ETH Zurich, Switzerland
- Professor Steve Blackburn, *ACM Fellow*, ANU/Google DeepMind
- Professor Jeff Foster, *ACM Fellow*, Tufts University, USA
- Professor David Lo, *ACM Fellow*, Singapore Management University
- Professor Anders Møller, Aarhus University, Denmark
- Professor Sophia Drossopoulou, Imperial College London, UK
- Professor Jan Vitek, Northeastern University, USA

Teaching

In my time at the ANU, I have created a new 2$^{\text{nd}}$-year Software Engineering course that involves a group project consolidating the programming skills the students have and preparing them for the modern real-world environment. As the outcome of the course, each student contributes a Pull Request to an acceptable medium to large established open-source project.

When I was at the VUW, I got a consistent "overall effectiveness" score of around 1.6 (on a scale of 1 to 5, with one being "outstanding"). I taught classes in a variety of sizes (5–25 at the 4th year level, 60–120 at the 3rd year level, 150–300 at the 2nd year level, 300–500 at the 1st year level) and always received positive feedback with a "perfect" 1.0 score for "Attitude towards students". My overall teaching evaluation for the newly developed course on Software Development for Mobile Platforms in 2018 was the best possible 1.0. In several courses in 2020, I had close to 1.0 or equal to 1.0 (e.g., in SWEN 430 (Compiler Engineering) in 2020), teaching evaluations, despite it being the COVID year.

In 2019, I led the introduction of the postgraduate Artificial Intelligence and Machine Learning program at VUW as part of my role as Program Director (Science). This involved staff management and conflict resolution in a timely fashion to meet the required government regulation deadlines.

In 2017, I led a review of Software Engineering (the largest major in Bachelor of Engineering at the VUW) involving all the academic staff in Engineering and Computer Science. I led a discussion by over 20 software-related academic staff members over the first half of 2017, which led to a proposal I wrote that the VUW Academic Board approved in the middle of 2017. This is the first and most significant overhaul of the Software Engineering at the VUW since its creation 8 years ago. This will feed directly into the 5-yearly Engineering re-accreditation in 2017, which I also led as the Program Director for Software Engineering.

Teaching Innovation and Course Development

1. **COMP 3500/4500/8715 TechLauncher re-design at the ANU.** In 2025, I was tasked with a redesign of a capstone project course for Computing and Software Engineering graduates with 400 students. I changed the course to a scalable mode of operation able to run with 600+ student enrolments by focusing on the time-boxed prescribed model with teams formed around "the most available day" in their schedule and asking them to work in person in the shared "The Hive" space on campus with up to 15 "pods" available for the teams. At the same time, a tutor can oversee up to four teams simultaneously. We have reduced the tutor budget by half while increasing the student satisfaction and client satisfaction with deliverables significantly, and still fulfilling the relevant ACS and EA capstone requirements. *COMP3500 in 2025 S1 had a perfect 100% overall satisfaction score from the students.*
2. **COMP 2120 re-design at the ANU.** I took over the 2nd-year "Software Engineering" course. I revised both the assessment and content to create a modern, group project-oriented course with the

ultimate goal of contributing a pull request to a large open-source project by every one of the 50+ teams.

3. **SWEN 325 at the VUW** — the creation of a brand-new course in 2018. To fill the gap in teaching our students high-level frameworks and software architecture as well as the in-demand skills in the development of software for mobile platforms and IoT, I developed from scratch a brand new final year undergraduate course that covers architecture for mobile systems, UX for mobile, being able to learn and work with multiple modern frameworks such as React Native and Ionic, and writing apps for and interacting with IoT devices. The course was very successful, with an overall teaching evaluation of me being 1.0 and a course overall evaluation of 1.8.
4. **COMP 361 Full Course Redesign in 2014 at the VUW.** I have addressed low enrolments in a 3rd-year "Algorithms" course (formerly known as COMP 303) by reinventing it as COMP361 with a readjusted aim and course objectives to make it more practical for modern software engineers. The redesign was a success: increasing the enrolments tenfold from 5–6 for COMP303 to around 60 from 2014 onwards. I am the sole person responsible for the course and its design and delivery. The final 2014 overall course feedback was 2.0, which is quite reasonable given that it originated from the smallest 3rd-year COMP course and has since become the second-largest after COMP 307 (Introduction to Artificial Intelligence) and the largest in the second Trimester for the 3rd-year COMP.
5. **SWEN 302 Full Course Redesign in 2014 at the VUW.** In 2013, SWEN302 ended up in need of a "rescue" when it received one of the worst course evaluations in ECS history: 4.0. I volunteered for the task as the course coordinator. I completely revamped the way this agile project course is run by bringing in mostly internal, very highly involved clients and introducing "time boxing": all the students were asked to pick one day every week that they had the least amount of lectures. The teams were then formed for those days of the week, and the requirement was that students should work only 8 hours on the designated day (in addition to 2 hours spent in a lecture). This worked wonders as it allowed students to control their time on one hand and to avoid any team members "slacking" as it was clear who had to be around when. More importantly, this made all the students learn the most essential aspect of agile management: being aware of the time spent on different parts of the project and being able to make accurate estimates at all times. The course received anecdotally positive feedback, and the overall score decreased significantly (to 2.0 in 2014).
6. **COMP 261 at the VUW.** Together with Dr Peter Andreae in 2010, we developed a brand-new course on Algorithms to fill the need for our up-and-coming Computer Graphics program, as well as providing an elective foundational course on Computer Algorithms for our Bachelor of Engineering students. The course was designed from the ground up to fit into the new 15-point system to keep the workload for students manageable. I put together a custom textbook with the help of the Pearson publisher that we use to teach COMP261. For the rest of my tenure at the VUW, COMP261 remained the most popular course in the undergraduate computer science degree.
7. **SWEN 430 Full Course Redesign in 2010 at the VUW.** Together with Dr David Pearce in 2010, we redesigned our graduate course to offer a strong course on compiler engineering and advanced Java programming. This course was a success: Innaworks and other Wellington start-ups sought after the students who took this course. In 2012, it was rated as the most useful course in course evaluations of the 4th-year Software Engineering courses.

Courses Taught

NB! 1st-year courses are usually in the 400 or so size range. The 2nd-year level is in the 300 or so range. The 3rd-year level is in the 100–200 or so range. The 4th-year level is around 20–30 people. Tech Launcher is 400 students (or 60 teams with external clients).

At the ANU:

1. COMP2100/COMP6442 Semester 2 of 2026 (Software Construction)
2. COMP3500/4500/8715 Tech Launcher Capstone Project Semester 1 of 2026
3. COMP3500/4500/8715 Tech Launcher Capstone Project in 2025
4. COMP 2120 Semester 2 of 2024 (Software Engineering)
5. COMP 2100 Semester 2 of 2024 (Software Construction) — convening only
6. COMP 4130 Semester 1 of 2024 (Managing Software Quality and Process) — convening only
7. COMP 2120 Semester 2 of 2023 (Software Engineering)
8. COMP 2120 Semester 2 of 2022 (Software Engineering)

At the VUW:

9. COMP 103 T3 (Jan/Feb) 2022 (Introduction to Data Structures and Algorithms)

10. COMP 102 T3 (Summer, Half) 2021 Nov/Dec (Introduction to Computer Program Design)
11. COMP 261 T1 (Second Half) 2021 (Algorithms and Data Structures)
12. COMP 361 T1 (First Half) 2021 (Design and Analysis of Algorithms)
13. COMP 103 T3 (Second Half) 2020 (Introduction to Data Structures and Algorithms)
14. SWEN 430 T2 (Second Half) 2020 (Compiler Engineering)
15. SWEN 325 T2 (Weeks 2, 3, 4) 2020 (Software Development for Mobile Platforms)
16. Kyoto University Graduate Seminars in Winter 2019/2020
17. SWEN 325 T2 2019 (Software Development for Mobile Platforms)
18. SWEN 325 T2 2018 (Software Development for Mobile Platforms)
19. COMP 361 T2 2017 (Design and Analysis of Algorithms)
20. COMP 261 T1 (Course Coordinator Only) 2017 (Algorithms and Data Structures)
21. SWEN 221 T1 (Course Coordinator Only) 2017 (Software Development)
22. COMP 103 T2 (Coordinator Only) 2016 (Intro to Data Structures and Algorithms)
23. COMP 361 T2 2016 (Design and Analysis of Algorithms)
24. COMP 261 T1 (9 of 12 weeks) 2016 (Algorithms and Data Structures)
25. COMP 361 T2 2015 (Design and Analysis of Algorithms)
26. SWEN 302 T2 (Essays and Course Coordinator Only) 2015 (Agile Methods)
27. COMP 261 T1 (First Eight Weeks) 2015 (Algorithms and Data Structures)
28. COMP 361 T2 2014 (Design and Analysis of Algorithms)
29. SWEN 302 T2 (Coordinator Only) 2014 (Agile Methods)
30. COMP 261 T1 (Second Half) 2014 (Algorithms and Data Structures)
31. SWEN 430 T1 (First Half) 2014 (Compiler Engineering)
32. ENGR 123 T2 (Labs Only) 2015 (Engineering Mathematics with Logic and Statistics)
33. COMP303 T2 2012 (Design and Analysis of Algorithms)
34. SWEN430 T2 2012 (Compiler Engineering)
35. COMP103 T1 2012 (Introduction to Data Structures and Algorithms)
36. COMP303 T2 2011 (Design and Analysis of Algorithms)
37. COMP261 T2 (Last 8 weeks) 2011 (Algorithms and Data Structures)
38. SWEN430 T2 (First 4 weeks) 2011 (Compiler Engineering)
39. SWEN423 T1 (First 2 weeks and last 4 weeks) 2011 (OO Paradigms)
40. COMP303 T2 2010 (Design and Analysis of Algorithms)
41. COMP261 T2 (2nd half) 2010 (Algorithms and Data Structures)
42. SWEN430 T2 (1st half) 2010 (Compiler Engineering)
43. COMP303 T2 (2nd half) 2009 (Design and Analysis of Algorithms)
44. COMP431 T2 (2nd half) 2009 (Compilers)
45. COMP304 T1 (1st half) 2009 (Programming Languages)
46. COMP462 T1 (1st half) 2009 (Object-Oriented Paradigms)
47. COMP303 T2 2008 (Design and Analysis of Algorithms)
48. COMP431 T1 2008 (Compilers)
49. COMP103 T2 (2nd half) 2007 (Introduction to Data Structures and Algorithms)
50. COMP471 T2 (1st half) 2007 (Special Topic: Compiler Technologies)
51. COMP462 T1 (1st half) 2007 (Object-Oriented Paradigms)
52. COMP103 T3 (2nd half) 2006 (Introduction to Data Structures and Algorithms)
53. COMP103 T2 (2nd half) 2006 (Introduction to Data Structures and Algorithms)
54. COMP101 T3 2005 (Introduction to Dynamic Web Development)
55. INET101 T3 2004 (Introduction to Internet Technology)

Full Publications List

Book Chapters

1. James Noble, Alex Potanin, Toby Murray, Mark S. Miller. *Abstract and Concrete Data Types vs Object Capabilities*. In P. Müller and Ina Schaefer (Eds.): Principled Software Development. Springer, 2018.
2. Alex Potanin, Johan Ostlund, Yoav Zibin, Michael D. Ernst. *Immutability*. In D. Clarke et al. (Eds.): Aliasing in Object-Oriented Programming, LNCS 7850, pp. 233–269. Springer, 2013.

Journal Papers

3. Emmanuel Iko-Ojo Simon, Chirath Hettiarachchi, Fatemeh Fard, Alex Potanin, Hanna Suominen. *A Survey of Algorithm Debt in Machine and Deep Learning Systems: Definition, Smells, and Future Work*. ACM Computing Surveys. Accepted for publication in March 2025.

4. Emmanuel Iko-Ojo Simon, Chirath Hettiarachchi, Fatemeh Fard, Alex Potanin, Hanna Suominen. *A Survey of Algorithm Debt in Machine and Deep Learning Systems: Definition, Smells, and Future Work.* Empirical Software Engineering. Springer. Accepted December 2025.
5. Zara Hassan, Christoph Treude, Michael Norrish, Graham Williams, and Alex Potanin. *Characterising Reproducibility Debt in Scientific Software: A Systematic Literature Review.* The Journal of Systems & Software. Volume 222, April 2025, 112327.
6. Tobias Runge, Tabea Bordis, Alex Potanin, Thomas Thum, Ina Schaefer. *Flexible Correct-by-Construction Programming.* Logical Methods in Computer Science. Volume 19, Issue 2, pp. 16:1–16:36. 2023.
7. Tobias Runge, Marco Servetto, Alex Potanin, Ina Schaefer. *Immutability and Encapsulation for Sound OO Information Flow Control.* ACM TOPLAS. Volume 45, Issue 1, Article No. 3 pp 1–35. 2022.
8. Isaac Oscar Gariano, Marco Servetto, Alex Potanin. *Using Capabilities for Strict Runtime Invariant Checking.* Science of Computer Programming, Volume 224, 2022.
9. Darya Melicher, Anlun Xu, Valerie Zhao, Alex Potanin, Jonathan Aldrich. *Bounded Abstract Effects.* ACM TOPLAS, Volume 44, Issue 1, March 2022.
10. Isaac Oscar Gariano, Marco Servetto, Alex Potanin, Hrshikesh Arora. *Iteratively Composing Statically Verified Traits.* EPTCS. Issue 299, Paper 7. 2019.
11. Chris Male, David Pearce, Alex Potanin, and Constantine Dymnikov. *Formalisation and Implementation of an Algorithm for Bytecode Verification of @NonNull Types.* Science of Computer Programming. Volume 76, Issue 7, pp. 587–608, 2011.
12. Alex Potanin, James Noble, Dave Clarke, and Robert Biddle. *Featherweight Generic Confinement.* Journal of Functional Programming. Volume 16, Number 6, pp. 793–811, 2006.
13. Alex Potanin, James Noble, Marcus Frean, and Robert Biddle. *Scale-free Geometry in Object-Oriented Programs.* Communications of the ACM. Pages 99–103. May 2005.
14. Alex Potanin, James Noble, and Robert Biddle. *Checking Ownership and Confinement.* Concurrency and Computation: Practice and Experience. Volume 16, Issue 7, pp. 671–687, 2004.

Refereed Conference Papers

15. Liyi Li, Anshu Sharma, Zoukarneini Difaizi Tagba, Sean Frett, and Alex Potanin. *Validating Quantum State Preparation Programs.* ESOP 2026. **Distinguished Paper.**
16. Feifei Cheng, Sushen Vangeepuram, Henry Allard, Seyed M.R. Jafari, Alex Potanin, and Liyi Li. *Embedding Quantum Program Verification into Dafny.* OOPSLA 2025.
17. Maximilian Kodetzki, Tabea Bordis, Alex Potanin, Ina Schaefer. *X-by-Construction: Towards Ensuring Non-Functional Properties in by-Construction Engineering.* Onward! 2025.
18. Emmanuel Iko-Ojo Simon, Chirath Hettiarachchi, Alex Potanin, Hanna Suominen, and Fatemeh Fard. *Automated Detection of Algorithm Debt in Deep Learning Frameworks.* ICSME 2024.
19. David Young, Ziyi Yang, Ilya Sergey, Alex Potanin. *Higher-Order Specifications for Deductive Synthesis of Programs with Pointers.* ECOOP 2024.
20. Amos Robinson, Alex Potanin. *Pipit on the Post: Proving Pre- and Post-Conditions of Reactive Systems.* ECOOP 2024. **Distinguished Artifact.**
21. Zara Hassan, Christoph Treude, Michael Norrish, Graham Williams, Alex Potanin. *Reproducibility Debt: Challenges and Future Pathways.* FSE-IVR 2024.
22. Tobias Runge, Alexander Kittelmann, Marco Servetto, Alex Potanin, and Ina Schaefer. *Information Flow Control-by-Construction for an Object-Oriented Language.* SEFM 2022.
23. Tobias Runge, Alex Potanin, Thomas Thum, and Ina Schaefer. *Traits: Correctness-by-Construction for Free.* FORTE 2022. **Best Paper.**
24. Manish Singh, Lindsay Groves, and Alex Potanin. *A Relaxed Balanced Lock-free Binary Tree.* PDCAT 2020.
25. Julian Mackay, Alex Potanin, Jonathan Aldrich, and Lindsay Groves. *Syntactically Restricting Bounded Polymorphism for Decidable Subtyping.* APLAS 2020.
26. Julian Mackay, Alex Potanin, Jonathan Aldrich, and Lindsay Groves. *Decidable Subtyping for Path Dependent Types.* POPL 2020. Reusable Artifact badge.
27. Aaron Craig, Alex Potanin, Lindsay Groves and Jonathan Aldrich. *Capabilities: Effects for Free.* ICFEM 2018. Pp 231–247. Springer.
28. Jens Dietrich, Kamil Jezek, Shawn Rasheed, Amjed Tahir, Alex Potanin. *EvilPickles: DoS Attacks Based on Object-Graph Engineering.* ECOOP 2017. **Distinguished Artifact.**
29. Darya Melicher, Yangqingwei Shi, Alex Potanin, Jonathan Aldrich. *A Capability-Based Module System for Authority Control.* ECOOP 2017.
30. Garming Sam, Nicholas Cameron and Alex Potanin. *Automated Refactoring of Rust Programs.* ACSC 2017.

31. Joseph Lee, Jonathan Aldrich, Troy Shaw, Alex Potanin. *A Theory of Tagged Objects*. ECOOP 2015.
32. Cyrus Omar, Darya Melicher, Ligia Nistor, Benjamin Chung, Alex Potanin, and Jonathan Aldrich. *Safely Composable Type-Specific Languages*. ECOOP 2014. **Distinguished Paper.**
33. Marco Servetto, Julian Mackay, Alex Potanin, and James Noble. *The Billion-Dollar Fix: Safe Modular Circular Initialisation with Placeholders and Placeholder Types*. ECOOP 2013. Pp 205–229.
34. Constantine Dymnikov, David Pearce and Alex Potanin. *OwnKit: Inferring Modularly Checkable Ownership Annotations for Java*. ASWEC 2013. Pp 181–190.
35. Alex Potanin, Monique Damitio and James Noble. *Are Your Incoming Aliases Really Necessary? Counting the Cost of Object Ownership*. ICSE 2013. Pp 742–751.
36. Daniel Atkins, Alex Potanin and Lindsay Groves. *The Design and Implementation of Clocked Variables in X10*. ACSC 2013.
37. Jan Larres, Alex Potanin and Yuichi Hirose. *A Study of Performance Variations in the Mozilla Firefox Web Browser*. ACSC 2013.
38. Hien Tran, Craig Anslow, Stuart Marshall, Alex Potanin, Mairead de Roiste. *Lessons Learnt from Collaboratively Creating Maps on a Touch Table*. CHINZ 2011.
39. Yoav Zibin, Alex Potanin, Paley Li, Mahmood Ali, Michael D. Ernst. *Ownership and Immutability in Generic Java*. OOPSLA 2010. Pp. 598–617.
40. Radu Muschevici, Alex Potanin, Ewan Tempero, and James Noble. *Multiple Dispatch in Practice*. OOPSLA 2008. Pp. 563–582.
41. Chris Male, David Pearce, Alex Potanin, and Constantine Dymnikov. *Java Bytecode Verification for @NonNull Types*. CC 2008. Pp 229–244.
42. Neil Ramsay, Stuart Marshall, and Alex Potanin. *Annotating UI Architecture with Actual Use*. AUIC 2008.
43. Yoav Zibin, Alex Potanin, Mahmood Ali, Shay Artzi, Adam Kiezun, and Michael D. Ernst. *Object and Reference Immutability using Java Generics*. FSE 2007. **ESEC/FSE 2007 Distinguished Paper.**
44. Alex Potanin, James Noble, Dave Clarke, and Robert Biddle. *Generic Ownership for Generic Java*. OOPSLA 2006. Pp. 311–324.
45. Alex Potanin, James Noble, and Robert Biddle. *Snapshot Query-Based Debugging*. ASWEC 2004. Pp 251–261.

Refereed Workshop Papers

46. Zara Hassan, Christoph Treude, Graham Williams, Michael Norrish, Alex Potanin. *Managing Reproducibility Debt in Scientific Software: A Practical Framework*. SERS 2026.
47. Sasha Pak, Fabian Muehlboeck, Alex Potanin. *A Verified Thread-Safe Array in Rust*. IWACO 2025.
48. Abhaas Goyal, Alex Potanin and Jonathan Aldrich. *A Comparative Study of Traditional versus Capability-Based Module Systems for Modern Programming Languages*. PLATEAU 2024.
49. Amos Robinson and Alex Potanin. *Pipit: Reactive Systems in F Star (Extended Abstract)*. TyDe 2023.
50. Baptiste Pauget, David Pearce, and Alex Potanin. *Towards Compilation of an Imperative Language for FPGAs*. VMIL 2018.
51. James Noble, Sophia Drossopoulou, Mark S Miller, Toby Murray and Alex Potanin. *Abstract Data Types in Object-Capability Systems*. IWACO 2016.
52. Du Li, Alex Potanin, and Jonathan Aldrich. *Delegation vs Inheritance for Typestate Analysis*. FTfJP 2015.
53. Darya Melicher, Alex Potanin, and Jonathan Aldrich. *Wyvern: Impacting Software Security via Programming Language Design*. PLATEAU 2014.
54. James Noble and Alex Potanin. *On Owners-as-Accessors*. IWACO 2014.
55. Jonathan Aldrich, Cyrus Omar, Alex Potanin and Du Li. *Language-Based Architectural Control*. IWACO 2014.
56. Cyrus Omar, Benjamin Chung, Darya Kurilova, Alex Potanin and Jonathan Aldrich. *Type-Directed, Whitespace-Delimited Parsing for Embedded DSLs*. GlobalDSL 2013.
57. Ligia Nistor, Darya Melicher, Stephanie Balzer, Benjamin Chung, Alex Potanin and Jonathan Aldrich. *Wyvern: A Simple, Typed, and Pure Object-Oriented Language*. MASPEGHI 2013.
58. Marco Servetto, David Pearce, Lindsay Groves, and Alex Potanin. *Balloon Types for Safe Parallelisation over Arbitrary Object Graphs*. WoDeT 2013.
59. Daniel Atkins, Alex Potanin, Lindsay Groves. *Clocked References in X10*. LaME 2012.
60. Julian Mackay, Hannes Mehnert, Alex Potanin, Lindsay Groves, Nicholas Cameron. *Encoding Featherweight Java with Assignment and Immutability using The Coq Proof Assistant*. FTfJP 2012.
61. Yoav Zibin, Alex Potanin, Paley Li, Mahmood Ali, Michael D. Ernst. *Ownership and Immutability in Generic Java (OIGJ)*. PLDE 2010.

62. Paley Li, Stephen Nelson, and Alex Potanin. *Ownership for Relationships.* IWACO 2009.
63. Paley Li, Alex Potanin, James Noble, and Lindsay Groves. *Towards Unifying Immutability and Ownership.* IWACO at ECOOP 2008.
64. Christo Fogelberg, Alex Potanin, and James Noble. *Ownership Meets Java.* IWACO at ECOOP 2007.
65. Alex Potanin, James Noble, Tian Zhao, Jan Vitek. *A High Integrity Profile for Memory Safe Programming in Real-time Java.* JTRES 2005.
66. Alex Potanin, James Noble, Dave Clarke, Robert Biddle. *Featherweight Generic Ownership.* FTfJP at ECOOP 2005.
67. Alex Potanin, James Noble, Dave Clarke, Robert Biddle. *Defaulting Generic Java to Ownership.* FTfJP at ECOOP 2004.
68. Alex Potanin, James Noble, Dave Clarke, and Robert Biddle. *Featherweight Generic Confinement.* FOOL at POPL 2004.
69. James Noble, Robert Biddle, Ewan Tempero, Alex Potanin, and Dave Clarke. *Towards a Model of Encapsulation.* IWACO at ECOOP 2003.
70. Alex Potanin and James Noble. *Checking Ownership and Confinement Properties.* FTfJP at ECOOP 2002.

Edited Journals

71. Alex Potanin and Bor-Yuh Evan Chang (Associate Editors). *OOPSLA 2024 PACMPL Volume 8 Issue OOPSLA1 and OOPSLA2.*
72. Alex Potanin and Gail Murphy (Editors). *Special Issue on APSEC 2016.* Science of Computer Programming. Volume 163, October 2018.
73. Alex Potanin (Editor). *Special Issue on NOOL 2015.* Journal of Object Technology. Volume 16, no. 2, April 2017.
74. Taso Viglas and Alex Potanin (Editors). *Special Issue on CATS 2011.* IJFCS, 2013. Vol. 24, Issue 1.
75. Taso Viglas and Alex Potanin (Editors). *Special Issue on CATS 2010.* CJTCS, May 2011.

Patent

- A computer implemented translation method. EP2122464A4.

Other Refereed Publications

76. Asma Shakil, Marie Devlin, KellyAnn Fitzpatrick, Sarah Carruthers, Mirela Gutica, Ronnie Howard, Stoney Jackson, Chris Johnson, Edward Latorre, Tyler Menezes, Joseph Mertz, Tominiyi Olupitan, Alex Potanin, Floris Westerman. *Exploring the Impact of Gen-AI on Team-Based Computing Capstone Projects.* ITiCSE 2026 Working Group, Madrid, Spain. ACM.
77. Sasha Pak, Richard Willie, Umang Mathur, Fabian Muehlboeck, Alex Potanin. *View Types in Rust.* Poster at SPLASH 2025.
78. Matthew Britton, Alex Potanin, Sasha Pak. *Verifying Extract Method Refactoring in Rust.* Poster at SPLASH 2025.
79. Zara Hassan, Christoph Treude, Graham Williams, Michael Norrish, Alex Potanin. *Reproducibility Debt in Scientific Software.* Poster at SPLASH 2025.
80. Manish Singh, Lindsay Groves, Alex Potanin. *A Relaxed Balanced Non-Blocking Binary Search Tree.* Poster, ICPP 2019.
81. Isaac Oscar Gariano, Marco Servetto, Alex Potanin, Hrshikesh Arora. *Iteratively Composing Statically Verified Traits.* Extended Abstract, VPT 2019.
82. Darya Melicher, Yangqingwei Shi, Valerie Zhao, Alex Potanin, Jonathan Aldrich. *Using Object Capabilities and Effects to Build an Authority-Safe Module System.* Poster, HotSoS 2018.
83. Aaron Craig, Alex Potanin, Lindsay Groves, Jonathan Aldrich. *Capabilities and Effects.* OCAP 2017.
84. Darya Melicher, Yangqingwei Shi, Valerie Zhao, Alex Potanin, Jonathan Aldrich. *Using Object Capabilities and Effects to Build an Authority-Safe Module System.* OCAP 2017.
85. Jonathan Aldrich and Alex Potanin. *Usably Expressing and Enforcing Design in Wyvern.* NOOL 2017.
86. Jens Dietrich, Kamil Jezek, Shawn Rasheed, Amjed Tahir, Alex Potanin. *EvilPickles (Artifact).* DARTS.
87. Darya Melicher, Yangqingwei Shi, Jonathan Aldrich. *A Capability-Based Module System for Authority Control (Artifact).* DARTS.
88. Jonathan Aldrich and Alex Potanin. *Delegation Revisited.* NOOL 2016.
89. Jonathan Aldrich and Alex Potanin. *Naturally Embedded DSLs.* DSLDI 2016.
90. Darya Melicher, Alex Potanin, and Jonathan Aldrich. *Modules in Wyvern: Advanced Control over Security and Privacy.* Poster, HotSoS 2016.
91. Joseph Lee, Jonathan Aldrich, Troy Shaw, and Alex Potanin. *A Theory of Tagged Objects (Artifact).*

DARTS 2015.

92. Cyrus Omar et al. *Safely Composable Type-Specific Languages*. Poster, ECOOP 2014.
93. Darya Melicher et al. *Type-Specific Languages to Fight Injection Attacks*. Poster, HotSOS 2014.
94. Cyrus Omar et al. *Extensible Type-Driven Parsing for Embedded DSLs in Wyvern*. Parsing@SLE 2013.
95. Cyrus Omar et al. *Extensible Type-Driven Parsing for Embedded DSLs in Wyvern*. Poster, SPLASH 2013.
96. Jonathan Aldrich et al. *DSL support in Wyvern Language*. DSLDI 2013.
97. Jan Larres, Alex Potanin, and Yuichi Hirose. *Performance Variance Evaluation on Mozilla Firefox*. NZCSRSC 2011.
98. Mairead de Roiste, Hien Tran, and Alex Potanin. *What makes a map?* IRLOGI 2010.
99. Chris Andreae, Donald Gordon, Alex Potanin, James Noble, Robert Biddle. *Terrier: Static Query-Based Debugging in Eclipse*. Poster, OOPSLA 2004.
100. Alex Potanin. *Generic Ownership: Practical Ownership Control in Programming Languages*. Doctoral Symposium, OOPSLA 2004.
101. Alex Potanin. *Practical Ownership Control in Programming Languages*. Doctoral Symposium, ECOOP 2004.
102. Alex Potanin. *A Tool for Ownership and Confinement Analysis of the Java Object Graph*. Poster and SRC entry, OOPSLA 2002. 2nd place in graduate division; accepted to ACM SRC Grand Finals; 2nd place in undergraduate category.

Theses

- Alex Potanin. *Generic Ownership — A Practical Approach to Ownership and Confinement in OO Programming Languages*. PhD thesis, 2007. Supervised by James Noble, Dave Clarke, and Robert Biddle.
- Alex Potanin. *The Fox — A Tool for Object Graph Analysis*. Honours report, First Class Honours, Victoria University of Wellington, 2002.