

Keith WANSBROUGH

| | | | |
|---------------------|---|---------------|-----------------------|
| Address: | 14 The Beeches, Woodhead Drive Cambridge CB4 1FY | Email: | keith@lochan.org |
| Tel: | 01223 763 581 (W) 01223 704 822 (H) | Web: | www.lochan.org/keith/ |
| Nationality: | New Zealand (work permit <i>not</i> required) | | |

Overview Keith Wansbrough is an experienced designer and developer with a solid background in research. He specialises in distributed systems, networks, compilers, languages, modelling, and prototyping.

Employment

2003 – present **St Catharine’s College and the Computer Laboratory, University of Cambridge, England**
Michael and Morven Heller Research Fellow

Designing the programming language *Acute*, for building reliable distributed systems (especially middleware), combining novel ideas and the best current research into a practically useful system. Building a flexible and efficient prototype implementation.

Constructing a formal specification of TCP/IP and the sockets API, paying particular attention to the failure behaviour. Using a custom symbolic model checker to validate this specification against the Windows, Linux, and BSD implementations. Design, implementation, and maintenance of distributed protocol analysis and validation systems.

Leading a team of five software engineers working on the above projects (2002 – present).

2000 – 2003 **Computer Laboratory, University of Cambridge, England**
Research Associate

Developing formal semantics for distributed systems, as a foundation for distributed middleware development and reasoning about failure-tolerance, reliability, and security.

Funded by EPSRC (wide-area programming) and EU project PEPITO (peer-to-peer computing).

1998 – present **University of Cambridge, England**
Supervisor

One-on-two academic supervisions of undergraduate students at all levels in computer science. Supervision of three final-year projects.

Education

1998 – 2002 **Clare Hall, University of Cambridge, England**
PhD (co-supervised, University of Cambridge Computer Laboratory and Microsoft Research)

Research into a program analysis for functional languages. Implementation of a highly-optimising production compiler; type and constraint systems; operational semantics. Extensive programming and development experience, including working directly with Glasgow Haskell Compiler development team. Funded by Commonwealth Scholarship.

1997 – 1998 **University of Glasgow, Scotland**
PhD (transferred to Cambridge)

1996 **University of Auckland, New Zealand**
MSc with Distinction

Research into a synthesis of two specification methods for domain-specific languages (monadic semantics and action semantics), with a prototype implementation. Elective courses in ancient Greek history, parallel and distributed computing (including PVM). Dean of Science Prize.

- 1992 – 1995 **University of Auckland, New Zealand**
BSc(Hons) First Class
Major: Computer Science (compilers, algorithms, information theory, artificial intelligence, program derivation). Minor: Mathematics (computer science foundations, algebraic structures, statistical theory, topology). Average A+. Elective courses in physics, chemistry, English literature. Final-year project on *Tracing Lazy Functional Languages*. Six prizes and scholarships.
- 1987 – 1991 **Selwyn College, Auckland, New Zealand**
Senior Scholar (co-dux)
University Bursary (final year): Chemistry, English, Calculus, Statistics, Physics (all A+).
Sixth Form Certificate (penultimate): Chemistry, English, German, Physics, History, Graphic Design.

Skills

- Design and development *Problem-solver*: able to establish quickly what the problem is, identify likely approaches to solving it, investigate them, and design and build the solution.
- Team worker*: productive in a variety of rôles: leading and managing within a small team; coaching in a team situation to develop domain expertise and intuition; collaborative management; working as team member towards a common goal.
- Quick learner*: able to jump in at the deep end, learning a new language or application from scratch and be productive within days.
- Listener*: good at drawing out real rather than stated requirements. Can quickly see and articulate ways of improving design and/or process. Able to communicate with non-technical users.
- Communicator*: experienced at writing and presenting technical papers, conveying key ideas clearly within strict time and space constraints. Good documentor. Good tutor/teacher.
- Enabler*: experienced in specifying, designing, and building visualisation tools, test harnesses, and integration code to enable the development process.
- Local expert* on tools such as Perl, LaTeX, Postscript. Contributor to TCP/IP and programming-language mailing lists and newsgroups.
- Experienced* in distributed systems, networks, concurrency, protocol design, language design, compiler design and implementation, prototyping, mathematical modelling.
- Motivation I delight in my broad and deep understanding of computing and information technology, from high level languages to assembler, from operating systems to logic gates, and from type theory to the software development process.
- Getting the right design is important to me: designing for maintainability and extensibility, appropriate use of standards, usability and consistency.
- I enjoy variety, intellectual challenge, and the chance to play with new technologies.
- I have an excellent memory for detail, as well as an eye for overall structure.
- Languages Experienced in: C (Unix/POSIX, Win32, TCP, pthreads, lex/yacc, etc), Perl, shell (and the standard Unix toolset), OCaml, MoSML/Standard ML, TeX/LaTeX, Haskell, HTML/CSS.
- Competent in: Java, C++, Visual Basic, ECMAScript/Javascript/DOM, Postscript, PDF, assembler (x86 and others), Scheme/Common Lisp/Logo, awk, Forth, Prolog, PVM, Delphi/Object Pascal.
- Used: SQL, XML, elisp, Tcl/Tk, Xlib, Gtk, Win32, MacOS, Matlab, Ada, Modula, HOL.
- Tools CVS, emacs, vi, make, gdb, strace, Glade, Visual Studio, CorelDraw!, AutoCAD, GIMP, Photoshop, Word, Access, Excel.
- Other Unix and Windows administration, installation, support.
- Linux and BSD kernel development and debugging.
- Basic German, written and spoken; elementary Japanese.
- Full UK driving license.

Achievements

- 2003 – 2004 Built automated testing system, raising coverage a thousandfold and transforming our development model.
- 2001 – 2004 Developed a typesetting tool supporting our unique requirements; used for all our academic papers and technical reports in the last three years, and also by several colleagues.
- 2004 In two days, learnt Javascript/DOM and wrote a test result visualisation tool that has enabled even junior research assistants to find bugs and extend the product.
- 2002 Patched the compiler we use to improve support for large programs; incorporated into next release.
- 1994 Extracted data from a proprietary accounting system into Microsoft Office.
- 1991 & 1996 Represented NZ at ACM Programming Contest 1996 and International Mathematical Olympiad 1991.

Earlier Employment

- 1999 **International Student Welcome Programme, Cambridge**
Volunteer Coordinator and committee member
Coordinating ninety volunteers over eleven days in two locations, welcoming international students as they arrived in Cambridge. Assisting with publicity, printing, organising orientation events.
- 1997 – 1998 **University of Glasgow, Scotland**
Tutor
Introductory software engineering course for engineers.
- 1991 – 1997 **Roland Corporation, Auckland, New Zealand**
Product Specialist
Systems and network administration; technical support within the company; on-site technical support for computer-aided signwriting machines and software; graphic design of promotional materials; preparation of demonstrations; database and spreadsheet development for sales and management; liaison with product development team in Japan.
Spent two weeks at Roland DG's Miyakoda plant in Hamamatsu, Japan, January 1994.
- 1991 – 1997 **Auckland Baptist Tabernacle Church, Auckland, New Zealand**
Sound System Coordinator
Live sound mixing for a variety of events; training and supervising operators; coordination with church management team; purchases and upgrade proposals; repairs to audio equipment.
Also (1999 – present): sound system operator, St Andrews Street Baptist Church, Cambridge.
- 1994 – 1996 **Auckland University Christian Union, Auckland, New Zealand**
Small Group Leader
Preparing and leading participatory Bible studies; facilitating group communication; pastoral care.
- 1993 – 1995 **Reticulation Development Limited, Auckland, New Zealand**
Contract programmer
Writing and maintaining job accounting and sales tracking databases.

Personal interests

- Sport Hiking, hill-walking, camping (both participating and organising); cycling for pleasure.
- Travel Visited Australia, Japan, Korea, US, and various countries in Europe for both business and pleasure.
- Other Cooking; photography; film; electronics (primarily digital); opera; reading (science fiction, fantasy, English classics); music listening (classical, electronica).

Selected Publications

- Refereed *Global abstraction-safe marshallng with hash types*. In International Conference on Functional Programming, ICFP 2003. With James J. Leifer, Gilles Peskine, and Peter Sewell.
- Dynamic rebinding for marshallng and update, with destruct-time λ* . In International Conference on Functional Programming, ICFP 2003. With Gavin Bierman, Michael Hicks, Peter Sewell, and Gareth Stoye.
- Rigour is good for you and feasible: reflections on formal treatments of C and UDP sockets*. In Tenth ACM SIGOPS European Workshop, 2002. With Michael Norrish and Peter Sewell.
- Timing UDP: Mechanized semantics for sockets, threads and failures*. In European Symposium on Programming, ESOP 2002. With Michael Norrish, Peter Sewell, and Andrei Serjantov.
- The UDP calculus: Rigorous semantics for real networking*. In Theoretical Aspects of Computer Software, TACS 2001. With Andrei Serjantov and Peter Sewell.
- Simple usage polymorphism*. In Third Workshop on Types In Compilation, TIC 2000. With Simon L. Peyton Jones.
- Once upon a polymorphic type*. In Twenty-sixth ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages, POPL 1999. With Simon L. Peyton Jones.
- A modular monadic action semantics*. In USENIX Conference on Domain-Specific Languages, 1997. With John Hamer.
- Tracing lazy functional languages*. In Computing: The Australasian Theory Seminar, CATS 1996. With Jeremy Gibbons.
- Theses *Simple Polymorphic Usage Analysis*. PhD thesis, Computer Laboratory, University of Cambridge, England, 2002.
- A Modular Monadic Action Semantics*. Master's thesis, Department of Computer Science, University of Auckland, 1997.
- Other *Specifying and Developing Abstractions for Distributed Computation*. Discussion paper for UKCRC Grand Challenges for Computing Research exercise, 2004. With Peter Sewell.
- The TCP Specification: A Quick Introduction*. Draft, 2004. With Stephen Bishop, Matthew Fairbairn, Michael Norrish, and Peter Sewell.
- Acute: High-Level Programming Language Design for Distributed Computation*. Submitted for publication, 2004. With Peter Sewell, James J. Leifer, and others.

Referees

Available on application.

Cambridge, England, Tuesday, 24 August, 2004