Reuben Thomas: CV

Full name Reuben Rhys Thomas
Nationality British
Email rrt@sc3d.org

This CV was last updated on 7th March 2023. The most up-to-date version is available at https://rrt.sc3d.org/Work/CV.

1 Skills

I am a software designer and engineer with a background in research. I can **untangle complex problems** to **focus on what is important**, **identify simple solutions** that involve **minimal disruption**, **build consensus** around their adoption and **guide their implementation**. I have written, maintain and contribute to a wide variety of free software, and consult for a range of individuals and businesses.

Software development I have a **firm grasp of the software development process** from analysis and design through coding and testing to delivery, support and maintenance. I have been an effective collaborator on many free software projects (see section 2), both reporting bugs and providing patches as a contributor, and fixing bugs, integrating patches and working with downstream packagers as a maintainer. I focus on developing portable software, accommodating the needs of users on particular platforms where possible, and am familiar with most major programming languages.

Writing I can **structure, write and edit documents** (see section 3.2) in a **fluent yet terse prose style**. Besides my **text book translation** and **PhD thesis**, I have published **several papers** on my research, and worked extensively on **documentation for dozens of programs**. I am expert in LaTeX (some of my packages are available on CTAN), and am also familiar with **DocBook**, **Markdown** and **AsciiDoc**.

Oral communication I have **excellent oral communication skills**, including **fluent French**, conversational Italian and basic German and Dutch, and am **practised at addressing audiences large and small**.

2 Free software

I have contributed extensively to a vast range of free software, with particular emphasis on improving **robustness**, **portability**, **longevity** and **interoperability**. To make these improvements, I rely on:

Simplification I **remove dead code**, **use standard APIs** to replace platform-specific code, **remove conditional compilation**, and where possible, **remove little-used or obsolete functionality**.

Testing I often **add test suites** to programs I maintain, primarily to help ensure that my changes do not break existing functionality.

Modernization I **take advantage of recent standards** (but not too recent! they must be widely deployed at the point of release), and new features available in old languages and tools, such as the C language, the **GNU autotools**, and **gnulib**, to reduce code size and complexity. I reduce the complexity of using memory management by using the tried-and-tested BDW garbage collector in C programs.

Translation I **rewrite C programs in other languages**, such as Perl and Vala. This reduces code size and complexity through the use of more expressive languages, while retaining portability (Perl runs on almost every significant platform; Vala compiles to C and relies on only widely-deployed C libraries).

Consolidation I **integrate patches from packagers** such as GNU/Linux distributions, **fix bugs** reported both up- and downstream, and **implement requested features**. This not only improves the software for users, but simplifies future packaging, lowering the barrier to keeping the program packaged and available for use.

Projects I currently maintain or co-maintain include (starred entries indicate that some of the work was paid):

GNU Hello the GNU demonstration package, available on every GNU system, and a model for other GNU packages. I updated the build system, and mentored my successor as maintainer (whom I later succeeded).

GNU a2ps the "anything to PostScript" converter. I volunteered to become a maintainer in 2022, am currently the only active maintainer, and in March 2023 made the first release since 2007, modernizing the code and build system.

GNU Zile the text editor which is a portable and resource-light clone of Emacs (to which I am also a contributor). After making some contributions, I took over from the original author and over some years I completely rewrote the original C version, removing functionality inappropriate to a minimal editor, and making it work precisely like Emacs, and adding a test suite. Later I rewrote Zile in Lua, but though this was an interesting experiment, it was not a suitable replacement for the C version; later still, I rewrote Zile in Vala, giving a 50% reduction in code size.

Recode the text encoding converter. I took over this program after its original author's death, tidied up several years' worth of unreleased improvements and fixes, merged downstream patches that had accumulated in the mean time, set up multi-platform CI, and made the first release for ten years. Since then I have continued to fix bugs, improve the build system, and reintroduce functionality that had been removed by the original author's changes since the previous release.

Enchant the meta-spellchecking library. I took this over from its previous maintainer, updated it to work with more recent versions of the spell-checking libraries that it supports, removed its hacky library version of Ispell (for most users, Aspell, which Enchant supports, will be better in any case), fixed and added some public APIs, and removed previously-deprecated APIs. I made major improvements to the ispell-compatible standalone front-end, and contributed Emacs support for it, so that Emacs can now use Enchant. I made an updated release binary-compatible with the previous release before releasing the new APIs. Several years later, the new version is in widespread use.

PSUtils a suite of utilities for manipulating PostScript documents, which I rewrote from C to Perl, simplifying the code, fixing bugs, and adding a test suite.

libpaper a library for paper size information. I rewrote it, added a test suite, and simplified and improved its configuration interface, allowing users to specify their own paper sizes as well as configure their preferred paper size.

Caffeine I redesigned and rewrote this utility for preventing personal computers from sleeping, splitting it into a background application that does not need configuration or interaction that simply keeps the computer awake when there is a full-screen window in the foreground, plus a manual toggle and command-line command wrapper, for users with more complex requirements.

LaTeX I maintain several LaTeX packages on CTAN which are included in most LaTeX distributions.

Ruth* I produced this simple but general libre XML templating tool while working at the legal tech startup Mattereum in 2021–2; I also made several contributions to the libre Node.js XPath/XQuery implementation fontoxpath.

Nancy* a simple text templating tool (of which Ruth is a specialization to XML), which I wrote and maintained for building online reference material from 2002–14, and have continued to maintain personally since.

Projects to which I have made significant contributions in the past include (starred entries indicate paid work):

Vala I improved automake support for Vala, and supplied fixes and improvements to core C bindings, and to the documentation system. I also took on maintenance of the Emacs editing mode for the language, and implemented long-string support for it.

Metapolator* a tool for manipulating entire fonts which I worked on as a consultant to Google Fonts in 2014. I also provided many patches to its JavaScript dependencies.

Fontforge I overhauled the build system and cleaned up a lot of code, fixed bugs, and worked on the collaboration mode of this font editor.

SoX the sound converting and processing program: code clean-up, build system modernization

file the file type identifier: improved MIME type support, started a test suite, many miscellaneous fixes

Lua I wrote and maintained several widely-used fundamental libraries, as well as making minor contributions to the core language implementation.

luaswig* A patch to the C bindings generator SWIG that added support for the Lua scripting language, which I developed while working for the healthcare software company Protechnic Exeter in 2001–2.

3 Other work

For most of my career I have worked simultaneously in a number of other fields:

3.1 1989–present: IT consultancy

As a consultant to individuals and companies, I work with clients to identify and then meet their needs at the level that works for them, whether they are technically advanced or neophytes, and whether they want to be heavily involved in planning and decision-making or would prefer to have me take decisions for them.

I have taken on roles that involve considerable trust, such as forensic analysis of sensitive personal data.

3.2 2003–present: Editing, writing and proof-reading

I have worked as editor, translator, copy-writer and proof-reader on a variety of projects, including translating a major text book in theoretical computer science from French into English, editing and writing copy for personal and corporate web sites, and editing and proofing documents ranging from academic papers to career-critical professional letters.

I understand the problems non-native users of English typically have, and am adept both at writing clearly for non-native readers, and understanding what non-native writers mean, so as to reword it more comprehensibly.

3.3 1982–present: Singer

I am a professional classical singer, working regularly with a wide range of groups and appearing as a soloist. From 2008–18 I sang daily at **Westminster Cathedral**. For more details, see my **Singing CV**.

4 Education and Academic Posts

4.1 2008: Invited researcher at Université Paris-Est Marne-la-Vallée

I worked full-time for three months on the Vaucanson Project, a free software platform for computing finite automata, simplifying and consolidating the code in a project that had become overly complicated at the hands of several generations of programmers, mainly students who each worked on the project for a year.

4.2 1999–2001: Postdoctoral research assistant at the University of Glasgow Department of Computing Science

Working at **Microsoft Research Cambridge** on the **Glasgow Haskell Compiler**, I was responsible for maintaining and enhancing the Windows port of the compiler, the .NET back-end, and the build system.

4.3 1995–2000: PhD at University of Cambridge Computer Laboratory

I designed and implemented **Mite**, a virtual machine that allows multiple languages to be compiled into binary-portable object code, and translated at load-time into native code. Producing good quality native code is left to the compiler, and does not affect the speed of the fast load-time translator.

4.4 1992–1995: BA in Computer Science with Mathematics at St John's College, Cambridge

I obtained first-class honours in Computer Science with Mathematics, and was awarded several prizes and a college scholarship for my examination results (which included obtaining the second highest mark in the university two years running), and the St John's College Master's Essay Prize for an essay on the future of computing.

5 Other interests

I am a published composer, lyricist and translator, and also enjoy performing and recording stories and poetry, and book design.