Early Unix Culture at Coach House Press

John W. Maxwell
Canadian Centre for Studies in Publishing
Simon Fraser University
jmax @ sfu.ca

In the early 1970s, the Coach House Press, a tiny literary publisher and fine-art printing house in Toronto drove headlong into the digital era, anticipating by three or four decades the moves that their peers in the book industry are beginning to make only now. How did this small press manage this, given marginal capitalization (indeed, much of it was funded through arts grants), immature technologies, and the apparent divide between the arts and sciences? The answers to this question offers numerous insights into a cultural history of computing in the 1970s—which, by way of Internet culture, still underpins much of our media ecology today.

A Call for Software Enthography

In his 2004 ACM Turing Lecture, computing pioneer Alan Kay remarked that the terms "computer science" and "software engineering" are at best premature if not entirely regrettable attempts to exhalt the intellectual status of computing. Of engineering, Kay went so far as to suggest that the contemporary state of the art in computing is comparable to the construction of pyramids in ancient Egypt. Kay's concern was not to criticize the field, but to find a proper place from which to approach computing education.

While computing may indeed lack the 'rigorous' intellectual backbones of science or engineering to which it aspires, what it certainly does have is culture—or cultures, plural. This is an aspect well recognized and represented in the study of "cybercultures" and "online communities" but still nearly entirely effaced in the context of software development communities. Surely, in order to engage in any kind of critical inquiry into digital technology, we need to be properly aware of cultural forms and distinctions and of the meaning-making that cultures enable. In short there is a needs for *software ethnography.* At the very least, there is a need for more cultural history of software. Such is the orientation of the current paper. [1]

In this paper I will be (mis)representing two groups—the early Unix community of the 1970s and early 1980s, and the people associated with a small, Toronto-based literary publisher and printer called the Coach House Press during the same period. My relationship to both is marginal: I am not old enough to have much direct experience with the old Unix world, but as a sometime software developer and system administrator, I am among the generation who directly inherit the traditions initiated by the early Unix users and developers—one of the strongest and most influential software cultures in existence, whose traditions underlie practically everything we do with computers today.

My relationship to the Coach House Press is not so different; the firm exists today (under the name Coach House Books), and a number of the principals have been in place for forty years or more, but most of the story I tell here took place when I was still in elementary school. As such, my access to the details of the story have been by way of interviews with the people who were there—on both sides I have found plenty of old-timers happy to talk about their work. The collected traces of both communities are for the most part readily available, this a significant byproduct of the virtues of openness and documentation shared by both communities—and of course made much more accessible by the Internet and free software.

For its part, the Coach House itself is still standing—just a block from Innis College on the University of Toronto campus. In addition to housing the present operations of the publisher and the contemporary printing operation, the premises are themselves a living—if somewhat crumbling—museum of printing technology. Though I had been investigating Unix and other computing cultures for years, [2] I began a series of pilgrimages to the Coach House in 2007, with the aim of piecing together the story of how a tiny Canadian literary publisher managed to be at the forefront of so many innovations in digital media.

The Early Unix Community

To speak of computing culture at all is to step out on a limb; there is precious little ethnography documenting the rituals, kinship structures, and worldviews of various computing communities. [3] Homi Bhabha once suggested (1994) that culture is knowable only in the encounter with *difference,* and if this is the case, then the moment of encounter with something like Linux makes the point: here is something very different than we are used to in our quotidian desktop environments. Alternatively, if we treat culture (methodologically) *as history,* then we can note that today's Linux and the free/open source projects surrounding it are extensions of the much more venerable Unix tradition.

Unix was originally developed by hackers Ken Thompson and Dennis Ritchie at AT&T Bell Labs around 1970. It was something of an unofficial project that gathered momentum as programmers both within AT&T and beyond came to appreciate its virtues. In 1973, Ritchie and Thompson publicly unveiled it at an ACM conference, and Thompson took it with him when he went to teach at UC Berkeley in 1974. Unix spread almost despite any formal efforts through the 1970s—AT&T had been judged a monopoly by the US Justice Department in the 1950s, and the regulations which followed this ruling effectively prevented AT&T from bringing software to market directly. Unix spread, largely unofficially, by individual contact: Thompson, Ritchie, and their colleagues sharing tapes with people they encountered, mostly at Universities—we might call this an early form of 'peer to peer' file sharing. Steven Weber (2004) writes:

> Before an AT&T researcher wrote the first Unix-to-Unix copy program (UUCP) in late 1976, it was almost impossible to send files over the Internet from one Unix machine to another. So Unix spread in the most traditional of ways—scholars on sabbatical and research fellows going back and forth between BTL, Xerox PARC, and

> university departments. There was a physical community of Unix users long before anything like a virtual community could have been imagined. (p 29)

Unix found an interested audience among computer researchers in the 1970s for a number of reasons: it was small, modular, elegant in design, and portable from one kind of hardware to another. It had little of the baggage of larger systems with corporate or large-project agendas. Most importantly, it was *open* to a significant degree; Thompson and his collection of colleagues moved the software forward by distributing tapes with bug fixes and updates, and the community responded in kind, contributing their own fixes and patches and additional components. AT&T corporate had little idea what to do with it for the first several years, and so they enabled a liberal licensing scheme with a good deal of leakage.

John Unsworth (1995) noted the apparent paradox in the 'open,' 'public,' even 'democratic' technologies springing from the pure research divisions of American monopoly capital—the Pentagon's ARPA project, AT&T Bell Labs, Xerox PARC—and their relative insulation from quarterly results and other real world economic pressures that collossal institutions could uniquely provide. The ARPA community in the 1960s widely shared the fruits of their efforts, seeing their work as an extension of the scientific/academic tradition of publishing research so that other researchers could build upon it. The underlying technologies of the Internet and most of the computing architecture surrounding it were developed and disseminated according to these ideals. The Unix operating system and the community and tradition surrounding it also clearly embodied this model of development, since it meant that the development community was distributed, consisting of programmers and researchers from a number of different sites and institutions, some corporate and some academic, contributing to a larger, common good.

By the mid 1980s, Richard Stallman's articulation [4] of the rationale and ethics of of "free" software put a formal face on a practice that had been widespread, though informal, for decades. By turns, [5] Unix became the dominant computing platform by the 1980s—and ultimately the key platform for the Internet—because of this development and distribution ethic. Its dominance continues today, both in its obvious incarnation in the free-licensed Linux software, and in its less visible manifestation behind the scenes in countless systems (Apple's OS X a notable example). Unix' cultural impact, however, is a deeper and more complex story.

To know Unix well is to come to appreciate its history—much moreso than in any other computer system in widespread use today. Although many Unix and Unix-like systems today have familiar-looking windows and menu-driven graphical interfaces, the "Unix way" is to type interactively in a command shell, and the commands one issues are in many cases understandable only by reference to the historical contexts that produced them. Eric Raymond's (2003) *Art of Unix Programming* is a contemporary case in point, a work of cultural ethnography at least as much as a guide to software development, which proceeds by tracing the virtues and values emergent within this now venerable culture of computing. Novelist Neal Stephenson wrote that "Unix is not so much a product as it is a painstakingly compiled oral history of the hacker subculture" (Stephenson 1999). A better characterization—and one that avoids the clichéd, pith-helmet anthropologist trope—comes from software philosopher Ward Cunningham, who observed that "Unix is the Latin of computer science," to which some wag replied, "Except it ain't dead yet." [6]

When we consider that Unix and its direct descendants (like Linux) are the visible and lasting fruits of J.C.R. Licklider's ARPA vision from the early 1960s (almost literally kept alive through the dark ages by monks) and that they serve as the foundation stones of the Internet, the Web, and the free/open source software movement, one begins to appreciate Unix's pervasiveness and—significantly—its resilience in the face of wave after commercial wave of 'innovation' designed to make it obsolete. Nikolai Bezroukov wrote that "Unix Renaissance OSS proved to be the most important democratic movement in software development in the 20th century" (Bezroukov 1999).

That said, there is no doubt that the tradition which comprised Unix and the bulk of the free software movement is a *curriculum* par excellence, that it constitutes a literacy in the richest sense of the word. In his excellent work on the culture of free software, anthropologist Chris Kelty notes that Unix's long term survival is partly due to its existence as a *pedagogical* system in addition to being a functional computing system. Kelty notes that John Lions' seminal 1977 document, *A Commentary on the Sixth Edition UNIX Operating System* was essential to the development of community ownership of Unix—making Unix a *pedagogical* text rather than it being proprietary code. Lions, an Australian professor, produced a document which both exposed and annotated the Unix source code—a tremendous boon to students and researchers working with the system. Lions' *Commentary* was illicitly photocopied repeatedly and passed hand to hand, as Unix itself had passed, by tape, from lab to lab. Kelty writes,

> Lions' commentary ... shows how UNIX was ported not only to machines (which were scarce) but also to the minds of young researchers and student programmers (which were plentiful). (2008 p 133)

> "Four remarkable things were happening at the same time. One, we had discovered the first piece of software that would inspire rather than annoy us; two, we had acquired what amounted to a literary criticism of that computer software; three, we were making the single most significant advancement of our education in computer science; and four, we were breaking the law." (Peter Reintjes, quoted in Kelty p 134)

Lions' *Commentary* was not the only pedagogical text in circulation among this new community. Another key text was Brian Kernighan & P.L. Plauger's 1976 book, *Software Tools,* a book about Unix for people who didn't have access to Unix itself. The book introduced core Unix ideas and structured programming methods but implemented them in the older and relatively ubiquitous Fortran language by way of a preprocessor. "RATFOR provided C-like control structures to Fortran." [7] *Software Tools* extended the Unix community and its virtues well beyond the still-small installed base of the software itself.

Even more fundamentally, the Unix system documentation—the collection of "man pages"—was distributed within the system itself, and a supporting collection of documentation tools allowed them to be viewed, formatted, and printed. In fact, Unix's first real "application" at AT&T was text processing; this document-handling functionality should be seen as a key part of Unix' appeal as a portable programmer's toolkit, containing everything required to make the system maintainable. It also underscores the idea that Unix is and was a text.

Meanwhile, in Toronto...

The Coach House Press was founded in Toronto in 1966 by printer Stan Bevington and has remained a central fixture of the literary scene in Canada ever since, having published work by the likes of Margaret Atwood, Michael Ondaatje, bpNichol, George Bowering, and more recently Guy Maddin, Sean Dixon, and Christian Bök. The firm has traditionally operated on two fronts: a printing operation based on the ground floor and a publishing operation upstairs; these two were separated briefly in the mid-1990s as the firm faced financial difficulties, but were re-united in 1998 under the name Coach House Books.

Always an innovator in the way he ran the printing operation, Bevington became acquainted in 1972 with Professor Ron Baecker, who ran the Dynamic Graphics Project at the University of Toronto. Baecker, an MIT Media Lab alumnus, was working on advanced computer graphics and set up a lab run in similar fashion to the ARPA-funded labs he had seen in the 1960s. Baecker's grad students were exploring a variety of graphics and imaging applications, including typesetting. Baecker's lab and grad students—and in particular, a grad student named David Tillbrook, whose thesis was on an automated newspaper layout system—had a major influence on the people at the Coach House.

Bevington, Baecker, Tilbrook, and ex-*Globe & Mail* computer programmer David Slocombe, along with Ed Hale from Mono Lino Typesetting began exploring the state of the art of computer-driven typesetting. In the early 1970s, this meant producing punched paper tape that could automatically drive a linotype machine to produce lines of lead type. But the Coach House group stretched far beyond this relatively straightforward technique, directly connecting an early minicomputer (a Datapoint 2200) to a Mergenthaler V-I-P (Variable Input Phototypesetter). A computer-driven phototypesetter meant that the process could be manipulated and fine-tuned at both ends: both in the composition of the set type, and in the typographic image itself, as this was generated entirely photographically instead of being cast in metal.

About the same time, Baecker's lab at the University of Toronto brought in one of the first Unix installations in Canada, running on a PDP-11/45 with a high-end Three Rivers "Graphic Wonder" display. Not only did this allow the group to explore digital typesetting, but also brought the new Unix technology and culture directly to the Coach House. [8]

Unix concepts preceded real UNIX at Coach House. In the late 1970s, Unix and C were unavailable to Coach House programmer David Slocombe on the Datapoint 2200. The bridge came from Kernighan & Plauger's book *Software Tools,* which inspired Slocombe to create his own preprocessor, not for Fortran but for the 2200's 8008 assembler. Based on the preprocessor, Slocombe next created a recursive macro processor that became a typesetting language for driving the Mergenthaler V-I-P. The macro processor's syntax was influenced by contemporary developments in GenCode and (S)GML by Charles Goldfarb at IBM, which they had seen at a GCA conference.

Stan Bevington reported incorporating real Unix into the Coach House production workflow via a leased line to a University of Toronto machine, [9] possibly the UTZoo host which was a key Usenet node in the 1980s. A 1984 map of Canadian Usenet hosts shows 68 nodes; 64 of them were university-based; 3 were at software companies, and one at Coach House Press. [10]

By the early 1980s, Unix-based workstations became available. In 1984, Bevington put together arts grant money (~$30k) to buy a Pixel Systems workstation. The arrival of Unix in various forms meant a move away form David Slocombe's preprocessor, and moving to Unix tools.

Troff and the Documentor's Workbench

One of the earliest and most successful applications for the Unix operating system was *troff*, a text-formatting and printing program which could drive a Graphic Systems C/A/T typesetter at AT&T Bell Labs. Troff (roughly, "typesetter runoff") was developed in the early 1970s at Bell Labs by Joe Ossanna, one of the early Unix team. Troff became the core of a suite of Unix tools called *The Documentor's Workbench,* which played a key part in the evolution of the Unix community: from almost the very beginning, the Unix "man" pages were essential to the system and its adoption. Troff and related tools were required to make the documentation system work: they were part of the self-reflective nature of the software, in that one used Unix to learn Unix.

Troff proved to be a popular tool. By 1979, AT&T's Brian Kernighan had created "ditroff" or *device-independent"* troff, to enable use with devices other than the C/A/T typesetter. It was good, but not good enough for the trade printers at the Coach House. Coach House programmer David Slocombe noted that though ditroff supported proportional fonts, "it was woefully deficient in quality from a commercial typesetting perspective. It was good enough for academics and grad students ... but not good enough for commercial book publishing." [11]

Slocombe's work at Coach House in the early 1980s further extended the troff package such that a significant amount of troff development was happening in Toronto. In 1984, with the goal of making troff into a typesetting tool capable of passing muster with professional printers (e.g. kerning, hyphenation and justification), Bevington, Slocombe, Patrick Dempster, and the writer/impresario Yuri Rubinsky spun off a new company, SoftQuad, to develop and market an enhanced version called sqtroff.

> SoftQuad set out to make production quality typesetting software for UNIX. "Production quality typesetting" in this regard means the type of type (so to speak) that would be acceptable in an old-style craft printing house. There is quite literally nothing available that is up to these typographer's standards. Not Scribe. Not TeX. Not AT&T's ditroff. Not Mac-Anything. Not PageMaker. All these are fine products in their own field, have been used to typeset many papers and theses and a small mountain of books. But none of them will stand up in a job printing shop with high standards and commercial pressures. (Ian Darwin, UseNet comp.text newsgroup, 6-Feb-87)

By the mid- to late-1980s, SoftQuad's focus began to fall in line with a new international standard for text and document processing: Standard Generalized Markup Language (SGML). Seeing the opportunity, Peter Sharpe at SoftQuad created software called Author/Editor, which was the first dedicated authoring environment for SGML. At the Coach House Yuri Rubinsky went so far as to train the Coach House authors—Margaret Atwood among them—to use the Author/Editor software to mark up their own manuscripts in SGML, an otherwise unheard-of merging of talent and technique. In a sense, the Coach House led Unix culture well out of its native habitats and into the literary establishment, finding common political and ethical ground there.

SoftQuad became one of the world's leading SGML software developers, a position it would hold for decades. SoftQuad itself branched out into Web-editing software with their pioneering HotMetal program (the web's HTML being a direct application of SGML). HotMetal later evolved into the industry-leading XMetal software for editing XML content.

By the early 1990s, Macintosh-based desktop publishing technology had become mainstream, and the World-Wide Web was born. At the Coach House, SGML-marked manuscripts were used to feed the more contemporary QuarkXPress layout program (Alschuler 1995, 152ff). But with the advent of maturing desktop publishing software, the point of a computer-driven *typesetting* system faded in favour of of an integrated imaging model based on Adobe's Postscript technology—the paradigm still dominant there today. At the Coach House, the old Unix machines are still there, sharing space with the collection of linotype and letterpress equipment. But working, contemporary Unix systems of course run the company's online presence.

Metanarratives: Innovation and Resilience

Today, in a period when book publishers worldwide are facing the challenges of "digitization" —often involving making heady investments in XML-based 'workflows,' the story of the Coach House's early experience stands out: here was a small art printer and poetry house that not only *adopted* this technology but *innovated* and indeed owned a good chunk of the state of the art for a time. The Coach House had no large capital base to draw on (unless one counts the Canada Council for the Arts) and no computer science PhDs (except those numbered among their friends), but certainly had no trouble innovating.

But innovation is not the only theme in the Coach House story. *Resilience* is another—for the professional printers at Coach House Press, this massive importation of technological culture and technique was unambiguously for the purpose of producing fine printing work, a tradition and aesthetic with a far deeper roots, and, arguably, far more stable virtues and ideals. The Coach House was not interested in transformation for transformation's sake; the books and posters and broadsheets that came out of the press through these years were not recognizably digital, neither nor aesthetic sensibility nor origin (cf. the easily identifyable 'DTP' look that proliferated in the late 1980s when Macs and laser printers became affordable). Far from it; the edge that Stan Bevington and his collegues sought was to be a better fine printer, to be able to achieve levels of quality and efficiency that either gave them a competitive advantage or satisfied their brocoleur natures.

We are at this point in history accustomed to celebrating the transformative potential of digital technoculture. But there is also a range of expression that includes the hybridization of cultural moments; I believe the Coach House Press story demonstrates this, and it may be that a relatively mature digital culture like Unix more readily reveals such examples. I believe also that this underscores the democratic potential of open systems and free software (though these are both terms applied retroactively to the Unix phenomenon). That a system can spawn both large- and small-scale promethian stories (as with the Internet and cyberculture in general) as well as more craft-oriented and almost conservative movements—without internal contradiction—speaks to the enormity of its scope. My hope is that this serves as an invitation for a good deal more inquiry into the software culture.

Notes

[1] Cultural histories of software are few and far between; Christopher Kelty's (2008) *Two Bits: The Cultural Significance of Free Software* is an exceptionally good one. The appearance of this text bodes well for this field of study.

[2] See my PhD Dissertation (University of British Columbia, Curriculum and Instruction): *Tracing the Dynabook: A Study of Technocultural Transformations,* available at: http://thinkubator.ccsp.sfu.ca/Dynabook/dissertation

[3] Notable works that might be called ethnography include Peter Salus' (1994) *A Quarter Century of Unix* and (2008) *The Daemon, the Gnu, and the Penguin*; Eric Raymond's (2003) *The Art of Unix Programming;* and his *Jargon File* at http://catb.org/jargon/

[4] See the GNU Project and Free Software Foundation: http://www.gnu.org

[5] The story of Unix' survival through the 1980s is too long and complex to recount here. See Kelty 2008 for an excellent treatment.

[6] See the entry for "UnixCulture" at the Portland Pattern Repository: http://www.c2.com/cgi/wiki?UnixCulture

[7] David Slocombe, personal interview, Toronto ON, June 17, 2008.

[8] Salus' *A Quarter Century of Unix* notes that several of the Bell Labs Unix team had been alumni or otherwise associated with the University of Toronto. Ron Baecker's colleague Mike Tilson received the Unix tapes directly from Ken Thompson in 1974.

[9] Stan Bevington, personal interview, Toronto ON, June 14, 2007.

[10] Henry Spencer, "CanNet map" post to can.general Usenet group, May 14, 1984.

[11] Slocombe, interview.

References:

Alschuler, Liora. *ABCD . . . SGML: A User's Guide to Structured Information.* Boston: International Thomson Computer Press.

Bezroukov, Nikolai. 1999. "Open Source Software Development as a Special Type of Academic Research: Critique of Vulgar Raymondism." *First Monday 4* (10).

Bhabha, Homi. 1994. "The Commitment to Theory," in *The Location of Culture.* London: Routledge.

Coffee, Clark & Daniel Haagens. n.d. "Graphion Museum: Old Phototypesetter Tales." Available at http://www.geocities.com/danhaag/oldtype.tpl.html

Darwin, Ian F. & Geoffrey Collyer. 1984. "A History of UNIX before Berkeley: UNIX Evolution: 1975-1984." Available at http://www.darwinsys.com/history/hist.html

Kelty, Christopher. 2008. *Two Bits: The Cultural Significance of Free Software.* Duke University Press.

Kernighan, Brian W. 1982. "A Typesetter-independent TROFF." Computer Science Technical Report No. 97. AT&T Bell Laboratories.

Kernighan, Brian & P.L. Plaugher. 1976. *Software Tools.* Addison-Wesley.

Lions, John. 1977. *A Commentary on the Sixth Edition of the UNIX Operating System.* Unpublished paper. Available at http://www.lemis.com/grog/Documentation/Lions/

Powers, Shelley & Jerry Peek. 2002. *UNIX Power Tools,* 3rd Edition. Sebastopol: O'Reilly.

Raymond, Eric S. 2003. *The Art of Unix Programming.* Reading: Addison-Wesley.

Salus, Peter H. 1994. *A Quarter Century of UNIX.* Reading: Addison-Wesley.

Salus, Peter H. 2008. *The Daemon, the Gnu, and the Penguin.* Reed Media.

Stephenson, Neal. 1999. In the Beginning was the Command Line. New York: Avon.

Weber, Steven. 2004. *The Success of Open Source.* Cambridge: Harvard University Press.

Wershler-Henry, Darren. 2000, Jan. "Coach House and Technology." Coach House Books website, circa 2000–2001.

Many thanks to David Slocombe, Stan Bevington, Ron Baecker, and Darren Wershler-Henry for their time and input in the preparation of this piece.