History, Nostalgia and Software

David Holdsworth

History, Nostalgia and Software

David Holdsworth
Computer Conservation Society and Leeds University, UK
ecldh@leeds.ac.uk

Abstract. The early history of computing is dominated by hardware development, but once we got non-trivial machines to work, the character of the machines was defined by their software not their hardware. Modern computers can be programmed to emulate computers of yesteryear, and then run original software. Sadly, much software from the past has been lost with cavalier disregard for its historic significance. However, we are having some success in resurrecting past systems, and can run such software as survives so well that past users of these old systems often react with nostalgic glee on first encountering one of these emulations. We can do this even where the software only survives in the form of printer listings. The challenge is to make such emulations relevant to people who never knew the original.

Keywords: software, preservation, emulation

1 Introduction

As we preserve technology from the past, we seek to make it comprehensible, and even eye-catching, to current and future generations. We need to address a vast range of levels of historical interest from the school child on a museum trip to the serious historian working to learn more of the technology that is long past current usage. With regard to the visitor in a museum (real or virtual), the problems seem to be particularly acute with regard to computing, where the hardware had very little of visual interest, few moving parts, and rarely made interesting sounds. In the past, computers on film and TV usually appeared as images of spinning tape decks, but once disk drives became common place that opportunity was denied to film makers, although flashing lights stayed around for some time, especially on IBM 360/370.

Actually, the real nature of a computer is defined by its software at least as much as by its hardware. Around my desk are three active computers, two Intel and one ARM, plus an Android smart-phone which is also ARM. From a user perspective the software is what describes these machines. The one Win98 system is rather the odd-one-out (Intel CPU). The other three machines all run variants of GNU/Linux, and the two most similar machines are the Ubuntu [1] laptop with its Intel CPU and the Raspberry Pi [2] with its ARM CPU. The two Intel machines are not very much alike.

It is not really possible to demonstrate a preserved computer without running any of its software, whereas we can demonstrate the software without actually possessing the original hardware. We can go beyond demonstration and even do useful work [3]. In the heydays of time-sharing systems many users of a computer sat at teletypes and

never saw the actual hardware. What they saw was software − software which sadly has all too often been completely lost, or may survive only as tatty printer listings.

The goal of the Computer Conservation Society (CCS) activity in software is long-term preservation of historic software in digital form, along with emulation software that will allow realistic execution of the preserved software, and meaningful access to the source text. We have achieved this both for ICL's George3 [4] and for Whetstone Algol on the KDF9 [5]. We are aware of emulations of the BBC micro [6] [7], IBM mainframes [8], DEC machines [9], and SIMH [10].

We have had real success in producing systems that can be used by those who knew the original. The challenge is to go from this nostalgia to preservation with historical relevance, both to serious historians and to passing museum visitors and Googlers. There is some encouragement in a post on DesignSpark [11] where the author has found our system for George 3 and got it to work quite independently. In an effort to investigate the scale of this challenge, I am currently working on resurrection of Leo III software. Before the project I had no non-trivial knowledge of the Leo III. So I am experiencing at first hand the problems of understanding the past when lacking personal nostalgia. The nostalgia of others is proving invaluable.

Software is key to the preservation of computer history, keeping the originals as objects of study, and writing tools (notably emulators) to enable that study.

2 Documentation

Just as Disraeli should have said "Lies, damned lies, statistics, − and spreadsheets", so our title should perhaps be expanded to "History, nostalgia, software, − and documentation". Software without documentation can be difficult to use or understand, but documentation without software can be a sterile read. Unfortunately, whereas documentation often survives in libraries, the software itself has not often survived. In making preserved software (and also hardware) meaningful for future generations, documentation will be vital, both the original stuff and newly written material. Moreover, it needs to be digital.

For today's systems, user documentation is on-line (or maybe CD), and searchable with today's software tools. The systems of the 1950s and 60s had user documentation that ran to a few volumes. We have scanned copies of some of this stuff, and the quantity is sufficiently small that the serious historian is not put off reading it in its entirety. Our own work in CCS has combined OCR with manual editing to produce searchable manuals whose on-line appearance resembles very much the original, and offers the prospect of enhancement with hyper-links. For a taster from our on-going Leo III project, look at [12]. We have used the same technique on system documentation for KDF9 [13].

When we come into the 1970s, documentation is still normally on paper, often produced by a traditional hot-metal process, but is now very voluminous − metres of shelf space of A4 sized manuals. However, the quality of printing is much better than that of earlier documentation, giving hope that the emerging generation of OCR software will get near-perfect recognition. A minority of documents already exist as searchable on-line documents such as IBM's *360 Principles of Operation* [14]. As

well as such searchable access to the original documents for scholars of IT history, we need material which is much more compact for the customers in the Clapham computer shop. Back in the 1970s, several user institutions rightly concluded that this large quantity of manufacturers' documentation was going to be a hurdle to end-users, and wrote their own user documentation, such as Leeds University's description of George3 [15]. The manufacturers also got wind of this issue and produced handy pocket cards for use in an era when more computer staff wore jackets. IBMers were always well-known for their dress sense and IBM's green and yellow cards are rightly famous [16].

3 Nostalgia as an Asset

Although nostalgia can be a false friend in blinding implementers to the fact that their emulations are only meaningful to those who knew the original, it is becoming clear that it is also an indispensable asset.

Nostalgia is a great asset to the Leo III project. I have been lent User Manuals and software listings. The Leo Society has recruited a team of enthusiastic volunteers who are copy-typing the listings in duplicate using the techniques explored in the resurrection of Whetstone Algol [5]. Not only is nostalgia a great motivator of such a team, but it also teases out recollections of those parts of a system that never quite made it into the documentation. Quite early in the project it became clear that the surviving documentation and preserved software would not be sufficient to achieve a working system such as was produced for the ICL1900 or KDF9. The vocabulary of computing has changed since the Leo III manuals were written. It was not too difficult to see that a "compartment" in the Leo III store is a "word" in today's parlance, but it took several e-mails with the old Leo hands to tease out just what was meant by a "switch". It turns out to be rather like an Algol 60 switch, or perhaps more like FORTRAN's computed GOTO. The manual was obviously written by someone who expected the reader to know exactly what a switch was. As time progresses, we can expect that this vocabulary drift will render the earliest documentation more mysterious, and perhaps even misleading.

It seems likely that most old computers do not have their properties sufficiently accurately documented to enable emulators to be written which are accurate enough to run real software. Once written, such an emulator becomes a definitive description of the workings of the machine and deserves a place in the historical record. The fact of its successful execution of original software gives it that credo.

Nostalgia is, of course, a wasting asset as mortality takes its inevitable toll. There is an urgent need to collect old software from garages and probate sales, and get the stuff working while we still can.

4 Beyond Nostalgia

In seeking to put antediluvian software in a historic context, it is vital to provide facilities to enable a computer user of today to have some sense of the experience of using the computers of yesteryear. When we look back to the days when computers had operators, we see that the experiences of an operator were different from those of an end-user. Going back before the days of time-sharing or multi-access, programmers would write code by hand on coding sheets, and the code was then typed onto cards or paper tape by data-preparation staff.

Fig.1: Dennis Ritchie and Ken Thompson and their PDP-11 (reprinted with permission of Alcatel-Lucent USA Inc.)

If we were to show this picture of Ritchie and Thompson (fig 1) to a person whose first experience of computers was a Windows PC or iPad, what would be their understanding of this scene? If we were to show this coding sheet from Leo III (fig 2) to a such a person would they have the first idea of how it was used?

Serial No.	Action	Reference	Item
0			
1			
2			
3			
4			
5			
6			

Fig.2: Coding sheet for Leo III

If we are to give a feel for the computing activities pre-PC, we need to give access to the keys of that era, card punch, paper-tape punch or on-line teletype. Given that keeping these machines working is rather difficult (especially teletypes), we need helpful software emulations. When searching on-line for emulators for teletypes or card punches, we find facilities that emulate the computer end, producing images of paper tape or cards. We need emulation of the user end, with sound effects, and operating at the real speed. We should also have videos of the actual hardware in operation, and put these on-line.

Those who never used such computer systems will not easily appreciate the effort required to get anything to work. I have thought only half seriously of offering a website that runs programs with an over-night turn-round.

5 Software as Part of History

In our preservation of George3, we were able to copy the actual system from a working installation. In particular, we made files that are images of the original magnetic tapes, in such a way that we can reproduce the effect of reading the tape. I do have a copy of MSDOS from 1988. We can (and should) keep this material indefinitely [17]. For the most part, the companies that produced the software of yesteryear have survived even less well than their software. IBM is a notable

exception, and persistent exploration of their website, can eventually tease out historical activity such as SIMH [10]. Understandably, IBM's website is much more directed towards the present and to the future.

Older material has rarely survived in digital form, but nostalgically stored printer listings do turn up from time to time. We have encountered preserved source text in digital form, but equivalent binary code had not been preserved. Our own work [5] has shown that such relics can be brought back to life. It is fair to ask to what extent such a resurrection is only a replica. At the same time we should accept that in a digital world, where copies are perfect, the concept of the original object is no longer really valid. For all that, I suggest that we do the best service to historians of the future by retaining digital copies as close to the original bit stream as we can sensibly do [17].

Whatever form we choose for retention of digital materials, our criterion should be that it "*must allow the recreation of the significant properties of the original digital object, if one assumes that appropriate hardware technology is available*" [18]. On the assumption that an intelligent choice of "significant properties" is made, such a criterion ensures that someone studying the material in the future will not be hampered by a loss of information. Such ambitions point to keeping the original byte-streams in files that are copied onto current technology from time to time. This has the advantage of being a very low cost strategy.

6 Software as a Tool of History

It is only possible for historic software to play its part in history if alongside its preservation we implement software tools to make it accessible to users of current technology. A major weakness of the retention of printer listings of software as historical documents is the difficulty of appreciating just how the software actually operated, either by seeing it run, or by browsing the source text. For KDF9 software we have used software to generate HTML both to give the on-screen appearance that closely resembles the printout from KDF9 data-preparation equipment (viz, Flexowriter), and to provide hot-links that enable the following of subroutine calls in an assembly language program [19]. The technique is readily applicable to pretty well any programming language.

Whether studying software, or merely exercising a bit of healthy curiosity, there is nothing to beat actually being able to run the stuff. CCS activities in this area can be seen on our website (http://sw.ccs.bcs.org), where we have a variety of simulators as free-standing programs, mostly written in C or Ada. SIMH [10] offers facilities for simulation of a number of systems, running on different platforms.

Although the CPUs of early (and not so early) computers tended to be particular to the machine, peripheral equipment showed less variety. Quite early on we designed a data format for storing images of magnetic tapes in files that enables them to be read in a variety of emulation environments. It was the UNIX operating system that brought us the notion that any data stream could be considered as a file. Emulation of a disk drive by a file is trivially easy. For the most part emulations also use files for representation of decks of cards, reels of paper tape or printer listings, with text

editors and browsers to give access to their contents. On-line teletypes can be emulated easily by the increasingly elusive *telnet* command connected to an emulator by TCP/IP. but this gives an overly sanitised view of 1960s on-line computing. I have so far searched the net in vain for a 10 characters-per-second teletype emulation complete with sound effects. Even better would be to have an image of a teletype head, like football results were once presented on UK TV.

In this world where the running of custom software on one's own hardware becomes rarer, we have looked at how best to offer a meaningful experience via a web browser. We have at the α-test stage a system for running KDF9 Algol [20] programs on a webserver. It takes the user's program typed into a window on a form, or uploaded from a file, converts it into KDF9 paper tape code and presents it to the genuine KDF9 Algol compiler running on an emulated KDF9. The Algol program is shown to the user as it would have appeared on a KDF9 Flexowriter (Rolls-Royce teletype), and the line printer output is also delivered to the web browser. All this running on a Raspberry Pi, little bigger than a credit card.

We have already alluded to the indigestibility of much of the documentation of the 1970s era, and to its existence only in paper form. Our use of OCR software in this area has been an invaluable aid, leading us to look forward to a time when its accuracy will eliminate the need for proof reading. Just as Google's PageRank software [21] has tamed what seemed like an amorphous sprawl in the early days of the World Wide Web, so we can hope that software will be developed that deals with the paper legacy from computing's earlier phases.

7 Long Term Preservation

Although digital material may seem somewhat ephemeral, its long term retention is actually much easier than for more tangible historical material. I have personally suffered the loss of historic listings as a library ran out of storage space. If our historic software and associated software tools are to persist into history, it is vital that we have a reliable strategy for their preservation in digital form. This is covered in reference [17], but here are some key points.

Despite the rapid development of digital technology there are aspects of IT that endure. The key to digital longevity lies in relying only on such enduring aspects. It would appear that the byte is here to stay, and that a sequence of bytes will be able to be preserved for a very long time − just plain files periodically copied onto current media, ideal for keeping the original software in a byte-stream whose contents never change. We have no faith on long-lived media. Even if the medium is indestructible, the device to read it is not. Technological evolution is tracked by maintenance of the access tools. It is important that these tools (e.g. emulators) are written with a view to longevity [22]. Reference [23] (cited in [22]) hints that a subset of C may be the best choice for writing emulators. There is so much important software written in C, that we can be confident that it will still work for many decades to come. A recent survey of emulators held by the CCS showed those written in C to be the most durable, slightly more so than those written in Ada. For documentation, we can confidently use HTML, and JPEG, which are non-proprietary open standards. Adobe have started

hinting that PDF readers are reaching the end of the line. Long-term validity of proprietary data formats is rarely in the interests of their owners.

8 Conclusion

We need to pay more attention to software. Its importance to the history of computing has been underestimated for some time. We need to keep old software, both source and binary, and to write new software to enable our successors better to appreciate the systems of the past. As an illustration of the historical significance of software let us return to Fig 1. There is a lot of hardware visible in this picture, but the chances are that little of it survives today, and the company that made it certainly does not survive. Of the people in the picture Denis Ritchie died in 2011, but both he and Ken Thompson survive in Wikipedia. The software in this picture is invisible, but its legacy is ubiquitous today, and likely to remain so long into the future.

As well as keeping historic software as part of the historical record, we also need to have software tools which both enable the serious study of that historical record, and permit some understanding and appreciation by the casual visitor whether in a museum or on-line. Some of today's casual visitors can be turned into tomorrow's enthusiasts by appropriate software, both old software preserved on current hardware, and new software written to give access to the old. Much of the marshalling of this software relies upon the nostalgia of ageing veterans.

Acknowledgments

The rescue of software from printer listings has involved heroic efforts on the part of Brian Wichmann, Graham Toal, Roderick McLeod, Bill Findlay, Ray Smith, Geoff Cooper, Ken Kemp, Chuck Knowles, John Daines, Tony Jackson, Dave Jones.

References

1. ubuntu, http://www.ubuntu.com/ubuntu
2. Raspberry Pi, http://www.raspberrypi.org
3. Spoor, B.: Problem Solving with George 3 Today. Resurrection ISSN 0958-7403 no. 36 (2005), http://www.cs.man.ac.uk/CCS/res/res36.htm#e
4. Holdsworth, D.: George3 − Emulation of the ICL 1900, http://sw.ccs.bcs.org/CCs/g3/
5. Holdsworth, D.: Rescuing Software from Lineprinter Listings. Resurrection ISSN 0958-7403 no. 57 (2012), http://www.cs.man.ac.uk/CCS/res/res57.htm#e
6. B-EM, A Freeware BBC Micro Emulator for DOS, Windows and Mac OS X (2012), http://b-em.bbcmicro.com
7. Hedstrom, M., Wheatley, P.R., Sergeant, D.M., et al: The CAMiLEON Project, http://www2.si.umich.edu/CAMILEON/
8. The Hercules System/370, ESA/390, and z/Architecture Emulator, http://www.hercules-390.eu
9. West, J.: pdp11 home page, http://www.pdp11.org

10. Jones, M.T.: Emulation and computing history (2011), http://www.ibm.com/developerworks/opensource/library/os-emulatehistory/
11. Black, A.: Running George 3 on a Raspberry Pi, Blog post on DesignSpark (2013)
12. Holdsworth, D.: Leo III Resurrection (2013), http://sw.ccs.bcs.org/leo/
13. English Electric.: KDF9 Director Manuals, http://sw.ccs.bcs.org/KDF9/directorManuals/manuals.htm
14. IBM.: IBM 360 Principles of Operation, http://bitsavers.trailing-edge.com/pdf/ibm/360/princOps/A22-6821-0_360PrincOps.pdf
15. Hock,A.A.: Leeds University User Manual - section E (1976), http://sw.ccs.bcs.org/CCs/g3/LeedsDoc/sect-e.htm
16. Alcock, D.: Dave's Green Card Collection (2004), http://planetmvs.com/greencard/
17. Holdsworth, D.: Curation Reference Manual, Digital Curation Centre (2007), http://www.dcc.ac.uk/resources/curation-reference-manual/completed-chapters/preservation-strategies
18. Holdsworth, D. Sergeant, D.M.: A Blueprint for Representation Information in the OAIS Model, 8th NASA Goddard Conference on Mass Storage Systems and Technologies (2000), http://www.storageconference.org/2000/papers/D02PA.PDF
19. Holdsworth, D.: Whetstone Algol resurrection (2011), http://sw.ccs.bcs.org/CCs/KDF9/walgol.htm
20. English Electric: Algol Programming (c1963), http://www.findlayw.plus.com/KDF9/EE%20KDF9%20Algol%20Manual.pdf
21. Page, L., et al: The PageRank citation ranking: Bringing order to the Web, Stanford (1998), http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf
22. Holdsworth, D., Wheatley, P.R.: Emulation, Preservation and Abstraction RLG DigiNews, Volume 5 Issue 4, (2001), http://worldcat.org/arcviewer/1/OCC/2007/08/08/0000070511/viewer/file3149.html
23. Holdsworth, D.: C-ing ahead for digital longevity (2001), http://www.leeds.ac.uk/CAMiLEON/dh/cingahd.html