

Craft tradition in the history of computer programming 1974 - 2004

A position paper for:

TeamEthno-Online

<http://www.teamethno-online.org/>

Ian Martin

The Open University

i.martin@open.ac.uk

Introduction

Relatively little has been written concerning a history of the work of computer programmers. Historians of computing have so far tended to focus their efforts on chronicling advances in computer hardware, providing biographical accounts of computing pioneers, or detailing the economic and social implications of widespread computer adoption. When a software perspective has been taken this, for the most part, has focused predominantly on finding a family-tree type history of programming languages. The few history of computing textbooks that provide a general overview of the field, for example Ceruzzi (1998) and Campbell-Kelly & Aspray (2004) dedicate relatively little space to software developments. Even Campbell-Kelly's (2003) book devoted to the history of software doesn't really concern itself with the processes and people involved in software production. Levy's (2001) look at hackers as 'heroes of the computer revolution' is closest to a history of computer programming from a labour perspective, but this by design tends to focus on authenticating the legends, and possibly myths, of the early hacker celebrities of the programming world.

Nevertheless, there has been renewed interest from some quarters in writing an academic history of the people and processes involved in computer programming. Just what form that history might be has been the subject of spirited debate. The Dagstuhl Seminar 9635 on History of Software Engineering (Brennecke & Keil-Slawik, 1996), Mahoney (2004), and Zwerman (1999) are three key sources here that suggest what that history might be. It is from out of these debates that a credible proposal of writing a history from a labour perspective has emerged. This alternative "way of knowing" (Pickstone, 2000) the history of computer programming is one that I intend to take forward with an overall aim of providing some fresh insights that will contribute towards a broader history of computing.

The majority of the debates concerning software history have made use of the term software engineering, generally because they take as a common starting point the North Atlantic Treaty Organisation (NATO) Software Engineering conference of October 1968 that first popularised the term. However I am intentionally making use of the older term, computer programming, in this work. The name software engineering was deliberately chosen by NATO at that time to be provocative in order to stimulate debate. The term computer programming, whilst not without its own connotations, has a wider currency and is equally applicable to endeavours made by

the hobbyist in the home as it is to large-scale business developments. It also provides a starting position pre-dating both the terms software and software engineering from which to chart the progress that has been made by interested parties in defining and stratifying the discipline. It is these attempts to define and professionalize the practice of computer programming away from a celebration of the craft skills of the individual that have motivated me to find a history of the occupation as one that can be seen to uphold certain craft traditions.

Previous ethnographic research investigating the working conditions of programmers has uncovered occupational practices in information system (IS) development that rely on, and celebrate, the craft skills of talented individuals and small teams. Previous research in this area (Martin, 2005) has uncovered strong suggestions of craft skills and knowledge being essential to IS development projects. Whilst the focus in this qualitative study was on the concomitant long hours culture, it does offer a useful previous insight from which to trace the historical development of those craft skills. Nandhakumar and Avison (1999) also posit that these craft skills are often masked by IS development methodologies that create a 'necessary fiction' of control. Software development as a craft rather than an engineering discipline is also one that programmers who write for other programmers appear to have a strong affinity. Indeed, McBreen (2002) has produced what amounts to a manifesto for software craft that has been generally well received by many in the programming community. Similarly, Graham (2004) aligns computer programming with the creative activities of painting, writing and architecture. Representative of much of this ilk is Brooks' seminal *Mythical Man Month* (1995). Throughout this influential work Brooks refers strongly to the creative nature of programming and talks of the 'joys and woes' inherent in the craft.

Research Position

My research aims to construct a history of computer programming from this perspective that views much of programming labour as craft. It is important to note that the word craft has multiple meanings that are heavily dependent on context. These meanings can be used internally by computer programmers and externally by other stakeholders to structure perceptions of the nature of programming work. Uncovering attempts to shape others' readings of programming work will form an important part of my research, but here I use the term craft specifically to refer to those kinds of craft principles embodied by the Arts and Crafts Movement of the late 19th century. This movement was championed in Britain as a reaction to the negative impacts of the Industrial Revolution by those such as John Ruskin and William Morris. While some like Ruskin saw it as a romanticised return to the ideologies of the medieval craftsmen, others such as Marx and Engels embraced its socialist ideals as a realistic opportunity for the worker to readdress the power balance shifted in the class struggle of the Industrial Revolution. For some in the movement craft meant anti-mechanisation, but others like the US architect Frank Lloyd Wright saw mechanised tools as offering new opportunities for creative design as long as they remained under the control of the worker. There is not space to go into a detailed discussion of the Arts and Crafts Movement here, suffice to say that it is multifaceted and should not be seen as merely a sentimental desire to return to better times gone by, or some kind of Luddite frustration at technology's impact on skilled work. Adams' (1997) book neatly summarises the principles behind the British and US Arts & Craft Movements and is a useful introduction to the subject.

Where the fruits of computer programming labour may differ from those of other crafts is that a computer program is an intangible artefact. In some ways this could be seen as similar to the issues concomitant with Turkle's idea of (2005, p.61) 'opacity'. This poses an interesting question regarding handiwork and craft that needs to be addressed. Does an intangible product negate the idea of craft, or can an intangible product be seen as a craft product if it has the necessary tangible evidence of its production? For example writing is often referred to as a craft, yet an arrangement of words cannot be touched or felt. It is possible for writing that the formation of the words into a finished product such as a book provides the necessary tangibility. Music could provide something that is more akin to the abstract nature of a computer program. There are many musicians who have embraced the craft tradition and view their musicianship as very much a craft activity; see for example, Fripp (1989). It may be that in this case it is their public sites of practice, or the tangible nature of their instruments used as tools, that provide the semblance of physicality to their end product. This is an end product that is now becoming increasingly digitised and divorced from its traditional physical presence. The reaffirmation of a craft tradition may be a reactionary move against the growing elusiveness of this tangible end product. Intangible artefacts have always been a feature of computer programmers' work, but it is possible that certain infrastructural changes have had an impact on the perception of this tangibility. The increasing complexity of software applications necessitating a greater numbers of developers per application, the move away from mainframe to distributed computing, and the current prevalence of web-based applications may have all increased the distance between artefact and creator. A craft approach could be seen as an attempt to put the craftsman very firmly back at centre stage.

Evidence will be gathered to show existence or otherwise of a craft tradition in computer programming by drawing from sources recorded in written, oral, visual and digital formats. This will allow for a close following of the progress programmers, and other interested stakeholders, have made in the task of defining what it means to be a computer programmer. Whilst I accept that there is undoubtedly a strong argument for following the progress of this debate through academic journals articles and conference papers, I am deliberately choosing to focus more strongly on programmers' own representations of themselves than other authors have perhaps done so far. Interestingly, there have been claims from others that computer programming is best viewed as an art (Knuth, 1974); an application of a science or engineering discipline; or something best aligned with industry and business as a branch of learning. See Mahoney (2004) a comprehensive review of a number of positions regarding the nature of programming work. It is not my intention to argue the case for a history of computer programming as a history of a craft tradition as in any way superior to these other models, but rather to provide a convenient lens with which to view these other attempts to shape just how computer programming should be viewed as an occupation or profession.

This idea of continuing unsuccessful attempts to form a profession that would serve as a home for the *discipline* of computer programming is one of six pillars upon which this history is to be built. These pillars are:

- *discipline*
- *identity*
- *knowledge*
- *reward*
- *power*
- *structure*

Each one offers a convenient sub topic within the overarching craft tradition framework to provide a thematic rather than chronological history of computer programming from 1974 to the present day. These themes will be intentionally left open for revision as the research progresses and the thematic divisions will be revised and replaced as necessary. The next phase of this research project is the production of a comprehensive literature report, with one eye on these themes that critically evaluates relevant historiographies concerned with software development, craft tradition and craft knowledge, and the formation of the professions.

References

- Adams, S. (1997). *The Arts & Crafts Movement*. Grange.
- Brennecke, A., Keil-Slawik, R., and Parnas, D., editors (1996). *Position Papers for Dagstuhl Seminar 9635 on History of Software Engineering*.
- Brooks, F. P. (1995). *The Mythical Man-Month: Essays on Software Engineering*. Addison-Wesley, 20th anniversary edition.
- Campbell-Kelly, M. (2003). *From Airline Reservations to Sonic The Hedgehog: A history of the software industry*. MIT Press.
- Campbell-Kelly, M. and Aspray, W. (2004). *Computer: a history of the information machine*. Westview Press.
- Ceruzzi, P. E. (1998). *A History of Modern Computing*. MIT Press.
- Fripp, R. (1989). *A preface to guitar craft*. Available at: <http://www.guitarcraft.com/?np=5&&id=6>, last accessed 23 February 2006.
- Graham, P. (2004). *Hackers & Painters*. O'Reilly.
- Knuth, D. E. (1974). Computer programming as an art. *Commun. ACM*, 17(12):667–673.
- Levy, S. (2001). *Hackers: Heroes of the Computer Revolution*. Penguin.
- Mahoney, M. S. (2004). Finding a history for software engineering. *IEEE Annals of the History of Computing*, 26(1):8–19.
- Martin, I. (2005). *Long hours and stress in the UK's IT profession: has the European Union's Working Time Directive offered relief?* Gender, Work & Organization 4th International Conference.
- McBreen, P. (2002). *Software Craftmanship*. Addison-Wesley.
- Nandhakumar, J. and Avison, D. E. (1999). The fiction of methodological development: a field study of information systems development. *Information Technology & People*, 12(2):176–191.
- Pickstone, J. (2000). *Ways of knowing: A new history of science, technology and medicine*. Manchester University Press.
- Turkle, S. (2004). *The second self: Computers and the human spirit*. MIT Press.
- Zwerman, W. L. (1999). Profession/occupation without a history. *IEEE Annals of the History of Computing*, 21(1):66–70.