HUMAN TECHNOLOGY

An Interdisciplinary Journal on Humans in ICT Environments ISSN: 1795-6889

www.humantechnology.jyu.fi Volume 11(1), May 2015, 4–29

AN ORAL HISTORY OF PROGRAMMING PRACTICES: GENDER AND AGE DYNAMICS AND DIGITAL (DIS)ENGAGEMENT AMONG COMPUTER PROGRAMMERS IN FINLAND

Antti Silvast
Science, Technology and Innovation Studies
The University of Edinburgh
Edinburgh, United Kingdom

Abstract: *Since the 1980s, educational policies in many countries have aimed at improving the computer literacy and programming competencies of the population. Over the same period, the possibilities that people have seen regarding programming and everyday programming practices have emerged as an area of strong interest within historical scholarship. The paper contributes to these discussions by drawing on techniques of oral history to focus on programming hobbies and practices in Finland. Examining data from a massive survey of computer hobbyists (*N = *1,453) and their recollections about personal computer use (largely during the 1980s), the paper gathers new information on what leads to people's pursuit of or interest in programming and how their programming habits have changed over time. The study links together the gender and age dynamics in programming and shows how the respondents not only engaged with but also could become disengaged from programming for various reasons.*

Keywords: *oral history, programming, online survey, gender, digital disengagement.*

INTRODUCTION

According to historical scholarship (Abbate, 2012; Mahoney, 2008), the traditional history of computing has centered on two themes: the study of computer hardware and of the software industries. Much of the history concerning computer programming seems to be subsumed under these same research interests. There are several studies about the programming languages used historically by computing centers to control the behavior of computer hardware (e.g., Benediktsson, 2009; Nordal, 2009). Much discussion has focused also on computer software industries, their development over time, and the profession of programming (e.g., Gram et al., 2005; Oinas-Kukkonen, Similä, Pulli, & Kerola, 2009). Over the past decades, however, considerably more interest has emerged in programming as an everyday

DOI: http://dx.doi.org/10.17011/ht/urn.201505061738

VERTAISARVIOITU KOLLEGIALT GRANSKAD PEER-REVIEWED www.tsv.fi/tunnus

activity. Princeton historian Michael S. Mahoney named this focus "communities of computing": groups that "saw different possibilities in the computer" and "had different experiences as they sought to realize those possibilities" (Mahoney, 2008, p. 10). Such communities are found among scientific, business, and government organizations (Mahoney, 2008), but have been seen in various other situations as well. The possibilities that people see in programming is a recurring theme in classic studies about children and adolescent coders (Turkle, 2005) or computer hackers who mastered computers through programming (Levy, 1994; Wajcman, 1991). People's programming experiences also are central in the study of the cultural histories of various topics from legacy computer platforms (Maher, 2012) to do-it-yourself programming (Saarikoski, 2005; Wasiak, 2014). Furthermore, public and educational policies frequently border on how people experience and perceive programming. The notion of computer literacy, including the knowledge and mastery of programming as part of general education, originated in the 1980s in many parts of the world (Saarikoski, 2011; Schofield, 2014). Recent coding campaigns for women and children (e.g., Hello Ruby,[1] Koodi2016,[2] Lasten koodikoulu,[3] and Rail Girls[4]) have expanded this discourse by noting how everyone, regardless of age and gender, should have the possibility to become excited about learning to program.

This focus of the majority of studies on programming is significant. The studies generate new understanding about programming in everyday life and, hence, also suggest how learning to code can be enhanced. However, the contexts influencing a programming hobby and its long-term dynamics still merit more attention. In particular, gender, age, and people's engagement with programming—or, more commonly, lack of engagement—often are linked together yet easy to misread. In this article, I seek to address these issues by posing and answering the following research questions: (a) What leads to people's pursuit of or interest in programming, and (b) How does the practice of programming evolve over time? The data are drawn from a 2013 massive online survey of Finnish computer hobbyists[5], a portion of which included their recollections about programming.

According to the political discourse of computer literacy, gaining knowledge of programming gives essential abilities for the future information society (Saarikoski, 2011) and provides ample employment opportunities in a nation's information and communication technology industries (Opetus- ja kulttuuriministeriö, 2014; Sajaniemi & Kuittinen, 2008). The findings of many studies do not support such aspirations, in that, although almost anyone should be able to attain and share in the benefits of programming literacy and many do become excited about coding, this still seems to occur only within a portion of the population.

A well-known factor influencing programming disposition is gender. For several decades, women have been underrepresented in programming education in many countries (Jones & Burnett, 2008; Wajcman, 1991). In the last 5 years, however, the number of women graduating with degrees in computer science, for example, at Stanford University in the United States of America, has increased. Nevertheless, over a longer period, the number of women in the American technology sector has declined (Kantor, 2014). Scholars have attributed the relative lack of women in computer science to the masculine culture of computing and programming (Saarikoski, 2004, pp. 167–189). Programming education in schools exemplifies this with its close ties to typically male-dominated school subjects such as mathematics and engineering, according to feminist scholar Judy Wajcman (1991, p. 152). It has been claimed that women prefer more interactive and emotional programming styles (Turkle, 2005) and seek to distinguish themselves from (male) programmers for image reasons (Nordli, 2001). This gap between male and female styles

of programming has a long historical precedence: 40 to 50 years ago, most hands-on programmers of computers were women, and men were responsible for the systems analysis that formulated and planned the processes that would be automated through programming (Abbate, 2012, pp. 42–43). Thus, early programming practices typically were not a male—but rather a female—area, mainly viewed as low-status clerical work (Wajcman, 1991, p. 158).

However, to many other scholars, the notion that men utilize computers rationally and women use them more interactively and emotionally becomes more difficult to sustain. Consumption and media studies have shown repeatedly that women have to justify their computer use rationally, whereas men seek to utilize computers for experimentation and leisure time (see e.g., Buse, 2009; Naskali & Silvast, 2014). These statements and the ones above are not at odds. Rather, they seem to reflect the need for more contextual information. To determine why people learn or do not learn any form of digital literacy, it is useful to ask what personal motivations, experiences, emotions, peers and role models, and other factors led to the adoption of or disengagement from computing (Bowen, 2011). To begin to understand this issue, more information is needed on the role gender plays in people's personal histories of encountering and adopting programming in everyday life.

Another gap in knowledge concerns programming and age. Many researchers have identified an age bias (Bowen, 2011) in discussions about digital literacy. They have noted that although the share of older people using digital technologies is growing (Olphert & Damodaran, 2013), research still is absorbed in children's and adolescents' computer use (see Buse, 2009). Appropriately, several scholars have explored programming and its subcultures among young people in their twenties, as teenagers, and even as children (e.g., Maher, 2012; Nordli, 2001; Saarikoski, 2005; Reunanen & Silvast, 2009; Turkle, 2005). Researchers investigating computing and old age (e.g., Bowen, 2011; Buse, 2009; Olphert & Damodaran, 2013), on the other hand, typically have not discussed programming either because the respondents did not code or because it did not surface in the analysis. Whether older people have chosen to become programmers or not, they might have specific and original conceptualizations about coding as active, passive, leisure, or work activities (Buse, 2009). Such issues are absent in many studies and, therefore, this paper seeks to address them, as allowed by the data.

A final point I want to raise about programming and its uptake concerns what scholars have called "digital disengagement" (Olphert & Damodaran, 2013), which, in short, demonstrates a decision to stop using information and communication technologies. Often, the assumption is that once people have the equipment, motivations, and skills to program, they will continue doing it in the future. This results in a static view and provides relatively little consideration for how practices of programming evolve over time and possibly even cease. However, studies about how computers and other products become embedded in everyday life have had a different premise for a long while. Their analyses typically started with explaining how new technologies were obtained (Haddon, 2011, pp. 312–313). But, sociologist Turo-Kimmo Lehtonen (2003, p. 381) added that, eventually, most technology is disposed of, and not merely because it seems outdated. For example, people can become disengaged from using the Internet for many reasons: the increased complexity of technology, the loss of technical support from other people, financial restraints, or its declining relevance in their lives (see Olphert & Damodaran, 2013, p. 567). With this article, I contribute to research on digital disengagement by examining how and why people cease programming, which was a salient finding in the data.

The inspiration for my analysis of these three topics—programming in relation to gender, age, and cessation—originates from the concept of domestication to which I add themes from historical research. In science and technology studies, the domesticating technology perspective concerns how people use technologies in ordinary life and what meaning they attribute to its use. Domesticating technology refers to a slow process of becoming first interested in new technology, assessing its need collectively and individually, fitting it into existing social and material relationships, and its eventual disappearance from daily life (Aune, 1996; Haddon, 2004, 2011; Lehtonen, 2003). Many of these studies are concerned with the domestication of particular devices and artifacts—the mobile phone or home computer, for example. However, in this paper, I use the concept in a less orthodox way in order to ask how programming practices themselves provided the means for domesticating computers in everyday life.

The materials used in the multidisciplinary domestication studies have varied from surveys and quantitative materials to in-depth qualitative case studies (Haddon, 2004). However, particularly common are qualitative methods, such as fieldwork and individual and group interviews (e.g., Aune, 1996; Bowen, 2011; Buse, 2009; Lehtonen, 2003). This follows from theoretical interests: Rich qualitative information is important to understand how technologies and their adoption relate to other parts of people's lives, including use of time, routines, and household arrangements. The data behind this paper, however, come from a massive online survey of Finnish computer hobbyists. The material was selected because it offers an original and broad overview of programming practices in relation to other likewise wide themes such as gender, age, and evolving hobbies (although many respondents were also professional programmers, we inquired mainly about their hobbies). The paper adds to in-depth qualitative case studies by mapping the complexity and basic characteristics of taking up programming as a hobby. For this purpose, a massive online survey is a highly appropriate research method (Suominen, 2014).

The survey data do not offer the possibility to seek correlations or make generalizable conclusions about programmers in Finland (cf. e.g., Jones & Burnett, 2008). This is because the sample is a nonrepresentative, theoretical sample where groups are selected "according to their relevance to the research questions, the researcher's theoretical position, and analytical framework" (Gobo, 2008, p. 112). When posting and advertising the survey online, my colleagues from the full survey and I expected that respondents would be individuals who were enthusiastic about computing and programming and, therefore, not be a random sample of the Finnish population (Naskali & Silvast, 2014), which proved true. The responses do, however, offer a rich and broad picture of people's pursuit, interests, and cessation of programming. I interpret the data with tools from oral history research (see Misa, 2009), which is the study of recent history through people's personal memories. In this study, the participants gave an account of their lives during a period when home computers first came to market. Some degree of programming skill was inherent even for ordinary, presumably leisure-time use, such as running games on these computers (Swalwell, 2012). This almost ubiquitous and mundane "contact with the 'bare machine'" (Turkle, 2005, p. 7) offers a suitable vantage for the study of how people saw programming and its possibilities at the time.

In the remainder of this article, I draw on the survey data to consider these issues. In the next section, I outline the empirical materials in greater detail and explain the methodology, an oral history approach combined with a qualitative grouping of responses and some quantitative

calculations. In later sections, the survey results are explored more thoroughly, followed by a discussion and suggestions of starting points for future studies.

MATERIALS AND METHODS

The data for this article are part of a 2013 online survey of Finnish computer hobbyists and users. Information about the survey was spread mainly online through social media (including Facebook), on Web sites dedicated to gaming and computing, on academic mailing lists, and through other communication channels of Finnish universities (Naskali & Silvast, 2014, p. 11–12). The survey was completed by 1,453 respondents. Men were overrepresented in this survey (77%). The respondents were also relatively young adults and highly educated: Most were 25 to 39 years old (i.e., born in the 1970s or 1980s), and 80% of the women and 58% of the men had a university-level degree (Table 1). To contrast these levels to the general Finnish population, in 2013, 9% of women had a higher academic degree and 11% a lower academic degree, while

Table 1. The Profile of the Survey Respondents.

	Women	Men	All
N[a]	329 (23%)	1,119 (77%)	1,453 (100%)
Age[b]			
10–19	7 (2%)	24 (2%)	31 (2%)
20–29[c]	117 (36%)	246 (22%)	365 (25%)
30–39	110 (33%)	491 (44%)	601 (41%)
40–49	53 (16%)	211 (19%)	264 (18%)
50–59	28 (9%)	95 (8%)	123 (8%)
over 60	13 (4%)	52 (5%)	65 (4%)
Education[d]			
Comprehensive education	4 (1%)	38 (3%)	42 (3%)
Upper secondary/ vocational education[e]	62 (19%)	429 (38%)	495 (34%)
Academic education	262 (80%)	645 (58%)	907 (62%)

Note. Full demographic information was not given by some respondents. Therefore, they will only be included in analyses where sufficient demographic information is available.
[a]Five of the respondents did not disclose their gender. [b]Four respondents (1 female, 3 of unknown gender) did not disclose their age. [c]Two respondents in this age group did not disclose their gender. [d]Nine respondents (1 female, 7 males, 1 of unknown gender) did not disclose their level of education. [e]Four respondents with upper secondary/vocational education did not disclose their gender.

the share of men with these same degrees was 8% and 9% (Tilastokeskus, 2013). Of Finns with a higher degree in technological disciplines, around 80% are men and 20% women (Tilastokeskus, 2012), which coincides closely with the gender division in this survey.

The survey consisted of 29 open-ended and 16 multiple-choice questions,[6] in Finnish, covering personal information, first computing experiences, current ownership of or access to computers, various past computer uses and computer-related hobbies, and memories of past use. The respondents were asked to respond with the frame of their hobby. Four questions concerning programming were asked:

- When did you first become acquainted with programming, and how old were you at that time?
- What prompted your interest in programming?
- What programming languages/environments have you used during your hobby?
- Could you provide a few examples about your use of programming skills: What have you programmed, and have your interests changed over the years?

In addition to these questions, the survey included a question about the totality of the respondents' computer hobbies. The hobbies listed as options were programming, gaming, producing content, computer graphics, composing music, writing, fan fiction, hardware modifications, and collecting classic games, hardware, and other (which respondents could specify). The term *hobby* was defined in this survey as computer uses that are voluntary and relate to your own areas of interest.

The methodological inspiration concerning the gathering and interpretation of the data came from oral history: the collection of materials about recent history through people's personal memories (Fingerroos, 2004; Misa, 2009, pp. 2–5; Naskali & Silvast, 2014, p. 13; Ukkonen, 2000; Ullberg, 2013, pp. 13–21). Since the 1980s, oral history has been deployed widely in historical and folklore studies. What memories about recent history capture, in particular, are the characteristics of common everyday life, that is, how the respondents themselves recollect and interpret things that happened to them.

The methodology in this paper shares some commonality with the project Mass Observation Archives,[7] which first spanned the early 20th century and was revived in 1981. In this recurring British study, participants are asked to keep diaries and respond to surveys concerning various aspects of their everyday lives. The observation project has been also critiqued methodologically for its combination of different types of information in a less systematical way than is usually the case in a single case study (Pollen, 2013). Likewise, in historical scholarship, personal memories and academic research have had a strained relationship: Historical research has been held to a standard of more rigor in regard to the objectivity of what really happened, with people's memories of events only as "subjective, emotional, and empathetic" (Ullberg, 2013, p. 17). Indeed, it is now widely agreed that memories do not offer a view on how things actually transpired. To this end, the memory is too prone to forgetting and misinterpreting, and memories can be transformed and reshaped by later experiences and current beliefs (Ukkonen, 2000).

Nevertheless, it can be argued that people's memories offer one substantial subjective source of historical knowledge among many others. Assessing the reliability of information in the oral history context means acknowledging that the material consists of conceptualizations and interpretations that have been shaped by particular perspectives of the informants (Misa, 2009, p. 3). As folklorist Taina Ukkonen (2000) summarized, recollecting is both reminiscing the

past (reconstructing) and interpreting it. Indeed, interpretations may be inherent to any source of historical material, whether academic or nonacademic (Fingerroos, 2004). At the same time, people's own observations provide important first-hand accounts about daily life that would be difficult to capture with many other research techniques (Pollen, 2013).

With these issues in view, oral history offers a rich resource to the research presented in this paper for several reasons. For one, people who used computers for programming in their childhood and youth in the 1980s are now in their 30s and 40s. Therefore, they are still easily available to reminiscence about the recent past. Alongside traditional documentary material, including computer magazines and books, and computing artifacts, such as software (Mahoney, 2008), people's memories are an appropriate source for assessing how programming manifested in ordinary life, what people did through it, and why. In the programming section of the survey, Question 1, regarding one's age when and the year of first encountering programming, yielded numerical data to analyze by drawing on normal quantitative methods, such as averaging and cross-tabulation against gender and current age, in particular. The final three questions in this section were open-ended: the interest in programming (Question 2), programming languages and environment (Question 3), and programming projects and one's change in interest (Question 4). These questions gathered a total of 2,787 qualitative answers.

Within the three main categories (based on the final three questions above), the responses were analyzed by reading each in depth and, based on the content, similar responses were grouped into subcategories (which were added as needed). For example, if a respondent told she wanted to master computers through programming, then we added this as a new subcategory if the response did not relate to an earlier added subcategory. However, this technique is not entirely the same as an inductive, bottom-up grounded theory or coding approach that is typical in qualitative research (see Gobo, 2008, pp. 225–258). As opposed to grounded theory, which is meant to start without strong a priori social theories or expectations, the structure of the response data had been preconfigured in several ways before my analysis began. First, the key organization of the findings (into programming interest, programming languages, and programming projects) were not extracted from the data, but derived primarily from the survey questions, in order. Second, when drafting these questions, the research project behind the survey had various analytic interests, including the historical significance of computer hobbies, the computer hobbies in broader communities and not merely among enthusiastic computer hackers, computing in everyday life and its domestication patterns, and the interrelationship between programming and gaming. These topics, too, guided my analysis to a great extent. Gender and age, although not explored directly as a research question, are inherently important for the cultural history tradition theme of the project (see, e.g., Saarikoski, 2004). The survey inquired gender and age as background variables as is typical (Suominen, 2014); hence, once again, the significance of age and gender did not emerge from the data but was known beforehand as an issue that needed analysis.

Yet, in this paper, I do not merely seek to answer preconceived survey questions deductively. The written answers to the open questions tended to be long, perhaps because people have become accustomed to writing long texts with the computer, as cultural historian Jaakko Suominen (2014) notes. Many specific results in what follows were based on reading the texts and, hence, were found more inductively, for example, the cessation of hobbies and people's rich reasons for starting programming. The descriptiveness of these answers forms the main part of the analysis

that follows. In summary, the research design combines inductive and deductive research styles, but leans slightly more to the deductive than is typical in qualitative case studies.

REMINISCING PROGRAMMING

First Exposure to Computer Programming

Of the 1,453 respondents to the full survey, 990 (68%) indicated that they had tried programming at some point in their lives. This comprised 854 male respondents (76%) and 135 female respondents (41%) who had programmed at least once. However, an even smaller number of all survey respondents (38%) said that programming has been or is currently their hobby. The difference between these results possibly could be explained—at least in part—by the wording of the question. Although the survey questions were designed to reveal whether programming is familiar to the respondent, the actual question generally asks about all of one's computing hobbies and defined a hobby as those computer uses that are voluntary and relate to one's own areas of interest. As a result, a person may have become familiar with programming yet did not find it interesting and/or chose not to pursue it, as I discuss below. That said, programming was among the most noted hobbies for women, second only to fan fiction production being female-dominated at 63%. The most male-dominated areas of computer use, according to the results, were hardware tinkering and music making (both 95% male) and collecting classic games and computers (93% male). However, compared to the other computing uses sought in the survey, this large figure is on par with the popularity of digital gaming.

The 990 who had tried programming are the focus of the balance of this paper, although not all of these respondents are represented in every analysis because some did not provide complete demographic information. Of the 990 who had tried programming, 86% were male, a higher proportion than in the survey in general (77% male). Figure 1 provides information on the male respondents' reported age of their first encounter with programming in any form (i.e., aware of the process, trying the process, or taking it as a hobby); Figure 2 provides similar information for females.

For the men (Figure 1), similar proportions of those who had or had not tried programming and those who programmed as a hobby exist across all age ranges. As expected, not all respondents had ever tried programming, and not all who tried it became hobbyists. However, I did not anticipate that there would be no notable variation between ages.

For women (Figure 2), however, the situation was different. The age distribution showed several visible gaps: Very few female programming hobbyists were over the age of 50, and not many more over 40, in comparison to the number of female respondents within those age groups. Experimenting with programming was distributed more evenly among the age groups, although women over 60 who have tried programming are few. As for the younger respondents of the seven women in the 10 to 19 age range who responded to the survey (admittedly few), only one had ever tried computer programming. These observations are tentative due to the small number of female respondents. However, they do suggest that if women have been discouraged from programming, as many researchers have claimed, this may relate only to women of specific ages—roughly under age 20 or over 50.

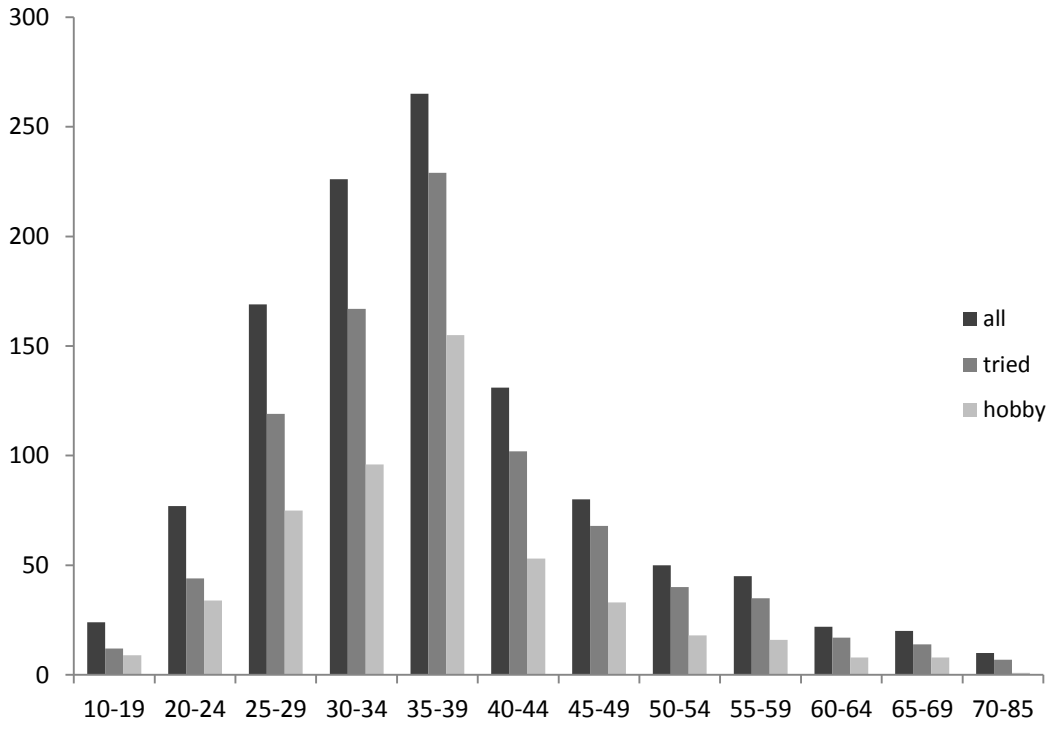

Figure 1. Frequency distribution of the age of all men in the survey (n = 1,119), those who had tried programming (n = 854), and those who reported a programming hobby (n = 506).

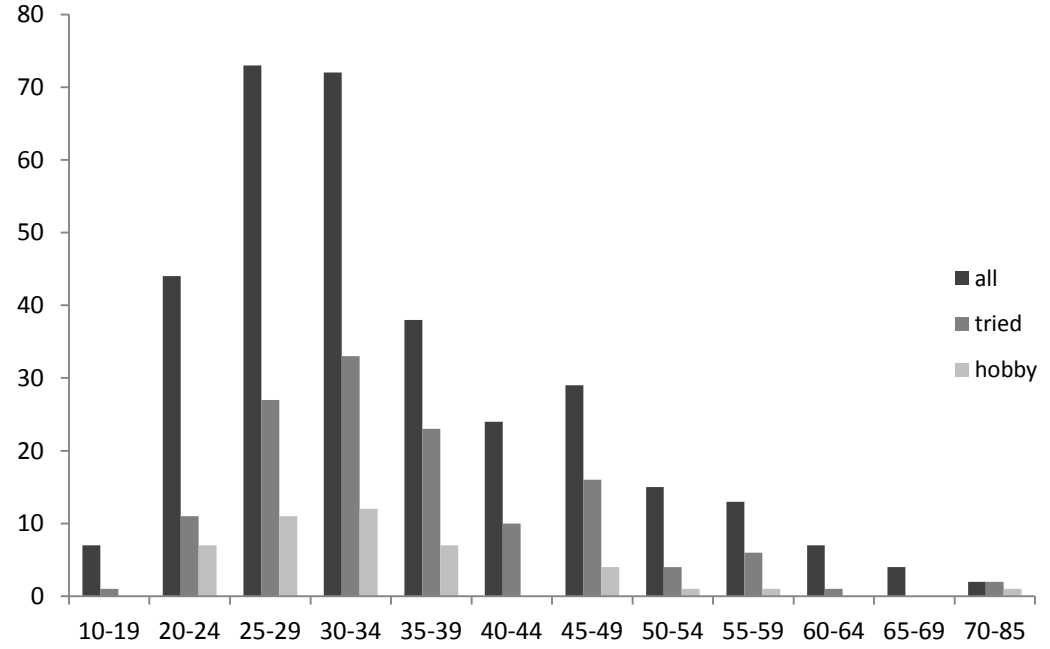

Figure 2. Frequency distribution of the age of all women in the survey (n = 328), those who had tried programming (n = 134), and those who reported a programming hobby (n = 44).

In questions specific to programming, the respondents were asked for details about their hobbies: at what age they first tried programming, how their interest was sparked, what programming languages they have used, what kind of programming projects they have implemented, and if there were any changes in their programming disposition. Here, change also could refer to discontinuing a practice. One in five respondents claimed to be no longer programmers, thus disengagement with programming became an important theme in the present analysis.

The onset of programming for people very closely resembled that of another hobby studied in the survey: digital gaming. Among respondents, initiation into gaming spiked in the 1980s when many were very young. Figure 3 shows that only few respondents were first introduced to programming in the 1960s; starting to program was most common in the early 1980s. Both gaming and programming are, therefore, seemingly linked with the domestication of the computer: When computers increasingly arrived in homes, workplaces, and schools, they apparently were used for programming early on (Saarikoski, 2011). This is not altogether surprising considering what these early computers were like. Many initial home computers' operating systems were the programming language (e.g., BASIC in the Commodore 64), and computer magazines published long code listings that made programming doubtlessly seem like a normal practice. After the significant decline from the late 1980s to early 1990s, a slower increase in the number of experimenters among respondents is visible after the mid-1990s, especially among the women at the turn of the century. This may be linked to the increased teaching of information technology in Finland (Saarikoski, 2011), the diffusion of the Internet, and a growing demand for job applicants with programming skills. All in all, however, the growth in the 1990s did not

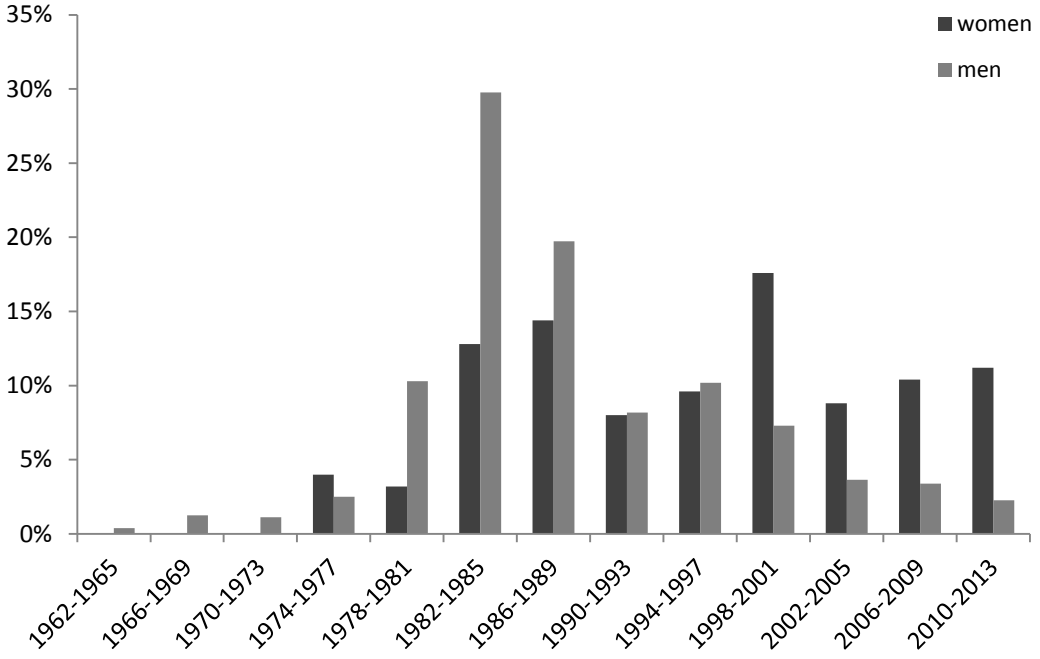

Figure 3. Percent distribution, by gender, of the year of that respondents first encountered programming (women *n* = 135; men *n* = 854).

exceed the 1980s levels. The gender difference in Figure 3 indicates that women started programming later than men. This finding concurs with other research showing that, the practice of programming was male-dominated already in the 1980s.

The respondents' ages when they first tried programming is shown in Figure 4. A few had begun before they were 7 years of age, but the number grows most clearly immediately thereafter. For men, the peak age was reached between 7 and 10, whereas it was 19–22 years for women. Compared to the digital gaming survey figures, people started programming later and at a wider range of ages: In gaming, the clear peak starting age for both men and women was 5–8 years, and some people had played games even earlier, as young as 1 year of age. The variance between the starting ages of gaming and programming may be linked to comprehensive education. More so than for gaming, programming often requires the ability to read and write, typically in English; children in Finland start elementary school at age 7, and most start learning English only a few years later. In the 1980s and 1990s, computer classes did not commence until junior high school, when students were 13 to 15 years old. The apparent trend that women began programming after age 19 probably is linked to university and university of applied sciences education. Nevertheless, those who started programming when they were 7 to 10 years old in the 1980s did not receive direct computer education in Finland. It is more likely that the schools had computers that interested students could use. New friends at school also may have influenced them by offering support, guidance, and cooperation; they may even have lent computers or software. This support of people nearby is similar to how people of older

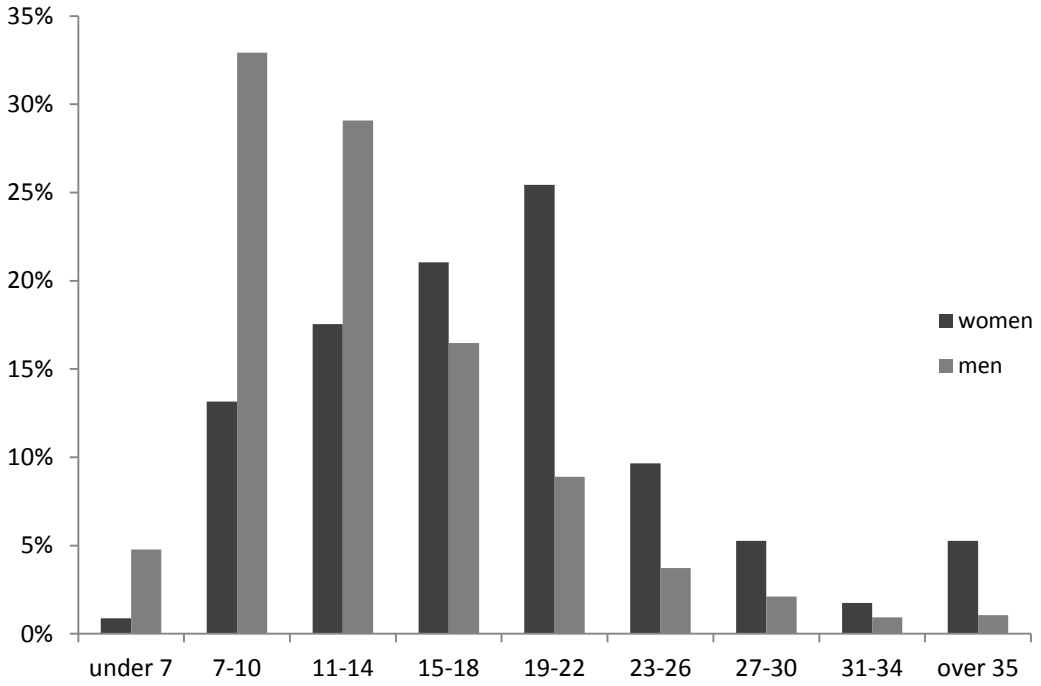

Figure 4. Percent distribution of the age of respondents at first exposure to programming (women n = 133; men n = 826).

Note. Two women and 28 men disclosed only the year and not their age of first exposure to programming.

age adopt computing (Bowen, 2011). Finally, 7-year-old children may have been considered old enough to attend a computer camp, a community that commonly spread the programming hobby in Finland, according to cultural historian Petri Saarikoski (2005).

Growing Interests, Assessing Needs

In the qualitative questions of the survey, the respondents were asked in more depth why they became interested in programming and about their programming skills and projects. When asked why they had wanted to try programming and why they pursued it further, three overlapping responses were the most common: an interest in mastering information technology (28% of the respondents to this question), the creative outlet that programming offered (24%), and a course at school or elsewhere (22%). Of these three explanations, the mastering and creativity of programming was most prevalent among younger male respondents, while school was a more typical response among females (though less common among all respondents born in the 1970s). The gender and age dynamics of pursuing an interest in programming are discussed more thoroughly in the next sections. The first two answers were demonstrated in the following responses, which highlighted, on the one hand, the charm of controlling the computer and, on the other hand, romanticized the creation of something new from scratch by means of programming.

> *It was interesting to get something accomplished, and programming involved a lot of problem solving.* (Man, born 1975)
>
> *I was attracted by the ability to control the machine and get it to do what I wanted.* (Woman, born 1967)
>
> *Unlimited possibilities. An empty canvas. Creativity. I can, by myself, invent what something does and how it works—the environment is not an obstacle.* (Man, born 1981)

These notions of controlling the machine evoke images of power and accomplishment, common in discourses about computer hacking in general (Levy, 1994; Wajcman, 1991). The style of the responses also may have been influenced by popular 1980s hacking and computing movies, such as *Tron* and *WarGames*, which were shown in Finnish cinemas. In addition, a link to psychologist Sherry Turkle's (2005, p. 101) notions of soft versus hard mastery in programming is seen in the quotes presented above: The second informant, a woman, displayed an "imposition of will over the machine," to use Turkle's terms, and the third, a man, was more like a creative artist in his expressions. However, even though Turkle (2005, p. 105) stated that boys are intrinsically hard masters and girls soft masters, the gender roles are reversed here.

In this context, creativity was not an explicit concept in the survey questions, but was used frequently by respondents. In their own discussions, programming hobbyists sometimes have juxtaposed the notion of creative and active programming with the perception of digital gaming as more passive (Reunanen & Silvast, 2009). One common expression of this in Finland originated in the 1980s, when the Commodore 64 computer was dismissed as merely a gaming machine (Saarikoski & Reunanen, 2014). The data here too have some of this confrontation: For example, according to one respondent, "*programming certainly interested me because of the possibility of creating new things; whereas, with games, the possibilities quite strictly were prearranged*" (Man, born 1977). Based on the results, however, gaming and programming are more mutually dependent than their separation among some hobbyists suggests. To a large number of respondents, programming, even becoming attached to it, was inspired first by

games, either because people wanted to make their own games or to understand how games work or because computer games were not available for all home computers at the time.

> *My brother and I both were interested in fiction and wanted to bring this up in some way. We both also loved computer games. I began to learn programming to produce my own computer game with my brother, but I quickly noticed that I loved programming in the first place.* (Man, born 1981)

> *When games had first made the machine familiar, it was fascinating to get a "look under the hood" and make your own game. It helped me to understand how it all works.* (Man, born 1976)

> *The computers of the time had no mass memory and all the programs, for example, games, had to be written yourself before you could use them. And enthusiasm grew along with programming skills.* (Man, born 1968)

All in all, nearly 200 survey respondents (19%) mentioned games in explaining why they became interested in programming. Purchasing games often was not even possible, as in the third quotation above: The respondent's computer was considered of little use without programming it (see also Swalwell, 2012). Others also stated that "*programming really couldn't have been avoided"* (Man, born 1979) and that "*the Commodore VIC-20 didn't do anything if you didn't program it to do something*" (Woman, born 1974).

The link between digital gaming and programming did not go unacknowledged in the media of the time. Computer magazines frequently included long program listings of games or detailed instruction on how to write code for games, as did the Finnish computer programming guidebook Amigan *pelintekijän opas* published in 1992. It promised the reader to "open up the fascinating world of game programming by revealing all the secrets of Amiga hardware programming" and purported, rather than to "teach machine language," to show how the Commodore Amiga's notorious "special chips" could be "mastered" by the use of machine language, especially by game coders (Keskikiikonen & Kiuttu, 1992, p. 7). A clear suggestion of creative, clever, and fun coding emerged here, rather than the use of programming merely for rational ends, such as learning how a programming language works for its own sake.

Institutional Channels and Adopting or Resisting Programming

Though guide books were important, they were not the sole channel for learning about programming. The respondents were attracted to programming via courses at school and elsewhere (22% of respondents to the question) and, albeit to a much lesser extent, computer clubs, employment, and anticipated job opportunities (each 1–2%). School, as already mentioned, was especially important for women, being raised by 62 female respondents (nearly half of all women who had tried programming) versus 141 males (one fifth). It was also the prevalent reason for pursuing programming of respondents born in the 1950s and 1960s, losing prominence for those born in the 1970s, and then becoming more important again for the those born after the mid-1980s. Nevertheless, access to computers and technology teachers in schools were particularly relevant initiators of programming for many.

> *I already knew about the Finnish computer magazine Prosessori, and I had been interested in computers in the past. The final spark came from a high school computer course.* (Man, born 1966)
>
> *In computer courses in junior high school, we already trained in BASIC programming. In high school, we continued with Turbo Pascal. I am the only one in my circle of friends who had such good computer teachers!* (Man, born 1981)
>
> *The school had Apple II machines and, as they were state-of-the-art technology, of course one had to try to make something with them.* (Man, born 1968)

Other important attracting factors included friends and relatives (13% of respondents), typically male, such as fathers, uncles, and brothers, but in some cases also mothers. This observation of the expertise of relatives and friends being drawn upon is typical in studies about domesticating new products (Bowen, 2011; Lehtonen, 2003). The results show that there are gender and age variances: 21 women (16% of those who pursued programming) as opposed to 106 men (14%) stressed other persons and the younger the respondent, the more prevalent this reason (suggesting older programmers discovered programming by themselves as default). In addition, magazines (i.e., program listings) and the aforementioned books (i.e., computer literature) were attributed as a starting point by 10% of respondents (slightly more often among men and among respondents born in the 1970s). Moreover, the hobby of creating demos—real-time programmed audio-visual presentations—is often raised by scholars as a crucial predecessor to programming, game coding in particular (Reunanen, 2010). However, it should be noted that demos, the demo community (known as the demo scene), and demo events, such as demo parties, were mentioned in just 14 responses (1%) as reasons to pursue programming. It seems that a demo hobby starts a little later than programming, as one gains inspiration from other supports for coding, such as school, friends and relatives, digital gaming, and the users' more general need to create something new. This finding is also in line with Turkle's (2005, pp. 23–24) taxonomy of generations among young programmers. Accordingly, younger coders are first interested in understanding how computers work and in mastering the machine, and only later (in the teenage years) do questions about one's subjective identity and the wider meaning of the hobby (e.g., Wasiak, 2014), such as belonging to a particular scene, become more central.

The respondents, thus, were asked what made them become interested in programming. Notably, a small minority of subjects (13 men and 5 women, mainly born in the 1970s) emphasized that they actually have never been enthusiastic about programming. They just had to do it for one reason or another, and under duress.

> *In computer classes at school, programming was mandatory. However, programming has never been a part of my voluntary activity, but rather has been linked with my studies.* (Woman, born 1977)
>
> *Nothing, and it is still not of interest. A necessary evil.* (Man, born 1975)
>
> *I was not interested. I was forced to do some things on a machine that I cannot do without it.* (Man, born 1988)

In other words, programming was seen by some only as a mandatory part of studies or work. If the excerpts provided earlier in this paper suggested that computer programming was a self-motivating activity, it is important to note that when computer literacy becomes a widely

shared aspiration and permeates education, not everyone takes computers and programming languages into use voluntarily. This finding is once again not new to domestication studies: People can choose to become nonusers of technology even if it is available, when "they are trying to balance a range of considerations in their lives at any moment and 'on balance' a new technology does not fit in, or is a low priority, or is not worth the effort or cost" (Haddon, 2011, p. 320). In the above cases, the respondents deemed it not worth engaging with programming more thoroughly than for completing a task in a course or at work.

Gender and Age Dynamics among Programmers

The gender and age dynamics of programming have been highlighted above along the results, but this section brings together these findings as they are the primary focus of the paper. The first thing worth stressing is that the experience of being motivated by the mere necessity of programming was shared across gender, despite the evidence that girls and women historically have been underrepresented in university computer science courses and more reluctant than men to become programmers (Jones & Burnett, 2008, p. 50; Nordli, 2001; Wajcman, 1991). Programming under duress was mentioned by 13 men (2% of the men who explained their interest in programming), but only 5 women (4%). Similarly, little or no gender difference was found in the proportion of respondents who specified that they started programming at work (the results are provisional as the number of people was small). However, technical interest and the creativity of programming, as well as the limitations of old computers and willingness to experiment, was attributed by men more than women as reasons to start programming. Female respondents, in turn, highlighted courses, training, education, and, to a slightly greater extent, another person as reasons. These differences suggest a similar result that was discovered in the digital gaming portion of the same survey (Naskali & Silvast, 2014, pp. 47–53). Apparently, men are presupposed to experiment with and become skilled in using computers, whereas women's interest in using computers is posited to be mediated by a factor, such as a school class or another person. At least this is how people recollect it in the 1980s, though clearly the gendering of programming skills and expertise has changed over time (Abbate, 2012; Wajcman, 1991).

The person's age also was a factor in their initial reasons for programming, as was mentioned earlier. The desire for experimentation, as a reason for pursuing programming, varies slightly with a person's age. Seemingly, the programming of earlier devices was experimental, as a rule (otherwise people would not have used them). In addition, starting programming at work clearly was more common the older the person was. Other reasons followed a slightly different trend: Magazines, guides, machine limits, and technical interest first arose as reasons to try programming for those born before 1980s, but started to decline for the younger respondents. Courses and studies showed a contrary trend: being less important among respondents born before 1980, after which courses were more commonly the motivator. The picture that emerged for the majority of the respondents born in the 1970s and 1980s is as follows: Spontaneous programming of limited machines arose from technical interest and/or with the help of computer literature and magazines more often than through school classes although programming in school, for the majority of respondents, was clearly also important. The youngest respondents that had tried programming (those born after the mid-1980s) showed once again another feature. Being introduced to programming by another person and

the creativity of programming were higher as motivations the younger the respondent was. The first explanation could be trivial—younger persons of course often use computers first with their teachers and parents—but the second suggests children's and young people's disposition to programming as a very concrete activity. Some respondents recalled, for example, *"I wanted to make a logic port simulator. I started designing logic circuits on paper at age nine"* (Man, born 1996), *"I found the GameMaker program from pelikulma.net"* (Man, born 1994), and *"we had started a band and I wanted to create a Web site, like all bands have"* (Man, born 1994). Many young respondents also laconically mentioned just "game making" as a reason to start experimenting with programming.

Computer Language Skills

Those who had engaged in programming reported a wide range of expertise in programming languages (Figure 5). A brief review of what these languages are about is necessary before moving on to the results. Professor Ray Toal (2015) of the Department of Electrical Engineering and Computer Science at Loyola Marymount University (Los Angeles, USA) has made an eight-scale classification of programming languages. For the purpose of the present paper, I will use the six of his types that overlap with the most typical programming languages mentioned by the respondents. The first is machine code, which comprises binary numbers that computer hardware can interpret; includes a few low-level instructions such as addition, subtraction, division, and square roots; and is tied directly to a particular computer processor such as Intel or Motorola. The second, assembly language, wraps over machine language by providing some simplistic encoding to make it slightly more readable to programmers. Unlike the first two types,

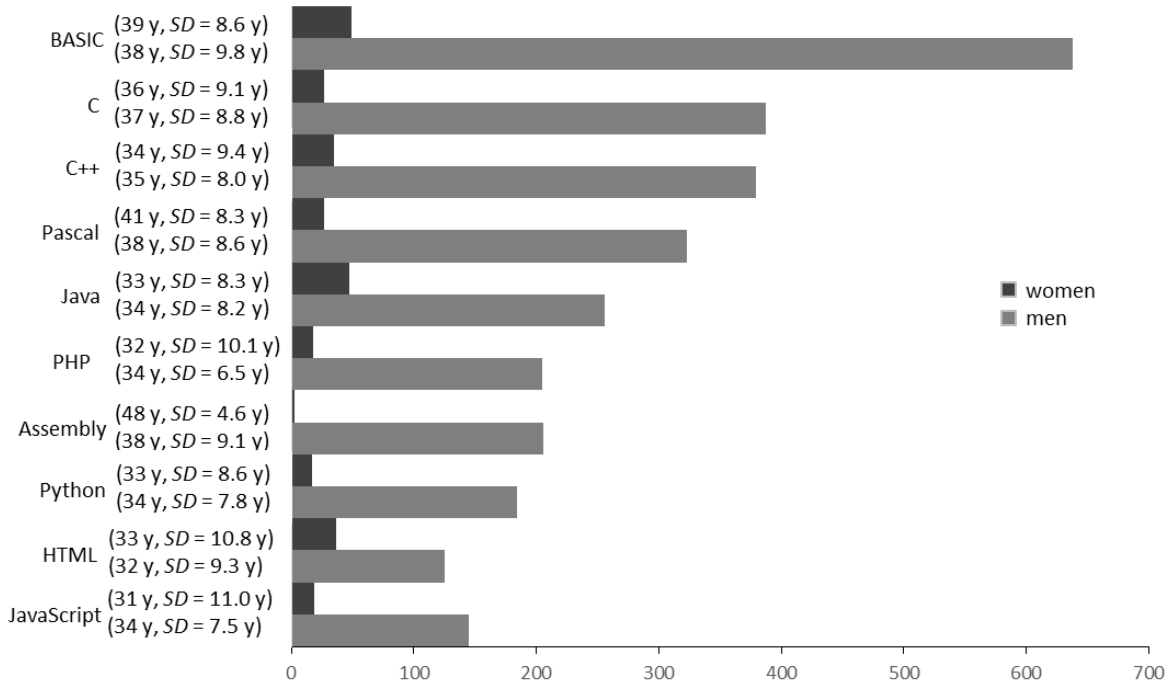

Figure 5. The 10 most common programming languages that respondents could use, by gender (with the median and standard deviation of the age of respondents; women n = 123; men n = 846).

the third type, high-level languages, is not tied to a particular processor and includes richer structures, from variables to complex expressions, control structures, and beyond. The fourth type, scripting languages, connects different systems together with a very little code, often for the specific purpose of increasing functionality. The fifth is system languages, which process low-level tasks such as memory and process management. The sixth, domain-specific languages, are used in highly special-purpose areas. In what follows, I classify BASIC, C, C++, Pascal, and Java as high-level languages; Assembly obviously as an assembly language; PHP, Python, and JavaScript as scripting languages; and HTML as a domain-specific language (although some respondents contest its very status as a programming language below). In practice, the categories overlap, with C being both a high-level and a system language and Perl being a scripting and a high-level language, for example.

The most commonly known programming language among the respondents—with 638 men (75% of those that knew a programming language) and 49 women (40%) and covering all age groups—was BASIC, an educational high-level language, followed by other even higher-level languages, such as C, C++, Pascal, and Java, which have more system functionalities. The distribution of competence in BASIC can be explained, at least partly, by its status as a teaching language and its installation on many older computers, functioning even as their user interface (Swalwell, 2012; see also Schofield, 2014). This is followed by mainly scripting languages, including PHP, Python, and JavaScript. Additionally, HTML, used for Web site markup, was perceived as a programming language by more than 150 respondents, although many also contemplated whether it really is considered a programming language. This suggests that designating a programming language was not altogether a simple category to the respondents and that such a language ideally should keep some link to the computer hardware and the operating systems, even the high-level languages. These few respondents demonstrated this confusion:

> *Does HTML count?* (Man, born 1987)
>
> *HTML hardly counts as programming and neither does the historical BASIC nor making DOS BATs.* (Man, born 1965)
>
> *Maybe HTML does not count?* (Woman, born 1980)
>
> *HTML, JavaScript, and the like are not counted as programming, but of course in work I have to deal with them constantly, like SQL.* (Woman, born 1974)

In contrast to the ambiguous reaction to HTML, low-level machine coding clearly was appreciated by the respondents. In use since the 1950s (Levy, 1994, p. 32) and being succeeded by more user-friendly, higher-level languages soon after inception (Abbate, 2012, p. 78), machine language contains "only the most basic hardware operations," hence, "any task beyond … basic arithmetic had to be built up from a lengthy sequence of these simple machine operations, making the resulting programs difficult to understand or debug" (Abbate, 2012, p. 76). In spite of these characteristics, coding in machine language, especially its symbolic variant Assembly, has been credited frequently by hobbyists and game coders for its bare-metal approach, granting almost direct access to the processor and the hardware and direct control over the operating system and system libraries (Maher, 2012, p. 185). In the survey data, Assembly appeared as the seventh most common language, above even HTML and Python, which indicates how vested the respondents were in the lowest level programming. In view of reminiscing and forgetting, it is interesting to add that only 21% of respondents reporting to know Assembly labelled the language correctly, more consistently among younger respondents.

A further 18% called it Machine Language—nearly the same thing but without symbols and highly difficult to write as noted earlier—and the majority, 61%, called it Assembler, which strictly speaking is not a programming language but rather a program that compiles Assembly code into Machine Language. Perhaps these words simply are used synonymously in everyday language. However, another possible explanation is that people wanted to say that they know Assembly—because of its inherent value—even if they had not used it so recently as to remember what exactly the language was called. An alternative explanation is offered by learning: If the respondents initially heard about the language in one guise, whether correct or incorrect, for example in the name of a course, then those recollections might have persisted to this day.

This observation is not to belittle the apparent programming competences of the respondents based on their own accounts. Overall, the respondents knew an average of five programming languages, but women on average knew only one and respondents under 30 one to two languages. Of all the respondents, 100 knew only 1 programming language, almost 500 could code in 2 to 5, and about 350 could use more than 5 languages, and those who knew more than 10 programming languages were about 100, as many as those who knew a single language. In other words, those who know how to program usually know multiple programming languages, based on this survey's results.

The respondents' programming skills also related to age and gender differences, similar to the other results. The average age of programmers was either 34 or 35 years for almost all of the ten most popular languages in our survey in Figure 5, with the following exceptions: Experts in Assembly, Pascal, BASIC, and C (but not the object-oriented C++) were on average a few years older, whereas HTML programmers were a few years younger. The men who knew JavaScript were slightly older than the women who knew this language, whereas the women who knew Assembly and Pascal were older than men (though this likely is a result of the number women in these groups). The differences in average age are small, and it may not be justified to refer to them as different programming generations. Tentatively, however, one could interpret that BASIC, C, Pascal, and Assembly are the most classic languages; C++, Java, PHP, Python, and JavaScript the more general modern languages; and HTML an emerging language known by the youth and used for Web site markup. More variance exists in the ages of BASIC and Assembly programmers than the other languages, showing how these two languages attracted users of all ages. An even clearer finding is that the competence in programming languages is gendered. Only three women (2% of women programmers) reported that they knew Assembly language but, more typically, women comprised just under 10% of those knowing how to program in most other languages. This differed for JavaScript, Java, and HTML, where women comprised 12%, 15%, and 23% respectively.

Compared to the charts of currently used programming languages, collected from the Web site Programming Language Popularity (2015), the most notable difference is in the absence of several languages from use by this study's respondents. Microsoft's high-level programming language C#, Apple's similarly high-level object-oriented Objective C, the object-oriented scripting language Ruby, and programming scripts in the context of Unix Shells, although among the top 10 languages in current use, were not in the top 10 of this survey although they were mentioned by some respondents. This suggests that the programmers in the survey used slightly more marginal languages than the general public and that their recollections were shaped by nostalgia to the languages of the past.

Projects and the Changes in Programming Disposition

The respondents' most common programming projects (Figure 6) were tools and utilities, whether created at work or on a more voluntary basis at home or at school. Thereafter, the most common projects were again self-programmed games. Games were followed by Internet programming and, to a lesser extent, multimedia (e.g., demos and music), coursework (e.g., coding assignments), embedded systems (e.g., home automation), and scientific programming (e.g., simulation modeling). Women were represented more in the domains of coursework, the Internet, and tools and utilities, whereas some hobbies comprised almost exclusively males, in particular multimedia, embedded systems, and science programming. Furthermore, the men who engaged in programming utilities, science programming, multimedia, and embedded systems were relatively older, whereas women who had done Internet programming, multimedia, and science programming (though there were only four of the latter) were younger.

The respondents were able to list more than one project, and the distribution of these fell into two broad categories. Slightly more than half had done only one type of programming, typically in the context of their studies. In the second category, the respondents had engaged in two or more, at times quite different, programming activities. The clear majority of respondents, those that undertook more than two programming projects, were men. For example, the following persons appeared to have moved—or perhaps even drifted among—a number of programming projects during their hobby. The middle quote below, in particular, also indicates how programming projects and language competence meet: Specific languages are for certain types of projects, which may explain the respondents' wide knowledge of various programming languages.

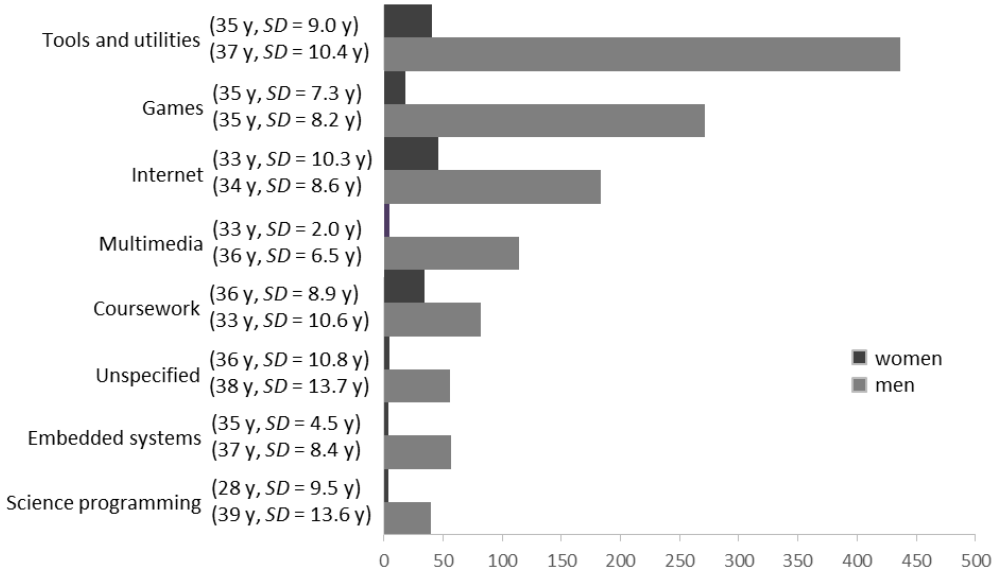

Figure 6. The respondents' most common types of programming projects by gender (with the median and standard deviation of the age of respondents; women, n = 110; men, n = 764).

> *Web sites, information systems for workplaces, many tools for research use—for example, compilers and simulators. As a hobby, small tools and a few bigger projects and, among others, a couple of small-scale games.* (Man, born 1984)

> *As a child I did all kinds of small things with BASIC, with no particular aim. At work as an IT professional, I made shell scripts for environment building, small applications in Perl, and so forth. While studying, I completed schoolwork with Pascal and Java. In later studies, I tinkered with Java to build Web graphics. As a hobby, I have created overly intricate electronics with Arduino and programmed a few things for it with the Processing programming language.* (Woman, born 1977)

> *At first I programmed for my own pleasure, because there were not really any options. With the Commodore Amiga I started to code demos and related things, mostly music libraries. In this way, I even got a job where I have coded everything possible for different embedded systems, mobile devices, and servers. Diverging, I drifted, via programming drivers for a Graphics Processing Unit chip, into designing the circuits themselves.* (Man, born 1976)

Many respondents perceived that their programming hobby changed over time. In total, more than two thirds of the respondents (predominantly male, 36 years of age on average) felt that their interests had shifted. Studying and work, as well as aging, generally brought new interests into their lives. Such changes often resulted in the programming hobby coming to an end. Nearly 200 respondents (17%; 17% of men, 15% of women, 36 years of age on average again) said they had stopped programming completely or almost completely. The typical reasons given were declining interest, programming having been left behind in studies and a perceived lack of programming abilities, in addition to the proliferation of packaged software that reduces the importance of or necessity for programming. The following quotes illustrate how work had destroyed the person's programming hobby (of all the respondents, 70% of men, or some 800, and 40% of women, about 130, said they worked in a computing-related job). They also show how advances in information technology discouraged programming, and how a person's skills were not perceived as adequate enough to be a programmer.

> *Today, programming is my job, and the job has killed the hobby to be pretty much nonexistent, unfortunately.* (Man, born 1977)

> *In my days, I wrote membership registration (applications) and all kinds of mathematics things. But this programming side has been left behind in recent years as you can achieve a lot of things with macros of software tools.* (Man, born 1952)

> *School assignment for the most part. After school, programming pretty much has ended because I really do not know how to.* (Man, born 1983)

For the cessation, less common causes were the lack of time and aging—one respondent reported having programmed only as a child. All in all, the respondents did not attribute their cessation to having too many other things to do or as a result of reprioritization in adult life. The primary reason for stopping was not a specific waning in either social or rationalized reasons, such as no longer having friends to code with or perceiving that programming lacks rational relevance (cf. Olphert & Damodaran, 2013, p. 567). What is more relevant is that the respondents had significant reasons for programming, involving, for instance, their personal interests, external motivations, notions of creativity or intrigue, fun and leisure, and their own fondness of programming. When this motivation is taken away, it may be difficult to continue the hobby regardless of whether the reason may be time constraints due to work or the availability of

commercial software. Finally, though, almost one third of those who had stopped (44 men and 4 women, 33 years old on average) did not offer a reason for the end of their programming hobby—they simply reported stopping over the passage of time.

DISCUSSION

The aim of this paper was to answer two research questions: What leads to people's pursuit of or interest in programming, and how does the practice of programming evolve over time? To answer the questions, responses to quantitative and open-ended questions were drawn from a massive survey of Finnish computer hobbyists, a portion of which included their recollections about programming. Among the key findings is the clear agency that the survey respondents had exercised over programming, programming languages, and practices of use. In the answers, the governmental aims of computer literacy discourse had only a minor impact on increasing public knowledge of computers and the essential capacities for the future information society, especially programming skills (Saarikoski, 2011). Rather than following needs projected from above, the rationales for appropriating programming seemed highly personal, including the personal aspirations of the respondents and their skills, albeit often appropriated together with relatives and friends. These factors also suggest why programming hobbies may end: When no longer part of one's own needs or routines, the hobby becomes difficult to maintain and is stopped. All in all, it seems that learning to program, like learning to use any other technology in ordinary life, takes time and is a precarious achievement. The experiences documented here suggest that having opportunities does not help everyone become a programmer; some may be reluctant to appropriate it and may even become opposed to the practice, which could happen either at an initial introduction or much later, after skills already have been acquired. At the same time, many more respondents started programming as a hobby for reasons that seem closely aligned with their own notion of creativity, leisure, and experimenting. Also important for those significantly invested in the hobby was their pride in mastering computers through tinkering with difficult programming problems and with challenging languages, such as Assembly, the programming language just above the lowest level language.

These motivations sound precisely like posited male ideals for using computers (e.g., Aune, 1996; Nordli, 2001; Wajcman, 1991)—mastery coupled with activity, experimentation and doing, rather than planning carefully ahead regarding how technology should be used. Indeed, the results of this study present gender differences that suggest such divisions. Whereas issues such as future employment and willingness to try programming (the themes of computer literacy discourse) were similar across genders, men more than women emphasized technical interests, the creativity of programming, experimentation, and the challenging limitations of old computers as a reason to code. Women, on the other hand, attributed their initiation to programming to training, education, and other persons who had introduced them to programming. A more rational reason was that they had the opportunity to try programming in the first place. The data also almost exclusively lacked female programmers who were, at the time of the survey, younger than 20 or older than 50. Image reasons associated with programming and programmers may explain some of the absence, at least with the younger respondents (Nordli, 2001). With older programmers of both genders, it seems that their culture of computing had been rather different from the younger respondents. For the older generation, programming

mainly happened on the job and in university courses, whereas the subculture of coding (e.g., magazines, guides, and other means of intervening) was less important than for the respondents born in the 1970s and the 1980s. Maybe this culture of computing gave limited visibility for women hobbyists even if they may have experienced programming just as men did. Nonetheless, in line with domestication studies, the results suggest that younger men view the computer playfully and women more rationally (Buse, 2009). At least in this group of respondents—highly educated young adults—it was no longer the case that men approach programming as solutions to be preconfigured before actual coding takes place (cf. Abbate, 2012).

Lastly, the time period of this study lends the results a historical specificity. I concentrated on a period when programming was no longer merely a concern of highly trained professionals in science, technology, government, and business institutions, but rather became an inherent part of experiencing computers in ordinary life. To psychologist Sherry Turkle (2005, p. 7), 1980s computer programming, with its hands-on approach, kept the user in "contact with the 'bare machine'" as it no longer does. Her study concentrated on educational, high-level programming languages that, perhaps to many engaged programmers, seem removed from, rather than in contact with, the bare machine (Abbate, 2012, p. 76; Maher, 2012, p. 185). Yet, her intuition on how machines and programming interrelated seems to be confirmed by these results. For example, at the time, practices such as playing digital games often required running them via a programming language or, in many cases, typing in long program listings from computer magazines (Swalwell, 2012). Tinkering then was inherent even for leisurely computer users (see Naskali & Silvast, 2014; Saarikoski & Suominen, 2009). For the present analysis, this offers a good vantage into a time when programming was likely a more normal and regular practice than it is today, when the needs of its incorporation into daily life were more or less apparent whether people embraced it or not. Because programming became less popular among younger people as the results suggest, it is appropriate to gather oral historical materials about programming now. At some point, programming practices may fade from people's memories, as computers become more integrated into, and thus less and less apparent in, daily life.

CONCLUSION

In this paper, I presented a study of the appropriation of computer programming by Finns, mainly in the 1980s, by drawing on two analytic lines, now commonplace in historical research and science and technology studies—notions about domesticating technology and about oral history. Their applications demonstrated several specific issues regarding the problems identified in the paper. First, learning to program and adapting its practices is a slow, step-by-step process that can fail before people have become programmers or even after they have developed some level of competence in programming. Second, the ways that people both remember and forget experiences in the past offer interesting interpretations from a period when programming was still a more popular practice and was more or less required by the technologies of the time period. The group of respondents was not random, and the findings are, hence, not representative or generalizable. Nevertheless, the survey data offer an important view on the complexity of programming practices and their common characteristics in a large group of people. The variety of things that people do with programming, their own diverse reasons for doing so, and the link of these issues to gender and age were the main outcome of

the inquiry here. Future studies may use this information to make more focused in-depth case studies in households, schools, or other situations where specific groups of people, perhaps selected to represent a particular gender or age group, learn to code.

ENDNOTES

1. See https://www.kickstarter.com/projects/lindaliukas/hello-ruby
2. Koodi2016 [Code2016]. See http://koodi2016.fi/
3. Lasten koodikoulu [Code School for Kids]. See http://koodikoulu.fi/
4. See http://railsgirls.com
5. The survey was coordinated by and data are in the possession of the University of Turku, Faculty of Humanities, Digital Culture. The Kone Foundation in Finland funded the research. The data comprise responses from 1,453 Finns (in Finnish). Results from the survey have been summarized and reported in Tiia Naskali and Antti Silvast (2014). Naskali and I split the data analysis and report writing duties by subject area; I analyzed all the results presented in this paper. Naskali and the other research project members, Petri Saarikoski, Jaakko Suominen, and Markku Reunanen, also commented the results.
6. The survey was in Finnish, and the respondents needed to be proficient in this language to respond. Excerpts were translated by the author into English for use in this paper.
7. Specializing in everyday life materials from Great Britain, The Mass Observation Archive (see http://www.massobs.org.uk/index.htm) is maintained at the University of Sussex (parts of the archive are also accessible at http://www.massobservation.amdigital.co.uk/). The materials include people's diaries from 1939 to 1967, day surveys (one-day diaries) 1937–1938, people's responses to directives (open-ended questionnaires) 1939–1955, interviews, correspondence, and a host of publications prepared from these data.

REFERENCES

Abbate, J. (2012). *Recoding gender. Women's changing participation in computing*. Cambridge, MA, USA: MIT Press.

Aune, M. (1996). The computer in everyday life. Patterns of domestication of a new technology. In M. Lie & K. Sørensen (Eds.), *Making technology our own. Domesticating technology into everyday life* (pp. 91–120). Oslo, Norway: Scandinavian University Press.

Benediktsson, O. (2009). FORTRAN II—The first computer language used at the University of Iceland. In J. Impagliazzo, T. Järvi, & P. Paju (Eds.), *History of Nordic Computing 2* (pp. 149–155). Berlin, Germany: Springer.

Bowen, L. M. (2011). Resisting age bias in digital literacy research. *College Composition and Communication*, *4*, 586–607.

Buse, C. E. (2009). When you retire, does everything become leisure? Information and communication technology use and the work/leisure boundary in retirement. *New Media and Society*, *11*, 1143–1161.

Fingerroos, O. (2004). Haudatut muistot—Rituaalisen kuoleman merkitykset Kannaksen muistitiedossa [Buried memories—The meanings of ritual death in the memory knowledge of Kannas]. *Elore*, *11*. Retrieved January 26, 2015, from http://www.elore.fi/arkisto/2_04/fin204.html

Gobo, G. (2008). *Doing ethnography*. London, UK: Sage.

Gram, C., Laaksonen, T., Ohlin, T., Lawson, H., Skår, R., & Stangegaard, O. (2005). History of the Nordic computer industry. In J. Bubenko Jr., J. Impagliazzo, & A. Sølvberg (Eds.), *History of Nordic computing* (pp. 179–190). Berlin, Germany: Springer.

Haddon, L. (2004). Empirical studies using the domestication framework. In T. Berker, M. Hartmann, P. Yves, & K. Ward (Eds.), *Domestication. The enactment of technology* (pp. 103–122). Maidenhead, UK: Open University Press.

Haddon, L. (2011). Domestication analysis, objects of study, and the centrality of technologies in everyday life. *Canadian Journal of Communication*, *36*, 311–324.

Jones, S., & Burnett, G. (2008). Spatial ability and learning to program. *Human Technology: An Interdisciplinary Journal on Humans in ICT Environments*, *4*(1), 47–61.

Kantor, J. (2014, December 23). A brand new world in which men ruled. *New York Times*. Retrieved January 26, 2015, from http://www.nytimes.com/interactive/2014/12/23/us/gender-gaps-stanford-94.html?_r=0

Keskikiikonen, M., & Kiuttu, P. (1992). *Amigan pelintekijän opas* [Amiga's Game Maker's Guide]. Keuruu, Finland: Sanomaprint Erikoislehdet.

Lehtonen, T.-K. (2003). The domestication of new technologies as a set of trials. *Journal of Consumer Culture*, *3*, 363–385.

Levy, S. (1994). *Hackers: Heroes of the computer revolution*. New York, NY, USA: Dell Publishing. (Original work published 1984)

Maher, J. (2012). *The future was here. The Commodore Amiga*. Cambridge, MA, USA: MIT Press.

Mahoney, M. (2008). What makes the history of software hard. *IEEE Annals of the History of Computing*, *30*(3), 8–18.

Misa, T. (2009). Organizing the history of computing: 'Lessons learned' at the Charles Babbage Institute. In J. Impagliazzo, T. Järvi, & P. Paju (Eds.), *History of Nordic computing 2* (pp. 1–12). Berlin, Germany: Springer.

Naskali, T., & Silvast, A. (2014). *Tietokonekerhoista blogosfääriin, pöytäkoneista älypuhelimiin. Kokemuksia tietokoneharrastamisen arkipäiväistymisestä* [From computer clubs to the blogosphere, from desktop computers to smart phones. Experiences about domesticating computer hobbies]. Turun yliopisto (University of Turku), Kulttuurituotannon ja maisemantutkimuksen koulutusohjelman julkaisuja 44.

Nordal, O. (2009). Tool or science? The history of computing at the Norwegian University of Science and Technology. In J. Impagliazzo, T. Järvi, & P. Paju (Eds.), *History of Nordic computing 2* (pp. 121–129). Berlin, Germany: Springer.

Nordli, H. (2001). From "spice girls" to cyber girls? The role of multimedia in the construction of young girls' fascination for and interest in computers. In M. van Lieshout, T. M. Egyedi, & W. Bijker (Eds.), *Social learning technologies: The introduction of multimedia education* (pp. 110–113). Farnham, UK: Ashgate Publishing.

Oinas-Kukkonen, H., Similä, J., Pulli, P., & Kerola, P. (2009). The impact of computer science on the development of Oulu ICT during 1985–1990. In J. Impagliazzo, T. Järvi, & P. Paju (Eds.), *History of Nordic computing 2* (pp. 195–208). Berlin, Germany: Springer.

Olphert, W., & Damodaran, L. (2013). Older people and digital disengagement: A fourth digital divide. *Gerontology*, *59*, 564–570.

Opetus- ja kulttuuriministeriö [Finnish Ministry of Education and Culture] (2014). *Opetusministeri Kiuru: Ohjelmointi peruskoulun opetussuunnitelman perusteisiin* [Education Minister Kiuru: Programming to be added to elementary school's curriculum foundations]. Press release 21 January 2014. Retrieved January 26, 2015, from http://www.minedu.fi/OPM/Tiedotteet/2014/01/Koodauskoulu.html?lang=fi

Pollen, A. (2013). Research methodology in mass observation past and present: 'Scientifically, about as valuable as a chimpanzee's tea party at the zoo?' *History Workshop Journal*, *75*, 213–235.

Programming Language Popularity (2015). Retrieved January 26, 2015, from http://langpop.com/

Reunanen, M. (2010). *Computer demos. What makes them tick?* (Licentiate thesis). Aalto University School of Science and Technology, Helsinki, Finland.

Reunanen, M., & Silvast, A. (2009). Demoscene platforms: A case study on the adoption of home computers. In J. Impagliazzo, T. Järvi, & P. Paju (Eds.), *History of Nordic Computing 2* (pp. 289–301). Berlin, Germany: Springer.

Saarikoski, P. (2004). *Koneen lumo: mikrotietokoneharrastus Suomessa 1970-luvulta 1990-luvun puoliväliin* [The lure of the machine: The personal computer interest in Finland from the 1970s to the mid-1990s] (Doctoral dissertation, University of Turku, Finland). Jyväskylä, Finland: Jyväskylän nykykulttuurin tutkimuskeskuksen julkaisuja 83). Jyväskylä, Finland: University of Jyväskylä.

Saarikoski, P. (2005). Club activity in the early phases of microcomputing in Finland. In J. Bubenko Jr., J. Impagliazzo, & A. Sølvberg (Eds.), *History of Nordic Computing* (pp. 277–287). Berlin, Germany: Springer.

Saarikoski, P. (2011). Computer courses in Finnish schools, 1980–1995. In J. Impagliazzo, P. Lundin, & B. Wangler (Eds.), *History of Nordic Computing 3* (pp. 150–158). Berlin, Germany: Springer.

Saarikoski, P., & Reunanen, M. (2014). Great Northern machine wars: Rivalry between user groups in Finland. *IEEE Annals of the History of Computing*, *36*(2), 16–26.

Saarikoski, P., & Suominen, J. (2009). Computer hobbyists and the gaming industry in Finland. *IEEE Annals of the History of Computing*, *31*(3), 20–33.

Sajaniemi, J., & Kuittinen, M. (2008). From procedures to objects: A research agenda for the psychology of object-oriented programming education. *Human Technology: An Interdisciplinary Journal on Humans in ICT Environments*, *4*(1), 75–91.

Schofield, J. (2014, April 30). Baby we were born to RUN: Celebrating 50 years of Basic. *The Guardian*. Retrieved January 26, 2015, from http://www.theguardian.com/education/2014/apr/30/celebrating-50-years-of-basic

Suominen, J. (2014). *Helposti ja halvalla? Nettikyselyt tutkimusaineiston kokoamisessa* [Easy and cheap? Internet surveys in gathering research data]. Manuscript submitted for publication.

Swalwell, M. (2012, June). The early micro user: Games writing, hardware hacking, and the will to mod. *Proceedings of Digital Games Research Association (DiGRA) Nordic 2012 Conference: Local and Global—Games in Culture and Society.* Retrieved January 26, 2015, from http://www.digra.org/wp-content/uploads/digital-library/12168.37411.pdf

Tilastokeskus [Statistics Finland] (2012). Tieteen ja teknologian henkilövoimavarat [The resources of science and technology]. Helsinki, Finland: Statistics Finland. Retrieved May 1, 2015, from http://www.tilastokeskus.fi/til/tthv/2012/tthv_2012_2014-03-20_tau_002_fi.html

Tilastokeskus [Statistics Finland] (2013). Väestön koulutusrakenne [The education structure of the population]. Helsinki, Finland: Statistics Finland. Retrieved May 1, 2015, from http://tilastokeskus.fi/til/vkour/2013/vkour_2013_2014-11-06_tau_001_fi.html

Toal, R. (2015). *Classifying programming languages.* Retrieved January 26, 2015, from http://cs.lmu.edu/~ray/notes/pltypes/

Turkle, S. (2005). *The second self: Computers and the human spirit* (20th anniversary edition). New York, NY, USA: Simon and Schuster. (Original work published 1984)

Ukkonen, T. (2000). Muistitieto tutkimuksen kohteena ja aineistona [Memory knowledge as a research topic and as data]. *Elore, 7.* Retrieved January 26, 2015, from http://www.elore.fi/arkisto/2_00/ukk200.html

Ullberg, S. (2013). *Watermarks. Urban flooding and memoryscape in Argentina* (Doctoral dissertation). Stockholm University, Stockholm, Sweden. Retrieved March 16, 2015, from http://su.diva-portal.org/smash/get/diva2:618130/FULLTEXT01.pdf

Wajcman, J. (1991). *Feminism confronts technology*. London, UK: Polity Press.

Wasiak, P. (2014). Amis and euros: Software import and contacts between European and American cracking scenes. *WiderScreen* 1–2/2014.

Author's Note

I acknowledge the helpful comments from Markku Reunanen, Patryk Wasiak, Petri Saarikoski, Jaakko Suominen, Tiia Naskali, Leda Kopach, and Anjuli Gunaratne and am especially grateful to Tiia Naskali for collecting the survey data that this paper elaborates further. The research presented in this paper was funded by the Kone Foundation and was part of the project *Kotitietokoneiden aika ja teknologisen harrastuskulttuurin perintö* [Home Computer Era and the Heritage of Technological Hobby Culture].

All correspondence should be address to
Antti Silvast
Research Fellow
Science, Technology and Innovation Studies
School of Social and Political Science
The University of Edinburgh
1.01A Old Surgeons' Hall
High School Yards
Edinburgh, EH1 1LZ, UK
antti.silvast@ed.ac.uk

Human Technology: An Interdisciplinary Journal on Humans in ICT Environments
ISSN 1795-6889
www.humantechnology.jyu.fi