Confessions of a nomad. A personal view on the history of computing

Liesbeth de Mol

Confessions of a nomad – a personal view on the history of computing[1]

Liesbeth De Mol
liesbeth.de-mol@univ-lille.fr
CNRS, UMR 8163 Savoirs, Textes, Langage
Université de Lille, France.[2]

When I was invited to write a chapter for this book, I initially had significant doubts. Of course, there was the usual problem of being already overworked but the real problem for me was the topic proposed to me: *Writing Computing History from a Philosophical Perspective*. The chapter that ultimately resulted is written as a personal and self-reflective essay. A disclaimer to the reader is thus appropriate: the present text is not a conventional academic text in which I make an argument to support a certain conclusion, but is more focused on the sharing of an experience of someone who has often felt as an outsider to the field of the history of computing, and what are and have been, in that experience, the pitfalls and opportunities encountered and observed. The result is a *personal* view on the history of computing, the potential of history to improve understandings of our current situation and aimed towards the future. Such perspective, I know and understand, is not unproblematic for a pure historian and requires, from that historian, to engage in a discussion on accepted methodological assumptions. These are recast in this chapter as obstacles which if identified can help to open up the field of the history of computing towards other domains.

1. Becoming an academic nomad

Let me now return to the original problem I had with accepting to write this chapter and which is anchored in my own personal history as a researcher. After a master's degree in art history[3] (1999) I believed I had found my one and true love, philosophy. I embarked on a second master in philosophy (2002), followed by a PhD in philosophy. But during my doctoral years, engaging with history of computability and computing became unavoidable and I steadily realized that historical research too was a love, only forgotten for some years.

At my philosophy department, the tendency was towards analytical philosophy. During my years as a philosophy student I was introduced quite heavily into logic and the idea that there is only one reliable method which is the scientific method. Underlying that, was a strong belief that truth and the good are achievable as long as we are rational and so the question of what rationality is, was and is of course a basic one. But there were also some other courses that introduced me to authors like Levinas, Foucault, Derrida, Heidegger, Jankelevich, Bloch, Arendt, Sartre, etc – philosophers which were identified as "continental", where that adjective was used by the analytical philosophers in a deprecatory manner. It was from some of those lectures that I would get to a basic source for my later research. This was the essay "The question concerning technology" by Martin Heidegger.[4] At my department, Heidegger was considered by most as one of the most obscure philosophers to be avoided. That view is historically anchored in how earlier analytical philosophers, most notably Rudolp Carnap, rejected Heidegger's work as an example of metaphysical pseudostatements.[5] Anecdotal but telling here is perhaps the way one professor introduced Heidegger: they opened *Sein und Zeit* at an arbitrary page, read an arbitrary passage, to use that as an argument to show that we could all be happy to ignore such nonsense. These kind of experiences as a philosophy student learned me at quite a young age that one should not take for granted what some authority is stating for a fact and dare to question it. It was probably there that I started to become an outsider to the field in which I was being trained: I refused to accept without questioning the main disciplinary

assumptions required to become an analytical philosopher. I refused in a sense what I was being taught.

At that time I had a rather difficult relation with computers and tried to avoid them as much as I could. In retrospect I see this as a case of alienation in the sense of the French philosopher of technology, Gilbert Simondon,[6] flirting with neoluddism. But my reading of the Heidegger essay changed my perspective fundamentally and brought me right on the path of computing and programming. Computational technologies can be read as the perfect embodiment of what Heidegger called “das Bestand”, the “standing reserve”. Roughly speaking, this refers to a particular mode of existence in which humans “order” the world as a “standing reserve”. This is the essence of modern technology, the particular manner in which we bring things into our world. While on the basis of that analysis one might easily become a technopessimist,[7] the message I got from that work was that there is a way of out of this: instead of avoiding such technologies out of fear of becoming part of das Bestand, what I should be doing is to try and *understand* those very same technologies, how they function, how they came to be, what is their foundation. If I could have an improved understanding of what happens “beyond” the interfaces I was using, alienation could become emancipatory, enabling me to see a way out of das Bestand rather than a way into it.[8] But what could that mean, to understand what happens “beyond” the interface. At the time it resulted in two related but different decisions.

First of all, I was going to teach myself how to program. That experience had quite a profound effect on me – my first program, the typical “Print ‘Hello world’”, was a revelation and later, I would use that to develop an experimental attitude towards computation and programming leading me to the idea of recasting human-computer interactions as conversations.[9]

Second, I decided to go back to some basic historical sources for modern computing. Given the opacity of computing technologies of that time – an opacity that has only increased since then – perhaps going back to the sources would give me some sense, some understanding of our current situation. Steadily, I developed a view of how historical research can help us to make our current situation comprehensable, to see the historical roots and traces of particular problems related to computational technologies and to try and see new opportunities to turn them around, at least, locally. To give a very simple but hypothetical example, we could ask ourselves how most of us became a particular type of user of such computational technologies, the so-called end-user, and what this entails. To answer that question what is needed is a longer, careful and diversified history of the conceptions of “user” to contextualize historically the most common view of the User as commercialized by Macintosh and, later, Microsoft and Google and anchored in visions of personal computing of the late 1970s and early 1980s.

After having graduated in philosophy, I decided to try and continue along the path of computing and programming, combining my renewed interest in historical thinking and my steadily developing experiences in programming. My PhD project was aimed at the theoretical formalism of logicians like Emil Post, Alonzo Church and Alan Turing. They developed these formal devices in the 1920s and 1930s with the aim of showing that there does not exist a uniform finite method or procedure to solve certain problems relevant to the foundations of mathematics and known as decision problems.[10] One speaks here of (recursive) unsolvability results. One of the more familiar formalisms to the reader is probably that of the Turing machine, a formalism aimed at formalizing the intuitive notion of computability. These formalisms were later recast as formal foundations for the modern computer and programming.[11] These impossibility results were, however, quite abstract and in the proofs one could not find any real indication of the conseuqences of these impossibility results on actual mathematical practices and, most important to me, concrete computational practices. The aim then of my PhD was to understand the connection between these abstract results and the concrete computations of real computing machines.[12] The latter resulted in two different research problems. The first one asked in how far the emergence of the first computers could be historically related to these so-called logical foundations. Second, in order to understand “unsolvability” as a phenomenon of concrete computations, I focused in part II of my PhD on a

particular formalism known as tag systems and developed by Emil Post.[13] I was able to prove a number of theoretical results related to solvability and unsolvability of tag systems and conducted a number of so-called computer experiments to explore a series of characteristics of tag systems which could be interpreted as phenomena of unsolvability. These experiments were much inspired by my reading of Stephen Wolfram's book *A new kind of science.*

Today I look back at that dissertation as a somewhat naive enterprise containing quite some mistakes and even conclusions I no longer support. Still, much of what I learned and experienced during my doctoral years has been quite basic to my later work. In fact, when I hesitantly went back to parts of this dissertation for the writing of this chapter, I was surprised to find there a preemption of my current stance with respect to disciplines and their boundaries and which I develop in this chapter in later sections (Sec. 6):

> As is clear from the outline of this dissertation I have not restricted myself to one specific research domain or methodology. […] It is a current trend both of the humanities as of the exact sciences that one needs to specialize into one sub-sub-...-sub domain to get anywhere. This is even becoming a harder reality for the younger researchers who seem to have no other choice but to specialize. This evolution I regret. Although I am the last to say that one should not know his or her subject well-enough, otherwise one can only make mistakes, I think being able to cross the borderlines of domains, trying to link them up, is at least as important as this specialization. In my personal opinion, one of the ways to make progress is to have the freedom to mix up domains.

My publications and talks during my postdoc years were varied but did not make a real effort to really open up towards one another the different disciplines in which I had been working. Still, it was unavoidable that my style was not really "pure". I steadily started to feel more like an outsider in whatever conference I attended. That subjective impression was sometimes confirmed by the public. I might meet historians and they would tell me that my work, was not *really* historical, driven as it was by concerns with current developments in computing or as being too technical; I might be attending a philosophy conference and people would perceive me as a historian and not as a *real* philosopher or blame me for not following the *proper* philosophical method taking distance from the practice itself.[14] I return to these points in more detail later. It is then perhaps no coincidence that I was hired as a permanent researcher in 2013 by one of the interdisciplinary committees of the *Centre National de Recherche Scientifique*.[15]

But while I surely felt and often still feel like an "outsider" sometimes, I steadily realized that this does not correspond to reality (anymore): if I really would have been an outsider, how could I present my results in these different communities, how could it be that these different communities accepted my submissions to their journals and invited me to their events? A more appropriate description of my own way of being an academic is that of an academic nomad:[16] someone who travels across different academic communities and is usually welcomed and accepted *as* a nomad.[17]

To conclude this section, I would like to return to the problem I had with accepting to write this chapter: I simply could not write a chapter on the history of computing from the perspective of philosophy given my steadily developed nomadic position with respect to both fields. Instead, I chose the more radical personal style and to write from the perspective of someone who does not subscribe any longer to one particular discipline. In the remainder of this chapter, I would like to develop how this nomadic way of being has steadily resulted in specific methodological stances and their implications for my *personal* view on the field history of computing.

In what follows, I consider first a methodological tool coming from epistemology to develop my personal view on the history of computing (sec. 2) and then provide a brief description of my initial struggles within philosophy given my topic of research. Based on an obstacle I inherited from my philosophy education, I provide a first opportunity of where I believe historians of computing have

something to contribute to philosophy and beyond (sec. 3). In Section 4, I describe my initial methodologies for doing historical research and the obstacles it created for myself. On that basis I argue for a more pluralist historiography of computing based on two concrete cases. Given my background in philosophy, it is probably not too surprising that I am rather presentist in my historical research. While acknowledging that this is not an obvious if not subversive position for the historian, I provide a number of arguments of why I believe we should dare and have a discussion on this also within history of computing (Sec. 5). In a next section, I give then a brief description of my own initiatives towards a more interdisicplinary approach in history and philosophy of computing (Sec. 6) to end with a short concluding section (Sec. 7).

2. Overcoming obstacles

As explained, as a nomad, I have travelled across a variety of different academic and sometimes less academic worlds. During that travel, I have met with many different people and had to learn how to open up towards and engage with the methods, knowledge and views of people from different communities. In that process, I steadily learned to detect and uncover my own biases, limitations and hidden assumptions – an unending process. But I also steadily learned about those of others and, in particular, the methodological assumptions and habits as implicitly or explictly defined by a particular field of research. Such assumptions are mostly productive and are even necessary for the foundation of a discipline. But these assumptions can also become counterproductive when the context in which these were originally put forward has changed significantly.

It is in this setting that I would like to introduce the notion of obstacles inspired by the French historical epistemologist Gaston Bachelard in his book *The formation of the scientific mind.*[18] He introduced the notion of *epistemological obstacles* to achieve what he called a *psychoanalysis of reason*[19] which aims at revealing epistemological obstacles in the history of scientific thought. Roughly speaking, an epistemological obstacle is a particular way of thinking, of using words, of being educated which might, in the longer term, become obstacles to seeing things differently. To put it in Bachelard's words:[20]

> An epistemological obstacle will encrust any knowledge that is not questioned. Intellectual habits that were once useful and healthy can, in the long run, hamper research.

I do not develop Bachelard's notion here in full detail[21] but introduce it in a general manner to show how a nomadic perspective on *both* history and philosophy of computing allowed me to identify and see more clearly such obstacles and their consequences in my own work and that of others. In what follows I consider a number of struggles and obstacles I encountered myself during my own travels within philosophy and history and indicate how they led me to my current position.

3. Computing as an object of study

At the time when I decided to have computing, programming and computers as my objects of study, that topic was not a common one within philosophy and I had to find my own communities in which I could present my work. Anecdotal was my accidental meeting with a colleague philosopher in the coffee room of my department. We started to chat about our topics and when I told him about mine the immediate reaction was: "well, that is quite a special topic especially for a woman."

At the time I was working at the Centre for Logic and Philosophy of science. In that group, there were quite some connections to computing.[22] First, the topic of logic related quite naturally to the above mentioned foundational work of Church, Post and Turing. Second, at the time, there was a new subcommunity being established around the philosophy of mathematical practices, a community that explicitly involved also more historical work, with one key actor being affiliated to my research centre.[23] I presented some of my work in both of these communities. There was a

general interest in the topic but never a strong engagement. Often I was the only one at conferences who was doing something in connection to computing. In retrospect, I would say that, at the time, for many analytical philosophers computers and computing belonged to the field of technology and less to that of science, whereas the latter was considered superior to the former.[24]

A recurring question then was if and how such a technology could really have a basic impact on, say, mathematics. One could for instance wonder in how far computer-assisted proofs are really fundamentally different from other, more traditional, proofs. As an example I take a paper by Jeremy Avigad, a well-known philosopher of mathematics who has also actively engaged with computing. In a paper titled *Computers in mathematical inquiry* the conclusion could not have been any clearer:[25]

> Ask not what the use of computers in mathematics can do for philosophy; ask what philosophy can do for the use of computers in mathematics […] What we need now is not a philosophy of computers in mathematics; what we need is simply a better philosophy of mathematics

To a contemporary historian of technology, such view might appear alien since it is not uncommon for a historian to look at the effect of a particular technology on discourse and practice without being bothered by essentialist questions such as: has the nature of proof changed because of the computer? But for many a philosopher, this was not a given and one needs to develop means to overcome what I would like to call the obstacle of historical contingency. That is, the assumption that the effcts of technological developments *are* contingent and can be bracketed when raising what I would call foundational questions like that of the nature of mathematical understanding. So, one of the objectives I put for myself *after* the PhD was to write on how and why the computer as a technology did not just have a contingent but a basic impact on mathematics. There, of course, a focus on *practices* was key and my tentative conclusion was that of a bottom-up approach tracing down characteristics of computer-assisted practices which are considered as specific to the use of the machine.[26] I would soon give up on the topic of computer-assisted mathematics as inscribed within philosophy of mathematics alltogether, realizing that I would probably never be able to provide a satisfactory answer to such foundationalist endeavours. Today, I would probably see it as the struggle of someone who, philosophically, subscribed more to existentialism than to essentialism and, by consequence, the rejection of the idea of *a priori* truths all together.[27]

My approach to this was anchored, on the one hand, in a historical research of particular cases in the history of computer-assisted mathematics and, on the other, my own experiences with computer experiments in the setting of computer science and mentioned in the previous section. The mix of these two approaches was not always appreciated by my colleague philosophers. My supervisor once told me that, we, the philosophers of science, have the task of telling the scientists how they *should* be doing their work. At the time, I was rather surprised by that remark because of the implicit assumption of a basic distinction between, on the one hand, the practicioner and, on the other, the philosopher, where the latter knows what is best. The reason for this was clear: I was no longer myself a pure philosopher since I had engaged with the practice and so, by consequence, could not have that distance anymore. This, I learned steadily, was considered a methodological no-go not only to some philosophers of science but also historians.

In my ambition to argue for the epistemological impact of the computer and computing in mathematics, I steadily also built an obstacle for myself and one which I call that of the historical singularity: in its more extreme form, it comes down to the idea of a fundamental historical disruption between the times before and after the first computers. We can find an instance of this, for example, in the work of the Dutch computer scientist Edsger Dijkstra when he wrote:[28]

> automatic computers represent a radical novelty and that only by identifying them as such can we identify all the nonsense, the misconceptions and the mythology that surround them

By the time I was working on computer-assisted mathematics, I had done some significant work on what is known as “one of the first” computers, the ENIAC (see also the next section). My study of that machine and the discourse that developed around it in the work of practiconers like John von Neumann, Derrick H. Lehmer and Douglas Hartree, provided significant historical arguments for why the first computers were considered disruptive from an epistemological viewpoint. But I also know now that I was misled in taking the claims from the discourse too much for granted and, by consequence, assigned too much historical value to “one of the firsts”. It was my collaboration with a historian of mathematics in the setting of a larger historical project on numerical tables in the history of mathematics which helped me realize that many of the ideas associated with “the firsts”, most notably that of a machine being general-purpose, had to be situated in a much longer history of developing more general devices, even when viewed only from the perspective of the history of mathematics.[29] Over the years my encounters (in person and in writing) with historians of science, technology and computing in particular,[30] have made this obstacle of the singularity more visible and problematic. Indeed, in the historical discourse, claims about “revolutionary ideas and technologies” are considered suspicious and rightfully so. The consequence of this encounter with historians thus resulted in a more nuanced view: while I do support the view that computing technologies have and are changing our ways of organizing and living our lives, extreme care is required when critically examining their effects, avoiding a discourse of disruption and singularities.

It is this historical technique of seeking continuations and parallel histories in what appear to be disruptions, which might offer something not only to philosophers, but most notably to practicioners, and by extension, non-informed citizens, who tend to blindly accept such discourse of disruption without further investigation. A recent example here is when ChatGPT and other similar machine learning technologies were launched in the programming discourse as being the end of programming (as we know it),[31] and, in the general discourse, as a disruptive technology threatening a wide range of human activities including teaching, journalism and art.[32] It is in such contexts that I believe historians could play a key role in helping to demistify the discourse of disruption, by pointing, for instance, at the larger business and labour histories underlying such claims or the recurrence of such claims throughout the history of the discourse.

Already during my PhD student years, I realized that analytical philosophy was not the only possible community for me and I actively seeked out others. At the time, I was not too aware of SIGCIS or SHOT, I only subscribed to the SIGCIS mailing list in 2010; I also attended a number of conferences that focused on Philosophy and Computing but by far the most important community I joined then, was that around the conference series “Computability in Europe” and which later resulted in the founding of the Association Computability in Europe. A key figure in that setting was Barry S. Cooper. My first CiE conference was in Swansea in 2006 and the welcoming words by Cooper deeply impressed me: the vision was that of an interdisciplinary conference bringing together researchers to think and discuss around topics related to computability, whatever their background. That is, *the community was not defined by a discipline but by the topic*. Most participants though had a background in mathematics, logic or computer science, but they also explicitly welcomed historical and philosophical contributions. Since that time I have attended thirteen CiE conferences and, steadily, became more and more involved with the Association and its conferences. At the time, it was the community with which I felt most at home even if most of the talks were quite far removed from my comfort zone. The reason why I felt at home there was threefold. First, it was a community which, very early on in my career, showed an honest interest in my work even if more historical. They had a clear interest in involving what they considered to be “professional” historians and philosophers in their community and, in particular, appreciated my technical knowledge of the topics I spoke about (see Sec. 4). Second, it was a conference that had the intention of being truly interdisciplinary. That resonated very well at the time with my own stance which was one of bringing together a variety of methods and disciplines (Sec. 6). Third, unlike the other conferences I had attended up to that point, the atmosphere was friendly,

welcoming and generous. It was a key concern of the people behind the conference to be as diverse as possible,[33] a situation that was not the one I was used to in philosophy.[34] Each of these dimensions have become key in my own later explorations into community-building (see Sec. 6) but the one I would like to focus on in the next section is the first and which can be seen both as an obstacle to myself and as a means to overcome others: the attention to and insistence on (technical) details.

4. The pertinence of details[35]

As explained earlier, my way into computing was anchored in my interpretation of Heidegger's essay *A question concerning technology.* I understood the computer as a technique that embodied the very idea of the "standing reserve". Thus, in order to work against a mode of existence in which not only the world around us, but also humans would be "ordered" into the "standing reserve", I considered it key to try and understand the historically constructed systems that enabled and at the same time were hidden away in my experience as an end-user.

While this is not a standard interpretation of that essay, I learned I was not the only one to have developed such view. Most important here was perhaps Friedrick Kittler, a well known media theorist who was very familiar with the Heidegger essay and had insisted in some of his work on the illusions created by software and the consequences of those for the so-called end-user. In particular, I was most impressed by one of his conclusions to the essay *Hardware – das unbekannte Wesen* in which he appealed to the philosophers as follows:[36]

> In this way, computers are sold whose architecture is not so much defined by the state of the art but by a pre-history or firm bureaucracy that crystallises into hardware right away. And if the ideal of software […] would ever triumph, the bureaucratisation would be perfect: The hardware, in spite of its programmability, would irrevocably be obscured under its packaging. To stop this coincidence from happening seems to be an eminent political goal. If computers are the first machines to reduce the contingency or incomputability of some, though not all futures to a finite degree, its own contingency should remain as open as possible. [...] If somebody went and wrote all the programmes hitherto running under the name of philosophy into hardware, that would be the goal itself.

It became a basic source of inspiration, amongst others, for my later work on ENIAC.
For the younger me, the only way for me to break through these created illusions of the familiar meant to engage actively with the techniques and technology underlying the interfaces with which I interacted as user. Since I had no training as a software designer or programmer, I assumed that the best way into this was to go back to some of the historical origins of the modern computer, assuming that these would be readable and understandable even for a simple end-user like me. As explained before, since I had had some training in logic, I was driven towards the basic papers by Church, Post and Turing, which, at that time, were often viewed as providing the logical fondations to the modern computer. These texts are quite technical and the reading was not easy but they allowed me to develop my own interepretation of what is today known as the Church-Turing thesis.[37] That interpretation was heavily anchored in my reading of Emil Post's work. That work was essentially a combination of historical and philosophical methods: archival work; understanding the historical relations between these different authors and interpreting their work against the logico-mathematical background of that time to understand their views on the "Church-Turing thesis". And, based on that, understand its possible relevance (hisorical and epistemological) for the modern computer and computer science. Clearly, that work required me to engage actively with mathematics and logic.
Shortly after the PhD, I also went more deeply into what was presented as one of the first computers, the ENIAC. That research was largely due to an accidantal meeting with another media theorist, Martin Carlé, who was working on an art project around ENIAC in which, amongst others,

the simulated sounds of ENIAC's vacuum tubes would order dancers in a choreography. But to achieve that, reconstructions of ENIAC programs would be required. To me, the idea of actually understanding how programmed computations can be achieved on a hardware level resonated very well with the above mentioned Kittler quote and so, together with Maarten Bullynck, I started to work on ENIAC. Also that work was quite "technical" in nature: our aim was to look into a multiplicity of practices that developed around ENIAC focusing on the work of several actors who had worked with the machine. The idea was to reconstruct the programs prepared by those actors. Based on those we wanted to understand the evolution in their practices to show the epistemologial and technical effects of using a high-speed programmable machine. Since our aim was to understand the practices that developed around ENIAC we had to engage actively with the details of the programming techniques and mathematical methods and problems to be tackled with the help of ENIAC. One series of papers focused on the work by Derrick H. Lehmer and contained detailed paper reconstructions of the number-theoretic problem that was set-up on the machine. We also did a reconstruction of two conditionals – in current terminology these are "If … then ...else" constructs – on the early ENIAC.[38] These detailed investigations not only provided us with an improved understanding ofwhat it means to run a program *physically*, but also gave insight into how and why a practice is affected by the particularities of a machine, resulting, ultimately, in new machine and programming techniques. One example here are the reflections by Haskell B. Curry – a logician – who, based on his ENIAC work and anchored in his logical background, developed a so-called theory of program composition for the IAS machine and which was quite different from the more famous flow diagram approach by von Neumann and Goldstine.[39]

So, roughly speaking, my earlier historical work can be divided into two clusters: one that focused on the early history of mathematical logic (1920s-1940s) and one that focused on programming practices around one of the first computers. These two clusters were methodologically related through an approach that went deeply into the mathematical and programming techniques involved. The main connection between these two was anchored in a question of my PhD, left largely unresolved or underexplored: what, exactly, are the historical and epistemological connections between the theoretical techniques and methods developed in logic and mathematics on the one hand and actual concrete programming practices on the other?

The particularities of that work confronted me with an important obstacle that I had to overcome: the above question already presupposes that logic *did* play a role in the shaping of the computing field and, in particular, the rise of the first computing and programming practices. I know now that during my PhD I was too biased by a common narrative which claimed that the modern computer can be (partially) anchored in logic and mathematics. While I still believe this to be true to a certain extent – roughly speaking, the first computers were built to solve mathematical problems – I started to question that narrative. The technical work on ENIAC was key there: it allowed me to realize that these so-called logical foundations and, in particular, the notion of the universal Turing machine, was in no way "essential" to the conception of the first computers or the notion of programs.[40] More important perhaps was the insight that technical histories *alone* cannot provide a proper explanation for the origin of such narratives. If we want to understand, for instance, where the idea of logic being foundational to computing comes from, one not only needs to study the technical discourse but also the social dimension of a new academic field, struggling to find a disciplinary identity. The study of the technical discourse enables an understanding of when and how certain logical insights were effective for the technique itself. One could call this the epistemological level of analysis. An example is the introduction of the universal Turing machine concept in the 1950s discourse to tinker with the idea of building the smallest possible computers (see Sec. 5 for more details); another is the use of constructive type theory to build proof assistants like COQ. But the study of the general discourse in terms of disputes, community-shaping, etc also enable one to see when and how certain logical insights were introduced to shape and support a particular social identity. A generic example here comes from the so-called "software crisis" where people like Perlis and Dijkstra relied on a mathematical view on programming to give the field (academic) legitimacy. The realization that I

was initially too biased by what appeared to be standard narratives, contributed to a general scepticism towards accepted narratives and a conviction that any historiographic method, ultimately, has its own blind spots and biases.

My earlier historical papers were not published in "traditional" venues for historians of computing (e.g. *IEEE Annals for the history of computing*). Some were published in edited volumes on foundations of computing/philosophy of science; most of the ENIAC papers appeared in special issues resulting from the CiE conferences and some ended up in venues for history and philosophy of (mathematical) logic. I had regular interactions with historians but these were mostly European historians of mathematics which have quite a different style compared to US-based historians of technology. In all of these contexts, it was considered failrly unproblematic to draw historical conclusions based on a detailed study of particular practices.

It was thus quite confronting when I started to meet some "professional" historians of computing and engaged more actively with the standard work in that community. I was reading up on historiography and in particular on Michael Mahoney's work. An influential insight from that work was that we should move away from machine-centred histories of computing and consider instead histories that focus on communities of computing, social groups of people who conceive of and envision computing in particular manners.[41] I was also following closely some of the discussions on the SIGCIS mailing list; and I also had some personal encounters that, initially, were quite alienating for me. It appeared that in the "professional" community of historians of computing, the kind of work I was doing was considered "old school", "internalist", "too much history of ideas", "not for historians but for computer scientists", 'machine-centred"; "lacking social and cultural context"; etc. Steadily I learned about the reasons why my work was classified by some in this manner: apparently, I had missed out on the history of the history of computing which developed, so goes one official narrative, from an internalist, technical and machine-centred and hero-centred kind of history writing, written by computer scientists, to more externalist histories that focus on topics such as the programmer's profession; the problem of gender in computer science and programming; the business of programming; to name but a few. Each and every one of these topics are basic and each require their specific approaches.

But that need not be a reason for being deprecatory about histories that *do* focus on the practices of programming and machine use itself nor to assume that somehow these more technical developments can be safely ignored when writing, for instance, about the profession of the programmer. A clear argument here was given by the historian of mathematics, Karine Chemla. She argued that historians of science also need to aquire a certain level of literacy when dealing with historical sources to understand the different skills that are required by different scholarly milieux.[42] That is, having as a historian a certain level of literacy is intrinsically important to the field of history of science itself.

The depreciation of technical histories was, of course, but a personal experience and should not be generalized. However, it was a fact that, at the time, the majority of history of computing books being published, hardly engaged at all with the more technical aspects of computing, programming and software design with some even explicitly opposing them to highlight the novelty of the work. A confirmation and clarification for this arrived, for me, when in 2014 a discussion ensued on the SIGCIS list around a recent talk by Donald Knuth, a well-known computer scientist and important advocate for the history of computing. The talk was titled "Let's not dumb down the history of computing". As is clear from that title, Knuth was not feeling very happy by the current state of affairs in professional history of computing as it appeared to him. The feeling was based on his reading of a paper by Martin Campbell-Kelly titled *The history of the history of software* and published in one of Knuth's favourite publications, IEEE Annals for the history of computing. In that paper, Campbell-Kelly sketched a development in the history of the history of software whereby earlier publications were mostly technology-focused, identified as being of the "low-hanging fruit variety", and it was celebrated that, steadily, these kind of more technical publications

disappeared from the field of history of software. Knuth , who is known to have drawn insights from the history of computing and mathematics for his own work, was of course quite upset since, in his reading, the paper implied that we should be happy that technical histories are no longer being written. One conclusion from the discussion on the SIGCIS mailing list was that, of course, current history writing is not in the style Knuth prefers, since it is not realistic for someone who seeks a position at a (US) history department to write in that style *and* find the necessary appreciation for that kind of work at a humanities department. If computer scientists prefer more technical history, then they should make sure there would be funding for such positions at their departments. The problem was thus regarded largely as an institutional and social problem.[43]

From my own experience at the time, however, technical histories are not just relevant from a computer science perspective and the debate that ensued hinted for me at a number of methodological blindspots within the field of the history of computing itself. Of course, it is true that the "job market" forces researchers to do their research in a particular manner, but that need not mean we should agree to ignore a particular kind of method simply because it does not fit the agenda of hiring committees. What was missing for me in the Knuth passage was a deeper reflection on the potential of a history of computing which, rather than being defensive about particular methods and offensive towards others, attempts to consider the value of a multiplicity of approaches for the field *proper*. Let me explain this from the more technical perspective which is mine, keeping in mind that this is an illustration of the plurality point and not a generalization of the necessity of technical histories. I discuss two cases exemplifying two seperate points.

First, there is a significant risk of blind spots and simplifications if one relies on just one level of analysis. I take the example of Nathan Ensmenger's book *The computer boys take over Computers, Programmers, and the Politics of Technical Expertise*, to illustrate this point. The book's approach is quite clearly inspired by Michael Mahoney's historiographical work mentioned above and explicitly focused *not* on the machines, but on the people that built and maintain the software. That decision, to focus on the people, however comes with a method that inscribes the history of (the profession of) the programmer in a social history whereby, "*[w]hat might on the surface appear to be disagreements about the particular technical challenges associated with software development were in reality local disputes about organizational power and authority and […] about the peculiar character of the people involved with software development.*" In other words, technical discussions are recast as discussions about social identity and professional status. This is of course a valid and valuable approach but in its focus *only* on the social dimension, there is a risk of no longer being able to differentiate between what I called earlier the epistemological level of analysis and the social level of analysis. Indeed, if developments in programming are recast as social phenomena then it becomes difficult to perceive, for instance, the finer-grained differences between *why* different people used or introduced a particular programming technique like object-orientation;[44] or, the realization that the dichotomy between programming as an art or as a science, while perhaps true from an externalist view, breaks down once one engages with the details of the work of some of those supporting the "scientific" view.[45] Moreover, in reference to the above literacy argument of Karine Chemla, a focus only on the social level of analysis without engaging with the more technical discourse, makes it hard if not impossible to diversify between the different skills required or assumed by different (historical) conceptions of the profession of the programmer. To be clear, this is not intended as a criticism on this book per se, but rather the expression of a general worry: if one sticks to just one level of analysis *and* assumes that other levels of analysis can be reduced to it or can be considered irrelavent to the research questions one wants to resolve, one might end up with a narrative that orders and represents concepts and techniques in a particular manner with the risk of ignoring other possible orderings that would contribute to a more completed picture.

Second, if ones level of analysis largely ignores the technical discourse then it is quite unavoidable that you might have to rely on central concepts and notions that were historically developed in a more technical and historically or geographically distant discourse and which you simply take for granted in their more comtemporary meaning without further study. Examples for the history of

computing are: “algorithm”, “code”, “software system”; “computability”; “A.I.”; “automatic programming”, etc. I would like to briefly focus here on the notion of algorithm. Historians, when using that notion, in fact rely on a very recent and particular interpretation of that concept which essentially comes down to the idea of having a finite step-by-step method to solve some problem, inherited essentially from the way Donald Knuth and the ALGOL community introduced it in the US programming discourse. The comparison with recepies is never far removed. While perhaps one could then savely ignore the complex semantic and technical history of that notion when it is used in passing only, this becomes more problematic if the concept is a key notion in the narrative being developed. I illustrate this here with one example.

Common semantic associations with the common sense understanding of algorithm is that of control, formality and solvability and this is also the manner in which the notion is introduced in the book *How reason almost lost its mind: The strange career of cold war rationality.*[46] In that work, the notion of algorithm is given a central role to describe the rise of so-called cold war rationality in a US context.[47] [48] The elaboration of the notion in the monograph *Theory of Algorithms* by the Russian mathematician A.A. Markov is mentioned in passing only. During my PhD years I had explored to some extent writings from Russian mathematicians and logicians from the 1940s to 1960s and noticed the prominence of the notion without really understanding where it was coming from. This was all the more interesting to me since it was largely absent from the Western logical discourse: Post nor Turing really make it a topic and use other terms, whereas Church used it merely in reference to mathematics. Moreover, in the first one or two decades of the Anglo-Saxon history history of the modern computer, “algorithm” is also largely absent from discourse and replaced by the more common “method”. It would be many years later that I would finally have an answer to the question of the introduction and use of the notion in the Russian-speaking context thanks to a collaboration with two historians, Ksenia Tatarchenko and Anya Yermakova. A key outcome of that collaboration was that an explanation of “algorithm” in its original Russian-speaking context requires a longer term perspective which does not fit with the US perception of cold-war rationality. To the contrary, it fits much more with what one could call a humanist ideal of computer literacy.[49] That outcome was only possible because of our integrating of institutional, social, cultural and more technical histories focusing on the content of certain logical and mathematical work. In other words, the US narrative of cold war rationality was counterbalanced by an alternative history of *algorithm* which is more about humanistic values and visible only through the integration of a number of different approaches.

Let me know briefly return to my own biases and the Knuth passage. When I made my first steps into the history of computing, my preference for a more technical history writing went hand-in-hand with a certain bias that required correction by integrating different levels of analysis and, in the case of logic, to try and carefully differentiate between what is socio-historically constructed and what is epistemologically relevant as much as possible. The latter level of analysis is one that is heavily anchored in my particular background in analytical philosophy but also in my regular meetings with computer scientists in my first years. The former is one I steadily learned to accept thanks to my meetings and discussions with a variety of historians, but also my meetings with computing people who simply do not care much about logic when it concerns their practice. The Knuth passage then for me was both a confirmation of a situation I had experienced myself but also a disappointment: instead of having a serious debate about the possibility and desirability of integrating or, at least, complementing of different levels of analysis within the field of history of computing *proper*, the discussion resulted in a conclusion of an academic division of methods due to institutional and social realities. The need for such a methodological debate is, I believe, partially anchored in the diversity of computing as a practice and as a concept to which I return later. It is one towards which the current book is actually making a significant contribution.

5. Presentism or why are we writing histories of computing?

In the SIGCIS discussion on Knuth's lament about the lack of technical histories in the history of computing, Bill Aspray wrote:

> This discussion is not new. When the history of science began to professionalize after the Second World War, at first the scientists (mostly physicists) welcomed them with open arms. But the honeymoon soon ended when they found that the historians were not going to simply be apologists for the scientists but might have their own set of interests and interpretations that did not square with those of the scientists.

Indeed, the professionalization of the history of science was quite heavily anchored in sometimes virulent fights with the scientist's whose fields professional historians were attempting to study.[50] In a recent paper by the historian of science Kostas Gavroglu titled *The Sisyphean Fate of History of Science,*[51] Gavroglu describes this professionalization of the history of science and the disciplinary fights that were part of that process in mythological terms referring to Sisyphus. Sisyphus was punished by the Gods because he had cheated death not once but twice. The professionalization of the history of science, despite the resistance from scientists, is then described as history of science having tricked death a first time.

This historical passage, of course, could not have happened to the history of computing in the 1970s since the field did not exist yet. That is, at the time, it were mostly still computer scientists who were writing their own histories.[52] But, as suggested by Bill Aspray, the Knuth passage could be read as an instantiation of such passage in the history of computing. It is in such settings that historians of science and technology developed a rather high sensitivity against what is known as anachronisms or, more specifically, Whiggish history. In its more extreme form this comes down to rewriting the history of field *x* in terms of field *x'*s contemporary terminology. In general, it is the idea of writing history with reference to the present. Anti-whiggism then is about "treating the past on its own terms as much as possible, and not simply as a runway aimed at the present" as Michael Gordin described it in his lengthy review of Hasok Chang's presentist *Is Water H2O?: Evidence, Realism and Pluralism*.[53] Such opposition against Whiggism is almost natural to historians of science and technology, that is, "we have wired it into the central core of our field *as a discipline*."[54] Chang, in the reviewed book, proposed a presentist view on the history of science under the banner of *complementary science.* Roughly speaking, it is the idea of learning from our pasts for current scientific practices, as a means to asking questions that are excluded from current practice; to encourage re-examining accepted wisdoms and for showing how certain truths and methods have been accepted as common sense. Chang, however, is not a pure historian of science but part of the school known as integrated history and philosophy of science. The inclusion then of a more presentist view can be seen as a normative dimension anchored in certain philosophical traditions.

Not knowing too much about all this several years ago I was not very conscious that, in fact, I was, to a certain extent, a presentist too nor of the reasons why this might be a problem for the "pure" historian. But then I had an encounter with a well-known historian of computing who confronted me with this "issue" telling me that it is bad practice to write history in function of the present. Initially, that comment came as a surprise and I was upset if not offended by. However, I understand now that (1) they were in fact right but only to a certain extent and (2) that this was perhaps an issue for thé historian but not for me, a nomad who, clearly, cannot and does not want to escape fully her own education as a philosopher.

So what does my presentism entail? It surely has certain aspects in common with Chang's presentism: a deep motivation for doing history of computing to critically think about our current digital condition. In that sense, one could say I give purpose to a history which it surely did not have at the time it was happening. Emil Post, when writing about solvability surely was not thinkig

about the limits of the modern computer; and Lehmer, when working around ENIAC was surely not using a notion of “program” that can be equated to our current notion. By doing this kind of history writing in function of today, it is unavoidable that I have sinned against the hard-wired anti-whiggism that is part of the general fields of history of science and technology and, by consequence, also history of computing. I have chosen particular topics and methods and ignored others because of my presentism and, probably, have sometimes fallen into the trap of describing things from the past in current terminologies.

Let me elaborate a bit more on this sin. As described before, my motivation for doing history of computing is anchored in the belief that our current situation is problematic and that we are in need of a deeper, critical reflection to change that situation. It is a normative dimension comfortable for philosophers, less so for historians. In particular, for me, it is about investigating the conditions which have led to most of us having become illiterate end-users, an investigation which, rather naively perhaps, started with a more technical history writing on the so-called logical foundations and the first computers. To put it differently, for me, doing history serves another purpose which I prefer to call *political*. In general, it is the idea to render transparent computing by showing how it is shaped by its material, economical, social, notational and other practices and conditions to go against certain “ideologies” which I consider to be dangerous. It implies also that when I am writing about the history of computing I do not only have in mind a readership of fellow historians but people who share with me similar concerns, whatever their background, thus also including, but not exclusively for, programmers and computer scientists.

In most of my historical work, the “sin” manifests itself most clearly when introducing certain new concepts to study a longer historical development. One example is given in the paper *Less Is More in the Fifties: Encounters between Logical Minimalism and Computer Design during the 1950s*,[55] that was briefly mentioned in Section 4. *My* co-authors and myself introduced the notion of logical minimalism to tie together and compare certain developments in mathematical logic from the 1920s-40s with engineering practices from the 1940s to early 1960s. In particular this was to relate developments in matematical logic to tinkering about the minimal number of components to built circuits and computers. One important methodological challenge of that work was to understand the transformations and transpositions that certain logical techniques had to undergo when introduced in the discourse of engineers.

A more recent example is given in a forthcoming chapter *Notations. There is no escape,* written by the PROGRAMme collective.[56] In that chapter, we provide a framework to think about the meaning of programming beyond the common “programming in a programming language” to include, also, for instance programming with visual icons or prompt engineering. To that end, we introduced the notion of *notational programmability* as a means to compare different programming notations relative to their programmability, keeping in mind that any such notation always needs to be seen in its own local historical context. It is used as a positive explanation for the explosion of programming notations instead of the more negative “Tower of Babel” metaphor that is often used in the programming discourse. It allows us to conclude that any attempt at fully automating the programmer has significant implications for the user of the notation. That chapter spans a history which starts around 1800 and ends in our current times. It clearly makes the sin of presentism despite the rather extended historical scholarship on which we have relied for the development and elaboration of the concept. But it is one that had to be made for the purpose of the chapter, since it allowed us to evaluate developments in the past and current discourse from a very different angle, one that is historically motivated and informed. That is, while we do use a new concept anchored in concerns with present developments, it is that very same concept that also allows us to take distance from the current discourse to place it in a non-linear and complex historical development of programming notations.

These two examples hopefully give the reader an impression of my presentism. I know from experience that “pure” historians might struggle with it. Typical replies to presentations of these

works at conferences with historians in the public are: but are you saying that practicioners themselves were guided by notational programmability in their choice for or reliance on a certain notation? How can that be true since they never used such concept explicitly? Is it not misleading to speak about logical minimalism with reference to two very different practices with neither of them using that concept explicitly? These are criticisms that, by now, I respect and understand but, at the same time, given their frequency, have also made me wonder about more hidden whiggisms in the works of "pure" historians. I think here for instance of the examples from the previous section, that is, the use of current terminologies from the technical discourse (often that of Big Tech) without taking into account the rich and more diverse histories of these very same terminologies.

Besides the perhaps too obvious argument that no one can wash their hands in complete innocence here,[57] the more important question that I want to raise here is in how far our anti-whiggism has become too much of an obstacle, anchored too much in our own pasts and our struggles for disciplinary independence. It is here that I would like to briefly return to Kostas Gavroglu's paper and the possible second death of the history of science and, in general, the social sciences and humanities. He argues that our field is under attack again, this time perhaps less by the scientists but more by the politics of academic funding. More than ever, decisions about how to spend research and education budgets are guided by instrumentalist short-term visions in terms of job markets and economic impact. We might call these short-term presentist views. The humanities and social sciences are the first victims to fall: they have to survive by focusing only on the immediate societal returns from investments in science and technology.[58] This is of course a presentism that should be avoided since it does not consider or take into account our fields as being fields *on their own*. Interestingly, Gavroglu sees a way to trick death a second time by refocusing the social sciences and humanities on our digital condition and the threats and challenges it poses to our identities, our freedoms, our thinking, our educational systems, our democratic political systems:

> in the past 200 years democracy has been threatened by wars, dictatorships, coups d'état, famine, pandemics, and civil strife. But never before has it been threatened by unseen, immaterial, and mathematical entities called algorithms that dominate our everydayness. We need to acquire a deep understanding of this historically unique state of affairs. What other arguments do we need in order to let the social sciences and humanities chart their own research strategies instead of constraining them to produce "useful" results?[59]

While one might well argue that this should not be the *only* raison d'être for our fields, it surely is a viewpoint that could be picked up by historians of computing to contribute to "*acquir[ing] a deep understanding of this historically unique state of affairs."* It would mean that perhaps we should dare and critically question our old defenses, anti-whiggism included, which were effective perhaps in the 1970s and 1980s but might be less effective under this new attack. This is not to say that we should get rid of all of our previous standards, but rather to have a serious discussion on what could be an acceptable presentism from a historian's viewpoint, if any.[60] This is not to imply that my personal presentim is on the right track and there are certainly other options to explore. One interesting path was offered quite recently in the monograph by Thomas Haigh and Paul Cerruzi.[61] In an extended methodological reflection on the use of grand narratives, Haigh explicitly considers the risks of Whig history in a setting that describes the history of the computer as that of the computer steadily becoming (practically) universal and indicates how its unaccaptable form can be avoided.[62]

6. The computer is not one thing but many different things

As explained earlier, already during the writing of my PhD I was quite convinced that computing requires a multi-disciplinary approach. Clearly, my presentism results in an approach that combines historical research with philosophically inspired views and insights from the practices of computing

and programming. But these should be seen against the background of a much broader multidisicplinarity, anchored in two well-known and related views:

(1) computing and/or programming itself cannot be reduced to any particular existing field of research (e.g. mathematics, literature, engineering, logic, etc). This is a view that has received some acceptance in the history of computing field itself. It was most famously captured by the following quote by Michael Mahoney: "The computer is not one thing but many different things and the same holds true of computing."[63] As is known, in the history of the computing field proper, there have been quite some discussions about the nature of the computing field as a discipline going from more mathematical conceptions to engineering views and multidisciplinary views.[64] But there have also been other views such as Donald Knuth's conception of programming as a literary activity;[65] or Warren Sack's historical contextualization of computing and programming from an arts and humanities perspective.[66]

(2) the technology of computing has steadily become ubiquitous.[67] This generally refers to the idea that computational technologies have steadily pervaded our lives (scientific, personal, social, etc). It implies that a study of computing and programming should not and cannot belong to just one field of study.

These two perspectives naturally lead to a study of computing and programming that is open towards a variety of disciplinary and methodological perspectives. Amongst others, it provides an additional argument for a braoder methodological debate within the history of computing proper about the need and potential of bringing together a variety of historiographic methods which is one of the aims of this book volume.

But how can one achieve such diversified perspective on computing and programming as an individual? Even as a nomad one must be realistic and know that no individual can ever achieve that on their own, determined as we are by that path and the limits of time. One possible path then towards such a methodological and disciplinary diversified perspective is through collaborative community building. In the last 10 years I have dedicated much of my professional time exactly to this kind of activity, to the detrement of others.

A first basic step in that direction was in collaboration with Giuseppe Primiero, a philosopher and logician who worked at the same philosophy department as I did as a postdoc. Just like me, he felt that we needed a space in which we could combine and mix different views and approaches. During a meeting in the coffee room of the philosophy department we started to talk about some shared frustrations of that time – this was in 2009 – and we developed the idea of organizing a conference on the history and philosophy of computing which would explicitly involve also more technical talks and views by trained computer scientists and programmers. The aim was to think together about the computing field without assuming that one field would be superior to another. The first HAPOC conference was a fact in 2011 and, to our feeling at least, it was als a success. Its leading quote of that conference was the above mentioned Mahoney quote.[68] HAPOC soon developed into an organization under the umbrella of the IUHPST (International Union for History and Philosophy of Science and Technology) and in particular its two divisions DHST and DLMPST.[69] We steadily built up a number of recurring events including the bi-annual HAPOC conference (odd years) and the bi-annual symposium on the History and Philosopy of Programming (HAPOP, even years). In the main HAPOC conference we have invited well-known historians of computing[70] alongside philosophers, media theorists, artists and computer scientists. One key feature of this "community" is that people who identify with it often have rather clear disciplinary and methodological stances but enjoy and see a need to open up their minds to those of others from other milieux.

But HAPOC also has its shortcomings: besides the fact that sometimes talks might be so far removed from someone's interests and competencies, that they become incomprehensable to some, the conference really is about *pluri*disciplinarity. That is: what we achieved was the creation of a

space where people with diverse backgrounds feel relaxed enough to share their views and research results with a diverse public, hoping that this would be enriching for the different participants. But going from there to a collaboration which integrates the different fields and methods is quite a different step. This has happened occasionally in the wake of HAPOC but was not its core business.

A step in that direction was made with the PROGRAMme project which brings together a variety of researchers (programmers, historians, logicians, artists, media theorists, computer scientists, philosophers, etc) to try and work together *as a collective* on one and the same question: what is a computer program? The rationale behind that project has been from the start the idea that conceptions and understandings of programs really depend on the historically contextualized practice in which one is designing, programming, studying and using programs. To put it quite bluntly: a user of phone Apps is not the same user as that of ENIAC and both will have very different conceptions and understandings of programs. To capture this, the project has been defined around four core modalities of programs. As abstract objects, programs are formal; as textual objects they are notational; in the sense that programs are stored and executed on a computing device, programs are physical; and to the extent that programs are made, shared and used in our human world, they are socio-technical. These modalities enable a range of different views and ways of handling programs. They are reflected in a number of different fields that focus on programs, both within the humanities and the computing field, each of which tend to prioritize one of the modalities. For instance, to view programs as text is the approach favored in the broader domain of critical code studies but also corresponds to the way programmers often view programs when "coding" them.[71] In the project, rather than favoring just one of these modalities, or any of the methods resulting from them, the idea is to look at programs from the four different modalities together. This is rooted in the idea of breaking through illusions created by one perspective only and to open up the possibility of other configurations. It implies, amongst others, that standard narratives are not taken for granted but critically examined. A good example here is the chapter that was mentioned earlier *Notations. There is no escape.* For some in the group it was initially considered unconventional that this chapter would not have “programming languages” as a main topic. But we did not want to start from such standard notion or conception that history has given us, but challenge it by focusing instead on the very different notion of notations and their programmability. That particular chapter was written and conceived by a PROGRAMme subgroup including three historians (or four, if I count myself as a historian); two computer scientists with different backgrounds, a media theorist and a manager turned philosopher. It took around three years, and numerous on-line and in-person meetings, to finally arrive at a text that was acceptable to all. This is to indicate that such collaborations are not evident or straightforward but assume significant commitment of those involved.

From a history of computing viewpoint one might of course wonder what is there to gain from such a mixed approach given the time and commitment it requires. Here I can only speak from a personal perspective. For me, the gain is in the confrontation not only with other methods but also other insights, other problems, other facts that I simply would not have known about or have been able to perceive. It has opened up my view on the literature and the sources, allowing me to see certain passages in the history of computing in a different light; it has forced me to focus more on the bigger picture than on the particular details of a number of specific cases; and it has made me realize quite explicitly some of my own obstacles and biases of which I described some in this chapter.

7. Concluding – opportunities as a nomad

As announced at the start of this chapter, I have tried to develop my personal views on the history of computing based on my academic nomadism. In doing so, I have indicated ways in which the history of computing could contribute to an improved understanding of our current situation, with a focus on the future. My personal process has led me to the realization of a number of obstacles in

my own work but has also to question certain methodological assumptions that are necessary for the shaping and definition of a particular field. In particular I argued for the significance of bringing together a diversity of methodological approaches in history of computing and questioned the anti-whiggish stance that we often find in the more general field of the history of science and technology. I also briefly sketched my own alternative approach which tries to bring together a number of different fields and methods and which requires significant efforts into community-building.

It has been my experience with such broader collaborations across disciplinary divides that has given me the courage to write this rather personal self-reflective chapter; to be explicit about some of my own obstacles and biases; and to be honest about and share my personal views on the history and philosophy of computing. It has led me to the conviction that what we need today is not an intensification or confirmation of disciplinary and methodological boundaries by working against one another. That would only confirm and encourage a political system that celebrates a naive and ideological vision of neodarwinism applied to research and education. Instead, I hope that in the future there will be more opportunities to explore ways of working across such boundaries together for changing locally and perhaps only temporarily our current and historically developed academic situation.

1 This is an earlier draft of a chapter to appear in the volume: Bill Aspray (ed.), *Writing Computer and Information History: Approaches, Reflections, and Connections*, Rowmann & Littlefield, forthcoming.

2 Supported by the ANR PROGRAMme project ANR-17-CE38-0003-01.

3 The official title of my study was “Kunstwetenschappen”, “Art sciences” in English. The main focus however was on art history. The first two years were a mix of art sciences, history and archeology; during the last two years we had to chose a specialty. I chose Visual Arts and Architecture and did a master thesis on linear perspective.

4 Martin Heidegger is a well-known German philosopher who has been very influential amongst others on some basic French thinkers like Derrida or Foucault. The essay mentioned is considered a key source in the philosphy of technology. It is known that Heidegger supported the nazi regime. In april 1933, shortly after Hitler had become chancellor of Germany, he was elected as the rector of university of Freiburg. He accepted that position and soon after became a member of the nazi party. About one year later he offered his resignation but remained a member of the nazi party until the second world war ended. This chapter is of course not the place to discuss how to handle an influential philosopher who was also a nazi and, in particular, the question of the relation between Heidegger’s nazi politics and his thinking. There have been recurring and heated discussions on this amongst renowned philosophers like Derrida and Jankelevich and I refer the reader to these.

5 Carnap famously called metaphysicians like Heidegger “” “musicians without musical ability” in his most significant critique Rudolph Carnap, *Überwindung der Metaphysik durch logische Analyse der Sprache"*. Erkenntnis. **2**: 219–241. 1931 (English translation: Rudolph Carnap. *The Elimination of Metaphysics Through the Logical Analysis of Language.* In (ed.) Ayer, A. Logical Positivism. New York: The Free Press, Co 1959). The critique of Carnap on Heidegger went quite far and is often seen in the history of philosphy as indicative of the larger divide between “continental” and analytical philosophy. See for instance chapter 6 *A Case Study in Misunderstanding: Heidegger and Carnap* of: Simon Critchley, *Continental Philosophy: A Very Short Introduction*, Oxford University Press, 2001.

6 Alienation is a well-known term in marxist and Hegelean philosophy. Simondon, a well-known French philosopher of technology used it in his *On the modes of existence of technical objects* (Gilbert Simondon, De mode d’existence des objets techniques, Editions Aubier, 1958; translated to English as: Gilbert Simondon, *On the modes of existence of technical objects*, Translated from the French by Ninian Mellampy with a Preface by John Hart, University of Western Ontario, 1980. A more recent translation is: Gilbert Simondon, *On the modes of existence of technical objects*, Translated by Cecile Malaspina and John Rogove, University of Minnesota Press, 2016). In that work he speaks of how humans have become alienated from the machine caused, by, amongst others, their lack of understanding of the machine. As he wrote: “Culture behaves towards the technical object much in the same way as a man caught up in primitive xenophobia behaves towards a stranger. This kind of misoneism directed against machines does not so much represent a hatred of the new as a refusal to come to terms with an unfamiliar reality. Now, however strange this reality may be, it is still human, and a complete culture is one that enables us to discover that this stranger is indeed human. Still, the machine is a stranger to us; it is a stranger in which what is human is locked in, unrecognized, materialized and enslaved, but human nonetheless. The most powerful cause of alienation in the world of today is based on misunderstanding of the machine.” (Idem, p, 1)

7 This is how the Heidegger essay is often interpreted in philosophy of technology.

8 Essential in my interpretation of the Heidegger essay is his use of a quote from the German poet Hölderlin: “Wo aber Gefahr ist. Wächst das Rettende auch” (English translation: B”ut where the danger is, also grows the saving power”) from the poem Patmos.

9 Liesbeth De Mol, *How to talk with a computer? An essay on Computability and Man-Computer conversations*, Off Topic, Zeitschrift für Medienkunst der KHM, vol. 1, 2008, pp. 80-89; and Liesbeth De Mol, *How to talk with a computer? An essay on Human-Computer conversations continued*, unpublished, available at: https://hal.univ-lille.fr/hal-01305968/document

10 A decision problem is, essentially, the problem of finding a uniform and effective procedure for solving every instance of that problem. An example of a solvable problem is the problem to decide for any two integers x and y whether or not y is a divisor of x. Indeed, there exists at least one finite procedures (we would call thee algorithms today) to solve every instance of that problem. An example of an unsolvable problem is to decide for any enunciation in first-order logic whether or not that enunciation is derivable or not in first-order logic. This latter problem is the most famous version of the Entscheidungsproblem.

11 The Turing machine model was introduced in the computing discourse in the 1950s and early 1960s as a formal model of the modern computer; the formalism of Post and Church were more relevant in theoretical reflections on programming. Church’s formalism known as lambda-calculus is today still a key formalism for people working in formal verification; Post’s production systems were picked up in the foundations of compilers as well as in certain early developments of AI (most notably the work of Newell and Simon). However, as has been shown elsewhere (see eg L. De Mol, M. Bullynck and E. G. Daylight (2018), *Less is more in the fifties. Encounters between logical minimalism and Computer design during the 1950s*, IEEE Annals for the history of computing, vol. 40, nr. 1, 2018, pp. 19-45), extreme care is required from the historian here to understand that the use of such formal devices in a different discourse (eg programming) involves a complex process of transpositions often resulting in a rather weak connection between the original discourse and the target discourse.

12 The title of the PhD was: *Tracing Unsolvability: A Mathematical, Historical and Philosophical analysis with a special focus on Tag Systems.* It was defended on May 23, 2007 at Ghent University. My jury consisted of two

analytical philosophers (Erik Weber and Jean-Paul van Bendegem), one logician (Diderik Batens) and two theoretical computer scientists (Maurice Margenstern and Martin Davis).

13 Initially, I hoped to rely on tools of visualizations using fractal geometry but realized quite early that these would not get me any further. That initial research did result in my first published paper: L. De Mol, *Study of fractals derived from IFS-fractals by metric procedures*, Fractals, vol. 13, nr. 3, 2005, pp. 237-244.

14 The only public I ever really felt at home was a public of artists who think less in terms of disciplines. My meetings with that community showed me the way of understanding and looking at research in a very different manner.

15 In particular this is committee 53, *Sciences en société : production, circulation et usages des savoirs et des technologies* (Sciences in society: production, circulation and uses of knwoledge and technologes)

16 The philosophically informed reader might see here an implicit reference to the concept of nomad science as developed by Deleuze and Guattari in their *A thousand plateaus* and, in particular, the *Treatise on Nomadology – war science.* Andrew Pickering used that concept to describe cybernetics (Andrew Pickering (2010). *The Cybernetic Brain: Sketches of Another Future.* Chicago, Illinois: University of Chicago Press.). In that sense, there are probably some similarities to be found. Since I am not a specialist however on this work, I did not want to fully engage with that concept and rather use the notion of nomad in a common sense manner.

17 The above description of my personal journey might create an impression that it was always my intention to be a nomad. That, of course, is not true and but a projection in retrospect. Reality is that it took me many years to realize that my "home" is not and should not be in any particular field of research and that there are opportunities in the academic system for researchers like me.

18 Gaston Bachelard, *The formation of the scientific mind. A contribution to a Psychoanalysis of Objective Knowledge*, Clinamen Press, Machester, 2002. Originally published in French as: Gaston Bachelard, *La formation de l'esprit scientifique. Contribution à une psychanalyse de la connaissance objective, Vrin, Paris,* 1938.

19 Ibidem, p. 29.

20 Ibidem, p. 25.

21 Bachelard's analysis was focused first of all on science and not on history and philosophy per se. Moreover, one should keep in mind the times when this work was being written (1930s) when many philosophers and epistemologists had a rather positivist and progressivist view on the sciences which I do not share.

22 Initially, I was affiliated to another group (Laboratory for Applied Epistemology). However, over the course of my first year, I encountered a number of basic and unresolvable problems with my then supervisor and was happy when someone else at the department was willing to take over as a supervisor.

23 This was the philospher of mathematics Jean-Paul van Bendegem. There were other topics that naturally related computing to logic and philosopy of science. For instance, philosophy of language is a classical field in philosophy of science which has some clear connections to programming languages; the question of valid reasoning had strong ties with certain developments in AI, etc. These topics however were not really a theme in my research environment of that time.

24 Of course there were some exceptions. Most notably here was Peter Galison's work which, amongst others, showed how technology and science cannot be strictly seperated from one another. The interest in philosopy of technology has increased over the years. The so-called Delft school of the philosophy of technology has been quite influential here in its study of technology from a more analytical perspective. See for instance: https://plato.stanford.edu/entries/technology/ Note that there has been s similar development in history of science and history of technology. The latter was initially treated rather depreciatory by the community of historians of science. On an international level it is telling that the Division for the History of Science added "Technology" to its name only in 2013 to become the DHST.

25 Avigad, J. (2008), *Computers in mathematical inquiry*, in: P. Mancosu (ed.) "Philosophy and the many faces of science", Oxford University Press, pp. 302–316.

26 L. De Mol, The proof is in the process. A preamble for a philosophy of computer-assisted mathematics. Galavotti, M.C., Dieks, D., Gonzalez, W.J., Hartmann, S., Uebel, T. and Weber, M., New Directions in the Philosophy of Science, Springer, 2014.

27 A well-known phrase in this setting, due to Jean-Paul sartre, is "existence precedes essence". For me it comes down to the idea that truth is always relative to ones own historical and cultural context.

28 Dijkstra, E.W. (1989) *On the Cruelty of Really Teaching Computing Science*. "Communications of the ACM", 32, 1398-1404.

29 L. De Mol and M.-J. Durand-Richard, Calculating Machines and numerical tables – a reciprocal history, draft version available here: https://hal.univ-lille.fr/hal-01396846. Note that this chapter will be published in a volume edited by D. Tournès but that it was written in the period 2011-2013.

30 I think here for instance of the work by historians like John Agar (John Agar (2003), *The government machine. A revolutionar history of the computer*, MIT Press) and James Beniger (ames R. Beniger (1986), *The control revolution. Technological and Economic origins of the information society*, MIT Press) which contextualize the rise of the modern computer in a longer and broader socio-historical development that relates to the reorganization and systematization of information processing. Another important source for me was Michael S. Mahoney (2005) *The histories of computing(s)*, Interdisciplinary Science Reviews, 30:2, 119-135. In that paper Mahoney considers two different models of viewing ENIAC. The first is the machine-centred view assuming a more singlar view on ENIAC, the other is a model from the perspective of communities of computing in which ENIAC becomes but one

of many parallel developments.

31 Matt Welsh, *The end of programming*, Communications of the ACM, vol. 66, nr. 1, pp. 34-35, 2023.

32 See for instance: https://www.theguardian.com/commentisfree/2020/sep/08/robot-wrote-this-article-gpt-3

33 To give just one clear example: the people behind the series have made significant and succesful efforts to have a good gender balance at the conference by: aiming for a fair number of female keynotes and a fair number of female PC members. When for instance during PC discussions an imbaance seems probable, people have no reluctance to interfere and make sure the baalnce is kept. Since 2007, they also have an annual Women in Computability workshop, providing a space for sernior women to speak about and share their own experiences; there is a mentoring program and a grant programs.

34 Indeed, philosophy in particular is reputed not to create such friendly atmospheres and was, until a couple of years ago, known for its sexism with, amongst others, plenty of conferences without any female keynotes. Luckily, that situation has steadily changed.

35 This is an implicit reference to a paper by the etnomethodologist Michael Flynch (Lynch, Michael, *Ethnométhodologie et pratique scientifique: la pertinence du détail*, Cahiers de recherche sociologique 5, pp. 45–62.)

36 Author's translation from: Friedrich Kittler.1998. *Hardware - Das Unbekannte Wesen*. in: Krämer, Sybille(ed.), "Medien, Computer, Realität: Wirklichkeitsvorstellungen Und Neue Medien." Suhrkamp, Frankfurt Am Main 1998. Original German quote: *"Wenn Computer die ersten Maschinen sind, die die Kontingenz oder Unberechenbarkeit einiger, nicht aller Zukünfte um endliche Grade vermindern können, sollte ihre eigene Kontingenz so offen wie möglich bleiben. Die Entwicklung möglichst komplexer und möglichst vielfältiger Hardwareplattformen ist daher eine Aufgabe, an der zum Beispiel Europa, von der bewundernswerten Ausnahme des Transputers abgesehen, bislang noch kaum teilgenommen hat. Wenn mein Beitrag zur Ringvorlesung es vermocht hätte, diese Aufgabe wenigstens wahrnehmbar zu machen, wäre ein Teilziel schon erreicht. Wenn jemand hinginge und all jene Programme, die bislang unterm Titel Philosophiestudium liefen, in Hardware gießen würde, das Ziel selber."*

37 Thé Church-Turng thesis in fact does not really exist as has been argued elsewhere by Edgar Daylight (Daylight, E.G., Schüttpelz, E. (2022). "The Turing Machine as a Boundary Object: Sorting out American Science and European Engineering." Filmed July 2022 at WAI22: Wittgenstein and AI, London. Video, https://www.dijkstrascry.com/Daylight_Schuettpelz_BoundaryObject) and, to a certain extent, myself. Rather, what we have, are a variety of recast versions that fit to particular discourses. One well-known version of it is that anything that can be computed can also be computed by a Turing machine.

38 It is well-known that ENIAC had two lives: in its first life it was a highly parallel and modular machine that was hard to program (it required, essentially, a recabling of the entire machine for every new problem to be setup); in its second and much longer life, it was, essentially, rewired into a serial machine that could be "coded" by using the machine's order code. A main historical source for understanding these transitions is: T. Haigh, M. Priestley and C. Rope, ENIAC in Action Making and Remaking the Modern Computer, MIT Press, 2018. They see the rewired ENIAC as the first computer within the so-called modern code paradigm.

39 In fact, Curry provided a partial compiler on paper. However, he soon left computing again to get back to working in logic and so his work was never picked up and, by consequence, had no influence on further developments.

40 See for instance: L. De Mol and M. Bullynck, Roots of program revisited, Communitations of the ACM, vol. 64, nr. 4, 2021, pp. 35-37. In that short paper we showed that the notion of "program" as used in the ENIAC discourse, if anchored in a decades old engineering discourse around so-called programme clocks and program devices. The original conception of programs in ENIAC was thus completely disconnected from any logical discourse.

41 See for instance: Michael Mahoney, The histories of computing(s), Interdisciplinary science reviews, vol. 30, nr. 2, pp. 119-135.

42 As she writes there: "The fact that historians feel aware in some respect of how they are illiterate, when they tackle a new set of documents, is the best indicator for the existence of the specific skills that the actors possessed. In my view, without describing the skills and attempting to acquire them, historians of science cannot adequately deal with their sources" (Karine Chemla, *Literacy and the history of science*. *Reflections Based on Chinese and Other Sources*, in: David R. Olson and Nancy Torrance, The Cambridge Handbook of Literacy, pp. 253-270, 2009)

43 Thomas Haigh, The tears of Donald Knuth, Communications of the ACM, vol. 58, nr. 1, pp. 40-44.

44 In Ensmenger's book object-orientation is presented as a means to reorganizing software labor. While this is of course true to a certain extent, there were others who relied on it as a means to develop a software system that could be personalized (think of the motivation to use objects instead of functions in the Smalltalk system).

45 I elaborated on this point during a talk at the SIGCIS workshop at SHOT in 2019 and which is available here: https://hal.science/hal-03287418v1/file/talk_SHOT_LDM_final.pdf

46 P. Erickson, J. Klein, L. Daston, R. Lemov, Th. Sturm, and M. Gordin, eds., How Reason Almost Lost its Mind: The Strange Career of Cold War Rationality, Chicago: University of Chicago Press, 2013.

47 As they write in the introduction of the work: "an ideal of cold war rationality can be described as follows: First of all, this rationality should be formal, and therefore largely independent of personality or context. It frequently took the form of algorithms - rigid rules that determine unique solutions - which were moreover supposed to provide optimal solutions to given problems, or delineate the most efficient means toward certain given goals (taken, in this instance, for granted). Second, complex tasks and episodes were analyzed into simple, sequential steps; the peculiarities of context, whether historical or cultural, gave way to across-the-board generalizations; analysis took

precedence over synthesis. And finally, at least ideally, advocates hoped that the rules could be applied mechanically: computers might reason better than human minds. This ideal type shows the marks of its historical origins, as we will see in the ensuing chapters: on the one hand, in the mathematics of algorithms, linear programming, and game theory; on the other, in the theory and practice of economic rationalization." (Idem, p. 4)

48 I am very much indebted here to Ksenia Tatarchenko who brought up the issues around that use of algorithm in that work when I was collaborating with her and Anya Yermakova.

49 See: K. Tatarchenko, A. Yermakova and L. De Mol (2021), *Russian Logics and the Culture of Impossible: Part II—Reinterpreting Algorithmic Rationality*, IEEE Annals for the history of computing, vol. 43, nr. 3, pp. 57-69. and K. Tatarchenko, A. Yermakova and L. De Mol (2021), *Russian Logics and the Culture of Impossible: Part I—Recovering Intelligentsia Logics*, IEEE Annals for the history of computing, vol. 43, nr. 3, pp. 43-56. It should be pointed out here also that my contributons to these two papers were modest compared to the contributions from Ksenia Tatarchenko and Anya Yermakova.

50 One well-known case here are the Unguru debates in the history of mathematics. In 1975 Sabbetai Unguru published a paper in the *Archive for the history of exact sciences* where he argued that Greek (or Babylonian) mathematics is not algebra, and should thus not be put in modern mathematical terms, but has to be considered as a discourse in its own. Unguru's paper resulted into a controversy during which a number of famous mathematicians would send angry letters to the editors of the Archive. It was symptomatic of the growing divide between mathematicians and historians vis à vis history of mathematics, which eventually led to a greater intellectual independence of the historians from mathematicians and to the slow recognition of the history of mathematics as a specialisation in its own right. For a discussion of these debates and its relevance for the history of the history of computing, see: L. De Mol and M. Bullynck, *Making the History of Computing. The History of Computing in the History of Technology and the History of Mathematics*, Revue de Synthèse, Volume 139: Issue 3-4, 2018, pp. 361–380.

51 K. Gavroglu, The Sisyphean Fate of History of Science, Centaurus, vol. 64, nr. 2, 2022, pp. 809-828.The paper is based on Gavroglu's Neuenschwander prize lecture presented at the 2022 ESHS conference held in Brussels and with the same title.

52 For a short history of the history of computing see for instance: Thomas Hiagh, *The History of Computing: An Introduction for the Computer Scientist*, in : Akera, A. and Aspray, W., Using History To Teach Computer Science and Related Disciplines, Computing Research Association, 2004, pp. 5–26.

53 Michael D. Gordin, Book Review: The Tory Interpretation of History, Historical Studies in the Natural Sciences, Vol. 44, No. 4, 2014, p. 421-422.

54 Ibidem, p. 320.

55 L. De Mol, M. Bullynck and E. Daylight 2018, idem.

56 This is one of the chapters of the fortcoming book written by the PROGRAMme collective and discussed in more detail in Sec. 6.

57 For an argument, see Hasok Chang's reply to Gordin's review and in particular section 2.2.: Hasok Chang, *Presentist history for pluralist science*, Journal for general philosophy of science, vol. 52, 2021, pp. 97-114.

58 One example discussed by Gavrogli concerns the European Horizons programwhere the social sciences and humanities used to have their own niche. But since Horizon 2020 this is no longer the case – instead they have been "fully integrated" into the program "to maximise the returns to society from investment in science and technology" (Ibidem, p. 818).

59 Ibidem, p. 824.

60 There is an implicit reference here to Latour's critical reflections on critical theory of which he is supposed to be one of the founders: B. Latour, *Why Has Critique Run out of Steam? From Matters of Fact to Matters of Concern*, Critical Inquiry, vo. 30, 2004, pp. 225-248. In that text he uses the military metaphor of the officor having to consider revising their military strategies when faced with a new situation: "I simply want to do what every good military officer, at regular periods, would do: retest the linkages between the new threats he or she has to face and the equipment and training he or she should have in order to meet them—and, if necessary, to revise from scratch the whole paraphernalia. This does not mean for us any more than it does for the officer that we were wrong, but simply that history changes quickly and that there is no greater intellectual crime than to address with the equipment of an older period the challenges of the present one. " (Idem p. 231)

61 Thomas Haigh and Paul Ceruzzi, *A new history of modern computing*, MIT Press, 2021

62 Thomas Haigh, *Finding a Story for the History of Computing*, Collaborative Research Center 1187 Media of Cooperation, Working paper series, nr. 3, July 2018.

63 Michael Mahoney, *The History of Computing in the History of Technology*, Annals of the History of Computing 10, 1988, 113-125

64 For a detailed account see: Matti Tedre, *The science of computing: Shaping a discipline*, CRC Press, 2014. Peter Denning is known to have frequently launched debates on the nature of the field. Most well-known is probably the 1989 report Comer, D. et al, Computing as a Discipline, Communications of the ACM, vol. 32, nr. 1, 1989, pp. 9-23. In that report a multi-disciplinary view is proposed.

65 Knuth, Donald E. (1984). *Literate Programming*, The Computer Journal. British Computer Society. **27** (2): 97–111.

66 Warren Sack, *The software arts*, MIT Press, 2019

67 In the recent book by Tom Haigh and Paul Cerruzi (Thomas Haigh and Paul Ceruzzi, *A new history of modern computing*, MIT Press, 2021) the notion of ubiquity is replaced by that of universality but the two seem quite related if not identical. The technical notion of ubiquity is of course a rather common one in the computing field itself (see, for instance the ACM UbiComp conference or the ACM Jounral Ubiquity). The more technical notion of ubiquitous computing was launched in the late 1980s by the computer scientist Mark Weiser working at Xerox Parks where ubiquitousness required that the technology itself should disappear from sight hidden behind interfaces that are "human-centered", familiar, intuitive. Interestingly, he was inspired by Heidegger's philosophy too.

68 In the call for papers we wrote: "The number of researchers working in fields related to computing is growing rapidly in many different directions. As Mahoney once stated, "the computer is not one thing but many different things, and the same holds true of computing". As a consequence, the computing sciences collect the most diverse complex of experts: philosophers, logicians, historians, mathematicians, computer scientists, programmers, engineers. The number of involved subjects grows accordingly: from the foundational issues to their applications; from the philosophical questions to problems of realizability and design of specifications; from the theoretical studies of computational barriers to the relevance of machines for educational purposes. Given the significance of computing for modern society, the relevance of its history and philosophy can hardly be overestimated. The aim of this conference is to bring together these two streams: we are strongly convinced that an interplay between researchers with an interest in the history and philosophy of computing can crucially add to the maturity of the field."

69 DHST stands for Division for the History of Science and Technology; DLMPST stands for Diviison for Logic, Methodology and Philosophy of Science and Technology. Each of these divisions has its main conference every four years. While sister divisions, their inner eays of working are not entirely the same. For instance, the DHST has a long-standing tradition of commissions which are specialized in particuar themes such as history of physics or history of mathematics, whereas DLMPST has fewer commissions.

70 Amongst other: Janet Abbate, Bill Aspray, Nathan Ensmenger, Con Diaz, David Gugerli and Tom Haigh.

71 Other examples are: programs as formal objects fits well with a more logical approach one finds in analytical philosophy but also with work on formal verification; programs as socio-technical objects, fits well with the approach of Science and Technology Studies but also with particular strands of software engineering or ethical analyses of computer programs; programs as physical objects fits well with certain approaches of media studies but also with the views of hardware engineers; etc.