Middlesex University London

Middlesex University Research Repository

An open access repository of

Middlesex University research

http://eprints.mdx.ac.uk

De Mol, Liesbeth and Primiero, Giuseppe (2014) Facing computing as technique: towards a history and philosophy of computing. Philosophy & Technology, 27 (3) . pp. 321-326. ISSN 2210-5433 [Article] (doi:10.1007/s13347-014-0169-4)

Final accepted version (with author's formatting)

This version is available at: https://eprints.mdx.ac.uk/15152/

Copyright:

Middlesex University Research Repository makes the University's research available electronically.

Copyright and moral rights to this work are retained by the author and/or other copyright owners unless otherwise stated. The work is supplied on the understanding that any use for commercial gain is strictly forbidden. A copy may be downloaded for personal, non-commercial, research or study without prior permission and without charge.

Works, including theses and research projects, may not be reproduced in any format or medium, or extensive quotations taken from them, or their content changed in any way, without first obtaining permission in writing from the copyright holder(s). They may not be sold or exploited commercially in any format or medium without the prior written permission of the copyright holder(s).

Full bibliographic details must be given when referring to, or quoting from full items including the author's name, the title of the work, publication details where relevant (place, publisher, date), pagination, and for theses or dissertations the awarding institution, the degree type awarded, and the date of the award.

If you believe that any material held in the repository infringes copyright law, please contact the Repository Team at Middlesex University via the following email address:

eprints@mdx.ac.uk

The item will be removed from the repository while any claim is being investigated.

See also repository copyright: re-use policy: http://eprints.mdx.ac.uk/policies.html#copy

Facing Computing as Technique: towards a History and Philosophy of Computing

An essential role of philosophy has always been to investigate, reflect upon and contextualize scientific research, its paradigms, and their results. Philosophy has acted as a methodological criterion for the sciences, especially in the form of logical reasoning, since its Aristotelian formulation. Later, it has played a foundational role in guiding mathematical research, during and beyond the *Grundlagenkrisis* of the XXth century. This process is well-known to us today in modern epistemology, with the various incarnations of philosophy as reflection about physics, biology, engineering, to name the most common ones. Each scientific field presents to philosophy its own range of challenges and offers ways forward in philosophical thinking itself. Famous examples are the "paradoxes" of quantum physics and the large philosophical reflection on causality, inspired by the natural sciences. Along with the methodological and foundational role played by philosophy, contextualizing and historicizing scientific research is an essential task for its understanding. Indeed, as has been pointed out by Lakatos, paraphrasing Kant, "*Philosophy of science without history of science is empty*".[1] Failing to acknowledge the historical development of science(s) and ignoring the rich and fluctuating content of scientific progress leads to a production of abstract classification criteria, bearing little or no relation at all to how science develops in reality. Conversely, as Lakatos continues, "*history of science without philosophy of science is blind*". Merely studying the history of science as history will only tell us about the facts of the history, but will not put us in the position of understanding and anticipating future scenarios. In such combined historical and philosophical landscape of the study of the sciences, computation and its incarnation, the computer, have represented a real game-changer. The concurrent development of conceptual and diachronic analysis towards the sciences is much needed in the context of computing, with its short but complex history and its fundamental philosophical questions, which are far from being solved and depend on the evolution of what computation is now, and can be later at the theoretical and technological level.

This combination of historical and philosophical research forms the methodological basis for the activities of the DHST[2] commission for the History and Philosophy of Computing (HaPoC), an organization which is responsible, amongst others, for overseeing the organization of the HaPoC conference series[3]. This special issue collects five papers that resulted from the first of these conferences, held from 7-10 November 2011 at Ghent University, Belgium. The aims of this conference series and the HaPoC commission in general are rooted in the certainty that we are not only in need of deep historical and philosophical reflection on computing, but that any such reflection needs to be an open discussion among computer scientists, mathematicians, engineers, logicians and any other community involved in computing.

1 Lakatos, I. (1970), History of Science and its rational reconstructions, Proceedings of the Biennial Meeting of the Philosophy of Science Association, pp. 91-136.

2 DHST is the Division for the History of Science and Technology of the International Union for the history and philosophy of science.

3 See www.hapoc.org for more details

Computing has revolutionized scientific research itself. The way to construct hypotheses, create models, test them and formulate theories has been re-defined entirely in view of computational systems and methods, with our capabilities being pushed more and more forward. The recent impressive simulation of 20-years evolution of the Universe is a very clear example of such progress.[4] But the evolution cannot be put simply in terms of brute computational force: computing is in many ways suggesting a ground, albeit not yet fully understood, methodological shift. As an example, consider the current trend in data analytics: there is no formal property of scientific models that cannot be revealed by sufficient data classification, accompanied by a decent set of association rules, able to predict transformation and induce relevant clustering[5]. A theoretical change of such proportions is possibly already inducing a new revolution of the Kuhnian kind, one that will need sufficient philosophical reflection.

Moreover, the computer has not remained confined to the laboratory. Rather, it has become an omnipresent object in our society and, along with it, computation has become a dominant technique: the variety of applications, and how we depend on them, are multiple. From one of its first applications onwards -- making computations to develop the H-bomb -- it was clear that the effects of computing machinery were certainly not restricted to scientific research, resulting in a technology that affects every man in the street. For this reason, within a relatively short period of time computing has become an area of philosophical interest, not only for its relevance to science, but also from the human perspective. Computing as ubiquitous technique has become a full-fledged philosophical topic. As such, computing represents the last formulation of a long-standing tradition of analyses on the notion of techné/technique/technology. One important source in this context is Martin Heidegger's analysis of "technique", expressing the way we act in and perceive of the world.[6] This work is often interpreted as a critique on the "technification" of modern society, but it can also be understood as an appeal to man to become aware of the way perception and action in and with the world are possible. In this interpretation of "technique", computation becomes the dominant technique in our current society and computing devices represent its physical realization. In the context of the computational society, therefore, the above reflection suggests that, in order to understand "technique", it is essential that we do not reduce ourselves to mere "users" of a computational "instrument", but that we actively attempt at understanding what we are using, how we are using it and how it functions. This is the only route to become aware of the

4 See: M. Vogelsberger et al (2014), *Properties of galaxies reproduced by hydrodynamic simulation*, Nature, vol. 509, pp. 177-182.

5 This strong assumption drives the research areas of data mining and big data analytics, although not necessarily taken with its full methodological implications: "[D]ata mining is the process of discovering interesting knowledge from large amounts of data stored in databases, data warehouses, or other information repositories. [...] By performing data mining, interesting knowledge, regularities, or high-level information can be extracted from databases and viewed or browsed from different angles. […] . [D]ata mining is considered one of the most important frontiers in database and information systems and one of the most promising interdisciplinary developments in the information technology." [Han, J. and Kamber, M., *Data Mining – Concepts and Techniques,* Elsevier, 2006.]

6 M. Heidegger (1954), Die Frage nach der Technik, in: M. Heidegger, Vorträge und Aufsätze, Verlag Günther Neske.

epistemological window through which we perceive, interact and construct our world and hence stop it from being an epistemological obstacle.

From this perspective, the complex of relations between humans and computations remains a crucial cultural issue in need of reflection. Such reflection becomes even more critical in the face of the multi-layered structure of computing:

- as a discipline that develops and investigates computing devices, computing is the study of technological artefacts and the ``phenomena'' that surround them;
- as a study of algorithms, it is a mathematical discipline;
- and as a study of procedures, it is a natural science that defines its own subject matter (as opposed to other natural sciences whose subject is external and given).

This variety of aspects, to be taken inclusively, qualifies the above issue of “technique” as an epistemological obstacle in a unique way: computing imposes not just knowledge of its use, but also knowledge of this diversity of forms. While use is certainly a step towards knowledge (as the essential divide between digital natives and digital analphabets illustrates), manipulation and creation constitute an entirely different level of interaction with technology and requires another kind of understanding. The fact that the majority of mankind *uses* many of the possibilities of the computer, without ever knowing what is beyond the interface they are offered and that has developed historically in a certain monolithic development system, is just one concrete example of this problem. Indeed, the only reason for this type of monopoly being possible is the fact that most of the people never get beyond their graphic user interface. This form of "unaware user" is obviously also the result of a historical process, rooted, amongst tohers, in the leap from low-level machine codes to symbolic assembly language to high-level languages and structured and object-oriented programming. The problem is certainly not *per se* this development of multiple, increasingly more productive, levels of abstraction. The problem is, rather, that as soon as these levels induce forgetfulness about what lies beyond a certain level of abstraction, one gives away the control, not to the “instrument”, but to the bureaucracy that actively promotes this forgetfulness.

This brief analysis has essentially two core points. First, there is a sense in which computing inherits today the essential role that *techne* and technique have played in previous eras, and with it comes a bundle of fundamental questions concerning its relation to humans and the need for acknowledging and understanding it as a way to access and act in the world. Secondly, for its own nature, computing and the computer are multi-layered and profoundly multi-structured, and the result of a combination of hardware, language, abstraction, modeling, interaction etc: this essential feature of computing is made even more relevant by the drive towards more efficient and faster ways of defining and using computational process in order to satisfy our needs. It has also generated the forgetfulness we have hinted at before. It is exactly the combination of these two points, computing as a hidden, multi-faceted technique, that forms the main drive behind HaPoC: we are committed to opening up this technique through historical *and* philosophical reflection. How can we, for instance, expect to unravel the complex relation between humans and computation without paying attention to its rich development in history? Conversely, how can we expect to understand the evolution of, for instance, the method of levels of abstraction, without taking into account the very idea of abstraction in computing as a practice and as a philosophical concept?

However, in order to offer such critical reflection on computing, our analysis also shows the need for an understanding of its "technical" aspects. While requiring that everyone becomes a kind of omniscient computer scientist is a fairly unrealistic and possibly unnecessary approach to the problem, the role of historical and philosophical reflection cannot avoid to put this issue at the front-line of its critical reflection about computing: this requires not only acute awareness of the sociological, political, anthropological problems surrounding computing. It also imposes to the philosopher and the historian a willingness and ability to master the technical details that are at the root of this technique, be they mathematical abstraction, logical structure or engineering patterns. We are strongly convinced that tackling the multiple fundamental and philosophical problems arising within computing can only succeed if one also gives space to the developing techniques that underpin the concept. From this perspective, answering the question “What is computation?” becomes impossible without being open to the technological and formal practices that have historically shaped and are continuously redefining the technique. We consider the promotion of a combined historical and philosophical reflection on computational techniques as a true challenge for the humanities to engage with the philosophical issues and historical evolution of computing in its multiple contexts, from the foundations of computing to the engineering practices. It is for this reason that, from the first conference onwards, HaPoC has been conceived as a truly interdisciplinary conference, viz. not only a meeting by and for historians and philosophers, but also as a venue for mathematicians, logicians, programmers, artists, sociologists etc. Our aim is to create the open platform needed to accommodate a fundamental reflection on computing that includes all of its facets and where each discipline contributes to the shaping of the research questions leading other disciplines. We are very happy to see that each of the papers of this volume reflects precisely this attitude of conceptual and technical precision, along with an openness towards the intersection of the distinct research methods and problems that become intertwined in the computational sciences.

Federico Gobbo and Marco Benini start off this Special Issue with a contribution that combines historical and philosophical analysis of technically-laden issues in computing. They analyze the changing relations between the computer and its “users” throughout its short history, showing how the evolution of modern computers has been shaped through abstraction. More specifically, they rely on the methodology of levels of abstraction, which the Philosophy of Information has inherited from Computer Science, to analyse the formal and structural relation between operators, programmer and user by way of a categorical model (in the sense of category theory). The result is that the information flowing along categories is interpreted as a structure-preserving entity and information hiding can be considered the main epistemic feature of such structures. The model focuses on historical milestones of multi-tasking architectures, presenting abstractions on a single process, the OS, the von Neumann machine model, co-operative multi-tasking and network-distributed applications.

The philosophical openess of the HaPoC spirit is at play in Sybille Kraemer's contribution. She challenges the idea that computation is a non-sensual, purely formal kind of operation, and connects it back to a material, linear spatialization of visual terms. She argues that a technique of spatialization has been crucial in representing and operating with non-spatial, invisible objects of knowledge, hence formalism and perceptivity do not oppose each other but are intertwined. This philosophical take on calculation is contextualized in the work of Descartes, claimed to be among the first to use such spatialized formalism as a figurative and operative writing of the mind. In this way, the crucial problem of abstraction is also echoed in this paper by questioning the relation between the abstract and the concrete, through a study of work that goes back to a time when there were no computers in the modern sense of the word.

Raymond Turner offers another perspective on the Philosophy of Computer Science, this time with respect to the formal nature of programming languages. This analysis starts in the formal nature of grammars, and then it follows the hybrid nature of its object, to explore their abstract nature in the semantics, and their concrete one in the implementation by compilers. This is a crucial and recurring theme in the philosophy of computing at large, the nature of which is partly formal science, partly engineering, partly language. In his task, which takes into account idealization, correctness and intention as modes of the ontology appropriate to programming languages, Turner resorts to the comparison with the notion of technical artefact, which finds its origins in the Philosophy of Technology, and has already highlighted some feature that are reproducible in this new endeavour: design, reproducibility, modifiability.

The multi-faceted nature of computing, already approached in the previous contribution, returns in a new version in the contribution by Viola Schiaffonati and Mario Verdicchio: the nature of computing from a methodological perspective, in particular related to computational experiments, their role and structure. The authors consider the standard experimental method from traditional scientific disciplines, and reveal how the strictly innovative and rapid development of computing has, on the one hand, hindered the applicability of a similar structured way of accounting for scientific hypotheses; and on the other, it makes it necessary to improve on the accountability of computational results. They conclude with a plea for a more historically aware approach in the development of such experimental strategies and for a more intense collaboration of philosophers and computer scientists, in the line with the recent development that the Philosophy of Technology has taken in its combination with Engineering.

One fundamental problem in computer science is the question: what does it mean to compute? Hector Zenil offers a very interesting answer to this question in the form of a behavioural approach in terms of programmability with environment-reacting as its main property. This approach has its roots in the definition of complexity as we find it in algorithmic information theory and is applied to a wide variety of "devices". This wide applicability is due to the fact that the proposed definition is independent of a particular model. As such, it can be applied both to natural as well as to artificial systems and can thus be used as a broader epistemological framework of computation which includes physical systems.

We believe HaPoC has created a platform of discussion and research relevant to all the different areas of computing. We hope this Special Issue will represent a means for many to join our community and contribute to develop a new level of understanding of the computing sciences.

Liesbeth de Mol
CNRS – UMR 8163
Universite' de Lille 3

Giuseppe Primiero
Computer Science Department
Middlesex University London