

Luca M. Possati, *Software as Hermeneutics: A Philosophical and Historical study*, London: Palgrave-Macmillan, 2022, 193 pp., £87.50, ISBN: 9783030636104

Evangelos Kalfopoulos (University of Sofia)

Luca M. Possati's book *Software as Hermeneutics: A Philosophical and Historical Study* attempts to shed light on one of the most widespread artifacts humans have ever invented. In fact, the ubiquity of software requires not light as such; it is the kind of light that matters. Possati's book is both philosophical and historical but, even more interestingly, it is also post-phenomenological and critical; or, put differently, hermeneutical. The study takes its place as a work of philosophy of technology, though it is not technology related to machines; it has to do with artifacts and especially those found everywhere, which usually hide their very nature: writing and software as a kind of writing. Indeed, one of Possati's main theses is that software is a text.¹³ But before focusing on the main arguments, let us try to grasp the book's angle on the philosophy of technology.

Possati, a researcher and lecturer at the University of Porto in Portugal, maintains that the most fruitful approach to software is post-phenomenological, an approach that combines aspects of Husserl's thought with aspects of American pragmatism and offers "a particular mode of science-technology interpretation,"¹⁴ one that takes science and technology to be fundamental mediations between the human being and the world. Post-phenomenology makes the techno-scientific mediation between humans and their world indispensable in two senses: first, there is no direct subject-object (human-world) relation, but rather an always mediated human-technology-world relation; and, secondly, this techno-

¹³ Luca M. Possati, *Software as Hermeneutics: A Philosophical and Historical Study* (London: Palgrave-Macmillan, 2022), 7.

¹⁴ *Ibid.*, 67.

scientific mediation actually constitutes both the subject and the object. As Possati puts it, “intentionality is not a bridge between subject and object, but rather a fountain from which both of them emerge.”¹⁵ To account for the dual nature of techno-science as both constraining and opening up possibilities, post-phenomenologists introduce the notion of “multi-stability,” which states that technologies gain new interpretations in different contexts, a notion that Ihde, the founder of post-phenomenology, credits to Heidegger’s initial conception that “an instrument is what it does, and this in a context of assignments.”¹⁶ However, in general, post-phenomenology is far from accepting the general techno-pessimism of Heidegger, for whom the new world of technology poses grave and insurmountable dangers for the authenticity of human living; instead, it opens up the possibility of a relational ontology mediated by an indispensable technology. One could say that post-phenomenology’s acceptance of the inevitability of technology leaves behind the techno-optimism/techno-pessimism dichotomy for a more honest approach to our relationship with technology. It is in this spirit that Possati approaches the phenomenon of software in the present book.

Although present everywhere, software is still not generally well defined. Possati claims that software is a complex thing that cannot be casually defined simply as the immaterial part of the computer – with hardware as the material side. The basic reason is that software also has a material side, since it uses written algorithms. But the nature of software is not confined to an algorithm. Actually, Possati claims that software is “a specific entity defined by (a) a problem, (b) an algorithm (logic + control) and (c) a code (a formal language).”¹⁷ At an ontological level, these characteristics are fundamental for software and exhibit a complex unity. But there are other aspects that Possati also captures, built upon the fundamental characteristics that make software unique in relation to other technologies in its phenomenological break between lived digital experience and the code that enables it. Software is not just a tool but rather an entire culture;¹⁸ it is a living organism that lives be-

¹⁵ Ibid., 68.

¹⁶ Ibid.

¹⁷ Ibid., 48.

¹⁸ Ibid., 43.

yond its implementation and has an artistic dimension.¹⁹ All these aspects are captured best by the book's main thesis that software is basically a text.

Possati follows Paul Ricoeur's definition of the text as written discourse in which the forces of distancing and appropriation operate dialectically. Narrative is another important concept for Ricoeur, one that mediates between human reality and the text. Possati distinguishes three basic steps in Ricoeur's textual hermeneutics: the pre-understanding of the world of action, the emplotment or configuration of the text, and lastly the effects of the text on human experience and the human world.²⁰ Possati further summarizes these three phases as prefiguration/configuration/reconfiguration²¹ which form a hermeneutic circle, even a spiral, that "would carry the mediation past the same point a number of times, but at different altitudes,"²² as Ricoeur says. Possati claims that software, which mediates between concrete problem solving in the human world (prefiguration) and the action of the machine that instantiates the software (reconfiguration), exhibits Ricoeurian distancing. That has the effect of placing software in the realm of pure writing,²³ as it occurs in different layers,²⁴ uniting them towards greater abstraction. Software is a procedure of writing and re-writing that corresponds to the level of configuration in Ricoeur's scheme, with the difference being that its configuration takes place in both the user and the machine (and not just in the reader). But this difference is an extension of Ricoeurian hermeneutics.²⁵ In configuration, reflective judgment in the Kantian sense, about which more will be said below, plays a central role for both the classic Ricoeurian text and software. We are thus led to a reconfiguration that happens when the machine, acting upon software instructions, translated to the appropriate machine language narrative, finally acts on the human world, creating new meaning. We arrive at a new hermeneutic circle – analogous to the Ricoeurian one – summarized

¹⁹ *Ibid.*, 44.

²⁰ *Ibid.*, 71.

²¹ *Ibid.*; see figure 3.1.

²² *Ibid.*

²³ *Ibid.*, 76.

²⁴ *Ibid.*; see table 3.2.

²⁵ *Ibid.*, 77.

in figure 3.2.²⁶ The software reconfiguration is more pervasive in terms of its effects on the human users of technology than other technological effects categorized and studied in post-phenomenological research.

There are two further points that I would like to stress regarding Possati's thought-provoking book. First is his attempt to address the main claim that software is a type of writing, more precisely a type of pure writing.²⁷ In a thorough historical analysis of writing as technology, Possati uses both anthropological and philosophical sources to show that software can be perceived as an extension of the materiality of writing, in particular as the culmination and apogee of the symbolic revolution that took place in the development of mathematical language, primarily in the work of Descartes. Writing is indeed shown to be a material technology that revolutionized human civilization. Socrates offered a critique of it in the Platonic dialogue, *Phaedrus*; a further development with important consequences, when writing became mechanized, was the invention of typography. Finally, through the symbolic writing of mathematics, we come to software which "*must eliminate ... meaning and reduce itself to a pure process of writing and rewriting – a circulation and transformation of electrical signals*. From this point of view, software takes the revolution begun by Descartes to its extreme consequences."²⁸ The second point worth stressing is the notion of a digital reflective judgment introduced in chapter 5. Reflective judgment is a Kantian concept that "presupposes a particular individual or narrower concept and advances from it toward a universal or more general concept."²⁹ Aesthetic and teleological judgments are typical examples of reflective judgment. A common characteristic of reflective judgment is the a priori transcendental principle of finality.³⁰ Possati claims that software acts as a digital reflective judgment because it confronts a particular problem that is then subsumed under a higher-order principle. This is possible through the imagination of the programmer, who provides a solution to a real-world problem through acts of comparison, reflection and abstraction. The graphic materiality of written software plays a cru-

²⁶ Ibid., 79.

²⁷ Ibid., 76.

²⁸ Ibid., 139; emphasis in the original.

²⁹ Ibid., 146.

³⁰ Ibid., 148.

cial role in grounding the need for design, and this also gives rise to a feeling of delight, which points to the principle of finality.³¹

In the final chapter, Possati situates his research in the broader context of the newly established field of critical code studies. For Possati, critical code thinking aims to analyze biases and to use the narrative turn in post-phenomenological studies to understand the narrative that software tells us about society and the times we live in. The narrative turn tries to reconcile post-phenomenology with Heidegger's thought, especially by incorporating the element of time, but also through analyzing social elements and adopting a more critical stance towards the conception of subjectivity used by post-phenomenology. Possati claims that "software has put an end to one phase of capitalism and opened the doors to an entirely new post-capitalist society based on sharing goods, as demonstrated by the phenomenon of open source. Therefore, software represents the most powerful, but also the most dangerous, tool for capitalism."³²

Possati's book is a demanding, well-grounded study that draws from many and various sources to address one of the most important, ubiquitous and interesting entities of our new digital era; software. It explores new, involving and above all fruitful and promising paths in the philosophy of technology, particularly in the philosophy of software seen from a continental perspective. It is basic reading for anyone interested in the future of continental philosophy in the age of digitalization and in the post-Fordist, capitalist, information-based societies of our present and future.

³¹ Ibid., 170.

³² Ibid., 188.