The role of aesthetics in the understandings of source

Pierre Depaz
under the direction of Alexandre Gefen (Paris-3)
and Nick Montfort (MIT)

ED120 - THALIM

February 2023

1 Introduction

While the functional expressiveness and aesthetics of executed code is undeniable, the artistic expressiveness of a source code text remains elusive. This doctoral thesis therefore inquires into the place of aesthetics, of a sensually pleasing formal manifestation, in the writing and reading of source code. The source code, based on a syntactic system similar to the natural language, reveals itself to be a system of communication as well from human to human as from human to machine.

This raises the question of a text whose formal manifestations are destined to disappear, and whose reading is only a collateral process of its execution. The main research question of this project is thus that of the *role of the aesthetics in the comprehensions of the source code*. This entails uncovering the aesthetic properties specific to source code, and identifying how these properties enable sense-making, and what justifies them.

In addition to an empirical approach of the source codes themselves, it is also a matter of uncovering the ways in which these properties are related to other fields of creation mobilized by the programmers, notably to the fields of literature, architecture, mathematics and engineering. By examining these relationships, this thesis is thus part of a research work on the cognitive dimension of the aesthetic phenomenon within the specific medium of source code.

On the one hand, the literature in computer science and engineering does take into account the existence of an aesthetic of source code, from the point of view of the productivity of those who write code, and from the cognitive point of view of those who read and need to understand code bases (Oram & Wilson, 2007; Cox & Fisher, 2009; Gabriel & Goldman, 2001; Martin, 2008; Detienne, 2012; Weinberg, 1998). On the other hand, research in the humanities deals with the interaction between aesthetics and code, while remaining mostly attached to an abstract and disembodied conception of "code", elaborating an aesthetics of digital without going into the details of the source code texts themselves (Cramer, 2019; Hayles, 2010; Mackenzie, 2006; **?**). There are however a certain number of works of humanities approaching more directly the material question of the code, that it is at the cultural level (Montfort et al., 2014), political (Cox & McLean, 2013) or sociological (Paloque-Bergès, 2009).

2 Methodology

The main approach of this project is empirical. It consists in examining the source codes themselves, as well as the discourses of those who write and read them. For this, I rely on the work of Kintsch and van Dijk and their studies of discourse comprehension strategies, with a particular attention being paid to the metaphorical approach as a comprehension strategy.

This empirical approach to this thesis has brought forth a variety of code-writing and code-reading individuals, which I have grouped into four categories (Hayes, 2017), with nonetheless porous boundaries. Developers in a collaborative economic context, with a functional goal and sustainable in the long term; artists writing code in an expressive and artistic context, such as *code poems*, mainly intended for a human audience. Hackers focus on unique, idiosyncratic and highly contextual technical solutions. Finally, academics are individuals who write code to illustrate computational concepts, representing abstractions rather than concrete uses.

The constitution of this corpus was done mainly by consulting online resources (e.g. GitHub, BitBucket, blogs or forums e.g. StackOverflow, Quora) and includes these primary sources (the code itself), and secondary sources (the commentary by an author, or by a public, justifying the aesthetic aspect of a presented code).

Finally, the analysis is done through a theoretical framework starting from aesthetic philosophy and literature in order to define my use of the term *aesthetic*. From a literary point of view, I rely on Gérard Genette's work and his distinction between fiction and diction, considering the aesthetics of the code as its diction, while the poetics would be its fiction. Between literature, philosophy and cognitive sciences are the works of Paul Ricoeur (Ricoeur, 2003) and George Lakoff (Lakoff, 1980), linking verbal composition and evocation of mental images. Finally, the connection between understanding and surface manifestation of a work is taken up by the art philosopher Nelson Goodman in his analysis of the languages of art (Goodman, 1976), and notably in his exploration between syntactic systems and communication, as well as his scientific approach of the artistic phenomena.

Ricoeur and Lakoff's work on metaphor also makes it possible to productively include adjacent areas mobilized by programmers to justify their aesthetic judgments. It is by studying the contact zones of these domains that it is possible to identify the specific aspects of the source code.

3 Development

There is not ”one” source code, abstract and disembodied, but a multiplicity of source codes, existing within different practices and groups. These source codes all possess a possibility of aesthetic manifestation, manifestations subjected to the manifestation and the transfer of knowledge. Since source code possesses both a prescriptive aspect (what the code *should* do) and an effective aspect (what the code *does*), the aesthetics of the source code are spread along this axis, with poets and academics focusing on the prescriptive aspect, evoking concepts through the machine, and hackers focusing on the effective aspect, evoking concepts from the machine, while deveopers focus on representing the problem-domain and its computational processing.

To communicate the intention of the program through its textual implementation and in relation to its field of activity, programmers emply different techniques. It is possible to develop some representation of the problem through various levels of linguistic metaphors (procedural rhetoric (Bogost, 2008), double-coding (Cox & McLean, 2013), double-significance (Paloque-Bergès, 2009)) and a precise choice of vocabulary. These techniques are defined by the fact that source code is a double kind of language: machine language and human language whose syntactic tensions can create a semantic richness.

Then, these linguistic mechanisms are complemented by choices of structures and syntax calling upon fields of architecture and engineering (Gabriel, 1998; Schummer et al., 2009), which in turn make it possible to better apprehend an execution of the source code-an execution whose speed makes it initially incomprehensible to humans. The aesthetics of the source code then allows to represent in a condensed manner the spaces of evolution of the information flows during the execution of a program.

These aesthetic standards are all grouped around the notion of elegance-that is, the use of a minimal syntax for maximum expressiveness. Particularly present in groups of hackers and academics, this is also declined along a machine-concept axis. Hackers tend to use only what is necessary to perform a particular action, well visible in the competitions of *demoscenes* (Kudra, 2020), while academics will testify of the same approach, but to communicate fundamental concepts of computer science, decoupled from the specific machine that runs the program in question.

Ultimately, we have shown that metaphorical representations of code, representations of code as language, as architecture, and as material, are found in the expression of a dynamic semantic space. This spatial conception thus infuses the aesthetic properties of the code itself, in that the syntax and structure of the source code focus primarily on the description and navigation of those semantic spaces, even if the different socio-technical contexts in which these source codes are written will themselves modulate the nature of these spaces.

References

Bogost, I. (2008). The Rhetoric of Video Games. In K. Salen (Ed.) *The Ecology of Games: Connecting Youth, Games and Learning*. Cambridge, MA: The MIT Press.

Cox, A., & Fisher, M. (2009). Programming Style: Influences, Factors, and Elements. In *2009 Second International Conferences on Advances in Computer-Human Interactions*, (pp. 82–89).

Cox, G., & McLean, C. A. (2013). *Speaking Code: Coding as Aesthetic and Political Expression*. MIT Press.

Cramer, F. (2019). *Exe.cut(up)able statements: Poetische Kalküle und Phantasmen des selbstausführenden Texts*. Wilhelm Fink.

Detienne, F. (2012). *Software Design – Cognitive Aspect*. Springer Science & Business Media.

Gabriel, R. P. (1998). *Patterns of Software: Tales from the Software Community*. Oxford University Press.

Gabriel, R. P., & Goldman, R. (2001). *Mob Software: The Erotic Life of Code*.

Goodman, N. (1976). *Languages of Art*. Indianapolis, Ind.: Hackett Publishing Company, Inc., 2nd edition ed.

Hayes, B. (2017). *Cultures of Code*.

Hayles, N. K. (2010). *My Mother Was a Computer: Digital Subjects and Literary Texts*. University of Chicago Press.

Kudra, A. (2020). AoC \textbackslashtextbackslashtextbar Art of Coding – The Demoscene as Intangible World Cultural Heritage. In C. Yackel, R. Bosch, E. Torrence, & K. Fenyvesi (Eds.) *Proceedings of Bridges 2020: Mathematics, Art, Music, Architecture, Education, Culture*, (pp. 479–480). Phoenix, Arizona: Tessellations Publishing.

Lakoff, G. (1980). *Metaphors We Live By*. University of Chicago Press.

Mackenzie, A. (2006). *Cutting Code: Software and Sociality*. Peter Lang.

Martin, R. C. (2008). *Clean Code: A Handbook of Agile Software Craftsmanship*. Pearson Education.

Montfort, N., Baudoin, P., Bell, J., Bogost, I., & Douglass, J. (2014). *10 PRINT CHR$(205.5+RND(1)); : GOTO 10*. The MIT Press, illustrated edition ed.

Oram, A., & Wilson, G. (Eds.) (2007). *Beautiful Code: Leading Programmers Explain How They Think*. Beijing ; Sebastapol, Calif: O'Reilly Media, 1st edition ed.

Paloque-Bergès, C. (2009). *Poétique des codes sur le réseau informatique*. Archives contemporaines.

Ricoeur, P. (2003). *The Rule of Metaphor: The Creation of Meaning in Language*. Psychology Press.

Schummer, J., MacLennan, B., & Taylor, N. (2009). Aesthetic Values in Technology and Engineering Design. In A. Meijers (Ed.) *Philosophy of Technology and Engineering Sciences*, Handbook of the Philosophy of Science, (pp. 1031–1068). Amsterdam: North-Holland.

Weinberg, G. M. (1998). *The Psychology of Computer Programming*. Dorset House Pub.