Pierre Depaz
pierre.depaz@nyu.edu
Paris 3 – Sorbonne Nouvelle

# Aesthetic uses and possibilities of code-writing: from programming languages to poetic speech

## Overview

The most important texts of our era are the most invisible. The hundreds of billions of lines of codes which constitute the source of our technical, and more and more of our social, systems still remain widely excluded from aesthetic and literary analysis. However, programming languages are a unique field of human expression, being the only language that can be executed by a machine. These coexist ambiguously between computational objectivity and individual subjectivity, understood by both, including the exactness of the former, and the ambivalence of the latter. This set of signs and structures which enable to describe precise instructions to be carried out by a machine are still *in fine* changes in electric current. It is precisely this necessity to give a purpose, a meaning to these abstract electrical variations which make programming languages a highly constrained literary means of expression, and it is these constraints which can then inscribe the practice of writing code in the wake of artistic research of the 20th and 21st centuries.

I intend to study the practice of code, both in its reading and writing, as a creative and aesthetic practice. The large amounts of programming languages are specific sites of expression: beyond this intrinsic necessity for efficiency, questions of structure, style, elegance and beauty cannot be separated from discourses around code-writing, and reveal specific interrogations around questions of the beautiful and the useful specific to this craft. Within the emerging field of software studies, this project will examine not acting code, but redacted code, under its static, and not electric, form. Indeed, it has been established that these new modes of writing underpin new poetic practices (e.g. digital art, generative art, net.art, etc.), building on the unique aspects of computation in order to explore new possibilities for artistic expression. These practices range from interactive fiction and generative novels to kinetic poetry and automated conversational agents; all exist within systems of rules that are not without connection to practices of groups such as Dada, OuLiPo or Fluxus. Still, this use of software as aesthetic medium do exist within a certain tradition of contemporary writing, but remain at the level of *result* rather than *process*. Often, its aesthetic value lies in the appreciation of the object, discarding the fact that the very becoming of this object

implies the execution, the disappearance of the code-as-text which constitutes it in the first place, and which therefore has received little academic attention.

This project builds on my trajectory of academic and artistic activities, which began with my thesis work at New York University's Game Center, where I completed an MFA thesis under the supervision of M. Bennett Foddy, and continued throughout my career as a digital arts teacher and practitioner. This work included publishing and communicating within artistic festivals (*Miracle Marathon* at the *Serpentine Gallery*), symposia (*Canadian Literature Symposium*, Toronto) and journals (*NYU ITP Adjacent Possible, First Person Scholar*). I've also published several projects around the dynamics of written code, including *Coroutines* (*OfficialFan.Club*, New York, 2018), a collection of code poems. Both academic and practical, I aim through this work at witnessing the tensions between a so-called "universal" language and the diverse practices and tactics specific to the process of human writing as they manifest themselves through the prism of programming languages. The particular choices of phrasing, denoting and structuring make of code-writing an object of study where lie the personalities and subjectivities of their writers.

This is situated precisely at the moment when digital arts and digital humanities are becoming fully-fledged artistic and academic fields, bringing a more complete understanding of the computational, as a set of possibilities which can be engaged with beyond the sole realm of engineering and mathematics. Several universities, particularly in the United States, Germany and the Netherlands have established a certain number of research projects with the materiality of digital objects as their focus. This materiality, often dismissed at first by the surface effects of user interfaces, is then also set as a synonym for the general denomination of *code*. Such a notion is rarely tied back to its very instantiations, to its concrete existence through the varied programming languages. It is the ensemble of these languages, and the texts which they allow us to write, which constitute my object of study.

I therefore intend to further investigate these research directions, and to consider the materiality of these new systems in how they manifest themselves through programming languages. These corpora exhibit specific features, both in their lexical fields as well as their syntactic and grammatical structures, and could therefore be examined through the lens of literary, aesthetic and semantic theories, such as Kenneth Burke's "*terministic screen*"[1], Ludwig Wittgenstein's "*Sprachspiele*"[2], Roman Jakobson's *"functions of language"*[3] or Nelson Goodman's "*references*"[4]. Within this context, it will be possible to uncover and establish a set of aesthetic practices depending on the specific of writing in programming languages.

1 BURKE, K. Language as Symbolic Action, Oakland, 1966.
2 WITTGENSTEIN, L., Recherches Philosophiques, Paris, 2014.
3 JAKOBSON R., Questions de poétique, Paris, 1973.
4 GOODMAN, N., *Languages of Art: An Approach to a Theory of Symbols,* Cambridge, 1983.

Furthermore, one of the ontological differences between machine languages and human languages lie in the process by which they come into, and are established within, the world. Machine languages are designed and implemented by a known, restricted set of individuals, documented, justified and published, then updated, deprecated or abandoned. It therefore seems necessary to take into account this design process within a study of the aesthetics of code, in order to highlight how these specifications can influence and shape the artistic practices and literary texts which rely on them.
This unique nature of programming languages opens up the possibility of formulating the hypothesis of a large variety of expressive practices, reflecting those that already exist within human languages, both at a micro- and at a macro-level. On the micro-level, the basic units which constitute most of the programming languages (i.e. variable, type, function, condition, try/catch, arguments, comments, etc.) could be examined as potential stylistic figures, symbols or motifs, insofar as they carry within them representations of abstract computing concepts. These could then be organized in a dictionary of rhetoric of sorts, specifying each of the potentialities to support in a more or less explicit fashion the expression of certain "analog" ideas. On the macro-level, attention would be drawn to the general paradigms which inform the different approaches of designing and writing code, and replacing them within both a stylistic context and a cultural context. Here, questions of linearity, division, description, documentation and legibility can initiate a prolific discussion on poetic properties -in their world-making dimension- of lines of code, in the light of narrative arcs, plots and schemas.

## Analytical and artistic approaches

This research project is therefore articulated around the aesthetic possibilities of programming languages, considered as assemblages specifically designed in order to achieve a technical function that is clearly defined, and yet which always support an aesthetic practice on its peripheries and outskirts. The central question is then that of how programming languages allow for poetic texts. What are the respective roles of the functional and the superfluous, of the technical and the cultural, in an aesthetic of source code? How can the framework of aesthetic philosophy offer a new perspective on these texts? And how can these very aesthetic properties be influenced by the very process of designing programming languages?

In order to approach each of these interrogations, this research project will be based on a triple approach; first analytical, then historical and artistic. One of the main reasons for such a methodology is the lack of existing corpora centered specifically around the creative uses of code. The first step will therefore be to gather these texts, and establish a typology of creative practices of code-writing. This corpus will include, but not be limited to, texts from engineering (the comments of graduate students left in the Apollo XI landing module source code), from entertainment (the misogynistic variable naming in the source code of *Dead Island*), of amateur worlds

(the playful layouts of the *Obfuscated C Code Contest*), and artistic communities (the code poems shared on platforms such as *PerlMonks*). The process of constituting this corpus will therefore be synchronous to its analysis, without first limiting oneself to historical or sociological conditions. This wide-ranging approach will allow us to identify the broad strokes of creative practices in the writing and reading of source code, fully acknowledging the playful creativity of programming, as well as the porosity between amateur and professional practices. However, since this work is anchored within a tradition of aesthetic philosophy and literary semiotics, the limit of this corpus will be that of the opposition between uselessness and usefulness, and therefore exclusively on written code rather than executed code. Finally, a secondary corpus will include both "classical" programming languages (e.g. C++, Python, Java) as well as 'esoteric' languages, and the texts written through them.
Indeed, the examination of aesthetic possibilities of programming languages reveal that they situate themselves along a linear gradient along the limits of our field, between absolutely functional, and absolutely superficial. It is around this tension of use(ful/less)ness that the majority of creative practices in programming seem to be organized, as written and read by and for humans, rather than by and for machines. From there, the relationship between creative writing in programming languages and creative writing of programming languages will not be avoided, since they both engage in those underlying themes. Programming languages, such as *Piet, Befunge*, or *TrumpScript,* will therefore be a necessary addition to our corpus, since their very design is in itself an aesthetic usage of code.

Following the constitution and the analysis of this corpus, it will be necessary to introduce a historical perspective. Despite the lack of a comprehensive cultural history of programming languages, this project will add to the previous typologies and contextual dimension. I am formulating the two hypotheses that (1) economic, social and cultural circumstances in which a programming language was designed and gained popularity play a role in the unfolding of aesthetic practices within them, and (2) changes in paradigms and patterns (functional programming, object-oriented programming, lambda programming) share a relationship with contemporary social reconfigurations, with possible manifestations in a style of writing. The relationship between the constitution of artistic communities around the Perl language and the fact that it is released under an Artistic License, the obfuscation of C source code resonating with its role as a main component of low-level infrastructure, and its manipulation as modern *calligrammes*, and the looseness of JavaScript syntax enabling amateur experimentation through greater fault-tolerance; these are all examples of how historical contexts could relate to objects and why such contexts will be unavoidable in the study of source code.

Finally, the relative youth of this field of study itself and the limitation of existing texts calls for a third, artistic approach. This practical component would allow us to uncover, through the creative process, new avenues for this research project; since this project is centered around the *concrete* possibilities of code, it seems necessary to contribute

ourselves to these practices, in order to confront the scientific approach to the artistic activity that is code-writing. My personal artistic practice is already shaped around these questions, and will bring to this thesis project an aspect of speculation in what an aesthetic of code *could* be. The hypothesis, here, is to consider the evolution of programming languages, from punch cards of Jacquard looms until the ubiquity of Python in high school classrooms, in the light of the evolution of human languages through the last five millennia. These human languages initially exclusively represented trade relations, quantities of wares, then of currency, then royal and imperial edicts, and funerary eulogies. The current poetic, aesthetic use of language is therefore not immediately implied by its apparition as a tool for communication and recording. The current use of programming languages does not, therefore, prevent the possibility to imagine and use it as a new means of aesthetic expression, closer to prose or poetry than to protocol or directive. This artistic approach would allow for the generation not only of further research questions and leads of answers, but also other primary sources for future work on this subject.

## Sources and Methodology

The subject of this thesis project being directly related to digital technologies of computation and communication, the majority of my primary sources have been created, distributed and archived online. The writing of code is indeed closely linked to the philosophy of *open-source*, of radical sharing of the result of coding processes, a sharing philosophy which Linux programmer John Hall attributes to a certain aesthetic satisfaction and pride[5]. It is therefore relatively easy to access the works of enlightened amateurs with well-defined sites, such as the mailing list *nettime*, hosting several works of artists such as Mezangelle or Alan Sondheim, the *PerlMonksl* forums or the *runme.org* digital network. This open-source ethic also facilitates the access to the source of more elaborate systems, such as the Linux kernel, or leaked versions of the Microsoft Windows source code. Finally, the rise in popularity of user-friendly version control systems, such as *GitHub* or *Bitbucket* make public both the professional projects and the personal drafts of each. These sites will therefore be the main locus of exploration and documentation of tests underlying our analytical work. I have already begun such a work as preparation for courses that I have designed in the past at NYU, particularly *Politics of Code* and *Software Art: Text*.

The second component of these primary texts concern a somewhat niche ensemble of print publications where source code is the first and foremost object of attention, such as Ishaac Bertram's *{code poems}*, Nick Montfort's *#!*, Daniel Temkin's *Weird Languages* or Los Pequenos Glazier's *Algorithms*. While these works have a stronger footing in the world of art books or *zines*, compared to the aforementioned open-sourced texts, I have developed a familiarity with those communities and have already acquired most of these books. It is through these relations that I intend to further my

5 BLACK, M. J., The Art of Code, thesis defended in 2002, Philadelphia.

research on the practices of this artistic scene, and possibly be redirected towards aspects of creative code writing that I could have overlooked until then.

The methodological component of this research, and particularly focusing on the artistic production, will be articulated along three parts: intra-linguistic, inter-linguistic and meta-linguistic.

The intra-linguistic level will consist in writing and re-writing texts (both poems and novels as well as algorithms or typical applications, such as an HTTP server) in one give language in order to highlight the aesthetic specificities in the different writings of a same language, as Raymond Queneau did in his *Exercices de Style*. This will investigate the different directions in which the subjectivity of writing can be manifested, and how this subjectivity is further reactivated through the human reception without changing the final, functional result.

In respect to the inter-linguistic level, a similar program will be rewritten in different programming languages. The practice of translating a unique conceptual structure and transcribing it in a different implementation of the "universal language" will allow us to move from the macrostructures of style, formula and structure towards the microstructures of variable types, classes, interpretation, compilation, as well as the peculiarity of the respective ecosystems within which these languages live, such as modules, libraries, tutorials, package managers, community values, etc. The aim here is the development of a series of examples illustrating the different typologies established throughout the analytical component of our project, but also providing the possibility to open additional research avenues through this creative process.

Finally, the meta-linguistic level will consist in shifting the focus no longer on the writing of code through a programming language, but on the writing of a programming language in code. The design and implementation of a complete programming language is ontologically first when it comes to creative uses of code, and they are themselves systems that range from the useful to the useless. The process here would be to sketch possible implementations of alternative, or esoteric, programming languages to complement the analytical and historical research done beforehand in order to foster aesthetic and literary uses of code, the same way that languages like Rust foster thread-safety and memory-safety or how Perl fosters the input, process and output of string datatypes.

These multiple methodologies, first on the constitution of a corpus, then on a scientific analysis and historical contextualization, and finally on artistic creation, will allow this project to best encompass a still developing and largely unexplored field of research.

## Current state of the research

The approach proposed here stands at the intersection of multiple academic disciplines, such as aesthetic philosophy, computer science, media studies and linguistics. In that respect, this research project builds a large theoretical body of

work, and yet opens up a new position by reconfiguring it around the object of source code.

The main bibliography concerning programming languages belongs first and foremost to the field of computer sciences. These texts are mainly concerned with a functional style of writing programs, and not with the learning of programming itself, with an audience that is expected to already know how to write code, and which therefore is already getting closer to the realm of the aesthetic.

While being foundational texts in computer science and software engineering, texts such as *Structures and Interpretations of Computer Programs* and *The Art Of Computer Programming* are highlighting the tension between scientific coherence and the inherent craftsmanship in writing programs. This part of our references will allow us to start from a basis of what code writing should be from a strictly technical perspective, and thus to identify how creative code-writing can both draw and distinguish itself from it.

The study of code beyond the computer scientific field then takes place within media studies. If the work on the impact of coded systems on the different levels of our societies is under significant development, the close reading of the material and literary bases of these systems are more sparse. The works of N. Katherine Hayles, Lev Manovich, Wendy Hui Kyong Chun in North America, or Friedrich Kittler and Siegfrid Zielinski in Europe, or Yuk Hui in China, focus on the philosophical, political and cosmogonic implications of the digital as a radically different system of writing. The field of software studies, through the eponymous series edited by the MIT Press, is slowly reaching out of north-american universities into their european counterparts, particularly through the work of Anthony Masure in France, analyses through this lens coded objects as cultural artefacts. It is within this trajectory of cultural analysis of technical objects that this project intends to situate itself.

The practice of digital art is also an expanding field, both from a historical and cultural perspective and from an aesthetic and literary one. The computer-assisted work of art, gaining in popularity, and therefore gaining in critical framing, is still largely limited to actions and interfaces -which is to say to running code. It is in this context that the work of french scholars Philippe Bootz and Serge Bouchardon exist, yet still remain precious resources for having established a background work in terms of digital poetry, technotext and cybernetic aesthetic. The research around digital poetry encompasses all the textual forms which use computation as a primary artistic medium, from codeworks to interactive fiction and digital games. These analyses are particularly valuable in terms of conceptualizing the creating of artistic objects within computational systems, as formulated by the likes of Jack Burnham, Janet Murray or Charles O. Hartman.

It is also necessary to take into account that only a small subset of those fields take written code as a main object of study. The works of Camille Paloque-Berges is mainly focused on the circulation of these coded texts across networks, and the

doctoral thesis of Joseph Black highlights the specific literary traditions and existing texts with which code can be compared. Florian Cramer, as well as Geoff Cox and Alex McLean have published to major studies on the subject[67], which will be of a great help, while still maintaining the line of this current project rather on the aesthetic side, as opposed to the strong cultural theory approach that these two publications exhibit.

In order to achieve this, this thesis project will rely on a tradition of aesthetic and literary philosophy. The works of Wittgenstein, Nelson Goodman and Kenneth Burke, along with Roman Jakobson, Roland Barthes and Jacques Ranciere will be essential in the qualification of the relations between symbols, interpretations and contexts within source code, and will enable us to inscribe the analyzed practices of code-writing within the larger tradition of the study of language and art. These studies will be complemented by the works of pre-computing poets, such as the L=A=N=G=U=A=G=E group and the OuLiPo.

While the interdisciplinary nature of our project can thus allow for a vast bibliography, the specificity of our approach around the aesthetic affordances of writing source code clearly delineates our object of study. The unique nature of this project is therefore to build upon these multiple approaches in order to highlight the concrete manifestations of the aesthetic, semantic and cultural complexities of writing code.

## Indicative Timetable

*Semester 1*
- Constitution of the corpus
- Readings of secondary sources (media studies, software studies)

*Semester 2*
- Beginning a typology of creative uses of code
- Readings of secondary sources (computer programming, aesthetic philosophy, literary studies)

*Semester 3*
- Finishing the typology of the corpus and its analysis
- Readings of secondary sources on the history and design of programming languages

*Semester 4*
- Writing of additional code samples
- Analysis of the creative uses of programming languages

6 CRAMER, F., Words Made Flesh: Code, Culture, Imagination, Rotterdam, 2005.
7 COX, G. MCLEAN, A., Speaking Code: Coding as Aesthetic and Political Expression, Boston, 2012.

*Semester 5*
- Writing

*Semester 6*
- Writing
- Proofreading

## Indicative Bibliography

### Computer Science

ABELSON, H. SUSSMAN J.G., SUSSMAN, J., *Structure and Interpretation of Computer Programs*, Boston, 1996.
FRIEDMAN, D., P., MITCHELL, W., *Essentials of Programming Languages*, Boston, 2014.
INCE, D., C., *Mechanical Intelligence, Vol. 1* (Collected Works of Alan Turing), Amsterdam, 1992.
KERNIGHAN, D. , RICHIE D., *The C Programming Language*, New Jersey, 1988.
KNUTH, D., *The Art of Computer Programming*, Boston, 2015.

### Media studies

BOGOST, I. *The Rhetoric of Video Games* in *The Ecology of Games: Connecting Youth, Games and Learning*, ed. SALEN, K., Boston, 2008.
CHUN, W., *Programmed Visions: Software and Memory*, Boston, 2011.
COX, G. MCLEAN, A., *Speaking Code: Coding as Aesthetic and Political Expression*, Boston, 2012.
FULLER, M., *Behind the Blip: Essays on the Culture of Software*, Oakland, 2003.
GALLOWAY, A. *Gaming: Essays on Algorithmic Cultures*, Minneapolis, 2006.
HAYLES, N. K., *My Mother Was A Computer: Digital Subjects and Literary Texts*, Chicago, 2005.
KITTLER, F. *Discourse/Systems 1800/1900*, Stanford, 1999.
MANOVICH, L., *The Language of New Media*, Boston, 2001.
MASURE, A., *Le Design des Programmes: Des Façons de Faire du Numérique*, thesis defended in 2014, Paris.
MCLUHAN, M, *Understanding Media*, Berkeley, 1994.
MONTFORT, M., BAUDOIN, P., BOGOST, I., DOUGLASS, J., MARINO, M. C., MATEAS, M., REAS, C., SAMPLE, M., VAWTER, N., *10 PRINT CHR$(205.5+RND(1)); : GOTO 10*, Boston, 2012.
SACK, W. *Software Arts*, Boston, 2018.
VEE, A., *Coding Literacy: How Programming is Changing Writing*, Boston, 2018.
ZIELINSKI, S., *Deep Time of the Media: Toward an Archaeology of Hearing and Seeing by Technical Means*, Boston, 2006.

## Art Digital

CHOI, T., *Poetic Computation: A Reader*, http://poeticcomputation.info/ consulted on 09/12/2018.
CRAMER, F., *Words Made Flesh: Code, Culture, Imagination*, Rotterdam, 2005.
BLACK, M. J., *The Art of Code*, thesis defended in 2002, Philadelphia.
BOOTZ, P., *Poésie Numérique: Du Cybertexte Aux Formes Programmées*, Paris, 2006.
BOUCHARDON, S., *La Valeur Heuristique de la Littérature Numérique*, Paris, 2014.
BURNHAM, J., HAACKE, H., *Systems Aesthetics*, QUINZ Emanuele (ed.)
Dijon, 2015.
HARTMAN, C. O., *Virtual Muse: Experiments in Computer Poetry*, Middletown, 1996.
MURRAY, J., *Hamlet on The Holodeck*, Boston, 1998.
PALOQUE-BERGES, C., *Poétique des codes sur le réseau informatique*, Paris, 2009.
Esthétique & Linguistique
ADORNO, T. W., *Théorie Esthétique*, ed. ADORNO G. et TIEDERMANN R., trad. JIMENEZ, M., Paris, 2011.
BURKE, K. *Language as Symbolic Action*, Oakland, 1966.
GOODMAN, N., *Languages of Art: An Approach to a Theory of Symbols,* Cambridge, 1983.
JAKOBSON R., *Questions de poétique*, Paris, 1973.
RANCIERE, J. *Aisthesis : Scènes du régime esthétique de l'art*, Paris, 2012.
WITTGENSTEIN, L., *Tractactus Logico-Philosophicus*, Paris, 1993.
WITTGENSTEIN, L., *Recherches Philosophiques*, Paris, 2014.