Le rôle de l'esthétique dans les compréhensions du code source

Pierre Depaz
sous la direction d'Alexandre Gefen (Paris-3)
et Nick Montfort (MIT)

ED120 - THALIM

Février 2023

1 Introduction

En tant que texte dont le but même est de disparaître (transformé en changements de courant électrique), le code source est un objet hybride, autant description qu'action, aux interlocuteurs multiples. Il peut être communication entre humain et humain, entre humain et machine, ou entre machine et machine. Cependant, en tant que création humaine, il est possible de s'interroger sur les modalités de ses manifestations, notamment en termes d'expressivité. Si l'expressivité fonctionnelle d'un code source implémentant un algorithme est indéniable, l'expressivité artistique d'un texte de code source demeure élusive. Quelle est donc la place de l'esthétique, d'une manifestation formelle sensuellement plaisante, dans l'écriture ou la lecture du code source? Le code source, basé sur un système syntaxique similaire au language naturel, se révèle être un système de communication aussi bien d'humain à humain que d'humain à machine. Il

semble néanmoins être principalement destiné les humains, et secondairement aux les machines (Abelson et al., 1979). Une fois que la condition principale d'existence du code source (sa validité d'exécution) est satisfaite, une condition secondaire semble être non pas sa beauté mais sa compréhensibilité.

Se pose donc la question d'un texte dont les manifestations formelles sont vouées à disparaître, et dont la lecture n'est qu'un processus collatéral de son exécution. Donald Knuth, dans son oeuvre *The Art of Computer Programming* (Knuth, 1997), commence par établir que la programmation (l'écriture et la lecture du code source) est bel et bien un art. Cette déclaration, dans les premières pages du premier volume, n'est pourtant pas réitérée ni élaborée dans les autres volumes du monographe. Si écrire du code peut être un art, certains codes sources peuvent donc exhiber des propriétés esthétiques, mais ces dernières sont rarement explicitées par la littérature sur le sujet.

La problématique principale de ce projet de recherche est donc celle du *rôle de l'esthétique dans les compréhensions du code sources*. Il s'agit tout d'abord de mettre à jour les propriétés esthétiques propres au code source, et d'identifier comment celles-ci permettent de faire sens. En plus d'une approche empirique des codes sources en eux-mêmes, il s'agit de mettre à jour de quelles manières ces propriétés sont similaires à d'autres domaines de créations mobilisés par les programmeurs, notamment au domaine de la littérature, de l'architecture, des mathématiques et de l'ingénierie. En examinant ces relations, cette thèse s'inscrit donc dans un travail de recherche sur la dimension cognitive du phénomène esthétique.

Une approche rapide de l'état de l'art sur ce sujet révèle deux tendances séparées: d'une part, la littérature en informatique et ingénierie prend en compte l'évidence de l'existence d'une esthétique du code source, du point de vue la productivité de ceux et celles qui écrivent du code, et d'un point de vue cognitif de la compréhension des bases de code (Oram & Wilson, 2007; Cox & Fisher, 2009; Gabriel & Goldman, 2001; Martin, 2008; Deti-

enne, 2012; Weinberg, 1998). Néanmoins, ces études considèrent une esthétique comme un phénomène donné, aux conséquences plus ou moins mesurables, sans pour autant systématiser leur approche au niveau conceptuel.

D'autre part, les recherches en sciences humaines traitent de l'interaction entre esthétique et code, tout en restant majoritairement attachées à une conception abstraite et désincarnée du "code", élaborant une esthétique du digital sans pour autant rentrer dans les détails des codes sources eux-mêmes (e.g. il ne semble pas pertinent qu'ils soient écrits en Perl, ou Python, ou Ruby, etc.) (Cramer, 2019; Hayles, 2010; Mackenzie, 2006; Lévy, 1992). Il existe cependant un certain nombres d'ouvrages de sciences humaines approchant la question matérielle du code de manière directe, que ce soit au niveau culturel (Montfort et al., 2014), politique (Cox & McLean, 2013) ou sociologique (Paloque-Bergès, 2009) Deux travaux doctoraux se sont précédemment intéressés à l'esthétique du code source (Black, 2002; Pineiro, 2003), mais se limitent à des communautés d'écriture du code bien précises, sans examiner les codes sources eux-mêmes. C'est au sein de cette approche sur les manifestations du code source que cette recherche s'inscrit.

2 Méthodologie

L'approche principale de ce projet est tout d'abord empirique. Il s'agit d'examiner les codes sources eux-mêmes, ainsi que les discours de ceux et celles qui les écrivent et les lisent. Pour cela, je m'appuie sur les travaux de Kintsch et van Dijk et leurs études des stratégies de compréhension du discours (Kintsch & van Dijk, 1978), l'approche métaphorique considérée comme une stratégie de compréhension. Le code source étant un texte, autour duquel sont produits des discours.

L'approche empirique de cette thèse a tout d'abord fait émerger une var-

iété d'individus écrivant et lisant du code, que j'ai regroupé en quatre catégories (Hayes, 2017). Les *développeurs* sont les individus qui écrivent du code dans un contexte économique, à but fonctionel et soutenable à long-terme; ils travaillent sur des bases de code extensives et souvent en collaboration. Les *artistes* sont les individus écrivants du code dans un cadre expressif et artistique, tels que des *code poems*; leur code est destiné principalement à être lu par un public, et seulement de manière secondaire à être exécuté. Les *hackers* sont les individus qui écrivent du code en tant que solution technique unique, idiosyncratique et hautement contextuelle; leur code est destiné à être exécuté par une machine particulière, et de manière secondaire par des humains. Enfin, les *académiques* sont les individus qui écrivent du code afin d'illustrer des concepts de computation, représentant des abstractions plutôt que des usages concrets.

Ces quatres groupes ont des limites poreuses, et un code source peut, souvent, appartenir à plusieurs de ces catégories en même temps (une référence artistique dans un contexte commercial, un hack présenté comme oeuvre d'art, ou une approche particulière au sein d'une base de code commerciale, etc.).

La constitution du corpus de cette recherche s'est faite principalement par la consultation de ressources en ligne. En effet, la plupart des bases de code existe sur des sites d'aggrégations (GitHub, BitBucket, etc.) ou sous formes d'extraits (*snippets*) présentés et commentés dans des forums, sur des sites personnels, ou des sites de question/réponse (e.g. StackOverflow, Quora). Cette constitution du corpus inclue non seulement ces sources primaires (le code en soi), mais également des sources secondaires (le commentaire par un auteur, ou par un public, justifiant de l'aspect esthétique d'un code présenté). De cette manière, je considère la valeur esthétique d'un code source comme étroitement lié au jugement de groupe émis à son encontre.

Ma métholodologie consiste en une approche interprétative du corpus constitué, dans un double-mouvement. D'une part, les références et dis-

cours des individus écrivants du code sont pris en compte afin de constituter un ensemble de standards esthétiques: ces standards sont des manifestations concrètes du code permettant une facilitation de la compréhension du *propos* du code (propos prescriptif, ou effectif). D'autre part, les standards exhibés sont appliqués à d'autres bases de code afin de vérfier leur applicabilité. Ã travers la multiplicité de ces standards, il s'agit donc d'identifier les stratégies métaphoriques de compréhensions des lecteurs de code.

Cette examination se fait également à travers un cadre théorique partant de la philosophie esthétique et de la littérature afin de définir mon utilisation du terme *esthétique*. D'un point de vue littéraire, je m'appuie sur les travaux de Gérard Genette et sa distinction entre fiction et diction (Genette, 1993), considérant l'esthétique du code comme sa diction, tandis que la poétique serait sa fiction. Entre littérature, philosophie et sciences cognitives se trouvent les oeuvres de Paul Ricoeur (Ricoeur, 2003) et de George Lakoff (Lakoff, 1980), reliant composition verbale et évocation d'images mentales. Enfin, cette manifestation en surface est reprise par le philosophe de l'art Nelson Goodman dans son analyse des languages de l'art (Goodman, 1976), et notamment dans son exploration entre systèmes syntactiques et communication, ainsi que son approche scientifique des phénomènes artistiques.

Les travaux sur la métaphore de Ricoeur et Lakoff permettent aussi d'identifier des domaines adjacents mobilisés par les programmeurs pour justifier de leurs jugements esthétiques. C'est en étudiant les zones de contact de ces domaines qu'il est possible d'identifier les aspects spécifiques du code source.

3 Développement

Premièrement, il n'y a pas "un" code source, abstrait et désincarné, mais bien une multiplicité de codes sources, existant au sein de pratiques et de groups différents. Ces codes sources possèdent tous une possibilité de manifestation esthétique, manifestations soumise à la manifestation et au transfert de *connaissances*. En ce que le code source possède tant un son aspect prescriptif (ce que le code *doit* faire) qu'un aspect effectif (ce que le code *fait*), les esthétiques du code source peuvent se décliner le long de cette axe, avec les poètes et les académiques se concentrant sur l'aspect prescriptif, évoquant des concepts à travers la machine, et les hackeurs se concentrant sur l'aspect effectif, évoquant les concepts de la machine.

Pour communiquer l'intention du programme à travers son implémentation textuelle et en rapport avec son domaine d'activité, les programmeurs disposent de différents possibilités. Il est possible de développer une certaine du problème vers à travers divers niveaux de métaphores linguistiques (rhétorique procédurelle (Bogost, 2007), double-codage (Cox & McLean, 2013), double-signification (Paloque-Bergès, 2009)) par un choix précis de vocabulaire. Le code est donc ici une double sorte de langue: langue machinique et langue humaine dont les tensions syntactiques peuvent créer une richesse sémantique.

Ensuite, ces mécanismes linguistiques sont complémentés par des choix de structures et de syntaxe faisant appel à des domaines d'architecture et d'ingénierie (Gabriel, 1998; Schummer et al., 2009), qui permettent alors de mieux appréhender une exécution du code source—exécution dont la vitesse à laquelle elle se produit la rend incompréhensible pour des humains. L'esthétique du code source permet alors de représenter les espaces d'évolution des flux d'informations lors de l'exécution d'un programme.

Finalement, ces standards esthétiques se regroupent autour de la notion d'élégance—c'est-à-dire de l'utilisation d'une syntaxe minimale pour

une expressivité maximale. Particulièrement présente dans les groupes de hackeurs et des académiques, celle-ci se décline aussi le long d'un axe machine-concept. Les hackeurs ont tendance à n'utiliser que ce qui est nécéssaire pour effectuer une action particulière, bien visible dans les concours de *demoscenes* (Kudra, 2020), tandis que les académiques vont témoigner de la même approche, mais pour communiquer des concepts fondamentaux de la science informatique, découplée de la machine spécifique qui exécute le programme en question.

4 Conclusion

En fin de compte, nous avons montré que les représentations métaphoriques du code, représentations du code comme language, comme architecture, et comme matériau, se retrouvent dans l'expression d'une conception d'un espace sémantique dynamique. Cette conception spatiale infuse donc les propriétés esthétiques propres du code, en ce que la syntaxe et la structure du code source se concentrent principalement à la description et à la navigation d'espaces conceptuels, même si les différents contextes socio-techniques dans lesquels ces codes sources sont écrits vont eux-mêmes moduler la nature de ces espaces.

References

Abelson, H., Sussman, G. J., & Sussman, J. (1979). *Structure and Interpretation of Computer Programs - 2nd Edition*. Justin Kelly.

Black, M. J. (2002). The art of code. *Dissertations available from ProQuest*, (pp. 1–228).

Bogost, I. (2007). *The Rhetoric of Video Games*.

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*.

Genette, G. (1993). *Fiction & Diction*. Cornell University Press.

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.

Kintsch, W., & van Dijk, T. A. (1978). Toward a model of text comprehension and production. *Psychological Review*, *85*(5), 363–394.

Knuth, D. E. (1997). *The Art of Computer Programming, Volume 1 (3rd Ed.): Fundamental Algorithms*. USA: Addison Wesley Longman Publishing Co., Inc.

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.

Lévy, P. (1992). *De la programmation considérée comme un des beaux-arts*. Textes à l'appui. Anthropologie des sciences et des techniques. Paris: Éd. la Découverte.

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.

Pineiro, E. (2003). *The Aesthetics of Code : On Excellence in Instrumental Action*. Ph.D. thesis, KTH, Superseded Departments, Industrial Economics and Management.

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.