An Ethnography of Codework

Making the Epistemology of Digital Humanities Coding Visible

Joris J. van Zundert
Huygens Institute for the History of the Netherlands
Royal Netherlands Academy of Arts and Sciences
Amsterdam, The Netherlands
joris.van.zundert@huygens.knaw.nl

Smiljana Antonijević
Penn State University
State College, US
smiljana@smiljana.org

Tara L. Andrews
Digital Humanities
University of Vienna
Vienna, Austria
tara.andrews@univie.ac.at

Abstract

Code and codework are often treated as an invisible hand in (digital) humanities. This creates various problems: of understanding the epistemological qualities of code, of academic credit for coding, and of scientific accountability for the digital objects that derive from codework. We argue that more insight into the practice of codework going on within digital humanities is needed, using the insights generated by an ethnographic approach to exemplify our argument. To facilitate comprehension by scholars we categorized our findings according to Cicero's framework of classical rhetorics. Based on these findings we contend that an encompassing strategy is needed to give codework a proper theoretical and methodological academic recognition, making code and codework visible, understandable, trustworthy and reputable within humanities scholarship. Theoretical discussions of codework should become an established trajectory in the humanities, along with the development of methods for documenting, analyzing, and evaluating code and codework. A form of peer review should develop that recognizes the reciprocal relationship between code and text and their respective strengths in the formation and expression of humanities theory and research.

Full proposal

Neither the humanities nor the digital humanities (DH) habitually engage with code and programming in an explicit and critical manner. This un-critiqued use of code means in turn that the scholarly quality and contribution of codework goes both uncredited and unaccounted for. As expressions of a technē whose inner workings are opaque to most humanities scholars, code and codework are all too often treated as an invisible hand, influencing humanities research in non-transparent ways. To prevent neglect of its epistemological contribution and so to not imperil one of the key components of knowledge production in

DH, we need more insight into code and codework in the humanities.

The purpose of our paper is to provide some of those insights in the form of an ethnography of codework, wherein we observe the decisions that programmers make and how they understand their activities. Our study follows in the footsteps of ethnographies of technoscientific practice (see: Forsythe, 2001; Coleman, 2013), critical code studies (see: Marino, 2010), and reflections on coding and tool development in DH (see: Schreibman and Hanlon, 2010; Ramsey and Rockwell, 2012). The study aspires not to be fully representative of DH coding practice, but to initiate a debate about overlooked elements of that practice.

Our exploration applies Latour's (1998) first rule of method to the context of narrative creation through codework, looking at the practices, dilemmas, and decisions of programmers. We used analytical autoethnography (cf. Anderson 2006) combined with collaborative ethnography (cf. Lassiter 2005). Written accounts of codework are the basis for a series of team discussions, both written and oral, that informed the results of our contribution. This methodological design enabled us to return from the final outputs of DH coding to scholarly uncertainties and resolutions that preceded them. Such reconstruction enables us to document some of the key phases in epistemological construction of coding artifacts, and to identify methodologically significant moments in the stabilization of those artifacts.

We grouped our observations into categories known as the five canons of rhetoric, proposed in Cicero's *De Inventione*. Originally developed for public speaking, these canons have proven to be an equally potent heuristic for analyzing written and, more recently, digital discourse (Gurak & Antonijevic 2009). The classical framework is applicable because code and codework, like text, can be understood as argument, congruent to Galey and Ruecker's (2010) view of the epistemological status of graphical user interfaces as argument. From an epistemic point of view, the practice of a programmer is no different from the practice of a scholar (Van Zundert, 2016): both are creating theories about existing epistemic objects (e.g. text and material artifacts, or data) by developing new epistemic objects (e.g. journal articles and critical editions, or code). By applying a rhetorical framework we do not seek to fit codework into a normative ontology, but hope to provide an explanatory form that facilitates interpretation by scholars.

Our investigation illuminated how codework reflects humanistic discovery (*inventio*) in that humanities-specific research questions drives coding. Similarly, crafting and organizing code resonates with development and arrangement of a scholarly argument (*dispositio*). Our study also illustrated that, like any humanities scholar, an author of software has her own style (*elocutio*) in the aesthetics of code and in her way of working to create code, and this style develops through both individual norms and norms of coding communities. We also showed that, parallel to books or libraries, code and codework serve as memory systems (*memoria*) that embed theoretical concepts in order to augment research methodology and create new theory. Finally, our ethnography illustrated how codework *actio* compares to the publication and reception of the software.

To give codework a proper theoretical and methodological academic recognition, with both the consequences and the rewards that such a recognition bears, a strategy for making code and codework visible, understandable, trustworthy and reputable within humanities scholarship is needed. Such a strategy should be comprehensive, both in the sense of accounting for the source code and the executed result of software. While we agree with Ramsay and Rockwell (2012) that providing source code is not sufficient for understanding the underlying theoretical assumptions, we disagree in viewing the 'dependence on discourse' as a feature that relativises epistemic and communicative capacities of code and codework. We argue in contrast that interdependence of code and text should be embraced as a means of acknowledging their distinctive yet corresponding methods of knowledge production and communication. We believe that theoretical discussions of codework should become an established trajectory in the humanities, along with the development of methods for documenting, analyzing, and evaluating code and codework.

An important element of a strategy to make codework visible is understanding codework as necessarily shaped by its social context, which influences the attitude and perception that both coders and other scholars hold towards their work. Often DH programmers are treated as service instead of research focused scholars. A necessary step therefore is to regard code as an alternative epistemology with equal research value and validity, instead of subordinating code and codework to 'humanities proper' (cf. Burgess & Hamming 2011 and Ramsay & Rockwell 2012), and the recognition of peer-reviewed digital outputs, including code, as research outputs (cf. Nowviskie, 2011; Presner, 2012; American Historical Association, 2015). A precondition for this are grassroots procedures for peer review of code (Fitzpatrick 2011) and critical examination of actual code, which is hardly even nascent in DH (Zundert & Haentjens Dekker 2017). Finally, reflexive accounts on (digital) humanities codework and ethnographic studies of actual work help us understand how codework is changing the humanities (Borgman 2009). An important step in illuminating the process and results of DH codework is to develop and explicate reflexive insights into its key epistemological, methodological, and technical aspects. Explaining, for instance, what kind of research questions give impetus to one's codework and how new research insights co-evolve during code development helps both DH programmers and their traditionally trained colleagues recognize the important epistemological connections between humanistic theory and scholarly programming.

References

American Historical Association, Ad Hoc Committee on Professional Evaluation of Digital Scholarship by Historians, *Guidelines for the Professional Evaluation of Digital Scholarship in History*, April 2015, p. 10 <http://bit.ly/1PC1tDL> [accessed 8 November 2017]

Anderson, Leon, 'Analytic Autoethnography', *Journal of Contemporary Ethnography*, 35 (2006), 373–95 <https://doi.org/10.1177/0891241605280449>

Borgman, Christine, 'The Digital Future Is Now: A Call to Action for the Humanities', *Digital Humanities Quarterly*, 3 (2009) <www.digitalhumanities.org/dhq/vol/3/4/000077/000077.html>

Burgess, Helen J., and Jeanne Hamming, 'New Media in Academy: Labor and the Production of Knowledge in Scholarly Multimedia', *DHQ: Digital Humanities Quarterly*, 5 (2011)

<http://digitalhumanities.org/dhq/vol/5/3/000102/000102.html> [accessed 2 September 2016]

Coleman, E. Gabriella, *Coding Freedom: The Ethics and Aesthetics of Hacking* (Princeton (US), Woodstock (UK): Princeton University Press, 2013) <http://gabriellacoleman.org/Coleman-Coding-Freedom.pdf> [accessed 8 November 2017]

Fitzpatrick, Kathleen, ‘Peer Review, Judgment, and Reading’, *Profession*, 2011, 196–201 <https://doi.org/prof.2011.2011.1.196>

Forsythe, D., and D.J. Hess, *Studying Those Who Study Us: An Anthropologist in the World of Artificial Intelligence*, Studying Those Who Study Us: An Anthropologist in the World of Artificial Intelligence (Stanford, CA: Stanford University Press, 2001)

Galey, Alan, and Stan Ruecker, ‘How a Prototype Argues’, *Literary and Linguistic Computing*, 25 (2010), 405–424 <https://doi.org/10.1093/llc/fqq021>

Gurak, L., and S. Antonijevic, ‘Digital Rhetoric and Public Discourse’, in *The Sage Handbook of Rhetorical Studies*, ed. by A.A. Lunsford, R.A. Eberly, and K.H. Wilson (London, Thousand Oaks: SAGE Publications, Inc., 2017), pp. 497–508

Lassiter, L.E., *The Chicago Guide to Collaborative Ethnography*, Chicago Guides to Writing, Edi (Chicago, London: University of Chicago Press, 2005) <http://bit.ly/2iLCmGY>

Latour, Bruno, *Science in Action: How to Follow Scientists and Engineers Through Society* (Cambridge, MA, USA: Harvard University Press, 1988)

Marino, Mark C., ‘Critical Code Studies and the Electronic Book Review: An Introduction’, *Electronic Book Review*, (2010) <http://www.electronicbookreview.com/thread/firstperson/ningislanded>

Nowviskie, Bethany, ‘Where Credit Is Due: Preconditions for the Evaluation of Collaborative Digital Scholarship’, *Profession*, 2011, 169–181 <https://doi.org/prof.2011.2011.1.169>

Presner, Todd, ‘How to Evaluate Digital Scholarship’, *Journal of Digital Humanities*, 1 (2012) <http://journalofdigitalhumanities.org/1-4/how-to-evaluate-digital-scholarship-by-todd-presner/>

Ramsay, Stephen, and Geoffrey Rockwell, ‘Developing Things: Notes toward an Epistemology of Building in the Digital Humanities’, in *Debates in the Digital Humanities*, ed. by Matthew K. Gold (Minneapolis: University of Minnesota Press, 2012), pp. 75–84 <http://dhdebates.gc.cuny.edu/debates/text/11>

Schreibman, Susan, and Ann M. Hanlon, ‘Determining Value for Digital Humanities Tools: Report on a Survey of Tool Developers’, *DHQ: Digital Humanities Quarterly*, 4 (2010) <http://digitalhumanities.org/dhq/vol/4/2/000083/000083.html> [accessed 9 November 2017]

Zundert, Joris J. van, ‘Author, Editor, Engineer — Code & the Rewriting of Authorship in Scholarly Editing’, *Interdisciplinary Science Reviews*, 40 (2016), 349–375 <https://doi.org/http://dx.doi.org/10.1080/03080188.2016.1165453>

Zundert, Joris J. van, and Ronald Haentjens Dekker, ‘Code, Scholarship, and Criticism: When Is Coding Scholarship and When Is It Not?’, *Digital Scholarship in the Humanities*, 2017 <https://doi.org/https://doi.org/10.1093/llc/fqx006>