Cayley, John. "The Code Is Not the Text (Unless It Is the Text)." *Grammalepsy: Essays on Digital Language Art*. New York: Bloomsbury Academic, 2018. 53–66. *Bloomsbury Collections*. Web. 26 Apr. 2023. <<http://dx.doi.org/10.5040/9781501335792.ch-005>>.

Downloaded from Bloomsbury Collections, www.bloomsburycollections.com, 26 April 2023, 11:36 UTC.

Copyright © John Cayley 2018. Released under a CC BY-NC-ND licence (<https://creativecommons.org/licenses/by-nc-nd/3.0/>). You may share this work for non-commercial purposes only, provided you give attribution to the copyright holder and the publisher, and provide a link to the Creative Commons licence.

4

The Code Is Not the Text (Unless It Is the Text)

The use of networked and programmable systems as both delivery and compositional media for literal and verbal art (and other forms of new media art) has provoked critical engagements which pretend to reveal and exam the various levels of code and encoding which are constituent of programmatological systems.¹ The title of the section of the p0es1s program which stimulated this paper—"Code as Text as Literature"—is a case in point.² In more extreme forms of such engagement, a radical post-human reductionism may be proposed, such as that, for example, which can be read from certain of Friedrich Kittler's essays, in which the ramifications of so-called human culture, especially as played out on new media, become qualitatively indistinguishable from "signifiers of voltage difference," demonstrably the final, lowest-level "ground code" of the increasingly familiar practices of cultural production which make use of programmable tools; and perhaps also essential to the brain activity which generates the objects and subjects of psychoanalysis.³ Nowadays voltage difference accounts for and instantiates everything from the encrypted transactional play of internet banking to the promised consensual hallucination of immersive virtual reality. However the purpose of this brief paper is to address a number of less productive confusions which arise from this engagement with code-as-text, citing a few examples of artistic practice and a number of critical sources.⁴ While allowing the value of certain metacritical statements such as Kittler's (which take on questions of what culture is or may become), my aim is to disallow a willful critical confusion of code and text, to make it harder for critics to avoid addressing one or the other by pretending that they are somehow equivalent, or that codes and texts are themselves ambiguously addressed to human readers and/or machinic processors (unless they are so addressed, however ambiguously).⁵

I have invoked reductionism, and by this I mean a critical thrust which, implicitly or otherwise, asks questions like, “What (ultimately) is this object we are examining? What is its structure? What are its essential or operative characteristics?” and then finds special critical significance in the answers proposed. In N. Katherine Hayles sophisticated version of what can be read as a code-as-text argument, this reductive inclination is in evidence. Her essay “Virtual Bodies and Flickering Signifiers” discovers a new or emergent object, the flickering signifier, and derives important consequences from its instantiations and methods. “The contemporary pressure toward dematerialization, understood as an epistemic shift toward pattern/randomness and away from presence/absence, affects human and textual bodies on two levels at once, as a change in the body (the material substrate) and a change in the message (the codes of representation).”⁶ In other words, Hayles suggests that the constituent structure of the signifier itself may be seen as changed in contemporary culture and especially as expressed in “new media.” Both the materiality and the represented content of cultural practice and production have been affected. Before examining parts of Hayles’s argument in more detail, I want simply to point out that it is clearly determined by its metacritical significance and has a reductive inclination: signifiers have come to be such and such, therefore—albeit in a cybernetic feedback loop—cultural production (in Hayles’s essay “the represented worlds of contemporary fiction”) follows suit. Hayles’s characterization of a multiply mediated signifier which flickers from level to level in chained coded structures is, as a metacritical statement, highly suggestive and useful. However, when it comes to art practice and the critique of this practice, how does such insight figure?

What is missing from Hayles’s analysis is a set of relationships—relationships constituted by artistic practice—between a newly problematized linguistic materiality and represented content. These would inevitably express themselves in formal as well as conceptual address to what she identifies as a changed matter of language and literature. Hayles’s chosen examples, with, perhaps, the exception of her use of William Burroughs, demonstrate conceptual, rather than formal, address; they represent flickering signification as concept rather than as instantiation in the language of the work. Hayles cites, most extensively, William Gibson’s *Neuromancer* as a prime example of represented content affected by and expressive of the flickering signifier. While Gibson brilliantly conveys the literally flickering, scanned and rasterized, apparent immateriality of an informatic realm, the “consensual hallucination” of “cyberspace” (his famous coinage) and its interpenetration of meatspace, he does this in a book—“a durable material substrate”—in a more or less conventional novel, one in which, indeed, narrative predominates over character development and in which language functions in a relatively straightforward manner. Not even the narrative perspective (omniscient author third person) is shifted or experimentally

inflected in any of Gibson's cyberpunk classics. The writing is sharp and inventive but entirely subject to paraphrase.

There are further significant ironies here, for Hayles begins her essay by discussing typewriting. The physicality and static impression-making of this process of inscription is contrasted with that of word processing where less substantial bodily gestures cause word-as-(flickering)-image to be scanned onto the surface of a screen. "As I work with the text-as-flickering-image, I instantiate within my body the habitual patterns of movement that make pattern and randomness more real, more relevant, and more powerful than presence and absence."⁷ However, the exemplar most present later in her argument, Gibson, has made some play of his preference for composing his novels using a typewriter.⁸ Thus not only are the formal characteristics and the materiality of Gibson's language at odds with the flickering signification of its represented content, but, at the very least, the once-preferred experience of this writer—his phenomenology of inscription—is an apparent denial of Hayles's critical progression. I want to emphasize, in making these remarks, that if the subjective experience of the critic or reader is brought forward as evidence for a change in the structures of signification, then it is all the more important to examine the practices of the writer and the formal qualities of the work produced by those practices. Gibson sitting at a typewriter composing a novel may well produce a representation of the concept of flickering signification, but his practice does not necessarily embody the potential for new structures of meaning generation, or instantiate a corresponding materiality of language.

We will return to practice, but first I would like to examine Hayles's flickering signifier in so far as it engages with the notion of code-as-text.⁹ "In informatics, the signifier can no longer be understood as a single marker, for example an ink mark on a page. Rather it exists as a flexible chain of markers bound together by the arbitrary relations specified by the relevant codes." At least since Saussure, it seems somewhat redundant to point to the arbitrariness of any signifier-signified relation. I suppose that Hayles is actually referring to these relations as "arbitrary" because they are not necessarily significant as human readings; they are not addressed to general human readers but only to the systems and systems-makers who have coded or specified them for certain purposes. They are, nonetheless, construable and are far from arbitrary when considered as addressed to the systems in which they are embedded. They have both significance and consequence. "As I write these words on my computer, I see the lights on the video screen, but for the computer, the relevant signifiers are electronic polarities on disk." That is, they are Kittler's (fundamental) signifiers of voltage difference.

Intervening between what I see and what the computer reads are the machine code that correlates these symbols with binary digits, the compiler language that correlates these symbols with higher-level

instructions determining how the symbols are to be manipulated, the processing program that mediates between these instructions and the commands I give the computer, and so forth. A signifier on one level becomes a signifier on the next-higher level.

Hayles goes on to discuss the “astonishing power”—in the now-familiar technological sense of “power”—which these “arbitrary,” hierarchically structured chains of codes generate, since manipulations, interpreted as commands at one level, can have cascading, global effects.¹⁰ By shifting the argument in this way, I think she has bracketed a more significant consequence of the structure of signification which she is delineating: the question of address, the address of the specific encoded “levels.”

In an article on “digital code and literary text,” Florian Cramer has pointed out that, as he somewhat obscurely puts it, “the namespace of executable instruction code and nonexecutable code is flat.”¹¹ From the context it is clear that he means that the same character or symbol set is used—for example—to transcribe both the text being word processed and (to be precise) the source code of the program which may be doing the word processing. On the level plains of letters and bits, there is no radical disjuncture in the symbolic media when we cross from a region of “executable” text to text “for human consumption.” From the human reader’s point of view, they are both more or less construable strings of letters; from the processing hardware’s point of view they are more or less construable sequences of voltage differences. On the one hand, this statement is related to the famous inter-media translatability of digitized cultural objects (once coded, regular procedures can be used to manipulate an image, a segment of audio, a text, etc., without distinction, disregarding the significance or affect of the manipulation). Cramer is, however, more concerned with the potential for sampling and mixing code and text (in the contemporary music sense). Again, as in Hayles’s analysis, the question of the address of specific code segments and texts is bracketed. Not only is it bracketed, but the range of positions of address is simplified, as if we are speaking of a flat letterspace for code on the one hand and text on the other; whereas, clearly, there are many levels. Both Cramer and Hayles recognize a multilevel hierarchy of codes without elaborating or distinguishing them in the course of their discussions. Within the field of networked and programmable media, at the very least, we can acknowledge: machine codes, tokenized codes, low-level languages, high-level languages, scripting languages, macro languages, markup languages, operating systems and their scripting language, the human-computer interface (HCI), the procedural descriptions of software manuals, and a very large number of texts addressed to entirely human concerns.¹²

For Cramer, and not only for Cramer, this simplified, bracketed, or ambiguous textual address has become a valorized aesthetic and even a political principle: “computers and digital poetry might teach us to pay more

attention to codes and control structures coded into all language. In more general terms, program code contaminates in itself two concepts which are traditionally juxtaposed and unresolved in modern linguistics: the structure, as conceived of in formalism and structuralism, and the performative, as developed by speech act theory.”¹³ To attempt a paraphrase: working or sampled or intermixed or collaged code, where it is presented as verbal art, is seen by Cramer to represent, in itself, a revelation of underlying, perhaps even concealed, structures of control, and also (because of its origins in operative, efficacious program code) to instantiate a genuinely “performative” textuality, a textuality which “does” something, which alters the behavior of a system. It has the “astonishing power” of other cultural manifestations of new technology and new media, the power that Hayles has also recognized as a function of the coded structures arranged at various “levels” in programmatological systems, chained together by a literal topography, which is “flattened” by a shared symbol set. We should pause to consider what this power amounts to. What are the systems whose behavior can be altered by this power?

In the criticism of theoretically sophisticated poetics, there is a parallel aesthetic and political agenda, which I am tempted to call the Reveal Code Aesthetic. It is partly documented and particularly well represented in, for example, Marjorie Perloff’s *Radical Artifice*, where “reveal code” is revealed as a project of L=A=N=G=U=A=G=E writers such as Charles Bernstein, after having been properly and correctly situated in the traditions of process-based, generative and/or constrained literature and potential literature by Modernist, OuLiPian, Fluxus, and related writers culminating, for Perloff, in John Cage and the L=A=N=G=U=A=G=E writers themselves.¹⁴ Although the political and aesthetic of program of “reveal code” appears to be shared with Cramer’s new media writers, in the context of Perloff’s poetics, the codes revealed and deconstructed in language per se (rather than digitized textuality) are as much those of “the inaccessible system core,” the machinic devices that conceal “the systems that control the formats that determine the genres of our everyday life.”¹⁵ While the progressive tenor of an aesthetic and political deconstruction underlies this project, there is something of a Luddite tone in Perloff.¹⁶ New media writers and artists necessarily have more ambiguous political and aesthetic relations with the control structures of the media which carries their work.

The code-revealing language artists discussed by Perloff, both in their work and in their performance—be it textual performance or performance art per se or activism or (academic) critical practice—represent far better examples of the instantiation of pattern/randomness (distinguished from presence/absence) than the novelists cited by Hayles, even including Burroughs or Pynchon. While retaining her focus on the contemporary or near-contemporary writers which she associates with an innovative, L=A=N=G=U=A=G=E-inflected poetics having avant-garde inclinations,

Perloff recalls an extensive tradition of poetic literature which is marked by both its attention to the materiality of language and its radicalization of poetic practices.¹⁷ Perloff invokes formations and works by individuals which are also referred to by critics of writing in networked and programmable media. Like Cramer, she discusses the OuLiPo (*Ouvroir de Littérature Potentiel*), the working group inspired and once led by Raymond Queneau, which is, perhaps, the primary reference for literary projects which are explicitly concerned with the application of algorithmic procedures, arbitrary constraint, generative or potential literature, and relatively early experimentation with the use of software. In doing so, she directly confronts the “repression” of “numerical,” generative procedures in poetry and poetics and turns to the work of John Cage as a cross-media figurehead. While only a minor aspect of his oeuvre, as compared with his major contribution to the art of (musical) sound, Cage’s mesostic texts, especially his “reading through” of Pound, Joyce, and others, stitch to together a range of concerns—inter-media art, procedural composition, the rereading (and implicit deconstruction) of the High Modernists—which are highly relevant to both contemporary poetics and writing in networked and programmable media. If Cage’s work is recalled in the context of the Fluxus movement, with which he is associated, then its relevance widens and deepens. Fluxus is a model of performative art practice, including explicitly literary practice, where the record of inscription is problematized (the work is an event, or the publication of a set of materials which must be manipulated by the reader/user), and where the presence/absence dialectic has been side-stepped by representations which may literally absent an artist-author. Perloff does not discuss Fluxus at length and so misses the opportunity to reassess and contextualize work by two of the most important practitioners of the “(numerical) repressed,” Emmett Williams and Jackson Mac Low, both of whom deserve serious study as precursors if not “anticipatory plagiarists” of writing in networked and programmable media.¹⁸ Fluxus also provides a historical, critical link to the traditions of visual and concrete poetics, which are discussed in Perloff’s account, particularly relevant work by Steve McCaffery and Joanna Drucker. The materiality of this work, considered as language art, visibly demonstrates a radical engagement with linguistic media and a requirement for the reader to engage with the codes—textual, rhetorical, paratextual, visual, and so on—by and of which the work is constituted.

If such prior work remains inadequately acknowledged in the discussion and reassessment of “codework,” this may be, in part, simply because the traces of its inscriptions are captured and recorded in the “durable material substrates” of print culture. Whereas Lacan’s “floating signification” is read as an analytic metaphor, applied to language borne by a delivery media—print—on which the signs of the interface texts literally “rest” (where they have been impressed) or, at best, “interleave” (they do not “float”), we read

Hayles's "flickering signifiers," as she encourages us to do so, as literally "flickering," and constituent, as such, of text which has become "screenic." As such, it seems to exist elsewhere, not on the page but through the window of the screen in the informatic realm.¹⁹ Undoubtedly, there are clear and historical distinctions of delivery media for text. Nonetheless, we must be careful to distinguish the effects of delivery media on signification and affect from those produced by shifts in the compositional media, and there is great congruence between the approach to compositional media of certain print-based writers, such as those discussed by Perloff, for example, and the potential use of compositional media which is suggested by new media, that is, new delivery media. This potential of text- and language-making is not necessarily engaged simply because new delivery media happen to be employed.²⁰ The locus classicus for a multilayered, multilevel code-inflected writing and reading is, of course, Barthes's *S/Z*, as Hayles explicitly acknowledges.²¹ *S/Z* was concerned with a short story programmed in "a persistent material substrate," but Barthes was nonetheless able to demonstrate the potential for an iterative flickering of hermeneutic attention across structured linguistic codes, implying, I would argue, perfectly adequate complexity, mobility, and programmability in the compositional media. Barthes's essay, after all, was not a demand for new media but a (re) call to new or latent ways of reading and writing.

We turn, nonetheless, to examples of what Cramer calls "codework." Cramer cites, among others, some of those writers in networked and programmable media whose work I, too, would consider in this context: Mez, Talan Memmott, Alan Sondheim, Jodi. Leaving Jodi to one side for the moment, these are all artists who both work with code and make coded, programmatological objects. They are particularly known and notable for working code and code elements into what we might call the "interface text," the words which are available to be read by the human audiences they address.²² The result is a language which seems to be—depending on your perspective—enlivened or contaminated by code. In the rhetoric of this type of artistic production, contamination or infection (see Cramer as quoted above and Hayles below) is more likely to be the requisite association since transgression of the deconstructed systems of control is an implicit aspect of the aesthetic agenda. For the moment, however, we are more concerned with certain formal and material characteristics of the resulting language.

The language certainly reveals code and code elements, but what code does it reveal? What does it tell a code-naïve reader about the characteristics and the power of code? Is it, indeed, still code at all? At what level does it sit in the chained hierarchies of flickering signification? Has it been incorporated into the "interface text" in a way which reflects its hierarchical origin, if it has one? Only if these and other questions can be given answers which specify how and why code is sampled in this writing would be it "codework" in a strong sense. (Perhaps we should reserve Mez's "code

wurk” for the weaker sense of code-contaminated language.) In the case of all of these writers (we’ll come to Jodi shortly), the code embedded in the interface text has ceased to be operative or even potentially operative. It is “broken” in the now-familiar programmer’s jargon. The breakdown of its operations eliminates one aspect of its proposed aesthetic value and allure, its native performative efficacy (which Cramer identified as a final throwaway without actually demonstrating or elaborating): the power of code to change the behavior of a system. The code-as-text is more in the way of decoration or rhetorical flourish, the baroque euphuism of new media. This is not to say that—as part of the interface text—it may not generate important significance and affect. In particular, the address of this type of intermixed, contaminated language is often concerned—as shown in the work of all of these writers—with issues of identity, gender, subjectivity, technology, technoscience, and the mutating and mutable influence they bring to bear on human lives and on human-human and human-machine relationships.

For the moment, however, we are more concerned with certain formal and material characteristics of the resulting language. In a recent conference paper, Hayles has discussed the language of Memmott’s *From Lexia to Perplexia* in terms of pidgins and creoles. “In this work the human face and body are re-coded with tags in a pidgin that we might call, rather than hypertext markup language, human markup language. Code erupts through the surface of the screenic text, infecting English with programming languages and resulting in a creole discourse that bespeaks an origin always already permeated by digital technologies.”²³ Similarly, Mez has characterized her textual production as written in a new “language/code system” which she calls “mezangelle.” It is perhaps unfair to treat what may be metaphoric usages as literal; however, I believe this use of pidgin and creole is, in particular, a significant misdirection. A pidgin is a full-blown language, albeit arising from the encounter and hybridization of two or more existing languages; a creole is a pidgin which has become a first language for speakers raised by previous generations who have created or used a pidgin. The point here is that, in the case of a pidgin, the elements which combine to generate new language are commensurate—linguistic material is not simply being injected from one hierarchically and functionally distinct or programmatologically-operative symbolic sub-system, which is subsumed within a full-blown culture-bearing system of human language use into another. The creation of a pidgin is, furthermore, the result of interactions by commensurate entities, that is, humans. In the code-as-text which we have seen to date—in the texts of a reveal code aesthetic—human-specified code elements and segments are, typically, incorporated into what I have called the “interface text” which is unambiguously and by definition an instance of some human-readable language. It may be contaminated, jargonized, disrupted language, but it is not a new language, not (yet) evidence for the invasion of an empire

of machinic colonizers whose demands of trade and interaction require the creation of a pidgin by economically and linguistically disempowered human users.²⁴ The codeworks currently available to us extend, infect, and enhance natural language, but they do not create, for example, Code Pidgin English.²⁵

The code has ceased to function as code. The resulting text pretends an ambiguous address: at once to human reader and to machinic processor, but both human and machine must read the code as part of human discourse. We would not try to compile the code in the interface texts of Memmott, Mez, or Sondheim. Nonetheless, this pretended ambiguity of address remains important to the aesthetics of this work. It assumes or encourages an investment on the part of its readers in the technology of new media and, especially, in the dissemination of textual art in new media. Thus, the experiences of the reader in these worlds can be brought to bear on their reading of the codework and they can appreciate, through more-or-less traditional hermeneutic procedures, the references and allusions to technology, technoscience, and the issues with which they confront us. However, I would argue, if this pretended ambiguity of address exhausts the aesthetics and politics of a project (I am not saying that it does in any of these cases), then it leaves open questions of the work's affect and significance when compared, for example, with previous poetic work in more durable material and linguistic substrates, some of which has been cited above.²⁶

The work of Sondheim needs to be singled out, in terms of practice and form, since his use of code is well integrated into a long-term and wide-ranging language art project. The print-media version of *Jennifer*, for example, reads more in the tradition of innovative or avant-garde writing than as subsumed within codework or a reveal code aesthetic.²⁷ Most of the texts in this selection are manipulated language, but often using procedures which are not directly related to codes and processing. Thus, while his overt subject matter—mediated gender and sexuality, explicitly inflected by computing and technoscience—and his explicitly chosen media keep him immediately allied with codeworking colleagues, Sondheim's work must also be read against earlier and contemporary writers working within or with a sense of the formally and aesthetically innovative traditions of poetics, and not only the poetics which intersects with Burroughs and Acker.²⁸

In the necessity to read the work in both a programmatological context and in the broader context of innovative writing—though in this sense only—Sondheim's engagement rhymes momentarily with that of Loss Pequeño Glazier. Glazier and his work represent a literal and explicit embodiment of "a set of relationships—relationships constituted by artistic practice—between a newly problematized linguistic materiality and represented content." Glazier has produced a body of work, grounded in an existing writing practice, which has covered a wide range of potential forms for digital poetics and he has, moreover, documented and analyzed this trajectory

in a series of critical contributions.²⁹ Glazier's work is characterized by his use of code and the language of code. In this, I believe, he affords himself significant ironies. He writes, for example:

The language you are breathing becomes the language you think ... These are not mere metaphors but new procedures for writing. How could it be simpler? Why don't we all think in UNIX? If we do, these ideas are a file: I am chmoding this file for you to have read, write, and execute permission—and please grep what you need from this! What I am saying is that innovative poetry itself is best suited to grep how technology factors language and how this technology, writing, and production are as inseparable as Larry, Moe, and Curly Java.³⁰

This is discursive prose of a kind, but it is infected or contaminated by both code and poetry. Glazier doesn't think in UNIX, nor would he ever wish to do so. But his language is not "mere metaphor" (poetry is not metaphor) it is centered on language-making (what poetry is), and it demands a poetic practice which is alive to new procedures and new potential and which is sensitive to the changes this practice produces in the materiality of the language itself. Apart from its engagement with code and coding, Glazier's work is also characterized by its bilingualism, or rather the multilingualism of "America" in the sense of a Latin America which exists as historical and political soul and shadow throughout, arguably, the greater part of the United States. I raise this point to highlight distinctions in the way we may choose to consider the non-standard English material in Glazier's (and others') texts, while recalling Hayles's metaphoric analysis via "pidgin" and "creole." In a Glazier text, there is a use of English intensified by an address to the materiality of language. There is the incorporation—in a strong sense, sometime within the body of a word—of linguistic material from Spanish and other languages, especially those indigenous to Mexico. There is a similar incorporation of linguistic material from code and from computing jargons.³¹ But whereas the use of other natural language material evokes significance and affect which is commensurate with human concerns—personal, political, social, and cultural history, and so on—the use of "codewords" evokes other concerns, closer to questions of technology and the technology of language. Glazier would rather think in Nahuatl than in UNIX, but in practice he prefers to think in P=O=E=T=R=Y.

Jodi takes us to another point in the textonomy of code-as-text, a relatively extreme position where code-as-text is, perhaps, all there is.³² It is difficult to say anything hard and fast in terms of more or less conventional criticism about a site which is hardly ever the same on successive visits. Instead, I want to refer to what I remember of a visit in which a dynamic HTML- and JavaScript-mediated experience proved to have been delivered by HTML source which was, itself, a work of ASCII art.³³ Here, the actual

code is a text, an artistic text. However, the code is not, in this instance, working code (at least not “hard-working,” shall we say). It is comprised of code segments which are ignored in the browser’s interpretation and rendering of the HTML. The syntax of this markup language is particularly easy to manipulate or appropriate in this way because comments—ignored by any interpreter, by definition—may be extensive and because interpreters, browsers in this case, are, typically, programmed to ignore any <tagged> thing which they cannot render. The code works, but it is not all working code. Again, it represents only a pretended ambiguity of address: its primary structures of signification were never meant for a machine or a machinic process.

I, too, have made a few “codeworks” of a not dissimilar kind. By extracting and manipulating segments of the close-to-natural-language, very-high-level, interpreted programming language, HyperTalk, I was able to make human-readable texts which are also segments of interpretable, working code:

```

on write
  repeat twice
    do “global” & characteristics
  end repeat
  repeat with programmers = one to always
    if touching then
      put essential into invariance
    else
      put the round of simplicity * engineering/synchronicity + one into
        invariance
    end if
    if invariance is greater than the random of engineering and not
      categorical then
      put ideals + one into media
      if subversive then
        put false into subversive
      end if
      if media is greater than instantiation then
        put one into media
      end if
    else
      put the inscription of conjunctions + one into media
    end if
    if categorical then put false into categorical
    put media into ideals
    put word media of field “text” of card understanding & “ ,text” into
      potential
  end repeat

```

```

if the mouse is down then
  put conjunctions into potential
  put potential into card field agents
  put true into encoded
  exit repeat
end if
inflect
wait manipulation
put potential into conjunctions
put ideals into world
if performed then put false into performed
if programmers are greater than control and media & comma is in field
  computer of card understanding & "text" then exit repeat
end repeat
if not encoded and not touching then
  if ideals are developed then wait five seconds
  lock screen
  put empty into card field agents
  put empty into card field system
  do "unlock screen with dissolve" & fantasies
end if
end write34

```

This text has genuinely ambiguous address—to a HyperTalk interpreter and to human readers. It could (and does, in some versions of the software) alter the behavior of a system, when included as one routine in a text generator. Its address to human concerns is clearly ludic and, perhaps, pretends more than it delivers in terms of significance and affect, but at least we can say, with little qualification, that this code is the text.

But where is such a codetext going, in terms, for instance, of its formal and rhetorical characteristics, in terms of its specific materiality? As a text—let us provisionally call it a poetic text in the sense of a text which implies some trial of language—which is addressed to human readers, it has distinct limitations, constraints which disallow or compromise its engagement with broader and more traditional concerns or sources of cultural value. Nonetheless, for me, it suggests new or newly highlighted rhetorical strategies which are specific to the materiality of language in networked and programmable media. For the moment, I will identify two such rhetorical fields of play: (1) the direct confrontation of strict logical-syntactic symbolic composition (programming *per se*) with natural language syntax and argument, and (2) what I think of as a potential “aesthetics of compilation”: the creation of linguistic or symbolic constructs which are designed, for example, to be read in one mode of address and at one level of code in a chained hierarchy of symbolic systems, while simultaneously

intended for compilation into a systematically related code at a different level within the hierarchy, with a different mode of address.

The first of these rhetorical fields represents an age-old and persistent problem: that, indeed, of logic vs. rhetoric, although recast in specific proliferating instances of logic-as-literature in new media. There is no time or space in this shortly closing essay to take this on.³⁵ Compilation in language and literature, however, directly addresses the interrelation of code and text; and it seems to me to be a good example of a rhetorical concept, hitherto of little use where literary objects were inscribed in persistent and durable material substrates, but of great potential in a literature constituted by flickering signification. Texts are already being made to be compiled, decompiled, recompiled, and so on.

I may have seemed to be arguing with flickering signification, by giving examples of writing which appeared to demonstrate its structures of code and text in systematically linked hierarchies, and then showing that these structures were collapsed in many of the examples to hand. In fact, I believe that the structures Hayles identifies and characterizes are clearly operative in writing in networked and programmable media, just as they are operative in certain types of innovative poetic practice. The writing of flickering signification does, indeed, contribute to changes in the body of literature, the literary corpus, both its “material substrate” and its “codes of representation.” However, rather than the intermixing and mutual contamination of code and text, we require not only a maintenance and practical understanding of the distinction between code and text, we need at least the same range and fineness of distinction as that which exist between all the levels of programmatological languages and codes. The “power”—including any affect and significance discoverable by interpretation—which such structures of signification generate is dependent on these distinctions, and on the compilation procedures, which I propose as rhetorical, by which they are systematically related. This “power” is also, typically, in this context, dependent on the concealment—the hidden working—of the code which is thus allowed to serve its function as program, to generate the text and offer it—iteratively, repeatedly, indeterminately—for instances of potential performance, including the familiar performance of reading.

In her discussion of the flickering signifier and its filial relation to the floating precursor of Lacan, mutation is the resultant process of a dialectic implied by a new structure of signification, as parallel term for castration in Lacan’s analysis. Mutation is “a decisive event in the psycholinguistics of information. Mutation is the catastrophe in the pattern/randomness dialectic analogous to castration in the presence/absence dialectic.”³⁶ Mutation—which evokes change, movement, the kinetic potential of text in new media, the mimetic engagement of literature with the culture of human time—is indeed a generative catastrophe for “literature” in the sense of immutable, authoritative corpus. As writing in networked and programmable media,

language and literature mutate over time and as time-based art, according to programs of coded texts which are embedded and concealed in their structures of flickering signification. For the code to function as generator, as programmaton, as manipulator of the text, it must, typically, be a distinct part of the global textual system; it must be possible to recompile the codes as operative procedures, as aspects of live-art textual practice. The code is not the text.