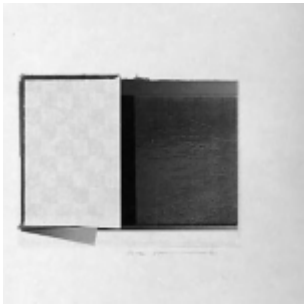
Interferences: [Net.Writing] and the Practice of Codework

electronicbookreview.com/essay/interferences-net-writing-and-the-practice-of-codework

January 31, 2012

Rita Raley
09-08-2002

thread: electropoetics



Rita Raley on the varieties of code/text, as discovered in the object-oriented aesthetic of Mez, Ted Warnell, Talan Memmott, Alan Sondheim, and others.

6.) Code. Use the computer. It's not a television. *Excerpted from Lewis Lacook's posting of his "rules" for net.art to the Webartery mailing list, reposted to the Nettime list (February 14, 2002).*

Codework refers to the use of the contemporary idiolect of the computer and computing processes in digital media experimental writing, or [net.writing]. Some of the prominent practitioners include Alan Sondheim, who has given the practice and genre its name, Mez (Mary-Anne Breeze), Talan Memmott, Ted Warnell, Brian Lennon, and John Cayley. These writers also use different terms to refer to work: Mez composes in a neologistic "net.wurked" language that she has termed m[ez]ang.elle; Memmott uses the term "rich.lit"; Warnell names some of his JavaScript poems "codepoetry"; Lennon refers to "digital visual poetics"; and Cayley produces algorithmic, generative texts, or "programmable poetry." Writers and artists who have taken up the general practice of codework heed the mandate - "use the computer; it is not a television" - and strive to foreground and theorize the relations between interface and machine and so reflect on the networked environment that constitutes and is constituted by a digital text. The precise techniques vary, but the general result is a text-object or a text-event that emphasizes its own programming, mechanism, and materiality.

Picture e.e. cummings, bp Nichol, or Emmett Williams upgrading their medium and exchanging their typewriter keys for the units of programming languages, and the result would in part resemble the contemporary mode of experimental writing and net.art called "codework." So, Mez, for example, expressly strives

2 uze computer kode kon.[e]vent.ionz spliced with irc emoticons and
ab[scess]breviations....
2 spout punctu[rez]ationz reappropri.[s]ated in2 sentence schematics....
2 illustrate the x.pansion of software potentialities of co:d][iscours][e
in an environment x.clusively reliant on it. *Mez, The Art of M[ez]ang.elle.ing. Also, see her introductory description to the data][h!][bleeding texts, which was short-listed for an ELO writing award (2001).*

In practical terms, the difference between Emmett Williams, "Meditation No. 1" or e.e. cummings's, "r-p-o-p-h-e-s-s-a-g-r" and a net.wurked text by Mez, Warnell, or Memmott is the difference between the typewriter and the computer, the difference of what the medium allows. *For another manifesto, see Blood Puppets and all. Though not individually noted as such, all excerpts from Mez's codework quoted in this essay are to be read as "[sic]."*

Broadly, codework makes exterior the interior workings of the computer. One formal purpose is to bring the function and code of the computer to a kind of visibility. That is, to illuminate the many layers of code - the tower of programming languages that underlies the representation of natural languages on the screen. For all of the differences among particular instances or events of codework, they all incorporate elements of code, whether executable or not. Code appears in the text, then, in whole or in part, in the form of a functioning script, an operator, and/or a static symbol.

As John Cayley's essay in this ebr release will indicate, there is a reiterative component to my initial description of codework. Cayley, Sondheim, Memmott, and others have outlined, discussed, and queried this branch of new media experimental writing, in forms ranging from the short gallery review to the listserv posting and conference presentation. We are, however, only now approaching a second wave of critical discourse on the subject, and some of the descriptive foundation still needs to be articulated as we move on to consider some of the more important questions and issues raised by the practice: the relations between natural and programming languages; the link between contemporary codework poetics and earlier, "avant-garde," found-object artistic practices; the schism between formal aestheticism and socio-cultural politics; and the question of cognitive transformation.

Codework participates in a larger movement that we might call the "art of code," in which the code used to produce the work seems to infiltrate the surface, the former domain only of natural languages and numeric elements. For example, on her recently released album, Head Slash Bauch, Antye Greie-Fuchs (AGF) reads lines of code deconstructed into syllables. [Click here for a sample] She intermingles English, German, elements of markup languages, and the language of code, such that "layer readme slash p ID blockquote slash layer" resonates aurally, symbolically, and technically. The art of code is not limited to any one particular media environment, and its use as a medium reflects the expanding symbolic database that is at once artistic and communicative. So, too, does it reflect the changes in natural and machine languages and the changes in our evaluation of both. That these encounters with code, however fleeting, partial, or incomplete, should necessarily result not only in revised cultural forms and practices, but also in anxieties about intrusion, contamination, and uncontrollability, is evident in Jessica Loseby's illustrative net.art work Code Scares Me. Loseby's installation uses Flash actionscripting to "domesticate" the monstrosity of code and directly thematize the fear of invisible and unknowable code, disturbing because she considers it to be "a language that is both hidden and alien to me."

For Loseby, code is initially understandable only in terms of impenetrable darkness. It lurks beneath the surface of the text, but it is not in direct dialogue with that text: it is read and yet not read at the same time. The fear, further, is that code is autopoietic and capable of eluding the artist's attempts to domesticate it and bring it into order: "I imagine it unlocking itself in my absence," she notes, conjuring a vision of code compiling itself, generating its own output, and moving toward self-organization. In this instance, code is `scary' because it is both unknown (`foreign') and known (understood to have emergent properties).

An art of code, though, would almost necessarily suggest that code can be beautiful instead of alienating. For instance, Geoff Cox, Alex McLean, and Adrian Ward argue at length for "The Aesthetics of Generative Code" in a paper that suggests that the beauty of code lies in its performance, functionality, and execution. *I have elsewhere explored the question of the theoretical differences between analog and digital text; see "Reveal Codes: Hypertext and Performance," Postmodern Culture 12:1 (September 2001).* This kind of affect for the elegance of code is shared by Ellen Ullman, though she holds out as well that elegance is linked to operation. So, too, does an aesthetic appreciation of code for these programmers require a knowledge of its operation, such that form and function (execution) are thought together. Ted Warnell's visual, computational poem, "Lascaux.Symbol.ic" serves as a particularly apposite illustration of this fusion, displaying as it does a JavaScript with both operational and visual style. Similarly, in "If () then ()", Jutta Steidl claims that code has and is capable of expressing an aesthetic, but she questions whether programming languages can express the inexpressible and whether they are capable of speaking to, and generating, literary affect. By emphasizing the affective deficiencies of code, she maintains an ontological distinction between programming language and literary language, even as she equates some of the historical properties of the literary, e.g. `beauty,' with the functionality of code.

The art of code and the practice of codework has a socio-cultural history, more specifically origins within origins, and it is not limited to our contemporary moment. Its genealogy includes many instances of codes used as a medium for art, including Oulipo's Algol code poems and the use of computer instructions in their texts; the long-term tradition of generative aesthetics and poetic programming, such as Tristan Tzara's `algorithm' for Dadaist composition (including in a similar vein La Monte Young's and John Cage's instructional scores); ASCII art; the composition of Quines; and Perl poems. *Also see Loss Pequeño Glazier's short discussion of the beauty of code in Digital Poetics: The Making of E-Poetries (Tuscaloosa: University of Alabama Press, 2002), 103-110. reviewed in ebr by Brandon Barr*

There are other analogues: though not limited to an online environment, the formal, aesthetic, and political principles of codework as a general category are echoed throughout, and indeed informed by, net.art discourse: the two often have a common basis in hacker culture; Open Source advocacy; anti-corporatist politics; authorial and

textual self-reflexivity; and software technics. Such a broad shared platform suggests that net.art and the information arts, and not the Eastgate and Brown schools of hypertext, provide the socio-cultural, historical, and textual context for the branch of experimental media writing classified as codework. But codework is not a homogenous, monolithic, or pure genre. In his definitive introduction to the practice, and to a special issue of American Book Review, Alan Sondheim has carefully outlined a taxonomy of three types of codework: "Works using the syntactical interplay of surface language"; "Works in which submerged content has modified the surface language"; "Works in which the submerged code is emergent content."

The last, which has found a fairly wide-spread audience in net.art and digital art circles, exhibits a pronounced aesthetics of destruction and failure. In this branch of codework, the buried, or deep, code, is text and content, and because this code occasionally has damaging effects - varying in intensity and seriousness - the practice associated with it is sometimes classified as virus art. The Digital Craft organization has recently issued an exhibition catalogue on the "I Love You" virus that explores the possibilities of considering the programmers of this and other computer viruses as artists and linking them to the poetic appropriators of virus code active in net.art and digital poetry circles for the last five years. *For a more extensive and detailed account of the precursors to codework, see Florian Cramer, "Program Code Poetry."* The net.art team Jodi might be understood as codeworkers in a similar fashion, with the obvious difference that Jodi's projects are not contagious or crippling in the way that a computer virus would be. *Florian Cramer also comments on "virus art" and notes that "computer viruses like Melissa and I LOVE YOU, small bits of text written in computer control code, strike me as perhaps the most dense and interesting examples of contemporary literature in the Internet." See Combinatory Poetry and Literature in the Internet (4) and Language, A Virus? Alan Liu's book, The Laws of Cool: The Culture of Information (Stanford UP), concludes with an extended discussion of the aesthetics of destruction and virus art.* This type of codework, then, concerns itself with troubling the distinction between form and content, between surface and depth, such that one generally has to look at the source code to discern the content, functionality, even "meaning" of the work. To understand this particular practice of codework, we might take a general claim about all hypertext made by Mark Amerika and make it more specific: creative content and the source code, he says, are "one and the same thing" (Hypertextual Consciousness).

Jodi's work leads us toward what I take to be the most practically useful heuristic for critical investigation: a binary structure for codework that draws a distinction between code that is operational and has depth and code that is isolated on the surface of a text. *For a discussion of net art and its "aesthetics of failure," with a specific focus on spectatorship, see Michele White, "The Aesthetic of Failure: Net Art Gone Wrong," Angelaki 7:1 (April 2002), 173-93. Of relevance here as well is N. Katherine Hayles's analysis of print and digital textuality in terms of surface and depth, respectively; see, for example, "Print is Flat, Code is Deep: The Importance of Media-Specific Analysis," Poetics Today (Fall 2001).* The codework that has depth exhibits what Cayley will call an "aesthetics of compilation," addresses the machine, and therefore "works," wields a performative power not equaled by the codework that treats code as a found object, a

ready-made that operates as a static work of art. For Cayley, the performative power of the text authentically and genuinely resides with operable code: the difference is that between Roland Barthes's theorization of the potentiality of language and the potentiality of an algorithmic poem, which changes the system in a materially visible way. That is, the codework that has depth, that is pure rather than mere "decoration or rhetorical flourish," signifies within the realms of both natural and programming languages; it can continuously function and be legible within both systems; and it is capable of altering either one. No mere game of techno-cultural reference or "language of the tribe," as TS Eliot might say, this practice of codework differs structurally, metaphysically, and practically from the codework that incorporates static, non-functional elements of code into the surface, or "Interface text." Specifically, this working code has a "genuine" rather than "pretended ambiguity" of address; it is simultaneously addressed, in other words, to the human and to the machine.

Cayley also powerfully critiques the critical tendency to refer to the intermingling of natural language and codes as a creolization, but this is not to discount the linguistic changes signaled by the codework of the surface. The codework texts that remix natural language (usually English) and programming languages result in a kind of hybridized, electronic English, a language not simply suggestive of digital and network culture but a language of the computer and computing processes. When English is filtered through the languages of the machine something like this type of codework emerges; it is an English that has been manipulated and encrypted into an electronic computer-speak. Florian Cramer terms this hybrid of natural and machine languages "post-combinatory," suggesting both a fusion and an integration on the one hand, and a separation, or an incomplete mixture on the other ("CP" 5). This emergent language is ubiquitous within contemporary advertising and mass media: it is the idiolect of mobile phones, pagers, instant messengers, and chat settings, even appearing in a recent back-page New Yorker cartoon by Roz Chast (February 4, 2002), "The I.M.s of Romeo and Juliet," which updated Shakespeare to a scene of bedroom-to-bedroom Internet messaging, complete with the abbreviating and punctuating elements of much current codework. The basic semantic formula includes these abbreviating and punctuating elements: substituting numbers for letters; n = `in' and `and'; use of the dot, brackets, hyphens, slashes, and the doublepipe. One can only assume, too, that this type of codework will continue to incorporate the alphabetic, non-alphabetic and metacharacters of programming languages, such as the "??" ("hookhook") operator, as they move into widespread use.

And, as the forms of digital writing and art continue to change in relation to production environments and programming languages, it is interesting to speculate whether increasing institutionalization will mean that these high-level machine languages will be sanctioned for study as a "foreign" language within the humanities. The different kinds of institutional training programs developing now, such as those at UC Santa Barbara, UCLA, and Brown in the U.S., suggest that we will indeed see programming languages included within degree certification programs in literature and the arts at the undergraduate and graduate levels. At the moment, however, what we see in most codework writing and art practices is less code per se than the language of code: codework that integrates elements of code into natural languages and brings code to the

surface as a medium for literary, artistic, and experimental composition. The codework practice of "netwurker" Mez, which again involves the use of a made-up code language as a mode of artistic composition and everyday communication, is paradigmatic. *An overview of Mez's work can be found in her recent JavaMuseum solo show (February 2002), commemorating her nomination as "Java Artist of the Year 2001." For a sample of the critical analysis of her work, see LaCook; Beaubien; Esdaile; Beiguelman; Stephanie Strickland; and Rossitza Daskalova, "Language Transformed by the Machine," CIAC Magazine 13. Also, Mez keeps an archive of clippings at http://www.hotkey.net.au/~netwurker/nav.htm.*

Mez's medium is a neologistic net.wurked language that she has termed m[ez]ang.elle, which has at least two narratives of origination. One is almost hallucinatory and never presented in straight English. In 1995, while working on an HTML document _cutting spacez," she realized that actively networked communicative circuits produce mixed and entangled data streams: "jumping fromme one terminal 2 the next//running three chat-rooms at once via three different terminal [behavior]z sew as 2 opt.tim[id no lonah]ize the time d-layzm chairz blurring b-tween as the monitorz flashed fiction wurdz that [k]neededObleed.ed e.vent[ingz]ualli into the cutting spaces doccoO." *See her rhizome.org interview with Josephine Bosma, February 1, 2000. She elsewhere terms this process the "de/reconstruction of language" but, in that this is a fairly broad statement, I am not convinced that this is the most useful description of her work. See her interview with Streetpress.* The other narrative of origination is more concrete: mezangelle emerged from Mez's borrowing and manipulation of text from email lists and chat settings and specifically from a lengthy email exchange with Matt Hoessli from the CADRE Institute on the 7-11 mailing list, beginning in 1996 and continuing with experiments with members of the Webartery and trAce online writing communities.

Though her codework has "][r][e.volved" and become more visually complex in its presentation with Flash, shockwave, and JavaScript, her methodology remains basically consistent: she continues to work with text, frequently taking portions of a network communicative exchange, and then mangling them with machine language elements such as brackets, colons, slashes, hyphens, and the double pipe (||). One of Mez's techniques - one among many she shares with NN/Antiorp and other codeworkers - is to replace letters with symbols and with other letters ("2 4m a text"), in a colloquial style associated in a general way with IM and musical forms like hip-hop and urban youth culture (for instance, `z' substituting for `s' = `boyz'). Incorporating elements of emergent idiolect of the computer, computer languages, and digital culture, she also makes heavy use of the dot, as a connector that groups together words or partial words; to name hierarchies, as with domain names; to form abbreviations and word combinations; and to separate an object from its properties, as with OO programming conventions. The result is a hybridized neologistic language rich with semantic and poetic possibilities. With some graphical and lexical consideration, then, m[ez]ang.elle - with its play on `elle,' `angel,' `mangle,' and `angle' - suggests a pointedly feminist aesthetics and praxis of linguistic mutilation. *In this respect, as well as with their shared concern with the bio-politics of the body, there is a striking resemblance between Mez's aesthetics and those of noted hypertext author Shelley Jackson. See Jackson's website, Ineradicable Stain.* After

reconfiguring the data elements of her source texts, she directly and indirectly re-channels them to the forums from which they emerged, broadcasting them either through listserv posting or through web-based net.art installations. As her manifestos and statements of artistic principles would suggest, this codework language is something of a formal experiment, but it also reflects on the medium and its artistic potential.

That this "net.wurked" component of her artistic practice should be understood as axiomatic is evident from Mez's frequent and reiterative email postings and statements of methodological principle on the subject. For example, Mez notes in the artist's statement accompanying her solo show on Java Museum that her writing has at its base an entire archive of ongoing, collaborative, and "multifarious....directed email/irc exchanges and performances." Further, her list-based commentary on The Art of M[ez]ang.elle.ing stresses that one of her artistic aims is "2 network 2 the hilt N create de[e]pen.den[ting]cies on email lizts for the wurkz dis.purse.all." She introduces her critically acclaimed data][h!][bleeding texts by noting that ".these t.ex][e][ts r _code wurk_ remnants d-voted to the dispersal of writing that has been n.spired and mutated according 2 the dynamics of an active network." And, in the introduction to her curated gallery exhibition at Incubation 2002, she describes her technique as "largely dependent upon an electronic method of production that is exclusive to a networked environment, and shares some characteristics with the very format that houses it [such as n- tegration of email/Web browser/IRC jargon and stylistic blueprints."

Because codework participates in a long-term discourse of referential layering and linguistic wordplay, Mez's techniques have also been preceded by experimental, "avant-garde" Language poetics, concrete poetry, and visual poetry. Although the symbolic database and mode of distribution differ, in both can one find a use of brackets, white space, punctuation, and techniques of excision and negation that is at once excessive and indeterminate. We can, however, again speak of the difference or particularity of Mez's theoretical and practical aims in relation to her poetic precursors, particularly with respect to the mechanism it uses for composition and transmission - its context, and her relationship to that context. The sheer ubiquity and viral spread of both her codework texts and her codework practice echoes back to the very networked environment in which it is situated.

This insistence on the production and delivery environment of her mezangelled texts brings to our attention the necessarily collaborative nature of codework, appropriated as much of it is from listservs and chat settings. Mez and Alan Sondheim, for example, not only build codework texts from communicative exchanges but they also incorporate the responses of their audiences and fold them back into their codework texts, in a continuation of the collaborative production of the text and a fulfillment of the logic of email itself. The reply function, in other words, is coded into a text that is distributed across networks. In this sense, the relationship between codework-text (system) and its generative field and ultimate audience (online environment of chat settings and listservs) could also be understood in terms of feedback, whereby system and environment are both altered as a result of their interaction. *Feedback, Principia Cybernetica Web.*

The writing of Mez and other codeworkers compels an extended technical dissection, or a line-by-line parsing that would explain the significance of her use of each and every element of different programming languages such as Unix, Perl, and Java (she does not tend to borrow from machine or assembly code but rather from the higher-level languages that are closer to English in structure and vocabulary). However, the fundamentals of that kind of analytical-tutorial work have already been established. Florian Cramer, one of codework's most prolific and important critics, has circulated intricate and extended annotations of one of Mez's "wurks," _Viro.Logic Condition][ing][1.1_, and its thematics of the virus, infection, and the mutual contamination of the organism and the computer. *Cramer text sent by mez breeze in a personal email, "n.sights & n.][elec][troductions" (June 14, 2002).* Some of his most illuminating commentaries consider the significance, within this specific text, of Mez's use of square brackets; Perl header lines "::"; and the Unix commandline double pipe "||" - the logical `or' condition - all of which open up multiple grammatical and semantic possibilities within a "line" of mezangelled poetic prose. A rigorously formalist analysis akin to Cramer's is often not equally revelatory or satisfactory with Mez's current work. In this sense there is a difference between her early and more recent practices: "mezangelle" has evolved to become a manner of writing, an idiosyncratic style that bears the signature of its author. It is not generally executable code, and it often does not even allude to the functional component of her code elements, but instead functions as a mode of poetic composition and general communication.

In that mezangelle incorporates the punctuating elements of programming languages, such as the dot and the bracket, it partially revives the old, and well-digested, postmodern (or, "(post)modern") language games played with parentheses and slashes for largely similar general purposes. As Mez notes of her use of the now-expanded database of punctuating "elle-E.ments," the aim is to "condense/dilute/refresh wordage and imagery meanings/established codes/cues of associative meaning/s" (The Art of M[ez]ang.elle.ing). So, if there were a critical exhaustion with the kind of linguistic double plays afforded by a set of parenthesis, Mez counters by using punctuation particular to the apparatus - Talan Memmott calls this set of punctuation "technical ideogrammatics" - such that there is no true single word to destabilize or negate. Witness three signatures from her current Webartery listserv threads: Cur.][O][va.ture;][co][De][e][p.rivation; and app][lick.ation][end.age.

By making widespread use of brackets and periods to split words (somewhat hypertextually) into multiple component parts - letters, prefixes, suffixes, and phonetic elements - Mez's strategy is to disturb, disorient, and defamiliarize, to shift "the units of information and communication from the usual and expected to the cryptic" (rhizome.org interview). Such a cognitive disturbance is also partly facilitated by her use of dynamic text elements, as one can find in her data][h!][bleeding texts. Her repetitive technique - the encoding of text into this new mezangelled form is more or less consistent throughout her various texts - produces patterns of dissolution and reiteration that continue to disorient, even once one has more-or-less mastered the rules to the point of perceiving the patterns at all. For Mez's art and writing, the political purpose of this aesthetic practice is precisely to imagine this kind of interruption and disturbance of the economy of informational transaction.

Given that Mez operates in an expanding field of codeworkers, most of whom also work with "technical ideogrammatics," whether they be executable or not, establishing the specificity of her artistic project should necessarily reach beyond punctuational difference. Such an analytic approach would be thwarted quite quickly: for example, Mez's putatively particular use of bracketed expressions in "m[ez]ang.elle" can be analogized to Talan Memmott's embedding of command structures in his Lexia to Perplexia ("PER[(p)[L(EX)]]ia"). *Deena Larsen briefly links Memmott and Mez in a short essay on e- poetries in Currents*. The components of Mez's work similarly invite comparisons to Netochka Nezvanova's style and artistic practice, which also involves a surface play of code, listserv posting, and ever-changing display signatures, avatars of a sort (Nezvanova performing as NN, Antiorp, Integer, and Fifo). In both can one see an artistic personality produced by and coded into the text. One exemplary and distinctive formal feature of NN's texts is the substituting of the "!" for "I," which in effect gestures toward a negation of the self (See Beatrice Beaubien and David Johnston on the parallels between them). Rather than splitting symbolic hairs and claiming particularity on the basis of NN's use of the operator "!" (= not) or Mez's reiterative use of square brackets, then, it is possible to take a wider look at Mez's placement within the field. First, there are quantitative differences. Her seemingly inexhaustible productivity - surpassed one can imagine only by Alan Sondheim - has directly contributed to her prominence, ubiquity, and the frequency with which critical commentary on her work appears. She has been able to capitalize on her numerous awards and commissions - all prominently detailed on her web site - precisely because she is both networked and extraordinarily skilled at networking. Her institutional currency also derives from her manifestos and from her frequent articulation, theorization, and defense of her project as a project; in that it comes bundled with its own explanation and critical commentary, the mezangelle project legitimates itself as poetic and political praxis.

Mezangelle, too, makes for distinctive sound bites, both because of its quotable form - it is not as easy to extract self-contained lines from Warnell's or Cayley's codework - and because of its thematics, which resonate with theoretical investigations in the field: the organic-machinic divide, infection, gendered subjectivity, data, and networks. Her work is thus situated within the apparatus in two respects: formally and materially on the one hand (through its technologies of inscription), and conceptually on the other. As a whole, her mezangelle project is reflective on the apparatus and on its cultural context. Again, this last description applies also to Memmott's Lexia to Perplexia ("PER[(p)[L(EX)]]ia") and Warnell's Viru2 to name just two examples, but there are generic variations worth noting: she is less of a theorist and visual artist than Memmott and less of a visual poet than Warnell. More important, because mezangelle is communicative, Mez's use of her invented language directly suggests and reflects the material and quotidian linguistic changes produced by the mingling of the elements of natural and programming codes. Codework for Memmott is not communicative in the same way; rather, Memmott is developing a language of network theory - not simply a language of the network - that is informed by Greg Ulmer's articulation of "electracy." Memmott's neologisms come out of a theoretical poetics, so to meditate on the relations between the subject and the apparatus, he coins the terms "Narcisystems" and "exe.termination."

Mez's techniques invite reading as complex combinatorial anagrams, other instances of excess linguistic disassembly and reassembly. Thus, the many semantic and syntactic units that comprise mezangelle are not just fragmented and re-spliced, as with Memmott and others, but also layered, which tends to disrupt the conventionally linear mode of constructing meaning. There are indeed only differences in the language of mezangelle, but there are almost no brakes applied to these differences: her data texts are mobile, fluid, and unstable, and thus continue to bleed, rather than congeal. Instead, the language flies off in many directions, and the invitation is to read beyond and even against the lateral, particularly given the frequent use of puns and homophones. In that Mez also provides translations, sometimes interlinear, of her mezangelled text assemblages, the sense of linguistic layering and inter-reference is intensified, as is the recognition that a linear translation is impossible because any translation of mezangelle must necessarily suppress the aural and visual elements. Any translation must perform a semantic extraction much like one does while reading one of Mez's texts out loud, when it is possible to pick up on the deconstruction of lines of code into syllables and units of noise and breath, but impossible to communicate the visual use of code. As she notes in another statement on her practice: "[sorry no immediate translation possible]." It is impossible to perform an immediate translation and fix a stable meaning, and yet her codework gestures toward this impossibility. To fix on the `plain English' that lies behind the code, or to isolate one lexical or phonetic unit so as to establish a concrete meaning, is to exclude the other units we are asked to apprehend.

In Mez's codework, elements of computer code are thus used to produce a multiplicity of reference and do not themselves usually maintain a lexical function; in this case their signification is visual. So, she uses "net.wurked" language as a referential structure (using syntax of the computer and of programming) but not to actual algorithms or algorithmic processes. In other words, she produces signs of computer code but not a code-language that is machine-executable. This code-language is brought to the surface as a static art artifact rather than as a functional program, such as with the title of one of her recent net.wurk postings to Webartery: *.imp loading[s] (August 27, 2002). She references the tower of machine languages that operate the system - from machine code up to C and Unix - but codework for Mez isolates the screen as surface. Like Minimalist art, Mez's work offers a kind of literalism, in that the screen does not simulate a system beneath or beyond; it is instead all opaque surface. In that she does not generally produce codes that activate operations or processes, but she will refer to process, as in this formulation - "alt[.ctrl.delete]etered" - her codework is technically non-referential and technically non-performative. (Florian Cramer and others have speculated as to whether some of her texts are `mangled' output, but no one has offered evidence of the execution or compilation of her codetexts, failed or successful.)

Cayley's argument for the importance of executability and for the need to understand the difference between technological symbolism and `real' code is apropos: Mez indeed writes verbal art and not operable code. As a point of contrast, the code that generates the thirty-two algorithmically generated poems in Cayley's River Island is addressed to the machine. Cayley has noted that his prioritizing of operationalism and performance derives from his long-term investment in etymology and linguistic history: without an acute

sense of linguistic strata, in this case the tower of programming languages languages, we run the risk of being ahistorical and disregarding the complex semantic and syntactic structures of codes. *Personal conversation with Cayley (July 16, 2002).* Further, his delineation of code and text emphasizes the materiality of the text and challenges the idea that the interface is an isolate or isolatable entity. The strength of his analysis on this point lies in its countering of the idea that hypermedia simply consists of a textual interface. In other words, "writing" in a digital environment consists of both text and code, and programming thus necessarily has a clear literary-poetic element of craft and style. Many of Ted Warnell's works in his "Poem by Nari" series (such as Lascaux.Symbol.ic, Viru2, and code.poetry::executables) would be paradigmatic in that the very script that executes the texts is displayed on the surface. Mez's codework, though, is precisely opposed to the value of functionality, which is anathema to her treatment of code as an object and to her aesthetics of disruption and interference.

As a partial response to Cayley, we might consider the notion of the separation of code and output in relation to Adorno's remarks about music as merely a consequence of the score. To consider music consequential in this way essentially privileges the score - the formal instructions for the production of music - over the music itself. *Florian Cramer briefly discusses musical composers and the shift from classical score to instruction code, particularly in relation to John Cage and La Monte Young, in "Digital Code and Literary Text."* However, codework does not suggest, nor does it need to, that code - the algorithmic score, the instructions that govern and produce a system - itself should be privileged. Rather, codework tries on the whole to move beyond this schism - the code and its `work' or operation - to make something new. It relies on this schism in order to produce its effects, but then there is a mixing, an interfusion, and something other emerges.

What emerges is an object-oriented aesthetic and a textual practice that objectifies the code structure of the mechanism. The language of OO programming invites certain theoretical questions, with particular respect to codework and net.art. First there is the problem of the digital object as object. An object is usually thought as that which can be perceived, whose state and behaviors are somehow visible. What is the relation, then, between the hypermedia object and the found, kinetic, tactile, or otherwise `physical' art object? One answer is that the codework of Mez and Warnell is an object work that does not replicate the executable aspect of code. Elements of the deep running codes of the apparatus are put on the screen as artifacts, not as programs or as functioning programming languages. This branch of codework, then, taps into the found art tradition in that its practitioners take a language designed for a pragmatic function and convert it to a language of art. Code is treated by them as a ready-made in that code is turned away from its functional, pragmatic, programmatic purposes, manipulated and situated in a context where it reveals its poetic potential as an aesthetic object. *Florian Cramer: "instead of constructing program code synthetically, they use readymade computations, take them apart and read their syntax as gendered semantics" ("Concepts, Notations, Software, Art" 7).* Its mechanistic purpose, in other words, is instead converted to the

poetic. Similarly, email and IMs might themselves be considered as pragmatic, functional, ready-made objects converted to aesthetic purpose. These code objects, however, are disturbed, and they disrupt normal channels of reception.

In the tradition of found-art practices (Rauschenberg, for example), codework comments on the shelf life and obsolescence of objects: it presents a recycling, re-use, and re-activation of objects that were oriented toward disintegration instead. In Warnell's Amerika, for example, the framing mechanism of code remains intact. In order for it to function it has to be revealed as code; otherwise it loses its codework quality. So, too, the structure of "Amerika" reminds us that code is a perceivable object. In a paratext to the same poem, Warnell associates code objects with avant-garde modernist aesthetics by drawing on the general anti-aesthetic sensibility displayed in the critical discourse on experimental new media writing. He specifically draws from Mark Amerika's meditations on "hypertextual consciousness," which are broadly constructed in the style and mode of a manifesto directed against the stifling constraints of materialistic commercial culture. That Warnell should make use of that excerpt from Amerika's manifesto that refers to the Dadaist ready-made, and that he should also fashion a code poem entitled Dadastream, is evidence of the embedded link between codework and the ready-made. *The rhetoric of the ready-made also surfaces in Cramer's critical commentary on the codework poetry of Mez, Warnell, and others, whom he describes as having "taken up impulses from Net.art by incorporating ready-made bits and syntax from programming languages, binary machine code, network protocols and markup conventions of interpersonal network communication" ("Combinatory Poetry and Literature in the Internet" 4-5).* The relationship between codework and the found art object, in other words, is not retroactively imposed but discursively situated.

To go further, in such a media context, we are further witness to a confusion and complication of the dichotomy between surface and depth, as Jean Baudrillard has suggested. In "Aesthetic Illusion and Virtual Reality," Baudrillard links the ready-made to the logic of the reality show: in both an object (or, a human being) is extracted from the real and "displaced...on another level to confer on it an undefinable hyperreality" (21). Once "mediumized" and transposed to this other level, stage, or screen, the virtual ready-made performs a live show of its own existence and thus appears as pure appearance. Baudrillard goes on to ascribe this logic of the virtual ready-made to all objects, events, and individuals in our postmodern moment and notes that "all of them might be described in much the same way as Duchamp implicitly categorizes his ready-made object: `It exists, I met it!'" (21).

Codework, like much hypermedia, is a kind of object in that it puts on displays for the user-viewer, whose reactions and responses it then incorporates within its field of performance. In this regard, the object can be imagined as a means to insist upon or instantiate, as it were, the materiality of the medium. The materiality of the digital text has been extensively theorized - notably by Hayles - in terms of the depth model of code and technologies of inscription and mediation, also with regard to the tradition of concrete poetry, and we might add to both a certain understanding of codetext as a found object.

There are links we might then draw between mezangelle and the artistic minimalism of Sol Le Witt or Carl Andre in that she works with found objects, basic visual and verbal elements, and establishes rules for each textual event that vary only slightly from those used before. *The comparison I wish to draw is conceptual rather than physical, and the comparison has its limits, as Alan Sondheim has noted in a personal email (May 28, 2002). There are indeed clear philosophical and material differences - her text-messages are of course not modular, geometrically or symmetrically arranged, nor do they use industrial materials - but the link profitably illuminates the extent to which the codework practices of Mez, NN/Antiorp, and Warnell produces and then operates with variable sets of elements.* Her plans for composition are not outsourced to an art team, but they might very well be in that she essentially constructs semantic and lexical rules, e.g. the substitutive principle that n = `in' and `and.' A kind of formal pattern outlined by the symbolic structures of programming languages, then, underlies many of Mez's "data bleeding texts." Somewhat analogous to Carl Andre's alphabetic-typographic poetic exercises, much of Mez's codework resembles a kind of ABC art in that it suggests elemental and interchangeable units of composition. Andre's "Pope Byron Andre" begins: "a a A A abroad all Amphibious Ass's at at/ board Bug Butterfly buzz." *Poetry 1958-73. Nicholas Serota's comments about Andre's poetry are germane here: "The source and manner are characteristic of Andre's written work: breaking down the existing structures of a text, often found, but sometimes written by himself, analysing and reordering words by such principles as frequency in the text, alphabetic order and number of characters. They are laid out or mapped according to simple visual patterns (e.g. repetition, squares, triangles) and arithmetical (e.g. prime number theorem) principles." Carl Andre (London: Trustees of the Whitechapel Art Gallery, 1978), n.p.* While Andre stressed to Hollis Frampton in 1962 that he strove to reduce a text into "its smallest constituent elements: the isolation of each word," one consequence of which would be "poetry which eliminates the poet," Mez similarly distills text down to a set of operational elements, including meta-, alphabetic, and non-alphabetic characters (12 Dialogues). Like Minimalism, too, Mez works toward the effacement of the singular signature of the author by incorporating one or many authorial avatars into her work (e.g. "ms postmodernism"), by situating the author "Mez" within "mezangelle" as a construction, and by enhancing and embellishing previously composed messages, a tactic that positions her simply as a mediating nodal point, a sysadmin with only partial write permissions. So, too, a basic or plain language always in some sense strives, or presumes to, erase the signature of the user-transmitter, with its attendant fallacy of neutrality and universality.

M[ez]ang.elle tends toward a linguistic minimalism as well. Like the idiolect of mobiles, pagers, IRCs, and IMs, it is laconic and often cryptic to the uninitiated eye. (She names an earlier incarnation of her net.wurk writing as "abbreviated geek speak" in "Non Compos Mentis.") It filters out the non-essential or peripheral `noise' in a communicative exchange (often an email or chat) and distills it down to a set of core or basic elements, often singularly syllabic. Having extended William Strunk's rule 13 on brevity, "Omit Needless Words" to include needless letters as well, Mez is similarly left with space and time usually filled with repetitive units (Elements of Style viii). M[ez]ang.elle is in this

sense "ez." However, it is not concise, and even the principle of compositional simplicity and shortness is reformulated such that "conceptual" is introduced as a bracketed regular expression in one of Mez's many comments on her own method:

<In short, the KISS princ[ess]able has some me[z]rit
<Keep It Simple Stupid.
<:In my case, it should read:
<Keep it [conceptual and] Sizzlingly Short

In the same email interview, Mez remarked further on the succinct, elemental, and compacted quality of mezangelle with particular reference to its medium and the online environment:

Obviously the medium of hypertext lends itself readily to the minimal...minimal in terms of a primary reliance on the most basic elements - text, screen [`doc swapping'] and image. My laconic use of these most base.hic[!] elements [or elle-E,ments as i would say if responding via regular mez/nschine communication channels] is governed largely by the need to condense/dilute.refresh wordage and imagery meanings/established codes/cues of associative meaning/s. ("The Art of M[ez]ang.elle.ing")

But mezangelle and Mez herself demonstrate a significant skepticism about the idea of plain or basic English and the principle of the "basic" in user-interface development: the mandate that the interface be immediately, uniformly, and universally legible for the general user. In this regard, it is worth noting that the designer of the user interface for the operating system of the PalmPilot, Rob Haitani, claims that GUI designers need first and foremost to think in terms of economy and basic communicability as these principles have been taught in American writing courses for much of the twentieth century: "I say if you only read one book to understand handheld user interface, it should be Strunk and White's The Elements of Style " (Information Appliances 94). Mez's own skepticism about the idea of "plain English," however, comes through in her ironic commentary on the number of emails that insist that she "just speak plain English" and start making sense (Streetpress interview). The ideal net.art, Mez notes in a related conversation, "makes me unsure of the very principles that govern the interface/project [and]... throws a reliance on hackneyed dataface terminology out the window" (rhizome.org interview).

Her own mangling of the interface attempts to interrupt and impede, rather than facilitate, the direct conveyance of information and the smooth operation of the circuits of transmission. To obstruct the felicitous transmission of data, she interjects signals such as the punctuating elements of code into her words. Such is it the case, then, that the mania hovering at the edges of her "data bleeding" texts is the mania that comes with a lack of a filtering mechanism, when signs and signals are everywhere and semiotic associations cannot be curtailed: "[Meaning code: if narrative is essential to comprehension, then TTT is not for you. Turn reading `off' and filter `on.' If, on the other hand, you enjoy dream sequences/sequentials, reverse the last.]" (Puzzle Pieces of a Datableede Jigsaw"). Mezangelle, then, is marked by both over-determination and a lack of determination. In this respect one could oppose the conceptual underpinnings of Mez's codework to what Clement Greenberg named as the "modernist reduction," the production of presumably

elementary and elemental, understandable, and directly transmitted art. Mez's intervention is instead to offer movement from relative clarity to obscurity, risking confusion in the interest of producing complexity in the ordinary sense of the word. Mezangelle leaves us with two poles of interference: complexity, whereby the text assemblages are excessively opaque and overloaded, and the basic, whereby the semiotic units are so abbreviated as to be enigmatic and at times indecipherable and the text assemblages are so excessively minimal that they fade into the screen with a kind of erasure (e.g. her horizontal alignment of the dot or the dash). It makes perfect conceptual sense, then, that the common complaint about Mez's work is that it is impossible to know what the signs signify and that mezangelle thus hinders its own functionality and transmission.

The dialectic between the plain and the ordinary and the complex might profitably be understood in the terms Walter Benjamin set down with reference to mass media: information and stories. Information is ephemeral, closed, and "shot through with explanation," while stories are enigmatic, open, complex, and self-interfering. The power of Benjamin's analysis lies in its appreciation of complexity and difficulty, and in its allowance for the disruption of a putatively closed system of language and communication. In broad terms, Will Alexander works with a similar dialectic in his poetic analysis of early online discourse, which, he implicitly suggests, epitomizes the concept of information articulated by Benjamin: "the generic cyberspace adherent, tainted by brutality and boredom....One becomes existentially benumbed by the ease in `checking stockquotes', in `ordering office supplies.' " Because of their material situation within such an environment of mass communication, codework practitioners and critics stress that codework texts and practices gesture toward, and even offers us a glimpse of, a new mode of cognition and textual engagement, libratory in the sense that it counters this precise preference for easily transmissible information that comes already `shot through with explanation.' Thus, for codeworkers who operate within this paradigm of mass media, awakening from the intellectual freeze, remaining cognizant, and countering the automated response is not facilitated by the silence of an artwork but by its noise, by its confrontation of silence, perhaps even by its disordered syntax, rhythm, and spelling. The continuous disequilibrium codework strives to produce might also be understood in the context of Victor Shklovsky's account of the power of poetic language to make the familiar strange, to produce a "a disordering which cannot be predicted" (24).

Offering their own version of defamiliarized, poetic language, Mez, NN, Warnell, and other codeworks tend not to favor integration, sequence, organization, order, and connectivity (the province of database aesthetics) but disruption and counter-organization (the province of email performance and Internet `happenings'). Although the performance paradigm can be brought to bear on codework, hypertext, and cybertext, it is particularly apposite for codework, an artistic practice often performed through mailing lists such as Webartery, Wryting, Poetics, and Nettime and thereby functioning as a kind of "happening." For Mez's art and writing, the political purpose of this aesthetic practice is to imagine a disordering, interruption, and disturbance of the economy of informational transaction. Warnell's garbling of American business English in the code poem, SLANGUAGEÑØÂÃÎ, also literalizes this theme of the disruption of functionalist

communicative action and the uninterrupted flow of information. In this text, Warnell treats American business English as a kind of unnatural language, which he interfuses with "international" alphabetic characters and the numerical elements of code, and from which he removes letters so as to render the resulting memorandum even more compressed and functionalist. That is, he realizes the logic of this mode of international communication and business language, which is to prioritize function over form and communicate a message without considering style or craft. The strategy for this poem and for a mezangelled poem is to disturb, disorient, and defamiliarize, to shift "the units of information and communication from the usual and expected to the cryptic" (rhizome.org interview).

So, as another example of the claims for the political potentiality of codework, Mez stresses that her practice - mezangelle - disrupts the apparently seamless surface of mediatized mass communication and expresses an aesthetic of interference rather than transmission. Specifically, she purports to use her language to expose and critique the politics of or within putatively neutral, functionalist programming codes. In this sense Mezangelle, NN/Antiorp/Integer's code language, and hacking languages alike exist not to function but materially to disrupt the operations of machinic communication and also to produce an awareness of the manipulations and changes within contemporary language. Beatrice Beaubien concurs and suggests that Mez and Netochka Nezvanova are illustrative of a "new paradigm of net communication" in that they create "phrases that disrupt, sometimes gently, empathetically sensual, sometimes violently, abrasively" ("mez|||net|!|zen - Net Fr!sson" 1). It is certainly the case that many readers find their texts perplexing because they violate traditional conventions of language. They are further perplexing because they invite, on the one hand, the assimilative relation of codework practices to prior practices of experimental art and writing. On the other hand, however, these codework texts thwart this apperceptive link to prior textual encounters and engagements, again not necessarily because of their basic form but because of their context and their relation to that context - which itself links the generative material substrate and the mechanism used for transmission.

Mez's claim is that her codework practice needs to be thought in terms of "a gradual re-educative filtration process" that will teach the reader-users "to recognize the source-modes and compile their sensory abilities along the lines of newly-produced expressions intimately related to the stuff of net.wurked life." *Mez Nettime listserv post, "The Dynamics of a Code.Wurk Meaning Trajectory" (January 3, 2002). The opening of the "data bleeding texts," which asks for the reader's childhood nickname as a mode of address, suggests that we are once again to be formally and institutionally brought into language (we could add, as another example, the use of the blackboard in Memmott's Lexia to Perplexia).* Such a cognitive disturbance is also partly facilitated by Mez's use of dynamic text elements, such as one can find in her critically acclaimed data][h!][bleeding texts.

Since the codework texts I am referring to here employ code as a signifier, it is certainly the case that reading them makes one alert to different components of reading, specifically to both the writer's use of programming languages - the uncompiled script processed by human operators - and the computer's execution of the compiled script. In

essence, one pedagogic end game suggested by Mez's work and statements of principle is that the reader-users will learn to process the meaning of some elements of code: a handful of operators, instructions, and characters. Another pedagogic end game suggested by Mez's work is that the reader-users will learn to process a somewhat-new, hybrid, shorthand language (and its semantic, syntactic, orthographic, and orthoëpic conventions) made possible by the digital technologies and now ubiquitous within the realms of advertising, journalistic print culture, and visual media. In the interest of facilitating a kind of oppositional literacy, then, the practice of mezangelle aims to jam the overloaded lines and awaken those that lie dormant; or, as Mez herself declares, it moves "through the neural in waves, swarming into active channels, critically hitting inactive potentials" ("Puzzle Pieces of a Datableede Jigsaw"). This claim regarding shifts in literacy practices is pursued in the critical discourse about Mez's work; for example, in a short analysis, Stephanie Strickland argues that the reading processes that mezangelle requires and the "simultaneity of reference" that it produces "tests fixed neuronal patterns."

Mez's work is conceptual and, like many that of many digital and minimalist artists alike, in its web-based, javascript-enhanced form, it requires a somatically engaged mode of perception. That is, the movement and activity of the viewer produces the meaning of the work and the work itself. Specifically, Mez's textual performances ask that the reader-user intermittently captures, processes, transmits, and even introduces data streams, all of which may themselves have different rhythms and tempos. Similarly, the poet Jim Rosenberg and rich.lit author Talan Memmott both construct complex, densely layered textual fields that are only legible once the layers are drawn apart with the mouse, but mouse movements can also dissolve the texts back into illegibility. These writers and codeworkers quite often present texts that are less concerned with offering a reading experience than they are with working with the language of code to offer comments on form and the materiality of language. Some representative texts will often gain a greater complexity and obfuscation with each mouse click; rather than moving into clarity the text moves into a greater visual and verbal opacity. *One example of the inverse style is Thom Swiss, The Narrative You Anticipate You May Produce, which invites the reader-user to mouse-over and arrange the layers and words such that they move from density and obscurity to legibility.* When we use the mouse to change the speed or visual orientation of one of Mez's permeable net.wurked text installations such as the data][h!][bleeding texts or _][ad][Dressed in a Skin C.ode_. and cause the text to fluctuate between states of legibility and illegibility, we engage in a complex mode of cognitive and physical interaction with the divergent and convergent currents of information within a networked environment.

Codework languages, for Mez and other writers, thus have both artistic and political potential. Part of the mezangelle codework project is to awaken us to - also to comment upon and recompile - the varied and various data streams that we engage, filter, and disregard while multi-tasking. She frames her texts accordingly as "residual traces from net.wurk practices that thrive, react and shift according 2 fluctuations in the online environment in which they][initially][gestated" ("Announcing The Net.Wurk Series"). If "net.wurked" life requires a cognitive adaptation and naturalization to the machine, her

"net.wurk" aims to disrupt its disciplinary and regulatory "sensory reverberations" and offer instead an "infoalert": informatic reverberations that shock and thus gesture toward new, and potentially liberatory, modes of cognition. Within its specific online environment, then, digital media experimental writing, and specifically Mez's codework, offers us a glimpse of a mode of reading, cognition, consciousness, and even pedagogical praxis that is not yet fully available to us.

We can take the practice of codework this far but there are certain limitations that we have to acknowledge: codeworkers such as Mez see their work as facilitating new sensory abilities and new modes of textual engagement and cognition, a claim that is generally true to the extent that a new cognition is embedded in every language. However, at least since Fredric Jameson's hypothesizing about the new form of "consciousness" required to perceive certain postmodern art forms such as the multiple screens of Nam June Paik in their totality, while still recognizing relational differences, the claim for newness is by this point simply part of the genre (31). That is, there is an incontrovertible tendency in contemporary digital art and media work to speak about the facilitation of new modes of cognitive processing and cognitive experience in terms that far exceed the phenomenological. We might note, too, that the claims for newness never really specify whether the "infoalert" as such is embedded into the codework text, or whether it instead emerges as an effect of reading. Further, the claim for a radically different or adaptive cognitive process goes beyond both what the codework texts actually achieve and what we really know about the operations of the brain: in this sense the claim cannot specifically be supported. While the discourse of newness has been made systemic for the practice of digital textuality - new browsers, new processors, new coding languages all purportedly present us with greater possibilities - and while the claim for newness has historically helped to legitimate it, the question of newness is perhaps beside the point in this instance, and at the very least, the province of neurobiology.

Instead, literary and cultural critics should ask both how good and how transformative a "new" mode of cognition would be. In what sense is it important? Would it, or does it, truly offer emancipatory power? To what extent does it lead us toward a synthesis of aesthetic formalism and socio-cultural critique? Given that these questions remain open, as we continue to move beyond an articulation of the principles and properties of codework and work toward the next stage of critical engagement, we need to strike a balance between the limits that codework has reached and the future that it is working toward - its poetic and political aims.

While codework in its many varieties is theorized as both craft and praxis, we are still presented with a radical schism between formal aestheticism and socio-cultural politics (a schism also inherent to the whole field of software art, as Florian Cramer notes *See Florian Cramer, Concepts, Notations, Software, Art (March 23, 2002), on "software formalism vs. software culturalism." Also see Cramer's co-authored juror comments for the read_me 1.2 software art festival: "very few of the pieces submitted had any political or activist usefulness, although several pretended to." Olga Goriunova, "read_me 1.2 winners and honorary mentions," Nettime post (May 22, 2002).*). It might be read, on the one hand, as a kind of hyper-aestheticism, devoid of political content and significance,

and as a techno-formalist fixation on code, a purely functional programming experiment not elevatable to the status of art and not capable of generating literary affect. But so, too, is might be read as a practice expressive of radical cultural politics. Many codework practitioners indeed stress the socio-cultural significance, potential, and content, of their work. They claim, in sum, that codework expresses a revolutionary sensibility within the corporatized environment of IT by turning a pragmatic language toward aesthetic purposes; by issuing manifestos on cultural, political and aesthetic themes; by further complicating our postmodern understanding of authorship; and by articulating an anti-aesthetic, anti-bourgeois code of ethical writing practices. For example, McKenzie Wark has joined with Mez, Sondheim, and Memmott in suggesting that codework's politics derives partly from its approach to writing as a complex, collaborative, multi-faceted activity, one practical component that allows for the claims for codework as an emancipatory aesthetic-political practice.

Sondheim and others also suggest that codework's politics are clearly manifest in the genre's thematization of subjectivity, identity, and the body. They raise the issue of gendered agency, for example, or theorize text as flesh and introduce the problematic of abjection in order to think through the permeation of the boundaries between texts and discourses, or violations of the threshold between code and text. The linguistic-code divide, then, is conceptualized as a binary between the organic and the inorganic and as a binary between flesh and text. Warnell's dialogic tribute to Sondheim, Niku Codepo, evolves from Sondheim's idea that "surface content" might be thought "as parasitic or/ flesh (Niku) covering the bones or workings of things." Continuing in the same vein, Warnell presents Niku Codepo 's decomposing body...subsumed by an "emerging skeletal structure," an anthropomorphizing metaphor for ergodic code that suggests anatomic depth and interiority. These code depths, though, have erupted on the surface, parasitically inhabiting and re-encoding, as it were, the flesh and organs. Suggesting also an architectural arrangement, with code as the skeleton underlying, and disturbing, organic, decaying flesh and the interface as face, this metaphor does not quite present a binary between the corporeal and the machinic. Rather, code is presented as machinic bones, with the two layers, or elements, of flesh and code interfused, as Mez's "skin code" texts also imply.

In that it brings components of code to the surface and intermingles the characters of natural and machine languages, this strain of codework presents a fusion at the level of language, substituting for, and functioning as, the figure of the cyborg. Like the cyborg, codework violates the categorical and epistemological boundaries between the organic and the inorganic, the public and the private, the visible and the hidden. The critical door that opens here - with the cyborg's and codework's fusion of the organic and the inorganic - allows for a reconsideration of post-human subjectivity, and it also allows for our consideration of another kind of synthesis. If we reduce the practice of codework to either its form or its content, we would produce a false, and falsely reductive, binary between aesthetics and politics. Criticism and the arts alike have the capacity to synthesize the two aspects, without neglecting either formal or socio-cultural analysis, by building on the set of relations that are nascent within the discourses of codework and net.art. Written out

as "code.work," as with `net.art,' `code' would be the object and `work' would be the property that is transferable to other contexts. Codework, in other words, contains within itself the means to theorize it as aesthetic craft and political practice.

Portions of this essay were presented at the Digital Cultures Research Group "Interfacing Knowledge" Conference, UC Santa Barbara (March 2002); the Technotopias Conference, University of Strathclyde (July 2002); and the trAce Incubation Conference, Nottingham Trent University (July 2002). Maria Damon, Katherine Hayles, Jennifer Jones, Alan Liu, Talan Memmott, Chris Newfield, and Alan Sondheim made particularly helpful comments. Thanks to Russell Samolsky, Timothy Wager, and Joseph Tabbi for incisive editorial suggestions.

notes

Bibliography - codework

Alexander, Will. "The Myrmidons of Oblivion" (April 2000).

Baudrillard, Jean. "Aesthetic Illusion and Virtual Reality." Jean Baudrillard: Art and Artifact. Ed. Nicholas Zurbrugg. London: SAGE, 1997. 19-27.

Beaubien, Beatrice. "mez|||net|!|zen - Net Fr!sson." American Book Review 22:6 (September/October 2001): 3-4.

Beiguelman, Giselle. "Liquid Texts." Leonardo Electronic Almanac 10: 8 (August 2002).

Bergman, Eric, ed. Information Appliances. San Francisco: Morgan Kaufmann Publishers, 2000.

Carl Andre Hollis Frampton 12 Dialogues 1962-1963. New York: NYU Press, 1981.

Cayley, John. "The Code Is Not the Text (Unless It Is the Text)." Electronic Book Review (September 2002).

Cox, Geoff, Alex McLean, and Adrian Ward. "The Aesthetics of Generative Code." Web-published paper.

Cramer, Florian. "Combinatory Poetry and Literature in the Internet" (October 19, 2000). Dec 19, 2000, Forum Ästhetik digitaler Literatur, Universität GHK Kassel online: dichtung digital (Roberto Simanowksi, ed.).

---. "Digital Code and Literary Text." BeeHive 4:3 (Fall 2001).

---. "Software Art and Writing." American Book Review 22:6 (September-October 2001): 8.

---. "Program Code Poetry." Cream: Net.art newsletter (Mar 28, 2001). Ed. Josephine Bosma.

---. "Concepts, Notations, Software, Art" (March 23, 2002).

---. "Language, A Virus?" I Love You: The Catalogue. Digital Craft exhibition.

Esdaile, Scott. "The Net.Wurk Series _][ad]Dressed in a Skin C.ode_." Nettime posting (April 17, 2002). Reprinted from fine Arts forum.

Jameson, Frederic. Postmodernism, or, The Cultural Logic of Late Capitalism. Durham: Duke University Press, 1991.

Johnston, David. "Programming as Poetry." Year01 Forum (April 12, 2002).

LaCook, Lewis. "Rot and Root." Suite101.com (February 22, 2002). Available: http://www.suite101.com/article.cfm/15239/89610.

Lennon, Brian. "Screening a Digital Visual Poetics." Configurations 8:1 (2000): 63-85.

Memmott, Talan. "E_RUPTURE://Codework"."Serration in Electronic Literature." American Book Review 22:6 (September/October 2001): 1, 6.

Mez. Nettime Announcements listserv post, "Announcing The Net.Wurk Series::_][ad] [Dressed in a Skin C.ode_ + JavaMuseum - [mez] solo show." February 3, 2002.

---. rhizome.org interview with Josephine Bosma (February 1, 2000).

Mez Breeze. "Non Compos Mentis: Zen-Tripping the Non-Conference Circuitry." fine art forum 15:5 (May 2001).

Rosenberg, Jim. Cybertext Yearbook 2002. Eds. Loss Pequeño Glazier and John Cayley. Jyväskylä: University of Jyväskylä, 2003 (forthcoming). 3 pp. manuscript.

Shklovsky, Victor. "Art as Technique." Russian Formalist Criticism: Four Essays. Trans. Lee T. Lemon and Marion J. Reis. Lincoln: University of Nebraska Press, 1965. 3-24.

Sondheim, Alan. "Introduction: Codework." American Book Review 22:6 (September/October 2001): 1, 4.

Steidl, Jutta. "If () then ()". I Love You: The Catalogue. Digital Craft exhibition.

Strunk, William, Jr. and E.B. White. The Elements of Style. New York: Macmillan Co., 1959.

Warnell, Ted. _Poems by Nari: Visual Poetry from the Cyberstream_.

Cite this Essay:

. “Interferences: [Net.Writing] and the Practice of Codework”, Electronic Book Review, September 8, 2002, .
Readers wishing to respond to an essay in *ebr* may send ripostes or short glosses to the journal’s Managing Editor, Will Luers.