Notes on the cultural dimensions of software and art

Andreas Broeckmann
abroeck@transmediale.de

□

□

[This text was first posted on *nettime*. This is the script of the talk that I gave on the last day of the ars; some of the themes discussed here over the last days resonate, and I thought it might be interesting to chip it in; apologies for the loose style, but it had to work as a talk way at the end of a 5-day conference; comments welcome, of course; -ab]

(Lecture manuscript; "Ars Electronica 2003")

At "transmediale" in Berlin, we have been organising a competition and conferences about software and generative art since 2001. It is curious that this initiative which brings me to this festival about Code started here at the ars five years ago when John F. Simon, with whom I was in the net art jury, said there should really also be an art category in the competition besides net art and interactive art, the new one devoted to artworks specifically dealing with computer software. In Berlin, we took up that challenge and have been exploring the field over the last three years; other initiatives devoted to software and art have, amongst others, been the eu-gene mailing list, the "Read_Me Festival" and the *Runme.Org* website, the "Generator" art exhibition curated by Geoff Cox in England, the "Electrohype" festival in Malmoe Sweden, and of course Christiane Paul's *CODeDOC* project in New York.

The polemical equation in the title of this year's "Ars Electronica", Code=Art, is of course wrong; code in general is not art, 1st because code is mostly written for completely instrumental reasons without an artistic intent or expression, and 2nd because art is not in the code, but in the social process that we call art and that involves the cultural context of production and reception in which art is articulated. There was some reference here to the separation between the liberal and the mechanical arts and to the fact that their separation is not universal but a product of European culture since Greek antiquity; while I agree that it is important to point out the fact that that separation thus has both historical and cultural specificity, I would also maintain that there are good historical reasons for this separation, and I would like to defend an understanding of art, call it Old European if you like, that places art at the intersections, and the lines of friction

between different social and political systems, where it dramatises these lines of friction, where it expresses the beauty and the rawness of the most unlikely possibilities, where it makes strange the most familiar constructions of our culture. Art can thus do much more than illustrating, pleasing, window-dressing. And I believe that art using software as its main material, can also work in this direction and push the boundaries of our understanding of art in the age of digital computing. I'll try to talk about that in the next 25 minutes.

The starting point of the debates about software and culture is the realisation that we have to take software seriously as a cultural artefact with a history, a sociology, and culture, in fact _different_ cultures and histories attached to them. As we have heard, these are a variety of cultures of invention, production and application, communities of different shapes and intent, programmers and user groups, nerds, geeks and DAUs.

A name that has been strangely absent from the debates of the last days is that of Matthew Fuller, an English writer and Software critic who has worked quite extensively about the social implications of software. (Another one is that of Graham Harwood, who has done both theoretical and practical work as an artist programmer on issues of social and critical software). Fuller's text "It looks like you're writing a letter" is an extensive analysis of Microsoft Word and the social assumptions that have been coded into this programme. In the more recent text, "Behind the Blip", Fuller makes the useful distinction between critical software, social software and speculative software. Critical Software questions socalled "normal" software by drawing out its hidden, yet traceable flaws, as Fuller did in an installation in which he printed all the hundreds of dialogue boxes that you can find in MS Word and pasted them on a wall. The other strategy Critical Software can take is specially written software that comes along looking like "normal" software, yet unexpectedly behaving very differently. Social Software, in Fuller's understanding, is software that directly addresses the social conditions of using specific software tools, by making them explicitly accessible and low-threshold, and Social Software is also engaged in and emerging from social networks and communities. Thirdly, Speculative Software is described very lucidly by Fuller "as software that explores the potentiality of all possible programming. It creates transversal connections between data, machines and networks. Software, part of whose work is to reflexively investigate itself as software. Software as science fiction, as mutant epistemology. Speculative software can be understood as opening up a space for the reinvention of software by its own means." - I used to argue that this notion of speculative software probably comes closest to my understanding of software art; but I now tend to believe that art projects can equally belong to the areas of critical or social software, and that the notion of art cuts across these different fields - I will come back to this later.

What we can easily glean from people like Fuller or Ellen Ullman, whom Fuller quotes, is that software is embedded in social practices. This is why we can speak of the cultural dimension of culture as the heterogeneous social field in which software gets built and used, in which it operates and in which it gets developed; the software "environment", this ecology, is of course technical, but by being technical it is also social and political - in its production cycles as well as in the fields of its application.

Think of the *Sobig.F* computer virus, a mail worm which has been plagueing the Internet since mid-August. Hundreds of thousands of E-Mail messages with attachments have been mailed to servers all over the world, clogging up the lines, servers and mailboxes, intended to prepare for a major attack on Microsoft servers at a given time. The other day I said to Pierre Levy that what we experience on the Net is often not a sign of collective intelligence, but of collective stupidity. I should have been a bit more balanced in saying that, but what I meant was that the Net is a social environment in which many things go wrong, in which there is a lot of spam, conflict, violence, and redundancy. I understand the value of connecting human intelligence in a network, and if we apply a notion of collective intelligence that is more fractured, so that it applies to smaller, definable collectives, then I am all in favour. I think the way in which the *Sobig.F* worm was dealt with by systems administrators was amazing - in my experience it took less than 36 hours from the first attacks flooding my mailbox, to the solution being implemented as software filters on the mail servers; through message boards, analyses of the worm code were shared and possibilities for stopping it were discussed, and the most effective solution, written I believe by a Viennese programmer, was then adopted world-wide. The guys at IN-Berlin were part of that exposition of collective intelligence, which made it possible for me to return to my mailbox without fear very quickly. But at the same time, that intelligence is not universal, because some people are still affected by the roaming worms, and the whole problem only started because many users were downloading and executing the worm software innocently, which is why it spread so quickly. What I meant in my comment to Levy was that I think that it sounds very ideological when he mentions collective intelligence, without referencing the dimensions of conflict on the Net, without referencing the widespread lack of media competence, and the inbuilt stupidity of some commercial software applications. My guess is that a semantic system that is based on a consensual social model will be doomed to fail. But that is, of course, my own ideological perspective.

The gist of my argument today is that the cultural topology of this software "environment" is articulated by art projects. I'm not saying that all art with digital media has to address the specifics of software, but I think that Software

Art should.

When Alex Galloway quoted me yesterday as the supposed author of saying that software was a cultural technique I was kind of surprised, because I believed that that is a widely shared understanding of any artefact, whether technical or mechanical, which has no "original author" any more (so I kind of refute that reference which Alex took from a text posted on Nettime and featured on Autonomedia's Interactivist blog). Academic training in post-structuralism in the 1980s spoon-fed me the rhetorical reflex that artefacts have specific historical, social, mostly also economic contexts, and that any conscious attempt to conceal that specificity must be hiding specific interests or motives. Such training makes for useful critical questions, and for good conspiracy theories, which these days turn out to be true more often than not.

Cultural techniques are the practices and applications that you can use for your everyday survival, and they can go from table manners and communication skills to the ability to programme your VCR or to set a filter in your E-Mail programme to avoid messages from certain people. Writing and reading software is a less widely distributed, yet very valuable cultural technique which can be empowering and otherwise satisfying in a variety of ways. Even a text-based Mac user like myself, completely code-illiterate, is confronted with this fact more and more often.

The "cultural topology of software" is the, excuse the metaphor, multi-dimensional "landscape", the different layers, plateaus, call them what you like, that intersect in the practices that are constituted by the practical application of software. This is of very general. What I mean is that when you take a web browser like *Nebula*, by Netochka Nezvanova, you have, for instance, the context of the World Wide Web and of the normalised assumptions about the representation of HTML code; connected with *Nebula* are also the social complications introduced by the NN or antiorp character of the author, the economic dimension of having to pay for downloading this alternative web browser, and so on. It does not make sense to strictly separate the software from this context, quite to the contrary, *Nebula* is an interesting project precisely because it plays on those different registers of software culture. Similarly, Adrian Ward's *Signwave Auto-Illustrator*, an enhanced and partly perverted re-engeneering of normal graphics programmes, plays on the aesthetic and ergonomic expectations normally brought to a piece of software. You can buy the *Auto-Illustrator* like any other software package, but what you get will make you think a lot about what your achieved notions of 'normal software' and its usage have been.

I don't have the time to elaborate on this much further, but I guess it becomes

clear what I mean by the cultural topology of software with its political, legal, economical, etcetera, dimensions. I took the two examples, *Nebula* and *Auto-Illustrator*, because they were the first winners of the "transmediale" software competition in 2001, which were followed by *LAN's Tracenoizer* and Alex McLean's *forkbomb.pl* in 2002, and by the Gnutella network browser *Mini-Tasking* in 2003. As a recent example of this kind of work I would like to mention Franz Alken's *Machines* will eat itself, which just won the German "Digital Sparks" student award and which allows you to create ficticious identities, bots, which then go about filling in forms on websites with their fake personal data, thus junking the databases of overly eager data mining companies. The cultural practices that emerge with technologies, like today weblogs or wireless communications, further transform this techno-social topology.

(projects to mention here include IOD's *Webstalker*, KRcF's *Minds of Concern*, Jodi's *Browser* and game manipulations, *retroYou r/c*, and *Gnutenberg.pl*)

When talking about software and art, we have to speak about aesthetics, that is engage the value systems that inform our experience of art, and our perceptions in general. References have been made to the traditions of Fluxus, Conceptual Art, or Net Art, each of which implies a set of assumptions about the ways in which to judge the artistic quality of artworks. Over the last 200 years, European culture has seen aesthetics of beauty, aesthetics of the sublime, aesthetics of ugliness, and aesthetics of formal order. But this history teaches us, that there are alternative ways of approaching software-based artworks than Max Bense's extremely formalistic *Generative Aesthetik* which he formulated in the 1960s.

Sakane san has discussed the different approaches to media art in the 20th century in his lecture on Tuesday, and he has shown how different the approaches to this kind of art practica have been. Just as an aside: I believe that it would also be interesting to revisit the debates about Realism vs Formalism between Lukacs and Brecht in the 1930s in this respect, if only to sharpen our perception for the level of critique that can be brought to significant artworks.

On this note, I fully agree with Christa Sommerer who called for a more engaged, more critical debate about specific projects and practices in the field of media art. That debate will hopefully help to distinguish the qualities and aesthetic specificities of different works, and even if we are not headed for some sort of normative aesthetics, it will hopefully help us to make and articulate our value judgements.

My own idea of art practice, which I also bring to this field of software-based work, is opposed to bland visualisations and translations from one formal

system to another. I understand the need for a kind of software formalism in an early period of exploring the material and formal specificities, but as Christa Sommerer said yesterday, these are sketches which should not be considered as serious attempts at making art. I believe that we need a strong notion of what constitutes art, and we must argue about that, but it would help immensely if we could agree on drawing a bottom line which excludes some attempts. For me, and again I put this up for discussion, art is about the transgression of boundaries, about making familiar experiences strange, about dramatising what pretends to be innocent, and about exploring the virtualities, the potentialities of technologies and human relationships.

I would like to spend the last minutes mentioning some such unlikely projects in order to open up the debate about what it is that interests us about software and art. The projects mentioned earlier by Christian Hübler (*Crack It!* - connective force attack, open way to public, and *Minds of Concern*) should certainly also come into that equation.

First there is the whole question of identity in the digital age, the issues of data-mining and privacy, the protection of our databodies, also the aspects of race and gender come into play here.

(Discuss Eva Wohlgemuth: *Body Scan*, mention Ulrike Gabriel: *Sphere*, Nathalie Jeremijenko's *tree cloning project*)

A second area that, as we have seen over the last days, is relevant here is, speaking in general terms, the technical infrastructure, the software code itself and the computer languages it is written in, the translation modes, the question of the representation of code, and of visualisation.

(Present Jaromil: *forkbomb*, discuss Jahrmann/Moswitzer: *Nybble Engine Tools*)

Let me say that the polemics I am putting forward here is not a claim for taking the fun out of art; to the contrary, I belive that both of these projects exhibit a very good sense of humour, they work like jokes in a Freudian sense exactly because they reference the cultural context in relation to which they formulate their own narrative or process.

In many cases, art projects relate to or express their cultural environment in very restrained or benign, at times even banalising ways. This is not only an issue in software-based art, but of digital art practice in general - it often tends to be affirmative of the technology, uncritical of its corporate politics and superficial in its formulations and expressions. Where is the desire for excess in

software-based art? Where do we find, as Stefan Riekeles said the other day, the surplus, the surprise, that which we do not know yet and that is not already legible in the software code or the technical dispositif that artists prepare so ardently?

By way of closing, I would like to read you the jury statement of the obscure "Lux Ziffer Award", which has been awarded for the second time at "transmediale" last February. Like the anonymous jury, I do not want to infer that the winning project is an art piece; but I do want to suggest that we look for art projects that are able to elicit such excited responses as this one:

□

> "Everyone has the right to freedom of opinion and expression; this right includes freedom to hold opinions without interference and to seek, receive and impart information and ideas through any media and regardless of frontiers."
>
> Article 19 of *Universal Declaration of Human Rights*

The anonymous "Award Lux Ziffer 03" goes to the anonymous artist Vladimor Chamlkovic alias Melhacker aka Kamil for his project *scezda*, a polymorphic superworm threat. He has successfully infected the international press with a new virus: mistaking Al Qaeda for al gorithm thus feeding the myth of cyberterrorism and mass hysteria. His threat to release a blended megavirus in the case of a us attack on iraq introduces new parameters to media art: boolean vengeance and political threat.

scezda is supposed to be a 3-in-one recombination of *sircam*, *klez* and *nimda*, the three virii having had the most impact within the last year. However, Melhacker's past background in artistic success is rather poor by number of infections, distribution, threat containment and ease of removal. in terms of quantity, his work has failed. In terms of quality, his publicity attack obsoletes the real existence of *scezda*, it has already raised a profitable discussion of security myths and hysteria amongst corporate fear-feeders: trojan whores consuming trojan horses, spreading the news of worldwide economic damage and loss of daily lifestyles.

We do never wish to see *scezda* in the wild, because this would merely mean, that fossil panic has triggered war. And this is bad. And so are we. If the unwise have an unwise leader, all are led to ruin.

Thank you for your attention.