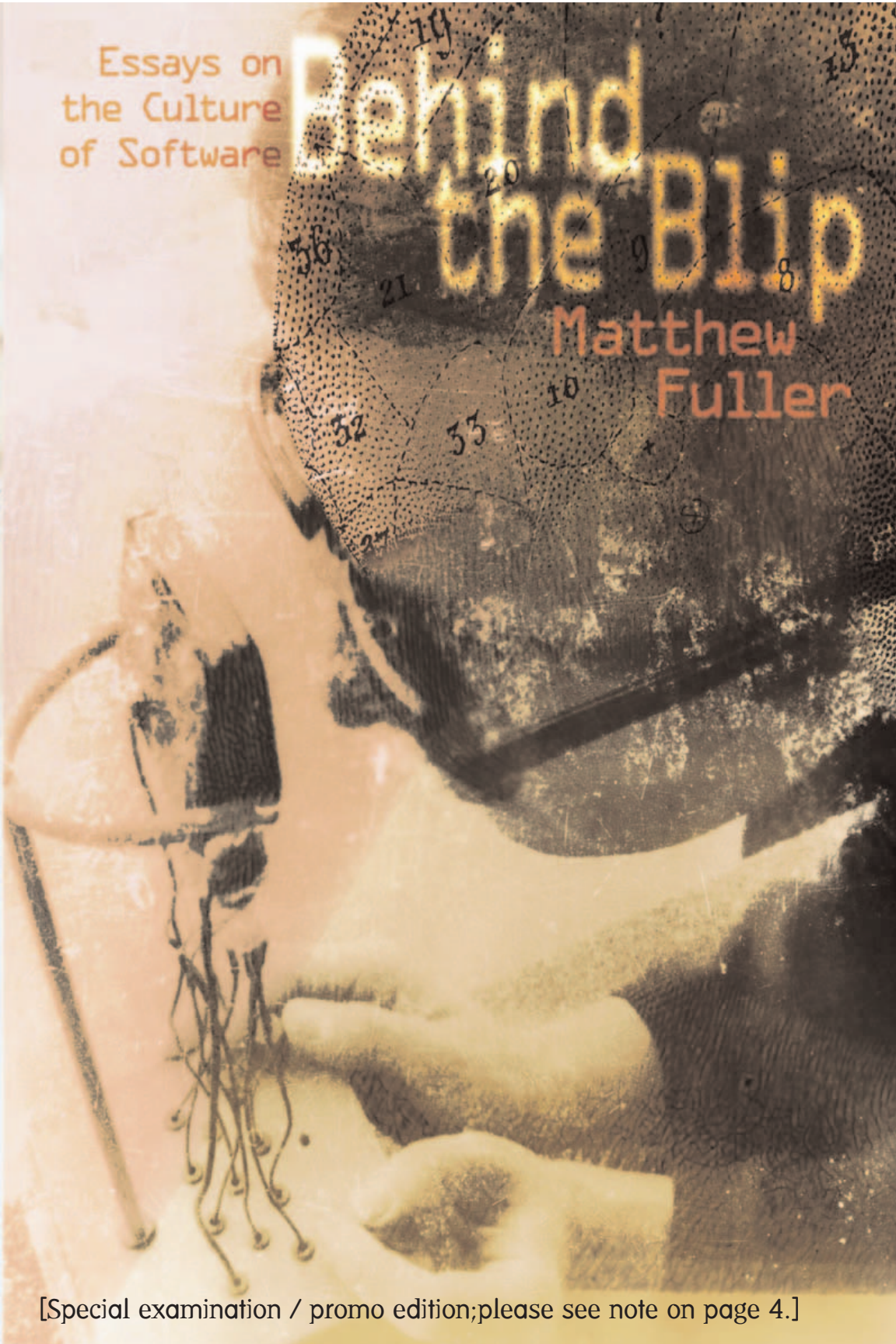

Also by the author

Editor:
Flyposter Frenzy: Posters from the Anticopyright Network
Unnatural

Co-Editor:
README! ASCII Culture and the Revenge of Knowledge

Author:
ATM

BEHIND THE BLIP

ESSAYS ON THE CULTURE OF SOFTWARE

MATTHEW FULLER

//AUTONOMEDIA

NOTE TO THIS EDITION:
This is an abbreviated promotional copy of *Behind the Blip* for examination use only; permission is granted to copy and freely distribute this electronic file. For purchases or inquiries about this or other Autonomedia books, please contact Ben Meyers, email ben@autonomedia.org, or visit the web site www.autonomedia.org/behindtheblip.

Autonomedia
P.O.B. 568 Williamsburgh Station
Brooklyn, NY 11211-0568 USA
Phone & Fax: 718-963-2603
email: info@autonomedia.org
http://www.autonomedia.org

Book design: Dave Mandl

ISBN 1-57027-139-9

Printed in Canada

For Mandie, Leon, Milo, and Rosa

ACKNOWLEDGEMENTS

"A Means of Mutation" arises from work by the group I/O/D (http://bak.spc.org/iod). Thanks to Simon Pope and Colin Green for several years of it. "Break the Law of Information" is connected to a project run on behalf of the group Mongrel. Thanks to Harwood, Mervin Jarman, Richard Pierre Davis, Matsuko Yokokoji, and all the other participants in Natural Selection. "Long, Dark Phone-In of the Soul" was originally published in *Mute*. Thanks to all the crew there. "Visceral Façades" and "It Looks Like You're Writing a Letter" were first published in *Telepolis.* Thanks to Armin Medosch, then-editor of this website. The work on Microsoft Word was supported by Norwich School of Art and Design. Thanks to Hilary Bedder, Helen Boorman, George MacLennan, and Simon Wilmoth for setting up a research environment sufficiently hands-off to let me get hands on such intractable material. "The Impossibility of Interface" was written during a sabbatical from Middlesex University. Thanks to all staff and students in Media, Culture, and Communications there.

All of the texts in this book have appeared in one shape or another on the mailing list Nettime. Thanks to the moderators and all those who make this such a useful resource. It should also be acknowledged that this work arises from the activity of friends, workmates, readers, users, distributors, copiers, events, discussants: activity that would be endless were it nameable—thanks.

Thanks to the Autonomedia collective and to David Mandl for engineering the book so generously and expertly.

CONTENTS

BEHIND THE BLIP: SOFTWARE AS CULTURE

(SOME ROUTES INTO "SOFTWARE CRITICISM," MORE WAYS OUT)

SOFTWARE CRITICISM?

There are two questions which I would like to begin with. First, what kind of critical and inventive thinking is required to take the various movements in software forward into those areas which are necessary if software oligopolies are to be undermined? But further, how are we to develop the capacity for unleashing the unexpected upon software and the certainties which form it?

Second, what currents are emerging which demand and incorporate new ways of thinking about software?

One of the ways to think about this problem is to imagine it as a series of articles from a new kind of computer magazine.[1] What would happen if writers about computers expanded their horizons from the usual close focus on benchtests and bit-rates? What would happen if we weren't looking at endless articles detailing the functionality of this or that new version of this or that application? What if we could think a little more broadly—beyond the usual instructional articles describing how to use this filter or that port? What, for instance, would it mean to have a fully fledged "software criticism"?

First, let's look at what already exists. Certainly, we are not short of examples of prior art. In terms of the academy, sociology,

for instance, offers: Jeannette Hofmann's descriptions of the gendering of word processor software and its patterns of use within work;[2] Paul N. Edwards's history of the development of computer technologies through the models of science promotable at the height of the early cold war;[3] Michael R. Curry's formulation of a technico-aesthetic economy of signification and ownership in geographic information systems;[4] Donald MacKenzie's work on the political implications of floating-point-unit calculations in the design of missile guidance systems[5]—the list goes on and extends to substantial areas in ethnography and anthropology.[6] Material based around philosophy and literature includes Michael Heim's *Electric Language*[7] and the contributions of Friedrich Kittler, despite his assertions that the object of attention here does not exist.[8] We can also look to texts which come out of bookshops, but that don't get libraried up so much: Howard Rheingold's *Tools for Thought*[9] and J. David Bolter's *Turing's Man,*[10] for instance. This list is certainly short, but it does continue. The creation of imaginary bookshelves is as good a way of thinking through combinations as the imaginary museum, and there are three areas in particular which seem to offer elements recomposable into a more thoroughgoing strand of thought about and with software.

HUMAN–COMPUTER INTERFACE

Human–Computer Interface (HCI) is obviously one area that should be turned to. This is, after all, the point at which the machinations of the computer are compelled to make themselves available in one way or another to a user. The way the computer makes available such use, and the assumptions made about what possible interactions might develop, are both fundamentally cultural.

Given this, HCI has an unusually narrow understanding of its scope. Much of the rhetoric is about empowerment and the sovereignty of the user, whose "personality" shapes and dialogues with the machine. It should be asked what model of a persona, what "human," is engineered by HCI. We should not settle for answers that stray anywhere near the singalong theme-tune of "empowerment." (Let us not forget that much of the methodology of HCI is still derived from theories that led B. F. Skinner to assume that he

could train pigeons—in the days before Cruise—to act as primitive guidance systems for missiles.)

It seems clear that the vast majority of research and production in this area remains concerned with imposing functionalist models on all those systems that cohere as the user. Perhaps given software's basis in boolean logic, where every action must be transmogrified into a series of ons and offs held in hundreds of thousands of circuits, this is inevitable at a certain level. Make no mistake, HCI works. It is productive because it belongs to a long line of disciplinary idealisations of the human that nevertheless have the capacity to latch onto flesh. The mainstream of HCI is considered here to be those largely positivist approaches which are represented in standard formulations of the discipline such as the *Handbook of Human–Computer Interaction*.[11] When it comes to arranging the most suitable combination of ergonomics and information-design to ensure that a pilot can drop bombs or stockbrokers can move funds in the most efficient, information-rich, yet graphically and emotionally uncluttered manner, HCI delivers the goods. Reaction times—the number of interactive steps from task identification to task execution—can be measured. The results can be tabulated against variants of the system. The whole can be fine-tuned, pixels shifted, operatives retrained: the loop between stimulus and response tightened into a noose. This is the fatal endpoint of the standard mode of HCI. It empowers users by modelling them, and in doing so effects their disappearance, their incorporation into its models.

There are, of course, many "human-centred" variants on such designs. Yet this kind of naming illustrates its fatal flaw. There is still a model of the human—what constitutes it, how it must be interfaced—being imposed here. Some developments in software design have been made by acknowledging this. Alan Cooper's[12] approach to interface design works, for instance, by establishing a number of stereotypical users of a system. They are imagined as full "characters," users of a system which is reworked, primarily in terms of interface, in order to meet an aggregate of their needs. The deliberate fiction of user identities is made visible at the design stage in order to allow greater insight into the techno-aesthetic

composition of the software. A small, useful step would be to make these manufactured identities, but treat them as psycho-social open source.[13]

More broadly, much could be gained by a change in the focus of HCI. In its emphasis on perception, on narrowly applied psychology, it has split the user from any context. One thing that is compelling about software is how it contains models of involvement with processes rather than simply with static elements—think about groupware, or the way in which most previously discrete applications have become part of wider suites of processes, to say nothing about the inherently modular nature of Unix. What would it mean to incorporate an explicitly wider notion of such processes into software—to reinfuse the social, the dynamic, the networks, the political, communality (perhaps even instead of, or as well as, privacy)—into the contained model of the individualised user that HCI has us marked down for?

We can see movements toward this in sociology- and psychology-derived currents within HCI such as Participatory Design. Here, there is a range of collaboration between users and designers that aims to stake out a territory for certain models of what a user becomes interfaced to. Notably, this territory can sometimes even be defined geographically, as in the institutional, corporate, and trade union uptake of this approach in Scandinavia. What these approaches allow is a removal of the more or less negative preconditions of the standard model of HCI that is simply applied to users by experts. The area of Computer Supported Co-operative Work brings some of these elements together, but largely as a way of making them function, of turning them to account.

One tendency that is of interest here is in the proliferation of higher-level languages and authorware. These allow for currents of design that place value on experimentation, rather than adherence to pre-formatted notions of functionality, to invade the conceptual and practical space of the computer. At the same time, capacities for invention do not belong solely to those who most often claim them; the problem of design, of interface, must be set in wider terms.

A key problem here, though, is the danger that a set of questions tend to stabilise out as particular techniques in which something gets solved. Software is a place where many energies and formations meet. At the same time, it constantly slaps up against its limitations, but these are limitations of its own making, formulated by its own terms of composition. Software is always an unsolved problem. We need ways of thinking into and activating this process of becoming, rather than some "kinder" or more "creative" design.

PROGRAMMERS' SELF-ACCOUNTS

Another pre-existing area that offers insights for an understanding of software as culture is the tradition of accounts of their work by programmers. Key texts are "Perl, the First Postmodern Computer Language,"[14] by Larry Wall, and *Close to the Machine,*[15] by Ellen Ullman. Both of these in their own ways document the interrelation of programming with other formations—cultural, social, aesthetic. These are drives that are built into and compose software rather than use it as a neutral tool.

These accounts of programming are somewhat at odds with the idealist tendencies in computing. In the recent film based on Robert Harris's novel *Enigma,* one of the characters makes the claim most succinctly: "With numbers, truth and beauty are the same thing." Such statements are the pop-science version of the attractions of so-called "pure" mathematics. It is also the vision of numbers that most often finds its way to the big screen. (Think also of the film *Pi,* where a cute crazy loner struggles for a glimpse of the numerical meta-reality.) But more crucially, they are a direct route to the cultural backbone of classical idealism. There are harmonious relations between forms of every kind that can be understood through the relations between numbers. The closer they are to achieving purity of form, the more beautiful they become. There is an endpoint to this passage to beauty which is absolute beauty. Access to and understanding of this beauty is allowed only to those souls that are themselves beautiful.

The consequences of such ordering are of course clear, if only in the brutality of their collaboration with and succour for hierar-

chies of every kind. The kitschier end of this tendency is found, in computing, in accounts such as *The Aesthetics of Computing*.[16] But it is far more violently enhanced by computing when it works to provide an aesthetics of social control. There are far more opportunities offered by constructional and fabulatory approaches. Numbers do not provide big answers, but rather opportunities to explore further manifold and synthetic possibilities—that is to say, they provide access to more figures.

CRITICAL THEORY

Under the aisle-headings Critical, Social, Political, Cultural, Material, Visual, Aesthetic or Blahblah Theory there is a warehouse of tools available, tools which are held back from invading the conceptual domains of software by the myth of its own neutrality as a tool. These rubrics themselves are only really of any use when they are disingenuous, when they don't quite fit. For this reason, there's no option of chewing through the Dewey Decimal System and tabulating them. (The use of the term "theory" is here meant simply as that which develops a model of an approach to the material it works on as it uses it, and with which it shares an equal importance in terms of its production. It therefore acts in relation to other such models at the same time as operating in the field on which it attends. This might be true to some extent of writings on HCI and in programmers' self-accounts, but these are always primarily, rather than equally, concerned in epistemological terms with the accomplishment of an instrumental task.) Here, it is only necessary to make two suggestions, one in terms of scale, the other in terms of activity.

In general, critiques of technologies, particularly media, are made on the basis of a category or class of objects, rather than specific instances of that class.[17] Perhaps the timescale of literary production precludes anything else, but it is also a question of pretensions to timelessness. Why spend time working into a piece of software, when it'll be reversioned in a couple of months? The kind of material that is now gathered to beat students about the heads with as

"cyberculture" is generally exemplary in this way. Would it not make more of the gift of your wisdom to the human race to ponder the verities of some enormous category that will combine shelf-longevity and discourse-redeployment potential? It is not that such work is strictly non-empirical, but that in being concerned with offering grand theory-panoramas and generic summations any chance of latching into particularities, particularly those against which such concepts can be tested, disappears under the clouds.

That timescales need also not be determined by corporate release schedules in producing an analysis of software is suggested by Donald Knuth[18] when he proposes a deceptively simple task for computer scientists: Analyse every process that your computer executes in one second. Writing at the end of the eighties, he suggests the number of tasks will be around 250,000. Perhaps this would provide sufficient scope? Timelessness condenses, and the researcher appears years later having annotated an entire second's worth of hundreds of thousands of instructions. Most of the transcript would of course consist of repetitions of instructions carried out on minutely incremental changes in variables. Why not contaminate this simple telling of the story of what goes on inside a computer with its all-too-cultural equivalent? The transcript of the contents of a mind over one day, or of a memory in the transit of a morsel of cake from plate to mouth, provided opportunities for sentences in "fiction" to slide in and out of scale, from layer to layer, in convulsions of sprouting, connecting text. Perhaps the same can also be done at this scale?

At another scale, one of the advantages of the work of Jakob Nielsen, Donald Norman, and others is precisely that they focus in on very specific problems, albeit those of a narrow cast and range of interpretation. Although they tend to deal in a somewhat over-literal application of cybernetic "constraint" rather than the generation of its twin, "freedom," their focus allows them to claim at the very least the rhetorical power of practice. Nit-picking has the capacity to become another mode of the war of the flea. Theorisations of software that are able to operate on the level of a particular version of a program, a particular file structure, protocol, sampling algorithm, colour-scheme, API, Request For Comment,

and so on, are necessary. Further, it is essential to understand any such element or event as only one layer or node in a wider set of intersecting and multi-scalar formations. That is to say that, whilst within a particular set of conditions its function might well be to impose stasis upon another element, such an effect cannot always be depended upon. In addition, whilst one might deal with a particular object, it must always be understood not as something static, although it may never change, but to be operating in participial[19] terms.

Such a focus on the unfolding of the particular—with an attention to how they are networked out into further vectors, layers, nodes of classes, instrumentalisations, panics, quick fixes, slow collapses, the sheerly alien fruitfulness of digital abundance, ways in which they can be taken up and made strange, mundane, and beautiful—will at least ensure two things. First, that it busts the locks on the tastefully-interiored prison of stratified interdisciplinarity. It would be a dire fate to end up with a repetition of the infinitely recessive corridor of depleted jargons and zombie conferencing of Film Studies. Second, and in terms of activity, that an engaged process of writing on software might reasonably hope to avoid the fate of much recent cultural theory, that is to say, to step outside of its over-eager subordination to one end of the schematic of information theory: reception.

AVERSION TO THE ELECTRONIC: A HALLMARK OF CONCEPTUALITY?

As an example of where theoretical work presents us with an opportunity to go further, I want to run through a particular example.

In their book *What Is Philosophy?,*[20] Gilles Deleuze and Félix Guattari present a back-to-basics manifesto. Philosophy has become the domain of men whose occupation is the construction of vast hulks of verbiage—immense dark ships with their single-minded captains, vessels constructed of words, unable, unwilling even to communicate amongst themselves and which, as a result, pass each other by in the night.

The book is at once a rescue of philosophy from its status as doomed élite subculture staffed by the populations of the soon-to-be closed ghost departments of the universities of Europe, and also a restatement of the primary task of philosophy: the invention of concepts. In order to state their case for this, they need to clear the decks of other ways in which the word *concept* is used. One of the problems they see facing their use of the term is that

> in successive challenges, philosophy faced increasingly insolent and calamitous rivals that Plato himself would have never imagined in his most comic moments. Finally the most shameful moment came when computer science, marketing, design, and advertising, all the disciplines of communication, seized hold of the word *concept* itself and said: "This is our concern, we are the creative ones, we are the ideas men! We are the friends of the concept, we put it in our computers."[21]

As is well known, their work is in many ways an immense, vibrant resource. However, it appears that there is a particular blockage, more so perhaps in the work of Deleuze than of Guattari,[22] when it comes to a useable theorisation of media. There is a tendency here which is typical, not just of their work, but of much theoretical work throughout the twentieth century. Whilst some media systems, such as books, music, painting, film, etc., are entered into with a profound spirit of exploration and invention, those that are electronic are treated as being fundamentally suspicious.

As a result, when they do touch on electronic media, their work jumps into and out of various similarly short and undifferentiated takes. In short, electronic media do participate in "conceptuality." The conceptual personae that Deleuze and Guattari so suggestively propose in *What Is Philosophy?* can be read as a proposal for an understanding of software as a form of digital subjectivity—that software constructs sensoriums, that each piece of software constructs ways of seeing, knowing, and doing in the world that at once contain a model of that part of the world it ostensibly pertains to and that also shape it every time it is used.

(This is what Kathy Acker is pointing to when the stolen software in *Empire of the Senseless* appears as a live, severed head.) Further, that each software element commonly interprets and remodulates what is understood to be the same, or a similar, process. For instance, the various takes on writing (plain text-editing, word processing, markup, and so on) presented by editors such as BBEdit, vi, Microsoft Word, LaTeX, etc.[23]

Whilst this domain of non-philosophical concepts is characterised as shameless and inane, it is unusual to find these materialists drawing such a concrete boundary beyond which creation and an experimental politics cannot exist. My impression, though, is that this is the result of a confusion, which can be read through conflicting tendencies in Deleuze and Guattari's own work. These should be read as pointers to problematics which certainly exist in the production of a theory of software. They are warnings, but ones that cannot be said to provide absolute stoppage to the inventive powers that lie in this area.

The tension between the approaches combined in their writings is clear. In terms of the wider field of electronic media, it is perhaps best seen in the way in which TV is described as a force that bridges the gap between the Althusserian models of repression and ideology, by offering simultaneous subjection and enslavement. That is, that viewers recognise themselves as the subject of interpolation of the television, but at the same time in a state of cybernetic submission to its sequence of switches, flashes of light, and bursts of input.[24]

Anyone who has watched CNN during the war over the monopoly on terror will know the moralistic slavery that is already presupposed of its audience by these broadcasters, the "we" that is called to order by its clatter of statements and opinions. What Deleuze and Guattari describe is clearly a tendency, an attractor, within media systems, but cannot be said to be a compelling description. Instead such theoretical positions need to be opened up.

Whilst they are almost useless in their direct characterisations of electronic media, the tools to do some of this opening up can of course be found in the same books. This is a characteristic of what

Robert Cooper calls their capacity to produce "generic,"[25] mobile concepts. In their writings on war machines—assemblages at any scale and of any type that attack or break free of total positioning systems—and their relationships to state formations, they note that

> (doubtless) the State apparatus tends to bring uniformity to the regimes, by disciplining its armies, by making work a fundamental unit, in other words, by imposing its own traits. But it is not impossible for weapons and tools, if they are taken up by new assemblages of metamorphosis, to enter other relations of alliance.[26]

Computers must be understood already as assemblages. In his *Lectures on Computation,* Richard Feynman notes research that specifies thirteen levels to an operating system. "This goes from level 1, that of electronic circuitry—registers, gates, buses—to number 13, the Operating System Shell, which manipulates the user programming environment. By a hierarchical compounding of instructions, basic transfers of 1's and 0's on level one are transformed, by the time we get to thirteen, into commands to land aircraft in a simulation or check whether a forty-digit number is prime."[27] Since the time of his writing, 1984, many more "levels" have become involved: The various protocols of interface, licensing, network, the ways in which computation has been coded and styled for various markets, are only a few examples. What is contended here is that any one of these levels provides an opportunity for critique, but more importantly for forms of theorisation and practice that break free of any preformatted uniformity. Since it is what they are further assembled with that determines their metamorphosis, it is the task of such practical and theoretical work to open these layers up to the opportunity of further assemblage.

Curiously, this is precisely the lesson that Deleuze and Guattari draw from another form of electronic media, the synthesiser. What is the "thought synthesiser"[28] that they suggest? By assembling modules, source elements, and elements for treating concepts (oscillators, generators, and transformers), by arranging microintervals, the synthesiser makes conceptualisable the philosophical process, the production of that process itself, and puts us

in contact with other elements of matter. In this machine composed by its materiality and force, thought travels, becomes mobile, synthesises.

Why, in their reading of the synthesiser, is there no dismay at humans merely providing a relay system between the variable actuations of a circuit board? It is certainly to pay attention to the wider assemblages which they form and are formed by. Because to describe the synthesiser as terminally as they do the TV would be to give up, to stop making a machine in the machine.

PRODUCTION

Instead of criticism, then—software criticism per se—what I want to suggest is that we pay attention to some practices within software production that emerge with and through thought out of whack with its simple reproduction.

Criticism proper, the self-abrogated privilege of judgement, is always predicated on finding itself absent from what it critiques. This true thought of the outside is that which can find no point of connection with what it surveys—except, that is, in pleasure in the announcement of its absolute corruption. Is anyone capable of such magnificent isolation? And this is why it is necessary to present some models of software production that contain engines for its theorisation. These are models that have arisen from work done over the last few years by a number of groups. No special claim is made that they exhaust any set of possibilities, nor that any of these models excludes characteristics given under another heading; they simply form notes on work going on.

CRITICAL SOFTWARE

One of the ways in which the currents described here first became manifest is in the creation of pieces of software designed explicitly to pull the rug from underneath normalised understandings of software. In 1957 Roland Barthes prefaced *Mythologies,* his collection of essays on the common-sensical mores of then-contemporary French bourgeois life, with the phrase, "Sarcasm is the condition of truth."[29] Nowadays there is no need to dispute sarcasm's unique

access to enlightenment. What is redundant now is any conditionality. Sarcasm *is* truth. Critical software is a voyage into that truth by means of its own devices.

What are the ways in which critical software operates? There are two key modes. First, by using the evidence presented by normalised software to construct an arrangement of the objects, protocols, statements, dynamics, and sequences of interaction that allow its conditions of truth to become manifest. This is the mode of operation of the installation "A Song for Occupations," which simply maps out the entire interface of Microsoft Word to reveal the blue-grey labyrinth in which writing is so happily lost. Richard Wright's CD-ROM *Hello World* takes a similar tack in making a comparative analysis of the interfaces and data structures—and consequent ways of knowing, seeing, and doing—of various video-editing and -effects packages such as Quantel, After Effects, and Flame.

The second way in which Critical Software may be said to exist is in the various instances of software that runs just like a normal application, but has been fundamentally twisted to reveal the underlying construction of the user, the way the program treats data, and the transduction and coding processes of the interface. Much of this work has been achieved in terms of games. Jodi's work on Wolfenstein and Quake is paradigmatic here, but there is a whole run of work, using mod files and patches, that can be seen in this light.[30] Additionally, there is a strand of work that has been cracked and messed with, by means of programs such as ResEdit, in order to gain access to its kernel of truth. The interfaces of standard software packages are rewritten.[31] Perhaps some of the actions defacing websites can also be said to belong to this current.[32] What this work does is make apparent the processes of normalisation operating at many scales within software—the ways in which, for instance, millions of separate writing acts are dedifferentiated by the various layers of a word processing program. By acting within it in a way that is both investigative and emetic, it points towards a move beyond the boundaries observed in simple institutional critique, towards other modes of creation. Not only that, but it performs the necessary task of allowing a negativistic maggot to

remain in all the golden apples of the two currents that follow, lest they be mistaken for a simply positive contribution to the empire of happiness.

SOCIAL SOFTWARE

Social software can provisionally be said to have two strands. Primarily it is software built by and for those of us locked out of the narrowly engineered subjectivity of mainstream software. It is software which asks itself what kind of currents, what kinds of machine, numerical, social, and other dynamics, it feeds in and out of, and what others can be brought into being.

The second strand is related to this. It is software that is directly born, changed, and developed as the result of an ongoing sociability between users and programmers in which demands are made on the practices of coding that exceed their easy fit into standardised social relations

In most cases, these two threads interweave. It is how they do so, how their multiple elements are brought into communication and influence, that determines their level of success.

I would like to suggest that Free Software can be usefully understood to work in these terms. It is a socio-technical pact between users of certain forms of license, language, and environment. The various forms of free or open-source software are developed as part of the various rhythms of life of software production. In addition, new social machines are invented to spawn the code, to diffuse and manage its development.

The pace and style of life in these forms of software development and diffusion can be understood to form their internal culture. For many, this is a functional utopia for coders, brought about by digital abundance. Much could be said about the way in which open-source code interrelates with the world of work—how class libraries function as a form of solidarity between programmers in minimising labour-time, but also how technical obscurantism is necessitated in order to maintain the caste privilege.

Thus, the second thread in this proposed conception of social software is partially met by the various strands of the open source movement. The ongoing sociability between users and programmers is there precisely because the users and programmers are one and the same. As is commonly acknowledged, this has provided the motivating force for the first stages of this movement. Why is Apache the best web server software? Because it is written by those who know these systems best.

But this has also formed a blockage to wider uptake of such systems. Free software is too internalist. The relation between its users and its developers is so isomorphic that there is extreme difficulty in breaking out of that productive but constricted circle. One way out of this is seen as finding ways in which free software can bring itself into communication with users who are not also its primary developers. This is crucial, but it is how it is done, and how it weaves this connection with the first thread of social software, that will determine its success. New imaginal and communicative capacities to enter into relations of becoming—of machine, technical, aesthetic, and social dynamics—are required. And it is here that free software now faces its biggest problem.

Free software taps into the dynamics of mutual aid, of shared resources, code conservation, and plagiarism, to get itself made. Now it needs to begin to set technico-aesthetic agendas which open and set flying the ways of sensing, knowing, and doing built into proprietary software. Death to bludgeoning pseudo-rationalism, and the feature-breeding world as office! Supposedly free software projects such as K Office are fundamentally flawed. They may have freedom in the sense of free speech, but this speech is not the result of free thought. Their composition is determined by a submissive relation to the standards set by Microsoft. This is a deliberate abdication of the imagination in dealing with the culture and structuration of all the kinds of work that take place in offices, a failure to take up the possibility of the reinvention of writing that digital technology offers.

In order to escape the impasse of open-source internalism, the developers of this mode of free software have attempted to connect to other kinds of users. But the users they are attempting to recruit

are precisely those formed and normalised by proprietary software. (By this I mean not the actual users of the software, but the models of them that are put into place by that software—and which it is therefore unable to distinguish and learn from.)

The mobilisation of free software by corporations is not my theme here, although what is perhaps most crucial but invisible in software—the model of life, the figuration of a user determined by these organisations—has yet to prove anything other than fundamentally entropic to innovation in these areas. The challenge to free software is that although it has massified its user base to some extent it faces the danger, not yet the actuality, of becoming conceptually stalled. This kind of reinvention might well be taken up by others.

One of the ways in which this is being done is via a mobilisation of elements in the first thread of social software. How far can the thinking about free software be opened by viewing itself as part of this wider tendency? One easy answer is that it allows the possibility of finding and communicating with users other than those modelled by pre-existing proprietary software. If the second thread of social software is born out of extended negotiation between users and developers, even to the extent that the difference between them is blurred, what are the ways we can ensure that that communication does not result in a closing back in on itself into another isomorphic circle? Primarily by insisting on the inevitable disequilibrium of relations between the user and the programmer. This is a political fact which cannot be avoided. Despite the fact that free software makes public the labour which is repressed from visibility under proprietary software, it is still the case that whoever is "closest to the machine" owns the space of possibilities which the relations have been established to explore.[33]

How can this disequilibrium be tipped over into a kind of movement other than that of absolute polar attraction by the "expert"? The first thread of social software offers us some routes into this problem. The answer is, inevitably, more careful work, more attention, more openness to difficulty and connection. We can only generate social software in its full sense through fundamental research into the machine, numerical, social, and other

dynamics that software feeds in and out of. However, these systems need to be understood in a sense expanded from that which software currently allows itself to know. The problem is not in recognising other forms of "expertise" and finding ways of accessing them. (We might consider as an opposite tendency the example of an artists' collective developing a city-mapping initiative in which they are only able to communicate with other "professionals" such as architects, critics, and theorists. Such is the stratified poverty of inter-disciplinarity.) There is a far more important need to recognise and find ways of coming into alliance with forms of intelligence that are excluded from the depleted culture of experts.

One of these, I would like to argue, is a poetics of connection.

There are ways in which technologies are taken over in ways that surpass product specifications. One of the most recent and notable examples is the use of the SMS protocol on GSM mobile phones. To manufacturers and network operators this cranky little texting facility was seen as a novelty, a little nothing, a gimmick. Instead, it has taken off and becomes what is well known today.

For many ostensibly radical theorisations of technology and media this is a problem. Perhaps we will always return here to a base–superstructure model: That is, property relations ultimately determine use. Under this rubric, there are two problem with texting, and with mobiles in general. First, the networks are centralised, running on a spoke-to-hub topology. They are owned by a multinational oligopoly. Second, their standards are not open: They cannot be accessed, improved upon, or reinvented except in compliance with the needs of these companies. This theory is able to account for why there has been no substantially innovative work by artists using mobile phones alone—there is no way of messing with the architecture. (It has to be collaged with other media systems in order to tease out new possibilities.[34]) And for this reason it is of fundamental use.

What it cannot account for is the way that this technology has been overrun and conceptually, if not infrastructurally, reinvented by hordes of what are seen as rather insignificant non-experts: teenagers, illegal workers, gossip-mongers, and so on. All of these subsist and thrive on their powers of connection, of existing in a

dimension of relationality rather than of territoriality. It is in their capacity to generate a poetics of this connection that they have reinvented this technology. (This is now a commonplace, of course, but only in retrospect. And as Sadie Plant notes, it was not even recognised as a possibility by those charitably concerned with widening access to networks like the internet.[35])

Such a dynamic has also formed the basis for the development of a piece of software, Mongrel's Linker.[36] This program is described more fully elsewhere, but it is essentially a small application that allows the fast authoring of multimedia collages. The software was developed by Mongrel to meet its needs for applications that can be introduced and used within a day or two. The functionality—when compared with the software used to create it, Macromedia Director—is massively stripped down. Instead of the interface being the usual grey windowed explosion of digital abundance, you get very little. The processing is shifted to the user. It relies on people's ability to generate narrative, political, melancholy, rhythmic, scattershot associations. It relies on the simple function of doing exactly what the name says it does—linking things. Here, the poetics of connection forms a techno-aesthetic and existential *a priori* to the construction of a piece of software.

This is a piece of software that has built itself up on learning from and through what occurs unofficially, the ways in which people, networks, drives, and languages coalesce to circumvent, parasitise, or overturn what codes, produces, and regulates them. Such an activity should not be understood as safely giving vent to an essential human need. It is pathological as much as anything else. But it is in paying attention to the way these dynamics work in particular instances, in acknowledging the intelligence built into them, that the potential for another form of software comes into view.

Poetics of connection is only one such dynamic. There are many others that could be worked into. The concept of social software, too, provides only something small, a little nothing. But with its two strands, in its necessarily unbalanced and mobile state, it provides another motor for creation, of the social as well as of software.

SPECULATIVE SOFTWARE

The best fiction is always also attempting to deal with the crisis of written language, in the way that it asks itself about the legacy built into text as the result of its birth in the keeping of records, in the establishment of laws, in assembling and managing tables of debt and credit. It does this perpetually, at the same time as reinventing and expanding upon the capacity of language to create new things. Speculative software fulfills something of a similar function for digital cultures. In Ellen Ullman's *Close to the Machine,* she states:

> I'd like to think that computers are neutral, a tool like any other, a hammer that can build a house or smash a skull. But there is something in the system itself, in the formal logic of programs and data, that recreates the world in its own image . . . We place this small projection of ourselves all around us, and we make ourselves reliant on it. To keep information, buy gas, save money, write a letter . . . We conform to the range of motion the system allows. We must be more orderly, more logical. Answer the question Yes or No, OK or Cancel . . . Then, slowly, we incorporate the whole notion of systems: We'll link registration data to surveillance,[37] to contract compliance . . . Finally, we arrive at a tautology: The data prove the need for more data! We think we are creating the system, but the system is also creating us. We build the system, we live in its midst, and we are changed.[38]

Ullman's book is the best account of the lived experience of programming that I've read, but I'm not quite sure who this "we" is. Perhaps it's the same "we" that always turns up when a voiceover speaks slowly over a heavy-concept TV documentary. There are pictures of traffic jams, mobile-phone users, nuclear power plants, cubicled workplaces, and ATMs, probably filmed in black and white, portentousness filters set to stun. The "we" is the "we" as in a tremulous, "What have we done to ourselves?" The "we" is an attempt to universalise rather than identify more pre-

cisely definable, albeit massively distributed and hierarchised, sets of conflictual, imaginal, and collaborative relations.

Elsewhere, speculative software has been suggested as being software that explores the potentiality of all possible programming. It creates transversal connections between data, machines, and networks. Software whose work is partly 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. That is to say that when, as Ullman suggests, the computer has "its own place where the systems and the logic take over,"[39] this is a place that can be explored, mapped, and messed with by a skewed application of those very same means.

In *Close to the Machine,* the narrator worries about a new payroll system that she's just been hired to work on:

> I'll wonder what I'm doing helping the IRS collect taxes. It will bother me that so many entities—employer, software company, bank, IRS—know so much about the simple act of someone getting paid for labour delivered. I'll think about the strange path of a paycheque direct-deposit, how it goes from employer to bank, company to company, while the person being paid is just a blip, the recipient's account a temporary way-station . . .[40]

Each of these entities—employer, software company, bank, IRS, employee—is composed by myriad interacting and agonistic relations. These blips, these events in software, these processes and regimes that data is subject to and manufactured by, provide flashpoints at which these interrelations, collaborations, and conflicts can be picked out and analysed for their valences of power, for their manifold capacities of control and production, disturbance and invention. It is the assertion of speculative software that the enormous spread of economies, systems of representation, of distribution, hiding, showing, and influence as they mesh with other systems of circulation, of life, ecology, resources—themselves

always both escaping and compelling electronic and digital manifestation—can be intercepted, mapped, and reconfigured precisely by means of these blips.

What are these blips? They are interpretative and reductive operations carried out on lived processes. They are the statistical residues of dynamics of association, escape, misery, acquiescence, and delight. They are not merely signifiers of an event, but integral parts of it. The figures in a bank balance, the links appearing in a web browser, are concrete arrangements, formations that determine relative degrees of potential movement within a specified level of analysis or use of a system. They have an implicit politics. Their aesthetics can be described as the result of the range of their potential combinatorial or isolatory capacity and its allowance of capture, invention, interrogation, or flight, the rhythms of peace or of compulsion that they put into place.

There are certain ways in which one is supposed to experience these blips. They are intended to mean that you are precisely broke at this time of the week, or that there are so many or no related web sites outside of the one you are currently viewing. Such statements, of course, are dependent on particular arrangements by which they can be made. Your wage statement is the cryptic blip that instantiates the enormous machine of class relations. A list of links is the result of a particular culture of association amongst a certain range of types of site, of which the site you are viewing is one instance.

These instances, these blips, are all manifest digitally. They can be picked out, mapped, arranged, examined, and placed in comparison with each other. Their modes of emergence and combination can be ascertained along with their conditions of repetition and change. The capacity of computers to perform these operations is what provides the fuel for speculative software—that is, software which refuses to believe the simple, innocent stories that accompany the appearance of these blips. Software that skews, misreads, and takes them for a little walk, but that not only reinterprets but leaves an invention of blips in its wake.

It is this capacity for invention and reinvention that is characteristic of digital abundance more generally, however little it is

taken up. What characterises speculative work in software is, first, the ability to operate reflexively upon itself and the condition of being software—to go where it is not supposed to go, to look behind the blip; to make visible the dynamics, structures, regimes, and drives of each of the little events which it connects to. Second, it is to subject these blips and what shapes and produces them to unnatural forms of connection between themselves. To make the ready ordering of data, categories, and subjects spasm out of control. Third, it is to subject the consequences of these first two stages to the havoc of invention.

NOTES

1. Pit Schultz made this suggestion as part of the preparatory work on the Software as Culture thread for Wizards of OS 2: Open Cultures and Free Knowledge, Berlin, October 2001. A version of this text was first prepared for that conference. Further information at: http://www.wizards-of-os.org/

2. Jeannette Hofmann, "Writers, Texts, and Writing Acts: Gendered User Images in Word Processing Software," in Donald MacKenzie and Judy Wacjman, eds., *The Social Shaping of Technology,* second edition (Buckingham: Open University Press, 1999), pp. 222–243.

3. Paul N. Edwards, *The Closed World: Computers and the Politics of Discourse in Cold War America* (Cambridge, MA: MIT Press, 1996).

4. Michael R. Curry, *Digital Places: Living with Geographical Information Systems* (London: Routledge, 1998).

5. Don Mackenzie, *Inventing Accuracy: A Historical Sociology of Nuclear Missile Guidance* (Cambridge, MA: MIT Press, 1990), and *Knowing Machines: Essays on Technical Change* (Cambridge, MA: MIT Press, 1996).

6. See, for instance, the work of Susan Leigh Star and others in *Cultures of Computing* (Oxford, Blackwell, 1995).

7. Michael Heim, *Electric Language: A Philosophical Study of Word Processing* (New Haven: Yale University Press, 1987).

8. Friedrich A. Kittler, *Literature, Media, Information Systems,* ed. John Johnstone, trans. various (Amsterdam: G&B Arts International, 1997).

9. An HTML version of the first edition of *Tools for Thought* is at: http://www.rheingold.com/texts/tft/

10. J. David Bolter, *Turing's Man: Western Culture in the Computer Age* (London: Penguin, 1986).

11. Martin Helander, Thomas Landauer, Prasad Prabhu, *Handbook of Human–Computer Interaction* (Oxford: Elsevier, 1997).

12. Alan Cooper, *The Inmates Are Running the Asylum* (Indianapolis: Sams Publishing, 1999). Cooper's approach is a particularly developed formulation of a range of procedures current in HCI's relation to users. A related set of processes for working through variable possibilities in interface and functionality is described, for instance, by Joy Mountford in "Tools and Techniques for Creative Design" in Brenda Laurel, ed., *The Art of Human Computer Interface Design* (Reading, Massachussetts: Addison Wesley, 1990) pp.17–30.

13. Of course, something of the sort is often done in product marketing, where potential customers are assumed to be able to identify with a range of typed user personalities. Phone companies use such approaches to sell tariffs and handsets. Such overt user-formatting is always responded to with the tactics of double-consciouness.

14. At http://www.wall.com/larry/

15. Ellen Ullman, *Close to the Machine* (San Francisco: City Lights, 1997).

16. David Gelerntner, *The Aesthetics of Computing* (London: Phoenix, 1998).

17. Of course, there are plenty of exceptions to this self-generalising statement. One of those that shows a way in which attention to the specificity of a particular technology is rewarded with great clarity is Bruno Latour, "The Berlin Key, or How to Do Words with Things," in P. M. Graves-Brown, ed., *Matter, Materiality and Modern Culture* (London: Routledge, 2000.) Latour's work here is

derived from a current of work, Actor-Network Theory (ANT), which, despite being specifically located in sociology, may well be of substantial use in developing a productive conceptualisation of software. A useful summary of the history of ANT can be found in the first couple of chapters of Mike Michael, *Reconnecting Culture, Technology and Nature: From Society to Heterogeneity* (London: Routledge, 2000).

18. Donald Knuth, "Theory and Practice," address to 11th World Computer Congress, San Francisco, 28 August 1989; archived as a TeX file at http://www-cs-faculty.stanford.edu/~knuth/preprints.html

Such an analysis might provide an insight into how CPU cycle allocation is made on the basis of hierarchies of tasks, which would inevitably contain models of the user. For a useful take on a related problem, see Harwood's "A Manifesto for Useless Art" at http://www.scotoma.org/

19. Elaine Scarry usefully introduces this term. Derived from grammar, it simply means a word that is both a verb and a noun, a thing and a motion. *Resisting Representation* (Oxford University Press, 1994).

20. Gilles Deleuze and Félix Guattari, *What Is Philosophy?,* trans. Hugh Tomlinson and Graham Burchill (London: Verso, 1994).

21. Ibid., p.10.

22. What can be seen as the beginnings of a useful theorisation of electronic media can be seen most clearly in Guattari's "Regimes, Pathways, Subjects," in Gary Genosko, ed., *The Guattari Reader* (Oxford: Blackwell, 1996), and also in Jonathan Crary and Stanford Kwinter, eds., *Incorporations* (New York: Zone, 1992).

The other text in which Guattari makes a real start on such work (but cannot of course be said to have this simply as his focus) is the chapter "Machinic Heterogenesis" in *Chaosmosis: An Ethico-Aesthetic Paradigm* (Sydney: Power Publications, 1995, p. 97).

Elsewhere, in this and other texts, Guattari simply makes passing references to themes close to the ideas of collective intelligence developed by Pierre Lévy, but also invests in the hope of reinventing a new kind of orality through machines (for instance, in

Nicholas Zurbrugg, *Postmodernism and Ethical Abdication,* and Genosko, op. cit., p. 115). Such technology has so far resulted in applications requiring very narrow sets of vocabulary, such as automatic telephone-answering or control of subsidiary dashboard functions in cars, but is of immense interest in terms of its potential to, for instance, reorganize language around archivable orality. (If full voice-recognition is developed, what are the implications for text? All linguistic data could be stored, searched, and cross-referenced as spoken word, with a potentially enormous effect on the way in which forms of speech, text, are currently valued, used, and ordered into hierarchies.)

The scope of the present essay is not a comprehensive philological examination of figurations of the electronic in Deleuze and Guattari, but it might be useful to point towards the material on music and synthesizers compiled by Richard Pinhas at Web Deleuze (http://www.webdeleuze.com/), and also their use of an information-theory model adapted from Rosenstiehl and Petitot to discuss technologies of social control in the "Rhizome" section of *A Thousand Plateaus*, op cit.

23. Thanks to Florian Cramer for a demonstration of vi which brought this sharply into focus. See also, "It Looks Like You're Writing a Letter: Microsoft Word" in this volume.

24. "For example, one is subjected to TV insofar as one uses and consumes it, in the very particular situation of a subject of the statement that more or less mistakes itself for a subject of enunciation ('You, dear television viewers, who make TV what it is . . . '); the technical machine is the medium between two subjects. But one is enslaved by TV as a human machine insofar as television viewers are no longer consumers or users, not even subjects who supposedly 'make' it, but intrinsic component pieces, 'input' and 'output,' feedback or recurrences that are no longer connected to the machine in such a way as to produce or use it. In machinic enslavement, there is nothing but transformations and exchanges of information, some of which are mechanical, others human.' (*A Thousand Plateaus,* p. 458). See also the brief section "If Literature Dies It Will Be Murder" in the interview "Mediators," included in Gilles Deleuze, *Negotiations,* trans. Martin Joughin (New York:

Columbia University Press, 1995), and the perceptive account of the way in which TV formats such as chat shows reduce thought, writing, and dialogue to a series of "positions" in Gilles Deleuze and Claire Parnet, *Dialogues,* trans. Hugh Tomlinson and Barbara Habberjam (London: Continuum, 2002).

25. Robert Cooper, "Assemblage Notes," in Robert C. H. Chia, ed., *Organised Worlds: Explorations in Technology and Organization with Robert Cooper* (London: Routledge, 1998) pp. 108–330.

26. Gilles Deleuze and Félix Guattari, *A Thousand Plateaus,* p. 402.

27. Richard P. Feynman, *Feynman Lectures on Computation,* Anthony J. G. Hey and Robert W. Allen, eds. (London: Penguin, 1996), p. 4. (The article he cites is P. J. Denning & R. L. Brown, "Operating Systems," *Scientific American,* September 1984, p. 96.)

28. Gilles Deleuze and Félix Guattari, *A Thousand Plateaus,* p. 343.

29. Roland Barthes, *Mythologies* (London: Paladin, 1973).

30. A program for the Mac through which the look of an interface, the text of dialogue boxes, and other more intricate resource allocations can be manipulated. An example of this mode might be Heritage Gold, a reversioning of Photoshop 1.0. A useful site on ResEdit is http://www.machacks.com/

31. Two sites monitoring and documenting this form of activity are: http://www.attrition.org and http://www.alldas.de

32. See for instance, the TextFM project to link users of SMS with a means of generating instant audio broadcast via radio: http://www.scotoma.org/TextFM/

33. In another context, the workplace, this "closeness" is meshed into a different set of inter-relations. The degree to which programmers have control over the rate and way in which they work, the way in which they define what is "possible," and the ways in which it might be achieved has of course been one of the key guarantees of their value as labour. Software production management techniques are developed precisely to counter and close down and rationalise such processes.

34. It is also clear that speculative uses of phones were being made by hackers and phreaks as soon as any new technologies or

routes into them became available, and for as long as they've existed in any form. How hacking can be understood to operate as a technico-aesthetic and perceptual activity with important consequences for the themes of this essay is developed amongst other places in Cornelia Sollfrank's Liquid Hacking http://www.obn.org/LHL/concepte.html and *Hacks,* a documentary by Christine Bader (1997). Info on this film at: http://www.choiproductions.com/

35. Sadie Plant, "On the Phone," Motorola, 2002.

36. The Mongrel web site is at: http://www.mongrelx.org/. Linker is available to download at: http://www.linker.org.uk/. A recent internet-based development of this software, called Nine, maintains its original features, but makes the code and the process of using it more available and open to development. To use or view Nine, check http://9.waag.org/.

A consideration of social software might also be made in relation to Piloot, a custom form of groupware whose development was led at the Society for Old and New Media, Amsterdam:j52

http://www.waag.org/

Another application that might well be understood on these terms is the essential, constantly updated database of reusable software serial numbers, Serial Box, and the program that it replaced, Surfer's Serials.

Perhaps there is something in that these pieces of software are focused only on combining a small set of functions and processes rather than acting as a metacultural factory typical of the large-scale applications. By being clear—or attempting to be so—about what they do, they can be perceived to a greater depth. There is no pretence to be anything but simple mechanisms, with a particular slant. It is perhaps this which allows their ready use or discarding.

37. Note, the specific forms of surveillance Ullman is referring to are workplace systems where logging-on prompts keystroke-counting, recording of web sites visited, etc. This form of worker surveillance forms an inverse of the kind of study that Knuth suggests.

38. Ullman, p. 89.

39. Ullman, p. 188.

40. Ullman, p. 188.