==

ANTITHESIS: THE DIALECTICS OF SOFTWARE ART

==

GEOFF COX

Antithesis: the Dialectics of Software Art
© 2010, Geoff Cox
Published by Digital Aesthetics Research Center, Aarhus University, DK
First written in 2006, Doctoral thesis, University of Plymouth, UK
Printed by Print-on-Demand-Worldwide
ISBN 87-91810-15-9

This publication has been made possible through the Digital Urban Living Research Center, partly funded by the Danish Council for Strategic Research grant number 2128-07-0011

A PDF of the book is available at http://darc.imv.au.dk/
Other reference materials available at http://www.anti-thesis.net/

==
ANTITHESIS: THE DIALECTICS OF SOFTWARE ART
==

GEOFF COX

Digital Aesthetics Research Center
http://www.digital-aestetik.dk/

========
CONTENTS
========

==========
0. PREFACE
==========

It is one of the assertions of the book that software art praxis can offer new critical forms of arts practice by embodying contradictions in the interplay between code and action. Contradiction is also embodied in the form the text itself takes, as both a conventional piece of academic writing (referred to herein as thesis) and a script written in Perl. Together, it presents an argument _about_ software art that is simultaneously an example _of_ software art. Crucially, both the thesis and the program can be interpreted and acted upon.

The book, derived from my Doctoral thesis completed in 2006, includes references to a number of other essays written during the time of study, as well as some key collaborative projects: notably the exhibition _Generator_, the publication _Notes Towards the Complete Works of Shakespeare_, and the project _The UK Museum of Ordure_ (details of all can be found in the 'references' section at the end of the book). Although referred to in the text, the projects do not illustrate the thesis but embody its argument. Similarly, the text is not a linked narrative to these projects but an example of software art practice in itself. In form and content, the thesis aims to express a dialectics of software art; expressing a move from in-itself to for-itself.

Finally, I would like to thank Roy Ascott and Andreas Broeckmann who supervised the thesis. Thanks also go to all those who have helped in its preparation and the production of related projects: Stuart Brisley, Rowan Green, George Grinsted, Joasia Krysa, Lau Thiam Kok, Mike Lawson-Smith, Alex McLean, Hugo de Rijke, Victoria de Rijke, Tom Trevor, and especially Adrian Ward. The publication of the book has been generously facilitated through the Digital Aesthetics Research Center, Aarhus University, following initial support from the Faculties of Art and Technology at the University of Plymouth for the research stage. As the text was first written in 2006, some details are a lttle outmoded in places, but I do hope its subsequent publication is not too late to be useful in some way to its readers.

GC

```
#!/usr/bin/Perl

print <<antiTHESIS;
```

==================================
the dialectics of software art
==================================

Geoff Cox

=================
1. *introduction*
=================

‘Beyond the words being read, others lie in wait to subvert and perhaps surpass them. Nothing any longer can be taken for granted; every word has become a banana peel. The fine surface unity that a piece of writing proposes is belied and beleaguered; behind it, in the realm of potentiality, a dialectic has emerged.’ (Mathews, in Motte 1998: 126)

An underlying assumption of this thesis is that software is not simply a functional tool but expresses wider cultural and technological processes that extend the critical potential of arts practice. In this way, the approach taken follows a critical tradition that aims to challenge the usual relations of production when making art using computers, where the efforts of the programmer are often hidden, and remain subordinate to that of the end-product or artist. Like the programmer, the source code that lies behind a software artwork also remains relatively hidden and consequently difficult to consider as an integral part of the work. The approach advocated in this thesis, in contrast, emphasises the process rather than the end-product - literally the ‘work’, not the object of art. Clearly this is not without precedence, but in the case of ‘software art’ the activity of work can be applied to both programming and the program (and even the result of these processes that in itself might be a further generative process), as well as the other material required for the program to run (which together is referred to as software). Indeed, software art might draw attention to any one or combination of these activities, but in general is considered to be not the artwork resulting from software, but software as artwork.

The thesis begins with a general overview of software art and culture (chapter 2), including some of the historical influences such as arts practice that have developed in the conceptual tradition and that draw upon systems theory. The chapter pays particular attention to the currency for the term software art, in contrast to the term ‘generative art’ associated with algorithmic and computer-based practices that tended towards an emphasis on more formalist concerns. Contemporary definitions of software art express both lines of continuity and discontinuity from these previous modes of creative production, but in general tend towards an emphasis on social context. Hence a critical practice in software art demonstrates the potential to be developed relative to the interactions of technical, cultural and political processes. Software art appears to be well-suited to comment upon the ways in which these processes increasingly utilise software but also ‘act’ like software. A critical practice in software art (something this thesis argues for) arises from these reflexive conditions, in which the form the work takes and its subject matter are entwined.

Software is always about itself in the sense that its source code both expresses what it will do, and does it, at the same time. That software appears to express this dual state of being and becoming, is analogous to a dialectical understanding of historical processes (chapter 3). These processes are thereby understood as dynamic and emergent phenomena that are analogous to the inner workings of software, and to systems in general that express ongoing processes of development and feedback. Historical materialism, rather than expressing a historical continuum, conceptualises the relation of the past to the present as a dialectical relation of ‘what-has-been’ to the ‘now’. To furnish this discussion, the dialectical method is described in more detail, and a position is adopted that stresses the importance of negation and the retention of contradiction, rather than any reconciliation at the point of synthesis. Like software, the approach suggests that nothing is finished or resolved but is in a continual state of becoming, appropriate to its emergent properties.

Building upon this dialectical understanding, the historical development of informational technologies can also be seen to express lines of discontinuity and continuity from previous modes (chapter 4) - discontinuity of technical form but continuity of capitalist logic. The contemporary network structure, for instance, although arranged in dynamic and distributed forms, can be seen to express control and feedback like any other system. A critical tradition that pays attention to systems, informed by both dialectical materialism and cybernetics, can be useful in revealing some of the new antagonisms that emerge. A problem arises in that the control expressed in complex systems remains relatively hidden, as it is expressed in ever more complex and 'immaterial' formations that obscure historical and material conditions, as well as the social consequences. Rather than an understanding of complexity and immateriality legislating against a dialectical approach (as is the case with much post-Marxist thinking), this thesis draws together a simultaneous understanding of complex systems and dialectical materialism, to take account of these new formations and provide critical insights into the power relations at work. This represents an optimistic turn for a critical art practice through software art, by drawing attention to the ways in which disorder can lead to a new sense of order (which leads to further disorder and so on).

In this way, dialectics is applied to an understanding of the conditions in which both the programmer and program can be seen to work, and produce artwork as software (chapter 5). Such attention to formations of labour derives from an understanding of its potential to express deeper social antagonisms, developed as a result of the increased collective and communicative nature of labour, and the recognition that social relations are embedded in humans and machines. The 'machinic' production of software presents a suitable case study in this respect, and the contradictions that arise from the open source movement and production of free software are particularly revealing as both a condition of, and reaction to, contemporary network social forms. Like the drudgery of work in general, the production of software is taken to be a negative condition under capitalism. Therefore a number of oppositional tactics are proposed, such as the refusal to work (non-executable code) or by working in a negative mode or unruly manner (dirty or messy code) outside the orthodoxy of passive working (analogous to proprietary models of clean and pure code). The argument is that software art holds the potential to make apparent contradictions within the relations of production, as well as be programmed to act in a disruptive manner itself, by calling upon its dialectical properties.

This way of working rejects determinism associated with software, for something far more speculative (and 'artistic' even). The final part of this thesis (chapter 6) examines the deployment of software in an artistic context, by concentrating on the work involved in writing code and the work that the code performs when executed. In the spirit of critical practices that seek to transform the technical apparatus, it further emphasises how dialectical thinking remains productive to understand how transformation is inherent to software. As a consequence, it is suggested that a critical practice in software art seeks to reveal these contradictions, with particular attention to source code as an expression of potential action. This performative dimension of code is important, as it emphasises the way that coding practices can break out of the 'means-end' chain of traditional software production and arts practice. In this way, a focus on coding practices, code, and the execution of code represents the privileging of potential - the potential that remains within and is ready to come into being. Therein lies the possibility of a critical practice in software art - and this is referred to as 'software praxis'.

As well as taking the form of a conventional research study, the thesis is simultaneously presented as software. The suggestion is that the critical potential of software art is demonstrated by the integration of theory and practice, in parallel to the way in which a practice-based submission for PhD is informed by theory. An understanding of software as art is extended to the presentation of a thesis as software, as something that can be read and that can be executed. Thus, this text aims to exemplify coding or programming as something that is to be executed or that embodies action, making a parallel between the production of the work itself (as both artwork and thesis) and the potential for its transformation. The text is both about software art and an example of software art.

On a technical level, this thesis has been produced using TextEdit 1.3 (v202) on a Macintosh PowerBook G4, running Mac OS X software (version 10.4.2) and saved as Plain Text (.txt).[1] It is simultaneously a Perl script or program. The program that interprets or compiles Perl code is typically called '/usr/bin/Perl' - hence the enhanced title of this thesis. Perl programs are generally stored as text source files such as this one, then translated or compiled into machine language by other programs (interpreters or compilers) at run-time.[2] The Perl script can be executed by typing 'perl' and the name of the file into a Unix command line shell (for more precise instructions on this, see note [3]). If run, the script will change text characters in the thesis and reposition them. Once the text has reached a critical state of disorder, the thesis will be published on the project web site and this version will be released under the Libre Commons License so as not to legislate against - and indeed to invite - its further development.[4] In the running of the program, this thesis expresses labour, action and new knowledge. In collapsing form and content, the aim is to position the thesis in terms of its reflexivity, and to exemplify the speculative potential of software art practice, rejecting fixed definitions or outcomes. Indeed the program performs, is thoroughly 'write-able', and reflects its intrinsic contradictory and potentially disruptive qualities, in the dynamic interplay between source code and action.

The thesis ends with a deliberate mistake. The last sentence contains no full stop, leaving the argument with an indeterminate ending. This is important on a functional level, as a full stop here would prevent the program from running. It also provides a dialectical composition that makes further reference to creative practices that employ constraints and then introduce an 'anticonstraint' to break the symmetry of the system. [5] The quote by Harry Mathews at the beginning of this introduction exemplifies this position, and how algorithms can be used to compose and decompose texts so as to demonstrate their latent dialectical potential. Such examples can be seen to inform the approach taken with this thesis, both in the absence of a full stop and more importantly by using the 'anticonstraint' software (antithesis) to break the system of constraints established by the finished work (thesis).

=================

2. *software art*

=================

‘Earlier much futile thought had been devoted to the question of whether photography is an art. The primary question - whether the very invention of photography had not transformed the nature of art - was not raised. Soon the film theoreticians asked the same ill-considered question with regard to film.’ (Benjamin 1999d [1936]: 220)

Software is not simply a functional tool but expresses wider cultural and technological processes and as such, holds the possibility of extending the critical potential of arts practice. In saying this, a contiguity is struck with previous technologies that threatened some of the founding principles of what constitutes art, how widely available it should be, and what purpose it serves. But whether software art is art is the wrong question to ask. Rather, contemporary practices and the emerging discourse in software art reveal not only how the nature of art may be subject to transformation but also that transformation is inherent to software culture itself. Accordingly, this chapter presents a general overview of software art and culture. In doing so, it stresses the ways in which technical, cultural and political processes increasingly utilise software but also can be seen to ‘act’ like software.

First of all, the term software needs some further description. Software refers to a computer program and the resources related to it that act upon the hardware of the physical machine components and machine. In more detail, this means software includes not only the instructions written in a particular language (such as Perl) as the program, but also the other materials required for it to run, that are usually combined for distribution. As a more general description, software is useful in this respect as it refers to the wider context within which the program runs. But it is also important to stress that it is the program itself as the source code that the computer executes. These instructions are loaded into memory, interpreted and then executed (or ‘run’) following the instructions as programmed until termination or an error is detected. Hardware is worked upon, and software performs the work. This link to performance also clarifies something about the use of the term ‘software art’, in describing not merely software used to produce art (a means to an end), but the software itself as the artwork (and this issue is something that the final chapter will return to in more detail). In other words, the programmers put the pre-existing hardware to work, in a similar way to artists producing concepts and manipulating materials in more traditional forms. There is little new in placing emphasis on process rather than end-product, but the assertion of this thesis is that software art exemplifies process-orientated practice in a way that lends itself to critical work appropriate to contemporary conditions.

Section 2.1 of this chapter defines the contemporary practice of software art, in comparison to the historical influence of ‘generative art’ and ‘computer arts’ practices of the 1960s and 70s. Clearly there have been many previous examples of artists and writers generating creative work in an algorithmic manner, using instructions and contraints, whether using computers or not. These principles have been especially explored in the parallel between program code and literature, suggesting an analogy with the linguistic distinction between syntax and semantics. Rules underpin all software practices, even those that seek to undermine these rules. Despite this, a general view seems to have emerged that older definitions associated with ‘generative art’ stress the formal rule-based and syntactical properties of software, and thus do not place sufficient emphasis on semantic concerns and social context. Although in general this thesis subscribes to this position and so adopts the term software art, it also retains formal concerns that are essential to understand the more cultural aspects and the generative or transformative aspects of software. It is argued that taken together, the terms generative art and software art emphasise contradictions inherent to both and between the two.

Section 2.2 elaborates upon the cultural aspect of software, emphasising some of the relatively hidden material concerns of software production; evoking a tradition of literary criticism and cultural studies. A closer engagement with culture reveals it to be an emergent, even generative, phenomenon that any criticism should address to utilise and develop appropriate methods that in themselves can adapt to changed circumstances. What might broadly be referred to as software criticism attempts to do this, in recognition of practices that acknowledge dynamic processes, structures and events that take place when software runs. Both operational understanding and more speculative inquiry are required to open up the possibilities for a critical practice in software art and culture - both following and breaking rules as the previous section suggested - that is critical of itself and that understands the historical conditions of its production. Examples are introduced here to establish software art as a coherent field of practice.

These practices build upon previous practices, and section 2.3 situates software art in the context of a broader art history and culture. There are many parallels to non-computer-based practices and this demonstrates the value of a historical perspective (but is not meant as a comprehensive history of software art). For example, in the 1970s and in parallel to the increasing visibility of computer technologies in culture, the term software was employed as a cultural metaphor to indicate a shift away from an emphasis on the (hardware) object of art. In this way, software art can be seen to operate in a conceptual tradition by placing emphasis on code as well as its execution, just as conceptual art's articulation of the 'dematerialisation' of the art object previously threw emphasis on the ideas and process of the artwork. Another clear historical influence on software art practice are Dadaist tactics, presenting a negation of the dynamic, transformational potentialities of technology and culture. However, these once radical practices now appear commonplace. For instance, techniques such as montage are now typified by computer techniques, and oppositional tactics ever more appear to run the risk of easy recuperation. It is suggested that the challenge for a dialectics of software art is to maintain contradiction in the process of transformation, for this is where politics is evident and where re-invention takes place.

2.1 - working definitions

Practices that combine the fields of art and technology have a complex history and employ a contested range of terms. The term software art has become popular to describe the contemporary artistic preoccupation with software production. Certainly ‘media arts’ is far too broad a description and one that would focus attention too heavily on the ‘medium’ and ‘mediation’ of software rather than emphasise its dynamic properties, processes and metaphors. Software art is clearly not just media art, as it expresses more complex processes than simply something mediated between sender, apparatus and receiver.[1] To contest whether the term or category software art is applicable, or useful even, illustrates part of the process of how terms that combine art and technology enter the public realm, and become normalised by festivals, conferences, publications, PhD submissions, and so on. This recuperative process is echoed in Alexei Shulgin’s comments about the term ‘net.art’, which became useless once it gained wider acceptance (1997). Whilst recognising this danger, the intention of this thesis is to stress that software art can remain useful as a practice and discourse, in revealing a range of contradictory tendencies in both art and software. The argument relies on the inherent character of software to express potentially disruptive qualities in the dynamic interplay between source code and action.

The recent attention given to software art is partly due to a range of cultural events that have provided critical consideration of the activity of programming and the materiality of code. Of particular importance are the _Readme_ festival and its associated _Runme_ software art repository in Moscow, Helsinki, Aarhus, Dortmund (2002-2005), and the _transmediale_ media arts festival in Berlin. Until recently, the transmediale festival included a category ‘artistic software’, and the jury statement of 2001 has become a key reference point for any definitions that have since emerged. For what was at that time a new festival category, jurors Florian Cramer and Ulrike Gabriel carefully drew attention to the structures of programming that lie behind the work (2001a). They argued this was part of a historical lacuna that tended to overlook the material and aesthetic aspects of software, predicated on the fact that programming code is inevitably a part of all art that is digitally produced, whether acknowledged

or not. Elsewhere too, Cramer has insisted that all digital art is software art in as much as it relies on, or is assisted by, other software to run, be it a browser, operating system, or network protocols (in Goriunova & Shulgin 2003). In this broad sense, if all digital art is software art, it could also be argued that all software is generative, in that it runs a set of processes and on execution a basic element is made to generate other forms and processes. The 2004 festival definition of software art written by Andreas Broeckmann and Cramer provided a useful description in this respect:
'Software > Generative Art: The Software category includes projects whose main artistic material is program code, or which deal with the cultural understanding of software. Thus, software is not understood as a functional tool serving the "real" artistic work, but as a generative means for the creation of "machinic" and social processes. Software art can be the result of an autonomous and formal creative practice, but it can also refer to the cultural and social meaning of software, or reflect on existing software through strategies like collage or critique.'

Interestingly both generative and software practices are included here, as this is an open call for festival submissions. Distinctions can be made between these terms, although Olga Goriunova pragmatically states that software art and criticism is justified in its very usage, by the fact that people find it a useful focus for discussion and to grant exposure to emergent practices. This is certainly the intention of the _Readme_ software art festival that she co-organised (with Shulgin); more particularly it aims to draw attention to works that lie outside mainstream festival culture and thus build an alternative community and discourse (in keeping with Shulgin's earlier comments on the fate of net. art). For instance, many examples of practice associated with the free software movement would not normally be considered within an artistic frame of reference. The festival therefore presents an alternative curatorial strategy, one in which people can submit works to an 'open' repository that is not selected or juried in a conventional manner. The associated _Runme.org_ software art repository is built upon an open database. Submitted works are contextualised, and this is not without difficulties particularly around categorisation, but in general the festival’s openness attracts a wider constituency than most, including demo-coders for instance, and those that subscribe to an open source ethos. As a result it suffers from a confusion of art and non-art codes but productively so. The working principles of free software are simply applied to its exhibition in the spirit of shared and collective development, in a manner that challenges the commodity status of art. The conventions of intellectual property, fees and prizes are substituted for the symbolic capital of the open source world.

Software art can be seen to challenge many of the precepts of arts practice, in the spirit of Walter Benjamin’s claim about how new techniques transform the very nature of art. In Bill Nichols’s ‘The Work of Culture in the Age of Cybernetic Systems’, this line of argument has been adapted to respond to issues around artificial life. He says ‘a presumption is made about a fixed, or ontologically given nature to life or art, rather than recognising how that very presumption has been radically overturned’ (1988: 37). His statement is no simple call for an end-of-art or end-of-life, but a redefinition of the terms and possibilities on offer. In any new definitions that emerge, there is some danger of making fixed definitions in a way that contradicts the very principles of software as something generative and non-definitive: work in progress. What is required is the constant redefinition of the terms and possibilities on offer.

# software either/or generative art

There are a number of competing definitions for generative art that establish correspondences between systematic and procedural approaches to production, across a variety of old and new media. In seeking to clarify what constitutes generative art, Philip Galanter’s definition is much cited and positions generative art as broadly rule-based:
‘Generative Art refers to any art practice where the artist uses a system, such as a set of natural language rules, a computer program, a machine, or other procedural invention, which is set into motion with some degree of autonomy contributing to or resulting in a completed work of art.’ (2003)

In a general sense, there is broad agreement that generative art is a term applied to artwork that is automated by the use of instructions or rules by which the artwork is executed. The outcome of this process is thus unpredictable, and could be described as being integral to the apparatus or situation, rather than a direct consequence of the artist’s intentions. Importantly, the description recognises that other agencies are at work, including human agency as an integral part of the production process in setting the rules. It is this line of thinking that informed the curation of the exhibition _Generator_ (2002/3), combining the work of artist-programmers and artists from a conceptual tradition who employ rules and instructions in their practice. The work of Alex McLean and Adrian Ward were presented in parallel to Sol LeWitt and Yoko Ono amongst others (the website contains more details on the exhibition [2]). All work was considered ‘live’ or performative in the sense that the artwork was generated from a process. To stress the point about agency, two examples are offered: Ono’s _Mend Peace for the World_ (2001), consisted of broken dishes from around the world and materials to mend them. The instructions, to be executed by those visiting the exhibition, were: ‘Keep adding more crockery as it gets fixed. Keep wishing while you mend.’ In contrast, McLean’s _forkbomb.pl_ (2001), was a Perl script designed to take a computer to its operational limit. A computer under such high load causes unpredictable results that pattern differently depending on the operating system it runs upon. Both examples - one tending towards reparation or reconstruction, the other towards destruction - emphasise a rejection of what one might refer to as ‘software-determinism’. They demonstrate how the producer can concede control to some extent - and this is an important qualification - over the production of the work but that human intervention is paramount to (software) production. In other words, the artwork is necessarily programmed - with or without the aid of a computer. Whether the artist was involved in the writing of the software or not is beside the point. Someone was.

In contrast to what has been said about these examples from _Generator_, much of the work in the field of generative art stresses issues of unpredictability and autonomy rather differently.[3] Defining generative art in 2003, John McCormack adds the influence of biology and emergent behaviour, and in particular the terms 'genotype' and 'phenotype'. He argues that software can be seen in terms of 'genotypes' (DNA in cells) as machine code, and 'phenotypes' (the higher level form of behaviour) as what happens when it runs. The programmer would set the parameters that defined the fitness, and the software would evolve 'autonomously'. Put simply, McCormack generalises that the authoring process is directed towards a genotype as the specification of a process, and when this process is executed it generates the phenotype as the 'experience of the artwork' (in Brown 2003: 5). It is worth noting the position of the artists in this description as responsible for the DNA of the artwork in the perpetuation of a 'creationist' myth (which chapter 4 will dispute, along with biological determinism in general). Clearly other external factors are at work in creative production in art and life.

In his essay 'What is Generative Art? Complexity Theory as a Context for Art Theory' (2003), Galanter also refers to generative systems as displaying emergent behaviour, but interprets 'autonomous systems' at far too literal a level when he claims: 'Generative art is ideologically neutral' (2003). This simply cannot be the case if such a process is already seen to be programmed. Any claims of neutrality ironically only serve to prove the point of how ideology works, even in the descriptions that deny its very presence. Galanter's definition of generative art is an eclectic one, contributing to wider discussions around 'not just art' and cultural practices which allows for the inclusion of algorithmic composition, as well as practices that do not necessarily involve computers at all. But to Inke Arns, this is part of the problem as the definition is far too inclusive, applied across many fields of practice that focus attention on the end-product of a process. She quotes Tilman Baumgartel's article 'Experimental Software' (from 2001) to stress the distinction between earlier work using computers and software art, where the latter is:
'... not art that has been created with the help of a computer, but art that happens in the computer, software is not programmed by artists in order to produce autonomous artworks, but the software itself is the artwork.
What is crucial here is not the result but the process triggered in the computer by the program code' (in Arns 2004: 184-5).

Both Galanter and McCormack’s statements do appear to verify an emphasis on end-product as opposed to Baumgartel’s emphasis on process. Arns is additionally thinking of statements such as: ‘the aesthetic value of code lies in its execution, not simply its written form’, and takes this to foreground execution (see Cox et al 2001). Admittedly there is a danger of emphasising the formal and syntactic aspects in using the term generative art. This statement was intended to emphasise the interaction between source code and its executed form. The emphasis on execution is thus a description of process, wherein the end-product remained a by-product. An example cited was McLean’s _forkbomb.pl_ (2001, described earlier in this section), to argue for the aesthetic appreciation of source code in parallel to a visualisation of the process when run. This is in fact how it was exhibited as part of the _Generator_ show, with the source code as an integral part of the work. Nevertheless, the criticism Arns is making is that the privileging of execution, even if in combination with source code, avoids some contemporary practice associated with software art. She is thinking of programs that are not necessarily executable, or executable only on a conceptual level. Following these remarks, the earlier statement that all software is generative should be further qualified by adding that this applies in the most abstract of ways (for instance, with a concept). Any definition of generative art requires improved description to shift emphasis from the object generated to the process of generation. To describe this generative or transformative aspect and in order not to dismiss it out of hand, more historical detail is required to stress its importance to an understanding of cultural activity using software, and of course to an understanding of software art.

# generative grammar

To generate something accounts for most creative activity in a very general sense. A more specific use of the term in relation to arts practice can be traced to a lecture ‘Generative Art Forms’ (presented at the Queen’s University, Belfast Festival) in 1972, by the Romanian sculptor Neagu (who also founded a Generative Art Group).[4] A more common reference is Noam Chomsky’s _Syntactic Structures_ (1972), first published in 1957, often cited as the source of the concept ‘generative grammar' (sometimes referred to as 'transformational grammar'). Chomsky assumes that somehow grammar is given in advance ('hard-wired') and therefore human consciousness contains innate grammatical competence that is pre-social (1972: 85).[5] This explains his interest in 'syntactic structures' by which sentences are constructed in particular languages to understand the properties that underlie successful grammars (1972: 11). This is an abstract endeavour to discover broad principles that can be applied to languages in general and thereby provide a method that can be applied to specific languages.

These concerns have also been the inspiration for much artistic experimentation using computers, as it lends itself to the procedural qualities of programming as an expression of transformative grammar.[6] An example is Bill Seaman's _The World Generator_ (1996) that generates emergent meanings by enabling users to make choices from a spinning interface of different media-elements and processes (including objects, images, texts, music and movies). His claim is that this 'techno-poetic mechanism' generates a poetics that is extended by computer-based technologies, becoming what he calls 'recombinatory poetics'. As a consequence, Seaman asks whether this constitutes a new form of writing or a 'new form of evocative exchange which cannot be defined in terms of past linguistic discourse?' (1999). Like Chomsky's work, this approach might be seen to suffer from universalism. Both the machine and consciousness in these examples are taken to follow rules at the risk of diminishing other factors, such as social interaction that is essential to the expression of action in the world.[7]

More commonly cited in connection with recombinatory work is the 'Ouvroir de Littérature Potentielle' (OuLiPo), a group of writers and mathematicians founded in 1960 by Raymond Queneau and François Le Lionnais. Their concerns were syntactic rather than semantic, concerned with contraints 'brought to bear on the formal aspects of literature: alphabetical, consonantal, vocalic, syllabic, phonetic, graphic, prosodic, rhymic, rhythmic, and numerical constraints, structures, or programs' (Le Lionnais, in Motte 1998: 29). An example of automatic transformation of text is Jean Lescure's 'S+7' method in which a text is taken and each word ('s' for substantive) is replaced by the seventh following it in a dictionary. Queneau points out that if a 2000 word dictionary is used, the 'S+2000' method would produce an exact copy of the original (Motte 1998: 61). In this sense, and according to Georges Perec: 'the Book is a cryptogram whose code is the Alphabet' (in Motte 1998: 96). Rather than a chance operation (such as in the work of John Cage), Oulipean texts are generated through the use of constraints or rules, wherein any ideas associated with freedom of expression is undermined.

A further Oulipean example that lends itself to computation is Queneau's _Cent Mille Milliards de Poemes_ [one hundred thousand billion poems] (1961) in which ten sonnets can be arranged according to formal rules. To each of the ten first lines, the reader can add any of ten different second lines, and so on. The sonnet has fourteen lines, so the possibilities are of the order of 10 to the power of 14, or one hundred trillion sonnets. Le Lionnais makes a claim for the significance of this in terms of technical superiority: 'the work you are holding in your hands represents, itself alone, a quantity of text far greater than everything man has written since the invention of writing' (Motte 1998: 3). Potential writing in this sense implies the impossibility of its potential reading – and both are exponentially bound. The full potential of this work lies unrealised for practical reasons, perpetually in a suspended state of its further reading. In an experiment to exploit the potential of the computer, Paul Braffort was commissioned to program some of the OuLiPo works, such as Queneau's _Cent Mille Milliards de Poemes_. In describing this enterprise as 'algorithmic literature', Paul Fournel argues that the machine allows the author to dominate the existing relations of computer, work and reader in new ways (Motte 1998: 140-2). Originality is clearly not the point in this work; originality is mentioned in connection to mark a distinction from other practices such as the work of the 'algorists' (associated with Roman Verostko) who explore the 'form-generating' possibilities of algorithms but at the same time their originality, in a way that parallels the conservative paradigm of originality in arts practice (2004).

Creative endeavour is seen to be programmable, and is considered in terms of its execution. But far

from a deferral of authorship, the computer offers new potentialities in this way. In ‘Prose and Anticombinatorics’, Italo Calvino demonstrates the potential of the computer in serving this purpose, proposing that:
‘... the aid of the computer, far from replacing the creative act of the artist, permits the latter rather to liberate himself [sic] from the slavery of a combinatory search, allowing him also the best chance of concentrating on this “clinamen” which, alone, can make of the text a true work of art’ (in Motte 1998: 152).[8]

Thus, the potential for permutations or ‘combinatorics’, what Le Lionnais calls a ‘combinatory literature’, is expanded greatly by the computer and its systematic compositional structure. This is further developed by Cramer’s web site _Permutations_ (1996-2000), which reproduces combinatory text systems, such as those of Queneau, in digital form. Many programmers would deny the ambiguity of expression in their work - it either works or does not in logical terms - but clearly there are poetic elements in code. The programmer Donald Knuth makes this apparent by pointing to programming as an ‘aesthetic experience much like composing poetry or music’ (1981: v). In _The Art of Programming_, he draws analogies to formal experimentation in literature when he produces codes to guide the reader through a series of workshop exercises. In the section ‘Procedures for Reading This Set of Books’, the instructions for reading the book are arranged in an algorithmic form that directly addresses the reader:
‘1. Begin reading this procedure, unless you have already begun to read it. Continue to follow the steps faithfully; [...] 5. Is the subject of the chapter interesting you? If so, go to step 7; if not, go to step 6. 14. Are you tired? If not, go back to step 7; 15. Go to sleep. Then, wake up, and go back to step 7.’ (1981: xv-xvi)

Work, such as this, using executable formal instructions, makes explicit the idea of software as potential literature, whether running on a computer or not. The analogy to language is further expressed in terms of input-output of data (abbreviated to ‘I/0’), with the input often referred to as reading and output as writing - hence the common use of the description ‘read me’ for explanatory texts). This is a similar

approach to Calvino's 'How I Wrote One of My Books',[9] referring to his own novel _If on a Winter's Night a Traveller_ (1981), in which he produces an algorithmic description of the book's structure:
'The reader who is there (L) is reading the book that is there (l); The book that is there relates the story of the reader who is in the book (L'); The reader who is in the book does not succeed in reading the book in the book (l'); The book that is there does not relate the story of the reader who is there; The reader who is in the book claims to be the reader that is there [...].' (1995)

It reads like a source code to the earlier novel. It is worth emphasising that an engagement with this thesis operates overtly in terms of reading and writing: it is a 'read me' (the text) but also carries the invitation to 'run me' (the program). Indeed all conventions of writing and reading, of both text and code, have in common that they are part of a set of abstract (coded) systems of input and output. In so-called natural languages , this is limited by the numbers of phonemes (letter-sounds), arranged in strings (which may be a sentence) or finite sequences with sentences - although the sentences are infinite. In other formalised systems such as programming, the logic adheres to rules and so can be considered a language like other artificial languages, with its own particular grammar that generates its grammatical sequences. What is significant about the work of the OuLiPo group is their tendency to follow an algorithm to the rule and then break it, introducing a flaw in the system to disrupt the symmetry: 'because when a system of constraints is established, there must also be anticonstraint within it. The system of constraints - and this is important - must be destroyed.' (Motte 1998: 20)

Programming languages can clearly be seen in terms of their grammar and syntax but also their poetic spatial arrangements, sometimes referred to as 'code literature' or 'code poetry' (typified in 2004 by an exhibition to assess the aesthetic implications of digital poetry: _p0es1s: Digitale Poesie_, held at the Kulturforum Potsdamer Platz, Berlin). The Perl programming language has often been used to 'port' other poems, such as Eric Andreychek's _Jabberwocky_ as seen in the Perl Poetry contest of 2001. While the output of the poem is not significant, the three characters of the poem are represented by three dysfunctional processes if the program is executed. It does not exactly crash the system it runs on, but does express the potential of algorithms to both compose and decompose texts.

## generative both/and software art

In the many comparisons between software art now and the older practices associated with generative art, McCormack explains one key difference was that in the 1960s and 1970s artists simply had to write (or ask someone to write) their own software in order to generate the outcomes (in Brown 2003). The now wide availability of authoring software has changed the conditions for the production of software art by the artist-programmer. It is with some of these issues in mind that Richard Wright traces the 'divergence between programmers and program users', based around the issue of whether a computer is considered a medium or a tool (2004). In a hierarchy of programming languages, Wright points out that not all programming practices are equal. He is thinking of the predominance of scripting languages such as Flash Actionscript (but also Lingo, Perl, MAX, JavaScript, Java, C++, as well as other programming and scripting languages) that use libraries of functions and a certain shared, if not prescribed, vocabulary of styles.[10] For Wright, this changes the terms of the discussion from a general issue of artistic programming to one of what kind of programming is being used. He cites the historical shift in Harold Cohen's practice from a painter to developing software to automate his artwork, through the use of what Cohen refers to as 'autonomous machine (art making) intelligence'. Developed from 1973 onwards, the _AARON_ program represents to Wright the historical transition towards contemporary culture, where the use of computers has become pervasive. As a result, the terms of practice have fundamentally changed for the artist-programmer. His argument is that:
'In a world where artists use software to write software that will be seen by virtue of other software, questions about the "aesthetics of the code" become a symptom of not being able to see the wood for the trees. Programming is not only the material of artistic creation, it is the context of artistic creation. Programming has become software.' (Wright 2004)

The distinction refers back to earlier practices that were characterised by artists working at the meta-level of programming – and there is even a certain amount of nostalgia here that somehow the early 'avant-garde' of computer arts has been forgotten or not fully appreciated. Alan Kay refers to artists working at the level of the 'metamedium'. The idea that artists using computers should engage with programming at a deep level is a position that many computer art education programmes propagated in the 1970s and 1980s – for instance, at the Slade School of Art and Middlesex University in the UK, with which Paul Brown and Wright were associated.[11] However programming in Cohen's work operates in a rather ambiguous relation to the overall artwork. Clearly in a general sense it is part of the artistic output but more in terms of a representation of his skills and technique, rather than as a constituent part of the artwork as such. The emphasis tends towards the completed work of art rather than the program or programming being a work in itself.

In contrast to Cohen's work, a more contemporary reference that situates software art overtly in terms of programming is the exhibition _CODeDOC_, first for the Whitney Museum of American Art's 'artport' web site (2002), and later at Ars Electronica (2003). The curator, Christiane Paul, set the invited artist-programmers an instruction to 'connect and move three points in space' in a language of their choice (Java, C, Visual Basic, Lingo, Perl) and to exchange the code with the other artists for comments. For example, in Rainer Mandl and Annja Krautgasser's _Pedigree_ (2003b), the Oedipal drama is revealed in the source code of the work. The three points connected in space represent the three protagonists of the myth – father, mother and child – who play their parts in the generative narrative. The viewers of all the works in _CODeDOC_ were invited to first read the written code and then see the executed work. This raised some controversy on mail lists at the time, for deliberately obfuscating or aestheticising code to non-programmers, rather than demystifying the creative process. Yet the significance is that code is taken to be part of the work and not simply meant to assist interpretation. The curatorial statement contains a number of useful comments on the intentions of the experiment and reiterates the potential of software itself as artwork:

‘In software art, the “materiality” of the written instructions mostly remains hidden. In addition, these instructions and notations can be instantaneously activated; they contain and - further layers of processing aside - *are* the artwork itself. While one might claim that the same holds true for a work of conceptual art that consists of written instructions, this work would still have to be activated as a mental or physical event by the viewer and cannot instantaneously transform, transcend, and generate its own materiality.’ (Paul 2003b)

This parallels some of the earlier curatorial decisions for the _Generator_ show, in particular McLean’s _forkbomb.pl_ where the source code was exhibited as an integral part of the artwork (see section 9.1). Whereas formerly artists had to engage with programming in early computer arts practice, the lack of necessity now allows for other social issues to be engaged (just as previously the invention of photography freed painting from figurative representation). In linguistic terms, artist-programmers appear to have shifted their attention from an engagement with the syntax of programming to semantic concerns.[12] This is indeed how Cramer makes the distinction between generative art and software art, by associating the former with syntax and the latter with semantics (2003). But this is not simply a shift from one to the other. Syntax, although not concerned with meaning in itself, certainly has implications for semantics, and both are required to inform an overall theory of language. The programmer Larry Wall clarifies this in relation to the programming language Perl and the wider cultural concerns that arise: ‘A language is not a set of syntax rules. It is not just a set of semantics. It’s the entire culture surrounding the language itself.’ (in Flor 2002) Yet what Cramer is trying to emphasise is a shift in software art from ‘pure syntax’ to ‘something semantic, something that is aesthetically, culturally and politically charged’ (2003). It is not a choice of one or the other but a change of emphasis.

Therefore, although this thesis adopts the term software art, it aims to retain the implicit generative aspect as an evocative technical and cultural process. The apparent dualism between generative art and software art is also something that Mitchell Whitelaw disputes in questioning the binary relation of formalism (associated with generative art) and culturalism (associated with software art). Rather than seeing this as an impasse, as Troels Degn Johansson does in ‘Mise en Abyme in Software Art’ (2004), Whitelaw suggests a ‘complementarity’ of positions that leads to alternative modes of being and relation (2005: 138). He calls this ‘critical generativity’ to stress the emergent and transformative properties that reflect social complexity and software’s latent cultural agency (2005: 152).

The dialectical approach of this thesis also argues for new critical forms, but rather than seeking Whitelaw’s complementarity or fusion, argues for a contradictory relation. That said, the competing definitions matter little in themselves but only in as much as they operate in terms of an overall contribution to a critical discourse around the practice of software art. Software includes, if only on a conceptual level, a generative process in which something is always ready to come into being, however latent. It is for convenience only that this is referred to as software art. It does so in recognition that these debates are appropriately in flux, and entirely open to contestation as part of the ongoing development of a critical discourse in software art and culture.

2.2 - software materialism

It is clear that software is a thoroughly cultural and not simply technical phenomenon. But in discussing ‘software culture’ and its critical potential, the term culture is far from uncontested territory. Culture is a notoriously ambiguous concept and even in terms of its scientific usage describes both a process (artificial development of microscopic organisms) and product (the organisms produced) (Williams 1998). [13] An understanding of culture as both process and product is useful for software culture, as it stresses the issues introduced in the previous section and some of the criticisms of practices that privilege product over process. Any cultural product such as this thesis cannot be divorced from the materials and institutions that produce and disseminate it. Otherwise the final text is privileged over the cultural form that it takes, separating it from the reality of its social production and material conditions. This principle underpins subsequent chapters and the overall form this thesis takes.

This section aims to introduce software production as a cultural and material activity, and provide some examples of software art that are produced with these ideas in mind. This description is in keeping with what Raymond Williams called ‘cultural materialism’ (as an elaboration of ‘historical materialism’ that chapter 3 introduces in detail) to emphasise that cultural production is itself material, as much as any other human activity. Rather than simply see culture as influenced by its underlying system of production (as in orthodox Marxism), cultural work is taken to be political because social processes are always embedded in it - in the most ordinary aspects of everyday life and in cultural practices. This is important as it suggests that cultural activity has transformative possibilities, of not just understanding social processes, but of an active involvement in their potential transformation. This transformational aspect relates to Pierre Bourdieu’s concept of ‘cultural reproduction’, describing both the determinacy of social structures in which dominant values are reproduced but also the agency inherent in the practice of social action. To Chris Jenks, interpretations of reproduction have tended to over-concentrate on the determining aspect of the metaphor at the expenses of its regenerative properties (1993: 2). The dynamical aspects of culture and of cultural production are in danger of being overlooked.

In engaging with software culture, this section engages with the dynamic technical processes associated with software, whilst at the same time recognising that culture and criticism are themselves dynamic. This is clearly an important criticism for the production of software art, or any form that strives to make explicit the operating system in which it runs. What is required is an understanding of cultural aspects, as well as the complex interactions and processes at work at a deeper level of operation that does not privilege execution or end-product. This calls for a ‘software criticism’ that takes account of practice rather than operating at an abstract theoretical level.

# software cultural criticism

A working principle has been established in the previous section: that any terms of reference are not definitive but only function as ideas in progress for further development. Criticism of the terms under discussion is an expected part of any critical work. The parenthesised subtitle of Matthew Fuller's essay 'Behind the Blip' suggests as much, reading: 'some routes into "software criticism," more ways out' (2003). Stressing software criticism that does not operate at a distance from practice but that takes account of practice, Fuller offers three categories towards a strategic definition. The first of these is 'critical software' designed to undermine normalised understandings, operating through two key modes: 'by using 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'; and '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' or even by adapting or hacking into existing software (2003: 23). He sees this as extending ambiguities built into the software itself, and perhaps all software is contradictory in this way.

An example of critical software, and one much discussed, is Signwave's _Auto-Illustrator_ (2000), that defies user expectation as a parody of the vector graphics design software Abode _Illustrator_. It looks like and indeed works like conventional commercial software, but carries some extra auto-generative functionality that render designs outside of the direct control or creativity of the user. Cheekily included in early releases was a license agreement that indicated that any designs were necessarily co-authored by the company Signwave who supply the software (aka Adrian Ward). Here, the parody operated particularly effectively, as some users were outraged that a company would insist on such a clause in a direct assault on their creative and intellectual rights. It highlights the issue that full authorship is rarely acknowledged in making art using software, as is the labour of all those involved in the process. The software was released as a boxed version for the exhibition _Generator_ with a 'User's Manual' that contained both technical detail and critical essays (2002). In this way, the commercial packaging added a further layer to its ironic critique of the commodification of art, and software as art.

The second of Fuller's categories is 'social software' built by and for those excluded from commercial software production, providing a subculture of software production with a different agenda. Related to this is software developed and changed through social networks of users and programmers, that emerges from a different set of social relations than the orthodoxy of software production. It is this separation from the mainstream that situates Fuller's use of the term, outside the usual description of software that simply connects people or allows for collaboration (such as the 'social software' group at MIT's Media Lab for instance). His example is Mongrel's _Linker_ (1999), that might be updated to the more recent _Nine(9)_ (2003). Both allow communities of users to form online collaborative archives (or 'knowledge maps'). In these examples, sociality goes beyond the software itself to the communities and individuals who use it, and who further develop it as a project. In a more general sense, the free and open-source software movement are examples where developers form 'a socio-technical pact between users of certain forms of license, language, and environment' (2003: 24). In this scenario, open source software development and relations of production present new configurations and contradictions of labour-power and criticism. The labour invested in producing the software is made public, unlike proprietary software but the control of the means of production is still managed according to capitalist principles. Also in this way, software is developed by a fairly closed community of 'co-producers': those actually using it and with the ability to make and change it. But do they mistakenly continue to exploit their own labour by not selling it? Clearly this is a much longer discussion about the politics of free software and its take-up by large corporations (an issue that will be returned to in chapter 5).

For Fuller, the problem lies in the closed loop (what he calls 'open-source internalism' 2003: 25) between developers and users: only when they are one and the same does this system actually work for mutual benefit, and therefore it needs to be expanded to be more widely available to other users. This point could be applied to the use of the operating system Linux, where the benefits of free software simply cannot be entertained without adequate instruction. The 'culture of experts' needs to be broken down, as Fuller puts it (2003: 26). Having stressed this point, there are numerous examples of projects that directly address this issue of access to skills and technologies. For instance, the Redundant Technology Initiative are an example of many groups that recycle redundant computers, install Linux and free software and train people to use them. Related to this but offering even more specialised open source knowledge are the Unix workshops as part of the free education initiative of the Faculty of Unix at the University of Openness, in London. Both examples stress that social software needs to ensure it operates inclusively and only then can genuinely be seen to be 'open' and 'social'. To do this, a critical approach needs to be developed that takes account of the layers and processes involved on a technical level and in relation to social context.

Exploring the potential for new forms of software, Fuller's third category is 'speculative software' that creates new connections between data, machines, and networks. He describes this as the 'reinvention of software by its own means', in using software to make software about software: 'Software whose work is partly to reflexively investigate itself as software, software as science fiction, as mutant epistemology.' (2003: 30) By breaking with conventions of production and criticism, some of the antagonistic social relations between the different agencies involved in software can be made visible. Fuller describes these potential spaces as 'blips', and this is where politics lies (behind the blip). As has been demonstrated in the previous section, the structural qualities of code lend itself to poetic forms, but speculative software offers the additional potential for new forms of critical practice. Written in 2001, Harwood's translation (or 'porting') of William Blake's poem _London_ (of 1792) into Perl, is a notable example of software art that is more than simply a formal exercise (2005: 151-8). In both old and new versions, statistics and the modulation of populations are used for social comment, but in Harwood's version material conditions are registered more overtly as both content and form. The politics of

Blake’s poem describing the social conditions of London are translated to a contemporary cultural and technical reality in which people are reduced to data.

The example demonstrates the potential to extend the expressive potential of programming and to develop critical forms that are reflexive - both being and becoming software. It is speculative software that arguably comes closest to what can be understood as an artistic approach to software (according to Broeckmann 2003), and one that particularly informs the approach to this thesis in revealing practices that use the formal qualities of programming to express how structures can be manipulated and reconfigured. Perl poetry such as _London.pl_ indicates how the concept of change might be embedded in the process of making programs. This speculative approach to software art and criticism makes reference to earlier critical modernist practices that engage with the apparatus of production and the materiality of language.

# code materiality

The formal qualities of language have influenced subsequent approaches to software art. The assumption is that language constitutes the determining model of all other signifying systems, as it is inherently rule-based and contains a finite number of elements. There is a history to this ‘formalist’ position typified by Vladimir Propp’s ‘Morphology of the Folktale’ (1927) that demonstrates a structural analysis or an algorithmic approach to criticism in which a universal formula is proposed.[14] Formalist experimentation in literature also follows this logic in which a text can be seen to be autonomous from the act of writing - a situation in which writing writes, not writers. Tzvetan Todorov in _Littérature et signification_ (1967) explains: ‘Every work, every novel, tells through its fabric of events the story of its own creation, its own history [...] the meaning of a work lies in its telling itself, its speaking of its own existence’ (in Hawkes 1986: 100).

The quote describes a situation where language itself has autonomy over the writer, where words are arranged in such a way that subjective intention does not appear to figure. This formalist or structuralist position is further developed by Roland Barthes's essay 'The Death of the Author' (1977) wishing to make the reader no longer a consumer but a producer of the text. The 'death' is a metaphoric gesture: 'the birth of the reader must be at the cost of the death of the Author' (1977: 148) as an expression of the author's inability to claim the privileged source of meaning or value of a work of art.[15] But where does authorship lie in all this? It has not simply disappeared but is recast in recognition of its own constructedness. To state the obvious, even Barthes is the author of his own position on the author's disappearance.[16] Text and code are both written and write.

Furthermore, the programmer or writer is intimately connected to the writing machine - be it book or computer. The material form this thesis takes is an attempt to draw attention to these issues as both a text and program script. This is also what N. Katherine Hayles attempts in _Writing Machines_ (2002), a book that draws attention to its material properties.[17] The production of literature is both material and immaterial in other words, expressing both the physical reality of writing and reading the book, as well as the imaginative world that the book depicts. A written text can bring into view the technical apparatus or writing machine that produces it, such as in the case of something typewritten, where the marks of the letters are imprinted in the paper. Working with code goes further than this sense of reflexivity associated with written forms. Cramer explains this as 'a recursive loop, in which literature writes its own instrumentation' (in Goriunova & Shulgin 2003: 54). He is interested in the ways that notation and the execution of a concept or of code are collapsed into one event. His key example (and what he considers to be a seminal software art work) is the Fluxus performance score of La Monte Young's _Composition 1961 No. I, January I_, a piece of paper with the instruction: 'draw a straight line and follow it'. Clearly code cannot be separated from an understanding of the overall structure of which it is part, that includes its writing and execution. This is even more the case with the self-replicating source code of a 'Quine' - a program whose output is exactly the same as its complete source code.[18]

In treating a work of art in terms of itself, the influence of the art historian Clement Greenberg is often cited and his position that each art must isolate and make explicit that which is unique to the nature of its medium (1992).[19] The problem is that this suggests a description of ‘pure form’ outside of social context, or one in which the work of art is autonomous. This criticism is often levelled at formalist experimentation with software, in which process and hence the social implications are underplayed, especially in generative art according to Arns (2004, and mentioned in the earlier part to this chapter). However, there is evidently an ambiguity here in the descriptions of formalism. Russian formalism (associated with the above references to Propp and Todorov) rejects the idea of pure form. A hard distinction between form and content is undermined. In software art too, there are plenty of examples of practices that reject pure form, and express far messier forms of critical engagement that signal a broader context outside of itself. For instance, messy (or dirty) code would suggest code that does not necessarily compile or be machine or human readable. Harwood’s _London.pl_ is an example of this, as a formal experiment that follows the syntax of Perl but is not intended to be, nor is, executable. There are many examples of artists working with ‘pseudo-code’ in this way, such as Mez, who writes in a hybrid ‘creole’ of English language and ‘pseudo-code’ that she calls ‘mezangelle.pseudo.codework’ (in Block et al 2004: 254).[20] Using a phrase from Mez, the title of Cramer’s essay ‘exe.cut[up]able statements’ exemplifies the creative potential of mixed language forms, using both a filename for sourcecode that is executable (.exe) and making reference to cut-up poetry such as in the work of William Burroughs (2003: 98). Texts that combine so-called natural and artificial languages can be seen to play with signification and undermine the semiotic categories of signifier and signified (a literal and metaphoric understanding of signification). In such examples, meaning and authorship remain in question (as does the dubious distinction between natural and artificial language).

It is entirely possible for software art to contain formal elements and at the same time social critique. However, in much practice an assumption is made that an engagement with formal elements and self-referentiality somehow equals political engagement. This is an issue

that Yvonne Volkart describes as similar to the debates around net-art in the late nineties, that:
'... as soon as software is used as a tool in a manner for which it was not intended, so that it at best generates and reveals its own regulatory mechanisms, it is interpreted as artistic, critical and political' (2004).

To Volkart, abstraction and what she calls 'code fetishism' are simply assumed to be radical and outside of commercial imperatives. For instance, JODI's _Jet Set Willy Variations_ (2002) is an example of the tendency to use very simple programs and assembly languages that seem to express the raw materiality of code - such as in their modification of the 1984 game _Jet Set Willy_ first designed for the ZX Spectrum computer. The significance is that low-level languages are seen to have a close proximity to the mechanics of the hardware and so evoke an engagement with the apparatus at a deeper level. But what is the significance of this engagement?

In examining open source culture and software arts practice, Josephine Berry Slater describes the practice of hiding source code as narrowing its creative potential, and enforcing a series of mythologies around creativity and property rights (2005). With a rejection of private property in mind, 0100101110101101.org's project _life_sharing_ rendered the data on a networked computer public property. The logic of open source was extended to the 'laying bare' of the apparatus associated with Russian Formalism [26] in what the artists refer to as 'data-nudism': 'It sets its kernel free and all the functions that concern it, in the same way as a programmer who frees the source code of their software.' (2001)

For Berry Slater, the approach of 0100101110101101.org confirms the engagement with code and the relations of production that are expressed in the shared production of free software. This allows her to question that if 'net artists use proprietary software to produce their work, to what extent can they be said to be transforming the apparatus of production?' She is asking what constitutes a radical work of software art in the context of previous claims for engaging with the technical apparatus. Berry Slater makes an explicit reference to Benjamin's essay 'The Author as Producer',

first presented as a lecture in April 1934 at the Institute for the Study of Fascism in Paris, to clarify the potential problem. In this essay, Benjamin claimed that:
'An author who has carefully thought about the conditions of production today [...] will never be concerned with the products alone, but always, at the same time, with the means of production. In other words, his [sic] products must possess an organising function besides and before their character as finished works.' (1992b: 98)

The significance of the engagement with the means of cultural production lies in requiring the producer to act as an active agent in the production process, to transform the apparatus and thereby effect a change in the relations of production. For Benjamin, it is not enough for cultural producers to express political commitment, however radical it may seem, 'without at the same time being able to think through in a really revolutionary way the question of their own work, its relationship to the means of production and its technique' (1992b: 91). The cultural producer must reflect upon his or her position within the production process like a technician, demonstrating expertise alongside solidarity, acknowledging whose interests or more particularly class interests the producing serves. This is an issue of property not least, in taking control over the means of production, and in the case of software production opening it up to potential transformation.[22]

This position will be developed in subsequent chapters but is one that relies on an engagement with software that is not separate from the materials and institutions that produce it. The approach goes some way to counter the criticisms levelled at software art that it provides the appearance of political engagement but without substance (privileging form over content, or product over process). Self-referentiality is not enough in itself - it must be combined with an improved apparatus if the materialist position of Benjamin is followed. Predating 0100101110101101.org's _life_sharing_ and with similar intentions, was the browser _Manifest_ (1999) by 46liverpoolst.org (now offline).[23] Using the browser rendered the user's hard drive public in the spirit of a rejection of private property (advocated by Karl Marx and Friedrich Engels in _The Communist Manifesto_, first printed at 46 Liverpool Street, London in 1848). In this way, the hardware were made open source in addition to software. The user's guide was simply a reprint of _The Communist Manifesto_ in the form of screen shots of the text in the browser. The suggestion of this section and the selected examples of projects, is that a materialist approach comes close to an understanding of 'speculative software' that reflexively investigates the conditions in which it operates. Through an engagement with the means of production, some of the antagonistic social relations involved in software production can be made more visible and open to change. This is inherent to a critical practice that recognises its material and historical foundations.

2.3 - software art history

The contemporary artistic preoccupation with software production clearly has a history.[28] In addition to a history of literature, it can be traced through an art historical lineage that would include Dada, Conceptualism, and other practices such as performance that have sought to challenge art's commodity form. Software art has an ambiguous relation to use-value in this respect,[25] and manages to challenge some of the precepts of what constitutes art. Combining software art and performance practices, The Museum of Ordure (UKMO) exploits this ambiguous relation to value by collecting ordure (rubbish, shit, waste), aiming to draw attention to what and how cultural systems assign value to objects. Operating in the spirit of artists dealing with the material of shit (such as Piero Manzoni, and Stuart Brisley who is also a trustee of UKMO), objects submitted to the museum archives in the form of data files are subject to a destructive process running on its server that corrupts these files. Statements on the web site suggest that the museum sees itself as a reflection of the destructive tendencies of capitalism itself. The museum implies that the best kind of ordure resists commodification.

Some of the examples of software art mentioned so far in this chapter represent a further development of radical practices in art prone to recuperation. This section will chart some of these histories, without intending to be a comprehensive historical or systemic account – it is far too selective and partial to make this claim. The key argument is that rather than software art representing a further art historical genre, it offers the potential to rupture this sense of continuum (the following chapter 3 will introduce historical materialism that underpins this claim). In terms of a comparison to previous practices, like 'the work of art of the Dadaists became an instrument of ballistics' according to Benjamin (1999d: 231), software art might similarly be deployed in explosive tactics. For instance, in JODI's website wwwwwwwww.jodi.org (1995) the source code behind an abstract arrangement of characters on screen reveals a diagram of a hydrogen bomb. JODI 'turn software inside out' according to Julian Stallabrass (2003: 38), revealing something of the hidden ideological nature of the system in clearly materialist terms.

Despite the grand claim to explode the continuum of art history, firstly a more straightforward historical approach is taken.[26] It is also worth noting that 'newness' is historically bound, and this is what Charlie Gere suggests with his paradoxical title 'When New Media was New' (in Kimbell 2004: 46-63) – derived from Carolyn Marvin's _When Old Technologies were New_ (1990). This principle underpins Gere's historical work on early computer arts (some examples of which were mentioned in the first section of this chapter) that weaves together information theory and artistic experimentation, and retains a belief in the radical potential of art.[27] To stress the importance of history in relation to emergent arts practices goes against the grain of the tendency for forward-looking theories associated with technology. The paradox is that much historical work tends towards futuristic theories, such as the 'visionary work' of Roy Ascott and other futurologists like Buckminster Fuller or the futurist Filippo Marinetti.[28] These figures are taken to be pioneers (as well they may be) or part of an 'avant-garde'. However the avant-garde has failed to deliver what it promised – both Dada and Conceptualism are perceived examples of art's inability to deliver social transformation. Is software art doomed to the same fate?

The suggestion of this section of the chapter is that art should deal with the central issue of transformation rather than representation reflecting the properties of software (whether used directly or not). On the surface this sounds like a very contemporary position, supported by the curator Nicolas Bourriaud's claim that the image is now defined by its 'generative power', and that art can be seen to be a program(me) for the generation of forms and situations (2002: 70). His term 'relational aesthetics' describes a practice that involves human interactions, social context and the new aesthetic and cultural concerns that arise from this. He is referring to artwork that is a programme to be followed, a model to be reproduced, or an encouragement to do something - and points to the parallel activities of artists engaging in ideas of interaction and sociability, set against the hype of interactive computer systems. To Bourriaud, artwork not using the computer has as much potential to make work about its effects. This may well be the case, but his statements appear to take their cue from systems thinking. They could easily be paraphrased from Gregory Bateson's 1971 position on art (from _Steps to an Ecology of Mind_) that focuses attention not on the message but the code (2000: 130). Working across multiple disciplines including anthropology, social science, linguistics and cybernetics, Bateson considers the production of art, and art as product, in terms of behaviours or rules that are embodied in the machinery that then generates transformations (2000: 271-2).

For art historian Edward Shanken, the emphasis on behaviour points to the 'paradoxical nature of knowledge and the contradictions inherent in formal epistemologies' (2003: 5), predicated on Bateson's critique of scientific method that combines strict and loose thinking (2000: 75). Bateson's cybernetic ideas were influential on Ascott's radical art pedagogy of the 1970s, in which art and the teaching situation were seen to be subject to feedback loops that produce creativity. To the artist and teacher Ascott, the production of art and learning were mutually supportive, becoming a 'force for change in society' (2003: 98).[29] Thus the potential for change in the system exists in the sense that: 'out of the flux, a many-sided organism may evolve' (2003: 102). This is a reference to _La Plissure du Texte_ (1983), one of Ascott's pioneering distributed authorship artworks that used early telecommunications

networks - or what he calls ‘telematic art’. The project extended his interest in cybernetics to the use of emerging pre-Internet network technologies that seemed to exemplify ‘connectivist’ thinking,[30] together with the post-structuralist reference to Barthes’s _The Pleasure of the Text_ (1975). Text is taken as tissue, behind which lies the realm of meaning. Ascott quotes Barthes to this effect:
‘the generative idea that the text is made, is worked out in a perpetual interweaving; lost in the tissue - this texture - the subject unmakes himself, like a spider dissolving in the constructive secretions of its web’ (2004: 198).

From this perspective, symbolic information systems containing numbers, text or code might be seen to be artistic material, to be rearranged accordingly. In the case of _La Plissure_, a distributed nonlinear narrative or improvised ‘planetary fairy tale’ was generated over the network, in the manner of weaving a textile with multiple authors. To some art historians, this is the beginning of ‘interactive art’ and of course ‘Internet art’.[31] If software art presents the possibility of software itself as art, then Ascott’s statement lays the historical ground in 1966: ‘The computer may be linked to an artwork, and the artwork may in some sense be a computer.’ (2003: 129) It is this line of thinking that underpins how software characterises arts practices that privilege the idea, code, process, system and its transformational qualities. Whether using a computer or not, art has become like software.

# software as cultural metaphor

Clearly there is a history to software art and a canon appears to have emerged. Andreas Broegger is one researcher amongst many who situates the contemporary term software art in the historical context of the _ Radical Software_ journal published by the Raindance collective (launched in 1970), and Jack Burnham’s exhibition _Software, Information Technology: Its Meaning for Art_ at the Jewish Museum, New York (also 1970). Broegger describes the ways the term software was used as a metaphor for arts practice at that time, to stand for the transmission of information using available communications technologies, in contrast to the ‘hardware’ of object-based art (2003a).[32]

Although any discussion of software requires an understanding of its relationship to hardware (even if it is accepted that software can exist without hardware), it is clear that the term software is being used in a rather different sense in the 1970s. In the field of art at least, the description runs in parallel to conceptualism and its associated shift away from the end-product at that time. A contemporary use of the term software reflects an emphasis on process, that has become the orthodoxy in contemporary cultural practices. In the light of this, Broegger insists on the need to inquire whether to accept software art as art or not. Again, this is the wrong question to ask (according to Benjamin); software is more than just art and expands an understanding of art's possibilities.

In Benjamin's artwork essay, the meaning of art changes with the character of its technical reproduction. As a result, he insists 'the total function of art is reversed. Instead of being based on ritual, it begins to be based on another practice - politics' (1999d: 218). It is this line of thinking that appears to influence the _Radical Software_ journal. A statement from the first issue gives some idea of its agenda to engage overtly with politics: 'Power is no longer measured in land, labour or capital, but by access to information and the means to disseminate it' (Ross 2003).[33] On a technical level (in tune with the democratic potential that Benjamin attributed to technical reproducibility), it was the widespread availability of the video portapak that inspired the belief that this could contribute to social transformation, through people gaining increased access to the means of production and becoming producers. In the context of its publication in the United States, the position of the journal was influenced by the rise of the civil rights movement, a general mistrust of the communications media on offer, requiring more independent and alternative media and cultural practices, combined with ecological concerns (according to Ross 2003). Those associated with this project 'imagined a world in which the contest of ideas and values could take place freely and openly' outside of the existing institutional and ideological frameworks of commercial telecommunications. They proposed 'a new information order in which the very idea of hierarchical power structure might be transformed or even eliminated' (Ross 2003). In this sense, what is radical about software is that it acts upon hardware. It operates as a metaphor for an emphasis on social processes that involve an engagement with relations of production and 'radical' transformation. In a sense, Benjamin's call for a politics of representation is upgraded to a politics of transformation.

In parallel to the _Radical Software_ journal, 'software as a metaphor for art' was explored in Burnham's _Software_ exhibition. The show can be seen as a product of its times with its overt structuralist and conceptualist concerns, and its aim to focus attention on the technical apparatus. It corresponds to what has since become commonplace in looking to the 'dematerialisation' associated with the conceptual arts tradition and the 'immaterialisation' of information and communications technology. In his essay 'The House that Jack Built' (1998), Shanken traces Burnham's concerns with particular reference to his book _Beyond Modern Sculpture: The Effects of Science and Technology on the Sculpture of this Century_ (1968a) that ends with an account of 'systems aesthetics'.[34] By an aesthetics of systems, Burnham refers to non-object based art and time-based based practices such as performance, interactive and conceptual art, but also public interaction that breaks down the false distinction between the operating systems of art and non-art. This is software metaphorically-speaking, the abstract 'internal logic' of a program receiving feedback from human subjects. For example, works in the exhibition included Hans Haacke's _Visitor's Profile_ where personal information was entered into a system, and Sonia Sheridan's _Interactive Paper Systems_ where visitors were encouraged to engage with the artist and a photocopy machine. Clearly neither of these artworks correspond to formalism but exemplify what Burnham refers to as examples of 'post-formalist art' (to include the influence of an understanding of subjectivity). The term 'post-formalist' (more commonly called 'post-structuralist') is used with reference to some of the influences on his thinking – such as Claude Lévi-Strauss's structural anthropology and Thomas Kuhn's critique of scientific objectivity, as well as Barthes's semiotic distinction between readerly and writerly texts.

In summary, the exhibition _Software_ was an attempt to reveal some of the contradictions between object and non-object, art and non-art, artist and non-

artist, evident in art's organisational and systemic logic. In this respect, particularly important to the development of Burnham's writing and the _Software_ show are the influences of information theory associated with Claude Shannon and Warren Weaver, systems theory associated with Ludwig von Bertalanffy, and cybernetic theory associated with Norbert Wiener. An interest in cybernetics and ideas of feedback had already been tested in Jasia Reichardt's exhibition _Cybernetic Serendipity_ at the ICA, London, in 1968, which is widely regarded as an historical marker for first combining art and cybernetic ideas. This approach arguably takes an even earlier cue from Ascott's interest in cybernetics and behaviour, encapsulated in the following quote from 'The Construction of Change', published in 1964:
'To discuss what one is doing rather than the artwork which results, to attempt to unravel the loops of creative activity, is, in many ways, a behavioural problem. The fusion of art, science and personality is involved. It leads to a consideration of our total relationship to a work of art, in which physical moves may lead to conceptual moves, in which Behaviour relates to Idea [...] "An organism is most efficient when it knows its own internal order".' (2003: 97)

# dematerialisation and conceptual traditions

The Ascott quote exemplifies an approach to arts practice that rejects the hardware of the physical object for process. Therefore, it is no accident of history that it also introduces Lucy Lippard's _Six Years: The Dematerialisation of the Art Object from 1966 to 1972_ (1997). Lippard's concept 'dematerialisation of the art object', first introduced in 1968 with John Chandler, characterises arts practice of this time in two ways – 'art as idea and art as action' (Lippard 1997: 43) – and of course not art as object. This approach departs from arts practices of preceding years and its perceived 'anti-intellectual, emotional intuitive processes of art-making', replacing it with 'an ultra-conceptual art that emphasizes the thinking process almost exclusively' (Lippard 1997: 42). Dematerialisation in this sense serves to de-emphasise not simply art as object but the related orthodoxies of originality, permanence, and beauty into an 'anti-form' or 'process art' or 'concept art'.[35] Its influences were derived more from Dada-ist ‘poem-objects’ or ‘found-objects’ such as Marcel Duchamp’s ‘ready-made’ to shock people into new understandings of the material world. For instance, Tristan Tzara had advised aspiring poets to cut a newspaper article into words and make a poem by shaking them out of a bag at random, revealing the hidden possibilities of language, and clearly undermining notions of creativity and genius by providing a way for anyone to work with words. With scissors and glue, words could be made to appear as arbitrary patterns, rhythmical noise, mere chance arrangements of words and sounds, reflecting the confusion and emptiness of the world and renouncing the language of the mass media. In this procedure, there is a consistent concern with everyday objects challenging the judgement of originality and authority, through the Duchampian ‘readymade’ and montage techniques. In a similar way, conceptual art established an attack on the conventions of the art-world and the commodity status of the work of art - what in a contemporary setting would be called ‘hacking the art operating system’ (Sollfrank 2001).

Conceptualism has been particularly influential in attempts to draw software art into an art historical register. Referring to Lippard’s portrayal of dematerialisation, software art is clearly both idea and action, and on a conceptual and technical level embodies a description of source code and its execution. The generative approach of conceptual artist LeWitt is evocative of software in this connection: ‘The idea becomes a machine that makes the art’. (in Lippard 1997: xiv)[36] The quote was used in the publicity materials for the _Generator_ show with this connection in mind. LeWitt provided explicit instructions for the production of artworks that are then executed by other people. [37] For instance, the program for _Wall Drawing #69_ (1971) reads: ‘Lines not long, not strait, not touching, drawn at random using four colors, uniformly dispersed with maximum density, covering the entire surface of the wall.’ (in Reas 2004: 277)

Casey Reas’s _{Software}Structures_ (2004) takes these instructions for wall drawings as the inspiration for software, writing source code that generates software drawings. Examples such as this demonstrate how it has become commonplace to position software art within a conceptual tradition as a continuation of the 'dematerialisation of art', treating code as artistic material in a similar way to taking ideas or concepts as material. Jacob Lillemose develops the parallel between programming and conceptualism by defining linguistic, political and performative modes. Firstly, he describes 'linguistic conceptualism' (associated with LeWitt, Art & Language and Joseph Kosuth), that considers art as a 'self-reflexive logistic system composed by writing and ideas, and a language in which form and content tended to merge', and the work of art as a set of instructions or composition (associated with La Monte Young and Cage in particular), 'as a purely mental, non-physical, phenomena' (2004: 139). In addition to this, there is a more cultural or political dimension (associated with Haacke, Victor Burgin, and Gordon Matta-Clark), and a more performative one (associated with Vito Acconci, Bruce Nauman and Chris Burden), according to Lillemose. Clearly it is possible to cast software art in these terms but the point for Lillemose is that both an internal logic and wider social issues are evoked. Paraphrasing Sarah Charlesworth's 'A Declaration of Dependence' of 1975 (that Lillemose adapts for the title of his essay 'A Re-declaration of Dependence'), he says: 'the contextual nature of conceptual art points towards an aesthetics based on the relationship between the internal structure of the work of art and external non-artistic structures' (2004: 140). Featured in the Runme.org software art repository, one example selected by Lillemose that makes context explicit is _Anti-Capitalist Operating System_ attributed to 'Together we can defeat Capitalism' (2003).[38] It is a platform for anti-capitalist activities that takes on the appearance of a conventional operating system, and a working prototype in this respect for full development. With 'browser art' and software art in general, the context for the work is an integral part of the work - the frame as part of the artwork. This conceptual attitude gained some legitimacy when the GNU/Linux operating system was awarded a prize at the Ars Electronica festival in 1999.

By historicising these software practices, Lillemose is trying to avoid describing emergent practices as 'avant-gardist' (2004: 145). Instead, he argues for a contextual understanding of software art and 'software not-just art' (like Burnham in _Software_). Adopting Peter Weibel's thesis that proposes three generations of artists interpreting the legacy of conceptual art, Lillemose proposes software art as a fourth generation. This is speculation perhaps (or 'speculative software art history'), as he does emphasise that art history is not an exact science but an interpretative form. However, it seems a mistake to simply place software art in an art historical continuum in this way. In the context of the argument of this thesis, software art holds the potential to break this continuum. This is an important point of emphasis.

# from representation to transformation

If art holds radical potential at all, the issue remains how to produce art that resists its seemingly inevitable commodification and how to reconcile the apparent failure of the avant-garde to deliver its promises (evident in Conceptualism and Dada). Do the historical tactics of radical art require complete overhaul or better implementation? Whereas Eric Kluitenberg, in 'Transfiguration of the Avant-Garde/The Negative Dialectics of the Net' (2002), sees the opportunity for avant-garde tactics to be deployed in the larger context of the network society, Duna Mavor is highly skeptical of such claims.[38] Mavor is bitingly cynical about the interventions of art-activist groups, regarding their strategies as tired repetitions of obsolete dialectical logic leading to inevitable recuperation. She is thinking of art-activist groups such as the anti-corporate corporation RTmark. Perhaps it is right to be suspicious of RTmark's work if it invites interest from the commercial art world but also their rejection and auction of their invitations to take part in the prestigious Whitney Biennial is a blatent attempt to resist this. It is worth quoting Mavor at length for her ideological verve (and despite her opposition to the dialectical approach of this thesis):
'Dialectics never died. It lives every time another tired exhibit of the relics of dada or situationism opens at the houses of culture across the world. It lives when the hackers who haunt the net repeat the slogans and gestures of the dead and then congratulate themselves when they are finally inducted into the halls of power of the Venice Biennale or Ars Electronica. It lives when the theorists and cartographers of new deterritorialized flows of desire sell their interests by entering a classroom to become functionaries of the empire of production, offering packaged knowledge to

students who eagerly produce whatever stupidity is asked of them in exchange for the general equivalent of a grade. It lives when the anti-globalization "multitude" faithfully ascend to the stage of negation to recite their memorized roles, proudly displaying the garments of an ideology that long ago betrayed its exhaustion. Dialectics consumes the desire of life as it beats its wings against the limits of the impossible.' (2002)

Mavor's position is in keeping with Peter Bürger's notion of the 'post avant-garde' (1984) with which he describes the failure of the historical avant-garde. This is also a trajectory reworked in Eric Hobsbawm's _Behind the Times_ (1998), on the failure of visual arts to deal with reproducibility and hence remain radical. But rather than give up on the tactics of the avant-garde altogether or indeed dialectics as a positive force, Jürgen Habermas's suggestion is to learn from past mistakes and from previous attempts at negating modernity. The radical potential of newness is associated with modernity and the avant-garde has encapsulated this revolutionary potential. This position is informed by a dialectical understanding of modernity, representing a transitional state between the old and the new – it remains an 'incomplete project', to use Habermas's phrase (1991 [1980]). He says:
'The avant-garde understands itself as invading unknown territory, exposing itself to the dangers of sudden, shocking encounters, conquering an as yet unoccupied future. [...] The new value placed on the transitory, the elusive and the ephemeral, the very celebration of dynamism, discloses a longing for an undefiled, immaculate and stable present.' (1991: 5)

In this way, Habermas accounts for the interruption of the continuum of history (drawing upon Benjamin's concept of history, which the following chapter explains in more detail) in articulating the present as a moment of revelation. Taking this interventionalist view of history, it is possible to 'make the continuum of history explode', in Benjamin's words (1999c: 253). He also refers to the inherent violence in Dada, such as the use of explosive and destructive tactics directly applied to the work of art as a metaphor for change in the social realm. In the 1960s, 'Auto-destructive' art associated with Gustav Metzger was similarly conceived as a way of transforming people's thoughts and feelings

about art and hence society, using an ‘aesthetics of revulsion’ including acid, ballistics, corrosion, radiation, and such like as artistic material (1996: 27).[39] Like Dada, the artistic tactic is one that appears irrational but is intended to be a rational response to the irrationality of society - particularly in response to the emerging ‘potential-probable destruction’ of nuclear, biological, and chemical weapons (1996: 28), just as Dada responded to the First World War. In Metzger’s view, artists should develop techniques in response to discoveries in science and technology (and clearly an understanding of entropy). He quotes László Moholy-Nagy’s prophetic _Vision in Motion_ of 1922 to stress the point:
‘Carrying further the unit of construction, a dynamic constructive system of force is attained whereby man [sic], heretofore merely receptive in his observation of works of art, experiences a heightening of his own faculties, and becomes himself an active partner to the forces unfolding themselves.’ (1996: 37-8)

Whilst recognising that growth was a further possibility in the form of what Metzger called ‘auto-creative art’ (a term introduced in 1960) (1996: 55), the ideological position of auto-destructive art remains rather different, with its social agenda to negate the destructive tendencies of the social world.[40]
A more contemporary example of destructive tendencies entering the art world is the _biennale.py_ virus that contaminated the Venice Biennale’s web site (produced by 0100101110101101.org with epidemiC, for the Slovenian pavilion of 2000).[41] The cultural form of a virus appears to embody the principles of auto-destructive art and negation. A virus describes the self-reproducing activities of a program that can simply spread and effect other programs, and thereby reflects the structural properties of the computer and the Internet it operates through. For Jaromil, the source code of a virus is potential lyrical poetry. Related to this, the elegance of his Unix shell _forkbomb_ (2002) encapsulates this aesthetic approach in presenting only thirteen characters to dramatic effect.[42] Once entered into the command line of a Unix shell and run, the program exhausts the system’s resources, causing the computer to crash. It was included in the exhibition _I Love You: Computer, Viren, Hacker, Kultur_ (held at the Museum für Angewandte Kunst, Frankfurt am Main, in 2002), referring

to the _I Love You_ virus (of 2000) that spread through the communities of the Internet. Opening the message 'love letter for you' would activate a program that would erase documents from your hard drive and then propagate itself by sending new copies of itself through the address book of your mail program.[43] The destructive potential of a virus operates in the spirit of auto-destruction and Dadaist tactics.

The comparison of software art to earlier avant-garde movements, and particularly the avant-garde activities of the 1920s in Russia and Germany, provides an historical understanding of radical forms and strategies. In the contemporary situation, it appears that many of the claims of the historical avant-garde have become:
'... embedded in the commands and interface metaphors of computer software. In short, the avant-garde vision became materialized in a computer. All the strategies developed to awaken audiences from a dream-existence of bourgeois society [...] now define the basic routine of a post-industrial society: the interaction with a computer.' (Manovich 1999)

The once radical technique of montage has become commonplace. On the surface, it seems that what was once a radical aesthetic vision to reveal the social structure behind the visible surfaces, has become a standardised form through the use of computer technology. Lev Manovich discusses these perceptions of change, and the ways in which ideology naturalises these changes. This reflects contemporary culture's reliance on appropriation, wherein recycling, re-working, and re-combining media are the standard techniques.[44] He concludes that 'the avant-garde becomes software' (1999) and that it continues to introduce revolutionary techniques but the terms are different:
'software does not simply adopt avant-garde techniques without changing them; on the contrary, these techniques are further developed, formalized in algorithms, codified in software, made more efficient and effective' (1999).

Whereas the 'old media avant garde' was concerned with vision and representation, the 'new media avant garde' is concerned with the manipulation, generation and transformation of data (a position in keeping with Bateson's, discussed earlier). Maurizio Bolognini's installations under the series name _Sealed Computers_ (1992-) exemplify this interest in generative and transformational processes. Computers are networked together, positioned across the floor of a space, but there is no way to access what processes are running, as the monitor ports are sealed with wax. Although society is saturated with images, in this work there appears to be no way to penetrate the inner world of the computer, in the way that radical practices have previously sought to make these processes transparent. If, as Manovich thinks, software has naturalised montage techniques, how can software be further developed as a radical project in revealing the ideological processes at work? In _Montage-Transformation-Allegory_, Wright argues that transformation operates in the spirit of montage, inducing new shock effects for the digital age (1998). This is a rather different position to Manovich, who emphasises a non-dialectical 'anti-montage' of digital compositing, in which elements are blended into a whole rather than brought into collision (2001: 143). For Manovich, it is the database that forms the 'new symbolic form of the computer age' (2001: 219) and his concept of the 'data-base movie' derives from this logic. In contrast, Wright draws directly upon Sergei Eisenstein's 'A Dialectical Approach to Film Form', written in 1929, in which reality is not described directly, but must be reconstructed to reveal the hidden structure that otherwise remains obscured by ideological preconceptions.

Rather than seeing digital imaging in terms of smooth and normalised transitions and imperatives, Wright argues for the possibility of a digital aesthetic that amplifies the dialectical method. Even a stockpile of fragments, such as a database, could be seen as a site of conflict in this way, and one where hidden structures should be revealed. Indeed, the 'tendencies of the montage method are not opposed by any unifying tendencies of the transformation but by its particular dynamics of dispersion' (Wright 1999). Wright's assertion is informed by Benjamin's concept of allegory, in which new understandings emerge through the bringing together of historical fragments.[45] Exploring some of these ideas, Wright's project _The Bank of Time_ (2001, produced under the name Futurenatural), is a screensaver that makes an allegorical comment on idleness and growth.[46] Wright notes how the germinating plant is a recurring metaphor in financial and investment advertising, as well as in Baroque imagery. The user's idle time is directly proportional to the rate of growth of the plant on their desktop   from seedling to fully grown plant through to its decay. In _The Bank of Time_, the more idle the user, the faster the plant grows, suggesting idleness at work as a creative force.

In dialectical allegories such as this, objects are brought together through montage to disrupt the continuity of historical and ideological conceptions. Through montage, the objects that constitute 'the material world could be rearranged out of their conventional, found or "natural" order so that the forces which shaped them would become visible, manifest and accessible to the senses' (Wright 1999). The principles of montage thus take on a wider political status in the context of a history of art, and even more so with respect to artwork that uses software - that in itself is transformative. Is it possible to apply the principle of software to history in a similar manner, by paying attention to its processes and events? This is the challenge for software art and for the argument of this thesis that aims to develop a dialectical approach. In _Das Passegen-Werk_, Benjamin applies the principle of montage directly to his writing: 'This work has to develop to the highest degree the art of citing without quotation marks. Its theory is intimately related to that of montage.' (1999a: 458)

Unfortunately the protocols of PhD submission legislate against taking such an approach with this thesis. The re-appropriation of existing materials presents Benjamin with the concept of applying materialist principles of montage to history, in order to understand its construction.[47] Crucial to this method is the retention of contradiction. The challenge for a critical practice in software art is to maintain contradiction in the process of transformation, for this is where politics is evident and where re-invention takes place. This assertion is carried over in the rest of this thesis. In terms of the legacy of previous radical arts practice, the lessons of art history verify the point that Lippard makes: that in a contemporary situation where conceptual strategies have become the orthodoxy of contemporary art and effectively recuperated, radical art can be found in social energies not yet recognised as art (1997: xxii). Perhaps software art and culture represents such an instance - for now at least.

 =====================
3. *emergent history*
=====================

‘There is an irony deep laid in the very relations of life. It is the duty of the historian as of the artist to bring it to the surface.’ (Trotsky [1938], in Mosley 1972: 11)

The previous chapter’s argument for contradiction in the production of software art is extended in this chapter, by introducing a dialectical approach to history itself. In this way, it develops an argument against ideas of universal history, in favour of a process of unfolding contradictions that are emergent and indeterminate. By adopting this approach to the conceptualisation of historical processes, an analogy is made to the emergent properties of software. It is also important to reinforce the argument that software art should not simply be placed within an art historicisation - for instance as a further example of previous work or as a new genre - but that it should be seen as an opportunity to rupture the continuum.

Section 3.1 begins by introducing some of the principles of historical materialism, that treat history as material which can be reconstructed like montage to reveal its inner workings, its constructedness and the inherent distortions in which technology plays no small part. It is human agency that is obscured in these processes and this is clearly one influence among many, in relation to ideas of emancipation and strategies for political action. This section describes dialectical thinking as a technique to reject causality or over-determinism, for an ongoing process of unfolding contradictions of development and feedback. What is important in this is the retention of contradiction, so as not to treat dialectics itself as a deterministic method, as it is often conceived. Thus, the inner potential and outside influences of an object are continually held in contradiction, between what is possible and what exists, the recognition of which reveals the possibility of further transformations.

In addition to this principle of an ‘incomplete synthesis’, section 3.2 provides more detail on the dialectical method to explain how complex arguments are developed. The concept of 'negation of negation' is crucial here, to understand how dialectics is not simply a method that proposes a reversal of one thing with another but a deeper engagement. Much of the skepticism over dialectics has failed to engage fully with this concept, in which negation can be seen to operate twice - once, and then again upon itself in a reflexive manner (that evokes 'speculative software', as described in chapter 2). Like political action, software remains written in advance of its action.

According to this position, things are decided before they are enacted in actuality, but there is a delay that forms a part of the dynamic of history. Section 3.3 develops an understanding of dialectics to reveal this relatively hidden relation between the past and the possibilities for future transformation. An openness of dialectical method is substantiated by the concept 'transformative praxis' (Bhaskar 1998) to reject 'endism' and to emphasise human agency. Historical processes are thereby articulated as dynamic and emergent phenomena, subject to feedback. The approach suggests that nothing is finished or resolved but in a continual state of change, appropriate to the emergent properties of software itself. The concept of software praxis arises from this, and is expounded in the final chapter of this thesis.

3.1 - historical materialism

Software, like history, reveals the possibilities for change in the present. Both can be seen to be dynamic and unsettled in this way. This approach draws particularly upon Benjamin's essay 'Thesis on the Philosophy of History' of 1940 ('Über den Begriff der Geschichte', sometimes translated into English as 'On the Concept of History', 1999c). The German compound word 'Geschichtsphilosophie' demonstrates how the two concepts come together as montage to suggest the reconstruction of historical material, in order to construct not a philosophy of history but a philosophy out of history (Buck-Morss 1995: 55). In contrast to traditional approaches to history, Benjamin rummages for truth in the 'garbage heap', in the 'ruins of commodity production' (Buck-Morss 1995: 217-8). He describes his method as: ‘literary montage. I needn’t _say_ anything. Merely show. I shall appropriate no ingenious formulations, purloin no valuables. But the rags, the refuse - these I will not describe but put on display.’ (Benjamin 1999a: 860)

The suggestion of this chapter is that this materialist approach might be applied to software in a similar manner, to reconstruct new understandings of software out of existing material and source code. This is common practice in general terms, but it is less common to think of software in terms of historical materialism, where what is present demonstrates its potential for further transformation, and where human agency is foregrounded. Historical materialism describes the application of Marxist thinking to historical development, to stress the importance of ideas or the active role of individuals in history. In ‘The Eighteenth Brumaire of Louis Bonaparte’ of 1851-2, Marx claimed that:
‘Men [sic] make their own history but they do not make it just as they please; they do not make it under circumstances chosen by themselves, but under circumstances directly encountered, given and transmitted by the past.’ (1980: 96)

In other words, there are social forces that intervene in the process of history, and the critical impulse of the historical materialist, according to Benjamin, is to brush received history against the grain, in order to ‘make the continuum of history explode’ (1999c: 253). For Benjamin, the _Angelus Novus_ image by Paul Klee (1910)[1] captures history’s capacity for progression and regression:
‘There is an image by Klee called _Angelus Novus_. On it an angel is depicted who looks as if he is about to distance himself from something that he is staring at. His eyes are wide-open, his mouth is agape, and his wings are spread. This is how the angel of history must look. He has turned his face towards the past. Where, in front of us, a chain of events appear, he sees one single catastrophe. This unrelentingly piles rubble on rubble and flings it at his feet. He would really like to stay, awaken the dead, and repair the smashed pieces. But a storm is blowing over from paradise, and it is tangled in his wings and is so strong that the angel can no longer close them. This storm forces him irresistibly

into the future to which his back is turned, while the pile of rubble in front of him grows skyward. This storm is what we call progress.' (Leslie [translating Benjamin from the German] 2000: 202)[2]

The angel wants to gather up the wreckage of terrible past events, wasted lives and worthless objects to make things better but cannot, because of the dominant forces at work. This is the catastrophe of the 'status quo'.[3] What is crucial to this approach is that any moment in time can be traced historically in order to reveal its constructedness, and hence reveal the possibility of change in the present. Benjamin calls this 'Jetztzeit' (the presence of the now): 'History is the object of a construction, whose site is not that of homogeneous and empty time, but one filled with now-time' (Leslie [translation of Benjamin] 2000: 198). The significance of the materialist presentation of history forces the present into a critical state: 'It is the present that polarizes the event into fore- and after-history' (1999a: 471) It is as if time stands still, and the past and the future converge not harmoniously, but explosively. The suggestion is that software might be similarly conceived as filled with now-time, held at a critical state where its past construction and future execution remain in dialectical tension (see final chapter for more detail).

History, and the history of technology, is full of the use of trickery to make it seem natural and beyond the scope of human intervention. Leslie cites a statement by Theodor W. Adorno (in a radio lecture of 1962) to insist that the angel of history is not only the angel of history but the angel of the machine. Benjamin argues that any conception of history changes with the times, as does its analysis in accordance with changes in the material mode of production:
'It is the particularity of _technological_ forms of production (as opposed to art forms) that their progress and their success are proportionate to the _ transparency_ of their social content.' (1999a: 465)

Although Benjamin is referring to glass architecture, the formulation can be applied to other technologies to emphasise the availability of either open or closed social content (like open source and proprietary models of software production). A Marxist view of history is informed by an understanding of the material factors in production and the relations of production. According to this position, capitalism is simply a temporary organisational form that is neither fixed nor desirable. It is therefore subject to change and influence by those involved in the material, productive forces, and social relations where class antagonism and consequently social transformation derives. Historical materialism (or what Marx himself called 'the materialist conception of history') indicates an understanding of the growth and development of human history founded on these principles.

# fake history

At the beginning of Benjamin's 'Thesis on the Philosophy of History', there is a short passage by means of introduction:
'The story is told of an automaton constructed in such a way that it could respond to each move in a game of chess with a countermove that ensured him victory. A puppet in Turkish attire, and with a hookah in his mouth, sat in front of a chessboard placed on a large table. A system of mirrors created the illusion of a table transparent from all sides. Actually a hunchback dwarf, who was an expert chess player, sat inside and guided the puppet's hand by means of strings. One can imagine a philosophical counterpart to this device. The puppet known as 'historical materialism' is always supposed to win. It can easily be a match for anyone if it ropes in the services of theology, which today, as the story goes, is small and ugly and must, as it is, keep out of sight.' ([translation by Leslie] 2000: 172)[4]

The critical method of historical materialism is introduced as the figure of the automaton, that relies on the services of a dwarf hidden from view. The dwarf further evokes the labour of the operator, or even consciousness according to Esther Leslie (2000: 173). The success of the automaton is contingent on the recognition that the dwarf has to gain control of the technology, rather like taking control of the means of production. The reference is to a chess-playing automaton, built by the Hungarian mathematician Wolfgang von Kempelen in 1769, that subsequently received widespread attention. There was much speculation as to whether the machine was driven by magic or by some other illusory device - a spectre or demon. By the time it was exhibited in London in 1783-4, a parallel interest sought to expose it as an illusion in another sense, as a trick based on a disbelief that a machine could demonstrate intelligence sufficient to play chess. Part of the theatre of the presentation was for Kempelen to show the audience the clockwork mechanism beneath the automaton, opening doors to compartments of the desk one by one and revealing what lay beneath the Turkish attire. On the one hand, Karl Gottlieb von Windisch described this as the 'automaton stripped naked' (as if making its source code open) and thereby its workings shown to be authentic.[5] On the other hand, Joseph Friedrich Freiherr zu Racknitz in a pamphlet of 1789 suggested that someone was hidden in the desk: 'the Automaton chess player is a man within a man; for whatever his outward form be composed of, he bears a living soul within' (in Wood 2002: 66). This is the dwarf that Benjamin refers to, first suggested by Henri Decremps, and embellished by Racknitz, who described the hiding place in detail and how the dwarf would operate the chess pieces by the use of magnets and a duplicate board hidden inside the machine. In engravings, the scenario is imagined in such a way that the dwarf looks like a puppet of the Turk rather than the other way around.

After Kempelen's death, Johan Nepomuk Maelsel bought the machine in 1818 and added some improvements including speech – the announcement of 'Echec' (check) by means of bellows. It is this version that Wiener refers to as a 'fraudulent machine' in his note on the accomplishment of artificial intelligence (2000: 165). This version is also referred to in Edgar Allan Poe's essay 'Maelzel's Chess Player' (1836), in which he makes direct comparison to Charles Babbage's calculating machine, in asking:
'What should we think [of a machine that operates] without the slightest intervention of the intellect of man? It will, perhaps, be said in reply, that a machine as we have described is altogether above comparison with the Chess Player of Maelzel. By no means – it is altogether beneath it – that is to say, provided we assume (what should never for one moment be assumed) that the Chess Player is a pure machine, and performs its operations without any immediate human agency.' (in Wood 2002: 72)

The link between technology and human agency is made explicit in such descriptions. The illusion is that human agency is not involved in these machine operations, and that they are somehow autonomous (as with artificial intelligence). In Benjamin's application of the chess-playing automaton, the illusion relates to received history that is also not autonomous and can be corrected by human intervention, assisted by the historical materialist approach. This concurs with Leslie's account of the story in that the dwarf has to gain control of the technology to reveal the otherwise hidden relations of production. It is the perceived autonomy of technology and history that are revealed to be fake. The theatrics show the lengths that proprietary interests and manufacture goes to in order to mask the operating system and source code of any mechanism. Pretending to reveal this through trick mirrors has become part of the illusion of the history of computing.

Much software production operates in this way too, giving the impression of access but one that often hides a more detailed understanding of the mechanism at a deeper level. For instance, the graphical user interface of operating systems reveals only a very partial view. Under the surface of the interface lie other complex technical procedures that are kept out of view. The task of the historical materialist is to reveal these inner workings in order to develop a counter strategy to received history. A historical materialist understanding of software would similarly be engaged in developing counteractions to received notions of historical and technological processes, so that these might be changed through collective action.

# dialectical and historical processes

The operations of history, like operating systems in general, are far more complex than surface impressions suggest. Certainly historical processes cannot be simply described in terms of progress from one point in time to another. If a more complex formulation is upheld that rejects teleological approaches, then the position of Georg W.F. Hegel is questionable: that history is no mere accident but happens 'necessarily': 'The history of the world is none other than the progress of the consciousness of freedom' (1953: 19). This belief in history as an unfolding of meaning towards freedom does not appear to account sufficiently for the complexities of human subjectivity or societies. This is certainly the view of Gianni Vattimo, who argues that the 'ideal of emancipation' needs upgrading to accommodate 'oscillation, plurality and, ultimately [...] the erosion of the very "principle of reality"' (1992: 7). However, he does not reject the idea of emancipation altogether but maintains that emancipatory potential lies in the 'relative chaos' of a 'society of communication' (see section 4.3 for more detail on this). Or even, bearing in mind the previous section, 'our images of the present do not identify agencies and processes of change' sufficiently (Levitas 1995: 265), so that any lingering possibilities of emancipation or social change remain hidden or consigned to fantasy.

Unashamedly utopian in spirit, Hegel's concept of history is predicated on the 'necessity' of progress, in as much as historical change and positive development can take place in the human condition and consciousness. In _The Phenomenology of Mind_ (1967[1807]), he points to the ways in which the mind itself appears to the observer; this is inextricably linked to history and progress towards a consciousness of freedom. In other words, to Hegel, history is the development of the mind. This presents a conceptual paradox that involves the complexity of consciousness, in that to study the mind is to study the way the mind appears to itself.[6] Knowledge of something is only what appears to be known to the mind, adding another level of consciousness, and so on, in a developmental and generative process of learning. Although the goal of such a method may be absolute knowledge, this clearly remains impossible in practical terms. What this dialectical description suggests is a process of critical reflection, where a thesis is posed only for an antithesis to be counter-posed that reveals inadequacies in the first concept, and the pursuit of new knowledge to make up for these inadequacies. Any synthesis can only be temporary in this way, and part of an ongoing critical process. With more reflection the synthesis will reveal itself to be a new thesis in some other respect, and so require the same dialectical treatment, and so on, in order to continue a chain of knowing something better, rather than towards a final resolution (as commonly attributed to Hegelian dialectics). The idea of synthesis as complete follows the influence of Christian doctrine (thus Marx, and other 'young Hegelians'

such as Ludwig Feuerabach, rejected Christianity and thereby the determinism of the approach). The principle of 'progressive unification' is grounded by Engels by emphasising matter and materiality in the concept 'dialectical materialism'. For Engels, without contradiction nothing would exist at all, let alone be able to develop or change.[7] Historical materialism stops short of applying the concept as universally as this. The cultural-historical dialectical methodology is explained by Benjamin:
'It is therefore of decisive importance that a new partition be applied to this initially excluded, negative component so that, by a displacement of the angle of vision (but not of the criteria!), a positive element emerges anew in it too - something different from that previously signified. And so on, ad infinitum, until the entire past is brought into the present in a historical apocatastasis'. (1999a: 459)[8]

For Marx too, the dialectical process of contradictory forces are accounted for in history itself. But this is where Vattimo identifies the problem of assuming a unilinear history and whether there is an ultimate 'reconciliation' that he identifies with both Marx and Hegel. This issue of closure is a complex and contentious one, with many competing interpretations (whether Hegelian, neo-Hegelian, or anti-neo-Hegelian). However in this account of the dialectical method, it should be emphasised that any harmonising dialectical synthesis must be rejected for an ongoing critical process that retains contradiction at all stages of the development.

# incomplete synthesis

Against the popular interpretation of Hegel's work (exemplified by Vattimo in the above comments), Slavoj Žižek stresses Hegel's 'tarrying of the negative' to describe the retention of contradiction rather than the perceived false harmonising at the point of synthesis - what is sometimes called 'higher-order synthesis' (1999a). The principle behind this is that the system becomes stale, unless it is continuously challenged. In other words, herein lies the necessity of contradiction, and the impossibility of achieving full synthesis on both practical and conceptual levels. In Hegel's terms, this is a move from 'in-itself' to 'for-itself' - from 'ground' to 'conditions' (where ground is the essence and the conditions are what brings this about). The two opposing factors must be combined without losing the antagonism between them, between the inner essence and the external circumstances that gives rise to that essence. Žižek stresses Hegel's position as a radical anti-evolutionary approach to dialectic synthesis in this way:

'The simultaneous reading of these two aspects undermines the usual idea of dialectical progress as a gradual realization of the object's inner potentials, as its spontaneous self-development. Hegel is here quite outspoken and explicit: the inner potentials of the self-development of an object and the pressure exerted on it by an external force are strictly correlative; they form the two parts of the same conjunction.' (1999a: 228)

The antagonism between internal and external factors means that the human subject exists within an 'absolute unrest of becoming' (Žižek 1999a: 239). This is an evocative emergent state that is thoroughly embedded in historical process, which raises the issue of whether human subjects create the external world from within, or as a result of external circumstances expressed by Marx (quoted earlier in this section) that they:

'... make their own history but they do not make it just as they please; they do not make it under circumstances chosen by themselves, but under circumstances directly encountered, given and transmitted by the past'. (1980: 96)[9]

Hegel, as Žižek points out, would reject the view that human subjects make their own history as far too deterministic. The statement does not take account of the ways in which inner essence can be transformed into external conditions and vice versa. Yet, seen from a different perspective, Hegel's work can be understood as deterministic. For example, Hegel's view of the State is that which holds society together as the culmination of human achievement. This is the Hegelian assertion of the 'end of history' - a history that culminates in the present, that Francis Fukuyama appropriated for his _The End of History and the Last Man_ (1992) to insist on the triumph of neo-liberalism over Marxist 'materialist economism', thus expressing a false totality.[10] To Marx, the State holds society together but is also subject to historical conditions, hence cannot be complete and so requires continual dialectical attention. Marx simply insists that human consciousness is seen as a ‘succession of changing stages and shifting moments’ (becoming) and sees a contradiction in Hegel’s insistence on the end of history (Lefebvre 1968: 28).

Marx both continues Hegel’s project, and at the same time breaks with him. His ‘Postface to the Second Edition’ of _Capital_ of 1867, makes this clear:
‘My dialectical method is, in its foundations, not only different from the Hegelian, but exactly the opposite to it. For Hegel, the process of thinking, which he even transforms into an independent subject, under the name of ‘the Idea’, is the creator of the real world, and the real world is only the external appearance of the idea. With me the reverse is true: the ideal is nothing but the material world reflected in the mind of man, and translated into forms of thought. [...] The mystification which the dialectic suffers in Hegel’s hands by no means prevents him from being the first to present its general forms of motion in a comprehensive and conscious manner. With him it is standing on its head. It must be inverted, in order to discover the rational kernel within the mystical shell.’ (1990: 102 & 103)[11]

Dialectics is thus reinterpreted in a non-idealist manner, to assert the contradictory and dynamic nature of the material world, outside our perception of it. The dialectical process is marked by stages and there is no limit to further development, following an open-ended process rather than a set of received truths or towards an ultimate truth. The point of this section is not to assert the importance of one thinker over the other or to make an undialectical claim for authentic meaning, but to use these differing positions to substantiate the claim that a dialectical approach to history is not deterministic. As with historical materialism, these dialectical principles allow for an understanding of software in historical terms: between what is possible and what actually exists.

3.2 - negation

As the previous section demonstrates, the dialectical movement is not from one extreme to its opposite extreme (from yes to no) and from there to ‘higher unity', but rather a 'radicalisation' of the first position (Žižek 1999b: 71). Žižek quotes Wendy Brown's _States of Injury_ to make this point more clearly of how dialectical arguments are posed. Brown describes a familiar scenario where an oppressed group imagine themselves in a future better world with the oppressor removed (1996: 36). She then describes how this logic is problematic, as it fails to recognise how one identity position is infiltrated and mediated by the other (as a result of the capitalist production process). Žižek likens this to the misunderstandings at the root of the Hegelian idea of 'negation of negation'. He explains: '... its matrix is not that of a loss and its recuperation, but simply that of a process of passage from state A to state B: the first immediate 'negation' of A negates the position of A while remaining within its symbolic confines, so it must be followed by another negation, which then negates the very symbolic space common to A and its immediate negation [...]. Here the gap that separates the negated system's "real" death from its "symbolic" death is crucial: the system has to die twice.' (1999b: 72)

As Žižek puts it, negation of negation presupposes no magical reversal (1999b: 77). Explained further but in different terms, the Hegelian terms 'abstract' and 'concrete' universality relate to this – something only becomes 'concrete' when it reintegrates with its primary state. This logic underpins the Hegelian principle that it is only through 'abstract negativity' that 'concrete universality' can be attained. One can easily see how the relationship between the universal and the particular is an entirely political struggle, and as such the basis of representational democracy and its distortions. Žižek expands on this idea of post-politics describing the traditional relation between the particular and the universal that underpins politics. In this way, the detail on how a single issue becomes representative of a wider struggle is foreclosed and kept in the realm of the particular (1999b: 204). So-called socially-engaged arts practice runs into similar problems by failing to see how it is often left at the level of the particular, and infiltrated and mediated by the wider culture and economy it is part of, and yet that it seeks to challenge. Recent cultural policies that address 'social inclusion' are a case in point, where the particularity of the focus fails to address the wider issues that lead to exclusion. Furthermore, the identification of the issue in itself paradoxically leads to a perpetuation of the same logic of exclusion. This relates to what Žižek calls the failure of identity politics, that casts multiculturalism as a 'disavowed, inverted, self-referential form of racism which empties itself of all positive content' (1999b: 216) wherein the 'Other' takes on an assumed superiority – or the 'privileged empty point of universality', in Hegelian terms. The multiculturalist vision of hybrid, fluid identifications or 'unity in difference' (endorsed by what Arif Dirlik calls the 'intelligentsia of global capitalism', 1997: 77), is really a tolerance of the Western capitalist world order in which all alleged differences are not differentiated but ultimately homogenised. In summary, it can be said that oppositional identities are essential parts of oppressive systems.

The role of negation, and its negation, in this context is important to understand some of the ways in which dominant ideas attempt to reproduce themselves, even when an oppositional stance is taken.[12] So where does this leave oppositional tactics? Clearly negation is only one part of a dialectical strategy and should be used with full knowledge of how the 'expropriators are expropriated' as Marx puts it (1990: 929). A politics of software requires similar attention to detail. One might envisage software that negates software, and then negates the wider operating system of which it is part. The negation of negation is crucial in this respect. For example, the production of free software first negates proprietary software, but still remains within the confines of private ownership, so an attempt must be made to further negate the whole principle of private ownership of the means of production, to avoid being appropriated. Expressed in Hegelian terms, free software needs to operate through 'abstract negativity' in order that 'concrete universality' can be achieved, based on inherent cooperative labour and its founding principles of common property. The difficulty of course is that these principles have been appropriated by capital, making the production of free software not simply an alternative to capitalism but an expression of new forms of labour tied to cultural production (see section 5.2 for more detail).

# problem of totality

In orthodox Marxist terms, the proletariat stands for human universality, partly because it is the most exploited class but more importantly because it is a 'living contradiction', revealing the inherent antagonism in the capitalist system (based on the theft of labour power). The logic of this relies on the belief that the exploited class can express free will or autonomy sufficiently. Hegel thought the process of history involved the human spirit becoming conscious of its alienated conditions. In _History and Class Consciousness_ (1976[1922]), Georg Lukács describes any positive change of consciousness as synchronous to the (false conception of) reality it seeks to change. 'False consciousness' is a description of the lag between the way things are and the way we know. However, it does not take sufficient account of the fact that when we know something it has already transformed into something else by the act of knowing it ('self-knowledge'). Lukács proposes that a 'true' recognition is the social whole, within which the proletariat can be seen to be positioned oppressively - what he calls 'the problem of totality' (1976: 151). [13] The proletariat alone holds the potential for emancipation because the bourgeois class cannot see the whole. In this way, Lukács's concept of totality evokes the concept of universality in Hegel (described in the previous section) at the expense of other contradictions. It is worth adding here that Lukács's work is criticised heavily for being 'essentialist', in as much as it centres everything on the Hegelian idea of totality at the expense of other contradictions (Eagleton 1997: 185).[14]

The social whole is treated somewhat differently by Louis Althusser, who challenges some of the central tenets of classical Marxism and the centrality of the economic base (that it determines the superstructure) by adding levels of feedback. To Althusser, writing in 1969, the superstructure (that contains culture) is both relatively autonomous and exerts a reciprocal action on the base (1997: 105). This is important as it stresses the politics of culture, and the effectiveness of what he calls the 'ideological State apparatuses' to describe the mechanism of ideology to make things appear natural. [15] Through this 'naturalising' of certain dominant ideas, what is presented to the human subject, and

internalised, is an imaginary representation of the real conditions of production and their place within these structures. However, these ideas are clearly formed as a result of:
'... material actions inserted into material practices governed by material rituals which are themselves defined by the material ideological apparatus from which derive the ideas of that subject' (1997: 127).

From this, Althusser asserts that there is no ideology outside subjectivity, and he includes himself and the reader in this scenario as both thoroughly 'in ideology'. To Althusser, we are 'always-already subjects' practising the rituals of ideological recognition: 'all ideology hails or interpellates concrete individuals as concrete subjects' (1997: 130). It interpellates or recruits subjects by hailing 'Hey, you there!' (1997: 131). Althusser further explores the Freudian connections in the construction of the human subject and in the willing acceptance of this condition, in that:
'... the individual is interpellated as a (free) subject in order that he [sic] shall submit freely to the commandments of the Subject, i.e. in order that he shall (freely) accept his subjection, i.e. in order that he shall make the gestures and actions of his subjection "all by himself". There are no subjects except by and for their subjection.' (1997: 136)

Drawing upon Althusser, but also the work of Hegel and Jacques Lacan, allows Žižek to formulate a critique of neo-liberalism and its ideological project, based on free market principles and the illusion of free choice. This is what Žižek calls the 'vulgar liberal notion' of freedom, as 'not a choice between a series of objects leaving my subject position intact, but the fundamental choice by means of which I "choose myself"' (1999b: 18). In other words, the human subject is interpellated and imagines him/herself to be a free agent. This is a paradoxical scenario where choice presents itself as no choice at all in effect, rather like the impoverished choices of 'yes' or 'no' in an interactive game (that remains 'interpassive' but without the user's realisation). The Internet operates in this way too: 'it hails you, it connects to you and gives you an IP number; it interpellates you into Imperial ideology' (Holmes 2003). The choices offered are spurious ones.

However, this is given a more positive interpretation by Žižek in his defence of Hegel, and the idea of totality. In the case of political action, the subject is called by history to act in the right way and make the right choice of action. In other words, this is not ideological manipulation but what is almost preordained - it exists outside the subject's knowledge of it but it exists all the same. Žižek quotes Bertrand Russell to illustrate the point: 'I did not know I loved you till I heard myself telling you so'. It was not that love existed without the subject's knowledge but that the subject loved all along (1999b: 54). There is a delay between an event in-itself and for-itself. The influence of Lacan is clear when he describes how this 'retro-active' response often works with speech, in the shift from a virtual language to actual language - it is speech-in-itself, or speech that pre-exists itself, 'speech before speech' (1999b: 54). Things are decided before they are enacted in actuality. The connection between the past and the present relates to human action. As with speech, software operates in the same way too, in as much as it is programmed in advance of its action (see section 6.2 for more detail).

In contrast to the Hegelian idea that the social whole or totality will eventually be conceived of as the truth, Adorno sees this as largely realised in negative terms and therefore the whole must be conceived of as false. In the preface to _Negative Dialectics_ (2000 [1966]), Adorno describes how the 'negation of negation' has been conventionally taken to achieve something positive by means of negation. In Adorno's view, theory and criticism had to be combined as negation responding to this apparent failure of political philosophy to realise its aims, or to put its claims into practice. [16] Positive criticism leads to nothing, as history appears to have demonstrated, and has become a self-serving commodity (art criticism is notoriously uncritical in this way and mostly serves the interests of the art market). The collective Adilkno accuse contemporary criticism of operating an 'ego trip of a better world that starts and ends with oneself' (Adilkno 1998: 57). This is not least a sobering thought for the production of academic work such as this thesis.

# dynamic history

In _The Dialectic of Enlightenment_ (1997 [1944]), Adorno and Max Horkheimer reject the view of historical process as the recognition of false consciousness, as it makes too many assumptions about consciousness and subjectivity. Skeptical of the central role of class conflict, they argued that critical theory had to take account of how domination was being expressed in cultural forms rather than simply economic ones, which had led to the proletariat being thoroughly integrated into the system and no longer able to recognise their historical calling.[17] The revisionary Marxism of the Frankfurt Institut in general, of which they form a part,[18] draws together Marx and Freud, replacing the centrality of class conflict with a dynamic approach to history. According to Martin Jay these influences operate in a dialectical fashion: 'For Marx, the past is pregnant with the future [...]. For Sigmund Freud, the future is pregnant with the past [...]' (1996: 86, quoting Philip Rieff). To rely on change taking place at the level of the means of production did not go far enough, according to Horkheimer, and change could only come about through a 'rupture in the continuum of history' (echoing Benjamin's phrase from 'Theses on the Philosophy of History'). The rupture is necessary because the past is perceived as being false.

To Benjamin, the representation of history operates like a series of dream images, that possess the potential to awaken consciousness that is otherwise fixed in a dream-state. These dream images, or indeed nightmares, operate like unprocessed montages, like source code, that reflect the conditions in which dreaming individuals find themselves. In this sense, political consciousness is 'slumbering in the "once upon a time" of classical historical narrative' (Tiedemann 1999: 933).[19] Rolf Tiedemann elaborates on this to explain that if history is dream-like:
'... past objects and events would not then be fixed data, an unchangeable given, because dialectical thinking "ransacks them, revolutionizes them, turns them upside down"; this is what must be accomplished by awakening from the dream [...]' (1999: 935).

Being awakened from this dream-like state refers to the idea of the human subject 'not-yet knowing' their role in history. According to Tiedemann, Benjamin is using the work of Ernst Bloch as a model (for instance, in his _Spirit of Utopia_) by applying the 'theory of not-yet conscious knowing'; the ultimate test for dialectics in penetrating former contexts to realise present actions (Tiedemann 1999: 936). This conception of not-yet conscious knowing was introduced earlier in the chapter by Žižek, who explains 'class consciousness' not as awakening as such but as the change from 'class-in-itself' to 'class-for-itself' (1999a: 231). The knowledge required for any sense of freedom is self-referential. In contrast to a view of consciousness as too complicated or too determined to allow humans to act as free agents, }i~ek asks: 'Is the status of consciousness basically that of freedom in a system of radical determinism?' (1999b: 60). The human subject is clearly not an autonomous agent that simply processes information through the senses like a computer. For Žižek (combining Marx and Lacan), the unconscious remains a useful framework for thinking, as it provides productive cracks that give rise to contradiction. Žižek is not seeking truth in this respect, but change.

The unconscious haunts consciousness in a similar way to how historical ideas can be seen to haunt current thinking. Trying to exorcise his ghosts and reconcile his 'post-Marxist' position, Jacques Derrida's _Spectres of Marx_ demonstrates how Marxism continues to haunt contemporary theorising (and politics).[20] Derrida likens the spirit of Marxism to Hamlet's ghost and the sense in which 'being' raises the question 'to be or not to be' (1994: 10). That Marx, on his own admission was not a Marxist (as he allegedly confided to Engels), is a further example of the ways in which ideas are historically bound and open to problems of translation: 'How is one to receive, how is one to understand a speech, how is one to inherit it when it does not let itself be translated from itself into itself?', Derrida asks (1994: 34). And how could Marx be a Marxist? It is clear to Derrida that Marx could not both be a follower of his own thought without becoming a ghost of his former self. Dialectical thinking encourages such reflexive moves, and Marx stresses the need for a continual reassessment of ideas (Engels too, in the 1888 preface, explicitly talks of the 'aging' process of the text). Derrida is impressed by this self-critique: 'What other thinker has ever issued a similar warning in such an explicit fashion? Who has ever called for the transformation to come of their own theses? [...] so as to take account there, another account, the effects of rupture and restructuration? And so as to incorporate in advance, beyond any possible programming, the unpredictability of new knowledge, new techniques, and new political givens.' (1994: 13)

Past ideas continue to have effects, and like the return of the repressed, re-emerge at unexpected moments. Against the relativism of postmodern theory with its tendency towards openness to anything, contradiction provides productive cracks. To Marshall Berman, the lack of recognition of contradiction in contemporary thinking accounts for the stagnation he observes, in which: 'Open visions of modern life have been supplanted by closed ones, Both/And by Either/Or.' (1999: 24) Open forms and conjunctions are key to this.[21] Crucial to Berman's position, is the retention of history as inherently unstable, restless, contradictory, dynamic and dialectical force. In titling his book _All That is Solid Melts into Air_, he is specifically referring back to _The Communist Manifesto_ of 1848, to argue that what appears solid is fundamentally subject to change and influence:

'All fixed, fast frozen relations, with their train of ancient and venerable prejudices and opinions, are swept away, all new-formed ones become antiquated before they can ossify. All that is solid melts into air, all that is holy is profaned, and man [sic] is at last compelled to face with sober senses, his real condition of life, and his relations with his kind.' (Marx & Engels 1985: 83)

The quote reads as a truism now. Difficulty lies in the fact that constant change and almost instantaneous obsolescence have become mainstays of contemporary culture. If all new forms ossify, then how can alternatives emerge without being condemned to the same fate? The process of transformation itself clearly contains contradictory tendencies. As much as providing positive potential, crisis appears to serve the vested interests of capital, even strengthening it. Catastrophes are turned into lucrative opportunities for redevelopment and as an integrating force for the renewal of capital (one only has to look at current economic activities around 'cultural regeneration' for verification of this, or the re-development of countries after war by the same parties that destroyed them). This sense of paradox is evident in Marx too, in that crisis is both the motor for the renewal of capital and the means of its demise. As a consequence of these worries, there exists a tendency to reject the strategies associated with dialectics, such as negation, as no longer able to respond to current conditions. In Marxist dialectics, this is where the concept of negation of negation is crucial. Berman for instance, does not want a way out of these contradictions but a deeper way into them.

This section has attempted to give detail to the dialectical method, to suggest how complex arguments can be posed to respond to complex situations, and to understand the potential of the human subject in history to exert influence on the dynamic relation between what exists and future possibilities. The parallels of historical processes and contemporary ways of understanding transformation are introduced here, to add weight to the suggestion that reinvention is possible, and that reinvention is an intrinsic part of dialectics.

3.3 - emergence

Historical development expresses predictable and unpredictable tendencies. Its unpredictability makes history full of emergent potential, and not simply a series of repetitive cycles or a straight line of progress, as some descriptions would have it. In very general terms, emergence describes a process by which complex patterns are formed from simple rules. This can be a dynamic process occurring over time, but to be considered emergent it should generally be unpredictable.

Emergent phenomena can be explained through systems theory, and more particularly through an understanding of adaptive behaviour. For instance, society is an emergent system that is self-organised, able to demonstrate adaptive behaviour. Steven Johnson in _Emergence_ (2001), presents an example of the software program _Tracker_ that uses genetic algorithms to simulate the behaviour of an ant colony of sixteen thousand ants. The software does not simply follow instructions but responds to the principle that simple instructions might lead to complex behaviour. In effect, an antagonistic balance emerges from feedback (Solé &

Goodwin 2000: 100). The system appears to demonstrate a dialectical relation between feedback and control. A common conclusion of this type of work is that despite our tendency to look for a controlling mechanism: '... we are starting to think using the conceptual tools of bottom-up systems. Like dialectical logic of the nineteenth century, the emergent worldview belongs to this moment in time, shaping our thought habits and colouring our perception of the world' (Johnson 2001: 66).

The world-view Johnson has in mind is one based on an understanding of complex adaptive systems, and clearly this has an impact upon an understanding of history. Although conceptual thinking can be situated historically, a rejection of dialectics, based on its correspondence to the industrial revolution, does not recognise the way that conceptual models are also subject to change. Johnson's rejection of dialectics also comes without reference to its emergent properties. The final section of this chapter attempts to address the concept of emergence in relation to dialectics. Wider issues around the social implications of systems thinking will be introduced but also returned to in the following chapter, as will a more detailed discussion on the relation between complexity theory and dialectics. What follows will try to establish the connection between emergence and dialectics in as much as it has a bearing on an understanding of historical processes in general.

# transformative praxis

A more complex understanding of emergence that takes into account dialectics is something that 'critical realism', associated with Roy Bhaskar, attempts to achieve. The objectives are ambitious (to say the least) in developing a general theory of dialectics that extends beyond Hegelian thinking, to form a critique of Western philosophy. In general, critical realism suggests that the realms of physics and history share a false perspective on natural science, in as much as one tends towards a causal explanation (positivism) and the other an interpretive understanding (hermeneutics). Instead, Bhaskar proposes a 'critical and non-reductionist, naturalism, based upon a transcendental realist account of science and, as such, necessarily respecting (indeed grounded in) the specificity and emergent properties of the social realm' (1998: xiv). This position is informed by a number of historical sources including Karl Popper, who argued that it was falsification not verification that laid the foundations of scientific method;[22] historians and sociologists of science such as Kuhn, who emphasised the social processes involved in scientific endeavour; and Ludwig Wittgenstein, who emphasised the mutable character of facts in science. In this way, scientific discovery can be seen to follow a dynamic logic that is revealed progressively, underpinned by generative mechanisms or laws. As a result (and pertinent to the preceding section of the chapter), Bhaskar considers society as both the condition and outcome of human agency, and that human agency both reproduces and transforms society. He explains: 'Social structure, then, is both the ever-present condition and the continually reproduced outcome of intentional human agency.' (1998: xvi) In a dialectical conjunction, human agents are described as actively able to transform society and yet simultaneously constrained by society.

Bhaskar's revisionist position (particularly in his _Dialectic: Pulse of Freedom_ of 1993) is based on the view that Hegelian thinking is closed rather than open-ended, and that Marx never fully described scientific realism. His description of the dialectic in Marx as scientific, explains the contradictions in society in terms of the contradictory relations generating them as historical (rooted in the changes of the circumstances described), critical (demonstrating historical conditions) and systematic (tracing the historical conditions back to the mode of production) (1998: xxi). As for Hegel, Bhaskar explains the 'rational kernel' as a process of better knowing or learning through the dialectical process of greater critical depth: it generates what is already implicit but not explicitly articulated and by repairing some inadequacy as part of the ongoing critical process. The interpretation is close to }i~ek's defence of Hegel's 'tarrying of the negative' but with a very different set of references. Where for Žižek it is the unconscious, for Bhaskar it is the concept of absence that is crucial to an understanding of negation. He is supplementing what Hegel calls the 'grasping of opposites in their unity or of the positive in the negative' (in Bhaskar 1998: 580). The negative does not simply cancel the positive but reveals the logic that:
'a genus always contains, explicitly or proleptically,

its own differentiae; [...] negation always leads to a new richer determination – this is transformative negation – so imparting to categories and forms of life an immanent dynamic and to their conflict an immanent resolution rather than a mutual nullification' (Bhaskar 1998: 580).

The immanent dynamic is what Bhaskar refers to as 'transformative praxis', where contradiction remains active, open and ongoing. Without this open-ended approach, endism and other 'fundamentalisms' will dominate, in what Bhaskar calls 'irrealist dialectics'. Whereas dialectics, in terms of critical realism, permits the gradual elimination of absences characterised in terms of emergence.

# dialectical emergence

Emergence, for Bhaskar, is the generation of new possibilities, a 'quantum leap: matter as creative or autopoietic' (1998: 564). It is contradiction that allows connections to operate both separately and as part of a whole or totality. Echoing earlier descriptions of historical materialism, the 'here and now' is characterised by the influence of the outside and the past, in such a way that social phenomena can be seen to contain emergent properties. Emergence in this way describes the creative, autopoietic operation wherein new properties are generated out of pre-existing material forms (as with montage). The quantum leap is one that goes beyond Hegel and Marx. Bhaskar interprets the dialectical materialist position as teleological (too causal) in its characterisation of progress towards a better society.[23] In contrast, he describes society obliquely as an 'open-systemic entropic totality, in which results [...] are neither autogenetically produced nor even constellationally closed, but the provisional outcome of a heterogeneous multiplicity of changing mechanisms, agencies and circumstances' (1998: 600).

An understanding of adaptive systems informs this view, and undermines the teleological understanding of history. A teleological position is what Ilya Prigogine and Isabelle Stengers call 'one time-directed evolution', that does not take sufficient account of the complexity of systems. They ask the question: 'What is the specific structure of dynamic systems that permits [others] to “distinguish” between past and future? What is the minimum complexity involved?’ (1985: 16) Prigogine and Stengers are extending Hegelian thinking that rests on the ‘qualitative difference between the simple behavior described by mechanics and the behavior of more complex entities such as living beings’ (1985: 89). They assert that at each iteration of a level in the dialectical hierarchy, there is a corresponding increase in complexity in nature.[24] The relationship to history is important too, where the oscillation between input-output is a consecutive relation of past-future (as is the case with computation of course). According to Wiener in _Cybernetics_ (2000 [1948]) in the chapter ‘Newtonian and Bergsonian Time’, the modern conception of automata conforms not to a Newtonian model but to a Bergsonian one, in keeping with the description of living organisms. The reference to Henri Bergson emphasises the inadequacy of a Newtonian description of biology: ‘the difference between the reversible time of physics, in which nothing new happens, and the irreversible time of evolution and biology, in which there is always something new’ (Wiener 2000: 38).

Duration has been much misunderstood according to Bergson, as a linear movement of an object through space, such that the non-linear complexity is avoided in favour of a convenient solution that best suits easy comprehension. Thus movement, to Bergson - referring to Zeno’s paradox that if an arrow [time’s arrow] has to pass through an infinity of points, how will it ever reach its target) - is not simply the passage between points but ‘a qualitative becoming that affects both the arrow, the archer and the overall context’ (Terranova 2004: 50). In dynamical systems, time and space work together to produce motion. Describing this added complexity, emergence is a useful concept, as it suggests non-causal, non-teleological formations reflecting a historical materialist approach. Emergence also allows Bhaskar to conceptualise human agency in terms of incompleteness or absence, that propels the dialectic as an ongoing transformative process. It is Bhaskar’s concept of ‘transformative agency’ that characterises his work as ‘extra-Hegelian dialectics’ (1998: 638). He claims ‘for emancipation to be possible, knowable emergent laws must operate’ (quoting _Scientific Realism and Human Emancipation_, in Goodwin 1997: 121).

The concept of emergence, in as much as it informs systems theory and complexity, will be developed in the next chapter. In terms of what has already been covered, emergence extends an understanding of historical materialism to deal with contemporary understandings of transformation. An ‘extra-historical materialism’ is implied in the way that transformative agency stresses the importance of ideas and the active role of people in historical development - recognising that people do not simply make their own history nor are determined by history, but both. As far as it develops the overall argument of a thesis about software, this chapter establishes an understanding of transformative praxis to introduce what will be later referred to as ‘software praxis’ (chapter 6) to stress the future possibilities of emergent action.

=======================
4. *complex technology*
=======================

‘Technology is neither good nor bad, nor is it neutral’ (Kranzberg, in Castells 1996: 65).

Building upon the dialectical understanding of history set out in the previous chapter, the historical development of informational technologies can be seen to express both lines of discontinuity and continuity from previous modes of production. Concepts are stored in memory and then processed. This chapter examines dynamic forms associated with computer technologies and, by drawing upon complexity theory, suggests a currency for dialectical thinking.

Section 4.1 provides a description of technological development through its interaction with both history and society, thus rejecting any sense of technological determinism (confirming that technology does not determine society but is embodied by it). With reference to the economist Ernest Mandel’s ‘periodising hypothesis’, wherein each period of technological innovation builds upon the previous one rather than making a distinct break, it argues that development is non-teleological, combining emergent and residual forms. The contemporary phase of development, characterised by a network model, is no different in this respect. Despite the appearance of a lack of hierarchy, control is exerted through distributed rather than centralised forms. Detail on the nature of the processes running remains relatively hidden (like source code or DNA), expressed in ever more complex and ‘immaterial’ formations that obscure their historical and material conditions.

An understanding of computational processes and networks reveals more detail on the dynamics of control and feedback within these systems. Section 4.2 presents a brief and partial history of these dynamic processes, paying particular attention to the concept of feedback, and the social and ethical dimension of cybernetics and systems theory. It seems clear that many of the concepts relating to self-regulating systems have been used to justify free market logic, at the expense of the use of systems theory for social critique. Computer processes and their interpretation operate as both a metaphor and description in this way, leading to a discussion of ‘digital dialectics’ that traces the binary underpinnings of machine code, in parallel to speculation on the potential of dialectical thinking for computer criticism.

The complexity of systems such as were briefly described in the previous chapter, far from legislating against a dialectical approach, can be seen to demonstrate what has been referred to as ‘orderly disorder’, wherein deep structures of order exist within seemingly unpredictable and disorderly phenomena. Building on an understanding of emergence introduced in the previous chapter, section 4.3 elaborates on the way in which emergent order has social and political significance, in as much as these systems are open to change from both external and internal factors. What is developed in the chapter is an approach that combines systems theory, and in turn complexity theory, with dialectical materialism. This is referred to as ‘systems dialectics’ and a more detailed description of this is presented with particular reference to an emergent view of dialectics, described in the previous chapter. In the combination of systems theory with dialectical materialism, the importance of negation is supplemented by disorder that can give rise to a new sense of order, and thereby gives rise to further disorder and so on (what might be referred to as ‘the tarrying of disorder’ in ‘extra-Hegelian’ terms). An argument for transformative praxis associated with software art arises from this conceptual approach in the final chapter.

4.1 - technological development

Many commentators have tried to characterise the so-called 'information technology revolution' in similar terms to the impact of the industrial revolution. Manuel Castells is sympathetic to this, describing it as 'a pattern of discontinuity in the material basis of economy, society and culture' (1996: 30), but is keen to emphasise that _The Rise of the Network Society_ needs to be understood through the dynamic intersections of technology, society and historical change. It is this basic point that this first section will introduce, in keeping with a dialectical understanding of historical processes that this chapter aims to extend to technological processes.

In general terms, capitalism has evolved into its contemporary form by embracing technological change, and done so whilst making sure it continues to protect its own interests. It can thereby be regarded as discontinuous from the industrial mode but its overall logic remains continuous. In dialectical style, Castells says:
'The rise of the network society [...] cannot be understood without the interaction between these two relatively autonomous trends: development of new information technologies, and the old society's attempt to retool itself by using the power of technology to serve the technology of power.' (1996: 52)[1]

Castells characterises these simultaneous conditions in his phrase 'informational capitalism' (1996: 18). The phrase avoids the danger of imposing too severe a model of change associated with terms like 'post-industrialism', or what Daniel Bell contentiously called 'postindustrial society' to assert that new social formations no longer obey the laws of industrial production (Jameson 1991: 3). The lines of continuity are stressed by using the term 'late-capitalism' (as Jameson does) to reject too harsh a distinction and emphasise that this is more like a modification of capitalism than a new form as such. Related to this (though more extreme) is Mandel's proposition 'senile capitalism', that suggests the irrationality of its continued logic (alluding to 'senile-dementia', 1990). The various choice of terms points to some of the ideological issues that arise from the descriptions of the force of change.

The discontinuity to which Castells points is the change in the ways in which technological processes are organised: from a mode of development focussed on economic growth and surplus-value (industrialism) to one based on the pursuit of knowledge and increased levels of complexity of information. Industrial production has been supplemented by information, and capital regenerated in a new form that reflects technological innovation. ‘Informationalism’ as Castells calls it, is the result of the restructuring of capitalism, and the new crucial material resource of informationalism is knowledge. In terms of the continuity of capitalist logic, Castells states: ‘Indeed, we observe the parallel unleashing of formidable productive forces of the information revolution, and the consolidation of black holes of human misery in the global economy.’ (1996: 2)

That wealth and power is ever more concentrated in a small number of giant industrial and financial corporations, lends weight to the Marxist position of Mandel, who observes that contemporary society reflects the model presented in _Capital_ (1990) in a ‘purer form’ than when it was first composed in 1867. He is writing this in 1976, and points to various examples of late-capitalism’s crises at that time (for example, from the student and workers’ riots in Paris of 1968, to ecological concerns as a result of nuclear power). He claims that contradictions continue to manifest themselves in all aspects of capitalism’s workings, however latent they may appear, such that impetuous growth is combined with its negation. Both Mandel’s position and the one adopted in this chapter are historical in scope, in order to situate the specific mode of production in the context of previous modes. The lines of continuity are easily overlooked in descriptions that rush to dramatise technological change and forget the lines of continuity. As much as technology might itself contribute to these changes, it does not determine them. Contradictions are inherent to the historical development of technology.

# third stage of development

Technological development can be characterised in terms of generation and feedback. Castells refers to this as ‘a cumulative feedback loop between innovation and the uses of innovation’. He charts ‘The Historical Sequence of the Information Technology Revolution’ (1996: 32 & 40-46) beginning with the invention of the telephone in 1876, and the invention of the first programmable computer after the Second World War. According to Castells, development only enters a significantly revolutionary phase by the 1970s with the interrelated fields of microelectronics, computers and telecommunications, and adds that biotechnology contributes a further phase that might well constitute a revolution in itself (but this remains unproven in his view, as with nanotechnology in a more current context no doubt).

The historical sequence used by Castells is an overt reference to Mandel’s ‘periodising hypothesis’ of expanding and stagnating economic cycles (in _The Long Waves of Capitalist Development_, 1978), a model further adapted by Frederic Jameson to chart trends in cultural production (in _Postmodernism, or, The Cultural Logic of Late Capitalism_, 1984).[2] The importance of the principle is that each period is seen to build upon the previous rather than making a distinct break, in emergent and residual forms. Clearly influenced by a dialectical materialist understanding of history, Mandel thus accounts for the evolution of technology:
‘The fundamental revolutions in power technology - the technology of the production of motive machines by machines - thus appears as the determinant moment in revolutions of technology as a whole. Machine production of steam-driven motors since 1848; machine production of electric and combustion motors since the 90s of the 19th century; machine production of electronic and nuclear-powered apparatuses since the 40s of the 20th century - these are the three general revolutions in technology engendered by the capitalist mode of production since the “original” industrial revolution of the later 18th century.’ (in Jameson 1991: 35)

The so-called ‘third phase’ has been supplemented by networked and interactive computer systems. In parallel, social and organisational structures and interactions have changed to a network model, made possible by telecommunications technologies enabling the interactive

‘real-time’ broadband networks of the present Internet and mobile cellular technologies – what is often referred to as ‘ubiquitous computing’ (computing that is omnipresent and everywhere).[3] At the core of technological revolutions, changes are ‘characterised by pervasiveness, that is by their penetration of all domains of human activity’ (quoting Kranzberg & Pursell 1967, in Castells 1996: 31). Biotechnology seems to encapsulate this pervasiveness in the recognition that organic and technical processes contain self-organising functions that genetic algorithms aim to replicate. The machine is seen to work like an organism, and as a result seem life-like. As Donna Haraway puts it in ‘The Cyborg Manifesto’:
‘But basically machines were not self-moving, self-designing, autonomous. They could not achieve man’s dream, only mock it. They were not man, an author of himself, but only a caricature of that masculinist reproductive dream. To think they were otherwise was to be paranoid. Now we are not so sure [...]. Our machines are disturbingly lively, and we ourselves frighteningly inert.’ (1991: 194)

Quite literally, the ‘dead labour’ of the machine has become reanimated by its comparison to biological systems (see chapter 5 for more detail). Such a view is typified by the position of Kevin Kelly, who enthuses about the technological future as one informed, if not determined, by biology (sometimes called the ‘post-biological era’ to indicate the force of change). The software program _Tracker_ mentioned in the previous chapter, is an example of simulating the complex behaviour of an ant colony. The study of ant colonies is a popular example to explain the lack of a discernable organisational hierarchy.[4] As opposed to mechanical systems or the factory assembly line, distributed systems have no obvious chain of command and represent what Kelly calls a ‘swarm model’ (2003: 39). However, such an enthusiastic reception avoids the ideological implications, making the lack of control somehow appear as ‘natural’ as an ant colony or swarm of bees, masking new forms of control that operate a distributed approach.[5] Control within distributed systems can be partly explained by the mathematical expression called a ‘power law’, that describes how few events manifest most of the action, reflecting the existence of large numbers of nodes but few hubs.

Albert-László Barabási uses this understanding of power to understand the 'laws behind complex networks' (2002: 73). He quotes the work of the Italian economist, Vilfredo Pareto, who observed the 'universal law' that 80 per cent of peas (or property) were produced (or are owned) by 20 per cent of pea pods (2002: 66). Although this '80/20 rule' seems rather generalised, it does offer some insight, not to universal laws of nature but to the politics of networked systems.

The revelation that Barabási's thinking was influenced by an economist strikes a historical parallel to the detail that Charles Darwin's theory of evolution was influenced by the economic theory of Thomas Malthus, who researched population growth and the efficient management of scarce resources. The ideological dimension of Darwinism (or what Sherry Turkle calls 'unnatural selection' 1997: 149) emphasises how scientific claims are developed in parallel to the cultural narrative of the time. As a further example, Sarah Kember describes how the neo-Darwinist position of Richard Dawkins's _The Selfish Gene_ (1976) can be read against the discourse around subjectivity during the 1970s, under the influence of post-structuralist thinking.[6] Hence the cultural narrative can be seen to be 'about displaced agency, about a subjectivity that has the illusion of control while the real locus of control lies with another agent who inhabits the subject and uses him for its own ends' (quoting Hayles, in Kember 2003: 18). Following this narrative, the human is correspondingly characterised as an autonomous 'selfish' machine. It is precisely to avoid genetic determinism that Dawkins proposes 'memes' to characterise the cultural aspect of evolution (a hybrid term combining the Greek word for imitate 'mimeme' and gene). These memes, according to Dawkins, are ideas that pass from human to human but still subject to the laws of natural selection. There may be some degree of human agency but only in a selfish sense (like the gene) in response to the scarcity of resources and survival. Kember sees a paradox and weakness here:
'Free will, it would seem, simultaneously counters and legitimises determinism. Metaphors of genetic and memetic agency and the ideological loop-hole which Dawkins constructs within them permeate the creation of artificial life worlds which are, to this extent, biologically determined.' (2003: 39)

In contrast to Dawkins's deterministic view of human agency, the neurobiologist Steven Rose argues for an active human subject that is capable of acting not only in, but upon the world. Paraphrasing Marx's statement from the 'The Eighteenth Brumaire of Louis Bonaparte' (1980), Rose claims 'we have the ability to construct our own futures, albeit in circumstances not of our own choosing' (in Kember 2003: 22). He wishes to characterise human agency in terms of 'auto-poiesis' (from 'poiesis' meaning creation), adapted from the work of Humberto Maturana and Francisco Varela (1973) who state:
'An autopoietic machine is a machine organized (defined as a unity) as a network of processes of production (transformation and destruction) of components which: (i) through their interactions and transformations continuously regenerate and realize the network of processes (relations) that produced them; and (ii) constitute it (the machine) as a concrete unity in space in which they (the components) exist by specifying the topological domain of its realization as such a network. (1980: 78)

Human agency in terms of auto-poiesis offers an argument against what Evelyn Fox Keller calls the 'end-game' of molecular biology's quest to discover the ultimate code or secrets of life (in Kember 2003: 25). As with Bhaskar's concept 'transformative agency' (in the previous chapter), human agency is not merely encoded or determined but is auto-poietic. Therefore, although descriptions of complex behaviour within distributed networks tend towards biological determinism, human agents express emergent behaviour that is transformative.

# networked production logic

With a historical perspective, networked informational production continues to make a calculation not in human terms, but by forcing together humans and machines to extend the productivity of labour at all costs.[7] In the industrial period, Marx refers to the worker who performs repetitive tasks coming to embody 'the living mechanism of manufacture' (1990: 458). This is even more the case with informational production, wherein constant networked interaction is required between workers, management and machines (see chapter 5 for more on 'machinic' relations). Systems must be networked and integrated in order to process information efficiently, arranged in decentralised and centralised working relations. This is all part of the shift from single entities of computer processing units, to systems served by computers that form complex and flexible networks. The network logic now deploys informational technologies and reflects the complexity of interactions and the unpredictable patterns of development arising from these interactions. For Kelly, rather than entrenching (or globalising) exploitation as Castells describes, this network logic 'revolutionises' social practices under the conditions of what he calls 'network economics' (2003: 236). Under these conditions, companies take on the character of software (2003: 244). Kelly's example is Microsoft's new operating system (at that time), pointing to the ways in which companies invest heavily in developing the manufacturing process rather than the product. Perhaps a better example (and indeed business model) would be the development of open source software, where many in the community of users are also involved in developing the software for free. However, in both network and previous economic frameworks, efficient development is contingent on bugs and errors in the system, making the analogy to software of further use in demonstrating how distributed production is combined with its negation.

These are the conditions of the networked 'factory without walls', as a self-perpetuating system which exhibits distributed, decentralised, and adaptive behaviours, evoking the working organism that Marx described in the industrial period, in which inherent human sociability is forced to turn against itself. These are the technical foundations of the industrial revolution, that the information revolution develops further. Quoting Andrew Ure's 'Philosophy of Manufactures' (of 1835), Marx describes the automatic factory as a fully integrated system:
'a vast automaton, composed of various mechanical and intellectual organs, acting in an uninterrupted concert for the production of a common object, all of them being subordinate to a self-regulating moving force' (1990: 544).[8]

Acknowledging these earlier descriptions, Castells claims that automation comes of age with information technology, and labour is transformed by the need for the required knowledge to operate it, offering new relational patterns in the performing of tasks (1996: 244). He characterises this new labour force as ‘networkers’ and ‘flextimers’, in terms of the increased individualisation of work and the fragmentation of society in general (1996: 201). New employment patterns correspond to the modes of development outlined at the beginning of this section – broadly, in relation to interlocking historical stages that reflect the agricultural, industrial (what could be described as post-agricultural) and informational (or post-industrial) forms. The necessity of the continued supply of cheap labour is an essential part of globalisation, as it has been since the sixteenth century when capitalism first sought to expand its operations on a global scale. What distinguishes the global economy now is that: ‘it is an economy with the capacity to work as a unit in real time on a planetary scale’ (Castells 1996: 92). Networked technologies serve this purpose.

Consistent with earlier phases of technological development, the networked computer combines contradictory impulses in encouraging highly socialised forms of labour. It would be plainly ridiculous to deny the possibility of technology, and of collective effort, increasing the social productivity of labour and contributing to social transformation. However, in the global economy, contradictions unfold:

‘Ironically, it is the very people whose labour is so carefully hidden inside the hygienic white boxes on the desks on the wired world [...] who will be left outside in the world their work creates. In this way, the production of the material infrastructure for the Internet is itself erased under the sign of the universality of its language, its claim to speak for all and with every voice [...] representation, in both the democratic and the semiotic senses, is in question in cybernetic technologies of communication’ (Cubitt 1999: 6).[9]

To a large extent, in the ‘over-developed world’, the assembly lines have been replaced by the network as the organisational model and metaphor for production in general. Castells thinks that increasingly:
‘... the multimedia world will be populated by two essentially distinct populations: the interacting and the interacted, meaning those who are able to select their multidirectional circuits of communication and those who are provided with a restricted number of prepackaged choices. And who is what will be largely determined by class, race, gender and country’ (1996: 371).

He formulates class distinctions on this basis too: between the networkers, who set up connections on their initiative; the networked, who are online but without any control over decisions; and another category of the switched-off who are tied to tasks and operate through non-interactive, one-way instructions. This indicates greater social stratification rather than the reverse: class divisions based on the difference between the information rich and poor.

# network control

There has been a tendency to think of networks as equitable systems, viewed as fundamentally random simply because they are too complex to comprehend how power is distributed in them (Barabási 2002: 24). In social systems, nodes gather together in clusters. Nodes with a large number of links or connections are present within diverse systems - from society to the cell (or cellular systems that includes genes, proteins and other molecules). This further accounts for the inappropriateness of a term such as randomness, as well as the perceived lack of democracy within systems. Despite the appearance of disorder in networks, clearly there is an underlying order - all nodes are not equal by any means (or some nodes are more equal than others). In _Empire_, Michael Hardt and Antonio Negri claim ‘the new paradigm is both system and hierarchy’, that demonstrates the structural logic of ‘governance without government’ (2000: 13, 14).[10] That any node connects to an exponential number of others is deceptively simple, and the historical development of the Internet reveals more detail on this. In 1964, searching for a robust communications infrastructure that could withstand attack, Paul Baran discovered that both centralised and decentralised models were too vulnerable, and so a distributed ‘mesh’ architecture was proposed, such that if any one node was attacked the distributed architecture of the network would compensate. This distributed structure is the basis of the Internet.

There is clearly more critical work to be done in this area of network theory, that requires an understanding of how the network works on a technical level, without losing sight of politics. For instance, and in the context of networked computing, Alex Galloway describes the Unix operating system and in turn TCP/IP (Transmission Control Protocol/Internet Protocol). He adopts the concept of the 'protocol' as more than just a metaphor – although clearly it is a compelling one all the same in suggesting correct or proper behaviour or social practices – to describe how computers in a network agree technical standards of action, by following the protocols that 'govern' their usage at the level of code (2004: 7). Protocols thus operate ostensibly as a distributed management system coding packets of information, documents and communication.[11]

The crucial issue remains about how power is articulated in a distributed model and and how power might be redistributed. Although the Internet is largely nonhierarchical in structure, conforming to the way TCP/IP connects one machine to others, it is also subject to the DNS (domain name system) information stored in decentralised databases but organised in hierarchical, inverted tree-structures. For example, in the current model the Internet's address structure (DNS), which enables communication between the world's computers, is managed by the California-based, not-for-profit Internet Corporation for Assigned Names and Numbers (ICANN) under contract to the US department of commerce (Wray 2005). In question is the centralisation of control and whether other countries should be allowed more control over their Internet domains.

This is not to say that control is bad of course, and certainly protocols have no vested interest in themselves. The issue lies in the fact that standards are set according to certain ruling interests – and is therefore a political issue. Certainly control might be exerted to undermine these interests. For instance, peer to peer networks are one obvious example of principles based on a different set of social practices, that simply optimise the existing open structure of the Internet. Technical detail here also reveals some of the cracks in the system. The Internet may be relatively robust as a result of its distributed topology, but it also displays elements of vulnerability if attention is paid to technical detail. For instance, 'cascading failures' are well known explanations of a situation where a local failure redistributes responsibilities to linked nodes, cascading through the system sometimes to disastrous effect, depending on how central the role of the node is within the system as a whole. This can be demonstrated with a wide variety of systems: failed routers within the Internet, or species within an eco-system, or in economics with the collapse of a certain company (Barabási 2002: 120).

Technical detail such as this is clearly of use to those wanting the system to become more robust, or those wishing to bring about its destruction. The more advanced the capitalist economy becomes, Marx argued, the greater the contradictions and the more it sows the seeds of its own destruction in that it creates its grave-diggers: the working class. Perhaps the seeds of destruction, that Marx thought to be internal to the production of the commodity, are now evident in code. Closer attention to computational processes might reveal some of these internal contradictions and the ways in which capitalism has sought to overcome these contradictions (what Castells calls 'the recapitalisation of capitalism', 1996: 85). Capitalism evidently contains the seeds of its own destruction but knows it - aware of its immanent crisis, in other words. An understanding of the interactions between history and the development of technology seems to confirm the point.

4.2 - dynamic processes

In considering the development of informational technologies and the ways in which they are embedded in society, a long history of mechanisms that relate to 'computation' is invoked - mechanisms that perform logical or physical processes of generation. A more detailed understanding of these mechanisms requires the linking of computational machines to the development of logical thinking (that would include Blaise Pascal's adding machine of 1642, Gottfried Wilhelm von Leibniz's multiplication machine of 1671, Babbage's analytical machine of 1835, as well as von Kempelen's chess-playing automaton mentioned in the previous chapter). For instance, a logic derived from the 'clockwork' mechanism has been particularly influential in setting mechanical production and thinking to the imperial standard of Greenwich Mean Time.
[12] Where and when invention arises is distinctly unreliable, as a result of the ways in which ideas emerge rather than occur at discrete times in history.

In a history of computing, Babbage’s ‘Analytical Engine’ is generally considered the first device that might be considered to be a computer in the modern sense of the word. However, it is Ada Lovelace who describes the engine as able to ‘compose elaborate and scientific pieces of music of any degree of complexity or extent’ (in Hofstadter 2000: 25). It was Lovelace who first envisaged an engine as not merely calculating numbers but arranging and combining letters and other symbolic systems - not as a calculator but as a logic machine. Adapting mechanical ideas of data storage and processing from Babbage, the von Neumann machine of the mid 1940s first presented a single structure to hold both the set of instructions on how to perform the computation and the data required or generated by the computation. This is the first modern computer in most accounts.[13] For Bolter, the combining of the program instructions and the data into the same code also represents the ultimate ‘assembly line’, and becomes an archetype for industry that requires specialised collective labour and knowledge (1984: 34). The resulting system can be described as self-regulating, in parallel to Marx’s description of the nineteenth-century factory as a ‘self-regulating system in embryonic form’ (1990: 503).[14] The description also lends itself to an understanding of the development of ideas through the simultaneous storing and processing of information.

As the previous section discussed, the computer is not merely a part of these cybernetic processes but also a metaphor for them, and as such it expresses ideological issues. In relation to cybernetic systems, Nichols refers to ‘the negative dominant tendency towards control and positive latent potential towards collectivity’ (1988: 23). Making explicit reference to Benjamin’s ‘Artwork’ essay, in which the technical apparatus was an essential part of the critical process and what he calls ‘the equipment-free aspect of reality’, Nichols argues that this equipment-free aspect is even more pronounced with cybernetic systems, deeply embedded in code and operating systems. It is no longer merely about suspension of disbelief but of our absorption into code, as our interest is diverted from products to processes. This section of the chapter introduces some key concepts related to computational processes such as feedback and recursion, to stress transformative possibilities of systems, and in order that in the last section these can be further described in dialectical terms. The histories of computing mentioned above demonstrate some of the dialectical principles of development and feedback expressed in systems, as a series of actions similar to historical processes that execute past results for future operations.

# systems theory and social critique

Feedback is either positive or negative, and most systems require both (positive feedback increases the change that brought it about, negative feedback reduces it). The classic example of the principle of feedback is the thermostat, switching on or off depending on the temperature of a particular space at a point in time. If all works well, the temperature remains constant. The governor of a steam engine is a classic example of a mechanical version of the same principle, regulating velocity depending on the load the machine bears and keeping its operations constant. In such a scenario, a 'compensator' (something that can be controlled from the outside because the load fluctuates) is required as well as an 'effector' (the input-output relations), in order to compensate for the faulty information feedback and to reinstate control (Wiener 2000: 113). This is a rather oversimplified description but the overall principles clearly hold relevance to both living organisms and artificial mechanisms, as exemplified by the full title of Wiener's book _Cybernetics: or Control and Communication in the Animal and the Machine_, of 1948. However, the human brain and the computer are clearly not reducible to one another, despite some operational similarities. In very general terms, both process information to execute actions by combining 'structure [hardware] with the instructions given it at the beginning of a chain of operations [software] and with all the additional information stored and gained from outside in the course of this chain [feedback]' (Wiener 2000: 146). Systems with long chains of instructions (such as the human brain) are particularly prone to errors, malfunctions and disorders, even breakdown. Complexity is one factor, but also some systems are just inefficient.

Free competition is a particularly inefficient system, according to Wiener. He calls it a simple-minded theory, in which the individual capitalist is regarded as a public servant who deserves the profits gained from his actions, as opposed to a selfish individual who steals surplus profit and disrupts the social equilibrium (2000: 158). According to Wiener, in the pursuit of profit certain tendencies emerge that lead to:

'... the elimination of the less profitable means for the more profitable; the fact that these means are in the hands of the very limited class of wealthy men, and thus naturally express the opinions of that class; and the further fact that, as one of the chief avenues to political and personal power, they attract above all those ambitious for such power. That system which more than all others should contribute to social homeostasis is thrown directly into the hands of those most concerned in the game of power and money, which we have already seen to be one of the chief anti-homeostatic elements in the community. [Tragically] the State is stupider than most of its components.' (2000: 161–2)

This moral dimension is an interesting aspect of Wiener's work on cybernetics. In the context of the free market, he relates this to a theory of games and a situation where there are winners and losers (no doubt, drawing upon John von Neumann and Oskar Morgenstern's general theory of games, in their _Theory of Games and Economic Behaviour_, 1944). Wiener provides numerous examples, including the idea of the automatic factory, in which a workforce of mechanical slaves performing human labour is imagined.[15] He remains open as to whether this is a good thing or a bad thing, but thinks any assessment cannot simply be formed in terms of the market; it must also include an understanding of the conditions of labour. To Wiener, any level of 'competition' between machine slave labour and human labour is a certain acceptance of the conditions of slave labour, even if on the surface it appears to decrease human suffrage. This is why the direction a society takes cannot be left to the market. Otherwise, development is determined by 'business cycles of boom and failure, in the successions of dictatorship and revolution, in the wars which everyone loses, which are so real a feature of modern times' (Wiener 2000: 159).

To Wiener, the solution lies in a better understanding of behaviour within social systems. In this way, his understanding of technical systems informed his critique of the social function of science and technology. For instance, the term homeostasis is taken from physiology and then applied to social and political systems. Small groups are efficient (where relative homeostasis can be discerned) and self-organise into relatively equitable conditions. In contrast, larger communities protect their interests by exerting inequitable property rights and individualism. Implicit to the idea of homeostasis (self-regulation) is that the results of any changes in the system's organisation are available as input to the system (such as recursive feedback).

# recursive transitions

The very structure of the system is reactive to its own actions, which includes the interaction of the scientist with the subject of study (Wiener 2000: 191). A reflexive approach in arts practice is commonplace but also has some currency in scientific practices, in recognition that to study something is distorted by the act of studying it. Additionally, any data on offer for study is not the equivalent of actual objects but representations of these objects, and therefore a certain transformation or recoding takes place between the object and the scientist. Furthermore, the selection of data is determined by what is possible to observe (or indeed, the vested interests of funding, and other outside influences). The data is therefore never 'raw' or pure, but subject to transformations by the human subject and the instruments used. It is from this position of skepticism over the verification of findings that the scientific process should proceed. What Bateson calls a 'critical faculty' is required on the part of the scientist to balance this 'mass of quasi-theoretical speculation' (2000: xxviii). In Bateson's work, this reflexive approach is referred to as 'recursive epistemology' (adopting Peter Harries-Jones's phrase) to incorporate processes of knowing that include the relationship between the knower and the known (2000: xiii). Clearly other non-verifiable and non-scientific work must also be taken account of in any analysis undertaken. It has become quite common to recognise the ways in which subjective elements impinge upon objective method (for instance, in contemporary anthropology that considers the subjectivity of the observer).

Knowledge, like the efficiency of cybernetic machines, is constructed for its transformative qualities in later usage (Latour 1999: 59). Thus, according to Bruno

Latour, science should be understood as a practice that is produced both in a social context and as a result of the technical and institutional apparatus. It is a 'disorderly mixture' revealed 'in action' rather than an 'orderly pattern of scientific method and rationality' (1999: 15). Scientists have tended to work on the reductionist assumption that by taking something apart we will gain an understanding of how it works from its constituent parts. But as with Humpty-Dumpty, from the knowledge of the pieces it does not necessarily follow that we understand how to put the pieces together or indeed how these parts operate together as a system (what Christopher Scholz refers to as the 'humpty-dumpty effect' of not being in a position to fully join two broken pieces back together again, in Gleick 1998: 106). The parts remain incomplete, according to the principles of fractal geometry.[16] Although at a gross scale, the broken crockery of Ono's _Mend Peace for the World_ (mentioned in chapter 2) appears to fit together, at a smaller scale it remains incomplete (or something to be continually strived for perhaps).

Facts appear to be held in dialectical contrast to speculations. For Bateson, advances in scientific thought come from a 'combination of loose and strict thinking', in which looseness is measured against 'rigid concreteness' (2000: 75) – evoking the Hegelian principle that 'concrete universality' can only be attained through 'abstract negativity'. Applied to the understanding of human societies based upon the analogy between society and organism as complex systems, Bateson sees social change as a slippage of the system, in which there is the possibility that a variable may reach a point of crisis. This reflects the dialectical method of contestation, although admittedly Bateson's concerns are more ecological.

Relays of 'on' and 'off' work as a series of actions, as part of a set of iterative processes that include 'memory' (the ability to use past results for future operations). For instance, Babbage's 'Difference Engine' rested on the logic of the 'method of differences' and employed the principle of the 'strange loop', capable of altering its own stored program (able 'to eat its own tail' in Babbage's words, cited in Hofstadter 2000: 25). Strange loops suggest rules by which new rules will emerge, rules that change themselves in self-organising structures. What was once a clean, linear and hierarchical structure has become a strange loop, or ‘tangled hierarchy’. Douglas Hofstadter further illustrates the strange loop by referring to M.C. Escher’s lithograph _Drawing Hands_ (1948) where ‘that which draws, and that which is drawn – turn back on each other’ (2000: 689). In these examples, some aspect acts upon the system as if it were operating outside the system. It is both simultaneously outside and inside the system, acting and being acted upon, both subject and object. As with the principle of feedback, it endlessly acts upon itself like a computer program running an infinite loop. [17] In terms of software art, an example mentioned in chapter 2 is McLean’s _forkbomb.pl_ (2001): a program that exhausts the system resources on which it runs and causes the computer to crash.

# digital dialectical logic

Instructions for computing operate in a binary mode of 0 and 1 – combining contingencies by using algorithms that follow this logic. One of the simplest expressions of this logic is ‘Boolean’, based on the dichotomy between ‘yes’ and ‘no’ (and in turn true or false, and so on [18]). Data follows both an arithmetical and logical binary form, as a set of choices between two conditions – for instance, in the case of switching between ‘on’ or ‘off’ in a series of relays. However, computation can also be extended to include more complex and conditional formations such as ‘or’, ‘and’, ‘not’, as well as rules about contradiction, consistency and implication (Bolter 1984: 69).

Further complexity relates to what has become known as ‘artificial intelligence’ (what Marvin Minsky described as ‘the science of making machines do things that would require intelligence if done by men [sic]’, in Bolter 1984: 193).[19] The term ‘artificial life’ also is relevant here (as a development of artificial intelligence), derived from von Neumann’s experiments in constructing self-replicating automata. A computer program can model these behaviours using self-replicating algorithms such as the example of Ray’s _Tierra_ software, in which synthetic organisms have been created based on a computer metaphor of organic life and evolution. The idea of software evolving or ‘learning’ becomes the way out of the paradox that an automated process, working on a repetitive cycle of answering ‘yes’ and ‘no’, would never arrive at an equilibrium. Through feedback, a computing machine might display 'conditioned reflexes' as a 'nervous computing machine' – more than simply a machine in action, which combines relays and storage mechanisms. This is clearly a description of a learning machine that might arrive at a solution but only (as with dialectics) as a result of an 'iterative process of successive approximations' (Wiener 2000: 130) – if indeed a final resolution is desirable at all (as with dialectics). This is the approach that Richard Levins and Richard Lewontin adopt in their dialectical approach to biology (1985). They focus on the relationship between the whole and its parts, and develop a less deterministic view of evolution following an open-ended dialectical process.

The parallel between digital and dialectical processes is what Peter Lunenfeld attempts in his introduction to _The Digital Dialectic_ (2000). However, he is cautious of too easy a conflation between the two. He claims that the on-off switching of cybernetic calculation does not create a synthesis, and merely reflects the contradictory condition of thesis and antithesis. This is a very generalised position and does not account for a deep understanding of negation or of incomplete synthesis. Lunenfeld is not dismissing the dialectical method altogether but drawing attention to its limitations. He sees its strength in the central dialectic of theory and practice, and in its application to detail in pursuit of the general (the difference between the particular and the universal, for Hegel). It is therefore ironic that Lunenfeld appears to overlook detail on the dialectical method itself. Also in _The Digital Dialectic_, Michael Heim distances himself from a dialectical materialist interpretation, instead concentrating on what he calls the 'joke or paradox that propels all dialectical thinking' (in Lunenfeld 2000: 26). He argues that an ongoing exchange between competing positions is a useful analytical strategy, and rejects deeper antagonisms related to power.[20]

There are plenty of detractors of the dialectical method, who are more systematic. Some commentators simply see the method as too crude to account for the ways in which communications are organised and dispersed in complex systems.[21] A challenge to Boolean logic comes from quantum computing (Bone & Castro 1997). Rather than considering a 'bit' in one of two states (0 or 1), a 'qubit' (quantum bit) can exist in three states (0 or 1 or both). Thus a layer of complexity is added to the description of a computer, suggesting possible directions of development that are less deterministic and more transformative. New ways of codifying and processing data might be envisaged as a result. The difference lies in that standard computing follows Newtonian physics and only allows a bit to be any one determined state at a time - on or off. Whereas a quantum computer 'exploits the possibility of quantum states of atomic particles to store digital registers in a definable but still undetermined quantum superposition of two states at the same time' (Floridi 1999: 189). Might this suggest a 'quantum dialectics'?

In _Philosophy and Computing_, Luciano Floridi suggests that Hegelian dialectics might help explain this principle of quantum superposition, whereby contradictory positions are reconciled in a higher unity by both being annulled and preserved in temporary synthesis (1999: 190). Although Floridi describes dialectics in 'absolute' terms (the example provided is the synthesis of the finite and infinite making the absolute, 1999: 190), synthesis might still be thought of as incomplete (it is explained that special logic gates would have to be developed to control the interactions between qubits and to generate coherent change). The potential is vast but largely speculative. In other words, the achievement of the absolute is only a potential state, and as such synthesis remains incomplete.

The suggestion in the context of this thesis is that digital-dialectical processes operate reflexively: as both a technical description of a system and a suitable critical method for its analysis. A well-formed theorem contains both the theorem and negations of theorems, as Hofstadter puts it (2000: 71). He points to the significance of recursion within computer science, in moving from level to level in an operation whilst storing in memory all previous levels such that operations might be returned to in 'recursive transitions' (2000: 128).

Following the dialectical principles described, this section has tried to introduce some of these technical principles, in order that they can be employed in the development of the overall thesis. As with recursion, it is hoped this is evident in the interlocking structure of any written thesis (such as this), and the way it returns to previous arguments at various points along the way, just as in programming parenthesis is used to produce sub-clauses, whilst retaining the overall flow of logic. Systems theory describes these dynamic interactions at both a technical and cultural level of understanding.

4.3 - complexity and dialectics

The recent interest in complexity theory derives from an understanding of dynamic systems. That a relatively small input can have massive consequences suggests analogies between complexity and social phenomena, even with history. In the concluding section of this chapter, the intention is to draw together an understanding of complexity theory and dialectical materialism - and the way both engage with systems - to stress their openness to ideas of transformation. Despite inevitable confusions, the suggestion is that together complexity and dialectics can help to identify some of the material working conditions within systems and networks. Although some of these principles have been introduced in the previous chapter through emergence, more detail will be added in this section to stress the possibilities of social transformation through disorder. The significance for dialectics is that the role of negation is upgraded to disorder, and that new order can be created out of disorder.

The contradictory phrase 'orderly disorder', taken from Hayles's 'Chaos as Orderly Disorder' encapsulates the phenomenon for the purposes of this thesis (1989: 305-22). Evidently, the relationship between order (that which can be classified and rationalised) and disorder (that which cannot, because it is too chaotic and generalised) does not lie simply in opposition but in more complex formations. What has been discovered by the science of complexity is that within the unpredictability of chaotic systems lie deep structures of order. In complex systems, what may appear to be unpredictable and closed, remains open to influence from internal and external factors.[22] The scientific basis for this can be explained by the notorious unpredictability of the weather. In Edward Lorenz's 1963 paper 'Deterministic Nonperiodic Flow', it was shown that very simple and small differences of input could have overwhelming consequences in terms of output. This has become known as the ‘butterfly effect’ in ‘popular science’ literature to describe the possibility of changes in weather conditions resulting from the movement of the wings of a butterfly in one part of the world stirring the air and thus potentially transforming the weather into storm conditions in another part of the world (from Lorenz’s essay ‘Predictability: Does the Flap of a Butterfly’s Wings in Brazil Set Off a Tornado in Texas?’ of 1979). This effect was further explained and visualised in the ‘Lorenz Attractor’ of 1963 that became known as the ‘strange attractor’ (that coincidentally looks like the wings of a butterfly), in which an ordered structure can be seen within a disorderly stream of data (Gleick 1998: 150-1).[23] The behaviour is so complex that conventional mathematical methods cannot adequately account for these processes.

Clearly there are implications here for wider social and ethical issues. In _Emergence_ (2001), Johnson describes the development of the industrial city of Manchester, revealing its chaotic emergence at the centre of the industrial revolution. He does so to make reference to Engels’s description of 1845, in ‘The Condition of the Working Class in England’, witnessing its:
‘... filth and disgusting grime, the equal of which is not to be found [...] a planless chaos of houses, more or less on the verge of uninhabitableness, whose unclean interiors fully correspond with their filthy external surroundings. And how can people be clean with no proper opportunity for satisfying the most natural and ordinary wants.’ (1978: 580, 582)

Engels is not observing chaos or an entirely unpredictable phenomenon. The city emerges according to an underlying order related to class interests. But how? Engels sought to explain this by the dialectical laws in nature but also the capitalist lust for profit, ahead of the welfare of people. In contrast, Johnson does not develop emergence as an ideological issue, but instead chooses to explain it as a ‘systematic’ complexity, and as ‘a strange kind of order, a pattern in the streets that furthered the political values of Manchester’s elite without being planned by them’ (2001: 40). Both explanations are productive. The final section of the chapter introduces the idea that systems theory extends dialectical logic. Both systems theory and dialectics share a common interest in conceptualising the material world in terms of processes, as the interaction between parts of the system and the development of the whole system that expresses dynamic interactions.

# systems dialectics

In _Systems Dialectics_ (1996), Wu Jie attempts to combine systems theory with dialectical materialist thinking, citing von Bertalanffy's _General Theory of Systems_ (1968) as a precedent for his approach. Von Bertalanffy's work is important in the development of 'second-order cybernetics', which sought to uncover some common principles that govern open, evolving systems; and he refers to 'dynamic transformation' in the case of the social realm. Wu also recognises systems theory reflects new pluralistic structures, that have replaced the post-cold war period of bipolar antagonisms – what Hakim Bey elsewhere calls the 'tweedledum/tweedledee clash of Capitalism and Stalinism' (2003: xi). That Wu is writing from the paradoxical political context of contemporary State Capitalism of Communist China, suggests that antagonisms have been internalised under free market principles (and the free market, according to Wiener, is a particularly inefficient system). Wu's position is similarly paradoxical in developing Marxist philosophy in the context of a recent history of China: reading Joseph Stalin's 'Four laws' alongside engineering physics in Russia in the 1950s, reading Mao Tsetung's work on contradiction whilst imprisoned as part of the 'cultural revolution', and then studying modern management science and systems science in the United States in the 1980s. To Wu, the time is right to argue that systems dialectics is to informational capitalism what dialectical materialism was to industrial capitalism, and this is what he refers to as its 'historical summons' (1996: 368).

Dialectical materialist philosophy can be seen to operate in parallel to the understanding of the transformation of systems, but requires a series of upgrades: from 'material object-centered theory' to 'contradiction-centered theory' to 'system-centred theory' (Wu 1996: 51-2). Constituent parts of systems, as well as the overall system itself, are continually in a state of change. Wu makes explicit reference to

Engels's 'Introduction to Dialectics of Nature' (1980 [1875-6]) that outlines the defining characteristics of the dialectical laws of motion. Echoing the phrase 'all that is solid melts into air' (in _The Communist Manifesto_), Engels summarises motion in the following terms:

'all rigidity was dissolved, all fixity dissipated, all particularity that had been regarded as eternal became transient, the whole of nature shown as moving in eternal flux and cycles'. [As a result:] 'All nature from protista to man, has its existence in eternal coming into being and going out of being, in ceaseless flux, in unresting motion and change.' (1980: 346)

That the dialectic is applicable to all nature is a highly contentious position. For instance, according to Herbert Marcuse, Engels had been wrong to assume he could apply dialectical thinking to nature as he would to history - that nature has a history but is not history (Jay 1996: 73).[24] However Engels's description of motion does sound uncannily contemporary, and Wu asserts that nature arises from differentiation and does so historically, making reference to systems theory for verification. Systems theory also confirms the truism that a system is more than the sum of its interconnecting parts. Wu quotes Hegel on this principle, that 'a severed hand is no longer a hand', to emphasise that a part is only a part in terms of its overall relation to the larger whole system (1996: 77). This applies to matter, but also in the case of information, where a node is not a node if it is disconnected from the network as a whole, and in which the interactions are expressed towards an optimisation of the whole network. Wu explains:

'So when the system is under self-organization, self-reproduction and self-catalysis, and is receiving feedback and exchanging mass, energy and information with its environment it may move and develop toward decreasing entropy and increasing order. In the end, it will gradually arrive at the optimum state of the whole system. This is what Hegel and Aristotle meant when they talked about "thesis, antithesis and synthesis" and "movement of the whole". In Hegel's view, the third category, synthesis, is the truth of the two former categories.' (1996: 81)

Although it should be added that the description of

the Hegelian position in terms of the completeness of synthesis in a universal concept like truth is misleading - in as much as continual improvement is necessary for the system not to stagnate - in preference, the synthesis should remain incomplete and open (as has been stated earlier in the thesis [25]). The development of human societies works in this way too, but is distinguished by the added element of subjective critical reflection in the optimisation process. The law of optimisation, according to Wu, is characterised by the self-perfecting process of negation of negation, part of the movement from disorder towards order and improved organisation, laying open the nature of hierarchy and diversity. He explains the logic as follows:
'This upgrades the law of negation of negation to the domain of the disordered - ordered - newly disordered - newly ordered, which is a more extensive and penetrating domain than the former.' (1996: 88)

Disequilibrium produces change in this way. It affects the equilibrium of the system, partly explaining the shift to what Wu refers to as 'synergetic equilibrium', where 'the whole system comes into a stable state in the ordered structure' (1996: 172). This is an explicit reference to Prigogine's theory of 'dissipative structure', more common in systems that are complicated and that express 'advanced motion forms'. In dialectical terms, systems express a cyclical development of equilibrium to disequilibrium to new equilibrium - or from order to disorder to new order. This explains his revision of dialectical materialism to account for an understanding of complexity theory:
'The birth of a system is a negation of nature disorder. The orderly system again contains disorderly factors, and the development of an orderly system with disorderly factors again leads the system into disorder, thereby a new orderly system is produced. [...] Without orderliness, there would be no processes, without the nature of disorder, there would be no development of processes. [...] The rule is that order conquers disorder, and disorder negates order, thus reaching a new orderly process. [...] The motional process of the whole world of systems is exactly the dialectical unity of order and disorder.' (Wu 1996: 53-4)

It is a description of emergent behaviour, in which new order might indeed be generated through disorder. This is verified by Prigogine and Stengers, who maintain that all systems contain sub-systems, and that within these systems and sub-systems, positive feedback loops might generate the further development of a process, to the point of causing a fundamental and unforeseeable change of the existing system (1985). This is an important principle, as it emphasises the constructive role that disorder might play in creating order, rather like the positive role that negation plays in dialectical materialism.

# radical uncertainty

A living system such as society, although determined by rules, is emergent and unpredictable at the same time. Like Wu, Sue Owens has also adopted theories associated with self-organising systems, to explain the possibilities of social transformation. In her essay 'Chaos Theory, Marxism and Literary History', she explains how bifurcation theory is a common explanation for how ordered structures can arise from disorder: 'At a bifurcation point, chance takes over, and it is impossible to predict what will happen; but in between times, determinism takes over again, until fluctuations force the new system into far from equilibrium conditions and a new bifurcation point is reached.' (1996: 88)

By 'bifurcation' she means splitting, the point where within a system, one path or another must be followed (as in Boolean logic). Although the choice is limited to one of two, the decision is thoroughly unpredictable. With increased frequency, bifurcations can lead to extremely complex systems. The link between bifurcation and literature here would suggest a number of examples such as Jorge Luis Borges's 'The Garden of Forking Paths' (1941), but also the tree structure of Queneau's 'A Story As You Like It' in which the reader is provided with two choices of how to proceed at each stage of the story (Motte 1998: 156-8).[26] Computer programs follow this logic of bifurcation too. The bifurcation point is 'revolutionary' in the sense that dramatic change takes place but it remains impossible to predict the direction change will take, and whether it will fall into a higher level of order or disintegrate into disorder. That the collective behaviour cannot be predicted at a global level, is analogous to the workings of society.[27] The

issue of unpredictability leads Prigogine and Stengers to explain history in terms of 'radical uncertainty' (citing André Neher). In _Order Out of Chaos_, they describe the emergent order within unstable systems such as societies, as:
'... immensely complex systems involving a potentially enormous number of bifurcations exemplified by the variety of cultures that have evolved in the relatively short span of human history. We know that such systems are highly sensitive to fluctuations. This leads both to hope and a threat: hope, since even small fluctuations may grow and change the overall structure. As a result, individual activity is not doomed to insignificance. On the other hand, this is also a threat, since in our universe the security of stable, permanent rules seems gone forever.' (1985: 312-3)

Despite the skepticism of many in the scientific community, Hayles describes the importance of Prigogine's work to validate 'the dialectic between order and disorder by finding analogous processes in physical systems. Moreover, it imparts an optimistic turn to such processes by positing them as sources of renewal [...]' (1991: 14). The description of the dialectic of order and disorder allows for the unpacking of deterministic or totalising theories and the possibility of conceiving positive change. This is in marked contrast to other contemporary cultural theories, that tend to deemphasise order in favour of randomness. [28] Many scientific theories fall into the same trap. For instance, Hayles points to the mistaken 'belief that the science of chaos opposes globalising theories is, then, a misapprehension about how these theories work' (quoted in Owens 1996: 90). The same can be said of the manner in which postmodernism became a totalising theory on the subject of anti-totalising theory. If every attempt to provide an anti-totalising theory becomes a totalising theory in itself, one response is to emphasise contradiction. Contradiction between parts is required for the complex whole to adequately describe the ways in which these parts express both disorder and order. Thus a postmodern discourse around fragmentation is rejected for an 'ordered complexity', that is neither ordered nor disordered but both. Along these lines of thinking and in general terms, orthodox postmodernism (deconstruction, post-structuralism et al) rests on 'bad science' and 'bad history', claims Owens (1996: 94) –

and bad politics of course.

This is a forceful argument but one not without difficulties. Clearly, disorder is not necessarily conceived of in these dialectical terms.[29] Many critics of dialectics think it too mechanistic. For instance, Prigogine and Stengers are no supporters of dialectical materialism, and despite recognising the undoubted similarities to the new science, think it too extreme in its rejection of determinism (1985: 252-3). However, this is a misunderstanding in the broader sense, as dialectical materialism does not see everything as entirely fluid and in flux but simply more so than conventional deterministic systems, according to Owens (1996: 105).[30] Paradoxically, she supports this claim for the coexistence of determinism and flux by quoting Prigogine and Stengers: ‘being and becoming are not to be opposed one to the other: they express two related aspects of reality’ (1996: 102). Determinism and unpredictability are held together in a manner that reflects open systems and a more open view of dialectics.

It is worth emphasising that the combination of systems theory and dialectics challenges the pessimism of much contemporary critical theory, by suggesting the possibility of transformation coexisting with a tight structural framework. It encapsulates the idea of ‘orderly disorder’, wherein positive change remains a possibility. In keeping with these explanations, this chapter has argued that dialectics continues to remain a useful conception and model of change, particularly to describe systems that appear to contain the same logic. Whether it is a law of nature, as Engels and Wu argued, seems debatable but an understanding of both complexity and dialectics manages to draw together the interconnections of nature, history, society, technology and politics. It also affirms that human subjects are constituted through their relationship to society and institutions, and that society cannot be described simply as a collection of individual subjects. Rather, it is a far more complex system that takes account of individual differences but also of collective actions and counteractions. The constructive role of disorder is consistent with Tiziana Terranova’s position in _Network Culture_, in which she describes informational dynamics as ‘creative destruction, that is a _productive_ movement that releases (rather than simply inhibits) social potentials for transformation' (2004: 3).

The suggestion is that software releases the social potential for transformation too. The following chapters add detail to this claim, first by focussing on work (chapter 5) and later action (chapter 6) to establish a better understanding of the dialectics of software art. This approach provides the possibility of change through human agency – at the point of bifurcation where two paths are possible.

5. *complex labour*

'Work is no longer work, it is work which is liberated from work.'
(Negri 1991: 160)

Attention to labour relations was, in the industrial period, the foundation for a dialectical materialist methodology. Clearly circumstances have changed since then and so various updates are required. However the importance of labour has not disappeared but has been transformed. In section 5.1, the labour of people and machines is articulated to take account of complex interactions, in what has been called 'machinic integration'. This integrated labour extends social relations from the interaction of workers to their interaction with machines. Consistent with the principle that the site of production remains where social antagonisms are expressed, the suggestion is that these new interrelations of humans and machines also presents new possibilities for operating in a disorderly manner. Rather than reject dialectics, as has been the case with much post-Marxist commentary, this thesis argues that contradiction needs to adapt to the times, to take account of the ways in which social relations are expressed in complex formations. This is how ideas introduced in the previous chapter become significant in practice.

In section 5.2, these new forms of work are further explained by making reference to ideas developed by autonomist Marxism, and in particular through the concepts 'immaterial labour' and 'general intellect'. These ideas result from the perceived importance of communicative interconnections in what the autonomists call the ‘social factory’ to describe the way in which the mode of production has been extended to the whole of society. What was once considered the living contradiction of labour relations in the factory has been been extended by collectivity and networked communications technologies. The concept of general intellect, drawn from Marx’s early writing, is considered to be particularly useful. For instance, the open source movement is organised in collective and open forms, to allow for the sharing of source code and expertise.

These developments present new contradictions expressed in the complexity of labour, characterised not least by the prevalence of free labour in the cultural realm, and the production of free software in particular. Section 5.3 also raises issues over the effectiveness of oppositional tactics that aim to respond critically to these conditions, such as the refusal to work. Negri’s quote at the beginning of the chapter is suggestive of the possibility of work that rejects the conditions under which work is currently performed. The discussion of work is allied to software work in a deliberately ambiguous way - to indicate both the work involved in making software, as well as the work that software does itself. The chapter ends by arguing for the continued relevance of dialectics to respond to these working contradictions, by drawing upon ideas introduced in the previous chapter.

5.1 - complex workers and machines

Classical Marxism would maintain that the relations of production constitute the base from which the superstructure is derived, at any point in human history. All social relationships for Marx lie in social production and relationships between people in production. These ideas are predicated on the understanding that there is a dialectical relationship between nature and human society integrated through labour. Human production emerges from nature, then utilises it and abuses it. In addition, technology energises the labour force, and hence the force of social development, combining human and machine labour working at local and global levels of complex interaction.

In _Chaosophy_ (1995), Félix Guattari refers to 'systematic disorganisation', that plays an important part in understanding both individual and collective forms of subjectivity. Although working from the assumption that subjectivity is constructed according to social and economic conditions, Guattari is breaking with Althusser's Marxist structuralism (as well as Lacan's psychoanalytic structuralism) that stressed the determining role of language and communication. Rather than simply taking technology as part of a process of interpellation, something far more complex is articulated, that involves the interaction of the labour of people and machines. So rather than the dead labour of machines replacing human labour, he states:
'On the contrary, I think that machines must be used - and all kinds of machines, whether concrete or abstract, technical, scientific or artistic. Machines do more than revolutionize the world, they completely recreate it.' (1995: 19)[1]

According to this position, any concept of social production and the relations of production must take account of more complex and disorganised interactions between people and machines - what Guattari calls 'machinic agency'. The disorganisation, or what he also refers to as 'craziness' in systems, is accounted for in his reference to chaos (in the term 'chaosophy'). Central to this idea is that change does not simply happen on a large-scale socio-economic level or in ideology but from mutations at a micro-scale molecular level. Here the link to complex systems would verify that a relatively small input can have massive consequences.

The first section of this chapter investigates the possibilities for social transformation, in the 'machinic integration' of the processes of production, circulation and information. Guattari would suggest that 'a mutation like that introduced by microprocessors changes the actual substratum of human existence and, in reality, opens up fabulous possibilities for liberation' (1995: 47-8). In other words, there is a dynamic tension between micro-politics and the body politic in general - integrating life and politics at all scales of operation. Guattari describes this reorientation of thinking as moving from 'dream to social reality, from poetry to science, from the most violent social reality to the most tender daily relations' (1995: 50).

In forging new ways of conceptualising these issues, Guattari calls himself an 'idea-thief'. Concepts are taken to be tools, not fixed or universal ideas, but ideas in flux, open to influence from other fields of interest - such as combining the seemingly heterogeneous fields of chaos theory and philosophy in the case of 'chaosophy'.[2] The combining of the heterogeneous fields of complexity theory and dialectics to open up new critical possibilities, was introduced in the previous chapter in a similar way.

# desiring production

The spheres of production, distribution and consumption have been considered relatively autonomous in classical Marxism. Guattari, with Gilles Deleuze, argues that this is predicated upon Marxist description of the division of labour and the idea of false consciousness. To them, the distinctions collapse, making everything production: the 'production of productions, of actions and of passions' (1990: 4). Therefore when Deleuze and Guattari claim that nature is now experienced as a process of production, they are arguing something quite different from Engels or Wu (described in the previous chapter). Their understanding incorporates the production of subjectivity itself, machine production and consumption, making the human subject a 'producer-product'.

Like the work of the Frankfurt Institut that preceded them, Deleuze and Guattari are drawing together Marx and Freud to open up new possibilities for the unconscious to be seen as productive and not simply 'false', although their emphasis is on desire rather than history. For instance, in _Anti-Oedipus: Capitalism and Schizophrenia_ (1990 [1972]), the unconscious is cast as a factory not a theatre (and thereby they aim to reject the oedipal drama[3]). This is a reference to the work of Antonin Artaud, who described the body as a factory, or more accurately the sick body as an 'overheated factory' (Guattari 1995: 75). In Artaud's 'Theatre and the Plague' (2001 [1964]), disorder in the form of the plague demonstrates the potential for transformation of the body and body politic:
'The plague takes dormant images, latent disorder and suddenly carries them to the point of the most extreme

gestures. Theatre also takes gestures and develops them to the limit. Just like the plague, it reforges the links between what does and what does not exist in material nature. [...] It restores all our dormant conflicts and their powers, giving these powers names we acknowledge as signs. Here a bitter clash of symbols takes place before us, hurled one against the other in an inconceivable riot.' (2001: 18)

For Artaud, the plague disrupts human progress (order) and encourages irrationality (disorder), unleashing the potential for radical change. For Deleuze and Guattari, the liberation of creative social expression is bound up with desire, as it is capitalism that represses desire (precisely because desire is where the potential for transformation lies). There is a further link to the work of the Frankfurt Institut here, in the work of Marcuse in particular, who speculated on the power of sexuality to unsettle the repressive work ethic that sustains capitalism (in _Eros and Civilisation_, 1972).[4] In Marcuse's terms, non-work or play allows the freeing of desire. Art is part of the potential solution for Marcuse, as it contains both a critical and anticipatory function that ties it to politics. According to Deleuze and Guattari too, the failure to liberate desire sufficiently accounts for the failure of social revolutions thus far. In fact, to Deleuze and Guattari, all ideologies, even oppositional ones, mask desire, or repress it, and so as not to replace one repression with another, it is desire that requires liberation.[5]

As such, desire can be seen to be potentially revolutionary, especially when promiscuous and outside the 'Mommy-Daddy' family circle of traditional 'bourgeois psychiatry'.[6] This reference again draws upon the work of Artaud, who says: 'I don't believe in father/in mother,/got no/papamummy' (quoted in Deleuze & Guattari 1990: 14). The critique of the Freudian Oedipal drama forces the analysis out of the family, to the wider mechanism of power that would include other social norms, many of which are irrational. This position owes something to the anti-psychiatry movement of Ronald D. Laing:
'In the context of our present pervasive madness that we call normality, sanity, freedom, all our frames of reference are ambiguous and equivocal. A man who prefers

to be dead rather than Red is normal. A man who says he has lost his soul is mad. A man who says that men are machines may be a great scientist. A man who says he is a machine is 'depersonalized' in psychiatric jargon. [...] Thus I would like to emphasize that our 'normal' state is too often the abdication of ecstasy, the betrayal of our true potentialities, that many of us are only too successful in acquiring a false self to adapt to false realities.' (1965: 11–2)

Indeed the capitalist machine is a rational framework under an irrational impulse, according to Guattari: 'Everything is rational in capitalism, except capital or capitalism itself' (1995: 54). Capital can simply be diagnosed as demented and at an advanced stage (similar to Mandel's description of capitalism as senile dementia), hence Deleuze and Guattari s use of the term schizophrenia. The solution is not to impose an alternative ideology but liberate desire, what they call 'desiring-production'. Desiring machines follow rules of association, and are networked:
'The productive synthesis, the production of production, is inherently connective in nature: "and..." "and then..." This is because there is always a flow-producing machine, and another machine connected to it that interrupts or draws off part of this flow (the breast – the mouth). And because the first machine is in turn connected to another whose flow it interrupts or partially drains off, the binary series is linear in every direction. Desire causes the current to flow, itself flows in turn, and breaks flows.' (1990: 5)

These breaks and flows evoke information networks and this is made explicit in Deleuze and Guattari's reference to code. They say that every machine has code built into it: 'The data, the bits of information recorded, and their transmission form a grid of disjunctions of a type that differs from the previous connections' (1990: 38). This appears to describe the differences between centralised, decentralised and distributed networks (described in the previous chapter). Desiring-production is networked, in that every machine is connected to another machine. Deleuze and Guattari proceed to draw a parallel between desiring-production and social production, allowing them to assert that capital is the 'body without organs' of the capitalist (from Artaud's phrase). The body without organs lies in opposition to desiring machines, in the realm of ‘antiproduction’ rather than desiring-production: ‘The genesis of the machine lies precisely here: in the opposition of the process of production of the desiring-machines and the non-productive stasis of the body without organs.’ (Deleuze & Guattari 1990: 9)

In comparable terms to the opposition between labour and capital in classical Marxism, desiring-machines can be seen to operate in parallel to labour in opposition to capital. Similarly, the body without organs can be seen to appropriate desiring production, just as the capitalist extorts value from labour. It is not simply labour that is stolen but desire too, or more precisely the energy associated with desire that is central to this extortion and its opposition (although it should be added that Marx also regarded labour as a living energy). What is distinctive in this formulation is that desire creates flows between units of production, and like the unconscious, fears it ‘lacks’ something and so strives for connectivity. However, it is not wholeness that is sought (for instance, as with the mother in a Freudian scenario), but something far less unified and multiple. This leads Serge Leclaire to think that the desiring machine is a ‘partial object’ in the sense that psychoanalyst Melanie Klein introduced. Klein explains that humans pretend that things are perfect and whole, to avoid the reality that they are flawed and in parts. She says: ‘It is a ‘perfect’ object which is in pieces’ (1988: 270). Deleuze and Guattari seem to concur: ‘We believe only in totalities that are peripheral. And if we discover such a totality alongside various separate parts, it is a whole of these particular parts but does not totalize them; it is a unity of all these particular parts but does not unify them.’ (1990: 42) But this does not really address Leclaire’s observation that Deleuze and Guattari’s theory is too ‘perfect’. He thinks it does not demonstrate flux sufficiently.[7]

In the context of this thesis, it is useful to stress the description of the desiring-machine as a system of interruptions or breaks, and one that works paradoxically by breaking down and becoming dysfunctional to Capital (in Guattari 1995: 103). As a consequence, the theory comes closer to systems dialectics, where disorder can be seen to be a catalyst for changes to the system. The machine can be seen to possess two characteristics, according to Guattari: the power of continuum and the rupture in direction or mutation. The machine, therefore, is a 'break-flow' process of connections and their rupture (1995: 126-7), in parallel to the idea that change takes place through a rupture in the continuum of history. Desiring-machines operate in this disruptive manner, and one might speculate upon desiring-software that represents a break-flow process, or indeed orderly disorder. Software, in these terms, might rupture the continuum, by doing something other than expected, perhaps simply by refusing to work or remaining non-executable, or crashing the machine it runs upon. These ideas will be developed later in this chapter.

# marx after marx

The concept of 'machinic agency' is one way to take account of desire and extend an understanding of production. Neither machine nor software simply acts on its own. In themselves, they demonstrate no sense of agency in promoting desire or repressing it. This is why Guattari would regard technology acting on its own (technological-determinism), under a sense of autonomy, as necessarily expressing a fascist tone in oppressing desire. Yet Guattari thought Marx mistaken in thinking social relations lie outside of the tool or machine. The issue can be traced historically, by the evolution of the tool into a machine that becomes more and more independent of the worker. To clarify this, Guattari reiterated Marx's distinction between machines and tools, in that machines are a factor of communication, whereas tools merely extend control through direct contact. This is a much misunderstood distinction between tool and machine, and one often repeated in connection to the computer. Marx claims: 'the tool is a simple machine and the machine is a complex tool [...] therefore, is a mechanism that, after being set in motion, performs with its tools the same operation as the worker' (1990: 492, 495). The machine extends the limits of human effort, and becomes part of a wider scheme of machines working together collectively, as part of an extended industrial (machinic) apparatus. The 'machinic' relations between worker and machine, although prefigured in Marx's description, does not go far enough for Guattari, who describes the worker and tool as part of the machine (1995: 142) - indeed both are engineered ever more overtly.

Building upon the thinking of Deleuze, Negri sees this conceptual trajectory evident in Marx’s early work. [8] The complexity of the argument is exemplified by Negri’s book title _Marx Beyond Marx: Lessons on the Grundrisse_, paradoxically expressing that what lies beyond Marx is in itself a return to Marx. This represents a return to the _Grundrisse_ of 1857-8 (1981) for its conceptual openness and rawness. Thus it is considered ‘beyond’ _Capital_ in conceptual but not historical terms - as the _Grundrisse_ notebooks predate and inform the argument of _Capital_ of 1867 (1990). An orthodox historiography would find this ‘beyond’ yet simultaneously ‘before’ problematic, but the paradox reflects processes of renewal and rupture.[9]

The preface to _Marx after Marx_ is written in the form of a dialogue between a prisoner and free man, reflecting Negri’s lengthy imprisonment. Clearly this is not simply autobiographical but allegorical: working in the factory is to be seen as equivalent to a jail sentence, to break out of jail is to break from capital (1991: xvi). The free man (Negri’s former and future self) states: ‘To be a communist today means to live as a communist’; to which the prisoner (Negri’s present self at the time of writing) responds: ‘This, I think, is possible even in prison. But not outside, until you free us all’ (1991: xvii). Here the dialogue evokes the dialectical Hegelian master-slave relation, in that it is only the slave who can become truly free. That is to say: ‘here, domination and reversal can only be accomplished by those who participate in an antagonistic relation’ (Negri 1991: 9).

When Negri uses the term antagonism, it is worth mentioning that he is a militant, exemplified in his views on the necessity of violence: ‘To suppress the violence of this process can only deliver it - tied hand and foot - to capital’ (1991: 173). Negri’s uncompromising position is bound up with the specific political context of Italy in the 1970s and the failures to engender social transformation (particularly the major strikes around the Fiat Factory of 1970). His position corresponds with the ‘Workerism’ movement [‘Operaismo’], that paradoxically, was against work in the sense they did not want it re-appropriated, but simply reduced. Rather than celebrating workers’ labour, the influence of Simone Weil (who experienced the factory production line first hand) is evident in questioning whether it was possible at all to conceive of production that was not oppressive (explained by Sylvère Lotringer, in Virno 2004: 8).

But Negri would not merely resort to dialectics here. Rather than see the critique embedded in its internal contradictions, Negri also points to another non-dialectical dimension. In addition to the imposed unity of dialectical relations between worker and capitalist, there is another logic of 'separation' from forms of domination. Like the revisionist work of the Frankfurt Institut, the principle here is that capitalism as an irrational system cannot be replaced by anything that employs the same logic. For Negri, dialectics is the temporary logic of capitalist times and part of its internal contradictions to be overthrown, along with its domineering class in the 'transition' from socialism to communism. Nevertheless, if one resorts to dialectics, this might be described as the negation of negation, in that the dialectical method once employed is further negated (as described in chapter 3). In other words, to the dialectician the issue of whether this is dialectical or non-dialectical appears as a dialectical conjunction in itself (this issue will be developed in the last section of this chapter). Although not following this logic himself, Negri is anxious to find a method that can respond to a power base that is ever more complex.

This is where Negri finds Michel Foucault's concept of 'biopower' productive, as a concept of power that is multiple and adaptive. Foucault's description of biopower is a significant intervention, operating in the tradition of materialist production and the ways in which subjectivities are constituted in complex and interactive relationships. To Negri, the idea of globalised biopolitical production does not mean that class antagonism has disappeared but is present in the wider social realm, and in everyday life: 'life is made to work for production and production is made to work for life' (Hardt & Negri 2000: 32). He argues that antagonism grows stronger as a result of these changes but stresses that Marx does not go far enough in describing the dynamic of capital. What is missing in Marx is an understanding of power in relation to

systems theory, and the importance of the machine that encapsulates production:
‘The machine is self-validating, auto-poietic - that is, systemic. It constructs social fabrics that evacuate or render ineffective any contradiction; it creates situations in which, before coercively neutralising difference, seem to absorb it in an insignificant play of self-generating and self-regulating equilibria.’ (Hardt & Negri 2000: 33-4)

The description of contemporary power as an adaptive system also encourages the view that alternatives can adapt too. In Negri’s view, power operates not through contradiction but through separation and it does this currently through ‘the world market’ (or what is commonly known as globalisation). He argues that the critical strategy of contradiction needs to be reinstated in the realm of production, as this remains where social inequalities are revealed and where alternatives arise. The potential for transformation lies in identifying and acting upon these contradictions. This section has tried to outline some of the concerns that any reconceptualisation of work must take account of. Importantly, the complexity of social relations operates through interactions of machines and people, as does any sense of machinic agency involved in the transformation of these relations. It is clear that machines cannot simply be regarded as ‘dead labour’ but are integrated into ‘living labour’, and involved intimately in disseminating creative human energies more openly and widely - for better or worse.

5.2 - work after work

In orthodox Marxism, the capitalist mode of production simultaneously produces and reproduces the antagonistic social relations between labour and capital. This situation is based on the need for workers to sell their labour and the corresponding need for capital to ‘extort’ value from the workers. The antagonism that arises from this is expressed particularly directly in the German language: ‘Arbeitgeber’ (labour-giver) and ‘Arbeitnehmer’ (labour-taker). As the previous section makes clear, labour is no longer contained by the factory walls, and as a consequence this serves to dislocate class antagonism.

This process of dislocation is what Marx referred to as 'real subsumption', to conceptualise the way that class exploitation is dispersed and subsumed into the wider (global) social realm.[10] This is clearly more evident under contemporary conditions than in the mid-nineteenth century, but the logic is consistent. What Negri referred to as 'separation', at the end of the previous section, is another example of dispersion, in which capitalism restructures itself to avoid dissent and to diminish contradiction. For example, the dissent of the 1960s and 1970s (notably the student and worker movements in Italy, and Paris in 1968) was acted upon in the 1980s and 1990s by spreading class antagonism far and wide, and undermining its potential for collective action. At the same time, real subsumption, assisted by informational technologies, has transformed labour and made it more shared, collective, and communicative. This second section of the chapter introduces these ideas in relation to the concept 'general intellect', drawn from Marx's early writings, that refers to the combination of socialised labour and technological expertise that has become important to production. Any critique of exploitation therefore must recognise social relations in terms of what the autonomists call a 'social factory', to describe the way the whole of society is turned into a site of production.

For capitalism to continue to produce surplus value, it has to construct not simply commodities, but also the appropriate subjectivities to do so. Subjectivities are constantly being generated and corrupted in the 'factories of subjectivity', claim Hardt and Negri (2000: 197), in a phrase that echoes an understanding of biopower described in the previous section. In the 'social factory', subjectivity as well as labour value is stolen from the worker (or 'autonomous subjectivity' is denied, in the terms of autonomous Marxism). Negri explains that two oppositions are at work: between use value and exchange value of orthodox Marxism, and in addition 'objectified labor against subjective labor' (1991: 68). In this latter opposition, Negri draws on a passage from Marx's _Grundrisse_ to characterise labour in terms of the subjectivities of 'worker and capitalist, collective worker and collective capitalist' (1991: 77).

Oppositional subjectivities reflect these conditions of the social factory (often referred to as the 'multitude' in contemporary commentary[11]). The term 'proletariat' as the subject of labour and revolt continues to be relevant but requires redefinition to stress more collective and communicative forms.[12] The redefinition is further explained by its original meaning, describing someone who only has the ability to reproduce themselves, according to Peter Linebaugh, extending the Marxist interpretation applied to someone with only their labour to sell (Dyer-Witheford 1999: 107). Hardt and Negri define the term to include all those whose labour is directly or indirectly exploited, and argue that labour is becoming ever more proletarianised. Negri goes further, and drawing upon an understanding of machinic subjectivity, claims that capital tries to capture the communicative capacity of the socialised labour force and turn it into information - even into software. From this, it can be deduced that subjectivity and technology are also becoming proletarianised.

Consequently, the control of communications, and the labour related to communications, have become key sites of antagonism.[13] For instance, in software production, the contradictions that arise from open source principles are bound up with the way society responds by both encouraging and limiting software development: on the one hand, by employing the technical possibilities of the Internet that facilitates free and easy information sharing, and on the other, by exploiting the commercial benefits through proprietary licensing. This second section of the chapter draws upon an understanding of the concept 'general intellect' to reveal some of the contradictions in the relations of production between networked machines and collective human labour. The issue of property is at the core of this.

# general intellect

The source of the concept 'general intellect' is a section in the _Grundrisse_ called 'Fragment on Machines' written in 1857-8, in which Marx describes that at a certain point in capitalist development, real wealth will be measured not on labour time in production but on technological expertise and organisation (1981: 705-6). In summarising the concept, Nick Dyer-Witheford stresses the importance of what Marx calls 'general powers of the human head', 'general social knowledge', and 'social intellect', all resulting from the increasing importance of machinery (1999: 220). Marx predicts that the productive forces of the intellect, of human knowledge and skills, will be incorporated into capital itself - into what has since become known as the 'knowledge-based economy'. Marx was thinking of the developing importance of automatic systems for production and the networks of the world market. The crucial issue, both then and now, is that general intellect unleashes contradictions by combining technical knowledge and social cooperation. For instance, increasingly socialised labour and the replacement of labour by machines undermines existing hierarchical structures that protect private property, wage structures and class exploitation.

It is in this context too, that the concept 'immaterial labour' is introduced by Maurizio Lazzarato and Negri to describe the nature of work, in a scenario where information and communication dominate the process of production. Immaterial labour is that which produces immaterial goods such as services and knowledge. It follows that as commodities and wealth have become less and less material and more defined by cultural, informational factors and knowledge, so too has labour. According to Lazzarato, labour constitutes itself in forms that are collective and, in terms of the network and flows, no longer just confined by the walls of the factory in a 'mutation of "living labour"' (1996). The productive labour of the industrial factory is becoming replaced by intellectual, immaterial and communicative labour, making everything like a factory and changing the social relations therein (this is where the term 'social factory' applies). Furthermore, the concept of immaterial labour describes a rupture in the continuity of production, that breaks away from the centrality of waged labour in orthodox Marxism. Lazzarato says: 'A polymorphous self-employed autonomous work has emerged as the dominant form, a kind of "intellectual worker" who is him- or herself an entrepreneur, inserted within a market that is constantly shifting and within networks that are changeable in time and space.' (1996: 139)

As a consequence of more emphasis on intellectual and creative work, the concept of immaterial labour accounts for the ways in which new management techniques appear to emphasise innovation, enterprise and problem-solving. All the same, these new techniques, such as ‘participative management’ may appear ‘flat’ rather than hierarchical but are still ‘techniques of power’ (Lazzarato 1996: 134). As with the discussion of network control in the previous chapter, power is exerted through _ facilitation_ rather than direct repression. Lazzarato thinks participative management techniques are more totalitarian than the production line, as they involve the willing subjectivity of the worker in the process (1999: 224). However, conflict still arises between capital’s objective control and the relatively autonomous, subjective nature of the work. An activity such as hacking is a good example of the contradiction at the heart of capital’s attempt at control, as it is both a necessary and potentially disruptive skill and embodiment of technical knowledge in the information factory. This example will be expanded upon in the next section, but hacking represents technical knowledge useful to fix a problem or create a problem. The context determines whether complimentary or derogatory meanings are implied.

These contradictions are particularly evident in new collective formations that utilise the networked technologies. For instance, Terranova regards mail lists as crucial to the development of network-organised forms of political organisation, enhancing connectivity and the open sharing of ideas. The composition of contemporary forms of protests in general, rejects the centralised form of mainstream broadcast media, for a counter-position that is distributed and collective (often referred to as a shift from a ‘one-to-many’ to a ‘many-to-many’ model of communication). Indymedia, a collective of independent media organisations and hundreds of journalists offering grassroots, non-corporate coverage, is an example of this model. Such examples allow Terranova to argue that the Internet materialises ‘general intellect’ that ‘implies the release of a surplus value of potential’ (2002).

Applied to the development of software in general, the collaborative gathering and analysis of information is reflected in the open source movement and what Felix Stalder and Jesse Hirsh call ‘open source intelligence’

(2002). They point to open source principles of peer review, the free sharing of products, and flexible levels of involvement and responsibility that are all derived from practice and the technical possibilities of the Internet technologies, that facilitate free and easy information-sharing among peers, exemplified in ‘peer 2 peer’ networks. Stalder and Hirsh’s examples are varied in scope: from the ‘collaborative text filtering’ of the _nettime_ mail list, itself running on the open source list package ‘majordomo’, to _Wikipedia_ the free encyclopedia built on open source principles, and the technological platform of ‘Wikiweb’, in which users can see the source code but also freely edit the content, archived and published under the GNU public license (2002). These examples illustrate general intellect and the ways that technical expertise and socialised labour work together. However, the ‘potential’ that Terranova refers to demonstrates both positive and negative tendencies, both releasing and limiting possibilities for future transformation.

# open source intelligence

The effectiveness of ‘open source intelligence’ is clear (in contrast to what might be called ‘proprietary stupidity’). It is arguably more reliable, stable and less bug-ridden, as a result of peer review and collective development. There are numerous examples of high quality applications, operating systems and platforms that have been developed utilising collaborative programming and development environments. For instance, Linux was recognised by Microsoft as being superior to its own Windows operating system in 2000, and since, it has become the orthodoxy to develop even commercial software in this way. Fundamental to the commercial interest is the tradition of open source as a development method, based on a belief in ‘shared culture’ and ecology. Re-using existing code is part of this working principle, to avoid unnecessary work and the overproduction of code. Eric S. Raymond describes this in the following terms: ‘This attitude gives the best return both in the “soft” terms of developing human capital and in the “hard” terms of economic return on development investment.’ (2004: 375)

Open source principles both contest and affirm capital investment on human and economic levels. The contradictions reveal themselves in anomalies like the floatation of the open source supplier 'Red Hat' on the Stock Market, and can be seen to subsidise rather than undermine corporate capital. The apparent openness to commodification of 'open code' requires a distinction to be made between open source and free software, along ideological lines. This is what Richard Stallman (and the Free Software Foundation) attempts when he clarifies the distinction: 'To understand the concept, you should think of "free" as in "free speech"' (1996). The 'rasta-programmer' Jaromil goes further in referring to freedom from slavery to proprietary software. He has produced _dyne:bolic_ (2001) with these principles in mind: a GNU/Linux distribution on compact disc, with a useful assortment of applications designed to run on old computers and even on Xbox games consoles. In both references to free software, a political concept of freedom is emphasised, whereas open source is linked with pro-business computer libertarians, and the idea of releasing source code and developing software collaboratively as a potential antidote to the market dominance of Microsoft (Medosch 2005: 182).[14] For Lawrence Lessig too, in _Free Culture: How Big Business Uses Technology and the Law to Lock Down Culture and Control Creativity_ (2004), free speech is also evoked by quoting Thomas Jefferson on the nature of ideas, claiming it is a fundamental right for ideas to remain freely available in the public domain.

That ideas are not free once made tangible, and are subject to intellectual property rights, is a byproduct of an economy that has sought to commodify knowledge. Capital tries to treat knowledge as it would any other goods but does not always succeed in commodifying it, because knowledge cannot simply be reduced to the market. This is a point that Lazzarato adds, drawing upon the work of the sociologist Gabriel Tarde. The example given is the production of books, and how the exchange value of a book can be determined by the market as a product but not as knowledge, which is more determined by moral issues of gift or theft (Lazzarato 1999: 162). It is easy to see evidence of this in the issue of intellectual property and what should or should not be in the public realm.[15]

Another consequence of this knowledge economy is that places of learning are drawn closer to capitalism. At the same time, it is argued by Dyer-Witheford (2005), drawing upon Lazzarato and Negri, that this also releases the potential for a more effective opposition (shifting antagonism from the industrial factory to the knowledge factory). For instance, as universities concentrate their energies on engineering and technology disciplines, forms of dissent employ the same tools. An example of this would be the work of The Institute for Applied Autonomy, who attempt to undermine the normalised ambivalence to art and social issues that characterises engineering practices, by their tactical interventions using robotics and mobile technologies (2005). The wide and free distribution of knowledge over the Internet is clearly a threat to the traditional learning institution (and the ideological state apparatus), and this has contributed to the rise of alternatives such as 'free universities' linked to the principle of 'open source knowledge'. Henriette Heise and Jakob Jakobsen describe an example of this: 'The Copenhagen Free University opened in May 2001 in our flat. The Free University is an artist run institution dedicated to the production of critical consciousness and poetic language. We do not accept the so-called new knowledge economy as the framing understanding of knowledge. We work with forms of knowledge that are fleeting, fluid, schizophrenic, uncompromising[ly] subjective, uneconomic, acapitalist, produced in the kitchen, produced when asleep or arisen on a social excursion - collectively.'

The University of Openness, in London, is another example of what has become known as a 'self-institution' that allows individuals and organisations to pursue their shared interest in emerging forms of cultural production and start a 'faculty' to socialise their research.[16] Its Faculty of Unix offers free workshops as an alternative to the commodification of knowledge and of proprietary systems in general, demonstrating the potential of open source knowledge. Unix is a strategic choice of study in this respect, as it represents a folk tradition of 'bottom-up' development where 'expertise' comes from the shared culture itself, and the idealistic logic that a better cultural understanding of technology or indeed software will lead to 'better' implementation (Raymond 2004). Developed through an engineering tradition, Unix undoubtedly has a technical culture but also a conceptual and political culture (Raymond's book

The Art of UNIX Programming (2004) is testament to this).

Proprietary software might be an inherently flawed concept on many levels, but private property remains a cornerstone of capitalism. The legal apparatus is particularly slow to adapt to the ways in which this paradoxical logic is challenged by the knowledge and information economy. This is the basis of the argument that Stallman makes in his ‘Why Software should not have Owners’, that to think of software in terms of material goods and to adopt its legal protections is anachronistic (1994). The legal apparatus remains poorly suited to dealing with the results of immaterial labour, as it still tends towards distinctions of property in terms of end-products after the making act, unable to deal with dynamic processes (Barron 2002). Thus even on its own proprietary terms of turning information into commodities, the legal system fails to deliver. In the context of these inadequacies, alternative licenses, such as those provided by the Creative Commons initiative, are useful in providing flexible copyright licenses for creative works, but do not adequately address the criticism that they represent the potential for the further commodification of creativity and cultural work. Rather than relying on the legal apparatus (and particularly with Creative Commons in mind), David M. Berry instead proposes the ‘Libre Commons’, rejecting ‘bureaucratic attempts to overcode the social through law’ and instead affirms the ‘intersubjective recognition and affirmation that commonality provides’ (2004).[17] Rather than deciding which license to use (from the pull down menu on the Creative Commons web site for instance), an alternative is simply deciding to reject the legal apparatus altogether. A weakness of open license agreements remains that they do not challenge intellectual property law at source - on the issue of property. In contrast, Berry’s position is in recognition of the importance of intellectual property and the antagonisms that arise from the issue of common property.

Property thus remains, as it did at the time of Marx’s writing, as a key area of antagonism but its emphasis has changed in its application to software. As for software art, the issue of property is something that Robert Luxembourg’s _The Conceptual Crisis of Private Property as a Crisis in Practice_ (2003) comments upon.

A program script (crisis.php), an explanatory text file (crisis.txt) and a screenshot (crisis.png) are presented as a conceptual puzzle. If the program is run, it parses the screenshot into the full text of the novel _Cryptonomicon_ by Neal Stephenson (of 1999). The project thus forms a neat conceptual loop between form and content, addressing issues of encryption, privacy and intellectual property rights. It allows the reader to gain access to the novel in such a way that the author of the software remains within the legal constraints imposed on the author of the novel by the publishers. The lengthy title of the work indicates the critical trajectory of this work, based on a quote from Hardt and Negri's _Empire_ (2000), and tests the limits of the legal apparatus that Hardt and Negri see as underpinning the power structures of contemporary capitalism. Property rights are only infringed on execution of the script. The software on the other hand is distributed overtly as free, open software under the terms of the GNU General Public License agreement.

On closer examination, the work emerges from earlier works by Project Gnutenberg and the production of the software that lies behind the encryption process (pngreader v1.1), that works under the principle that images and texts have the same underlying code of zeros and ones. Any output that is encoded in such a way (using the pngwriter) can be decoded (using the pngreader), allowing for the covert distribution of copyrighted materials. The user is 'instructed' to perform an illegal act by running the php script but whether he/she does this or not is beyond the capacity of the software itself. An extreme case of the use of this principle and subsequent legal proceedings was Project Gnutenburg's _walser.php_ (2002) in which the script generated a plain ascii version of Martin Walser's controversial novel (of 2002) _Tod eines Kritikers_ [Death of a Critic]. Walser.php does not infringe copyright itself, but only if executed.

The work arguably demonstrates a more radical strategy than clever licensing. It evokes piracy and plagiarism, already common practice in the wider culture, particularly in the distribution of music and movies, and peer to peer networks. Plagiarism is also an integral part of software practice in the free distribution and adaptation of source code - with or without attribution or acknowledgement. The

open source and free software movement clearly relies on forms of plagiarism for much of its work.[18] Vested interests around property are made evident in these operations of big business and the legal apparatus, in attempting to police these pirate operations.
The figure of the pirate takes on a heroic status in these debates (and in popular culture in general), partly derived from its significance through action, uprising or insurrection - what Bey calls 'pirate utopias' in his essay 'The Temporary Autonomous Zone' (2003: 97). The historical parallel to the days of early capitalism is a useful one (where slave ships also sailed), and points to some of the underlying antagonisms that open code and free culture present. Related to this line of thinking is the project _The Kingdom of Piracy_ conceived by Armin Medosch, Shu Lea Cheang and Yukiko Shikata (2004), that explored the world of free software and copyleft culture. The term piracy is used in such work, not to describe illegal activity on the high seas, but more metaphorically in opposition to the moral legitimacy of capitalist logic.

The moral issue is also something that the satiric project _Re-code.com_ highlights, responding to: 'the absurdity of a system that allows corporate theft to go unpunished while deeply criminalising petty consumer theft' (Conglomco.org & The Carbon Defense League 2004). The project web site contained instructions, scripts for generating UPC symbols, access databases of prices, etc, allowing consumers control over the prices they paid for supermarket goods. The site stayed online for ten days before being taken down in response to threats from Wal-Mart. The challenge to private property in work such as this is overt: stealing back the value and subjectivity stolen from the public in the first place. What is also stolen back is general intellect.

This section has tried to outline the concept of general intellect, to understand how the control of communications, and the labour related to communications, are crucial to the success and failure of the economy. However, this can only be a positive public force if it is at the same time political, as Paolo Virno explains:
'if the publicness of the intellect does not yield to the realm of the public sphere, of a political space in which the many can tend to common affairs, then it

produces terrifying effects. _A publicness without a public sphere_: here is the negative side – the evil if you wish – of the experience of the multitude.' (2004: 40)

These issues around open source code and work represent an enormous subject that could occupy a thesis in itself, but important in this context is that these developments and the contradictions that arise from them can be seen to respond to both social and technical conditions. The production of software makes a good case study in this respect, in as far as the inherent contradictions can be revealed and acted upon. The next section addresses this issue.

5.3 – software work

Workers of all kinds find themselves in ever more 'precarious' conditions. In the (overdeveloped) knowledge economy, the distinction between 'cognitive' and 'precarious' work has collapsed into what chainworkers.org call 'precogs'.[19] The term is their attempt to characterise the combination of the intellectual labour of the 'brainworkers' and the manual labour of the 'chainworkers'. To Lazzarato, the significance of this cuts across traditional class distinctions, and undermines the old distinction between material and immaterial labour introduced in the previous section. This also clarifies that immaterial labour has not replaced material labour but added to it. There is often confusion in this connection, as the term 'immaterial' does not mean insignificant or not material. Further qualification is required to emphasise that material goods are still being produced on a massive scale in parts of the world, and that the labour involved in producing immaterial goods is material in itself (Wright 2005).[20] But labour has also expanded to involve cultural activities not traditionally considered in terms of work, including creative labour. Creative labour in this way stands for the combination of information worker and artist, or what in the context of this thesis would be the artist-programmer or software artist, whether working on a paid or voluntary basis.

These developments present new contradictions over work, characterised not least by the prevalence of free labour in the cultural realm (something familiar to arts practice in general). Indeed, the once straightforward distinction between paid and unpaid work or non-work is also harder to differentiate. To Terranova, the complexity of labour in the digital economy is characterised by free labour in the production of free and open source software (2000: 33). She is skeptical of the link between free labour and the 'gift economy', to explain how gifts of time and ideas might indeed overthrow capital from within (2000: 36). She is referring to Richard Barbrook's reference to the 'high-tech gift economy' (1999), which is in itself a reference to Marcel Mauss's anthropological examination of 'gift economies' and systems of exchange, that lie outside capitalism (1970).[21] The rejection of the so-called free market for the 'commons' is characterised by Barbrook as 'anarcho-communism', but unlike Mauss's studies of societies outside capitalism, the high-tech gift economy is entwined in a complex relation to the capitalist market. What is considered 'free' is clearly based upon an infrastructure that is thoroughly commodified. The participatory ethic that Barbrook sees as shaping radical politics today in DIY culture would be understood by Lazzarato as an imposition of new forms of control and command over subjectivity, and like general intellect, thoroughly contradictory (1999: 224). Social transformation, once thought to be tied to working class agency, now appears to be based on the more complex connection between the production of new machinic subjectivities and the recomposition of workers as a class linked to general intellect.

The issue remains of how to organise a society in which producers-consumers give and receive, to the satisfaction of mutual interests that are not entirely based on individual reward.[22] This is a moral issue for Mauss, of how to shift emphasis from individualised benefit to social benefit. What he discovers is that because the 'producer-exchanger' is giving something of him/herself, he/she wants 'recompense, however modest, for this gift' (1970: 75). The recompense relies on a system of exchange not exclusively applied to goods and wealth, property and things of economic value, but other non-economic value. A cultural producer invests capital in the form of knowledge and skill, into a project with other rewards in mind such as peer recognition, what Bourdieu refers to as 'cultural capital'. Cultural

capital is heavily institutionalised in various fields, such as the convention of academics giving papers at conferences and sharing knowledge for free (or even paying to give papers). The production of free software operates in similar ways but tends towards an emphasis on social not individual creativity. Drawing on these ideas and associating them with general intellect, Terranova explains free software not as an alternative to capitalism but as an expression of new forms of labour, that have:
'... developed in relation to the expansion of the cultural industries and are part of the process of economic experimentation with the creation of monetary value out of knowledge/culture/affect' (2000: 38).

In this scenario, the information worker is often conflated with the artist worker to characterise 'creative labour'. To Marina Vishmidt this conflation raises issues, not merely over the generality of the term 'immaterial labour' but also the 'dogma of art or creativity' (2005: 94). The final chapter will deal with this connection to software art in more detail, but first in what remains of this chapter a more general approach to software work will be introduced. The parallel between software and work is encapsulated in the way the personal computer has become like a personal factory, in which established social relations remain unchallenged or become even further entrenched. But is simply refusing to use certain software or hacking technologies effective refusal? The suggestion is that many oppositional strategies are not transformative, but merely oppositional.

The chapter ends by arguing for the continued relevance of dialectics to respond to the contradictions in software work. Whereas the positions introduced in the first two sections point to the inadequacy of dialectics to deal with the crucial concept of immanence, the dialectical approach introduced in the previous chapter attempts to deal with this issue by integrating ideas associated with complex systems. In this way, contradiction can be reinstated.

# software for work

Microsoft remain the symbolic target for criticism in terms of software work. Naomi Klein claims Microsoft ‘wrote the operating manual’ for ‘engineering the perfect employee-less corporation’ through the extensive use of independent contractors, use of freelancers and outsourcing as a ‘disposable labour force’ (2001: 249). But clearly Microsoft is symptomatic of a more widespread logic that applies to employment practices in general. Preferred patterns of work, including critical work, are made explicit in the availability and prevalence of Microsoft products in workplaces and universities across the world. Much commercial software appears to be designed to predetermine its use and deny the user autonomy over their work. The user or worker simply becomes one of the objects of a proprietary operating system, that permits little deviance from the prescriptive tasks the system allows. In ‘The Macintosh Computer: Archetypal Capitalist Machine?’, William Bowles argues that control is not only enhanced by the development of new technologies but also expressed through the technologies themselves. Writing in 1987, he regards the ‘user-friendly’ graphical interface of the computer as a further development of the industrial period:

‘... where craft skills were stolen and locked into the industrial machine, then perfected to the point whereby general principles could be extracted and applied to ever more sophisticated machines, each in turn, requiring less and less skill [and labour] to operate!’ (2005: 50).

In the case of Apple Macintosh,[23] what they tried to do with their operating system was to make a ‘universal’ graphic user interface, to set a standardised way of operating a computer that enabled the relatively ‘unskilled’ user to gain access to computers. Yet specialised expertise is also required to maintain the ‘inevitable inaccessibility of the machine itself’ at a deeper level. The separation serves to reinforce the split between technical operations and wider cultural work. The average user interacts with the operating system via a command structure, using a toolbox, and so on, that parallels the kinds of standards developed in machine tools. It may be easy to use but it is made impossible to use it at a greater level of operation (until recently at least, with Apple’s adoption of a Unix-based operating system). It is a closed system that ‘mystifies’ the processes involved and the choices open to the user. The operating system ‘”masks” the “real” operation of the computer by interposing itself between the user and the Central Processing Unit', thus the Macintosh computer presents itself as a 'black box', denying access to its complexities (2005: 43).[24] This expresses dictatorial control, according to Bowles.

The desktop metaphor of the graphical user interface positions the user as an office worker. In the broader context and ubiquity of Microsoft Office, Fuller explains the 'disappearance of the worker is best achieved by the direct subsumption of all their potentiality within the apparatus of work' (2003: 139). In the essay 'It Looks Like You're Writing a Letter', Fuller provides a close analysis of the word-processing software Microsoft Word, distributed as part of the Microsoft Office package. Microsoft Word prescribes and universalises work and leisure activities as if the user is designed as part of the package, or even more so, the user's labour and subjectivity are made to disappear into it - installed into the system, as Fuller puts it (2003: 148). Lazzarato's assertion (referred to earlier in the chapter) that facilitation involves the willing subjectivity of the worker, or user in the process of production, is a further explanation of the effectiveness and totalitarian tendency.

In word-processing a text with Word, the writer becomes part of the machine, thoroughly embedded in the choice of computer and software program. This is one reason why Microsoft Word is not used to write this thesis, which also relates to Friedrich Kittler's apology for the software he used to write the essay 'There is no Software' (1996). Bowles makes an issue of this too, in declaring his tools: a Macintosh computer and Macwrite word processing software (2005: 49 & 44). TextEdit, used to write these words, is a far simpler and less prescriptive program, but clearly the same issues apply.

In the case of 'Word', Fuller's suggestion is to 'cut the word up, open, and into process' (2003: 163) which is exactly what he did for the installation _A Song for Operations_, at the Lux gallery, London in 2000. Accompanying the exhibition, the essay lays bare 'Word', revealing it to be intentionally over-complicated but packaged under the mask of user-friendliness. There is an excess of programmed functions that serve to de-skill the user, such as the various forms of 'help' available, exemplified (at the time of Fuller's writing) by the cartoon Office assistant equipped with limited 'artificial intelligence' to confront the user's assumed 'stupidity' and suggest 'correct' use of language. There are also a range of stupid templates available, such as 'CV Wizard', 'Envelope Wizard' and 'Letter Wizard' (hence the title of his essay), but alas no 'Suicide Note Wizard' (2003: 148) – something later attended to by Goriunova in 'Suicide Letter Wizard for Microsoft Word' (2002), combining work and death (as opposed to art and life).[25]

The underlying grammar of the software emphasises a set of standard tasks to be completed. The preferences of the program are particularly evident in 'autocorrect' with its automated spelling and grammatical corrections reflecting the dominance of correct English language as the globalised language of business. As a result, in the overall context of 'Office' (the software and the workplace): 'digital writing is not simply subsumed within an uninterrupted envelope for accessing various medial formations, but articulated, variegated, and positioned by the [...] culture of doing business' (2003: 150). It is proprietary software in the fullest sense. Back home, it is likely the user is using the same operating system and software that they use at work, either working at home literally or unwittingly at leisure.

# hacking work

Viable alternatives such as open source software can be identified but are they transformative? Fuller makes this point by asking whether 'free software is too content with simply "reverse-engineering" proprietary software' (2003: 162). For instance, OpenOffice's copying of Microsoft Office feature-by-feature and opening up its source code does not represent freedom as such.[26] Its claim to be a free office suite rather misses the point and reinforces much of the same logic. Worse still, it could simply represent the tendency to position free labour within the knowledge economy (as Terranova expressed earlier). Both work and software should be open to more radical transformations.

There are numerous examples of the ways users adapt and use what would otherwise appear to be prescriptive consumer technologies – what Michel de Certeau calls 'tactical' forms and 'makeshift creativity', to assert that users oppose established rules in

the most ordinary of circumstances (1984: xxiv).
[27] According to Mirko Schäfer, software products are particularly prone to adaptation and further innovation by users with technical competence, and by the use of network communications to share ideas under open source principles. His examples are turning the Microsoft Xbox into a Linux web server, a Nintendo Gameboy into a music editor, and enhancing the Sony AIBO robot pet dog with feral behaviour (2004: 63). Are these positive examples of 'general intellect'? Certainly modification or hacking of existing software and hardware demonstrates the creative and collective desire to adapt prescribed uses of technological goods, and positions the consumer as producer too. Yet these examples also serve to demonstrate how fast and adaptive companies are in recuperating these innovations. The unofficial development is recognised by manufacturers as free labour and research. Moreover, especially with an activity like hacking, the issue remains whether the activity is locked into resistance mode only, and does not engage sufficiently with the ways that capital endlessly restructures itself as an adaptive system.

The autonomists refer to this restructuring aspect as the 'cycle of struggle'. The term stresses a crucial issue: that resistance needs to transform itself in parallel to this recuperative process. This is what Mario Tronti calls a spiralling 'double helix' in which the restructuring of capital and the recomposition of the proletariat chase each others tails (in Dyer-Witheford 1999: 68). It is in recognition of this adaptive behaviour, that more tactical and strategic alternatives need to be developed. Bey's concept of the 'Temporary Autonomous Zone' is one example of this, in response to the observation that: 'Even the guerrilla Situationist tactics of street theatre are perhaps too well known and expected now.' (2003: 5) Adapting the tradition of 'independent media', 'tactical media' is another strategy for making 'temporary hybrids of old school political data and the aesthetics of new media'; for instance, producing anti-aesthetic software and other 'hackivist' strategies (Lovink 2002: 262).[28]

Geert Lovink and Florian Schneider explain that when no other choice is possible, 'sabotage can be seen as a sort of anticipated reverse engineering of the open source idea' (2001). For example, 'Floodnet' software was developed in 1998 by the Electronic Disturbance Theater, allowing for ‘virtual sit-ins’ (or online civil acts of disobedience) in the spirit of direct action, and offered as a tool to enable protestors to effectively shut down web servers of target institutions, by flooding them with requests.[29] In this example, the tactics associated with the refusal of labour in the material world are adapted to an understanding of immaterial and communicative labour.

The intersections of activism and the alternative use of computer technologies are bound together in the term ‘hacktivism’. As the previous section suggested, many of the alternatives to proprietary software do not attack the issue of property at source. This is the central argument of McKenzie Wark’s _The Hacker Manifesto_ (2004 [2001]) to maintain a central emphasis on property. Wark argues that post-Marxist critique does not address this issue sufficiently, and the ways in which informational technologies have influenced the concept. His starting point is a Marxist position that class relations derive from the privatisation of the property relation, firstly through land and subsequently through industrial capital. Additionally, Wark claims intellectual property to be a third, distinct form of private property, which gives rise to a third, distinct class antagonism. He explains:
‘Just as the development of land as a productive resource creates the historical advances for its abstraction in the form of capital, so too does the development of capital provide the historical advances for the further abstraction of information, in the form of “intellectual property”.’ (2004: 018)

Property rights have now been extended from land to capital to information. In Wark’s materialist formulation, class division is similarly regenerated, and it is the class associated with information as property and who reject its privatisation and commodification, who are the agents of social change. It is this ‘hacker class’ that hold the potential to exert a political agenda over property. Effectively, he describes a class war between those keen to privatise property (the legal hacks of patent and copyright by drug and media companies in particular) and those whose practice is involved in making property public (the hacks of file sharing and pirating activities on a popular level). Wark claims: ‘Information wants to be free but everywhere is in chains’, playfully combining, or hacking, both _The Communist Manifesto_ and the hacker slogan (2004: 126).[30] According to his logic, information like other goods is owned and controlled by class interests, and the hacker is in a privileged position (like the proletariat) to overturn these relations. The hacker is able to disrupt the commodifying impulse of the legal apparatus that wants to turn information into property (described in the previous section in relation to the focus on alternative licenses).

It is a convincing position, but there are difficulties (not least with the ambiguity of the term hacking itself). Certainly, there is no guarantee of a preferred ideological position in relation to hacking, despite its undoubted potential for forcing together new understandings from existing materials in the ways that montage might have done previously. Amy Alexander makes a similar point in stressing the ambiguity of the term and the apolitical motivation of much activity in this area. Although hacking generally describes an activity like crudely hacking a piece of wood with an axe, the application to computing is rather more subtle but still a general procedure of taking something apart, such as code. In general usage, the term refers to the illegal act of breaking into a computer, but Alexander explains the confusion between ‘hacking’ and ‘cracking’: the hacker as someone with proficiency and practical understanding of the structure and operations of computer networks and systems, but the cracker with more destructive tendencies: ‘some hackers crack, many hackers believe in exploratory cracking but not destructive cracking’ (2004).

For the purposes of this thesis, the importance is that the hacker expresses something of the sense of autonomy that the autonomists have described as lacking. This emphasis on the centrality of autonomous and creative labour is also characterised by Barbrook and Pit Schultz in their ‘Digital Artisans Manifesto’ (1997). It is argued that this transformation can come about by rejecting neo-liberal work patterns of the free market, the ‘californian ideology’ and formation of a ‘virtual class’. Instead they propose the ‘digital artisan’ in which autonomous work is made possible in the manner of past craft workers (1997). Yet both these formulations tend towards the assumption that ‘artistic labour is productive labour’ (to cite Tarde, in Lazzarato 1999: 165). Artistic labour may offer some critical potential in resisting commodification, but this is its potential more than what happens in practice. For the most part, the active productive human agent has been reduced to inert, irrelevant and useless labour-power, not least in the field of arts, to which it increasingly refers.

There are other difficulties with the figure of the artist-programmer (or programmer as artisan in this connection), where the processes of programming have correspondingly become closed off, mystified, based on elitist knowledge and hence contribute to the return of a romanticised myth of creative genius, embodied in the hacker class. These positions seem to represent oversimplified (or mannered) versions of class antagonism, that do not sufficiently incorporate complex formulations of machinic agency, or the misery of precarious working conditions both in the overdeveloped world, but more particularly in the underdeveloped or developing world.[31]

# refusing work

Conceptual problems remain between the reconstitution of the proletariat and the idea of a ‘stalled dialectics’, in which the proletariat no longer can be seen to be the _privileged_ agents of social change (Terranova 2002). To simply replace one privileged class with another, such as the hacker class, appears to miss the point. Perversely, as has been introduced earlier in the chapter, Negri looks to Marx to overcome the conceptual problems associated with revolutionary praxis. Referring to the _Grundrisse_, he argues that Marx uses both a dialectical and a non-dialectical logic suited to the development of the working class, from dominated labour power to a revolutionary class (1991: 150). Negri describes a more open description of the dialectical method:
‘Thus there is no linear continuity, but only a plurality of points of view which are endlessly solicited at each determinant moment of the antagonism, at each leap in the presentation, in the rhythm of the investigation, always looking for new presentations. [...] Each research result, in the presentation, attempts to characterize the content of the antagonism

and to see it, tendentially, in its own dynamism; when this dynamism takes off, we observe a veritable conceptual explosion.' (1991: 13)

The problem Negri has with dialectics is bound up with Hegelian resolution and its implied attack of Spinoza's sense of 'immanence', which remains to the autonomists a revolutionary theory.[32] Immanence in this sense, represents an emergent force with potential to resist power. However a dialectical approach can retain an incomplete synthesis (referred to in chapter 3, and argued by Žižek). Furthermore, Benjamin's approach to dialectics provides an intervention, as he tries to 'halt the flow of the movement, to grasp each becoming as being' – as opposed to the more classical Marxist approach where all social forms remain 'in fluid movement' (Tiedemann 1999: 943). Benjamin's concept of 'dialectics at a standstill' seems to draw together dialectics and immanence:
'For while the relation of the present to the past is a purely temporal, continuous one, the relation of what-has-been to the now is dialectical: is not progression but image, suddenly emergent. (1999a: 462)

It is Benjamin's idea of dialectics at a standstill that enables the rupture of the historical continuum and the possibility of transformation. Klee's angel of history is a dialectical image to Benjamin, and an example of the way: 'Images became dialectical for this philosophy because of the historical index of every single image', and the 'standstill' thus rescues the image from the conservative historical continuum – indeed, 'blasts' it out of the continuum (Tiedemann 1999: 944).

Rather than regard dialectics itself at a standstill, recent conceptual formulations of power continue to evoke dialectical conjunctions. For instance in _Empire_, contemporary power is described as: 'characterised by a fluidity of form – an ebb and flow of formation and deformation, generation and degeneration' (2000: 202). An argument can be made that the cycle of struggle needs to articulate itself in terms of generative processes, in lieu of the regenerative mechanisms built into capitalism itself. In another formulation, the term 'corruption' is used to refer to a perpetual becoming, that is the negation of generation. Hardt and Negri say capitalism is by definition a system of corruption and the task is to investigate 'how corruption can be forced to cede its control to generation' (2000: 392).[33] Thus, corruption might be thought of in terms of 'de-generation – a reverse process of generation and composition, a moment of metamorphosis that potentially frees spaces for change' (2000: 201). The dynamic of generation and corruption is especially evocative of dialectics, as it lends itself to the potential of destructive software to release further transformations. Hardt and Negri describe the structure of this, 'like a software program that carries a virus along with it, so that it is continually modulating and corrupting the institutional forms around it.' (2000: 197-8)

Software produced by The Museum of Ordure makes reference to ideas of corruption. For instance, _Dust_ (2000) is both a representation and a process of detritus that slowly 'corrupts', pixel by pixel. The corruption is triggered by viewing the image and in doing so, a pixel moves from one location to another. When no viewer is present (indicated by the sound falling below an ambient level), the pixels are rearranged back into their original order. The data is consistent, the pixels merely rearranged. [34] Its dynamic form is consistent with the logic that any characterisation of power as chaotic must be countered with something equally complex. Alain Joxe responds to this issue, in _Empire of Disorder_, asking how resistance can be characterised to lead to a more pleasant chaos (2002: 107). His point is that traditional standpoints of resistance seem powerless to resist power, because now power is more complex and has taken the form of resistance itself. Order is now expressed through disorder, in other words. This asymmetry between order and disorder is partly as a result of the 'decomposition' of bipolar cold war oppositions, replaced with the disorder of the free market. As a result, what is required is a response that draws on systems and complexity theory. This is somewhat verified by Joxe's statement that: 'Disorder is only a new beginning because it potentially contains a variety of possible orders, a variety of scales of possible orders. Disorder always opens a new choice of degrees of order.' (2002: 121)

In Negri's terms, examples of reversal have not gone far enough in transforming labour (and this is why he considers socialism to be a repressed alternative to capitalism [35]). For instance, labour time is more difficult to measure and is less distinct from time outside work, much of it now practised as ‘nonwork’, outside of traditional production processes – ‘notworking’ as opposed to networking. These tendencies can partly be recognised in relation to the computer, in the way it has redefined social practices and relations. Labouring practices follow this networked pattern in which physical labour, intellectual labour and machine labour are ever more undifferentiated.[36] It is work itself that needs to be transformed and made more autonomous according to Negri, not by the reappropriation of work but by the refusal to work. This position of refusal derives from Tronti’s essay ‘The Strategy of Refusal’ of 1965 (1980), following the logic that capital ‘seeks to use the worker’s antagonistic will-to-struggle as a motor for its own development’ (in Virno 2004: 11). Refusal is seen as an affirmation of the worker’s creative capacities, outside of capitalist relations of production.[37] The creative power to use technology differently, to reappropriate it, still rests with those who have the expertise to operate it. Systems operators, programmers, computer scientists, technicians, software engineers, designers, computer-literate office workers, and software artists clearly hold the potential to use and abuse this invention power. This is what Negri calls ‘”invention power” – the creative capacity on which capital depends for its incessant innovation’ (Dyer-Witheford 1999: 71).

The strategy of refusal represents not a liberation _of_ work, but _from_ work. In terms of negation of negation, what is required is to first negate capitalist exploitation and then negate the conditions for work. The more complex issue is how a refusal to work can be extended to encapsulate general intellect. It is clear that any interpretation of Marx must be adapted to the times and the restructuring of power. Much of the content of this chapter is recalled in this statement: such as Deleuze and Guattari’s call for machinic agency, and Artaud’s description of disorder, demonstrating the potential for transformation. Although Negri argues against the dialectic, there appears to be an agreement that old forms and new forms of protest do need to be brought together – for instance, both hacking and

sabotage. The activity of hacking, if seen alongside other antagonist strategies that take things apart, offers another form of praxis, suitable for contexts that involve software. In conversation with Negri, this is what Deleuze anticipated:
‘Computer piracy and viruses, for example, will replace strikes and what the nineteenth century called “sabotage” (“clogging” the machinery).’ (1990)[38]

New collective characterisations are required that respond to general intellect, according to Matteo Pasquinelli. For instance, he remains suspicious of the rhetoric around free software. His concern is how the discussion around open source and free software relates to action in the real world. As a result, Pasquinelli asks:
‘How can we turn the sharing of knowledge, tools and spaces into new radical revolutionary productive machines, beyond the inflated Free Software? This is the challenge that once upon the time was called reappropriation of the means of production. [...] How do we start building these machines?’ (2005: 4)

Can we begin to imagine radical machines of disorder? If software is considered as something produced as the result of work and something that does work, the refusal to work might be extended to the reinvention of software that is dysfunctional, that refuses to function through an intricate knowledge of its inner workings. Software such as this would deny its potential, and represent not a transformation of work but from work, in the sense that Negri suggests with the paradoxical quote at the beginning of this chapter. The further issue that this chapter raises is whether software can be liberated from software work. This is the challenge for software art-work.

====================
6. *software praxis*
====================

‘A successful work of art, according to immanent criticism, is not one that resolves objective contradictions in a spurious harmony, but one that expresses the idea of harmony negatively by embodying the contradictions, pure and uncompromised, in its innermost structure.’ (Adorno, in Jay 1996: 179 [from _Prisms_, 1967: 32])

Software both works and is worked upon. It follows that software art holds the potential to make apparent contradictions associated with the relations of production, described in the previous chapter. For instance, software can be programmed to act disruptively, such as by the refusal to work (non-executable code) or by working in a negative mode or disorderly manner (dirty code), and outside the orthodoxy of passive work (analogous to proprietary models of clean and pure code). These possibilities make it clear that software is not simply a functional tool but expresses wider cultural processes and transformative possibilities.

This final chapter returns to the central context of this thesis in arguing for a critical practice in software art that breaks the art historical continuum (what Benjamin refers to as 'dialectics at a standstill'). Section 6.1 examines the deployment of software in an artistic context, by concentrating on the work involved in writing code and the work that the code then actualises when it is executed. It argues for practices that encourage the reader to become a writer, to become engaged in the production process. In the spirit of past critical practices, it suggests that software practices can be developed that do not merely reveal the inherent contradictions but the ways in which the apparatus itself is subject to 'functional transformation'. The importance of this, as the quote by Adorno suggests, is that contradictions are not resolved but embodied.

This way of working rejects determinism associated with software, for something far more speculative and 'artistic'. Section 6.2 describes this performative dimension in more detail and relates it to the distinction between labour, work and action. Praxis, as action derived from theory, responds to these coordinates of what exists and the future possibility of its transformation. This description of praxis is derived from dialectical thinking, which undermines any undialectical opposition of theory and practice, and stresses the combination of creative and critical activity embodied in human action. These are words of warning for the traditional form of a PhD as the embodiment of privileged knowledge, and helps to stress the merits of practice-based PhDs (as in the case of this thesis when first written). Theory is made meaningful and tested by practice, thus resists any simplistic separation of the two concepts. Furthermore, the phrase software praxis takes account of complex systems, in which disorder can be seen to generate a transformation of the system. In this way, this thesis suggests that the operations of the programmer and program, taken together, relate to transformative action.

A focus on coding practices, code, and the execution of code represents the privileging of potential, expressing ‘immanence’ as something that remains within and is ready to come into being. The final section 6.3, as the term ‘coda’ indicates, contains some concluding remarks. It takes the form of a series of qualifying statements that suggest the key issues at stake but without resolving them into a false sense of finality, in the spirit of the dialectical approach it proposes. Taken together, they embody what is reserred to as software praxis.

6.1 - software art work

Software is a set of formal or logical instructions written in code. Computers execute these instructions but an emphasis on the program and the instructions demonstrates that these are written, that they are programmed through human intervention. In _The Art of Programming_ (1981), Knuth suggests the analogy between programming and recipes in a cookbook (1981), as a set of instructions that are to be followed. To look at the source code even of a meal reveals more information on the dish to be prepared, and whether this is likely to satisfy. The analogy is rather straightforward but reveals something of the vested interests involved in preparation, execution and consumption of the work. In relation to free software, Stallman has similarly argued the sharing of software is as old as computing, just as the sharing of recipes is as old as cooking (1998). The metaphor is also used by the Belgian artist group Constant in their ‘cuisine interne keuken’ in which they examine issues around cultural and precarious work. They explain ‘we mean that a work, an organisation is made of: the components (ingredients), the tools (utensils), workplace, and work and creation processes (recipes)’ (2003: 61).[1]

A discerning consumer should engage with the preparation and ingredients as much as with the end product – and investigate what is going on in the kitchen or farms, at the site of production.[2] With production in mind, Marx once remarked that you cannot tell from the mere taste of wheat who grew it (in Deleuze & Guattari 1990: 24). The significance of this is that the end product gives little impression of the history of its process of manufacture, and no indication of the relations of production that were involved. In a similar way, Benjamin quotes Bertolt Brecht to demonstrate the deceptive representation of the site of production: 'A photograph of the Krupp works or the A.E.G. tells us next to nothing about these institutions. Actual reality has slipped into the functional. The reification of human relations – the factory, say – means that they are no longer explicit. So something must in fact be built up, something artificial, posed.' (1992a: 255).

The surface appearance of the computer is particularly unrevealing, and like the factory, expresses little of the complexity of its inward operations (like Bolognini's installations of sealed computers mentioned in chapter 2). Even the interface of the operating system and other software that run on the machine hide their workings (as the previous chapter described). The trick of the software capitalist, then, is to hide the content (or labour) under a deceptive form (like von Kempelen s chess playing automaton mentioned in chapter 3), rather than to reveal the contradictions of value and hence divisions of labour involved. This section of the chapter outlines these concerns in as far as they offer a political dimension to working with code, by stressing the work involved in coding. What Alan Sondheim calls 'codework' explains this as follows: 'Every more or less traditional text is codework with invisible residue; every computer harbours the machinic, the ideology of capital in the construction of its components, the oppression of underdevelopment in its reliance on cheap labor.' (2004)

This is something that Leonardo Solaas's _Outsource me!_ (2005), makes explicit. Solaas reverses the usual outsourcing of programming work by seeking proposals for him to program, and in so doing confuses the usual power relations of a Western agency (even artist) using cheap labour and expertise from the developing world (Goriunova 2005). The work reflects the current conditions of much software development that is outsourced to software houses in India or the Caribbean (Mackenzie 2005: 71). In _Outsource me_, the programmer voluntarily provides cheap labour by seeking proposals to make software (commissioned by _Readme 100 Software Art Factory_), in an ironic twist where the site of technical production becomes a conceptual artwork, that addresses the precarious labour relations of outsourcing.

A critical appreciation of software development requires this simultaneous understanding of the production of its source code and execution, to elicit fuller technical and political detail. This has relevance for software arts practice too, in stressing all aspects of the work involved in making artwork to understand the contradictions that arise from production. It is quite common practice for examples of software art to hide the complex interactions of processes and code running on the computer behind the scenes, as well as the working processes of both programming and of programmers. Like most commercial software the source code that the programmer works with, remains 'closed-off' and inaccessible to the experience of the user. For the most part, the software compiles the code into an executable version that 'locks-down' the source to protect proprietary interests, including intellectual and artistic capital.[3] It is worth stating the obvious, as David-Olivier Lartigaud does, that if a work is not open source, how can anything other than its execution be appreciated (2004)? Certainly the argument can be made that an aesthetic appreciation of code requires an appreciation of its written form _and/or_ what it does when executed - analogous to poetry that takes both written and spoken forms. The essay 'The Aesthetics of Generative Code' (Cox, McLean & Ward 2001) drew an analogy between poetry and code, and argued that as appreciation of poetry may come from reading or experiencing a live spoken performance, code's aesthetic value lies both in its written form and its execution.

That the source code might be considered an integral part of the artwork, or even the artwork itself, remains outside the imagination or will of the software/art market, obsessed for the most part with property rights. This is where software art offers an alternative view, and is able to reveal contradictions over production, peeling away the layers of operation and the relations of production involved in working with code, and in code working. An example is McLean's _animal.pl_ (2003), software that ran continuously on a server connected to the Internet until a server crash ended its 'life' (a death notice was issued). The Perl script performed many activities, such as publicising its existence to various mail lists and making suggestions as to its life-like qualities. Although relatively autonomous, _animal.pl_ required a connection to the Internet to operate, and its actions were prescribed by the programmer. The last prescribed action _animal.pl_ performed was to apply the GNU Public License to itself, offering itself for others to modify and adapt in new ways.

Such an example goes against the grain of much software practice that displays an over-concentration on visual aspects, as is the case with arts practice in general, that has contributed to the neglect of more dynamic and complex processes. For the most part, artists collude with the 'software culture industry' on this issue, leaving other potentially creative realms relatively unexplored. In 'There is No Software', Kittler argues that hardware is obscured by software, and as a result confusion arises between the use of 'formal and everyday languages'. He claims: 'We simply do not know what our writing does' (1996: 332). By this, he is referring to the ways that graphical interfaces dispense with the need for writing and hide the 'machine' from its users. For Kittler, this is implemented at the level of hardware itself, and software does not exist as a machine-independent faculty (1996: 334).

Although the argument follows the concerns of the previous chapter and the way that subjectivity is embedded, clearly software can be hardware-independent. Recent practices in software art, such as Socialfiction.org's _.walk_ (2003) that does not require a computer, confirm this.[4] In this example, the decision-making processes normally assigned to machines is interpreted by the public and executed in the streets. It invites active participation in the execution of algorithms for walking, following the principles of 'generative psychogeography'. The implementation of the algorithms is thus decidedly unreliable, allowing for unpredictable and chance encounters. Levels of complexity and conceptual ingenuity are evoked, not so much by the programmer, but more so by the execution of the program by people – unlike the 'closed' operations of much contemporary software. The idea is endlessly re-write-able in its collective distributed form and as a collaborative developmental model of practice in the open source tradition. Furthermore, it is an example of software art that does not require a computer at all – the simple technology of a pen and paper would suffice. To refer back to Kittler's statement, what is also required is a person, or executant, who knows precisely what their writing does, and can act upon it.

The approach to programming is consistent with Benjamin's description of what he called an 'operative' writer, who reflects upon his/her position within the production process like a technician (in _The Author as Producer_, 1934). He refers to this as: 'Work itself puts in a word. And writing about work makes up part of the skill necessary to perform it.' (1992b: 90) His example of this is the Russian writer Sergei Tretyakov, who as a journalist (or 'hack') demonstrates a working practice outside the established canon of literary forms. What Benjamin has in mind is the way that popular forms, such as the newspaper, might challenge established separations: of academic and popular modes, of descriptive and creative writing, of individual and collective property, and between writer and reader: 'For as literature gains in breadth what it loses in depth, so the distinction between author and public, which the bourgeois press maintains by artificial means, is beginning to disappear in the Soviet press. The reader is always prepared to become a writer, in the sense of being one who describes or prescribes. As an expert – not in any particular trade, perhaps, but anyway an expert on the subject of the job he happens to be in – he[sic] gains access to authorship. Work itself puts in a word. And writing about work makes up part of the skill necessary to perform it. Authority to write is no longer founded in a specialist training but in a polytechnical one, and so becomes common property.' (1992b: 90)

This distinction between 'the passive consumer of the readable (lisable) classic realist text and the active producer of meaning who accepts the challenge of the writable (scriptable) text' (Belsey 1992: 125) refers back to the work of Barthes (in 'The Death of the Author’, 1977) - but in Benjamin the politics are foregrounded. In the context of software art, Cramer too draws upon Barthes’s _S/Z_ (1975) in making the same distinction between ‘readerly’ and ‘writerly’ texts, and applying this to operating systems (2003). Rather than the readerly properties of a GUI operating system that encourages consumption and hides the code, Cramer claims the command-line operating system of Unix is writerly, in terms of its openness and in encouraging the reader to become a producer of code. Arising from the open source movement and the social relations it engenders, Unix offers this potential to provide access to the hidden depths of code.[5] It is the ‘closest thing to a hardware-independent standard for writing truly portable software’ (Raymond 2004: 8). This contributes to what Cramer considers particularly significant, in breaking down the false distinction between the writing and the tool with which the writing is produced. He cites the 1998 essay by Thomas Scoville ‘The Elements of Style: UNIX as literature’ (2003: 102) to insist on the writerly aspects of programming (chapter 2 also provided many examples in this vein). Also emerging from Unix culture, the programming language Perl is eclectic in its combining and appropriating other languages, working against prescription: ‘It doesn’t try to tell the programmer how to program’ (Wall 1999). Perl holds multiple possibilities for transformation, and in using it programmers demonstrate the potential for good technique, in the sense that Benjamin describes in adapting the apparatus.

The potential for transformation of the apparatus is what Brecht calls the ‘functional transformation’ of the ‘forms and instruments of production’ (Benjamin 1992b: 93). This approach goes beyond an engagement with the apparatus or being satisfied with finished works, but seeks to transform the apparatus, because only in this way might the relations of production be transformed too. In the case of software production the position holds relevance, in that it is not enough to simply reveal source code, make it free or to stress its potential aesthetic form (as was argued in the previous chapter), but it also needs to be made available for further transformation. Brecht’s solution was to develop a new form of writing that foregrounded contradiction. In _Critical Practice_, Catherine Belsey additionally proposes a new form of critical writing developed along similar lines to examine the process of production of the text, and one that reveals the contradictions inherent in the form of its production (1992). This thesis follows these recommendations to argue for a critical practice that reveals contradictions related to the writing of software art.

A distinction needs to be emphasised in that artificial languages differ from so-called 'natural' languages, though clearly there is nothing natural about either of them, as they are both artificial but in different ways. Importantly, a program is not spoken as such and it is written for two very different readers: the computer that executes it and other programmers who may like to understand it and revise it (Bolter 1984: 127). These linguistic differences are also what Kittler has in mind when he points out that computer code is a very particular kind of language. He points to a key difference in that words of natural languages do not generally do what they say: 'No description of a machine sets the machine into motion' (1999). On the other hand, the artificial language of computer code generally does what it says - it executes and enacts its instructions or description. That computer code has both a legible state and an executable state, as Kittler puts it (1996), or contains both readable and writeable states at the level of language itself, is precisely the point for Cramer in that 'the score is not aesthetically detached from its performance' (2002a: 108). Once described in terms of performance (_.walk_, for example), some unpredictable elements are introduced, associated with human intervention and machinic agency. Contradictions between human and machine agency are central to Christophe Bruno's _Human Browser_ (2006). Using wireless headphones, a human actor hears a text-to-speech audio that comes directly from the Internet in real-time, and simply speaks the words.

The important issue is how writing text or code relates to action - a point stressed at the end of the previous chapter. The dynamic relation between code, and the actions that arise from it, are an indication of historical processes taking place. Whereas conventional software production suggests a particular kind of logic where execution is determined by the code, software arts practice offers the potential to introduce looser thinking, ambiguity and contradiction. It is the dialectical interplay between code and its execution that concerns this thesis and what this suggests in terms of the relation between past and future possibilities. Whatever strategy is decided upon is only ever part of an ongoing adaptive process, that is ‘never perfect, always in becoming, performative’, according to Lovink (2002: 264).[6] He describes this as ‘messy praxis’ (2002: 226) which is a good term for the advocated approach of this thesis. Similarly, the term software praxis continues to place emphasis on creative human action and the contradictions that arise from this.

6.2 - software action

Code is a notation of an internal structure that the computer is executing, expressing ideas, logic, and decisions that operate as an extension of the programmer’s intentions. Its written form is merely a computer-readable notation of logic, and is a representation of this process. Yet the written code is not entirely what the computer executes, as there are many levels of interpreting and compiling and linking taking place. Code is only really understandable within the context of its overall structure and the many processes that are running. In technical terms, the processor is obeying the instructions given to it and generating activity as part of a continuing performance. Many of the components are predetermined, but through the multiple interactions, combined with the dynamism and unpredictability of live action, the result is far from determined.[7] In the example _feedback.pl_ (2004) by McLean, a text editor is editing a piece of code that has the ability to modify itself when executed (see Cox at al 2004). These modifications happen directly to the code being edited in real-time, opening up the possibility for the code to fundamentally modify its own behaviour. Of course, this has major implications upon the act of programming and allows the programmer to edit code whilst it is being executed, and to respond to the live situation. The programmer is required to consider the code’s initial logic, as well as be able to follow the code’s logic after it has modified itself. The suggestion is that this is an example of software art that contains both theory (of its own agenda) and practice (of its own action).

In ‘On Code and Codework’, Sondheim clarifies the distinction made in the previous section between ‘declarative and performative’ codes (2005). His example of a declarative code is something like Morse code, where one thing is equivalent to another in a way that would be useful for encryption. When it runs it does what it says. In contrast, an example of a performative code is Perl. Sondheim explains how Perl codes procedure and thus works on a more semantic level of understanding. He draws upon Umberto Eco’s semiotics, in which the possibility of code is extended from rules to ‘a set of possible _behavioral responses_’ which places Perl in the realm of performance, according to Sondheim (2005). In the area of software arts practice, programmers make music in keeping with the expressive qualities of live performance, by using interpreted scripting languages (such as Perl) and coding in real-time using the command line interface, with the source code on public display as much as possible. For example, in the performances of slub (aka McLean and Ward), the intention is to open up what would otherwise seem to be determinate processes of how music is generated. Human intervention is foregrounded, and glitches become part of the creative output.[8] Any resulting sense of improvisation relies on a predictive understanding of complex processes or virtuosity, and an opening up to the transformative potential of code. Unlike a score that is followed but interpreted, a computer generally follows the instructions without interpretation. The intervention of the programmer allows for a less deterministic approach and an openness to other transformative possibilities, such as the possible and often unpredictable actions that result when a program runs, including mistakes. The program performs the music as much as the programmer, relaying instructions and acting upon them. But in the case of live coding performances, human agency is foregrounded.

An even more extreme example would be JODI’s recent live performance _Desktop Improvisations_ (2004), a reworking of their earlier work _My%Desktop_ (2002). They exploit the limited potential of supplied and prescriptive software in a formal performance setting with seated audience, using the obnoxious alert sounds supplied with a standard Macintosh operating system, using key commands to create mayhem with repetitive mouse clicking. In a sense, it operates like a ‘hack’ of both live coding and live music, that uses programming panache and improvisation as creative method. In this example by JODI, as with much of their work in general, a computer crash simply adds to the potential drama and an overall aesthetics of error. For instance, and as a response to the inevitable concessions of exhibiting at such a mainstream event as Documenta X, JODI simply produced a link that on clicking made the visitor's machine crash. In these examples, the performance of the programmer and program challenge the way an operating system interpellates the user, and subjects the relations to systematic abuse. It is this performative aspect that lies hidden behind the surface of the software that this section aims to stress, in terms of its potentiality for action. This issue is evident in Arns's 'Read_Me, Run_Me, Execute_Me' essay. The subtitle 'Software Art and its Discontents' (2004) suggests that the performative dimension lies repressed in relation to code (by making reference to Freud's 'Civilisation and its Discontents'). Using this analogy, a programming language such as Perl might offer therapeutic assistance in putting the programmer in touch with his/her, and indeed culture's, sublimated desires – that which is repressed under capitalism, as the previous chapter indicated. That freedom of speech relates to relatively unrepressed free software, may be one of the analogies that lead Arns to discuss the performative dimension of software, through its relation to speech act theory. She is making particular reference to John Langshaw Austin's _How To Do Things With Words_ (1962), to express: 'that language does not only have a descriptive, referential or constative function, but also possesses a performative dimension' (2004: 185).

The performative aspect of speech is evidently social and context-bound, broadly differentiated in linguistic studies as the distinction between syntactic and semantic realms – emphasising the performance (or 'parole') that is generated from the rules ('langue'). Software art is concerned with both, but places emphasis on the performative aspect. Using Ferdinand de Saussure's terms, software art is more concerned with 'parole' than 'langue' – more social and semantic concerns than structural or systemic ones. In semiotics, the abstract system (langue/competence) generates the concrete event (parole/performance). Arns sees speech as analogous to program code in that it says something and does something with consequences (2004: 186). Indeed, words determine actions and events, and there is something fundamentally performative in this.[9] An effective speech demonstrates the potential to incite action.

Also referring to Austin's _How To Do Things With Words_, Virno says: 'In the assertion "I speak," I _do_ something by _saying_ these words; moreover, I declare what it is that I do while I do it.' (2004: 90) Virno's interest in speech emanates from how work is now increasingly bound to speaking and the use of communications technologies. [10] A software program is particularly articulate in this sense, as it both says something and acts upon the instruction in an efficient way. It is this sense of action that software art might exploit, by challenging the expectations of the workplace.

The emphasis on action in itself makes a distinction from the term work or labour. This distinction is what Hannah Arendt identifies in her essay 'Labor, Work, Action' (from a lecture of 1964). She identifies how labour (poiesis) and action (praxis) tend to be under-acknowledged in relation to work (2000). Labour is where production and consumption are part of the same process, like activities needed to sustain life itself. Human labour is embedded in work only in as much as it is required to generate an income. Even in Marx's writings, she maintains, labour is tied too firmly to work at the expense of action.[11] Arendt's point is that in any of the differentiations that are attempted, action simply cannot be avoided. For instance, this is the case in her distinction between contemplation and action (what she refers to as 'vita contemplativa' and 'vita activa'), from which she concludes that active life simply cannot be avoided (2000: 167). She explains that rather than think that all action ends in contemplation or that contemplation leads to action, it is not possible to go through life without acting in it, whereas contemplation is optional. Put differently, unlike praxis, theory alone cannot transform society.

Drawing upon the earlier distinction, the work involved in making software involves a labouring component, even if it is offered for free as in free software. Also software works in itself, although this cannot be considered labour unless tied to the labour of the programmer. This would be an interesting line of inquiry to explore and a complex one, but the important issue here is how the work of art and software art undermine the established distinctions, and do not fit what Arendt describes as the 'means-end' chain (2000: 177). The work of art breaks out of this chain by not being 'useful' and thereby resisting commodification. Although the assumption might be made that software is generally useful, unlike the work of art, the work of software art is more ambiguous in this connection. In fact, much of software art is trying to break out of the commercial imperative to be useful, but also offers the potential to be useful in other directions, such as in the case of social or critical software, to use Fuller's categories (from chapter 2). That it evokes contradictions in this respect, is part of its attraction for critical practice.

For Arendt, human action or praxis lies in this realm of uncertainty, as something that cannot be fully known but that is crucially bound up with the principle of freedom.[12] Making reference to Arendt's essay forty years later, Virno confirms that the once unquestionable separation of labour (or poiesis), action (or praxis) and intellect has since dissolved.[13] Whereas Arendt argues that politics imitates labour, he thinks the opposite, where labour imitates politics, or indeed, that poiesis has taken on the appearance of praxis (2004: 50–1). That labour takes on the form of political action, or more to the point has depoliticised action, explains what Virno refers to as the current 'crisis of politics, the sense of scorn surrounding political praxis today, the disrepute into which action has fallen' (2004: 51). He thinks that the purpose of any activity is increasingly found in the activity itself. Quoting Aristotle, Virno further explains the point: 'For while making has an end other than itself, action cannot; for good reason itself is its end.' (2004: 52)

The importance of action is stressed in this statement, in that it breaks the 'means-end' chain. Virno chooses to explore this idea through a discussion of 'virtuosity' by looking at the special attributes of the performing artist. Here again, he is drawing upon Arendt's observation that the performing arts have a strong affinity to politics. A performance is characterised by its lack of an end product, or at least a product that is indistinguishable from the performance itself (2004: 52). Furthermore, it operates in real-time and has its own sense of purpose or fulfilment, in parallel to the way that a computer program undermines the distinction between its function as a score and its performance (described in the previous section).

It would appear that many of the attributes associated with virtuosity could be applied to programmers. For example, a hacker is someone who performs a 'hack': 'To qualify as a hack, the feat must be imbued with innovation, style and technical virtuousity.' (Levy 1994: 23, in Wark 2004) The programmer is required to apply their technical and cultural agility. Referring back to Benjamin, this alliance between cultural and technical skill is necessary to 'transform him[/her], from a supplier of the production apparatus, into an engineer who sees his task in adapting that apparatus' (1992b: 102). He is making an important distinction between theory and activism, and that it is simply not enough to have political commitment in itself. This emphasises Benjamin's view that 'technical progress is, for the author as producer, the basis of his political progress [sic].' (1992b: 95). What Benjamin defines as a producer is applicable to the figure of the artist-programmer involved in the production of software. It further relates to performance through the example of Brecht, who according to Benjamin, 'opposes the dramatic laboratory to the finished work of art' (1992b: 100).

The Internet suggests itself as a potential 'dramatic laboratory'.[14] Both politics and the performance require a 'publicly organized space', as does labour under post-Fordism (Virno 2004: 55). Virno also links this sense of vituousity to speech, as a phenomenon that has purpose in itself, does not produce an end product independent of the act of speech, and operates in a publicly organised space (again, the link between free speech and free software as an ongoing performance of shared score is evoked). He continues: 'It is enough to say, for now, that contemporary production becomes "virtuosic" (and thus political) precisely because it includes within itself linguistic experience as such.' (2004: 56)

The etymological root of the word program emphasises the material production of code as something before the act. In Greek 'programma' is 'what is in advance written' - a set of instructions to be executed that are fixed beforehand. The artist-programmer Antoine Schmitt calls the program 'prepared' in this sense (2003). This is a useful intervention, as 'programming' can thereby be understood as a set of utterances describing a forthcoming action or a set of operations to be implemented in order to get a result, further evoking speech or performance. Art that is programmed holds a close connection with any action that is conceived in advance of its execution, and clues to this are to be found in the source code. The question for Virno is: 'what is the _score_ which the virtuosos-workers perform? What is the script of their linguistic-communicative _ performances_?' (2004: 63). Added to this: what is the source code? Following Virno, the score and the source code is 'general intellect', as the 'know-how on which social productivity relies', as an 'attribute of living labour' (2004: 64-5). This know-how refers to the ways in which workers learn skills but also the rules of social behaviour by which labour-power is reproduced (and that maintain class divisions). The issue is whether this know-how is to be used for social good or not, as suggested in the previous chapter. The script, score and source code is by no means determined and does not have an end product in sight. It is in contrast: 'virtuosity without a script, or rather, based on the premise of a script that coincides with pure and simple _dynamis_, with pure and simple potential' (2004: 66).

Potential is that which is not yet present. That action might operate without a script, as a way out of the means-end chain, is in marked contrast to Adorno's comments regarding music as a by-product of a score. Adorno's essay 'On the Fetish Character in Music and the Regression of Listening' (1991: 29-61) suggests that the score is the work of art and that the listener reassembles the score internally. He explains that:
'... the essential function of conformist performance is no longer the performance of the "pure" work but the presentation of the vulgarized one with a gesture which emphatically but impotently tries to hold the vulgarization at a distance. [...] Vulgarization and enchantment, hostile sisters, dwell together in the arrangements which have colonized large areas of music.' (1991: 36)

To Adorno, the score is partly a purer form, more closely associated with production that affirms use value, rather than the exchange value of the performance itself. In the former, the listener is encouraged to become a producer by executing the score, and in the latter, a consumer of the commodity form of music. In this sense, use-value is also reinstated over exchange-value. Related to this, the performative aspect of working without a score but working with source code to avoid the end-product is evident in live coding, as well as other practices that privilege source code.

The technical performance of the code object is characterised in this way by Adrian Mackenzie, in his essay 'The Performativity of Code' (2005). He is making reference to the Linux kernal both in terms of technical description of performance but also cultural ones, typified by Scott Lash's description of power as performative (2005: 6); expressed through information and communications networks. The performative element is that which goes beyond reference and description. For instance, Radioqualia's _Free Radio Linux_ (2001) is a performance in this sense. The source code of the Linux kernal (the core component of the GNU/Linux operating system) was webcast over the Internet, using a speech synthesizer to convert the 4,141,432 lines of code into talk radio. It was broadcast like other speech materials and presented as displaying aesthetic value (in a similar way to Linux's prize at Ars Electronica mentioned earlier). To Mackenzie, Linux is a performative 'speech act' that produces an uncertain relation between the code object (the Linux kernal) and the code subject (the programmers), and thus challenges its property relations and corporate relations of production (2005: 13) – demonstrating collective social action.

For Virno, this potential of utilising general intellect for political action is something necessary. He proposes two strategies of civil disobedience and 'exit' or defection in opposition to servility, both evoking disorder and the transformative potential of the script, score, coda – and indeed source code. In order to resist commodification, positive potential must remain without end product, remain in the public realm, and remain performative. A dialectical approach can accommodate this by its rejection of determinism, following an open-ended process that reflects the structures it aims to transform. This is the critical task for software art praxis, to remain in a continual state of becoming, where contradictions remain active.

6.3 - coda

That multiple layers of meaning are possible in every part of a text, presents difficulties for someone trying to assemble a linear argument and offer a conclusion. What has been said in support of this thesis has been said, but in the interests of clarity it will finish with three qualifying statements (corresponding to the arguments in chapters 3, 4 and 5).[15] These are not intended as definitive statements but ideas for further development, as part of an ongoing dialectical process.

thesis 1:
Software art demonstrates emergent potential

The argument for a dialectics of software art runs in parallel to the way in which source code is ready for action. A program, like any programme of action, is conceived in advance of its execution, and holds the potential to act even when not executed. Similarly, there may be a delay between what is known and what is acted upon, where practice leads to the development of theory (which in turn leads to the development of practice) and so on. The dialectical process generates what is already implicit, though not explicitly articulated. The critical task for software art lies in releasing transformative potential in this way.

By analogy, software expresses the dynamic and emergent action between what exists and what is possible. A historical materialist approach to software describes a process where construction and execution remain in dialectical tension. The process remains incomplete to avoid critical stagnation, and one in which an open model is maintained over attempts to close it down. This clearly applies to open source code, as it does to the impulse to act in the world. The link to emergence confirms the conceptualisation of change, as something that expresses an immanent dynamic - or ‘transformative praxis’ - both as a condition and consequence of human agency, that generates new possibilities of change to the system. In terms of the production of software art, the programmer and the program can exploit this potential.

thesis 2:
Through disorder, software art generates transformative action

Both dialectics and systems thinking share an interest in dynamic processes and interactions, emphasising that relatively small input can have massive consequences. They also confirm that systems are not closed but open to influence and change from external and internal factors, releasing the potential for transformative agency. In systems dialectics, negation is recast in terms of disorder, in such a way that new order can be seen to be generated through disorder, as an ongoing process of more extensive and penetrating inquiry. Further development is generated at the 'bifurcation point', or point of antagonism, causing an unforeseeable change to the existing system.

As a consequence of machinic agency, tactics associated with negation and the refusal to work are extended to the desire for machines to break down and become disorderly. The refusal to work follows in a critical tradition that rejects the logic of the system and order it is part of. It is an antagonistic strategy that affirms the potential creativity and virtuosity of the programmer and program, and self-determination or autonomy over work. In this way, software art-work rejects itself as work and affirms its potential for transformative action. The importance of this lies in the recognition of the relationship between action and counteraction in the development of systems in general. Software art can be disruptive of the normative contexts in which it operates, and offer alternative concepts and actions.

thesis 3:
Software art embodies inherent contradictions leading to software praxis

The contemporary description of power as an adaptive system does not reject but extends the emphasis on the mode of production as the site of antagonism. Whereas once labour represented the privileged site of struggle in dialectical thinking, it now takes on a more open character. The openness is in recognition of more and more complex and disorganised interactions between people and machines, and hence in the relations of production that arise from these interactions. Any critique of the labour involved in making art must therefore recognise the ways in which labour has become more immaterial, collective and communicative.

Clearly any interpretation must be adapted to the times and so too with the dialectical method. This thesis contends that dialectics needs to be adapted to take account of the reconceptualisation of work and the complex ways in which human and/or machine social relations are expressed. One of the major objections to dialectics has been its inadequacy to deal with 'immanence' as an emergent and radical force. On the contrary, dialectics continues to be a useful critical framework to describe systems that appear to contain the same logic: combining a technical description of a system and a suitable critical method for its analysis. A dialectical methodology engages with informational dynamics, whilst at the same time recognising that culture and criticism are themselves dynamic processes. A dialectics of software art requires a critical and a practical understanding of both art and software.

A critical arts practice sounds like a contradiction in terms, in these post-political times. The once radical potential of conceptual or performance arts practice reveals how even an arts practice that strives to reject commodification is in turn recuperated. Software art can be seen to demonstrate radical enquiry and speculation, but on condition that it continues to transform itself as part of an ongoing dialectical process of seeking more critical depth. An open view of dialectics, that takes into account complex systems, allows for an ongoing chain of contradiction which is inherent but not yet present. It is the assertion of this thesis that the critical strategy of contradiction needs to be retained at all times, and that software art practice can offer new critical forms by embodying contradictions in the interplay between code and action.

Contradiction is also embodied in the form this thesis takes, as a theoretical work which takes account of practice. Its writing, like code, lies at this intersection of code and action as software praxis. This thesis follows the conventions of critical writing but at the same time is a Perl script that can be executed by typing ‘perl’ and the name of the file in the Unix command line. In this sense, it could only ever represent a work in progress, as something to be argued against, further adapted, and acted upon. It is both a thesis in itself and ready to express its dialectical potential by forming an antiTHESIS

```
use Net::FTP;

local $/;

open       SOURCE, “<$0”;
$source = <SOURCE>;
close      SOURCE;

$beginning = index($source, ‘antiTHESIS’) + 13;
$end       = index($source, ‘antiTHESIS’, $beginning + 1) - 1;

$byte1     = $beginning + rand($end - $beginning);
$byte2     = $beginning + rand($end - $beginning);

(substr($source, $byte1, 1), substr($source, $byte2, 1)) =
(substr($source, $byte2, 1), substr($source, $byte1, 1));
```

```
open  SOURCE, “>$0”;
print SOURCE $source;
close SOURCE;

if ($source !~ /disorder can lead to a new sense of order/) {
  $ftp = Net::FTP->new(“thesis.anti-thesis.net”);
  $ftp -> login(“antithesis”, “sisehtitna”);
  $ftp -> cwd(“Sites”);
  $ftp -> put($0);
  $ftp -> quit;
}
```

===============

7. *references*

===============

7.1 notes

CHAPTER 1:

[1] This is a reference to Kittler's 'There is no Software' (1996) in which he apologises for his use of proprietary software and hardware to produce a 'critical' text.

[2] 'Perl' is an acronym for 'Practical Extraction and Report Language', a high-level programming language, first developed for Unix by Larry Wall in 1987, and developed as an open source project. Perl programs are usually called 'Perl scripts' and are particularly useful for mixed-language script programming. 'Unix' is a trademark of The Open Group, but in general refers to any operating system that is either genetically descended from Bell Labs's ancestral Unix code (developed by Dennis Ritchie and Ken Thompson in 1969) or written in close imitation of its descendants (Raymond 2004: xxix). A longer description of Perl (written with Adrian Ward, and published as part of _Software Studies_ edited by Matthew Fuller) is available online (http://www.softwarestudies.org/).

[3] To execute the Perl script, please type the following into a Unix command line shell.
To run it once, type:
```
perl antiTHESIS.txt
```
To run repeatedly, type:
```
perl -e 'while(1){do "antiTHESIS.txt"}'
```
To run it 60,000 times, type:
```
perl -e 'for(1..60000){do "antiTHESIS.txt"}'
```
The perl script has been written with the help of Adrian Ward.

[4] When the text reaches a critical point of disorder, it will be published at:
http://thesis.anti-thesis.net/~antithesis/
Any subsequent published version of the text will be licensed under the Libre Commons Res Communes License (http://www.libresociety.org/library/libre.pl/Libre_

Commons) to express a cultural politics outside of the legal apparatus. It thereby rejects the recommended copyright statement for PhD submission. The program itself is distributed as free software, meaning you can redistribute it and/or modify it under the same terms as Perl itself (http://www.perl.com/perl/misc/Artistic.html).

[5] The use of the term 'anticonstraint' makes reference to the OuLiPo (Ouvroir de Littérature Potentielle) group that is discussed in more detail in chapter 2.

CHAPTER 2:

[1] However, many relevant histories of media arts exist, for instance: Paul (2003a), Rush (1999), Schwarz (1997), to name a few recent anthologies. Paul's _Digital Art_ refers to software art as a category in itself (2003a: 124-5). Stephen Wilson's _Information Arts_ (2002) offers an alternative category, one that applies information theory to arts practice, but this also remains far too broad a description for the purpose of this thesis.

[2] _Generator_ was curated by myself and Tom Trevor. See http://www.generative.net/generator/ for more detailed information on this exhibition and http://www.anti-thesis.net/ for other documentation. Since, there have been a number of shows that take a historical perspective on software art: _Abstraction Now_ at the Künsterhaus Wien in 2003 and _White Noise_ at the Australian Centre for the Moving Image in 2005 are two examples.

[3] The issue of autonomy in critical theory will be referred to in more detail later in this thesis, making reference to the work of Autonomia in particular.

[4] Generative art was also practised among others by Eduardo McEntyre and Miguel Ángel Vidal (1928-) in Argentina, according to Osbourne (1988). Max Bense's theory of 'generative aesthetics' (1971) which drew together Charles S. Pierce's semiotics with Claude Shannon's information theory, is another reference in this connection.

[5] This is sometimes called 'Cartesian linguistics' to describe the separation of inner consciousness and the outside social world. Perhaps this is what Galanter means when he states that generative art is not ideological.

[6] A concern with grammar has been particularly influential in generative music and composition, such as in the generative music of Brian Eno (using Sseyo's _Koan_ software), and in key texts such as Lerdahl and Jackendoff's _A Generative Theory of Tonal Music_ (1983) that combines the formal methodology and psychological concerns of Chomskian linguistics with Schenkerian music theory.

[7] Bourdieu's concept of 'habitus' has relevance in this connection. It might be compared to Chomsky's generative grammar to emphasise the creative and active capacities of human agents but without the associated difficulties in Chomsky of the universal mind. Habitus accounts for the ways in which agents can act in specific ways without simply being bound by or following rules. It is more a set of 'dispositions' that generate practices and perceptions through 'structured structures', almost as if by second nature. A good example is language, and the ways in which certain forms of language bind people together in groups. Thus habitus is the 'principle that regulates the act' (Bourdieu, in Jenks 1993: 14). One might extend this to include the use of programming languages and the social formations they elicit. The important point is that agents, knowingly or not, generate practices in this way, and they do so within a broader set of social relations.

[8] The 'clinamen' refers to the swerving of atoms in Epicurean atomic theory.

[9] Calvino's title was plagarised for the subtitle of the essay 'how I wrote one of my perl scripts' (Cox, McLean & Ward 2001).

[10] This evokes what Manovich refers to as the 'Flash generation'. However, his is a very general position emphasised by the loose statement that: 'Programming liberates art from being secondary to commercial media' (2002). This is what Wright is addressing, and more detail is required to make clearer the distinctions over proprietary issues and social relations engaged to uphold this view.

[11] Brown represents an older generation of artists associated with this field and has been involved in the research project _Cache_ (from 2002), aiming to 'recover

computer arts history' (http://www.bbk.ac.uk/hosted/cache/). Gere's forthcoming edited collection _White Heat, Cold Logic_, draws on this research.

[12] To clarify the terms: syntax is conventionally defined as the arrangement of words and phrases to create sentences, a set of rules for the analysis of this, and the structure of a statement in a computer language. Semantics is the branch of linguistics and logic concerned with meaning.

[13] Williams, in _Keywords_ (1988) stresses the term culture's complex historical development and the ways in which it has become important in several distinct intellectual disciplines, and in several seemingly competing systems of thought.

[14] This was further developed by Eco into a parody for generating movie scripts in 1972, preempting the commercial software _Plots Unlimited_ (1994) that exemplifies the standardisation of form in contemporary movie-making (in Cramer 2005: 81).

[15] The 'death' was intended to shift emphasis onto the words on the page, or the nature of the surrounding language and discourse - and away from associated myths of originality and genius, what Barthes refers to as 'the "message" of the Author-God' (1977: 146). The death of the programmer would be welcome in the sense that the programmer or software artist is often associated with myths of originality and genius.

[16] This is further explained in response to Foucault's question 'Who is speaking?' from 'What is an Author?' (1991): 'Mallarmé replies... the word itself... in a pure ceremony of the Book in which the discourse would compose itself' (in Burke 1992: 9). There is a pressing need to examine new demarcations, and the functions released by the alleged disappearance of the author.

[17] Hayles's book follows a format partly autobiographical and composed in close collaboration with a graphic designer. The results are mannered and awkward but the point is clearly made. The materiality of text or code is further verified by the property rights exerted on it - intellectual property would even cast (tangible) ideas as material objects in this

respect (and this is an important issue that will be returned to in chapter 5).

[18] A ‘Quine’ is named in honour of Willard Van Orman Quine, an influential mathematician and philosopher who died in 2000. See Gary P. Thompson’s ‘The Quine Page’ (http://www.nyx.net/%7Egthompso/quine.htm).

[19] Recent attention to Greenberg’s work has tended to concentrate on this formalist position and what he calls the ‘irreducible essence’ of pictorial art, in his 1965 essay ‘Modernist Painting’ (1992). Drawing upon Kant’s idea of ‘self-definition’, he stresses the flatness of the picture plane as a distinguishing characteristic of Modernist painting - even Jackson Pollock’s work demonstrates a tension inherent in the constructed flatness of the surface, acccording to Greenberg. In many accounts, his work on abstract expressionism implies an endorsement of neo-liberal ideology and American individualism.

[20] Elsewhere this is sometimes called ‘code slang’ and more generally ‘code narrativity’. Clearly it is possible to imagine a ‘creole’ consisting of natural language and code such as Mez’s work (a creole is a new language - not an amalgam like ‘pidgin’ - formed where two existing languages come into contact. A further example would be Antiorp/Netochka Nezvanova’s semi-legible creoles, in which meaning and authorship are held in question.

[21] ‘Laying bare the device’ is a phrase associated more precisely with Victor Shklovsky’s study of Laurence Sterne’s _Tristram Shandy_ (of 1759).

[22] In a similar way, Wark’s _The Hacker Manifesto_ requires that hackers take control and seek autonomy over what they produce, to identify their interests as a class in order to serve society as a whole, and strike alliances with other workers (2004).

[23] This was a project by myself, Tim Brennan and Adrian Ward, exhibited online and as part of exhibitions: _Manifest: Library_ (1999) as part of HUB, Bishopsgate Goodsyard, London (commissioned by Cityside & University of East London); and as part of _A Timely Place, or, Getting Back to Somewhere_, London Print

Studio (2000). The paper ‘Manifest: Reframing False Consciousness’ was presented as part of _Consciousness Reframed_, University College Newport, Wales, & _ Phenomenology_ conference, University College, Cork, Eire (both 2000). The User’s guide is called _Manifest_, published by Working Press 1999.

[24] It is tempting to playfully claim that ‘software art has no history’ - making reference to John Roberts’s _Art has no History!_ (1994) examining the ideological construction of art history. This in turn is a reference to Althusser’s statement that ‘Ideology has no History’ (1997), that referred to Marx’s _The German Ideology_ (1978 [1845/6]) in which he proposes that ideology has no history, ‘since its history is outside it, where the only existing history is, the history of concrete individuals’ (in Althusser 1997: 121). The idea that ideology has no history is thereby a negative thesis to indicate that ideology is pure illusion produced by those in power, but also its sense of history is a mere reflection of ‘real history’ - it has ‘no history of its own’ (1997: 122). If the same can be said of art history, can the same be said of software art history, to reveal that the power relations that it expresses are illusory in the same way?

[25] Use-value is something Saul Albert discusses in his essay ‘Useless Utilities’ (2002), opposing the romantic notion that defines art in terms of its lack of utility. In a worse case scenario Albert suggests that ‘art for art’s sake has been replaced by the idea of art for technology’s sake’ with software simply reduced to the role of tool (2002).

[26] There are key examples in a history of art and technology that might be mentioned in this connection, such as the ‘sci-art’ work of Leonardo da Vinci and the ‘experiments in art and technology’ (EAT) involving the engineer Billy Klüver working with John Cage and Robert Rauschenberg amongst others. Of particular interest is Klüver’s collaboration with Rauschenberg _9 Evenings: Theatre and Engineering_ (1960), which incorporated new technology developed by 10 artists, working with more than 30 Bell Labs engineers.

[27] This is the position taken by Habermas in ‘Modernity - An Incomplete Project’ (1991 [1980])

opposing emergent notions of post-modernity and post-history at that time. Modernity describes a transition state between the old and the new. New technology stands as an exemplar for the wanton post-modern consumerist condition of newness never being allowed to settle in the present - a paradoxical combination of an obsession with nostalgia and at the same time with the idea of almost instantaneous obsolescence.

[28] It should be mentioned that the 'Manifesto of the Futurists' is often cast as proto-fascist. Elsewhere, Benjamin adds that the Futurist obsession with the aesthetics of politics rather than the politics of aesthetics can lead only to one thing: war. And this is precisely what happened, in an appropriate ironic twist, where the leading figures of futurism were killed by the very machines they valorised. Benjamin makes a dialectical opposition of the aesthetics of politics and the politics of aesthetics: 'This is the situation of politics which Fascism is rendering aesthetic. Communism responds by politicizing art.' (1999b: 235)

[29] Pedagogy considered as an art form is exemplified by the issues of communication and distribution that conceptual art posed. Joseph Beuys coined the term 'social sculpture' with this in mind: 'To be a teacher is my greatest work of art. The rest is waste product, a demonstration[...]. Objects aren't very important to me any more[...]. I am trying to reaffirm the concept of art and creativity in the face of Marxist doctrine[...]. For me the formation of the thought is already sculpture.' (in Lippard 1997: xvii)

[30] In this context, 'connectionism' stands for 'order-emerging-out-of massive-connections', an approach to artificial intelligence that later became known as neural networks (Kelly 2003: 360-1).

[31] A further link can be drawn to the Planetary Collegium (http://www.planetary-collegium.net/) of which Ascott is Director, and the research context from which this submission for PhD derives.

[32] It should be said that the _Software_ show builds upon a range of other influences, such as _The Machine as Seen at the End of the Mechanical Age_ at the Museum of Modern Art (1968), and _Art by Telephone_ at the

Museum of Contemporary Art in Chicago (1969), as well as _Cybernetic Serendipity_ at the ICA in London (1968).

[33] This quote could easily have been taken from Wark's _The Hacker Manifesto_ that also argues for a politics based on an engagement with property that has shifted from land, to industrial production, to information (2004). The _Radical Software_ journal's current availability on the Internet as free PDF downloads is therefore thoroughly in keeping with the ethos of open content publishing initiatives (from http://www.radicalsoftware.org).

[34] Another key reference is Burnham's 'Systems Esthetics' (1968b) that informs his _Beyond Modern Sculpture_. Gere's essay 'Jack Burnham and the Work of Art in the Age of Real Time Systems' (as the title suggests) makes the following claim: 'that Burnham is to art in the age of real time systems what Walter Benjamin was to art in the age of its mechanical reproducibility' (2005: 149).

[35] It was the Fluxus artist Henry Flynt who allegedly coined the term 'concept art'. Other influences include Alan Kaprow's 'happenings', as well as concrete poetry, mail art, performances, body and street works.

[36] This is a statement from 1967. For the _Generator_ show, which used the quote in publicity, LeWitt presented a 'serial variation' (or algorithm) using found postcards of Chicago in which they become increasingly layered.

[37] Somewhat similar in spirit, but with added irony, is Cornelia Sollfrank's statement on her web site, made with reference to her net.art generators: 'A smart artist makes the machine do the work!'

[38] Kluitenberg writes: 'In this paradoxical environment [of the Internet], dominant discourses of social, political and economic power can be challenged at the level of the representational systems they employ. The classical avant-gardes provide a repository of ideas, tactics and strategies that are now played out in a radically enlarged context; no longer the context of art itself, but that of the network society.' (2002)

[39] This is perhaps a reference to Tristan Tzara’s statement of 1918: ‘There is a great negative work of destruction to be accomplished.’ (in Harrison & Wood 1998: 252) This further relates to the ‘negative dialectics’ of Adorno that will be introduced in the following chapter.

[40] For instance, see the project _Gustav Metzger is My Dad_ (1998) that Camerawork organised on the occasion of its funding cut, where much of the paperwork associated with funding was shredded (http://www.anti-thesis.net/projects/shredding/images.html). In parallel, a digital version shredded the html of web pages (http://www.anti-thesis.net/projects/shredding/source.txt). I was co-producer of these works.

[41] The source code of _biennale.py_ is available in spoken form (http://www.epidemiC.ws/love.mp3). That a virus might be regarded as a work of art has a history too. Citing Baumgärtel, Cramer describes the work of Artemus Barnoz, in 1988, secretly installing a systems extension that produced a new age peace message on every system startup (in Nori 2002: 76).

[42] It reads: ‘:(){ :|:& };:}’. To explain, ‘:()’ defines a function named ‘:’, run a copy of itself if the function is called ‘{:|:&};’, execute the function ‘:’.

[43] The false declaration of love is a particularly cruel one in a world that lacks love. The analogy of the virus is not without its problems of description either. Fuller makes reference to the work of David Wojnarowicz who died of an AIDS related illness in 1992, and his realisation that he’d not only contracted a virus but also the realisation that society was diseased (2004: 28).

[44] That new media looks ostensibly like old media is a similar observation to Jay Bolter and Richard Grusin’s idea of ‘remediation’ (1999) to describe the ways in which media are recycled into other media.

[45] Allegory is explained: ‘It has two important technical properties: the anti-symbolist ability to disrupt aesthetic illusions of the real, and the forcing together, through montage or image pile-ups, realms that are seemingly discrete, but actually connected.

[...] Allegory is a technical means to retransmit discontinuity, fragmentation and a catastrophic structure of history.' (Leslie 2000: 199).

[46] The description of _The Bank of Time_ is taken from the _Runme_ feature that I previously wrote in 2004 (http://runme.org/project/+BoT/).

[47] Reflecting on his _Passagen-Werk_ (what elsewhere he called the 'Dialectical Fairyland', in Tiedemann 1999: 932), Benjamin further explains his method: 'That is, to assemble large-scale constructions out of the smallest and most precisely cut components. Indeed, to discover in the analysis of the small individual moment the crystal of the total event. And, therefore, to break with vulgar historical naturalism. To grasp the construction of history as such.' (1999a: 461) More detail on this is included in chapter 3.

CHAPTER 3:

[1] The Klee painting was bought by Benjamin in 1921.

[2] Leslie's translation differs from the commonly distributed Harry Zohn translation (Benjamin 1999c: 249). It also prefigures Marshall McLuhan's statement that: 'We look at the present through a rear-view mirror. We march backwards into the future.' (1967: 74)

[3] This is a quite different sense of catastrophe than recent events that occupy the minds of those who fear terrorism. Slavoj Žižek would see fundamentalist terrorism in terms of the 'passion for the real' - in the case of 9/11, America simply 'got what it fantasized about' and the 'Real' violently entered everyday reality (2003). It was the unlikely figure of Karl-Heinz Stockhausen who pointed this out in his statement that 9/11 was the ultimate work of art.

[4] The passage is an intriguing one with many further references. For instance, the spiritual overtones are important for an understanding of Benjamin's work. His interest in Messianism and the Kabbalah clearly has a bearing on his views of historical progress and redemption. To elaborate on this would be too much of a tangent in this connection, as would an exploration of the figure known as the 'Turk' that engages Orientalist

fantasies of the time.

[5] This is also an unwitting reference to Duchamp's _The Bride Stripped Bare by Her Bachelors, Even (The Large Glass)_ (1915-23). Duchamp is an especially suitable reference in this connection as he was a keen chess-player, interested in the automated aspects of the game itself.

[6] Bateson's concept 'metalogue' (1972) suggests a similar reflexive logic in describing a conversation in which the form of discussion embodies the subject being discussed. This thesis operates in a similar manner by embodying the subject of software as software.

[7] There is nothing but this determining contradiction of matter in motion, explains Mao in his 'On Contradiction'. It is through this that different forms can be identified such as: 'positive and negative numbers in mathematics; action and reaction in mechanics; positive and negative electricity in physics; dissociation and combination in chemistry; forces of production and relations of production, classes and class struggle, in social science; offence and defence in military science; idealism and materialism, the metaphysical outlook and the dialectical outlook, in philosophy; and so on.' (1977: 36).

[8] 'Apocatastasis' does not appear in the dictionary - I assume it combines apocalyptic and stasis in describing damage due to lack of change on a dramatic scale.

[9] The quote continues: 'The tradition of all dead generations weighs like a nightmare on the brain of the living. And just when they seem engaged in revolutionising themselves and things, in creating something that has never yet existed, precisely in such periods of revolutionary crisis they anxiously conjure up the spirits of the past to their service and borrow from them names, battle cries and costumes in order to present the new scene of world history in this time-honoured disguise and this borrowed language.' (Marx 1980: 96) According to Marx, the more the present is in crisis, the more one has to borrow from the 'spirits of the past'. Thus the living borrow from the dead - to stress the emphasis that Derrida grants it in _Spectres

of Marx_ (1994).

[10] Fukayama’s argument is that American Empire brings an end to European history. Derrida calls Fukuyama’s work the new ‘gospel’, to refer to its Christian overtones. Fukuyama is explicitly drawing upon Hegel’s _Phenomenology of Spirit_ (sometimes called _Phenomenology of Mind_) but also this is a reference to the work of Alexander Kojève of 1947, and his ‘postscript on post-history and post-historical animals’ (1994: 70).

[11] Ironically the same criticism has been levelled at Marx for his mystification of the dialectical method - amongst others, by Popper, in _The Open Society and its Enemies_ (2003 [1945]). Popper is concerned to assess the contributions of Hegel and Marx on an understanding of history: ‘What I wish to show is that Marx’s “materialist interpretation of history”, valuable as it may be, must not be taken too seriously; that we must regard it as nothing more than a most valuable suggestion to us to consider things in their relation to their economic background.’ (2003: 120)

[12] Ideology describes how ideas reproduce themselves. The history of the term ideology itself reveals a further connection to software in describing a genetic theory of ideas - and Žižek’s term for ideology ‘generative matrix’ perhaps derives from this.

[13] Lefebvre, referring to Hegel’s ‘end of history’, accuses Lukács of conceiving of the ‘end of philosophy’ through his theory of class consciousness (1968: 36-7). Lurking in the background here is a more complex philosophical argument over the opposition of idealism and materialism, making reference to Marx’s ‘The German Ideology’ of 1845/6 and ‘Manuscripts of 1844’. Early Marx rejects both idealist and materialist philosophy for revolutionary praxis.

[14] Terry Eagleton explains that in effect, Lukács has adopted Hegel’s ‘absolute idea’ for the proletariat and that through the dialectical method, truth can eventually be found in the whole through overcoming ‘reification’. To explain reification briefly: it remains a useful concept under consumer capitalism, more in fact a precondition, as traditionally the ‘transformation of social relations into things’ but also the ‘effacement of

the traces of production' (Jameson 1991: 314), leaving people to happily consume free of guilt.

[15] Althusser is stressing the importance of the 'ideological State apparatuses' (including the family, schools, church, legal apparatus, political system, trade unions, communications media, arts and culture, etc.) that operates more covertly than the overt violence of the 'repressive State apparatuses' (including the government, army, police, courts, prisons, etc.) (1997: 110). This is not a new phenomena. In pre-industrial times, the ideological state apparatus worked through religion predominantly, controlling other apparatuses like education, communications and culture. He thinks this central position has now been taken by the education apparatus in capitalist social formations (1997: 116), and the contemporary conception of the importance of the 'knowledge economy' would appear to continue his emphasis.

[16] One can easily apply this to the academicisation of critical theory, the publishing industry that has built up around it, and the abstraction of theory from everyday praxis: 'The introverted thought architect dwells behind the moon that is taken over by extroverted technicians [as] no theory escapes the marketplace' (Adorno 2000: 3, 4).

[17] The integrative power and levelling tendencies of mass culture is what Adorno and Horkheimer's essay 'The Culture Industry' addresses directly (1997 [1944]).

[18] The Frankfurt Institut for Social Research developed Marxist social theory, influenced by Freud and Max Weber in particular. It is often charactised as 'critical theory' and is associated with Adorno, Horkheimer, Benjamin, Marcuse, Habermas, Arendt, amongst others. The work of Susan Buck-Morss (such as _The Dialectics of Seeing_), also quoted in this thesis, draws from this tradition. Jay's _The Dialectical Imagination: A History of the Frankfurt School and the Institute of Social Research 1923-1950_ (1996 [1973]) contains an extensive history.

[19] The idea of awakening contained a particular theological and mystical significance for Benjamin. The dialectic of waking and sleeping is further described in Guy Debord's _The Society of the Spectacle_ (1998

[1967]) in which the impulse is also to awaken from the bad dream of capitalism, to shake the sleeping political consciousness out of its slumber.

[20] ‘The Spectre is Still Roaming Around!’ as Žižek puts it elsewhere (1998). Indeed, the first noun in _The Communist Manifesto_ is ‘spectre’ and it immediately returns (like the repressed): ‘A spectre is haunting Europe – the spectre of communism’ [‘Ein Gespenst geht um in Europa – das Gespenst des Kommunismus’]. Žižek used the phrase ‘The Spectre is Still Roaming Around!’ for the title of his introduction to the 150th anniversary of _The Communist Manifesto_ published as a separate volume (1998). Elsewhere, in _The Ticklish Subject_, Žižek parodies _The Communist Manifesto_, and playfully begins: ‘A spectre is haunting western academia, the spectre of the Cartesian subject. All academic powers have entered into a holy alliance to exorcise this spectre [...] (1999b: 1).

[21] This book supports this view, not least in its use of the programming language Perl, in which ‘AND has higher precedence than OR does’, according to its creator Wall (1999). See the entry to _Software Studies_ on Perl (Cox & Ward 2007).

[22] Popper is also a well-known critic of Marxism. In _The Open Society and its Enemies_ (2003 [1945]), he accuses Marx of overstressing economism in what he calls ‘economic historicism’ (2003: 110).

[23] Bhaskar’s critique also casts Darwinian evolution as teleological and hence closed. It is important not to conflate this critique with the reactionary creationism of fundamentalist Christians, whose sophistication stopped with the first chapter of _Genesis_ according to Bateson (2000: 434). Equally misleading and apolitical is the alternative view that: ‘species go extinct not because of bad genes but because of bad luck’ (David Raup in _Extinction_, quoted in Goodwin 1997: 116). Furthermore, it makes an unacceptable political metaphor, of the inevitability and naturalness of free trade, open competition and market forces, where the rich get richer and so on. Engels summarises this problem as follows: ‘Darwin did not know what a bitter satire he wrote on mankind, and especially on his countrymen, when he showed that free competition, the

struggle for existence, which the economists celebrate as the highest historical achievement, is the normal state of the animal kingdom.' (1980: 351)

[24] Prigogine and Stengers cite Ludwig Boltzman who investigated the correlation of probability and irreversibility: 'Only when a system behaves in a random way may the difference between past and future, and therefore irreversibility, enter into its description.' (1985: 16)

CHAPTER 4:

[1] Following the economic crisis in the 1970s, Castells describes the conditions for the change in the evolution of capitalism 'to overcome its own contradictions', and to escape delimiting restrictions imposed by state-controlled industrial forces (1996: 51). Thus, reform sought to deepen the capitalist logic of profit-seeking and enhance productivity by globalising production, circulation and markets, whilst establishing state support for these policies, often to the detriment of social and public interests (Castells 1996: 19).

[2] Jameson relates these economic stages directly to cultural production as follows: realism (worldview of realist art), modernism (abstraction of high modernist art) and postmodernism (pastiche) (1991).

[3] There are further links here that acknowledge relatively distinct periodisations that relate to machines and capitalist restructuring: this sense of pervasiveness enabled by networked computers corresponds to what Gilles Deleuze calls a 'society of control' modelled on the 'third wave' of computerised machines (in his 'Postscript on Control Societies', from _ Negotiations_, in Galloway 2004: 3).

[4] Incidentally, although humans have named ants in pejorative terms, the 'queen' is not an authority or camp figure, merely an egg-laying ant and does not direct or exploit the workers as such.

[5] Although DNA's double helix, as the basic structure of life was identified in 1953 (by Francis Crick and James Watson), it was only by the 1970s that genetic engineering became widely practised with the cloning

of the first human gene in 1977 (Castells 1996: 48) and subsequently the idea of engineering life has become big business (with resultant battles over property rights and who should own the copyright on gene research). The ethical issue came to public attention in 1988 when scientist entrepreneurs at Harvard University challenged the moral and ethical agenda of God and Nature by patenting a genetically engineered mouse. By now, the human genome has been extensively mapped and despite the best efforts of those who would consider life to be in the public domain, scientist-entrepreneurs have gained legal and economic control. Thus, humans themselves are becoming increasingly privatised in a perversion of nature. Some artists are working in this area but tend to employ crude analogies or simply illustrate the issues. Eduardo Kac's 'transgenic' rabbit _Alba_ is one provocative exception that engages with the discourse and ethics of genetics (http://www.ekac.org/gfpbunny.html).

[6] Biology, like technology, is clearly caught up in complex cultural narratives of power, knowledge and subjectivity. This is reminiscent of the ways in which Foucault theorised the body and technology as bound together in the construction of power. The human genome project is an obvious example of the ways in which knowledge and power serve the interests of institutions over individuals (Kember 2000: 157). Foucault maintains that there is no unitary human subject, except that which is produced through discursive processes and forms of rationality that produce the subject as the object of knowledge – in the complex relationship of knowledge-power. Throughout the nineteenth century, the body was continually made subject to medical and psychological examinations to render ruling capitalist and imperial ideology as 'true' knowledge. This is the normalising power of the 'carceral network' that did not exercise power directly on the body but on the body as the object of knowledge. For instance, research in this area promises: 'a DNA-level quality control over the reproduction of labor power, control aimed not at the cure of disease but at the disgrading of potentially unproductive, oversensitive, or expensive units' (Dyer-Witheford 1999: 106). To the artist/activist collective Critical Art Ensemble, this is ostensibly an eugenics programme.

[7] In the industrial period, according to Adam Smith, the worker ‘generally becomes as stupid and ignorant as it is possible for a human creature to become [and] in every improved and civilised society, this is the state into which the labouring poor, that is, the great body of the people, must necessarily fall’ (quoted in Marx 1990: 483). This is perhaps more a question of disaffection as the worker is no more stupid than the system they labour for: ‘In fact, of course, this “productive” worker cares as much about this crappy shit he has to make as does the capitalist himself who employs him, and who also couldn’t give a damn for the junk’ (Marx 1981: 273). To be fair, Smith’s argument is intended to suggest that education is necessary for these very reasons, although he does not extend this to an understanding of how the education system itself ‘reproduces’ capitalist interests - something Althusser and Bourdieu describe in more detail (see chapter 3).

[8] Despite this tendency to imagine the worker-less factory, the process of production evidently still rests on living labour but it is organised in network forms. Marx puts it like this: ‘A machine which is not active in the labour process is useless[...]’. (1990: 289). Even a so-called autonomous system cannot produce value in itself.

[9] Klein says much the same in her observations on the way that labour is subordinated by the machine: ‘IBM claims that its technology spans the globe, and so it does, but often its international presence takes the form of cheap Third World labour producing the microchips and power sources that drive our machines. On the outskirts of Manilla, for instance, I met a seventeen-year-old girl who assembles CD-Rom drives for IBM. I told her I was impressed that someone so young could do such high-tech work. “We make computers,” she told me, “but we don’t know how to operate computers”.’ (2001: xvii) In the chapter ‘The Discarded Factory’, Klein describes appalling exploitation that is the reality of globalisation, paying particular attention to violence and corruption within free trade zones.

[10] By referring to both system and hierarchy, Hardt and Negri aim to make a hybrid of Niklas Luhmann’s systems theory (in which society is described in terms of autopoesis rather than made by humans as such) and John Rawls’s theory of justice. By ‘governance without government’, they are

referring to the title of a book by James Rosenau and Ernst-Otto Czempiel (1992). Elsewhere, this myth of a democratic, nonhierarchical, noncentred network structure is what Deleuze and Guattari describe as the 'rhizome' (1987: 3–25).

[11] This description of a distributed management system lies behind the _Kurator.org_ project as a distributed curatorial system for open source code - using protocols for different ends than centralised/decentralised and proprietary interests. Kurator.org asks: 'If the assumption is made that traditional curating follows a centralised network model, then what is the position of the curator within a distributed network model?' (Krysa & Sedek 2005) The suggestion of the project is that the artist-programmer characterisation is extended to that of the curator-programmer, and software art to software curation.

[12] This can be traced earlier to Leibniz in the seventeenth century, who thought that clockwork automata could express perfection when constructed by God, but not when constructed by mere humans. According to Cartesian logic at this time, mind and matter are seen to be autonomous entities but little attention is given to the dynamic interrelation of the two. This is why the work of Leibniz is particularly influential to Wiener (2000: 41).

[13] For a thorough technical history, see Paul E. Ceruzzi's _A History of Modern Computing_ (2003 [1998]). The book covers the development of the electronic digital computer in the 1940s to the spread of networking after 1985, and in the second edition through to the development of open source software after 1995. In any given history, there are vested interests in which history is preferred. For instance, in _The Language of New Media_, Manovich cites Konrad Zuse to situate the beginnings of 'new media' in keeping with his central analogy to the history of cinema (2001: 25). In contrast, Geoffrey Batchen, a historian of photography, disputes this version of events and proceeds to describe photography in binary terms: the presence and absence of light, and on/off tonal patterning representing numerical repetitions of units to make up a whole image (Batchen, in Kimbell 2004: 29). Indeed, he claims it is 'a fledgling form of information culture' made more explicit by Fox Talbot's 1839

proposal to replace the use of sunlight by the spark of electricity: ‘a making visible of electricity’ (in Kimbell 2004: 31). Batchen refers to a Fox Talbot image _Lace_ (of 1845) to make the link to a longer history of lace-making and computation.

[14] In describing the factory as a ‘self-regulating system in embryonic form’, Marx, using bio-technological metaphors claimed: ‘An organised system of machines to which motion is communicated by the transmitting mechanism from an automatic centre is the most developed form of production by machinery. Here we have, in place of the isolated machine, a mechanical monster whose body fills whole factories, and whose demonic power, at first hidden by the slow and measured motions of its gigantic members, finally bursts forth in the fast and feverish whirl of its countless working organs.’ (1990: 503) Is the achievement of technology simply to fulfill this nightmarish vision of automation?

[15] Workers as ‘second-order robots’ refers to a history of the term ‘robot’ itself. It was allegedly first used by Karel Capek in his play ‘Rossum’s Universal Robots’, in Prague in 1921, drawing upon the Czech term ‘robota’ which literally means ‘forced work or labour’ from the Latin ‘robor’ meaning power or force (Floridi 1999: 207). The play typically describes a scenario in which a factory that builds artificial agents is eventually taken over by them and the whole of humanity destroyed.

[16] This is also a reference to Benoit Mandelbrot’s _ The Fractal Geometry of Nature_, 1883, and his question: ‘How long is the coast of Britain?’. The answer is infinitely long or that it depends on the length of your ruler. Mandelbrot surmises that as the length of the measurement becomes smaller, the coastline gets longer - to the point where it is being measured at an atomic scale, when it becomes infinite.

[17] An infinite loop is a sequence of instructions in a computer program which loop endlessly.

[18] Boolean logic has many applications in electronics, computer hardware and software. In _Zeros + Ones_ (1997), Sadie Plant relates this logic to sexual politics. She explains with irony how ones and zeros, male and female,

penis and vagina, all make ‘lovely couples’ (1997: 35). In this sense, ‘It takes two to make a binary’ and set up the heterosexual paradigm. Taking the analogy to sex further, artificial life can be understood as a heterosexist discourse, with its emphasis on the desire for reproduction as one of the definitions of life (Kember 2003).

[19] The idea that a machine might demonstrate intelligence is derived from Alan Turing’s paper ‘Computing Machinery and Intelligence’ of 1950, hence the so-called ‘Turing Test’ to measure whether a machine might pass for a human. Hofstadter’s ‘A Coffee House Conversation on the Turing Test’ (1985 [1981]) is sceptical about the claims of artificial intelligence, setting the richness of human imagination and emotions against the mechanicist promises of artificial intelligence in the form of a conversation.

[20] In Heim’s view, class conflict is a thing of the past, which says something about the commodification of dialectics by academics and publishers alike, keen to appear radical to satisfy the market but not upset it.

[21] Gotthard Günther, in ‘Grundzüge einer neuen Theorie des Denkens in Hegel’s Logik’ [‘Main Features of a New Thinking in Hegel’s Logic’], situates classical binary logic as part of a more general and comprehensive multivalued or many systems logic (Paul 2000). What further captures the imagination is that Günther planned to build a ‘transputer’, a machine based on ‘polycontextural logic’ (how he perceived human consciousness). There is some contemporary interest in this logic in as far as it relates to networked technology, in that it arguably reflects Günther’s polycontextural logic.

[22] Owens is partly concerned to distance herself from what she sees as the mistaken (postmodern) view of complexity as proof of uncertainty, virtuality and scientific myth-making (1996). She is particularly thinking of Kuhn’s _The Structure of Scientific Revolutions_ (1970), and Paul Feyerabend’s _Against Method_ (1975). Owens suggests that scientific method has always embraced a strategic sense of uncertainty, not just the arts and humanities (as indicated in chapter 3 when discussing reflexivity and recursion). Similarly Brian Goodwin (in his _How the Leopard Changed Its Spots_,

1994) too easily equates this sense of uncertainty to a critique of modernity (1997: 114). He is assuming modernism to affirm determinism, whereas critical modernity has always embraced uncertainty and its own critique, and should therefore not necessarily be seen as deterministic but able to embrace its contradictions (as described in section 3.2, with reference to Berman in particular).

[23] The strange attractor demonstrates 'infinite regress', an inexhaustible sequence of folding and stretching a line. When a change takes place in a predicted chain of events, the strange attractor causes the initial system and the disturbed system to move apart exponentially fast (paraphrased from Gleick 1998: 150–1).

[24] Lukács in _History and Class Consciousness_ would de-emphasise the application of dialectics to nature, in favour of the social and conceptual realms only (1976). Whereas Jay describes Marcuse's position as: 'Natural being was different from historical being; mathematical, nondialectical physics was valid in its own sphere: "Nature," Marcuse wrote, "has a history, but is not history"' (1996: 73). Antonio Gramsci also shifts the dialectic away from the contradiction inherent in nature and emphasises the contradiction between reality and the will of the subject, in calling for a 'pessimism of the intellect, [but] optimism of the will'. This is usually attributed to Gramsci but is a variation of Romain Rolland's phrase 'pessimism of the intelligence, optimism of the will' (footnote, in Hoare & Nowell-Smith 1971: 174).

[25] More detail on this issue of incomplete synthesis was introduced in chapter 3. Otherwise, false totalities emerge. For example, Stalinism is accounted for its lack of open-endedness, as it wrongly assumed the dialectical process to have ended, and closed it down to drastic effect.

[26] An English translation and hypertext version of Queneau's 'A Story as You Like It' ['Un conte à votre façon'] is available online (http://www.thing.de/projekte/7:9%23/queneau_1.html).

[27] Prigogine and Stengers state: 'A society defined entirely in terms of a functional model would correspond

to the Aristotelian idea of natural hierarchy and order. Each official would perform the duties for which he [sic] has been appointed. These duties would translate at each level the different aspects of the organization of the society as a whole. The king gives orders to the architect, the architect to the contractor, the contractor to the worker. On the contrary, termites and other social insects seem to approach the "statistical" model. As we have seen, there seems to be no mastermind behind the construction of the termites' nest, when interactions among individuals produce certain types of collective behaviour in some circumstances, but none of these interactions refer to any global task, being all purely local.' (1985: 205)

[28] For instance, and according to Owens, a theory like deconstruction is 'trapped in the very dualism it seeks to circumvent' (1996: 91). Other examples were mentioned earlier in this connection (see note 22).

[29] Alternatively, there could be a forceful logic in making a historical link to Anarchist principles in describing a 'political system' that emphasises disorder and chaos. This is what Bey does in _T.A.Z.: The Temporary Autonomous Zone, Ontological Anarchy, Poetic Terrorism_ (2003 [1985]), describing anarchy as chaos, and chaos as the principle of continual creation, of 'all-potentiality' (2003: 70). He is making reference to what Prigogine calls 'creative evolution' to account for the creative potential of 'perturbations, crashes, and breakdowns in the Net' (2003: 111). By drawing upon Taoist thinking, Bey resists what he sees as the negativity associated with chaos theory, or its link to new ageism or science that sees it as a negative force of destruction or for enforcing order.

[30] There are a number of examples of the ways in which in practice, Marxism has sought to separate dialectics from materialism: otherwise remaining in impoverished form under Stalinism, or by focusing almost exclusively on contradiction through Maoism (Owens cites Mao's _On Contradiction_). According to Mao, there is nothing but contradiction of matter in motion, following in the tradition of Engels in this respect (see note 9 to chapter 3).

CHAPTER 5:

[1] Like Žižek, Guattari sees the reorganisation of better social relations as no more difficult to imagine than other scientific or aesthetic endeavours - no more difficult to 'solve than questions of quantum physics or the manipulation of genes' (1995: 46).

[2] Guattari refers to an 'intradisciplinary' approach, as the capacity to traverse different fields, in contrast to an interdisciplinary approach that would tend to make the mistake of making a synthesis of heterogenous positions.

[3] To state the obvious, the antithetical title of their _Anti-Oedipus: Capitalism and Schizophrenia_ (1990 [1972]) explicitly negates the oedipal drama; the subtitle indicates the 'intradisciplinary' principle (described in the previous note) of drawing together capitalism and schizophrenia.

[4] To Marcuse, so-called 'perversions' such as homosexuality operate as a potential challenge to the exploitative organisation of labour, as expressed in procreative social reproduction (Geoghegan 1981: 53-4). Deleuze and Guattari also cite Wilhelm Reich in this connection to understand the mechanics of fascism. Their emphasis on desire explains in a more sympathetic way Reich's astonishment that the masses do not steal and strike on a regular basis, and tolerate being humiliated and enslaved. Deleuze and Guattari would have us re-read Marx, but also Adolf Hitler, to understand how the desiring-machine operates (in Guattari 1995: 248). Similarly, in the work of the Frankfurt School, Oedipal resistance to the father lent itself to the study of authority (and by extension the relationship of the individual to society) in as much as they were trying to understand the psycho-social conditions in which workers rejected their historical role within Marxism to accept Nazism.

[5] Freud would advise that if you repress the existence of something, even repression itself, it will return anyway at unexpected moments, often as trauma. This Freudian model of latency is what Jameson calls the 'returns of the repressed of historicity' (1991: xvi).

[6] The surrealist Francis Picabia describes the machine as ‘the daughter born without a mother’ (in Guattari 1995: 125).

[7] Indicating his intellectual preferences, Leclaire would like to reintroduce some dualisms such as the real and the symbolic (Lacan), or the base and the superstructure (Marx).

[8] Negri also co-wrote _Communists Like Us_ with Deleuze (1990). As part of the Italian group ‘autonomia’ founded in the 1970s, Negri and others tried to open up new possibilities for the theory and practice of class struggle. Many of the ideas associated with autonomia were developed through the journal _Futur Antérior_ [future perfect], and the contributions of Hardt, Lazzarato, Negri, Virno, mentioned later in this chapter.

[9] It represents ‘simultaneous separation and coherence’, according to Jim Fleming in the ‘Editor’s Preface’ (1991: vii) in words that echo the process of connections and rupture described in the work of Deleuze and Guattari (in Guattari 1995: 126-7). To add more detail and theoretical connections, Negri’s _Marx after Marx_ results from a series of lectures at the Université Paris in 1978 at the invitation of Althusser.

[10] ‘Subsumption’ indicates the ways that one thing is absorbed into another. In this context, class exploitation is subsumed into broader social forms and life in general.

[11] The ‘multitude’ (taken from Benedict de Spinoza’s phrase ‘democracy of the multitude’), expresses the ‘coexistence of the positive and the negative on the terrain of immanence’ according to Hardt and Negri in _Empire_ (2000: 374). Their _Multitude: War and Democracy in the Age of Empire_ adds more detail on the possibility for a revolutionary democracy, as does Virno’s _A Grammar of the Multitude_ (2004), drawing particularly on the contrasting views of Thomas Hobbes and Spinoza to develop an understanding that is derived from Marx’s idea of general or mass intellect.

[12] Hardt and Negri say: ‘The proletariat is not what it used to be, but that does not mean it has vanished’

(2000: 53). A broader definition of proletarians would include the 'marginalised proletariat' of students, the unemployed and unpaid house workers. Even technologies that have changed the nature of work might also be described as somewhat 'proletarianised' according to John Armitage (2002). Some commentators, in what Armitage calls the 'neoliberal discourse of technology' (2002), would go further and suggest that not only is human labour no longer at the centre of production but technology is instead. For more on this, see Jeremy Rifkin's _The End of Work_ (1995) - although this is not a view this thesis supports.

[13] This understanding builds upon the work of Habermas in his _Theory of Communicative Action_ (1984), that updates the concept of historical materialism to take account of communicative action.

[14] The free distribution of source code is free only in the sense that it can be further adapted and changed (under certain conditions of course). This is what Stallman refers to as 'copyleft' protected by the GNU public license agreement for future free provision and distribution under the same conditions. See http://www.opensource.org/ and then http://www.fsf.org/ for more detail on the distinction between this and open source.

[15] The distribution of new knowledge associated with a PhD thesis is similarly revealing. This explains the purpose of the manner in which this thesis is distributed. It is first copyrighted in the standard way, but at a point in its future development the text will be published on the web and licensed under the Libre Commons Res Communes License. See chapter 1 (introduction) for further explanation of this.

[16] See the web site (http://twenteenthcentury.com/uo/index.php), and for its Faculty of Unix, running since 2002 (http://darq.org.uk/FacultyUnix).

[17] For more on the Libre Commons License, see the web site (http://www.libresociety.org/library/libre.pl/Libre_Commons/). In a recent posting to the _nettime_ mail list, Cramer is scathing of this approach: 'This is a romantic apolitical position because such a space "outside of all legal jurisdictions" does not exist. Wake up and get a life.' Berry counters this with the

following: ‘Incidentally, you may be interested to know that law requires a state to enforce it, and, to the best of my knowledge, we do not *yet* have a global state, and consequently the spaces between nation states (such as the high seas) are not subject to law as such (rather international treaties which attempt to govern these ungovernable spaces).’ (2005)

[18] It was le Comte de Lautréamont, who in 1870 claimed: ‘Plagiarism is necessary. Progress implies it. It embraces an author’s phrase, makes use of his expressions, erases a false idea, and replaces it with the right idea.’ This was later plagiarised by Guy Debord as follows: ‘Plagiarism is necessary. Progress demands it. Staying close to the author’s phrasing, plagiarism exploits his expressions, erases false ideas, replaces them with correct ideas.’ (from ‘Negation and Consumption’, in _The Society of the Spectacle_, 1998: 145). Stewart Home further claims that this was wrongly attributed and Lautréamont plagarised the quote: ‘Old discoveries belong to those who put them to use’. Home has reworked it too as: ‘Progress is necessary. Plagiarism demands it’ (these quotes are left unreferenced in the spirit of their contents).

[19] Chainworkers.org’s slogan is ‘Chain and brainworkers unite’ (http://www.chainworkers.org/) referred to by Lazzarato (2003).

[20] This also accounts for further misconceptions such as Lunenfeld’s term ‘dialectical immaterialism’ (2002), to contribute to critical discussions about ‘technology untethered to the constraints of production’. As much as the phrase is evocative of the approach this thesis takes, Lunenfeld’s statement is a severe misunderstanding of the ways in which production has expanded to the whole of society, with cultural work thoroughly integrated in the social factory.

[21] However, Barbrook’s position should not be dismissed out of hand, as it is also one that responds critically to what he calls ‘the Californian ideology’ that typifies the combination of technological determinism and free market principles. The example is _Wired_ magazine, that reproduces an ideology based upon ‘Darwinian thinking and techno-mysticism’ according to Stallabrass (2003: 149).

[22] For example, Mauss refers to ‘potlatch’, a ceremonial feast at which possessions are give away or destroyed to display wealth or enhance status (1970: 5).

[23] The first Apple Macintosh and its ‘desktop’ graphical user interface was introduced in 1984. Bowles is referring to the Apple II. Apple remains the machine of choice in most ‘creative’ contexts. In the most recent reprint of his essay, Bowles adds some more recent reflection but in general finds its general argument holds (2005).

[24] In a similar way, Latour describes a situation where the seemingly impossible task of opening Pandora’s black box is made possible by experiencing technology at work, not ready-made - but ‘in action’ and ‘before the box closes and goes black’ (1999: 21). ‘Black box’ is a phrase from cybernetics, applied when a piece of machinery or a set of commands are too complex to be easily understood. This applies almost by default to software where the complex processes and actions are obscured.

[25] The online description reads: ‘Suicide Letter Wizard for Microsoft Word helps you to create a suicide letter according to your preferences. Use professional design. Choose from a variety of styles. Make your letter look great.’ (http://www.dxlab.org/slw/).

[26] OpenOffice.org is a multiplatform and multilingual office suite and an open-source project. Compatible with all other major office suites, the product is free to download, use, and distribute (http://www.openoffice.org/).

[27] This would be in keeping with the position of de Certeau, who asserted that users oppose established rules in the most ordinary of circumstances (1984). Through what he calls ‘antidiscipline’, consumers negotiate discipline and power exerted on them. By employing what he calls ‘tactical’ forms and ‘makeshift creativity’, consumers ‘make use of techniques for re-employment in which we can recognize the procedures of everyday practices. A politics of such ploys should be developed’ (1984: xxiv).

[28] The term ‘tactical media’ is variously defined,

but emerges from a group of media activists in Rome in 1996. Lovink's involvement in tactical media emerges from the _Next 5 Minutes_ festival (which began in 1993) and other collaborative writings; for instance 'The ABC of Tactical Media' (1997) with David Garcia, and 'New Rules for the New Actonomy' (2001) with Schneider. For Critical Art Ensemble, the concept is a way of avoiding the 'dense arcane style of the Frankfurt Institut' (2002: 27), and a way of asserting difference from avant-garde practices for 'electronic civil disobedience' (2002: 13).

[29] For example, this 'distributed-denial-of-service' was used by the Zapatistas against the Mexican government and against the WTO at the time of Seattle in 1999 (Medosch 2003: 17).

[30] Or, as Eben Moglen puts it: 'A spectre is haunting multinational capitalism - the spectre of free information.' (2003: 216) Moglen is a lawyer who has contributed to the development of the General Public License (GPL) with Stallman. His full parody continues: 'All the powers of "globalism" have entered into an unholy alliance to exorcize this spectre: Microsoft and Disney, the World Trade Organization, the United States Congress and the European Commission. Where are the advocates of freedom in the new digital society who have not been decried as pirates, anarchists, communists? Have we not seen that many of those hurling the epithets were merely thieves in power, whose talk of "intellectual property" was nothing more than an attempt to retain unjustifiable privileges in a society irrevocably changing? But it is acknowledged by all the Powers of Globalism that the movement for freedom is itself a Power, and it is high time that we should publish our views in the face of the whole world, to meet this nursery tale of the Spectre of Free Information with a Manifesto of our own.' (2003: 216) Also making explicit reference to the _The Communist Manifesto_, Wark claims that what now haunts the world is the spectre of 'hacking'. However, he claims his manifesto is neither an orthodox Marxist tract nor post-Marxist repudiation, but a 'crypto-Marxist reimagining of the materialist method for practising theory within history' (2004: 024).

[31] The phrase 'precarious labour' has become increasingly

popular in the activist community to describe the material reality of intermittent and irregular work that 'teeters' on the edge of moral acceptability and the ability to generate a living wage, although it should be noted that it is not labour in itself that is precarious but the 'technical and cultural conditions in which info-labour' finds itself (Beradi 2005).

[32] Negri's position on Spinoza is developed in his essay 'The Savage Anomaly' of 1980.

[33] For more on this, see Reiner Schürmann, _Des hégémonies brisées_, Mouvezin: T.E.R., 1996 (in Hardt & Negri 2000: 389). The term 'corruption' is borrowed from Aristotle's _De generatione et corruptione_ (1982), again cited in _Empire_.

[34] The dialectical operation between states of order and disorder also underpins Signwave's _Anagrammar_ (2001), an unruly version of Microsoft Powerpoint or Apple Keynote (produced to accompany the essay 'The Aesthetics of Generative Code' for conference presentation, Cox, McLean & Ward 2001). Whilst presenting text slides on screen, it 'listens' to the current sound input source, and when a sound occurs, starts to jumble up the letters of the current slide. When the sound falls below an ambient level, the letters are rearranged back into their original order. Both examples demonstrate a dialectical play between two interconnected states of order and disorder, between generation and corruption, suggesting the potential for transformation. In the context of this thesis, the examples offer a dialectical approach that responds to an understanding of complexity theory (as argued at the end of chapter 4).

[35] Negri's negative view of socialism is perhaps informed by the various failed examples of 'real existing socialism' observed at the time of writing in 1985. With Hardt, he charts the tragic irony in that nationalist socialism comes to resemble national socialism, because the same machine of national sovereignty lies behind the logic of both. As a result, they maintain 'we are not anarchists but communists who have seen how much repression and destruction of humanity have been wrought by liberal and socialist big governments. We have seen how this is being re-

created in imperial government, just when the circuits of productive cooperation have made labour as a whole capable of constituting itself as a government' (2000: 350).

[36] Pasquinelli identifies action related to labour, politics and art, as integrated into each other, making everyone 'workers-artists-activists' (2005: 2).

[37] The Situationist International also made much of this strategy of refusal in the May '68 uprisings in announcing 'Don't Work!' and 'Never Work!' (Ford 2005: 119 & 123). The Situationist refusal to work is paralleled by the Neoist 'artstrike' calling on cultural workers to stop making or discussing their work from 1990 to 1993 – although this is simply plagiarising Metzger's 1974 proposal for an Art Strike, according to Home (1993). In the context of performance art, refusing to work can be a provocative action, such as the example of Roy Varra who simply stood in Tianneman Square, and although doing nothing, was arrested.

[38] Deleuze explains that a sabot was a worker's wooden clog. In the context of programming, 'deprogramming' is one example of calling to attention the structures and standard formats of software. This strategy makes reference to the Situationist 'détournement' of technology.

CHAPTER 6:

[1] Lévi Strauss's _The Raw and the Cooked_ (1970) provides a further reference in which the 'raw' associated with nature is opposed to the 'cooked' associated with culture. His approach is structuralist anthropology drawing upon semiotics, where the raw 'signifier' enters into the realm of the 'signified' when cooked. The analogy between recipes and source code is further explored in the barszcz source code repository that includes Jaromil's string based cooking (http://www.barszcz.net/).

[2] To Adorno and Horkheimer, the analogy to the production of food also reveals that: 'the culture industry perpetually cheats its consumers of what it perpetually promises[...] that the diner must be satisfied with the menu' (1997: 139). What is on offer is bad for the digestion.

[3] The artist-programmer Mark Napier says much the same: 'In the software industry the code is very valuable since it contains the knowledge, recipe or blueprint of how the software product is made. The binary "executable" is distributed to the world, but the source code is carefully guarded. As an artist I'm happy to share most of my source code with other artists.[...] Whoever owns the source code in effect "owns" the artwork.' (2000)

[4] Socialfiction.org's _.walk_ won the first prize in the software art category at transmediale in 2004 (see Cox, Reas & Rich 2003). A simple stroll algorithm follows: '// Classic.walk; Repeat { 1st street left; 2nd street right; 2nd street left }'. This is both clearly understandable even to the non-specialist and wildly unpredictable in its outcomes.

[5] Unix is open in the broadest sense in that its API (application programming interface) works across different computer platforms. Most servers rely on Unix, and it underpins the Internet protocol of TCP/IP.

[6] Elsewhere Lovink charts this crisis of the intellectual, tracing Gramsci's idea of the 'organic intellectual interfacing with ordinary people to the contemporary distrust of the concept of the intelligentsia in the post-political era. In the knowledge economy, the intellectual has become a faceless professional, and sadly lacks a public role in society. Accordingly, the suggestion is that the link between the intellectual and the public might be forged in virtual space - the 'virtual intellectual' (2002: 30). This might be wishful thinking, but expresses the potential for a new kind of collective engagement with ideas in keeping with a re-engagement with the Internet as public sphere (located in the sphere of the negative as Lovink puts it). Rejecting the 'free-market way of thinking' the virtual intellectual is more of a 'free-floating' knowledge worker (a less aloof term) who engages with other workers and is 'always under construction' (2002: 38-9). This also emphasises Virno's point referred to in the previous chapter in relation to general intellect - and the importance of the public sphere in generating positive potential.

[7] This description is adapted from the previous collaborative paper 'Coding Praxis' (Cox, et al 2004).

[8] For instance, the work of toplap (http://www.toplap.org/) who perform music using live coding and display their desktop screens in the spirit of transparency of process (Collins et al 2003). This is not intentionally a politicised practice at all (and consequently suffers from the problem of virtuosity as an individualised display of skill), but holds the potential to be a critical practice in the sense this section describes.

[9] Or, actions and events determine words. This is the irony of Bruno's _Human Browser_ (2006) mentioned previously in this thesis.

[10] Although it should be noted that Virno argues the opposite to Arns, in claiming that it is not the parole but the langue which is mobilised (2004: 91).

[11] The distinction between work and labour is hard to fathom, as both words broadly refer to the same thing. Arendt quotes John Locke: 'the labor of our body and the work of our hands' (2000: 170). She adds that most European languages make similar distinctions: 'arbeiten' and 'werken' in German; 'laborare' and 'fabricari' in Latin; 'ponein' and 'ergazesthai' in Greek. It seems that the human body is given over to labour, the reproductive process, the biological and the link to the human organism (even the pains of birth are associated of course). Thus labouring is tied more closely to the cycles of life itself, as it 'corresponds to the condition of life itself' and lasting happiness and contentment lies in 'painful exhaustion and pleasurable regeneration' (Arendt 2000: 172).

[12] This position is developed in Virno's 'Virtuosity and Revolution: The Political Theory of Exodus' (1996: 188).

[13] Praxis is clearly an important issue in Marxist philosophy. Lefebvre explains that human creation can be explained as praxis in which humans transform nature through 'the unity of the sensuous and the intellectual, of nature and culture' (1968: 39).

[14] The issue of the Internet as an extension of the public sphere makes reference to Habermas's _The Structural Transformation of the Public Sphere_ (1985) and texts such as Mark Poster's 'Cyberdemocracy:

Internet and the Public Sphere’ (1997).

[15] This approach to a conclusion is inspired by Virno’s _A Grammar of the Multitude_ (2004).

7.2 bibliography

0100101110101101.ORG (2001) ‘Data-Nudism’, an interview with Matthew Fuller about life_sharing, _Nettime_, April 14, http://www.nettime.org [first published by Gallery 9 / Walker Art Centre, http://www.walkerart.org/gallery9/lifesharing/] [last accessed 31 Dec 2005].

Adilkno (1998) ‘What is Data Criticism?’, in _Media Archive_, New York: Autonomedia, pp. 57-59.

Theodor W. Adorno (1991) ‘On the Fetish Character in Music and the Regression of Listening’, in J.M. Bernstein, ed. _The Culture Industry: Selected Essays on Mass Culture_, London: Routledge, pp. 26-52.

Theodor Adorno & Max Horkheimer (1997 [1944]) _Dialectic of Enlightenment_, trans. John Cumming, London: Verso.

Theodor W. Adorno (2000 [1966]) _Negative Dialectics_, trans. E. B. Ashton, London: Routledge.

Saul Albert (2002) ‘Useless Utilities’, in Signwave, _Auto-Illustrator Users Guide_, Plymouth/Exeter: i-DAT/Spacex, pp. 89-99.

Amy Alexander (2001) ‘Re: Hackers: the political heroes of cyberspace’ _Nettime_, March 16, http://www.nettime.org [last accessed 31 Dec 2005].

Louis Althusser (1997 [1969]) ‘Ideology and Ideological State Apparatuses: Notes Toward an Investigation’, in Slavoj Žižek, ed. _Mapping Ideology_, London: Verso, pp. 100-140.

Hannah Arendt (1999) ‘Introduction: Walter Benjamin: 1892-1940’, in Walter Benjamin, _Illuminations_, London: Pimlico, pp. 7-55.

Hannah Arendt (2000) ‘Labor, Work, Action’ [from a

Lecture 1964], in _The Portable Hannah Arendt_, New York: Penguin, pp. 167-181.

Aristotle (1982) _De Generatione et Corruptione_, trans. C.J.F. Williams, Oxford: Oxford University Press.

John Armitage (2002) 'Resisting the Neoliberal Discourse of Technology: The Politics of Cyberculture in the Age of the Virtual Class', http://www.textz.com [last accessed 31 Dec 2005].

Inke Arns (2004) 'Read_Me, Run_Me, Execute_Me: Software and its Discontents, or: it's The Performativity of Code, Stupid', in Olga Goriunova & Alexei Shulgin, eds. _Read_Me: Software Art & Cultures_, Arhus: DARC, pp. 176-193.

Antonin Artaud (2001 [1964]) _The Theatre and its Double_, [first published as _Oeuvres Completes_] trans. Victor Corti, London: Calder.

Roy Ascott (2003) _Telematic Embrace: Visionary Theories of Art, Technology, and Consciousness_, Berkeley: University of California Press.

Roy Ascott (2004) 'Orai, or How the Text Got Pleated: A Genealogy of La Plissure du Texte: A Planetary Fairytale,' in _Leonardo_, vol. 37, no. 3, pp. 195-200.

John Langshaw Austin (1962) _How to Do Things with Words_, Cambridge: Harvard University Press.

Albert-László Barabási (2002) _Linked: The New Science of Networks_, Cambridge, Mass.: Perseus.

Richard Barbrook & Pit Schultz (1997) 'The Digital Artisans Manifesto', http://www.hrc.wmin.ac.uk/hrc/theory/digitalartisans/t.1.1.html [last accessed 31 Dec 2005].

Richard Barbrook (1999) 'The High-Tech Gift Economy', in Josephine Bosma, et al, eds. _Readme! Filtered by Nettime. ASCII Culture and the Revenge of Knowledge_, New York: Autonomedia.

John D. Barlow (2001) _The Book of Nothing_, London: Vintage.

Anne Barron (2002), ‘The Legal Properties of Art’, conference paper, _Marxism and the Visual Arts Now_, University College London.

Roland Barthes (1977) ‘The Death of the Author’, in _Image Music Text_, trans. Stephen Heath, London: Fontana, pp. 142-148.

Roland Barthes (1975 [1970]) _S/Z: An Essay_, trans. Richard Miller, London: Cape.

Geoffrey Batchen (2004) ‘Electricity Made Visible’, in Lucy Kimbell, ed. _New Media Art: Practice and Context in the UK 1994-2004_, London: Arts Council of England with Cornerhouse, pp. 26-44.

Gregory Bateson (2000 [1971]) _Steps to an Ecology of Mind_, Chicago/London: University of Chicago Press.

Catherine Belsey (1992), ‘Towards a Productive Critical Practice’, in Critical Practice, London: Routledge, pp. 125-146.

Walter Benjamin (1992a [1931]) ‘A Small History of Photography’, in _One Way Street and Other Writings_, trans. Edmund Jephcott & Kingsley Shorter, London: Verso.

Walter Benjamin (1992b [1934 written]) ‘The Author as Producer’, in _Understanding Brecht_, trans. Anna Bostock, (first published as _Versuche über Brecht_) London: Verso.

Walter Benjamin (1996) _Selected Writings: Volume 1, 1913-1926_, Marcus Bullock and Michael W. Jennings, eds. Cambridge, Mass.: Belknap Press.

Walter Benjamin (1999a) _The Arcades Project_, trans. Howard Eiland & Kevin McLaughlin [first written as _Das Passegen-Werk_], Cambridge, Mass.: Belknap Press.

Walter Benjamin (1999b) Selected Writings: Volume 2, 1927-1934, trans. Rodney Livingstone et al, Michael W. Jennings, Howard Eiland & Gary Smith, eds. Cambridge, Mass.: Belknap Press.

Walter Benjamin (1999c) ‘Theses on the Philosophy of

History’ [written 1940, first published 1950], trans. Harry Zohn, in _Illuminations_, London: Pimlico, pp. 245-258.

Walter Benjamin (1999d [1936 written]) ‘The Work of Art in the Age of Mechanical Reproduction’, trans. Harry Zohn, in _Illuminations_, London: Pimlico, pp. 211-244.

Max Bense (1971) ‘The projects of generative aesthetics’, in Jasia Reichardt, ed. _Cybernetics, Art and Ideas_, NY: Graphics Society Limited.

Franco Beradi [Bifo] (2005) ‘ Info-Labour and Precarisation’, trans. Erik Empson (http://www.generation-online.org/t/tinfolabour.htm), [first published in Italian, http://www.rekombinant.org/print.php?sid3D2578] [last accessed 31 Dec 2005].

John Berger (1972) _Ways of Seeing_, London: Penguin/BBC.

John Berger (1980) ‘Why Look at Animals?’, in _About Looking_, London: Writers & Readers.

Marshall Berman (1999 [1982]) _All That Is Solid Melts Into Air: the Experience of Modernity_, London: Verso.

Josephine Berry Slater (2005 [2002]) ‘Bare Code: Net Art and the Free Software Movement’, in Geoff Cox & Joasia Krysa, eds. _Engineering Culture_, New York: Autonomedia, pp. 133-149.

David M. Berry & Giles Moss (2004) _Libre Culture Manifesto_ [Version 1. 62], http://libresociety.org/ [last accessed 31 Dec 2005].

David M. Berry (2005) ‘Re: <nettime> Libre Commons = Libre Culture + Radical Democracy’, _Nettime_ thread [in reply to Florian Cramer, 08 Dec], http://www.nettime.org/ [last accessed 31 Dec 2005].

Hakim Bey (2003 [1985]) _T.A.Z.: The Temporary Autonomous Zone, Ontological Anarchy, Poetic Terrorism_, New York: Autonomedia.

Homi Bhabha (1994) ‘The Commitment to Theory’, in _The Location of Culture_, London: Routledge.

Roy Bhaskar (1986) _Scientific Realism and Human Emancipation_, London: Verso.

Roy Bhaskar, Andrew Collier and Alan Norrie (1998) 'Dialectic and Dialectical Critical Realism' section, in Margaret Archer, Roy Bhaskar, Andrew Collier, Tony Lawson, Alan Norrie, eds. _Critical Realism: Essential Readings_, London: Routledge, pp. 559-739.

Friedrich W. Block, Christiane Heibach, Karin Wenz, eds. (2004) _The Aesthetics of Digital Poetry_, Ostfildern: Hatje Cantz.

Friedrich W. Block (2004) 'From Code to Screening and Vice Versa: Orientation in Digital Poetics between Concept and Perception' lecture notes, 'From Software to Software Art' symposium, _transmediale_, Berlin, February.

Maurizio Bolognini (2004), 'Programmed Machines: Infinity and Identity', in _Generative Art 03_, international conference, Politecnico di Milano, Italy, http://www.generativeart.com/papersGA2004/b9.htm [last accessed 31 Dec 2005].

Jay David Bolter (1984) _Turing's Man: Western Culture in the Computer Age_, Chapel Hill: University of North Carolina Press.

Jay David Bolter & Richard Grusin (1999) _Remediation: Understanding New Media_, Cambridge, Mass.: MIT Press.

Simon Bone & Mathias Castro (1997) 'A Brief History of Quantum Computing', http://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol4/spb3/ [last accessed 31 Dec 2005].

Pierre Bourdieu (1984 [1979]) _Distinction: A Social Critique of the Judgement of Taste_, trans. Richard Nice, London: Routledge.

Pierre Bourdieu (1993) 'The Field of Cultural Production, or: The Economic World Reversed', in Randall Johnson, ed. _The Field of Cultural Production: Essays on Art and Literature_, London: Polity Press, pp. 29-73.

Nicolas Bourriaud (2002) _Relational Aesthetics_, trans.

Simon Pleasance & Fronza Woods, Dijon-Quetigny: Les Presses de Réel.

William Bowles (2005 [1987]) 'The Macintosh Computer: Archetypal Capitalist Machine?', in Geoff Cox & Joasia Krysa, eds. _Engineering Culture_, New York: Autonomedia, pp. 39-61.

Stuart Brisley (2003) _Beyond Reason: Ordure_, London: Book Works.

Andreas Broeckmann (1997) in conversation with Ken Wark, 'Machine Aesthetics', _Rhizome_, http://www.rhizome.org/object.rhiz?439 [last accessed 31 Dec 2005].

Andreas Broeckmann (2000) 'Sociable Machinists of Culture', http://www.v2.nl/~andreas/texts/2000/networkers.html [last accessed 31 Dec 2005].

Andreas Broeckmann & Susanne Jaschko, eds. (2001) _DIY Media - Art and Digital Media: Software - Participation - Distribution_, festival catalogue, Berlin: transmediale 01, Podewil.

Andreas Broeckmann (2003) 'Notes on the Politics of Software Culture,' _Nettime_, 4 September [for the _Next5Minutes4 Reader_], http://www.nettime.org/ [last accessed 31 Dec 2005].

Andreas Broeckmann (2004) 'Questioning Software Art', _Programmation Orientée Art_, colloque organised by David-Olivier Lartigaud & Anne-Marie Duguet, CRECA, Université Paris, Sorbonne, 19/20 March.

Andreas Broegger (2003a) 'Gigliotti on "(radical) software"', as part of software art thread, _Rhizome_ (via Andreas Broeckmann) 10 Oct, http://www.rhizome.org/ [last accessed 31 Dec 2005]

Andreas Broegger (2003b) 'Software Art - an introduction', in _Artificial.dk_, http://www.artficial.dk/articles/software.htm [last accessed 31 Dec 2005]

Paul Brown, ed. (2003) 'Generative computation and the arts', in _Digital Creativity_, vol. 14, no.1, Lisse: Swets & Zeitlinger.

Wendy Brown (1996) _States of Injury_, Stanford: Stanford University Press.

Susan Buck-Morss (1995) _The Dialectics of Seeing: Walter Benjamin and The Arcades Project_, Cambridge, Mass.: MIT Press.

Peter Bürger (1984) _Theory of the Avant Garde_, trans. Michael Shaw, Minneapolis: University of Minnesota Press.

Seán Burke (1992) _The Death & Return of the Author: criticism and Subjectivity in Barthes, Foucault and Derrida_, Edinburgh: Edinburgh University Press.

Jack Burnham (1968a) _Beyond Modern Sculpture: the Effects of Science and Technology on the Sculpture of this Century_, New York: George Braziller.

Jack Burnham (1968b) 'Systems Esthetics', in _Artforum_, vol. 7 no. 1, September.

Alex Callinicos (2002) 'Toni Negri and Michael Hardt's Empire', _Marxism 2002_ conference presentation, July, London.

Italo Calvino (1995) 'How I Wrote One of My Books', in _Oulipo Laboratory_, trans. Iain White, London: Atlas.

Carbon Defense League & Conglomco Media Conglomeration (2004) 'Recode.com' in Geoff Cox, Joasia Krysa & Anya Lewin, eds. _Economising Culture_, New York: Autonomedia, pp. 111-119.

Manuel Castells (1996) _The Rise of the Network Society_, (Volume 1 of _The Information Age: Economy, Society and Culture_), Oxford: Blackwell.

Michel de Certeau (1984) 'General Introduction', _The Practice of Everyday Life_, trans. Steven F. Rendail, Berkeley: University of California Press, pp. xi-xxiv.

Paul E. Ceruzzi (2003 [1998]) _A History of Modern Computing_, Cambridge, Mass.: MIT Press.

Noam Chomsky (1972 [1957]) _Syntactic Structures_, The Hague: Mouton.

Nick Collins, Alex McLean, Julian Rohrhuber, & Adrian Ward (2003) 'Live Coding in Laptop Performance', in _ Organised Sound_, 8 (3), Cambridge University Press, pp. 321-329.

Michael Connor, ed. (2004) _Jodi: Computing 101B_, Liverpool: FACT.

Geoff Cox & Adrian Ward (2007) 'Perl', in Matthew Fuller, ed. _Software Studies_, Cambridge, Mass.: MIT Press.

Geoff Cox & Adrian Ward (2005) 'Why Look at Artificial Animals?', in Roy Ascott, ed. _Engineering Nature_, Bristol: Intellect, pp. 115-119; derived from conference paper, _Consciousness Reframed 03_, University College Newport, Wales, 2003.

Geoff Cox, Alex McLean & Adrian Ward (2004) 'Coding Praxis: Reconsidering the Aesthetics of Generative Code', in Olga Goriunova & Alexei Shulgin, eds., _Read_me: Software Art & Cultures_, Arhus: DARC, pp. 161-174.

Geoff Cox, Casey Reas & Kate Rich (2003) 'Software Art' jury statement, _transmediale 04_, Berlin.

Geoff Cox, Alex McLean & Adrian Ward (2001) 'The Aesthetics of Generative Code', in Eugene Thacker, ed. _Hard_Code: narrating the network society_, Boulder, CA: Alt-X Press; derived from _Generative Art 00_, conference paper, Politecnico di Milano.

Geoff Cox & Tim Brennan (2000) 'Manifest: Reframing False Consciousness', conference paper, _Consciousness Reframed 00_, conference paper, University College Newport, Wales.

Geoff Cox & Adrian Ward (1999) 'The Authorship of Generative Art', in Kestutis Andrasiunas, ed. _.agon_ [dotagon] online journal, Lithuania, http://www.o-o.lt/agon/; previously _Generative Art 99_ conference, Politecnico di Milano.

Florian Cramer & Ulrike Gabriel (2001) 'Software Art' jury text, in Andreas Broeckmann & Susanne Jaschko, eds. _DIY Media - Art and Digital Media: Software - Participation - Distribution_, Berlin: transmediale 01 festival catalogue, pp. 29-33.

Florian Cramer (2001) ‘On Literature and Systems Theory’, lecture notes from Tate Modern, April 8, http://cramer.plaintext.cc:70/all/literature_and_systems_theory/literature_and_system_theory.html [last accessed 31 Dec 2005].

Florian Cramer (2002a) ‘Concepts, Notations, Software Art’, in Olga Goriunova & Alexei Shulgin, eds. ‘Software Art: Thoughts’, _Read_me festival 1.2_, catalogue, Moscow: Rosizo, State Centre for Museums and Exhibitions, pp. 18-24.

Florian Cramer (2002b), ‘Discordia Concors: www.jodi.org’, written for jodi exhibition catalogue _jodi_anti_net_art.tex_, _Eu-gene_ posting, Nov 4, http://www.generative.net/eu-gene/ [last accessed 31 Dec 2005].

Florian Cramer (2003) ‘Exe.cut[up]able statements: the Insistence of Code’, in Gerfried Stocker & Christine Schöpf, eds. _Code - The Language of Our Time_, Ars Electronica, Linz: Hatje Cantz, pp. 98-103.

Florian Cramer (2005) _Words Made Flesh, Code, Culture, Imagination_, Rotterdam: Piet Zwart Institute, http://pzwart.wdka.hro.nl/mdr/research/fcramer/wordsmadeflesh/ [last accessed 31 Dec 2005].

Critical Art Ensemble (2002) _Digital Resistance: Explorations in Tactical Media_, New York: Autonomedia.

Sean Cubitt (1998) _Digital Aesthetics_, London: Sage.

Sean Cubitt (1999) ‘Orbis Tertius’, in _Third Text_, 47, Summer, London: Kala Press. pp. 3-10.

Guy Debord (1998 [1967]) _The Society of the Spectacle_, trans. Donald Nicholson-Smith, New York: Zone Books.

Gilles Deleuze & Félix Guattari (1987) _A Thousand Plateaus_, trans. Brian Massumi, Minneapolis: University of Minnesota Press.

Gilles Deleuze & Félix Guattari (1990 [1972]) ‘The Desiring Machines’, in _Anti-Oedipus: Capitalism and Schizophrenia_, [L’Anti-Oedipe] trans. Robert Hurley et al, London: Athlone, pp. 1-50.

Gilles Deleuze & Antonio Negri (1990) _Communists Like Us_, New York: Semiotext(e).

Gilles Deleuze (1990) ‘Control and Becoming’, conversation with Antonio Negri, in _Futur Anterieur_, trans. Martin Joughin, http://www.generation-online.org/p/fpdeleuze3.htm [last accessed 31 Dec 2005].

Jacques Derrida (1994) _Spectres of Marx: The State of the Debt, the Work of Mourning, & the New International_, trans. Peggy Kamuf, London: Routledge.

Arif Dirlik (1997) _The Postcolonial Aura: Third World Criticism in the Age of Global Capitalism_, Boulder: Westview Press.

Nick Dyer-Witheford (1999) _Cyber-Marx: Cycles and Circuits of Struggle in High-Technology Capitalism_, Urbana & Chicago: University of Illinois Press.

Nick Dyer-Witheford (2005) ‘Cognitive Capitalism and the Contested Campus’, in Geoff Cox & Joasia Krysa, eds. _Engineering Culture_, New York: Autonomedia, pp. 71-93.

Terry Eagleton (1997) ‘Ideology and its Vicissitudes in Western Marxism’ in Slavoj Žižek, ed. _Mapping Ideology_, London: Verso, pp. 179-226.

Ernest Edmonds & Mike Stubbs, eds. (2005) _White Noise_, Melbourne: Australian Centre for the Moving Image.

Sergei Eisenstein (1949 [1929 written]) ‘A Dialectical Approach to Film Form’, in Jay Leyda, ed., _Film Form_, New York: Harcourt, Brace and Company.

Friedrich Engels (1978 [1845 written]) ‘Working-Class Manchester’, in Robert Tucker, ed., _The Marx-Engels Reader_, New York: Norton, pp. 579-585.

Friedrich Engels (1980 [1875-6 written]) ‘Introduction to Dialectics of Nature’, in _Karl Marx and Frederick Engels: Selected Works in One Volume_, London: Lawrence and Wishart.

Micz Flor (2002) ‘”Hear me out” - Free Radio Linux’, _Nettime_, 12 May http://radioqualia.va.com.au/freeradiolinux/ [last accessed 31 Dec 2005].

Luciano Floridi (1999) _Philosophy and Computing: an introduction_, London: Routledge.

Simon Ford (2005) _The Situationist International: A User's Guide_, London: Black Dog.

Hal Foster (1996) 'The Artist as Ethnographer' & 'Whatever Happened to Postmodernism?', in _The Return of the Real_, Cambridge, Mass.: MIT Press, pp. 171-204 & 205-226.

Michel Foucault (1991) 'What is an Author?', in Paul Rabinow, ed. _The Foucault Reader: an Introduction to Foucault's Thought_, trans. Josué V. Harari, London: Penguin.

Francis Fukuyama (1992) _The End of History and the Last Man_, New York: The Free Press.

Matthew Fuller (2003) _Behind the Blip: essays on the Culture of Software_, New York: Autonomedia.

Matthew Fuller (2004) 'The Digital Object', in Olga Goriunova & Alexei Shulgin, eds. _Read_Me: Software Art & Cultures - Edition 2004_, Arhus: DARC, pp. 26-41.

Philip Galanter (2003) 'What is Generative Art? Complexity Theory as a Context for Art Theory', _ Generative Art 03_, international conference, Politecnico di Milano, Italy, http://www.generativeart.net [last accessed 31 Dec 2005].

Philip Galanter (2004) 'Generative art is as old as art', an interview with Thomas Petersen & Kristine Ploug, http://www.artificial.dk/articles/galanter.htm [last accessed 31 Dec 2005].

Alex R. Galloway (2004) _Protocol: How Control Exists After Decentralization_, Cambridge, Mass.: MIT Press.

Vincent Geoghegan (1981) _Reason & Eros: The Social Theory of Herbert Marcuse_, London: Pluto.

Charlie Gere (2005) 'Jack Burnham and the Work of Art in the Age of Real Time Systems', in Morten Sondergaard, ed. _Get Real_, The Museum of Contemporary Art in Roskilde: Informations Forlag, pp. 149-163.

Charlie Gere, ed. (2006) _White Heat, Cold Logic_, Birkbeck College & MIT Press.

James Gleick (1998 [1987]) _Chaos: The Amazing Science of the Unpredictable_, London: Vintage.

Gerrit Gohlke, ed. (2003) _Software Art: Eine Reportage über den Code/A Reportage about Source Code_, Berlin: Media Arts Lab at Künsterhaus Bethanien.

Brian Goodwin (1997) 'Complexity, Creativity, and Society', in _Soundings: Media Worlds_, Issue 5, Spring, pp. 111-122.

Olga Goriunova & Alexei Shulgin (2002) 'Artistic Software for Dummies, and, by the way, Thoughts about the New World Order', in Goriunova & Shulgin, eds. _Software Art: Thoughts_, Read_me festival 1.2, catalogue, Moscow: Rosizo, State Centre for Museums and Exhibitions, pp. 6-9.

Olga Goriunova & Alexei Shulgin, eds. (2003) _Read_ Me 2.3 Reader: about software art_, Helsinki: NIFCA publication 25.

Olga Goriunova (2004) 'Runme.org Repository: What you believe is what you get', symposium paper, _ Programmation Orientée Art_, organised by David-Olivier Lartigaud & Anne-Marie Duguet, CRECA, Université Paris, Sorbonne, 19/20 March.

Olga Goriunova & Alexei Shulgin, eds. (2004) _Read_Me: Software Art & Cultures - Edition 2004_, Arhus: Digital Aesthetic Research Center, Aarhus University.

Antonio Gramsci (1988) _A Gramsci Reader_, ed. David Forgacs, London: Lawrence and Wishart.

Steve Grand (2000) _Creation: life and how to make it_, London: Phoenix.

Clement Greenberg (1992) 'Modernist Painting', in Francis Frascina & Jonathan Harris, eds. _Art in Modern Culture: An Anthology of Critical Texts_, London: Phaidon, pp. 308-14.

Félix Guattari (1995) _Chaosophy_, Sylvere Lotringer, ed., New York: Semiotext(e).

Jürgen Habermas (1984) _Theory of Communicative Action_, trans. Thomas McCarthy, Boston: Beacon Press.

Jürgen Habermas (1989) _The Structural Transformation of the Public Sphere: An Inquiry into a Category of Bourgeois Society_, trans. Thomas Burger, Cambridge: Polity.

Jürgen Habermas (1991 [1980]) 'Modernity - An Incomplete Project' [first written as a talk in receipt of the Theodor Adorno prize by the city of Frankfurt and later published under the title 'Modernity versus Postmodernity'], in Hal Foster, ed. _Postmodern Culture_, London: Pluto Press.

Donna Haraway (1991) 'The Cyborg Manifesto: Science, Technology, and Socialist-Feminism in the Late Twentieth Century', in _Simians, Cyborgs and Women: the reinvention of nature_, London: Free Association, pp. 149-181.

Michael Hardt & Antonio Negri (2000) _Empire_, Cambridge Mass.: Harvard University Press.

Terence Hawkes (1986 [1977]) _Structuralism and Semiotics, London: Methuen.

N. Katherine Hayles (1989) 'Chaos as Orderly Disorder: Shifting Ground in Contemporary Literature and Science', _New Literary History_, 20, pp. 305-322.

N. Katherine Hayles (1991) 'Complex Dynamics in Literature and Science', in Hayles, ed. _Chaos and Order_, Chicago: University of Chicago Press.

N. Katherine Hayles (2002) _Writing Machines_, Cambridge, Mass.: MIT Press.

Georg W. F. Hegel (1953) _Reason in History: A General Introduction to the Philosophy of History_, trans. R.S. Hartman, New York: Library of Liberal Arts.

Georg W. F. Hegel (1967 [1807]) _The Phenomenology of Mind/Spirit_, trans. J.B. Baillie, New York: Harper & Row.

Georg W. F. Hegel (1969) _Hegel's Science of Logic_, vol. 1, trans. A.V. Miller, London: Allen & Unwin.

Georg W. F. Hegel (1993 [1823]) _Introductory Lectures on Aesthetics_, trans. B. Bosanquet, London: Penguin.

Michael Heim (2000) 'The Cyberspace Dialectic', in Peter Lunenfeld, ed. _The Digital Dialectic: New Essays on New Media_, Cambridge, Mass.: MIT Press.

T. F. Hoad, ed. (1993) _The Concise Oxford Dictionary of English Etymology_, Oxford: Oxford University Press.

Quintin Hoare & Geoffrey Nowell-Smith (1971) _Selections from the Prison Notebooks of Antonio Gramsci_, London: Lawrence & Wishart.

Eric Hobsbawm (1998) _Behind The Times: The Decline and Fall of the Twentieth-Century Avant-Gardes_, London: Thames and Hudson.

Douglas Hofstadter (1985 [1981]) 'A Coffee House Conversation on the Turing Test', in _Metamagical Themas_, New York: Basic Books, pp. 492-525; also http://www.cs.unr.edu/~sushil/class/ai/papers/coffeehouse.html [last accessed 31 Dec 2005].

Douglas Hofstadter (2000 [1979]) _Gödel, Escher, Bach: an Eternal Golden Braid_, London: Penguin.

Brian Holmes (2002) _Hieroglyphs of the Future_, Zagreb: Arkzin.

Brian Holmes (2003), 'Artistic Autonomy and the communication society', _Nettime_, Oct 26; conference paper for _Diffusion: Collaborative Practice in Contemporary Art_, Tate Modern, London.

Stewart Home (1993) 'Assessing the Artstrike 1990-1993', http://www.thing.de/projekte/7:9%23/y_Assessing_the_Art_Strike.html [last accessed 31 Dec 2005].

Richard Huelsenbeck, ed. (1998 [1920]) _DADA Almanach_, trans. Malcolm Green, et al, London: Atlas Press.

Erkki Huhtamo (2003) 'Web Stalker Seek Aaron: Reflections on Digital Arts, Codes and Coders', in Gerfried Stocker

& Christine Schöpf, eds. _Code - The Language of Our Time_, Ars Electronica, Linz: Hatje Cantz, pp. 110-118.

The Institute for Applied Autonomy (2005) ‘Engaging Ambivalence: Interventions in Engineering Culture’, in Geoff Cox & Joasia Krysa, eds. _Engineering Culture_, New York: Autonomedia, pp. 95-105.

Frederic Jameson (1991 [1984]) ‘The Cultural Logic of Late Capitalism’, in _Postmodernism, or, The Cultural Logic of Late-Capitalism_, London: Verso, pp. 1-54; also _New Left Review_, no. 146, pp. 59-92.

Martin Jay (1996 [1973]) _The Dialectical Imagination: A History of the Frankfurt School and the Institute of Social Research 1923-1950_, London: University of California Press.

Chris Jenks (1993) ‘Introduction: the Analytic Bases of Cultural Reproduction Theory’, in Jenks, ed. _Cultural Reproduction_, London: Routledge, pp. 1-16.

Troels Degn Johansson (2004) ‘Mise en Abyme in Software Art: A Comment to Florian Cramer’, in Olga Goriunova & Alexei Shulgin, eds. _Read_Me: Software Art & Cultures - Edition 2004_, Arhus: DARC, pp. 150-159.

Randall Johnson (1993) ‘Editor’s Introduction: Pierre Bourdieu on Art, Literature and Culture’, in Pierre Bourdieu, _The Field of Cultural Production_, Cambridge: Polity, pp. 1-25.

Steven Johnson (2001) _Emergence: The Connected Lives of Ants, Brains, Cities and Software_, London: Penguin.

Alain Joxe (2002) _Empire of Disorder_, Los Angeles/New York: Semiotext(e).

Alan C. Kay (1984) ‘Computer Software’, in _Scientific American_, 251(3), September, pp. 41-47.

Kevin Kelly (2003 [1994]) _Out of Control: The New Biology of Machines, Social Systems, and the Economic World_, http://www.kk.org/outofcontrol/ [last accessed 31 Dec 2005].

Sarah Kember (2003) _Cyberfeminism and Artificial Life_, London: Routledge.

Lucy Kimbell, ed. (2004) _New Media Art: Practice and Context in the UK 1994-2004_, London: Arts Council of England, with Cornerhouse.

Alexander Kojève (1947) _Introduction à la lecture de Hegel: Leçons sur "La Phénoménologie de l'Esprit"_, Paris: Gallimard.

Friedrich Kittler (1990) _Discourse Networks 1800/1900_, trans. M. Metteer & C. Cullens, Stanford: Stanford University Press.

Friedrich Kittler (1996) 'There is no Software', http://www.ctheory.net/text_file.asp?pick=74 [last accessed 31 Dec 2005]; also in Timothy Druckery, ed. (1996) _Electronic Culture_, New York: Aperture, pp. 331-337.

Friedrich Kittler (1999) 'On the Implementation of Knowledge - Toward a Theory of Hardware', _Nettime_, http://www.nettime.org/nettime.w3archive/199902/msg00038.html [last accessed 31 Dec 2005].

Melanie Klein (1988) _Love, Guilt and Reparation_, London: Vintage.

Naomi Klein (2001) _No Logo_, London: Flamingo.

Eric Kluitenberg (2002) 'Transfiguration of the Avant-Garde/The Negative Dialectics of the Net', quoted in Duna Mavor, 'avant.garde - tranfigured or dead?', _Nettime_, March, http://www.nettime.org/ [last accessed 31 Dec 2005].

Donald Knuth (1981 [1968]) _The Art of Computer Programming: Volume 1, _Fundamental Algorithms_, Reading, Mass.: Addison-Wesley.

Chris Kraus & Sylvère Lotringer, eds. _Hatred of Capitalism: A Semiotext(e) Reader_, Los Angeles: Semiotext(e), pp. 273-280.

Ronald D. Laing (1965 [1960]) _The Divided Self: An Existential Study in Sanity and Madness_, Harmondsworth: Penguin.

Dominique Laporte (2000) _History of Shit_, London: MIT Press.

David-Olivier Lartigaud (2004) ‘Introduction’, _ Programmation Orientée Art_, colloque organised with Anne-Marie Duguet, CRECA, Université Paris, Sorbonne, 19/20 March.

Bruno Latour (1999 [1987]) _Science in Action: How to Follow Scientists and Engineers Through Society_, Cambridge, Mass.: Harvard University Press.

Maurizio Lazzarato (1996) ‘Immaterial Labour’, trans. Paul Colilli & Ed Emory, in Paolo Virno & Michael Hardt, eds. _Radical Thought in Italy_, Minneapolis: University of Minnesota Press, pp. 132-146.

Maurizio Lazzarato (1999) ‘New Forms of Production and Circulation of Knowledge’, trans. Bram Dov Abramson, in Josephine Bosma et al, eds. _Readme! Filtered by Nettime: ASCII Culture and the Revenge of Knowledge_, New York: Autonomedia, pp. 159-166.

Maurizio Lazzarato (2003) ‘Digital Work’ seminar [transcribed by Leslie Robbins & Alejandra Nunez Perez], Media Design Research, Piet Zwart Institute & V2_ Organisation, Institute for the Unstable Media.

Maurizio Lazzarato (2004) ‘General Intellect: Towards an Inquiry into Immaterial Labour’, trans. Ed Emery, [first published in _Common Sense_] http://multitudes.samizdat.net/article.php3?id_article=1498 [accessed 31 Dec 2005].

Henri Lefebvre (1968) ‘The Marxian Concept of Praxis’, in _The Sociology of Marx_, trans. Norbert Guterman, New York: Pantheon, pp. 25-58.

Henri Lefebvre (1991) _The Production of Space_, trans. Donald Nicholson-Smith, Oxford: Blackwell.

Fred Lerdahl & Ray Jackendoff (1983) _A Generative Theory of Tonal Music_, Cambridge, Mass.: MIT Press.

Esther Leslie (2000) ‘The Work of Art in the Age of Unbearable Capitulation’, in _Walter Benjamin: Overpowering Conformism_, London: Pluto Press, pp. 130-167.

Esther Leslie (2004) ‘Globalica: Communism, Culture and the Commodity’ in Geoff Cox, Joasia Krysa, Anya Lewin, eds. _Economising Culture_ New York: Autonomedia, pp. 89-109.

Lawrence Lessig (1999) _Code and Other Laws of Cyberspace,_ New York: Basic Books; also http://lessig.org/

Lawrence Lessig (2001) _The Future of Ideas: The Fate of the Commons in a Connected World_, New York: Random House.

Lawrence Lessig (2004) _Free Culture: How Big Media Uses Technology and the Law to Lock Down Culture and Control Creativity_, http://www.free-culture.cc/freeculture.pdf/ [last accessed 31 Dec 2005].

Ruth Levitas (1989) ‘The Future of Thinking about the Future’, in Jon Bird, Barry Curtis et al, eds. _Mapping the Future_, London: Routledge.

Richard Levins & Richard Lewontin (1985) _The Dialectical Biologist_, Cambridge, Mass.: Harvard University Press.

Claude Lévi Strauss (1970 [1964]) _The Raw and the Cooked: Introduction to a Science of Mythology_, trans. John & Doreen Weightman, Harmonsworth: Penguin.

Steven Levy (1994) _Hackers: Heroes of the Computer Revolution_, New York: Penguin.

Lawrence Liang (2004) _Guide to Open Content licenses_, Rotterdam: Piet Zwart Institute.

Jacob Lillemose (2004) ‘A Re-Declaration of Dependence - Software Art in a Cultural Context It Can’t Get Out Of’, in Olga Goriunova & Alexei Shulgin, eds. _Read_Me: Software Art & Cultures_, Arhus: DARC, pp. 136-149.

Lucy Lippard, ed. (1997) _Six Years: the dematerialization of the art object from 1966 to 1972 [...]_, London: University of California Press.

Geert Lovink & Florian Schneider (2001) ‘New Rules for the New Actonomy’, _Nettime_, 25 June, http://www.

nettime.org/nettime.w3archive/200106/msg00141.html [last accessed 31 Dec 2005].

Geert Lovink (2002) _Dark Fiber: Tracking Critical Internet Culture_, Cambridge, Mass.: MIT Press.

Geert Lovink & Florian Schneider (2003) ‘Reverse Engineering Freedom’, _Nettime_, 24 Sept, http://www.nettime.org/ [last accessed 31 Dec 2005].

Peter Lunenfeld, ed. (2000) _The Digital Dialectic: New Essays on New Media_, Cambridge, Mass.: MIT Press.

Peter Lunenfeld (2002) ‘Snap to Grid’, quoted in _ Rhizome Digest_, May, http://www.rhizome.org/ [last accessed 31 Dec 2005].

Georg Lukács (1976 [1922]) _History and Class Consciousness: Studies in Marxism_, trans. Rodney Livingstone, Cambridge, Mass.: MIT Press.

Adrian Mackenzie (2005) ‘The Performativity of Code: Software and Cultures of Circulation’, http://www.lancs.ac.uk/staff/mackenza/papers.php [last accessed 31 Dec 2005]; also in _Theory, Culture & Society_, vol. 22, no. 1, London: Sage, pp. 71-92.

Ernest Mandel (1975) _Late Capitalism_, London: New Left Books.

Ernest Mandel (1990 [1976]) ‘Introduction’, to Karl Marx, _Capital: Volume 1_, London: Penguin, pp. 11-86.

Ernest Mandel (1995 [1978]) _The Long Waves of Capitalist Development: A Marxist Interpretation_, London: Verso.

Lev Manovich (1999) ‘Avant-garde as Software’, http://www.manovich.net/TEXTS_04.HTM [last accessed 31 Dec 2005] [first published in Stephen Kovats, ed. _ Ostranenie_, Frankfurt: Campus Verlag].

Lev Manovich (2001) _The Language of New Media_, Cambridge, Mass.: MIT Press.

Lev Manovich (2002) ‘Generation Flash’, http://www.manovich.net/TEXTS_04.HTM [last accessed 31 Dec 2005].

Mao Tsetung (1977) _Five Essays on Philosophy_, Peking: Foreign Languages Press.

Carolyn Marvin (1990) _When Old Technologies Were New: Thinking About Electronic Communication in the Late Nineteenth Century_, Oxford: Oxford University Press.

Karl Marx (1978 [1845-6]) ‘The German Ideology’, in Robert C. Tucker, ed. _The Marx-Engels Reader_, New York: Norton, pp. 146-200.

Karl Marx (1980) [1851-2]) ‘The Eighteenth Brumaire of Louis Bonaparte’ & [1845]) ‘Theses on Feuerbach’, in _ Karl Marx and Frederick Engels: Selected Works_, London: Lawrence and Wishart, pp. 96-179 & pp. 28-31.

Karl Marx (1981 [1857-8 written]) _Grundrisse: Foundations of the Critique of Political Economy (Rough Draft)_ trans. Martin Nicolaus [first published 1939, this translation 1973] Harmondsworth: Penguin.

Karl Marx (1990 [1867]) _Capital: Volume 1_, trans. Ben Fowkes, Harmondsworth: Penguin.
Karl Marx & Friedrich Engels (1985 [1848]) _The Communist Manifesto_, Harmondsworth: Penguin.

Harry Mathews & Alastair Brotchie (1998) eds. _Oulipo Compendium_, London: Atlas Press.

Humberto R. Maturana & Francisco J. Varela (1980 [1973]) _Autopoiesis and Cognition: the Realization of the Living_, Robert S. Cohen & Marx W. Wartofsky, eds. Boston Studies in the Philosophy of Science 42, Dordecht: D. Reidel Publishing.

Duna Mavor [aka Joanne Richardson] (2002) ‘avant.garde - tranfigured or dead?’, _Nettime_, March, http://www.nettime.org/ [last accessed 31 Dec 2005].

Marcel Mauss (1970) _The Gift: Forms and Functions of Exchange in Archaic Societies_, trans. Ian Cunnison, London: Cohen & West.

Marshall McLuhan & Quentin Fiore (1967) _The Medium is the Massage: An Inventory of Effects_, New York: Bantam Books.

Armin Medosch (2003) ‘Piratology: the deep seas of open code and free culture’, in Medosch, ed. (2003) _dive: an introduction to the world of free software and copyleft culture_, Liverpool: FACT, pp. 8-19.

Armin Medosch (2005) ‘Roots Culture: Free Software Vibrations Inna Babylon’, in Geoff Cox & Joasia Krysa, eds. _Engineering Culture_, New York: Autonomedia, pp. 177-201.

Gustav Metzger (1996) ‘Auto-Destructive Art’, in _ Damaged Nature, Auto-Destructive Art_, London: Coracle, pp. 25-63.

Eben Moglen (2003) ‘\DEF\MYTITLE{DOTCOMMUNIST MANIFESTO}’, in Ivet Curlin, Anna Devic, Natasa Ilic, Dejan Krsic, Sabina Sabolovic, eds. _What, How & For Whom: on the occasion of the 152nd anniversary of the Communist Manifesto_, Zagreb: Arkzin, pp. 216-223.

Nicholas Mosley (1972) _The Assassination of Trotsky_, London: Abacus.

Warren F. Motte Jr. ed. (1998) _Oulipo: a Primer of Potential Literature_, trans. Warren F. Motte, Illinois: Dalkey Archive Press.

Mark Napier (2000) interviewed by Andreas Broegger, ‘The Aesthetics of Programming’, in conjunction with exhibition _on/off_, Copenhagen, http://www.afsnitp.dk/onoff/ [last accessed 31 Dec 2005].

Antonio Negri (1991) _Marx Beyond Marx: Lessons on the Grundrisse_, Jim Flemming, ed., trans. Harry Cleaver, Michael Ryan & Maurizio Viano, New York: Autonomedia.

Antonio Negri (1999 [1998]) ‘Back to the Future: A Portable Document’, trans. Michael Hardt, in Josephine Bosma, et al, eds. _Readme! Filtered by Nettime: ASCII Culture and the Revenge of Knowledge_, New York: Autonomedia, pp. 181-186.

Antonio Negri (2002) debate on ‘counter-empire’, at _ sherwood_, 17 May, http://www.sherwood.it/controimpero/ [last accessed 31 Dec 2005].

John von Neumann & Oskar Morgenstern (1944) _Theory of

Games and Economic Behaviour_, Princeton: Princeton University Press.

Bill Nichols (1988) ‘The Work of Culture in the Age of Cybernetic Systems’, in _Screen_, vol.29, no.2, Winter, pp. 22-46.

David Noble (1995) _Progress without People: New Technology, Unemployment, and the Message of Resistance_, Toronto: Between the Lines.

Franziska Nori, ed. (2002) _I Love You: Computer, Viren, Hacker, Kultur_, exhibition catalogue, Museum Für Angewandte Kunst in Frankfurt am Main.

Sue Owens (1996) ‘Chaos Theory, Marxism and Literary History’, in Jody Berland & Sarah Kember, eds. _New Formations: Technoscience_, Number 29, Summer, London: Lawrence & Wishart.

Harold Osborne, ed. (1988) _The Oxford Companion to Twentieth-Century Art_, Oxford: Oxford University Press.

Matteo Pasquinelli (2004) ‘Radical Machines Against the Techno-Empire: From Utopia to Network’, trans. Arianna Bove, http://www.rekombinant.org/downloads/radical_machines.pdf [last accessed 31 Dec 2005] [french version, in _Multitudes 21_, http://www.eurozine.com/partner/multitudes/current-issue.html].

Christiane Paul (2003a) _Digital Art_, London: Thames & Hudson.

Christiane Paul (2003b) ‘Public CulturalProduction Art(Software){‘, in Gerfried Stocker & Christine Schöpf, eds. _Code - The Language of Our Time_, Ars Electronica, Linz: Hatje Cantz, pp. 129-135.

Joachim Paul (2000), ‘Gotthard Günther, the “Einstein” of Philosophy’, http://www.vordenker.de/ggphilosophy/ggeinstein_en.htm [last accessed 24 Nov 2005].

Georges Perec (1995 [1969]) _A Void_, [_La Disparition_] trans. Gilbert Adair, London: Harvill Press.

Norbert Pfaffenbichler & Sandro Droschl, eds. (2003)

Abstraction Now, Künstlerhaus Wien: Edition Camera Austria.

Sadie Plant (1995) _The Most Radical Gesture: The Situationist International in a Postmodern Age_, London: Routledge.

Sadie Plant (1998) _Zeros + Ones_, London: Fourth Estate.

Edgar Allan Poe (1836) 'Maelzel's Chess Player', [first published in _Southern Literary Messenger_], http://www.book-portal.net/poe/volume4/maelzels.html [last accessed 31 Dec 2005].

Karl Popper (2003 [1945]) _The Open Society and its Enemies_, London: Routledge Classics.

Mark Poster (1997) 'Cyberdemocracy: Internet and the Public Sphere', in David Porter. ed. _Internet Culture_, London: Routledge, pp. 201-217.

Ilya Prigogine & Isabelle Stengers (1985) _Order Out of Chaos: Man's New Dialogue With Nature_, London: Fontana.

Eric S. Raymond (2004) _The Art of UNIX Programming_, Boston: Addison-Wesley.

Casey Reas (2004) _{Software}Structures_, in Olga Goriunova & Alexei Shulgin, eds. _Read_Me: Software Art & Cultures_, Arhus: DARC, pp. 276-297.

Jonathan Rée (1999) _I See a Voice: a Philosophical History_, London: Flamingo.

Jeremy Rifkin (1995) _The End of Work: The Decline of the Global Labor Force and the Dawn of the Post-Market Era_, New York: Putnam.

John Roberts, ed. (1994) _Art Has No History! The Making and Unmaking of Modern Art_, London: Verso.

David A. Ross (2003) 'Radical Software Redux', http://www.radicalsoftware.org/e/ross.html [last accessed 31 Dec 2005].

Michael Rush (1999) _New Media in Late-20th Century Art_, London: Thames & Hudson.

Mirko Schäfer (2004) ‘Made by Users: How Users Improve Things, Provide Innovation and Change our Idea of Culture - Problems and Perspectives’, in Olga Goriunova & Alexei Shulgin, eds. _Read_Me: Software Art & Cultures_, Arhus: DARC, pp. 62-77.

Antoine Schmitt (2003) ‘software art vs. programmed art’, posting on _Rhizome_Raw_, 4 Oct, http://rhizome.org/thread.rhiz?thread=10403&text=20334#20334 [last accessed 31 Dec 2005].

Pit Schultz (2002) ‘Re: The Hacker Class/On Empire’, _Nettime_, 02 June, http://www.nettime.org/ [last accessed 31 Dec 2005].

Pit Schulz (2005) ‘The Producer as Power User’, in Geoff Cox & Joasia Krysa, eds. _Engineering Culture_, New York: Autonomedia, pp. 111-125.

Hans-Peter Schwarz (1997) _Media Art History_, ZKM Center for Art and Media, Karlsruhe: Prestel.

William Curtis Seaman (1999) _Recombinant Poetics: Emergent Meaning as Examined and Explored Within a Specific Generative Virtual Environment_, PhD thesis, CAiiA: Centre for Advanced Inquiry in the Interactive Arts, University of Wales.

Edward A. Shanken (1998) ‘The House That Jack Built: Jack Burnham’s Concept of “Software” as a Metaphor for Art’, in _Leonardo Electronic Almanac_, 6:10, November, http://mitpress.mit.edu/e-journals/LEA/ARTICLES/jack.html [last accessed 31 Dec 2005]

Edward A. Shanken (2003) ‘From Cybernetics to Telematics: The Art, Pedagogy, and Theory of Roy Ascott’, in Roy Ascott, _Telematic Embrace: Visionary Theories of Art, Technology, and Consciousness_, Berkeley: University of California Press, pp. 1-94.

Alexei Shulgin (1997) ‘Re: nettime: Art on Net’, _nettime_, 13 March, http://www.nettime.org/Lists-Archives/nettime-l-9703/msg00066.html [last accessed 31 Dec 2005].

Peter Singer (1983) _Hegel_, Oxford: Oxford University Press.

Ricard Solé & Brian Goodwin (2000) _Signs of Life: How Complexity Pervades Biology_, New York: Perseus Books.

Cornelia Sollfrank (2001) ‘Hacking the Art Operating System’, interviewed by Florian Cramer, Chaos Computer Club, Berlin; also version published in 2002-3, in Simon Yuill & Kerstin Mey, eds., _Communication, Interface, Locality_, Manchester University Press in association with Duncan of Jordanstone College of Art and Design.

Alan Sondheim (2001) ‘Introduction to Codework’, in _American Book Review_, 22 September, no. 6, pp. 1-4; also http://www.litline.org/ABR/issues/Volume22/Issue6/sondheim.pdf [last accessed 31 Dec 2005].

Alan Sondheim (2005) ‘On Code and Codework’, _Nettime_, 12 March.

Felix Stalder & Jesse Hirsh (2002) ‘Open Source Intelligence’, _First Monday_, vol.7, no.6, June, http://firstmonday.org/issues/issue7_6/stalder/index.html [last accessed 31 Dec 2005].

Julian Stallabrass (2003) _Internet Art: The Online Clash of Culture and Commerce_, London: Tate Publishing.

Richard M. Stallman (1994) ‘Why Software should not have Owners’, http://www.gnu.org/philosophy/why-free.html [last accessed 31 Dec 2005].

Richard M. Stallman (1996) ‘The Free Software Definition’, http://www.gnu.org/philosophy/free-sw.html [last accessed 31 Dec 2005].

Richard M. Stallman (1998) ‘The GNU Project’, http://www.gnu.org/gnu/thegnuproject.html [last accessed 31 Dec 2005].

Richard M. Stallman (2002) _Free Software, Free Society: Selected Essays of Richard M. Stallman_, Joshua Gay, ed. Free Software Foundation.

Gerfried Stocker & Christine Schöpf, eds. (2003) _Code - The Language of Our Time_, Ars Electronica, Linz: Hatje Cantz.

Tiziana Terranova (2000) ‘Free labor: producing culture for the digital economy’, in _Social Text_, 63, Vol. 18, No. 2, Durham: Duke University Press, pp. 33-58.

Tiziana Terranova (2002) ‘The degree zero of politics: virtual cultures and virtual social movements’, _Film-Philosophy_, 06 Feb, http://www.film-philosophy.com/ [last accessed 31 Dec 2005].

Tiziana Terranova (2004) _Network Culture: Politics for the Information Age_, London: Pluto Press.

Rolf Tiedemann (1999 [1988]) ‘Dialectics at a Standstill: Approaches to the Passagen-Werk’, trans. Gary Smith & André Lefevere, in _Walter Benjamin, The Arcades Project_, Cambridge, Mass.: Belknap Press, pp. 929-945.

Mario Tronti (1980 [1965]) ‘The Strategy of Refusal’, in ‘Autonomia: Post-political Politics’, _Semiotext(e)_ vol. 3, no. 3, New York: Semiotext(e), pp. 28-34.

Sherry Turkle (1997) _Life on the Screen: Identity in the Age of the Internet_, London: Phoenix.

Tristan Tzara (1998 [1918]) ‘Dada Manifesto’, in Charles Harrison & Paul Wood, eds. _Art in Theory: 1900-1990: an anthology of changing ideas_, Oxford: Blackwell, pp. 249-253.

Gianni Vattimo (1992) _The Transparent Society_, trans. David Webb, Cambridge: Polity Press.

Roman Verostko (2004) ‘Imaging the Unseen: A statement on my pursuit as an artist’, http://www.verostko.com/archive/statements/statement-recent.html [last accessed 31 Dec 2005].

Paolo Virno (1996) ‘Virtuosity and Revolution: The Political Theory of Exodus’, trans. Ed Emory, in Paolo Virno & Michael Hardt, eds. _Radical Thought in Italy_, Minneapolis: University of Minnesota Press, pp. 188-209.

Paulo Virno (2004) _A Grammar of the Multitude: For an Analysis of Contemporary Forms of Life_, trans. Isabella Bertoletti, James Cascaito, Andrea Casson, New York: Semiotext(e).

Marina Vishmidt (2005) ‘Precarious Straits’, in _Mute Vol. II #0 - Precarious Reader_, London: Mute, pp. 38-43.

Yvonne Volkart (2004), ‘Calculating and Calculators: The algorithmic and generative as an aesthetic strategy’, in _Springerin_, trans. Timothy Jones, http://www.springerin.at/dyn/heft_text.php?textid=1562&lang=en [last accessed 31 Dec 2005].

Larry Wall (1998), ‘computer code and linguistics’, http://www.ddj.com/documents/s=923/ddj9802a/9802a.htm [last accessed 31 Dec 2005].

Larry Wall (1999), ‘Perl, the first postmodern computer language’, http://www.wall.org/~larry/pm.html [last accessed 31 Dec 2005].

McKenzie Wark (2002a), ‘From Hypertext to Codework’, _HJS_, vol. 3, issue 1, http://www.geocities.com/hypermedia_joyce/wark.html [last accessed 31 Dec 2005].

McKenzie Wark (2002b), ‘on material and “immaterial” labour’, _Nettime_, 01 June, http://www.nettime.org/ [last accessed 31 Dec 2005].

McKenzie Wark (2002c) ‘The Property Question: Culture, Economy, Information’, _Nettime_ 10 Feb, http://www.nettime.org/ [last accessed 31 Dec 2005].

McKenzie Wark (2004) _A Hacker Manifesto_, Cambridge, Mass.: Harvard University Press [earlier version (2001) ‘Hacker Manifesto 2.0’, http://www.feelergauge.net/projects/hackermanifesto/version_2.0/] [last accessed 31 Dec 2005].

Tomas P. Weissert (1991) ‘Representation and Bifurcation: Borges’s Garden of Chaos Dynamics’, in N. Katherine Hayles, ed. _Chaos and Order_, Chicago: University of Chicago Press, pp. 223-243.

Mitchell Whitelaw (2005) ‘System Stories and Model Worlds: A Critical Approach to Generative Art’, in Olga Goriunova, ed. _Readme 100: Temporary Software Art Factory_, Dortmund: Hartware Medien Kunst Verein, pp. 134-153.

Norbert Wiener (2000 [1948]) _Cybernetics: or Control and Communication in the Animal and the Machine_, Cambridge, Mass.: MIT Press.

Raymond Williams (1988) _Keywords: A vocabulary of culture and society_, London: Fontana.

Stephen Wilson (2002) _Information Arts: Intersections of Art, Science, and Technology_, Cambridge, Mass.: MIT Press/Leonardo Books.

Gaby Wood (2002) 'An Unreasonable Game', in _Living Dolls_, London: Faber & Faber, pp. 55-103.

Richard Wray (2005) 'EU says Internet could fall apart', _The Guardian_, 12 October, http://technology.guardian.co.uk/news/story/0,16559,1589967,00.html [last accessed 31 Dec 2005].

Richard Wright (1998) _Montage – Transformation – Allegory: A Study of Digital Imaging in Dialectical Film Making_, PhD thesis, London Guildhall University.

Richard Wright (1999) 'Programming with a Paintbrush: The Last Interactive Workstation', http://www.runme.org/project/+Painting/ http://www.nettime.org/ [last accessed 31 Dec 2005]; also in _Filmworks_, no.12, Autumn 2000.

Richard Wright (2004) 'Software Art After Programming', _Metamute_, July, http://www.metamute.com/look/article.tpl?IdLanguage=1&IdPublication=1&NrIssue=28&NrSection=10&NrArticle=1397&ST_max=0 [last accessed 31 Dec 2005].

Steve Wright (2005) 'Reality Check: Are We Living in an Immaterial World?', in Josephine Berry Slater, ed. _Underneath The Knowledge Commons_, vol. 2 #1, London: Mute, pp. 34-45.

Wu Jie (1996) _Systems Dialectics_, Beijing: Foreign Languages Press.

Slavoj Žižek, ed. (1997) 'Introduction', in _Mapping Ideology_, London: Verso.

Slavoj Žižek (1998) _The Spectre Is Still Roaming Around!_, Zagreb: Bastard Books.

Slavoj Žižek (1999a) 'Hegel's "Logic of Essence" as a Theory of Ideology', & 'Is It Possible to Traverse the Fantasy in Cyberspace?', in Elizabeth Wright & Edmond Wright, eds. _The Zizek Reader_, Oxford: Blackwell, pp. 225-250 & 102-124.

Slavoj Žižek (1999b) ‘Introduction: A Spectre Is Haunting Western Academia...’ & ‘Part 1: The “Night of the World”’, in _The Ticklish Subject: the absent centre of political ontology_, London: Verso, pp. 1-123.

Slavoj Žižek (2003 [2001]) ‘A Holiday from History: and other real stories’, in Johan Grimonprez, _dial H-I-S-T-O-R-Y_, [first published in Janus #9], Ostfildern: Hatje Cantz.

7.3 projects cited

0100101110101101.ORG & EpidemiC (2001) _biennale.py_, http://www.0100101110101101.org/home/biennale_py/index.html

0100101110101101.ORG (2000-03) _life_sharing_, http://www.0100101110101101.org/home/life_sharing/index.html

46LiverpoolSt.org (Tim Brennan, Geoff Cox & Adrian Ward) (1999) _Manifest_, http://www.46liverpoolst.org [offline]

Eric Andreychek (2001) _Jabberwocky_, http://www.perlmonks.org/index.pl?node_id=111157

Antiorp/Netochka Nezvanova, http://www.m9ndfukc.com/ [currently unavailable]

Maurizio Bolognini (1992-) various iterations of sealed computers - see Bolognini (2004) section 7.2 above.

Christophe Bruno (2006) _Human Browser_, http://www.iterature.com/human-browser/

CODeDOC (2002) online exhibition, http://artport.whitney.org/commissions/codedoc/index.shtml; & _CODeDOC II_ (2003) http://www.aec.at/de/festival2003/programm/codedoc.asp

Harold Cohen (1973-) _Aaron_, http://crca.ucsd.edu/~hcohen/

Conglomco.org & The Carbon Defense League (2003) _Re-code_, http://www.re-code.com/ [project since offline]

Constant vzw, ‘Cuisine Interne Keuken’, http://www.

constantvzw.com/cn_core/cuisine/aboutEN.php

Copenhagen Free University [aka Henriette Heise & Jakob Jakobsen], http://www.copenhagenfreeuniversity.dk/

Florian Cramer (1996-2000) _Permutations_, http://userpage.fu-berlin.de/~cantsin/permutations/

Creative Commons, http://creativecommons.org/

Cybernetic Serendipity (1968) exhibition, http://www.medienkunstnetz.de/exhibitions/serendipity/

Digitalcraft.org (2002) _I Love You_, http://www.digitalcraft.org/iloveyou/index.htm

Richardo Dominguez/Electronic Disturbance Theatre (1998) _Floodnet_ http://www.thing.net/~rdom/ecd/ecd.html

Elmo, Gum, Heather, Holly, Mistletoe and Rowan (2003) _Notes Towards the Complete Works of Shakespeare_, book/DVD (produced by Geoff Cox), London: Kahve-Society & Book Works, as part of http://www.vivaria.net/

Free Software Foundation, http://www.fsf.org/

Matthew Fuller (2000) _A Song for Operations_

Futurenatural [aka Richard Wright] (2001) _The Bank of Time_, http://www.thebankoftime.com/ [also see http://runme.org/project/+BoT/]

Generator (2002-03) exhibition (curated by Geoff Cox & Tom Trevor for Spacex), http://www.generative.net/generator/

Hans Haacke (1971) _Visitors’ Profile_

Harwood (2002) _london.pl_, http://www.mongrelx.org/home/index.cgi?LondonPL

Indymedia, http://www.indymedia.org/

Jaromil (2002) _forkbomb_, http://www.runme.org/project/+forkbombsh/

Jaromil (2001-) _dyne:bolic_, http:/dynebolic.org/

JODI [aka Joan Heemskerk & Dirk Paesmans] (1995) _wwwwwwwwww_, http://wwwwwwwww.jodi.org

JODI (2002) _Jet Set Willy ©1984_, http://jetsetwilly.jodi.org/

JODI (2002) _My%Desktop_ (see Connor 2004)

JODI (2004) _Desktop Improvisations: My%Desktop Live_, performance & DVD commissioned by Fact, Liverpool as part of exhibition _Computing 101B_.

Joasia Krysa & Grzesiek Sedek (2005) _softwareKURATOR v.1_, http://www.kurator.org/

Sol LeWitt (1968-) _Wall Drawings_

Robert Luxembourg (aka Sebastian Lütgert) (2003) _The Conceptual Crisis of Private Property as a Crisis in Practice_, http://rolux.net/crisis/

Rainer Mandl, Annja Krautgasser (2003) _Pedigree_, http://www.aec.at/de/festival2003/programm/codedoc/krautgasser/project.asp

Alex McLean (2003) _animal.pl_, http://www.vivaria.net/taxonomy/examples/animal/ [only death notice online]

Alex McLean (2001) _forkbomb.pl_, http://www.runme.org/project/+forkbomb/

Alex McLean (2004) _feedback.pl_, http://yaxu.org/words/papers/feedback.html

Armin Medosch, Shu Lea Cheang & Yukiko Shikata (2004) _The Kingdom of Piracy_, http://residence.aec.at/kop/

Mez (aka Mary Anne Breeze), http://www.hotkey.net.au/~netwurker/

Mongrel (1999) _Linker_, http://www.mongrel.org.uk/Linker

Mongrel/Harwood (2003) _Nine_(9), http://9.waag.org/

Yoko Ono (2001) _Mend Peace for the World_, http://www.generative.net/generator/browse.cgi?page=ono

Ordure.org (2000) _Dust_, http://dump.ordure.org/www.ordure.org/291/dust.html

OuLiPo [Ouvroir de Littérature Potentielle], http://www.oulipo.net/

p0es1s: Digitale Poesie (2004), http://www.p0es1s.net/

Project Gnutenberg, _pngreader_, http://pngreader.gnutenberg.net/ [currently offline]

Project Gnutenberg (2002) _walser.php_, http://www.textz.com/trash/walser.php.txt

Raymond Queneau (1961) _Cent Mille Milliards de Poemes_ [one hundred thousand billion poems]

Radical Software journal (1970-74) http://www.radicalsoftware.org/

radioqualia (2002) _Free Radio Linux_, http://radioqualia.va.com.au/freeradiolinux/

Tom Ray (1990-) _Tierra_, http://www.his.atr.jp/~ray/tierra/

readme (2002-) http://readme.runme.org/

Casey Reas (2004) _{Software} Structures_, http://artport.whitney.org/commissions/softwarestructures/

Redundant Technology Initiative (1996-), http://www.lowtech.org/

RTMark (1996-) http://www.rtmark.com/

runme (2003-) http://www.runme.org/

Bill Seaman (1996-) _The World Generator/The Engine of Desire_, http://digitalmedia.risd.edu/billseaman/workSpcWorld01.php

Signwave (aka Adrian Ward) (2000-) _Auto Illustrator_, http://www.auto-illustrator.com/

Signwave (2001-) _Anagrammar_ http://www.signwave.co.uk/go/products/anagrammar/

Software, Information Technology: Its Meaning for Art exhibition (1970)

socialfiction.org (aka Wilfried Hou Je Bek) (2003) _.WALK_, http://www.socialfiction.org/dotwalk/

Leonardo Solaas (2005) _Outsource me!_, http://outsource.solaas.com.ar/

Cornelia Sollfrank (1999) _net.art generator_ http://obn.org/generator/

Together We Can Defeat Capitalism (2003), _Anti-Capitalist Operating System v2.0_, http://www.twcdc.com/

Toplap (2004-), http://www.toplap.org/

transmediale, http://www.transmediale.de/

UK Museum of Ordure (Stuart Brisley, Geoff Cox & Adrian Ward) (2001-) http://www.museum-ordure.org.uk/

University of Openness, http://twenteenthcentury.com/uo/index.php & its Faculty of Unix, http://darq.org.uk/

Roman Verostko (1969-) _Algorithmic Art_, http://www.verostko.com/

For other references, go to http://www.anti-thesis.net/