Cultural Studies Review
volume 20 number 1 March 2014
http://epress.lib.uts.edu.au/journals/index.php/csrj/index
pp. 30–52
© Jussi Parikka 2014

Cultural Techniques of Cognitive Capitalism

Metaprogramming and the Labour of Code

JUSSI PARIKKA
UNIVERSITY OF SOUTHAMPTON

—I

This article is about cultural techniques and software culture. The notion of cultural techniques stems from German media studies and refers to a range of epistemic, embodied, cognitive and affective orders. It shows its particular usefulness in how it extends our conventional notions of media.[1] Media become more than media, and can include 'inconspicuous techniques of knowledge like card indexes, media of pedagogy like the slate, discourse operators like quotation marks, uses of the phonograph in phonetics, or techniques of forming the individual like practices of teaching to read and write' as well as maps, doors, operations and practices of law, and so much more.[2] Cultural techniques participate in the formation of subjects, as well as constitute ways of knowing and organising social reality. I am in this context interested in how we can read some aspects of software culture and organisation of the labour of programming in relation to cultural techniques. Modes of organisation as well as practices of coding represent ways in which code and software regulate social reality but that they are also being regulated as a prioritised technique in digital economy that is at times related to discussions concerning cognitive modes of production.

ISSN 1837-8692

In this article, the notion of cultural techniques is coupled with a concept from a very different tradition to that of German media theory: I engage with *cognitive capitalism* and specifically cultural techniques of cognitive capitalism. My interest lies in addressing software cultures and techniques of work and labour as ways to understand the constitution of our capitalist technological environment. This set of practices and techniques has to do with management of code and code work, hence, as a punchline to summarise the article early on: cultural techniques of cognitive capitalism should not only to be about the cognitive, I argue, but a range of practices, techniques, management and organisation that happens outside the brain. In other words, what sustains the cognitive is a field of techniques. It is an argument that someone within the field of brain sciences could make as well as scholars observing that any cognitive activity happens on a distributed field of the self, and that the brain is anyway extended as part of its surroundings.[3] Besides neurosciences and cognitive theory, for instance theories of cognition and affect in design are taking such ideas into account.[4] Yet my focus is on aspects of media theory and software and I will leave the otherwise interesting cognitive science contexts out. Of course, one can excavate a media archaeology of cognitive capitalism through its ties with cognitive sciences and the wider scientific discourse concerning the brain, communication and cooperation but here I focus on software cultures and a more technological understanding of the cognitive.[5] We have to be aware of the way cultural techniques relating to code and programming are themselves one important framing of the cognitive. This idea relates even to very early formulations of the role of the computer as universal, programmable machine. Already in the 1940s the computer pioneer Alan Turing proclaimed that the future of computing was situated in the office—in the 'office work of programming' to be exact.[6]

Most often when defining cultural techniques people turn to a quote from Thomas Macho, a German cultural studies scholar:

> Cultural techniques—such as writing, reading, painting, counting, making music—are always older than the concepts that are generated from them. People wrote long before they conceptualized writing or alphabets; millennia passed before pictures and statues gave rise to the concept of the image; and still today, people sing or make music without knowing anything about tones or musical notation systems. Counting, too, is older

> than the notion of numbers. To be sure, most cultures counted or performed certain mathematical operations, but they did not necessarily derive from this a concept of number.[7]

Following the notion of cultural techniques, I want to excavate the specific techniques that sustain the notion of cognitive capitalism as used by, for instance, Yann Moulier Boutang.[8] The allusion to the powers of the brain as communicative, organisational and a coordinating factor in production of value needs to be investigated in relation to its technological conditions as well. This sounds like a media archaeological argument but here it will be approached through the concept of cultural techniques, particularly as it has been formulated in recent German media theory.[9] Following, for instance, Sybille Krämer and Horst Bredekamp's lead we can extend cultural studies investigations towards material, scientific and technological operations:

> cultural techniques are promoting the achievements of intelligence through the senses and the externalizing operationalization of thought processes. Cognition does not remain locked up in any invisible interiority; on the contrary intelligence and spirit advance to become a kind of distributive, and hence collective, phenomenon that is determined by the haptic contact humans have with things and with symbolic and technical artifacts.[10]

I want to focus on this extended notion of the cognitive as a way to investigate the historical nature in which such political and economic notions as cognitive capitalism also spread. So in this sense, the focus is on the mundane, habitual, repetitious procedures grounding the cognitive, a point that will be clearer with a close reading of some ideas of the characteristics of cognitive capitalism and its mediatic conditions of existence. What is gradually revealed are the various grey, laborious and tiring practices that brand the other side of work in digital culture.[11]

—II

I will continue with an excursion into 1980s software culture and a speech by Steve Jobs that made its way from an old C-tape recording of a conference presentation to digital formats and online.[12] In this talk at the 1983 International Design Conference in Aspen, Jobs outlines visions familiar from the early phases of personal computing but which for some are premediating our smart, mobile technology culture of App Stores and tablets. Jobs adapts a language that one can easily see relating to more famous media theory—that of Marshall McLuhan—when talking about the emerging new media cultures that are still embedded in '*old habits*'. The same pattern, argues Jobs, has happened in relation to earlier media changes concerning broadcast media of radio and television. Their essence was carved out only in slow progression and 'perfection' of the format. The 1950s was, in Jobs' words, a slow period of education in the world of television, with the help of programs such as *I Love Lucy*. But the true stakes of the medium were revealed with the funeral of John F. Kennedy in 1963 and the Apollo moon landing in 1969.

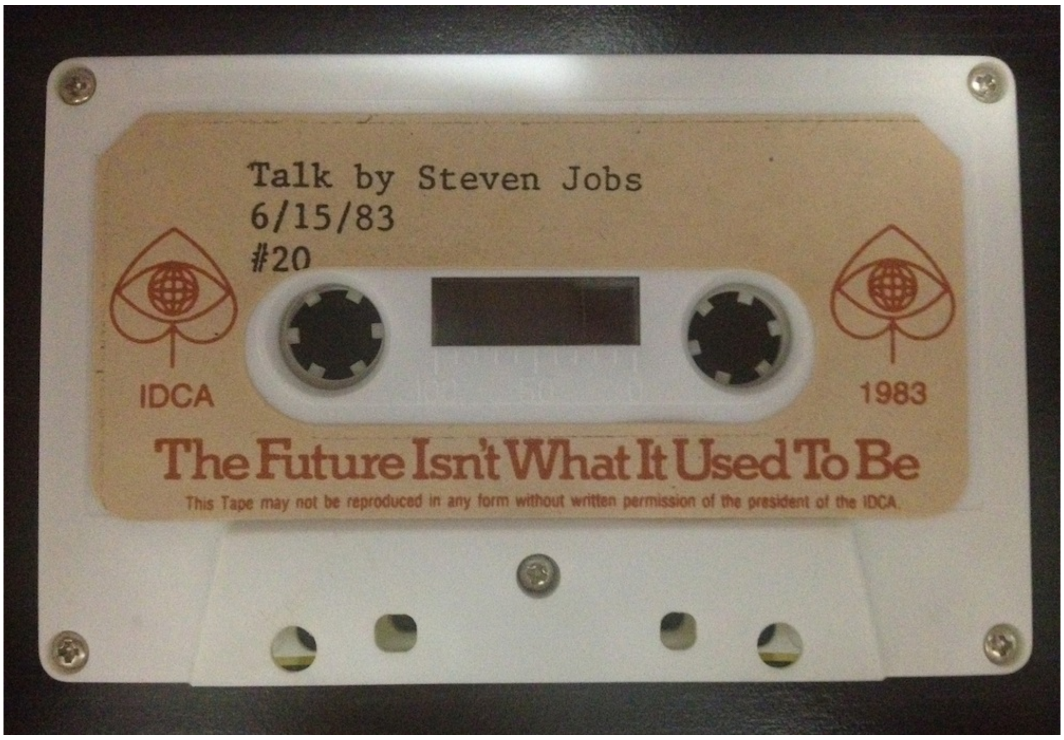
Talk by Steven Jobs
6/15/83
#20
IDCA
1983
The Future Isn't What It Used To Be
This Tape may not be reproduced in any form without written permission of the president of the IDCA

Figure 1: The original cassette tape recording of Steve Jobs' talk at the 1983 International Design Conference in Aspen

Source: http://lifelibertytech.com/wp-content/uploads/2012/10/Talk-by-Steven-Jobs-Cassette.jpg; Courtesy Marcel Brown

For sure, Jobs would have made a great lecturer for the first year 'What is Media Studies' course, had he not wanted to start developing his business empire. This is not merely a joke but a reference to how in this talk and throughout Apple's early business strategy the focus was on education and pedagogy: free computers to schools and the early education of children to become users of the new smart future of computerised learning. Jobs's concern was constantly about new generations—of creating conditions for psychotechnical drilling and training in the emerging computer culture.[13] Jobs, the software-pedagogue, is continually concerned with the user, techniques and habits, and with tapping into what could be called the involuntary part of the human being: ways of drilling the user into suitable conditions of use. This context flags the need to map the wider set of relations between institutions or corporations, interface technologies, business strategies and psychotechniques as the cultural techniques of cognitive capitalism. From the focus on early film theory, such as that of Hugo Münsterberg (1863–1916), we move to the digital technologies of habituation and drill: ways of seeing, acting, gesturing that perform the reproduction of the worlds in which relations of value production and exploitation can tap into the bios and zoe of living bodies and the social relations that characterise life in digital culture. This is also something that crosses into the field of cognitive sciences as well as design, even if I have to neglect that aspect here.

In the talk, Jobs demands a break with the old habits of media consumption and techniques, and he encourages planning new ways of engaging with software worlds. His encouragement sounds like it is addressed to both the producer and the user. Optical video disks, able to hold tens of thousands of images, or an hour of video, should not be solely used to play a movie but should also be used to take full advantage of its random access possibilities, he insists. We should learn more about the affordances of technologies—what their potentials are, and what the technological future might look like. The Lisadraw software is what, in his own words, allows Jobs to try to be an artist and deal with visuals as objects—to move, shrink, change textures, airbrush, soften and harden edges—all of which makes him realise how the talentless can draw. Besides the implications noted by a range of commentators about the renegotiations of talent, skill, artistry and, of course, *creativity* in post-Fordist economies and software culture, we are able to realise what drives Jobs's vision of code. For him—and I am not wanting to say that his views are that exceptional but perhaps just a way to tap into an earlier phase of emergence of tools, or environments, or indeed 'media' of cognitive capitalism—code makes things happen. It is an affordance of sorts. It is not only a tool but a whole mode of thought and platform in which potentials of use and consumption are created. Code is a condition of existence. It frames life. And yet, what Jobs doesn't discuss is what then affords software.

We should pursue the question 'if code makes the world, then who makes code, and what sustains such operations?' I want to trace a slightly longer line of thought that relates to code, code work and organisational ideas concerning software and digital culture. In this idea of programming, which relates not only to code but the user and producer as well, we have other examples from the burgeoning new media culture of the 1970s and 1980s. '*We need to be able to program the programmers too'*—a striking emphasis voiced by Charles Simonyi, a Hungarian-born computer scientist working at PARC Xerox Palo Alto labs in the 1970s.[14] Indeed, our computer histories often mention only the more famous products of Simonyi and others at PARC—the Bravo text editor, for instance, and the work that contributed in part to the success of Microsoft (Word) and Apple corporations. They relate to the technological solutions and the spirit that Jobs later captured as part of his hegemonic vision of digital culture: users are productive and creative and with the help of systems that combine efficient memory management one can develop more complex but easy to use computer graphics content to release our brain power. Instead of Bravo, I want to focus on ignored aspects of the work at PARC and the 'software factory' that produced Bravo—the question of what grounds the work on text editors and computer interfaces and, more generally, software itself. This means focusing on the notion of metaprogramming. In other words, how do you organise work of and in software environments, and program the programmers: how is software work organised as a factory?

This is where the already hinted at approach of cultural techniques comes into play, as well as that of cognitive capitalism, as one description that refers to the massive mobilisation of the brain, cognition and human communication as a production force for capitalist accumulation. The term refers to the mode of capture inherent in that smartness of this 'new' phase of capitalism; that processes of creativity, communication, expression, artistry, innovation and 'fun' are at the core of value production.[15] Technology becomes less associated with the repetitious dullness of factories and is constructed as an environment for self-expression. Or perhaps the infamous factory is distributed across a whole field of the social, and hence the social in its wider sense is participating in the production of value for the capitalist accumulation.[16] In any case, it's pitched as more *fun* and *innovative* than ever before.

—III

Cognitive capitalism, as used by Yann Moulier Boutang, refers to the widespread cultural and corporate harnessing of 'invention power'. It features as an integral part of current capitalist modes of organisation of production and value creation. So where does invention reside and take place? It becomes visible and perceptible in the amount of investment in education, training and other enhancements of social and cognitive capacities. It also persists as a way to infiltrate the social more widely. Our ways of talking, acting, gesturing, remembering, expressing, exploring and thinking are part of the wider picture in which capitalist organisations, and processes of accumulation, are taking into account *externalities*.[17] That capitalist accumulation is dependent on externalities is not new *per se*. Externalities referring to 'collateral effects, by-products or joint production', are, of course, an essential part of any social activity as people in cultural and media studies might take for granted.[18] But this realisation does not feature in economic theory textbooks that often. Boutang insists on the importance of accounting for externalities, not only in the classical sense of taking into account negative externalities (industrial pollution being a common one) but also positive ones and their role in the commodity production and the wider milieu of production incentives. Think of gentrification and cafes, restaurants and other services for the younger creative class—a topic so thoroughly discussed in context of creative cities. Jobs and Apple attempted similar manoeuvres in the early 1980s: if you create such conditions where kids have access to computers, they are obviously more likely to get hooked. Boutang talks less of children, and his example is tied to digital culture as well. He writes about the symptomatic case of the programmer in Silicon Valley who, stuck on a software problem, leaves his cubicle for the shared social space where other 'creatives' hang around and 'their conversation triggers something in your mind that enables you to solve a problem that you've been working on for months'.[19]

Communications in shared spaces count as externalities. They are conditions for the possibilities of sharing and all other characteristics that are now pitched as essential for this cognitive sort of labour on which capitalism rests. Of course, to an extent, some of the discourse on the common has tackled related themes. Years of post-Fordist theory, autonomist Marxism, and theorists from Hardt and Negri to Paolo Virno and Maurizio Lazzarato, Matteo Pasquinelli and Tiziana Terranova have offered their primarily Italian arsenal of concepts as insights into the cultural processes at the core of labour–capital relations in digital culture. Yet the question as it relates to technology persists and perhaps demands a more accurate fine-tuning, especially when considering software.

Boutang insists on the specificity of a digital phase of capitalism, which, he argues, is not reducible to actual technologies. Boutang claims that technologies are only a necessary, but not a sufficient, condition. Instead, this 'mutating capitalism' is less about muscles and more about brains—the Chattering, or Tweeting, Man, instead of the Factory Man.[20] This leads to the realisation that Boutang advocates: cognitive capitalism is not solely about the technological, it is also about the 'appropriation of knowledge' and 'the use of new information and communications technologies'.[21] In other words, it's more innovation than hardware—or even software. Hence, Boutang also wants to keep his eyes on modes of education supporting the innovation work.

On the economic level, Boutang is able to mobilise a range of interesting insights that discuss the regimes of knowledge and labour in this new mix. In his simple historical model, cognitive capitalism is set as the third phase of a longer development, following mercantile and industrial capitalism. This is also a move towards immaterial capital, knowledge and knowledge economy. The mutations of capitalism are not merely phase shifts with a definite break from earlier periods, but a rearrangement of technologies, infrastructures, skills, procedures and capabilities in relation to their role in value creation.

The short historical framework serves to show ways the link to earlier regimes of material labour is still present in various forms. It also demonstrates that a specific relation to science and knowledge characterises the abstract processes of capital as well as the notion of work and labour. Boutang tries to consolidate communication, social cooperation and creativity into a mode of production. Hence, besides hardware, software or even the notion of wetware, Boutang's version takes into account netware, or net*work*.[22] For him, this means a new paradigm of work that is not merely industrial, but which also consists of the modes of cooperation between brainy individuals. This refers to a sort of bio-production as it draws from the living reality of communication and affective social life.

The primacy of knowledge for value creation is evident in the centrality of political debates around property rights, management jargon, a focus on organisational methods and the wider role that human resourcing as well as pedagogy plays in the corporatisation of various industry sectors.[23] By bringing in elements of communication and networks, we are immediately inside discussions concerning organisations and management of such flexible institutional settings, where work and value creation need to be somehow paired in suitable ways. A lot of this analysis can be seen to be predated by, for instance, Luc Boltanski and Eve Chiapello's analyses of management language and practices since the 1960s, and a move towards a more individualised definition of work in terms of 'creativity, autonomy, and flexibility'.[24]

Boutang's theory needs to be also addressed critically and with an eye towards possible shortcomings of the notion of cognitive capitalism. For instance Steven Shaviro has noted that we should critically interrogate various aspects of Boutang's theory's main claims. Is there really such a definitive and clear break from earlier regimes of industrial labour? What are the harmful downsides of communicative brainwork, such as exhaustion? How does exploitation of non-work hours extend the reach of the corporation to the wider social field of our thinking, doing and gesturing, and harnessing 'free time' as part of value accumulation for the corporation?[25] Indeed, isn't cognitive capitalism merely describing a situation of rather cynical colonialisation of time and affect that reaches the most intimate spheres of subjectivity? What is called 'bioproduction' is actually an effective exploitation of the resources of non-paid time which, however, becomes increasingly central for value production. It is already something visible in, for instance, Terranova's important account.[26]

Let me focus especially on elaborating some points about cognitive capitalism's relation to technology and specifically software and argue that there is cultivation and ensuing exploitation of the most innocuous practices, spaces and techniques to the benefit of this relation of communicative capitalism.

What Shaviro flags, and what has been articulated in length by Matteo Pasquinelli,[27] is the danger of conflating such perspectives with the Silicon Valley idealisation of commons and other shared externalities that benefit us all, without being able to account for the actual unofficial investments in work and other practices to support the labour in/of/for commons. This resonates also for instance with Jodi Dean's term 'communicative capitalism' which flags the conflation of democratic ideals with technological infrastructures.[28] The dirty, libidinal side that Pasquinelli is after does not feature in Boutang's more polished view, which, I argue, would benefit from more rigorous media studies perspectives. By now, it should not come as a particular surprise that I want to discuss it in terms of cultural techniques. How to combine the two sides of European theory: the German love for the machine worlds, and Italian/French analysis of politics of the immaterial kind, but also attached to the world of network cultures?

—IV

As one way of articulating this question of the mediatic in cognitive capitalism, I want to pitch the idea of *cultural techniques of cognitive capitalism*. This is merely a theoretical opening with one case study relating to software and labour; it is in need of further elaboration by more empirical and historical work. Such a notion of cultural techniques of cognitive capitalism is a cross-fertilisation of two traditions that have not spoken to each other too much yet. I am referring to the political theories coming often from the tradition of Italy and their developments in France—the Autonomia-movement and post-Fordist political analysis that I have just nodded towards and which one has to admit is a very heterogeneous field. And at the other pole of this theoretical crossbreeding I introduced German media theory, which is not only represented by Friedrich Kittler, or media archaeology (of for instance Siegfried Zielinski or Wolfgang Ernst), but also by such terms as 'cultural techniques' as discussed by a range of scholars from Bernhard Siegert to Markus Krajewski and Cornelia Vissman to Sebastian Vehlken.[29]

As I have already briefly introduced, cultural techniques can be seen as ways of extending the media material ethos to a wider set of concerns where media are related to 'ontological and aesthetic operations that process distinctions … which are basic to the sense production of any specific culture.'[30] More specifically and perhaps more enlightening is Siegert's emphasis that not all *a priori* conditions of culture are technical *a prioris*, 'but involve the materiality of media in the broadest sense'. We can map the different operations that link humans and media: material media studies is once again able to offer a way to understand society.

Speaking of contemporary creativity and, for instance, software from a discourse of fun we can turn to the management of creative work. This includes the management of invention that runs across the contemporary cultural techniques of capitalism as a theme that is grounded in new forms of exhaustion, besides the at times enthusiastic—and hence inflationary—excitement for brain power and cooperative peer subjectivity. This aspect has been discussed in recent years by Franco Bifo Berardi but also relates to the thesis proposed by Bernard Stiegler: instead of smart cultures of skilled professionals in communicative industries, we should acknowledge the systematic stupidity and proletarianisation of 'creative' work. In Stiegler's words: 'We thus have pure cognitive labor power utterly devoid of knowledge: with cognitive technologies, it is the cognitive itself which has been proletarianized.'[31]

If we pursue this discussion concerning factories, the proletariat and the grey repetitious aspects of communication, we end up in a discussion of techniques and operations. I argue that we need to pursue this in terms of the operations that offer a link between human practices, media and society, as well as political economy.

I want to flag this as a necessary supplement to the more optimistic, and perhaps more 'Californian', take by Boutang. Indeed, one needs a slightly more evil approach, in the manner proposed by Matthew Fuller and Andrew Goffey. They call this evil media studies.[32] As part of their larger project, Fuller and Goffey take up on what they pitch as grey media; the tiny details, glitches and repetitious occurrences from current software and social systems, which in their minute detail can be seen as defining a whole field of interest in management cultures as media. Spreadsheets, workflows, auditing, call centre procedures all become understood as part of the constitution of the reality of management—and management as the reality of media.

It's not that technology is entirely missing from Boutang's theory. He is a fan of open software projects, and complements his economic analysis with examples of techniques of the social. One such example includes the call centre. For Boutang, telephone help centres offer an example of the joint project between database-based organisation of possible solutions and the mediating role of the actual worker. The worker is the curator, in real time attending to the needs of the customer and not only through mechanically finding the suitable solution from the database. As a creative worker of sorts, she or he has to demonstrate innovation by identifying 'cases that fall outside standard practice' and, where possible, 'offer new viable solutions'.[33]

But from an 'evil media' perspective, call centres are endless sources of delay and streamlining, of evasion and systematised boredom. Indeed, through the various steps in the formalisation of (natural) languages into modes of expression suitable to algorithmic logic—and, besides language, also gestures and patterns of cognition—one enters a grey zone where any celebration of virtuosity of language at the centre of the multitude subject sounds slightly exaggerated. Indeed, call centre work is extensively about management of social relations in relation to databases and algorithms, but in ways that produce 'the cognitive' as a matter of regularisation and disciplinary formalism. This means a social patterning that refers to a perversely idealised version of the real world that we call 'work flows'. Unfortunately, such idealisations found in organisational charts rarely actually correspond to the *real* world if one understands by that the number of exceptions, bugs, errors, mishearings and other events that occur amid actions and expressions, which constitute the core of Fuller and Goffey's evil media theory.

A look at history of software cultures as one of management reveals this other side of cognitive techniques and information work. They are part of *a media history of cognitive capitalism, or cultural techniques of cognitive capitalism*. But this is not meant as a valorisation or a rosy picture of brains in cooperation but rather patterns of management, and organisational operations and abstractions. If we really want to paint an accurate picture of such post-Fordist theory concepts, we need to take a look at the grey media and cultural techniques. These can include an analysis of techniques of interfacing from Douglas Engelbart to contemporary brain-computer-interfaces, as well as the psychotechnics of perception from film to augmented reality. The list could be continued, and in any case relates to the entanglement of perception or sensation with technological conditioning in a political economic setting of capitalist organisations and institutions.

What is often focused on with user and interface design, as in the emergence of Apple-centred digital culture discourse since the 1980s, can be complemented with the less fun aspects, such as some of the work at Palo Alto labs in the 1970s. For sure, these examples have not been altogether ignored, but the focus of attention has usually been on end-user-oriented innovation such as emphasising user-centered products like the Bravo-editor and interface.[34] PARC worked towards user creativity through meticulous ways of managing memory, graphics innovations and technical-conceptual innovations like the WYSIWYG. Instead of taking the obvious route let's focus on some other aspects which might, however, be more methodologically boring. We can take Stiegler's theoretical note as a guideline and track the specific procedures of proletarian approaches to the cognitive, and expand on the theme of software work as factory work. I will do this by briefly exploring one specific example of a research theme and practice at PARC: metaprogramming, or programming the programmers. This is in no way a dominant strand in software culture but it can help us to think through some connections between software, organisation and work: the software factory ensuring the fun for the end user.

—V

Instead of seeing user/creativity-oriented discourse as the only option, we also need to be able to account for other sorts of techniques. In this case, we can point towards software understood as management and production. This is evident in the ways programming has been thought of as work and production.

An exemplary early formulation is found in the already mentioned Hungarian emigrant Charles Simonyi's notion of metaprogramming, which hints towards David Hilbert's 1920s rethinking of mathematics as metamathematics in true modernist spirit of rationalised communicability. Also part of the 1970s Xerox Palo Alto Labs' activities and research, such approaches present interesting insights not so much into the celebration of innovation, creativity and software, but into issues of management, training and organisation that escort such notions into popular consciousness and as rhetorical refrains of digital capitalism. Hence, I want to offer a sort of an anti-Jobs pitch: we should aim a more grey, more evil media focus on cultural techniques of the cognitive.

Over the past few years, interest in coding has experienced a strong upward surge where the labour and even potential dullness of code crunching is entangled with a cheerful discursive embracing of new modes of collaboration, adaptability and even 'fun' that accompany the most discussed software management projects, that is, free and open software, as well as gaming.[35] This is also apparent in political and industry emphasis on wanting kids to get into coding: the word 'fun' appears often in reports and motivational material for various school and after-school activities. In general, coding is coded as something of an exceptionally exciting activity that even cynical teenage kids might engage with.[36] Working with software is inspirational and is indeed often seen as a form of self-expression, distanced from the laborious sides of code work. Everyone can code, and coding is a way to release potentials of expression. In the United Kingdom, this emphasis was primarily established by the current Tory-liberal government, whose education plans emphasise the need to increase skills and awareness of programming in school syllabi. The thinking behind such an emphasis was rather straightforward: coding might not feel like work but it pays off, and is the basis for a vision of a Digital Britain.

If we return to the 1970s situation, and code as work, for Simonyi, metaprogramming is about improving productivity. This happens by letting the *inspirational* programming happen on a very abstract conceptual level by a team leader, then implemented by technician-assistants. This narrows the creative aspect to a very limited role, where most of the work is just careful implementation of the broad-stroke approach.[37]

In short, metaprogramming is about managing the complexity that goes by the name of code and software. It is just one particular mode of understanding software work and management, but it can give us clues to think about software work in other contexts too. Simonyi's idea was rather traditional and seemed to point in a direction different from the user-centered and horizontal notions which are now celebrated but which perhaps hide the fact that software work can also be boring and exploitative—in the gaming industry and other creative fields. Simonyi's version of code work corresponds more widely to how code and labour entangle in corporate teams and under tight scheduling and monetary pressures.

Simonyi's definition of metaprogramming is in this sense about describing 'an organisational schema, designed to yield very high programming productivity in a simplified task environment which excludes scheduling, system design, documentation and other engineering activities'.[38] This actually means transferring engineering tasks into localised work groups where high-level skills are not cognitively necessary. Those groups need only the basic technician skills. This is where communication between separate work groups becomes crucially important, not only for the content that is communicated. How does one talk of code projects and software products? How can the language necessary for programming- and engineering-related activity in a design organisation be standardised? According to Simonyi the problem is to work around local languages and through a more universalised take on communication about, and naming of, the objects of communication. Software becomes a language, but in a different way, perhaps, from what Manovich meant.[39] In our case, language is a mode of command, control and management for the benefit of organisational efficiency. This does not mean a completely rigid system. Simonyi leaves this slightly more open when offering his definition of metaprograms as 'informal, written communications, from the meta-programmer, who creates the local language, to technicians who learn it and actually write the programs'.[40] But it still serves the purpose of standardised procedures, which sustain cooperative, synchronised communication.

More specifically, in software production this means a focus on aspects other than the long dominant modes of measuring lines of code per hour. Simonyi's early 1970s work marks a gradual shift in understanding coding and productivity. Having said that, the earlier mode of quantitative measurement is not entirely abandoned and forms the statistical part of Simonyi's study.[41] Metaprogramming is to be understood through confining the creative aspects of coding to one particular role. The metaprogrammer is the creative coder, the prototyper who writes sample code and feeds it to the technicians for implementation. The technician-coders work under the metaprogrammer and feed back in a cybernetic loop, being managed tightly and efficiently. Hence, this could be seen as a subsumption architecture of sorts: the laborious aspects of coding and even quantitative measurement is subsumed under particular roles in the organisation, and the creative part is limited to the metaprogrammer-manager at the top.

One way to understand metaprogramming is to think of it like this: in the manner that before computers were machines, they were human, often females, as before and during World War II.[42] Computers were about teams of humans managed as computational units. The organisation and management of humans labouring as computers was what defined that early 'computer architecture' before some technical solutions changed the role of the human programmer in relation to the machine that was now able to number crunch. Metaprogramming implicitly simulates the development of software language, transposed as an organisational diagram. Metaprogramming is about the programs coded, but also about coding the humans as computational aspects of the organisation. The organisation is the modern software computerised environment: abstracted higher level languages are where programming is seen through tentative planning; third-generation languages are actually the metaprogrammer, who then feeds such plans to the compiler; source code is command, but one that needs to be fed, processed and enacted so that it becomes what it attempts to be—a source.[43] Metaprogramming crystallises the idea of software at a particular point in history; it codes computers and people. Coding is labour, and software becomes a product. The materiality of such technological processes involves both ends: materialities of technology and materialities of labour. Techniques of managing and organising labour become increasingly central. Simonyi's thesis pays attention to the practices of educating and training staff in various ways; a lot of this training has to do with organisational commands, themselves cultural techniques, modes of managing the commands, communicating and error tolerance. They are modes of managing and responding to the patterns of organisational logic. The metaprogrammer might execute a command but only to reach the technician—the technician being the level of support for the higher-level functions so to speak, and a level which is defined by its own operations that are the object of training.

Issues of training, communication and management of work and even flexibility are important in this schema. It is not only the amount of executable code produced that is measured, but also the metaprograms—the creative work. This has to be managed in terms of 'reliability, user acceptance and modifiability of the products'.[44]

Indeed, this means the transfer of risks and costs in failure onto a safer level of planning before the technological implementation kicks in. In Simonyi's words:

> we have proposed uncertainty absorption for improving the software industry's ability to deal with the uncertainties inherent in large software problems. Uncertainty absorption—the promise of action which enables others to operate free of the uncertainty—is particularly simple when production can be organized as a continuous process which can be measured, controlled, and hence, optimized. We divided the software production task into an engineering phase, in which the user's problems are made well defined; and production phase, in which proto-software is produced by a continuous process. The proto-software is given back to the engineers for refinement to create the final product.[45]

What is fascinating is how the quote from Simonyi pitches the relevance of such a technique like writing 'proto-software'. It signals also important aspects concerning labour and organisation. I argue that Simonyi's ideas lead us to think about the wider set of techniques that mobilise ideas about code, software and work. Coding expands to the wider set of communicative tools with which the organisation has to function in relation to customers. What needs to be recognised, even from this short take on metaprogramming, is how considerations of software are entangled with other sorts of cultural techniques. In other words, where software culture is introducing its own range of new and re-emphasised modes of managing the world—from sorting to abstractions (including pseudocode and in this case 'proto-software'), error checking and debugging—it is also embedded in how we talk about, read and measure such activities as part of the organisation of software work. This relates further to the issue of measurement of productivity. For Simonyi:

> Productivity is traditionally defined as the relationship between the output of goods and services and the inputs used in their production. Applied to software production, the output of program bulk should be expressed as a function of the inputs: the time of programmers and other labor and possibly overhead costs.[46]

Whereas automation can be pitched as one way of improving such productivity,[47] one can start to understand the need to manage the cultural techniques of reading, writing and communicating involved in the productive process: from a controlling interest in practices of naming software objects in order to streamline communication between task and role groups, to the psychotechnical measurement of what we do in situations of work. Simonyi refers to other research, noting 'the observed programmers spent 14% of their time reading and 13% writing "with a list, card, or worksheet in evidence", that is in "productive capacity". "Talking or listening (Business)" took 17%.'[48] Hence, it becomes clear that tapping into the constituent use of techniques at work becomes a way to manage productivity so long as the idea of the creative process becomes tightly focused in the management scheme.

—VI

Coding is not necessarily very creative or fun. The fascinating role of the metaprogrammer doesn't necessarily even need access to computers, and becomes more that of a general systems planner in a cybernetic sort of organisational diagram. The emphasis on communication and training to specific roles demonstrates no clear-cut '*post*-Fordist' mentality of a transition from factory-based hierarchical management to a more horizontal method of connecting the brain (to adopt to Boutang's discourse). Instead, the team-based organisation we find in such early plans as 1970s metaprogramming testify to how differently software production can be understood—not as the horizontal clustering of creatives, as the mantra goes, extending to the idea of the creative end user and the flexible worker, but as the management of various roles, some of which are still quite straightforward technician jobs. Such a realisation might ring true for a range of current software organisations and companies—in the gaming industry, for instance. Whereas this view resonates with a range of cultural techniques in software and coding work in the current economic and organisational settings as well, it also hints at a different lineage from that of Boutang's theory and some other positive accounts of creative industries and digital economy. Instead, what if we in a way think of the celebration of the creative individual, the post-factory artist type, only as a short entr'acte[49] in the history of organising knowledge production? Even in early formulations—before programming meant software as a separate 'entity' from hardware—the laborious nature of programming was underlined. Innovation is embedded in the reality of labour—'dull labour' to be more accurate. Computer pioneer Grace Hopper's words from 1950s are strikingly apt: 'The novelty of inventing programs wears off and degenerates into the dull labour of writing and checking programs. This duty now looms as an imposition on the human brain.'[50]

The idea that cultural techniques of software culture and work are actually often much more physical, repetitious, uncreative and based in rather strict management, disciplinary and formalisation procedures needs further analytical attention. We are gradually realising that digital culture is sustained by hard and repetitious manufacturing processes outside the creative industries circle—for instance, the Foxconn factories in China—but the realisation that creativity is embedded not only in precarious but also in rather repetitious and tiring practices needs to be taken just as seriously. It is in this sense that the certain immaterial production part of cognitive capitalism is completely material to the point of exhaustion. In other words, we focus on the cognitive not only as concerning the thinking brain but as it concerns the wider set of techniques in which brains are connected to bodies and bodies to work patterns, and in which work patterns are set in a complex group of organisations that are abstract ways of tying the different techniques together. Besides specific case studies relating to the entanglement of code and labour, the theme of techniques of the cognitive point towards the need for political media studies, in the sense defined by Jonathan Beller: 'A political and politicising approach to media studies would insist upon the materiality of mediation as well as reckoning of the material consequences of even the most ostensibly immaterial and abstract mediations.'[51]

A cultural technique approach to code, software and digital culture might take on board notions such as cognitive capitalism, but also pokes the grey mass of the cognitive through excavations into what sustains it. What sort of techniques and technological practices are behind what we call the cognitive, and what sort of archaeological excavations are we able to engage with when we want to understand the differing uses of communication and organisation? An analysis of cultural techniques becomes then a way to extend into issues that are historical, mediatic and, indeed, political.

—

Jussi Parikka is reader in Media and Design at the Winchester School of Art, University of Southampton. He is the author of several books on digital culture and media theory, including *Digital Contagions* (2007), *Insect Media* (2010) and *What is Media Archaeology?* (2012). He is also the co-editor of *The Spam Book* (2009). <http://jussiparikka.net>.

—NOTES

[1] Jussi Parikka, 'Cultural Techniques and Media Studies. An Afterword', *Theory, Culture & Society*, vol. 30, no. 6, 2013.

[2] Bernhard Siegert, 'The Map is the Territory', *Radical Philosophy*, vol. 169, September/October 2011, p. 14.

[3] See for instance Andy Clark, *Natural Born Cyborgs: Mind, Technology and the Future of Human Intelligence*, Oxford University Press, Oxford, 2003. See also Katherine N. Hayles, *How We Think: Digital Media and Contemporary Technogenesis*, Chicago University Press, Chicago, 2012.

[4] See for instance Donald A. Norman, *The Design of Everyday Things*, MIT Press, Cambridge, MA, 1998.

[5] On media archaeology, see Erkki Huhtamo and Jussi Parikka (eds), *Media Archaeology: Approaches, Applications, and Implications*, University of California Press, Berkeley, 2011. Jussi Parikka, *What is Media Archaeology?* Polity, Cambridge, 2012.

[6] Till A. Heilmann, Textverarbeitung. Eine Mediengeschichte des Computers als Schreibmaschine Transcript, Bielefeldt, 2012, p.78. Heilmann is quoting Turing's 1948 paper 'Intelligent Machinery'.

[7] Quoted in: Ilinca Iurascu and Geoffrey Winthrop-Young, 'Cultural Techniques: Preliminary Remarks' forthcoming in *Theory, Culture & Society*, 2013. Originally the quote is from Thomas Macho, 'Zeit und Zahl. Kalender- und Zeitrechnung als Kulturtechniken' in (eds.) *Bild – Schrift – Zahl,* eds Sybille Krämer and Horst Bredekamp, Wilhelm Fink, München, 2003, pp. 179–92.

[8] Yann Moulier Boutang, *Cognitive Capitalism*, trans. Ed Emery, Polity, Cambridge, 2012.

[9] See the special issue on Cultural Techniques in *Theory, Culture & Society*, vol. 30, no. 6, November 2013 and edited by Geoffrey Winthrop-Young, Ilinca Iurascu and Jussi Parikka.

[10] Sybille Krämer and Horst Bredekamp, 'Culture, Technology, Cultural Techniques—Moving Beyond Text', trans. Michael Wutz, *Theory, Culture & Society*, forthcoming in 2013.

[11] On recent relevant research that excavates the global labour involved in internet cultures, and the expansion of notion of labour, see Trebor Scholtz, *Digital Labor: The Internet as Playground and Factory*, Routledge, London and New York, 2013.

[12] Marcel Brown, 'The "Lost" Steve Jobs Speech from 1983: Foreshadowing Wireless Networking, the iPad and the App Store', *Life, Liberty, and Tech*, 2 October 2012, <http://lifelibertytech.com/>.

[13] See Bernard Stiegler, *Taking Care of Youth and the Generations*, trans. Stephen Barker, Stanford University Press, Stanford, 2010.

[14] Charles Simonyi, *Meta-Programming: A Software Production Method*, Xerox Palo Alto Research Lab, Palo Alto, 1977.

[15] See, for instance, the discourses of fun in programming and gaming related to industries that produce interactive entertainment and digital entertainment. This brands various levels of the discourse about programming from marketing to public discourse related to the digital economy initiatives in Britain, which aim to get youth into programming. It is in this sense that 'fun', despite its vague denotative nature, is actually one of the most significant markers of digital discourse, whether around entertainment, learning or even work in software cultures.

[16] See for instance Michael Hardt and Antonio Negri, *Empire*, Harvard University Press, Cambridge, MA, 2000, pp. 29, 53, 196. For a critique of Negri, see Tiqqun, *This is not a Program*, Semiotext(e), Los Angeles, 2011. Hardt and Negri's *Empire* was instrumental in establishing a focus on the wider social factory and the role of affect and communication as biopolitical modes, something they referred to as the postmodernisation of global economy (p. xiii).

[17] Boutang, pp. 20–30.

[18] Ibid., p. 22.

[19] Ibid.

[20] See also Jussi Vähämäki, *Kuhnurien kerho. Vanhan tyon paheista uuden hyveiksi*, Tutkijaliitto, Helsinki, 2003.

[21] Boutang, p. 42.

[22] Ibid., p. 53.

[23] Ibid., see p. 57

[24] Jodi Dean, *Communist Horizon*, Verso, London, 2012, p. 61.

[25] Steven Shaviro, 'Cognitive Capitalism?', *The Pinocchio Theory-blog*, 3 February 2008, <http://www.shaviro.com/Blog/?p=620>. On exhaustion, see Franco Bifo Berardi, 'Exhaustion/Depression' in *Depletion Design: A Glossary of Network Ecology*, eds Carolin Wiedemann and Soenke Zehle, Institute of Network Cultures, Amsterdam, 2012, pp. 77–82.

[26] For the updated version, see Terranova, 'Free Labour' in *Digital Labor: The Internet as Playground and Factory*, ed. Trebor Scholtz, Routledge, London and New York, pp. 33–57. For an analysis of the temporal regimes of contemporary capitalism, see Jonathan Crary, *24/7: Late Capitalism and the Ends of Sleep*, Verso, London and New York, 2013.

[27] Matteo Pasquinelli, *Animal Spirits. A Bestiary of the Commons*, Institute of Network Cultures, Amsterdam, 2008.

[28] See Jodi Dean, 'Communicative Capitalism: Circulation and the Foreclosure of Politics', *Cultural Politics*, vol. 1, no. 1, 2005, pp. 51–74.

[29] For an English introduction to this discussion, see *Theory, Culture & Society* special issue, forthcoming in 2013.

[30] Siegert, 'The Map is the Territory', p. 14.

[31] Bernard Stiegler, *For a New Critique of Political Economy*, Polity, Cambridge, 2010, p. 46.

[32] Matthew Fuller and Andrew Goffey, *Evil Media*, MIT Press, Cambridge, MA, 2012.

[33] Simonyi, p. 74.

[34] The key study of Palo Alto research is Michael A. Hiltzik, *Dealers of Lightning: Xerox PARC and the Dawn of the Computer Age*, Harper Business, New York, 1999.

[35] As Chris Kelty well outlines in his extensive ethnography, such projects have been seen exactly as a way out from the mode of central planning to the emergent intelligence of distributed labour. The voluntary nature of projects such as Linux or Apache presents a case for considering such human energies articulated together with a range of technologies and techniques that are essential to sustaining the projects, and even the sense of community in them. Kelty talks of the recursive nature of such modes of organisation inherent to free and open software groupings, referring to how the public that constitutes the software through its participation is also itself constituted as part of the development, 'making, maintaining and modifying'. Christopher Kelty, *Two Bits. The Cultural Significance of Free Software*, Duke University Press, Durham, 2008, p. 29. The technology is itself grounding the public nature of the project, which in cases such as Linux can be tracked to the co-influence of social modes of organisation (and changes taking place since the earlier days of Linus Torvalds and the lieutenants as filters for acceptable modifications to the modularisation of the Linux project) and technological bits: source code management systems being the obvious example here. See also David Berry, *Copy Rip Burn: The Politics of Copyleft and Open Source*, Pluto Press, London, 2008.

[36] There are various examples from coding courses and academies to public (media) discourses. These are admittedly a heterogeneous bunch, but see for instance *Code Year* <http://www.codeyear.com/>, *Code Academy* <http://www.codecademy.com/>, *Google's Summer of Code* <https://developers.google.com/open-source/soc/>, 'Fun With Coding', *BBC Technology*, 22 June 2012, and '7 Sites that Make Programming for Kids Fun', <http://www.fractuslearning.com/2011/12/14/programming-for-kids/>.

[37] Hiltzik, pp. 197–8.

[38] Simonyi, p. i.

[39] Lev Manovich, *The Language of New Media*, MIT Press, Cambridge, MA, 2001.

[40] Simonyi, p. 26. In general, the transportability of existing instructions and code in programming environments and organisations is an important part of the cultural techniques of coding. An early example of communication and communicability in such work environments of, for instance, mainframe computing already was the use of code books which listed useful subroutines that could be

then reused: 'If I needed a sine subroutine, angle less than π/4, I'd whistle at Dick and say, "Can I have your sine subroutine?" and I'd copy it out of his notebook.' Grace Hopper quoted in Tillmann, p. 69.

[41] Appendix E in Simonyi, pp. 127–34.

[42] Jennifer S. Light, 'When Computers Were Women', *Technology and Culture*, vol. 40, no. 3, July 1999, pp. 455–83.

[43] Wendy Hui Kyong Chun, *Programmed Visions: Software and Memory*, MIT Press, Cambridge, MA, 2011, pp. 19–54.

[44] Simonyi, p. 84.

[45] Ibid., p. 19.

[46] Ibid.

[47] Ibid., p. 22.

[48] Ibid.

[49] For the methodological idea of the 'entr'acte', see Siegfried Zielinski, *Audiovisions. Cinema and Television as Entr'actes in History,* Amsterdam University Press, Amsterdam, 1999.

[50] Grace Hopper, 'The Education of a Computer', *Proceedings of the ACM National Conference* (Pittsburgh), ACM, New York, 1952, p. 243.

[51] Jonathan Beller, 'Digital and the Media of Dispossession' in *Digital Labor: The Internet as Playground and Factory*, ed. Trebor Scholtz, Routledge, London and New York, 2013, pp. 182–3.