ephemera | articles
theory & politics in organization

ISSN 1473-2866
www.ephemeraweb.org
volume 9(2): 93-112

Stop whining, start doing! Identity conflict in project managed software production*

Peter Case and Erik Piñeiro

abstract In this article we explore the relationship between software developers and IT project managers as expressed through narrative exchanges in an on-line discussion forum. We interrogate a naturalistic data set to show how the conflict between IT professionals and their immediate managers (project managers) is manifest through the identity work that they engage in. To this end, the article draws attention to strategies of resistance and dissent expressed in the narratives of software developers, contrasting these with the performative expectations espoused by project managers. The purpose is to contribute to a critique of project management stemming from the grassroots experience of those involved in its co-construction. While it is difficult to be precise about the demographics of the community studied (given the anonymity of bulletin board forums), the views of several hundred participants are represented in the discussion threads analysed. In response to the performative environments and disciplines of project management, programmers make recourse to performative strategies (in an Austinian sense) that preserve their status, sense of professional identity and organizational power. The aesthetics of programming appear to play an important part in the expression of programmers' identity; aesthetics which contrast, and are in conflict with, different forms of performative aesthetics present in the identity work of project managers.

Introduction

The encounter between workers and management has been subjected to a wide range of intellectual and academic analysis within organization studies. Accounts of the relationship range from texts that espouse functionalist visions of organization (inter alia, Daft, 1995; Donaldson, 1996; Schein, 1985) which seek to downplay or occlude the role of conflict through to Marxist-informed Labour Process and radical structural theories in which conflict is understood to be integral to industrial relations (Burawoy, 1979; Burrell and Morgan, 1979; Edwards, 1986; Thompson and McHugh 1995, Thompson and Ackroyd, 1995). In the last two decades or so, organization studies has been subject to a post-structural turn (Burrell 1988, 1998; Cooper 1989; Hassard and

* A previous version of this article was presented at the 'Making Projects Critical 4' conference, 31st March - 1st April 2008, Royal Institute of Technology, Stockholm, Sweden. The authors wish to thank the organizers, guest editors and anonymous reviewers of *ephemera* for comments on the original paper and to acknowledge questions and observations from delegates which helped refine the argument.

Parker, 1993; Chia, 1996, 1998) which has introduced a new set of analytical tools with which to undertand the dynamics of the employment relation. Debates between proponents of more orthodox Labour Process accounts of worker/management relations, which understand conflict in terms of fundamental differences of vested class interest, and those of post-structuralists of varying complexion, who propose subtler and more fluid interpretations, have been hotly pursued (Knights and Willmott, 1989; Thompson and Ackroyd, 1995; Thompson and Smith, 2000-1; Knights, 2001; O'Doherty, forthcoming; O'Doherty and Willmott, 2001a, 2001b; Friedman, 2004).

In this article we seek to make a modest contribution to such debates by attending empirically to the manifestation of conflict between software developers (with a focus on coders – or programmers – as opposed to system architects, analysts, etc.) and project managers, who most often are their immediate superiors in the corporate hierarchy. Our study also attempts to augment the existing organization studies literature concerned with knowledge worker identity (Alvesson, 2004; Alvesson and Kärreman, 2007; Alvesson and Willmott, 2004; Kärreman and Alvesson, 2001), focussing particularly on the co-construction of identity conflict. The narrative data that we explore seems to speak to accounts which could be interpreted as representing key aspects of the structuralist account of employment relationships while, simultaneously also being open to post-structural interpretation of power dynamics. One of the key elements in the argument put forward in this article is the particular nature of the relation between programmers and project managers. This particularity revolves around: 1) the difference in the nature of their professional knowledge; and 2) the symbiotic relationship between project management and IT systems.

Knowledge and academic status

According to structuralist accounts, worker/management conflict is undertsood to result from opposed *interests* which are ontologically grounded in a reality of socio-economic class division. Put in crude terms, according to this approach managers (as the actual or symbolic owners of the means of production) want to get as much out of workers for as little pay as possible and workers, by contrast, want maximum pay for minimum effort. While not speaking directly to the *ontological reality* (or otherwise) or this structural relationship, as we shall see shortly, analysis of our narrative data set offers some support to the idea that worker/management conflict is subjectively and inter-subjectively expressed in terms of this duality. The relationship between project managers and programmers in an IT context, however, introduces subtleties and differences that demand a more nuanced interpretation than a sheerly structuralist or critical realist (Ackroyd and Fleetwood, 2000, 2004) account provides. Employment relationships within IT projects bring with them anomalies that are not present in other contexts. Firstly, for example, IT project managers do not need (and often do not have) educational credentials that exceed those of programmers. Secondly, project managers do not need (and often do not have and cannot aspire to) the technical knowledge of programmers. These conditions give rise to a certain view among programmers about the 'real' power relationship between both parties; a view that places programmers (in their own minds at least) *above* managers in terms of organizational status (Piñeiro and Case, 2008). This we trust will become evident when we examine the narrative data, but

before proceeding to our analysis, we need to offer some further remarks on the particular complexity of organizational relationships that result from the meeting of project management and information technology.

Project management and information technology

Over the past fifty years or so IT has insinuated itself into just about every aspect of the social order in developed capitalist economies to the point where life, as we know it, would be inconceivable or, indeed, impossible in practical terms without its facilitation. Perhaps slightly less pervasive, but increasingly ubiquitous in (post)modern life is 'project management'.

Rather like the word 'strategy', the concept of the 'project' has permeated everyday organizational discourse, stretching its standardized meaning. Everything, from the most complex multidisciplinary endeavour, such as, developing a $48bn International Space Station, through to organizing a children's birthday party becomes, in everyday parlance at least, 'a project'. This may be, in part, due to the fact that the noun form of the word 'project' in English has a longer and more stable legacy than one might think. According to the *Oxford English Dictionary*, the word project finds etymological origins in the Latin *prōiectum* meaning 'projecting structure' (2nd c. A.D.) and *prōiectus*, the 'fact of extending beyond a surface or edge' – a spatial connotation which, in the intervening years, has become a *temporal* one. The word migrates to English from Middle French and takes on the meaning of 'plan' from *proiect* or *projet*. So by the 15th Century it already acquires a relatively stable meaning as '[a] plan, draft, scheme, or table of something; a tabulated statement; a design or pattern according to which something is made'. Its meaning as, '[a] planned or proposed undertaking; a scheme, a proposal; a purpose, an objective' also enjoys a lengthy heritage dating back to the 16th Century. Compounds such as 'project engineer' and 'project management' are, as one might expect, more recent semantic innovations dating only to the beginning decades of the 20th Century. The *OED* defines project management as, '[t]he theory, practice, or occupation of managing projects' and 'the group of people in charge of a project' (*OED* online), the first recorded use being in the *Nevada State Journal* of July 1913.

So what? Should we view the ubiquity of project- and PM-speak as a harmless consequence of (a) historical contingencies of English language development, and (b) its natural and pragmatic extension into other areas of modern life? Or, alternatively, may there be more sinister and compromising implications of this ubiquity and the movement toward a projectification of organizational life? We are certainly not trying to play the role of semantic policemen with respect to language use, but we do think that some reflection on the casuistic stretching of the term and the connotation of its applications in the workplace do warrant some critical attention. In our teaching, we sometimes give MBA students an exercise that entails avoiding use of the word 'strategy' in their workplace communication for one working day. Invariably, they report on the difficulty – if not the downright impossibility – of avoiding the term. Similarly, one might experiment with trying not to use the word 'project' for a day and see what happens. This little exercise would serve to show just how prevalent the

'projectification' of work is in contemporary organizations and give some indication of how the project mentality feeds its way, subconsciously, into a particular construction and interpretation of the world. In short, if one rejects simple correspondence theory and assumes that language plays more than a neutral role in discursively *producing* the social world, there are certain power effects (Foucault 1980) of this seemingly innocuous seven-letter word that we would do well to pay attention to. By thinking in terms of projects and enacting them, we constitute the world in a particular way and embrace a taken-for-granted epistemology and ontology; we embrace a certain way of *knowing* and *being* in the world. Projectification entails, in Foucauldian terms, certain forms of objectivisation and subjectivisation (Florence, 1994) and we shall be concerned in the empirical section of this article to examine how these microphysics give rise to identity conflicts in IT settings.

The development of management and organization science in the last century and a half, or so, has been coupled closely with the evolution of engineering and associated technologies. It is not our place here to fully review the particular history of PM or to offer a detailed critique of its body of knowledge. This has been comprehensively and admirably addressed by others (Cicmil and Hodgson, 2006; Hodgson, 2002, 2004; Hodgson and Cicmil, 2006; Lindgren and Packendorff, 2006) and mere rehearsal of the same points would serve no useful purpose. We would, however, like to make a few remarks on the relationship between engineering technologies and PM since this is directly pertinent to the study we wish to report on below. Shenhav (1999) has made a study of the genealogy of management, arguing that it was engineers in the early part of the 20th Century who 'manufactured management' by effectively creating space between 'capital' and 'labour' to be occupied by a new professional class. Shenhav's is one of the latest contributions to organization studies that have attempted to draw attention to the historical relationship between the evolution of engineering disciplines and the corresponding need to position employees within a rational matrix (Grint 1991; Jacques 1996). Most famously, of course, the genesis of 20th Century management is often accorded to the engineering-inspired manifestos of F.W. Taylor and his drive to develop *The Principles of Scientific Management* (Taylor, 1986 [1911]).

Throughout its history, organization theory has drawn metaphors and imagery from the latest scientific developments in order to help rationalize and make prescriptive sense of management practices. While not wishing to promote a case for technological determinism, as such, there is an undeniable correspondence between scientific and engineering innovation and management thinking. With particular respect to PM, it seems to us that the representational and calculative possibilities that are embodied in IT have been instrumental (both literally and metaphorically) to the emergence of PM's disciplines, its body of knowledge and its set of professional practices. Put simply, PM in its current form would not be feasible without the supporting information technology. The suite of computer programs and spreadsheets that are now the stock in trade of PM – one thinks of software provided by the likes of Microsoft and Primavera to support

PM methodologies: Earned Value Analysis, CPA methodologies and so forth – has given particular form to managerial fantasies, idealism and control aspirations.[1]

So this is the point at which our two ubiquities meet. IT is, we suggest, a necessary (but not sufficient) condition for the contemporary development and direction of PM methodologies. Both IT and PM are intimately intertwined in a kind of symbiosis. The skills and efforts of computer programmers are necessary to the creation of software that now facilitates the professional work of project managers, while, simultaneously, PM methodology is absolutely integral to the development of functioning software packages. This gives rise to a fascinating and complex set of relationships between the respective disciplines of IT programming and PM, some dimensions of which we shall explore through the empirical study presented below.

Data set and method

Building on earlier work (Case and Piñeiro, 2006; Piñeiro and Case, 2008), the empirical foundation of this study lies on naturalistic data derived from an on-line computer programmer discussion forum (www.slashdot.org). We have been visiting and researching the *Slashdot* website periodically for the past five years or so, over which time we have become familiar with both the form and content of bulletin board exchanges. The forum initiates about twelve discussions each day, with subsequent exchanges typically lasting anywhere between a couple of days to a week. The discussion threads we have chosen to study for the purposes of this article draw from threads concerned with 'the aesthetics of coding' and 'project management for programmers' since these contain particularly clear insights into the relationship between programmers, project managers and the PM environment. Bulletin board discussions had already ended by the time we drew on them for analytical purposes and we intentionally made no effort to engage in or to influence on-line exchanges.

The participants, who in most discussions are invariably programmers or other 'techie' IT professionals, may enter the discussion as members of *Slashdot*, in which case their on-line identities are available; or they may access it anonymously, in which case they appear on-line under the shared name of 'Anonymous Coward'. Furthermore, there is nothing that prevents a participant from having several on-line identities. However, *Slashdot* is not an 'identity laboratory' forum (Bruckman, 1992). Unlike other more chat-focused on-line 'hang-outs' and virtual social networks, this is not a place to meet and get to know people. Its design does not allow for chat and picture exchange; topics of discussion rarely touch upon personal subjects and there is a moderation system in place that has a sobering effect on the discussions. So even if there are cases of 'flamebaits' and 'trolls' (names given to entries whose goal is to provoke and / or mislead), the discussions can safely be assumed to present us with the participants'

[1] This accelerated proliferation of both phenomena (IT and PM) has urged a number of authors to express concern about (1) IT (Angell, 2000; Webster, 2001), and (2) the trend toward the so called projectification of society (see Cicmil & Hodgson 2006:5-6 for review of this critique), worried that PM is taking over our lives with effects that are not always benign.

earnest opinions, and thus with a rich picture of the diversity of programmers' attitudes and sensibilities.

The language of *Slashdot* is English, and the great majority of its participants are native English speakers – even if not always very careful with their mother tongue. A good deal of those participating in the discussions we have studied live in the US – our own crude, experience-based estimate is that about 80% are either US citizens or US workers. So even if there are opinions from programmers from other countries (particularly in those discussions that deal with, for instance, Australian issues), most of the entries originate in the US. It can therefore be said that the material is representative of US programmers, taking into account the heterogeneity of opinions expressed on the themes we explore. At any rate, the question of nationality only makes itself present in very specific discussions, such as those dealing with outsourcing or policy subjects (Piñeiro and Case, 2008).

Analytical tools

In analysing the narrative exchanges of the *Slashdot* community we have found it useful to make recourse to the theoretical notion of 'performativity'. There are two broad patterns emergent in our data that lend themselves to interpretation through two corresponding dimensions of this concept. Firstly there are performative acts (Austin, 1976) on the part of programmers in their talk about the aesthetics of coding that enable them to enact identity and membership. Secondly, there is a *performative commercial context* (Lyotard, 1984) within which they work, often mediated by PM methodologies. The PM context is almost invariably perceived to be a constraining force on the proper pursuit of the programming art. As they provide an analytical fulcrum for the interpretation of our data, we shall elaborate briefly on both these aspects of performativity.

1. Discursive performances

The concept of the performative was first introduced into the philosophical lexicon by John Austin and subsequently taken up by one of his students, John Searle as the basis for a broader programme of enquiry know as speech act theory (Austin, 1976; Searle, 1977). Post-structuralist thinkers (Derrida, 1977; Butler, 1990, 1993; Deleuze and Guattari, 1988) have also experimented with and developed Austin's original ideas. Speech acts and performatives have value, we contend, in understanding and representing the socially constructed world of the *Slashdot* programmers that we studied. Programmers interacting online in the Slashdot forum are *doing things* in their narrative exchanges. They are bringing about social effects through their displays of technical bravado, expressions of aesthetic preference and espousals of dissent, resistance and subversion.

In our data, the conflict between IT programmers and Project Managers does not always take the form of overt disputes between the parties. Often the respective parties – programmers and project managers – are not engaged in direct debate with each other

but are, instead, contributing to a general thread of some sort. It is in the emergent composite exchange that they make disparaging and critical remarks about the 'other'.

2. Performativity as the optimisation of input to output

We have so far dealt with only one side of the 'performativity' equation that we wish to employ, namely, that pertaining to the social effects of narrative performatives. For the other part of the equation we draw on Lyotard's critique of the role of knowledge in post-industrial societies (Lyotard, 1984). According to Lyotard, the episteme of modern science which found legitimacy in grand narratives of progress and emancipation – totalising stories that gave meaning to local narratives and practices – is being systematically eroded within post-industrial societies by the advancement of information-driven technologies. The search for Truth is replaced by a search for the Efficient under what Lyotard (1984: 111) terms the 'principle of optimal performance'. Lyotard theorises this new basis of knowledge – the optimisation of input to output – as *performativity* (1984: 112). One consequence of post-industrial technology's privileging of the ends of action over its means is that knowledge ceases to be a valid end in itself. Knowledge is assessed economically not by its truth-value, but by its *exchange-value*. Knowledge is produced to be sold.

The conflict between programmers and managers presented below hinges on a double application of the concept of Lyotardian performativity. Firstly, it is used to highlight antagonism. Programmers are clearly conscious of the forces of performativity intruding into and shaping their working lives. They articulate an explicit concern that their efforts, which they conceive of as artistry or craftsmanship, are all too often compromised by the tight resource constraints and disciplinary requirements of the PM environment. Managers take the opposite side, accusing programmers of 'whining' instead of doing what must be done to move the project forward. Secondly, both sides struggle to impose their own characteristic knowledge as the truly performative one. This double use of 'performativity' constitutes the themes along which programmers and managers structure their identities, in opposition to each other: aesthetic versus performance concerns and technical versus managerial knowledge.

Empirical observations

Conflict from the programmers' perspective

The performance of 'programmer' identity is context-dependent and different faces are presented to different audiences. In cetain instances programmers will express an interest in meeting deadlines and getting products ready for market but, as we shall see, when responding to the perceived performativity of PM discipline those same programmers will present a very different aspect of identity; one that is far more concerned with program quality and elegance. Our data indicate that the figure of the manager, in general, and of the project manager (often the direct superior), in particular, act as crucial foils in the construction of 'techie' identity.

The extracts below have been selected to demonstrate the double application of 'performativity' by programmers to construct identities *in opposition* to project

managers. As mentioned above, this oppositional relation in our data set manifests around two main themes: 'Code Aesthetics' and 'Technical Knowledge'.

Code aesthetics

This line of identity construction is predicated on the notion that code is worthy of attention for its own sake. Coding is a realm of human creation which celebrates the ethic of *gras artis gras*. It allows for personal styles and aesthetic convictions that offer programmers great satisfaction and that, in the eyes of the programmers, deserve respect. Managers are often accused of both failing to understand the aesthetic dimension of the coding art as well as not treating it with due respect.[2]

Often Slashdotters explicitly define themselves as 'computer geeks', 'hackers' and 'code jockeys', terms whose radicalism or 'cool-ness' contrast with submissive tenor of words like 'worker', 'programmer' or 'employee' which might also be used to describe their roles. This process of identification, self-presentation and affiliation is common in many of the exchanges either as a performative subtext or as part of the manifest communication. Consider, for instance, the following posting by MikeFM:[3]

> **software is like building w/ toothpicks** (Score:2)
> by MikeFM (12491) on Wednesday September 05, @12:59AM (#2254465)
>
> […] I like to write beautiful code.. as I imagine most real programmers do.. us geeks that live, breath, and dream in code.. but in real life there usually is not enough time or resources given to manage to write really well planned out code. This is why Microsoft sucks and a popular motto is 'When it's done!' among the truely geeky programming houses and why open source will eventually kill most commercial software.

MikeFM is here engaged in a micro-sociological process of expressing identification – 'us geeks' – and rehearsing key elements of a hacker credo. These elements are discussed at length in Pekka Himanen's account of the ethos and life-worlds of computer hackers entitled, *The Hacker Ethic and the Spirit of the Information Age* (Himanen, 2001). In this work Himanen identifies a set of characteristics and ethics that, he contends, mark hackers apart as a subculture, as follows: (a) hackers are passionate about their endeavours and are motivated by what they take to be the intrinsic value and craft in their work; (b) hackers are ambivalent toward – if not on occasion outright suspicious of – money and the commercial exploitation of computer systems and applications; and, (c) hackers are committed to 'facilitating access to

---

2 Programming aesthetics are more complex than we have space to address here. For a more detailed treatment see Piñeiro (2003) and Case & Piñeiro (2006). To make the main point briefly, a program that offers *exactly* the same function to the user can be written in a myriad of ways, allowing programmers great degrees of stylistic and structural freedom. Matters of style and structure are sometimes debated by *Slashdot* programmers and strong preferences expressed regarding: (1) the use or otherwise of comments in coding; (2) the use of indentation; (3) whether minimalism or, indeed deliberate obfuscation, is preferable to more extensive, discursive or literate forms of coding. There is also the direct issue of aesthetic ideals, with 'clean code', 'simple code', 'tight code' and 'structured code' being contested topics.

3 This and all subsequent abstracts are presented as they appear on the *Slashdot* bulletin board. For the sake of authenticity, we reproduce spelling, grammatical and syntactical errors wherever they appear in the original.

information and to computer resources' (2001: 51) such that access to resources should, ideally, be free of charge.

Mike FM's remark '[…] us geeks that live, breath and dream in code' may sound to us non-geeks like an over-reaction but it can be better appreciated considering Himanen's first – and perhaps most important – element of the hacker ethic, which contrasts the hacker's unique orientation to work with the concept of the Protestant work ethic propounded by Weber (1976). Unlike many workers who find their work dull and alienating, hackers are passionate about their endeavours and are motivated by what they take to be the intrinsic value and craft in their work. They are often willing to work long hours (in their spare time or for employers) on projects that they deem worthy of their efforts[4].

In addition, MikeFM states a preference for, and alludes to, the superiority of open source code, that is, systems and applications that are worked on free of charge by a self-selecting community of programmers. We also see evinced in MikeFM's posting a common hacker refrain, namely, 'Microsoft bashing'. It is almost obligatory to 'hate' Microsoft and be committed to open source, if one is to be a proper hacker. Accordingly, Microsoft is cast as a central villain and scapegoated by the Slashdot programming community on a routine basis in their exchanges. Rightly or wrongly, this company is seen as representing the antipathy of the hacker ethic and, as such, it plays an important iconic role in the identity work of those who wish to affiliate with other 'serious programmers'.

According to many Slashdotters, 'quality' is always a secondary consideration for project managers. What matters is simply shipping the goods on time and according to budget. In line with the hacker ethic, such constraints are perceived as a huge and compromising imposition on the aesthetic ideals of programmers who wish to produce high quality code. Accordingly, the actions and alleged values of villainous project managers are perceived as a direct threat to the individual and collective identity of members of the programming community. The following narrative by an anonymous Slashdotter illustrates well the manner in which programmers feel constrained and how they give vent to their frustrations by finding fault with the organizational conditions under which they work:

---

[4] Himmanen's assertion that we are witnessing the emergence of a new 'work ethic' within hacker-dom is both a touch ostentatious and romantic. The willingness of hackers to work above and beyond contract is indicative of a widespread shift in contemporary economies from what Etzioni (1961) refers to as 'utilitarian' forms of employment contract toward 'normative' involvment. Cederström and Grassman (2008) offer a more cynical and critical interpretation of the rise of 'neo-normative' organizational control suggesting that certain IT employers (e.g., Google) seduce 'techie' employees by providing a playful working environment within which their actual contractual employment conditions become invisible. This is a variation on the 'false consciousness' argument which would maintain that IT workers are blind to the way in which their labour is actually being exploited by employers. Whether persuasive or not, it remains the case that programmers have contributed significantly to the development of software, such as, the architecture of the Internet, which has accelerated the development of flexible information capitalism; a development which is at the very least ambivalent from the point of view of the Hacker Ethic. True, the Internet faciliates the exchange of software products (cracked or otherwise) free of charge but it also enables capitalist organizations to make profit on the sale of commercially produced and, in some cases, 'open source' software.

> **software manager managing bridge architects…** (Score:1, Insightful)
> by Anonymous Coward on Tuesday September 04, @05:20PM (#2253011)
>
> manager -> we need to ship this bridge in 3 months.
> engineer -> yes, but it's really big and really important
> manager -> yes, but it has to ship in 3 months.
> engineer -> so how much weight does it need to support?
> manager -> i dunno, I'll let you known in 2.9 months.
> engineer -> what is it bridging?
> manager -> why all these stupid questions, start building.
> engineer -> I should do an architectural drawing first.
> manager -> why bother, here's some metal, start slapping it together. Remember it ships in 2 months.
> engineer -> I thought you said 3 months?
> manager -> oh didn't I tell you, we heard a rumour that a competitor will be shipping their bridge in 2.5 months, so we have to beat them.
> continue forever.
> the reason there is no internal beauty is we (engineers) aren't given any time to build quality (although the argument could be made that the only way to build on schedule is by building quality). The other problem is, bugs actually translate into lucrative support contracts for most enterprise software vendors. Why improve quality? there is no revenue stream there. […]

In the following narrative, Greyfox places emphasis on the antagonistic relationship between the programmer and the project manager to whom s/he reports (beauty vs. deadline):

> **Oh Yeah…** (Score:2)
> by Greyfox (87712) on Tuesday September 04, @05:11PM (#2252945)
>
> Try telling your manager sometime that you want to redesign a piece of code because 'It's aesthetically displeasing' or because 'The design sucks.' He'll laugh you out of the office and quite possibly the company. Nevermind that you were right or that your redesign would drastically improve maintability and probably speed things up. Managers don't want good code. They want code that you can squat and shit as quickly as possible because the only metric they look at is the deadline. It may not smell good. It may self destruct in a few months. It will certainly keep your team in 'fireman mode' for the rest of the time they're at the company, but by God it made the deadline and that's all that counts. […]

There are important self-presentational aspects (Goffman, 1959, 1967) to the *Slashdot* exchanges as epitomized by Greyfox's put down of computer illiterate or, at least, computer-incompetent project managers who believe that programming language architecture will in itself produce effective software design. Contributors to the *Slashdot* forums routinely use their narratives to establish their own programming credentials and prowess, MikeFM's 'us geeks that live, breath and dream in code' being an extreme case. Which brings us to the second main theme of programmer identity construction: technical knowledge.

Technical knowledge

This line of identity construction is based on the predicate that (project) managers do not know what an IT system actually is. Such alleged ignorance extends not only to the programming side of things but also to more 'managerial' aspects such as the roots of bugs and why low-quality software requires far more long term maintenance than well constructed programmes (and hence incurs further costs for the company). Moreover,

according to this attributional predicate, managers are oblivious to the principles that result in high-quality software, and, more generally, to what decisions are in the best interests of the company.

Quite independently of Anonymous Coward, MikeFM uses a similarly powerful construction metaphor to illustrate the manner in which commercial pressures prompt project managers to adopt short-term palliative solutions to software design problems:

> **software is like building w/ toothpicks** (Score:2) by MikeFM (12491) on Wednesday September 05, @12:59AM (#2254465)
>
> I think in the book 'The Hacker and the Ants' there is a quote along the line of programming being like building out of toothpicks carefully glued together and if just one toothpick is out of place the whole thing comes crumbling down […].

The issue of the (project) managers' ignorance can become very loaded when it comes to the subject of outsourcing. It is demeaning for programmers to find themselves being laid-off by managers who, in their view, do not have the competence to make such decisions. The following quote is extracted from a discussion about IT off-shoring:

> **Maybe it's time for the technocratic war to begin**. (Score:5, Interesting) by Anonymous Coward on Friday December 19, @12:14PM (#7764834)
>
> The managers and CEOs of this country have no idea about how to make router connection [*sic*] or how to correct a line of code in their payroll systems.
>
> I'm on call 24x7x365 while the CEO sleeps.
>
> The none [*sic*] technical types need to understand where info power resides.

Here are some further entries that illustrate the way in which claims about technical knowledge are used to bolster programmer identity and deride the ignorance of project managers:

> **I couldn't disagree more…** (Score:4, Insightful) by gustar (125316) on Friday June 21, @08:46AM (#3742810)
>
> […] Over the course of my career I have dealt with legions of formal 'project managers', (folks who are pure project managers lacking any technical background) and I have yet to realize any value in my interactions with any of them, beyond the occasional willingness to record meeting minutes.
> To date I have found them to be glorified secretaries, whose primary tactic is to latch on to knowledgeable people and not only drain information but actually get them to perform the real tasks of project management, such as scheduling and resource estimation […]

The digitally self-styled 192939495969798999 describes how PM can wreck the fun aspects of programming by making decisions on the basis of resource contraints rather than 'scientific research':

> **Welcome to programming** (Score:3, Informative) by 192939495969798999 (58312) <info@devinmoor[ ]om ['e.c' in gap]> on Friday June 21, @06:34AM (#3742415)
>
> I think it's important for people to read this and realize that the PM is the reason that programming sucks 9 times out of 10 in the professional world. As much fun as it would ever be, the PM can

> wreck it by having no concept of reality. I don't much care if the PM doesn't understand exactly HOW to implement things, but it's been my experience (most recently as a lead programmer at a small outfit) that the PM will make decisions based on the immediate costs, rather than any solid scientific research […]

AppyPappy introduces a very colourful metaphor to characterize the condition of the technically neutered project manager. Again, attributions of lack of technical knowledge are used to malign PM:

> **Nope** (Score:4, Insightful) by AppyPappy (64817) on Friday June 21 2002, @07:30AM (#3742566)
>
> Programming skills and management skills are mutually-exclusive. I've always found project managers to be hired as programmers who were later found to be lousy programmers. […]

So far we have sought to analyse the way in which programmers construct their identity in opposition to the perceived constraints of environments governed by PM disciplines and the incompetence of project managers. What, then, of project manager identities? How do these compare and contrast with those of programmers in the *Slashdot* forum?

Conflict from the project panagers' perspective

Slashdot is not generally a good place to look for managers' opinions. Programmers are much more common participants in bulletin board exchanges and the general tone of exchange is, as Himmanen's hacker ethic suggests, highly critical of management values. However, the discussion entitled 'Project Management for Programmers?' opened up the door to views other than those of programmers. Unlike other *Slashdot* exchanges, this one lies at the cusp of technical programming and managerial identity. For some Slashdotters, there is an actual or – in normative terms – necessary overlap between project management and technical development roles. In other words, according to many *Slashdot* techies, project managers should be technically competent programmers in order to do their job properly. For other discussants, it is necessary to maintain a clear distinction between the skills and responsibilities of technical development, on the one hand, and those of project manager, on the other.

Rather unusually, because of the potential overlap of technical and managerial roles, both the 'technical' and 'management' perspectives are represented in the exchange. This stands in marked contrast to many discussion threads where technical concerns and perspectives enjoy an exclusive monopoly and management is generally a dirty word. The 'Project Management for Programmers?' thread begins with a question from a programmer – welshdave – who claims to have encountered resistance to his being promoted to a project management role by senior managers who feel threatened by his technical expertise. As s/he frames the question: 'Has anyone else found the barrier to project management is their technical knowledge. How did you get past it?'

In parallell to the discussion of programmer identity above, the extracts below have been selected to demonstrate the double application of 'performativity' by project managers to construct identities *in opposition* to those of programmers. This oppositional relation in our data set manifests around two main themes: 'Performance concerns' and 'PM Knowledge'.

Performance

This line of identity construction for project managers is based on the predicate that programmers are lazy, and that they hide behind technical buzzwords to avoid work[5]. The talk about the elegance of code and such is simply a way to avoid doing the practical work of getting properly functioning software to the customer on time and within budget.

In the following extract, smoon expresses exasperation at the constant 'whining' of programmers regarding resource constraints. Developing a seemingly apolitical position of rapprochement, s/he advocates a middle ground on which project manager and programmer can meet cordially. Here we see an assertion of the unitarist conception of management according to which conflict is both dysfunctional and unnecessary:

> **Stop whining, start doing.** (Score:5, Insightful) by smoon (16873) on Friday June 21, @06:36AM (#3742421)
>
> […] As a [project] manager I get very tired of hearing about the programmers, sysadmins, etc. complaining that such-and-such can't be done, or otherwise blocking progress. Much more often than not things that 'can't be done' just require a re-statement of the problem and some creative application of simple ideas […]

In the following entry, PinglePongle highlights directly the antagonism felt by project managers from programmers and stresses, in emotive and personal terms, the PM priority for getting things done:

> **Project Managers don't need to be techies…** (Score:5, Insightful) by PinglePongle (8734) on Friday June 21, @06:58AM (#3742478)
>
> […] The attitude of most of the posts in this subject has been 'huh, we're 200 times smarter than those idiots running the project, they're so stupid they couldn't blah blah blah'. Hey, if you're so smart, it's your job to use that intellgence to move the project forward, not whine about how what a bad job everyone else is doing.

PM knowledge

This line of identity construction involves project managers making claims about PM knowledge that run counter to the attributions of programmers. Just as programmers claim that project managers are ignorant of the technical knowledge necessary to produce high quality code, so project managers claim that programmers lack the technical knowledge required to manage projects effectively. While project managers acknowledge that programmers have narrow technical knowledge they are alleged to be ignorant about a fundamental business imperative: the process of getting functioning software to the customer in timely and financially viable way.

---

[5] There are distinct overtones here of Douglas McGregor's 'Theory X and Theory Y' account of managerial personality (McGregor, 1960). According to Theory X, workers are lazy, do not share the organization's goals, cannot be trusted and require close supervision. By contrast, Theory Y managers assume that employees are mature, self-disciplining, self-motivated and needing little direct organizational control. Project managers in the exchange we analyse seem to express Theory X assumptions.

In a seemingly frustrated mood, Thunderheart begins his/her posting with a bold assertion: 'engineers are NOT project managers'. To conflate the respective skill sets, s/he maintains, is to court disaster and should be avoided. This entry is typical of several postings that celebrate the technical demands and professional competency required of project managers. It seems to be an expression of the PM's annoyance with being portrayed by the *Slashdot* community as technically incompetent. Here is an emotionally charged plea on the part of a project manager to have his or her skills acknowledged by techies as being every bit as technically demanding and professionally challenging as those of the programmer:

> **Engineers are NOT Project Managers** (Score:3, Insightful) by Thunderheart (574412) on Friday June 21, @08:12AM (#3742673)
>
> Whether its IT, Municipal drafting Electrical or whatever, Engineers (regardless of how long they have 'managed' projects) are NOT Project Managers. You frustrate the hell out of me. I've been a Professional Project Manager for years and an Amateur computer geek. The thing that always stuck in my craw is the assumption that just because a person knows an Engineering Discipline that they automatically know how to manage projects. Project Management is a complex discipline and to manage projects well takes a solid educational background in that arena. It is a skill set unto itself. Document Controls, managing Gaant charts and schedules and (especially) managing the 'people' end of things takes a great deal of effort to excel at. But NOOOOOO, Engineers always assume that because they can conceive a project, they MUST be able to manage it, and it always ends up as a grand jitterbug called, 'Crisis Management' […]

In a more measured and rational tone, rsmah makes a very similar point about the difference between PM and programming skill sets. PM has, as it were, an extraverted orientation toward the wider organization and business requirements while programmers, by necessity, have a more introverted focus on architecture and coding:

> **Don't mix management and architecture** (Score:5, Insightful) by rsmah (518909) <.~moc.xobop. .ta. .hamr.> on Friday June 21, @06:38AM (#3742429)
>
> The problem your firm seems to be facing is that you are mixing project management with system design/architecture. What's the difference you ask? Project management is the process of resource allocation, scheduling, budgeting and task tracking. System design/architecture is the process of figuring out what should be built and how it should be structured internally.
>
> Good project managers need a different set of skills than system architects. Project managers think in terms of timelines, tasks and dollars. Architects think in terms of system components, their interactions, user requirements and technology. While there are some people who can do both well, they are quite rare as they require fairly different ways of thinking.

Like Thunderheart and rsmah in the previous extracts, bons argues that the solution to the original question posed by welshdave is to acknowledge the difference between the programmer and the project manager. The suggestion that a programmer should attend courses by the Project Management Institute (the dominant professional project management body in the USA) reaffirms the idea that there are fundamental things that programmers do not know.

> **Get certified and go to the local PMI meetings** (Score:4, Informative) by bons (119581) on Friday June 21, @06:42AM (#3742440)
>
> PMI [pmi.org] has all you need to know about certification and there are PMI meetings just about everywhere [google.com]. Attend a few of those and you'll either be networked enough to improve things or fins [*sic*] a better job.

Fnkmaster also emphasises the lack of business awareness of programmers and the need, therefore for PM and Technical Project Manager [TPM] roles:

> **Re:Oh Please!** (Score:5, Informative) by Fnkmaster (89084) on Friday June 21, @08:59AM (#3742869)
>
> This is pure trash. The fact is that most programmers don't and don't really care to understand much about the business. That's exactly the reason that you need technical leads or TPMs who understand both the business requirements and enough of technology to make reasonable trade-off decisions, and either work closely with a business-oriented PM/requirements person, or have excellent rapport with upper-management (i.e. have their trust – not be perceived as a lying technology person).

The following extract gives explicit form to the body of knowledge that constitutes the IT project management role. Once again, in this identity claim there seems to be some exasperation that programmers do not appreciate what PM entails:

> **It's not as easy as it looks** (Score:5, Informative) by James Youngman (3732) on Friday June 21, @06:59AM (#3742480) Homepage
>
> There's much more to successful project management than there appears, particularly when the PM is also managing the relationship with an external client. It's the PM's job to make the client happy and (usually) deliver a profit. In a software context, this normally means delivering some software that works […]
>
> As a technical person, skills that you will need to gain in order to be a successful PM will include
>
> * Understanding the business context and business drivers
> * Managing client relationships (even for internal clients)
> * Estimating and planning skills
> * Tracking progress against plan – and taking appropriate action (pay attention to this one!)
> * An understanding of what timescales are realistic. For example, is it realistic to estimate design:code:test in the ratio 3:2:1? (answer: no).
> * Understand that you need to make it possible for the client to change their mind half-way through
> * Delegation skills (you can't do it all yourself, you know!) and motivational skills (i.e. understand the kinds of things you can / can't ask of people).
> * Risk analysis/mitigation
> * Personal organisation and time management
> * (In some shops) Project accounting skills
>
> Also, don't underestimate how much work this is. […]

Finally, terrapyn promotes a unitarist vision of organizational performativity which, as with previous extracts, emphasises the *difference* between PM and technical coding roles:

> **Re:Consider it an advantage…** (Score:5, Insightful) by terrapyn (259226) on Friday June 21, @07:23AM (#3742546)
>
> I think the biggest contributor to project failures I have seen is the attitude you are expressing. The project manager has a distinct role that is NOT 'database designer', 'programmer' or other technical function, but which instead is focused on the coordination and communication among the team members and with outside parties (clients, management, users, finance, etc.) in order to JOINTLY build and execute a plan.

In these project manager narratives that stress the need for technical PM knowledge necessary to expedite the role (which, differs from that of the programmer), we find certain resonances and parallels with the aesthetic concerns that programmers express about the production of code. Project managers speak of having 'skills', of how you can 'excel' at managing projects or how you must build 'excellent' relations. The idea that project managers must be able to make 'reasonable trade-off decisions' is suggestive of how PM might also be conceived of as a craft or art which somehow *exceeds* its bureaucratic functionality. In our, admittedly small, data set we even found the management equivalent of 'ugly' code: 'jitterbug Crisis Management'. Perhaps the performative ideals of PM and the 'contemplative' and artistic ideals of hackers share much more than what might appear at first sight. In one sense, therefore, we might understand the conflict between programmers and project managers as representing, in part at least, a conflict of incompatible aesthetics. This may not be so surprising if we acknowledge that the ethics and politics of employment relations cannot be easily parcelled out from aesthetic concerns. To reflect on identity conflict in terms of its aesthetic dimension in a work domain that might usually be thought to be mundane and anti-aesthetic could, as our analysis suggests, shed fresh light on IT workplace relationships.

Conclusions

The relationship between project managers and software programmers that emerges from our analysis of the *Slashdot* exchanges has several dimensions. Firstly, there is an antagonistic quality to conflicting identity claims with opposition to project management being one of the fundamental aspects of programmer identity construction. Programmers represent themselves as (1) more knowledgeable (both technically and in general) than project managers; and (2) initiated insofar as they understand the intrinsic value of code and what makes for high quality software production. Project managers represent themselves as (1) more knowledgeable than programmers insofar as they understand fundamental aspects of software production, particularly with respect to business constraints, which programmers routinely ignore; and (2) action-focused, as opposed to programmers who simply 'whine' about how complex and difficult things are.

Secondly, there is a collaborative dimension to the relationship between programmers and project managers. These two groups are compelled into practical everyday collaboration by exogenous market forces and business imperatives. In order to keep their jobs they have to work together to produce functioning software.

Thirdly, there is an apparent parity between the two groups, resulting from their technical proximity, despite their unequal position in the organizational hierarchy. As common semantics imply, project *managers* are, by bureaucratic designation, positioned higher up the corporate ladder than programmers, but as managers of relatively low rank, they are very close to and heavily dependent on programmers. The dependence is reinforced by the fact that software programmers hold advanced technical knowledge that project managers often lack. This parity between the groups is illustrated by the way in which 'technical knowledge' is mobilized by both in order to represent their identities vis-à-vis the other. Furthermore, it is interesting to note how the aesthetic sensitivities programmers express for code have an implicit equivalent within project manager narratives.

Analysing the relationship between programmers and project managers in high-tech environments enables us to shed new light on workplace conflict. Here we find various members of the organisation possessing unique skills and knowledge, making specialized and mutually interdependent contributions to production and hence being similarly indispensable. We have attempted to show how this indispensability – even more pronounced in our case due to the close hierarchical proximity of project managers and programmers – does not minimise the antagonism between groups. The strong antagonism we have presented is, however, limited to the verbal realm, and is, in our data, mostly exhibited not in direct disputes *between* the groups but through discussions *within* the groups. For the most part, it appears that the respective parties talk past each other.

There is clear evidence from our data that participants in the *Slashdot* discussions – particularly the programmers – demonstrate a conscious awareness of what might reasonably be construed as a divergence of interest between management and workers. In other words, programmers make recourse to a vocabulary of motives that *instantiates* the orthodox Labour Process view of workplace conflict and resistance. Perceived *differences* between 'worker', 'manager' and resulting 'conflict of interest' are discursive resources that both programmers and project managers call upon in the co-construction of their workplace relationships. However, as we have argued elsewhere (Piñeiro 2003), programmers often openly espouse and adopt the same ultimate goals as managers: they claim that their concern about the elegance of code does yield better functioning, less 'buggy' and, eventually, more profitable software products. The divergence of interest that both parties keenly publicize lies not so much in an opposed view of the common goal at hand as in alternative ways to achieve it. For programmers, 'organizational goals' can be met through effective and elegant software production, while, for project managers, the solution resides in 'better management' (in the form of closer control, team working and the discipline of project management itself). What is important from our point of view, then, is that these displays of divergent interest and managerial rhetoric that seek to de-politicize organization through espoused visions of collective action are situated and expressed in localized micro-narratives. In other words, conflict is, as it were, an etic signifier that we, as analysts, use to condense and summarize features of the local co-creation and discursive production of organization. From this intepretative position, identity performances of the sort we have been concerned with in this article play a *constitutive* role in the production, expression and maintenance of workplace conflicts within IT environments.

Apart from the consequences it might have for daily collaboration, which our narrative data do not permit us to comment on , the antagonism we have seen plays a central role in the simultaneous creation and expression of identities. The conflict between two different discourses materialises in the continuous performative work through which identities are created, adopted and expressed in relation to a designated 'other' – be it project manager or programmer. Programers and project managers in the software industry are, as discussed in the introduction, connected through a complex series of dependencies, making it practically impossible to claim that conflicts in this case are based on an underlying asymetrical relation.

Conflict there is, however, and what we show in the excerpts above is its discursive (re)production and *enactment* through forms of narrative. Whether or not conflict derives from the kind of claims made by critical realists and Labour Process theorists about the underlying ontological *structures* of social relations within post-industrial societies remains a metaphysical question for this particular article. The principal contribution we wish to make resides in trying to reveal the *pragmatic* dynamics of identity conflict as peculiarly manifest in hi-tech, project-managed, settings. Where there is close technical proximity between managers and those they manage, the signification of 'programmer' and 'manager' carries forceful performative effects. It is the narrative source and expression of such organizational effects that we have sought to explore and theorise in this article.

references

Ackroyd, S. and S. Fleetwood (eds) (2000) *Realist Perspectives of Business and Organization*. London: Routledge.

Ackroyd, S. and S. Fleetwood (eds) (2004) *Critical Realist Applications in Organisation and Management Studies*, London: Routledge.

Alvesson, M. (2004) *Knowledge Work and Knowledge Intensive Firms*. Oxford: Oxford University Press.

Alvesson, M. and D. Kärreman (2007) 'Unraveling HRM: identity, ceremony, and control in a management consulting firm', *Organization Science*, 18(4): 711-723.

Alvesson, M. and H. Willmott (2004) 'Identity regulation as organizational control: producing the appropriate individual', in M.J. Hatch and M. Schultz (eds) *Organizational Identity: A Reader*. Oxford: Oxford University Press.

Angell, I. (2000) *The New Barbarian Manifesto: How to Survive in the Information Age*. London: Kogan Page.

Austin, J.L. (1976) *How to Do Things With Words*. Oxford: Oxford University Press.

Bruckman, A. (1992) 'Identity workshop: emergent social and psychological phenomena in text-based virtual reality', [ftp://ftp.cc.gatech.edu/pub/people/asb/papers/identity-workshop.rtf], visited 27 April 2008.

Burawoy, M. (1979) *Manufacturing Consent: Changes in the Labour Process Under Monopoly Capitalism*. Chicago: University of Chicago Press.

Burrell, G. (1988) 'Modernism, post modernism and organizational analysis 2: the contribution of Michel Foucault', *Organization Studies*, 9(2): 221-235.

Burrell, G. (1998) *Pandemonium: Towards a Retro-Organization Theory*, London: Sage.

Burrell, G. and Morgan, G. (1979) *Sociological Paradigms and Organizational Analysis*, London: Heinemann.

Butler, J. (1990) 'Performative acts and gender constitution: an essay in phenomenology and feminist theory', in S-E. Case (ed.) *Performing Feminisms: Feminist Critical Theory and Theatre*, Ed., Baltimore: Johns Hopkins University Press.

Butler, J. (1993) *Bodies that Matter: On the Discursive Limits of 'Sex'*, London: Routledge.

Case, P. and E. Piñeiro (2006) 'Aesthetics, performativity and resistance in the narratives of a computer programming community', *Human Relations*, 59(6): 753-782.

Cederström, C. and R. Grassman (2008) 'The masochistic reflexive turn', *ephemera: theory & politics in organization*, 8(1): 41-57.

Chia, R. (1996) *Organizational Analysis as Deconstructive Practice*. Berlin: De Gruyter.

Chia, R. (ed.) (1998) *In the Realm of Organization: Essays for Robert Cooper*. London: Routledge.

Cicmil, S. and D.E. Hodgson (2006) 'Making projects critical: an introduction', in D.E. Hodgson and S. Cicmil (eds) *Making Projects Critical*, London: Routledge.

Cooper, R. (1989) 'Modernism, post modernism and organizational analysis 3: the contribution of Jacques Derrida', *Organization Studies*, 10(4): 479-502.

Daft, R.L. (1995) *Organization Theory and Design*, 5th edition. St. Paul MN: West Publishing.

Deleuze, G. and F. Guattari (1988) *A Thousand Plateaus: Capitalism and Schizophrenia*. London: Athlone Press.

Derrida, J. (1977) *Limited Inc.* London: Johns Hopkins University Press.

Donaldson, L. (1996) *For Positivist Organization Theory: Proving the Hard Core*. London: Sage.

Edwards, P. (1986) *Conflict at Work: A Materialist Analysis*. Oxford: Blackwell.

Etzioni, A. (1961) *A Comparative Analysis of Complex Organizations*. London: Free Press.

Florence, M. (1994) 'Foucault, Michel, 1926-', in G. Gutting (ed.) *The Cambridge Companion to Foucault*. Cambridge: Cambridge University Press.

Foucault, M. (1980) *Power/Knowledge: Selected Interviews and Other Writings*. Brighton: Harvester.

Friedman, A. (2004) 'Strawmanning and labour process analysis', *Sociology*, 38(3): 573-591.

Goffman, E. (1959) *The Presentation of Self in Everyday Life*. New York: Doubleday.

Goffman, E. (1967) *Interaction Ritual: Essays on Face-to-Face Behavior*. New York: Doubleday.

Grint, K. (1991) *The Sociology of Work: An Introduction*. Cambridge: Polity.

Hassard, J. and M. Parker (eds) (1993) *Postmodernism and Organizations*. London: Sage.

Himanen, P. (2001) *The New Hacker Ethic and the Spirit of the Information Age*. London: Vintage.

Hodgson, D.E. (2002) 'Disciplining the professional: the case of project management', *Journal of Management Studies*, 39(7): 803-21.

Hodgson, D.E. (2004) 'Project teams: the legacy of bureaucratic control in the post-bureaucractic organisation', *Organization*, 11(1): 81-100.

Hodgson, D.E. and S. Cicmil (2006) 'Are projects real? the PMBOK and the legitimation of project management knowledge', in D.E. Hodgson and S. Cicmil (eds) *Making Projects Critical*, London: Routledge.

Jacques, R. (1996) *Manufacturing the Employee: Management Knowledge from the 19th to 21st Centuries*. London: Sage.

Kärreman, D. and M. Alvesson (2001) 'Making newspapers: conversational identity at work', *Organization Studies*, 22(1): 59-89.

Knights, D. (2001) 'Hanging out the dirty washing', *International Studies of Management and Organization*, 30(4): 68-84.

Knights, D. and Willmott, H. (1989) 'Power and subjectivity at work: from degradation to subjugation in social relations', *Sociology*, 23(4): 975-995.

Lindgren, M. and J. Packendorff (2006) 'Projects and prisons', in D.E. Hodgson and S. Cicmil (eds) *Making Projects Critical*, London: Routledge.

Lyotard, J.-F. (1984) *The Postmodern Condition*. Manchester: University of Manchester Press.

McGregor, D. (1960) *The Human Side of Enterprise*. New York: McGraw-Hill.

O'Doherty, D. (forthcoming) 'Retreiving the 'missing subject' in labour process analysis: towards emancipation and praxis', in Alvesson, M., Bridgman and H. Willmott (eds) *Handbook of Critical Management Studies*. Oxford: Oxford.

O'Doherty, D. and H. Willmott (2001a) 'The question of subjectivity and the labour process', *International Studies of Management and Organization*, 30(4): 112-132.

O'Doherty, D. and H. Willmott (2001b) 'Debating labour process theory: the issue of subjectivity and the relevance of poststructuralism', *Sociology*, 35(2): 457-76.

Oxford English Dictionary [Online http://www.oed.com/, visited 28 March 2008.]

Piñeiro, E. (2003) *The Aesthetics of Code*. Stockholm, Arvinius Förlag.

Piñeiro, E. and P. Case (2008) 'Outsourcing in high-tech corporations: voices of dissent, resistance and complicity in a computer programming community', in D. Jemielniak and J. Kociatkiewicz (eds) *Management Practices in Hich-Tech Environments*, Hershey PA: IGI Global.

Schein, E.H. (1985) *Organizational Culture and Leadership*. San Francisco: Jossey-Bass.

Searle, J. (1977) *Speech Acts: An Essay in the Philosophy of Language*, London: Cambridge University Press.

Shenhav, Y.A. (1999) *Manufacturing Rationality: The Engineering Foundations of the Managerial Revolution*, Oxford: Oxford University Press.

Taylor, F.W. (1986 [1911]) *The Principles of Scientific Management*, Easton PA: Hive Publishing.

Thompson, P. and S. Ackroyd (1995) 'All quiet on the workplace front? a critique of recent trends in British industrial sociology', *Sociology*, 29(4): 615-33.

Thompson, P. and D. McHugh (1995) *Work Organizations*: *A Critical Introduction*. 2nd edition. London: Macmillan.

Thompson, P. and C. Smith (2000-1) 'Follow the redbrick road: reflections on pathways and out of the labor process debate', *International Studies of Management and Organization*, 30(4): 40-67.

Weber, M. (1976) *The Protestant Ethic and the Spirit of Capitalism*, trans. T. Parsons, 2nd ed., London: Allen and Unwin.

Webster, F. (ed.) (2001) *Culture and Politics in the Information Age*. London: Routledge.

the authors

Peter Case is Professor of Organization Studies, Bristol Business School, University of the West of England, and Director of the Bristol Centre for Leadership and Organizational Ethics. He served as chairperson of the Standing Conference on Organizational Symbolism from 2002-7 and is general co-editor of *Culture & Organization*.
E-mail: Peter.Case@uwe.ac.uk

Erik Piñeiro is a researcher at the School of Technology and Health, Royal Institute of Technology, Stockholm. He recently returned from a postdoc at the Scandinavian Consortium for Organizational Research at Stanford University. He was previously with the Department of Industrial Economics and Management at the Royal Institute of Technology.
E-mail: erikp@kth.se