Journal of Organizational Change Management, Vol. 20 No. 4, 2007 pp. 491-508
(this is the final version of the article, reformatted to satisfy Emerald publishing policies, page numbers are as in the original, which can be found at http://www.emeraldinsight.com/journals.htm?articleid=1615745&show=abstract).

Managers as lazy, stupid careerists?

Contestation and stereotypes among software engineers

Dariusz Jemielniak
Kozminski Business School, Warsaw, Poland

Abstract

Purpose – The purpose of this paper is to present the results of a qualitative study of software engineers' perception of dress code, career, organizations, and of managers.

Design/methodology/approach – The software engineers interviewed work in three European and two US companies. The research is based on ethnographic data, gathered in two longitudinal studies during the period 2001-2006. The methods used in the study include open-ended unstructured interviews, participant observation, collection of stories, and shadowing.

Findings – It was found that the majority of software engineers denounce formal dress-codes. The notion of career was defined by them mostly in terms of occupational development. They perceived their own managers as very incompetent. Their view on corporations was also univocally negative. The findings confirm that software engineers form a very distinctive occupation, defining itself in opposition to the organization. However, their distinctiveness may be perceived not only as a manifestation of independence but also contrarily, as simply fulfilling the organizational role they are assigned by management.

Originality/value – The study contributes to the organizational literature by responding to the call for more research on high-tech workplace practices, and on non-managerial occupational roles.

Keywords Software engineering, Workplace learning, Managers

Paper type Research paper

Thus, spake the master programmer: "Let the programmers be many and the managers few-then all will be productive" (James, 1986).

Introduction

Although the managerial literature is dominated by the perception of culture as a company's integrative factor or even manageable asset (Hofstede, 1980; Ouchi, 1981; Peters and Waterman, 1982; Schein, 1985), many authors show that organizational realities are never so simple. In fact these realities abound in conflicting and chameleon subcultures, quite often challenging the dominant managerial view (Rosen, 1991; Van Maanen, 1991; Martin, 1993). Conflict between managers (basing their power on the company owners' mandate and formal structures) and professionals (in turn basing their power on knowledge) occurs in many, if not most, organizations (Hall, 1986; Abbott, 1988; Trice, 1993). A good illustration of this tension is given by Pondy (1983), who cites the example of accountants who have a proverb that their job is 'protecting the company from the managers'.

The author would like to sincerely thank Davydd Greenwood, Pauline Gleadle, and the anonymous reviewers, whose constructive critique and comments helped in improving this paper.

DOI 10.1108/09534810710760045

Kunda comments that “managers must learn to squeeze the most out of engineers and development groups” (Kunda, 1992, p. 44). Software engineers are a professional group that is particularly subject to constant managerial pressure and a consequent burn-out. They also experience “time famine” having to work to constant deadlines and tight budgets (Kunda, 1992; Perlow, 1997; Perlow, 1998; Cooper, 2000; Jemielniak, 2005). This is perhaps hardly surprising, in view of the finding that high-tech workers are often subject to normative control, to an imposition of values and feelings from the “greedy” organization (Coser, 1974; Kanter, 1977; Kidder, 1981; Kunda, 1992; Hochschild, 1997). Viewed in this light, identity shaping, indoctrination, and “creation of emotions” are all tools used by management (Jackall, 1988; Kärreman and Alvesson, 2004).

Indeed, it is argued that “managers and professionals (particularly engineers) are those who most closely identify with the companies for whom they work” (Kunda and Van Maanen, 1999, p. 64). However, as commented elsewhere, software engineers form a quite unique and distinctive (counter)culture (Kraft, 1977; Bucciarelli, 1988; Trice and Beyer, 1993; Garsten, 1994; Kunda and Van Maanen, 1999; Hertzum, 2002; Pin˜eiro, 2003; Vallas, 2003). They manifest their distance from organizations they work for in many different ways (Kidder, 1981; Perlow, 1997). They form also, as some authors claim, the avant-garde of the “brave new workplace of (the) electronic age” (Gephart, 2002).

The current study is an ethnography of a particular work group with distinctive attitudes at odds with management. The study is interesting for the way it addresses and illuminates the nature and implications of work-place culture and other differences which co-exist within a work setting.

Method and its limitations

A longitudinal ethnographical study was conducted in three Polish and two US IT companies. The research method was based on non-participant direct observation, collected written stories (asking interviewees to write a story beginning with a phrase “Once a software engineer met a manager ...”), shadowing of the selected actors, and open unstructured interviews, lasting typically 40-50 minutes (55 software engineers and five managers in the study). Importantly, all interviewees were salaried workers, not contractors. However, in the companies studied the majority of programmers were employed full-time, the sole exception being the temporary coders. Those were not treated by the fully employed as really belonging to the group and thus were not interviewed.

To assure anonymity, the interviewees’ names were replaced by a company’s fictitious nickname and a number. The results are performative, not ostensive, as in Latour’s (1986) terminology. In this sense they aim to understand and explain the point of view of the interviewed, rather than at offering a definitive interpretation of the analyzed problem, resulting from a preconceived theoretical model. Thus, the choice of questions was very much dependent on how the interview progressed and the outcome is to a large extent under the influence of the interviewees (Whyte and Whyte, 1984).

For structural reasons, carefully selected excerpts are presented from interviews found particularly representative of what most of the informants said. The title of the paper, on the contrary, reflects the author’s own synthesis of the interviews. The paper is organized into four sections. The first addresses the most visible issue of dress code and software engineers’ perception of this. The second covers their view of career. The third is dedicated mainly to their comments about organizations and managers. Finally, the last section sums up and places the findings within the framework of professionalization theory. We conclude that paradoxically, the rebellious role programmers play in many organizations may result from the strong expectations organizations explicitly present them with.

Coders' dress code

The way people dress conveys many meanings, sometimes even resulting in the name given to an organization group. The Brown Shirts formation is one of many historical examples, but organizations commonly have such groups. For example, according to Johnson (1990), "suits" is a term commonly used by blue-collar workers to describe managers. Similar tendencies can be observed in high-tech businesses. According to Kawasaki (1990), a top manager in Apple, calls programmers "T-shirts" and labels people from finance or marketing "ties".

Indeed, software engineers are quite often depicted as dressing very informally (e.g. by this group's ethnographic pioneer: Kidder, 1981). According to an Australian corporate stylist, Melanie Moss, IT workers are perhaps the "worst dressed" professionals in all industries (Hearn, 2005). In the current study of five companies, out of 55 software engineers interviewed, none regularly wore a suit or a tie. In contrast, all managers and salespeople encountered wore both at all times. The difference was so marked that it was decided to explore reasons behind these differences.

Most of the programmers initially expressed a belief that their profession did not have any dress-code. They also quite commonly said that the company they worked for did not impose any rules in this respect. "You can dress however you like" "Anything goes" "Wear something, that's the rule [laughter]" were the recurrent comments. The general tone of the responses is represented by the following excerpt (Wodan2):

> I never really liked suits, I didn't feel comfortable in them, but I don't know ... My friends who are software engineers, too, but work for the big companies, they often have to wear suits. They have something like a policy or an agreement, about what they have to wear, and so on. But here it's different.

"Here, it's different" was what interviewees often repeated, no matter which company they worked at. Even in the big corporations studied here (American, one of the world leaders in speech recognition, and Polish, a Central European leader in business software solutions) the programmers not only shared this belief, but also stated that they did not like "other big companies' policies" that enforced a strict dress code. Viewed in this light a suit was described as a symbol of being boring or even uncultured (minicorp4):

> [Q:] What about the way software engineers dress? Can you tell me more about this?
>
> [A:] Well, about software engineer's dress ...I'm not sure; I'd think what you wear is not that related to any particular job at all. Or it is rather related to some conventions of a firm. I mean, in general I noticed, that in today's world a white shirt and a suit are like a military uniform of a sort, there is some uniformity. I guess it is supposed to suggest working culture, but it suggests the lack of it. No offence, but you wear a white shirt for some special occasion, not every day, this is why it's special. Here, at our company, it is a small one anyway; we don't have any dress code whatsoever. It is a fact, though, that when we go to see a client, we do wear a suit, and our boss asks us to do so. But this is rather obvious, it in a way shows respect for the client (...).

This interviewee defined a suit as a "military uniform"; consciously or not recognizing its historical origins, expressing a very negative opinion of dress codes *per se.*

However, he did believe, and so did most of the interviewed, in the necessity of wearing a suit for meetings with clients.

Although in general interviewees at first did not recognize that a dress code existed, many of them added that their peers in fact do exert subtle pressure on how to dress. These observations were consistent with the negative perception of formal dress codes (Wodan2):

> [Q:] If you wore a suit, how would it be received?
>
> [A:] Well, colleagues would laugh definitely. But it wouldn't be mean, just friendly comments. You know, we don't usually wear suits, so if somebody appears in one it somehow causes reactions ...

This pressure was described by many of the informants. The dialogue typically went as follows (Minicorp5):

> [Q:] Could you tell me more about how software engineer dress?
>
> [A:] Well, in our company you can dress as you like.
>
> [Q:] I understand ... But in some companies people are not told what they are supposed to wear, but they still dress in a particular way, e.g. a suit.

> [A:] Sure. You can wear a suit and tie here, if you want, sure. But people prefer not to.
>
> [Q:] Why? What would happen, if you did?
>
> [A:] Well, people would snigger, naturally. I guess it is difficult to describe, it's specific... When somebody has a meeting with a client, people start commenting on it.
>
> [Q:] Really? In what way?
>
> [A:] They make silly remarks, like "ho ho!" and so on ...comments that you're in a suit, and that you look so cute, and that maybe you are going to propose to your girlfriend.

So wearing a suit was accepted only for meeting clients (Sand7):

> [Q:] Do you often wear a suit?
>
> [A:] Occasionally, and only if I am told earlier that I have a meeting with a client, then yes. Let's say a day earlier I am told, so I wear a tie, and go to the client. You know, you have to look presentable when you meet a client (...)

Software engineers described, how they rejected a formal dress-code within the organization, while observing it outside[1]. When outside the organization programmers felt they represented the firm and so for this reason they accepted, even if not welcoming, the external dress code. Wearing a suit was a sign of respect for the client. Inside the firm, though, they perceived wearing a suit and tie as being beyond the pale.

The failure to wear a suit is a sign of not belonging to certain organizational groups, usually management, marketing and sales. Crane (1997) views bohemian dress style as an avant-gardist aesthetic practice meaningful mainly in opposition to mainstream fashion. For programmers, a formal dress code was not perceived as a sign of high status, as is the case in many other contexts (Rafaelli and Pratt, 1993). They did not emphasize their independence by common choice of some particular clothes. On the contrary, their status was defined rather by their freedom of preference in this respect. They enjoyed their freedom from a formal dress code, instead joking about their peers who had to conform. Paradoxically, dressing supposedly carelessly could be viewed here as a sign of power and competence. But to understand the meaning of this symbolism, it is necessary to explore software engineers' views of management, and of organizations.

Organizing and managers

Most of interviewees did not mince their words when they spoke of big companies. Many criticized corporations, without prompting (Wodan6):

[Q:] Would you like to work in a bigger company?

[A:] No. I get goose pimples whenever I think about big firms. I mean, well, I don't really like the idea of them. Even without thinking too much about it, it's obvious that the bigger the organization, the less flexible it is. Things take longer, everything has to go through more people. And second of all, I was never really impressed ...I was never impressed by big companies, all this paperwork, I think I always associated it with something bad ...maybe with propaganda.

[Q:] Propaganda?

[A:] Yeah. Just think what happens in these companies ... They brainwash people: you are the best, you are a team member, you are ... you have to achieve something, and then something else, and else ...That's my impression and this is what my friends, who work in bigger firms, tell me. You are simply a cog in the wheel there.

The programmer described a typical big corporation. He stressed the ideological character of managerial rhetoric by labeling it "propaganda". Many other informants expressed a similar distaste for managerial rhetoric (Sand1):

[Q:] What irritates you about the company?

[A:] Oh, occasionally many things...I think what I dislike most is when somebody from the higher echelons, some big boss, comes here and blabs on about how much they care about our work and how important we are, and when we ask about the promised raises he behaves as if we offended him. Seriously, we had this situation a couple of months ago. And we have this sort of blah-blah on an everyday basis.

The interviewees clearly identified the agitprop character of many of the official organizational announcements. "Blabbing" and "blah-blah" was how they saw the bosses' rhetoric. Programmers stressed their anti-ideological stance. Ideology itself was denounced, even if was not particularly striking in organizations they worked for, it constituted the most disliked element of the stereotypical corporation. Whether it was what they experienced in their organizations or not, managerial indoctrination was listed as being exceptionally repulsive.

Correctional officers in Klofas and Toch's (1982) research often shared a belief that the "mainstream culture" of their profession does exist, even though they did not belong to it. Similarly, interviewees in the current study, even the ones from really big firms, unanimously said that even though the companies they work for are exceptions to the rule, in general big organizations are unfriendly and hierarchical.

All programmers from two corporations operating in the international markets made such statements (Sand18):

> [Q:] What attracted you to this company?
>
> [A:] Well, I'd say I came practically in off from the street, I didn't know anyone here. But I must say, that it is different from what people say, that the company's big, takes advantage of people, and so on. The atmosphere here is definitely humane, we're a separate team anyway, a separate cell, a bit away from the rest of the firm, so you don't normally even notice that the company is big, maybe just occasionally.

For a pretty neutral question about the reasons for working in the firm he worked, the interviewee was making excuses, questioning "what people say". He justified his choice and emphasized that even though the whole company is big, his unit is small and structurally independent, indeed "definitely humane"; in contrast to elsewhere. Other interviewees, in similar tone, referred to bureaucracy as a typical and particularly striking flaw of bigger firms (Wodan4):

> [Q:] What made you apply for a job in this company, when you didn't even know it?
>
> [A:] Small size; that for a start.
>
> [Q:] Why is that important?

> [A:] Well, its not that being small is important, it's that being big is a problem. I mean, I, you know, have friends from university, a couple of them at least, who went to work with big corporations, I won't give you their names, but there the internal problems are somehow multiplied.
>
> [Q:] What kind of internal problems?
>
> [A:] For example: a stupid boss. And apart from this there is also another, namely bureaucracy. I mean, in particular, it happens often, that a guy gets an order ... I mean for example, gets an order, spends two days on preparing some document, and then a bigger boss comes and throws this away. Says that it's something totally else. In small firms you can communicate directly, there are fewer people in general; work is not wasted as much. In big corporations nobody minds that somebody lost two days on some crap, that wasn't even used after all. And you have to spend lots of time on in-fighting. You have to intrigue and drive people out, so as to prevent yourself from being driven out.

The inevitability of power struggles in bureaucracies was emphasized in many interviews. Software engineers criticized the highly political nature of organizational life and its irrationality, resulting from undue bureaucracy. They also resented the fact that in many companies they have to engage in the game on managerial terms, so as not to be marginalized.

Negative views of management (stupid boss) were repeated in many interviews. Large company size was associated with inefficiency and in-fighting. There was a common fear of incompetent bosses in big corporations (Minicorp3):

> [Q:] What do you like about the company you work for?
>
> [A:] First of all, that it's small.

[Q:] So what's so important about being small?

[A:] Well, a friend from work, who brought me here, put it nicely: he said there is no dilbertization. I guess you know what I mean. And I think it's because the company isn't big, you know.

[Q:] What is dilbertization?

[A:] Well, I mean, it's not an exact term, ok? But in general, when there is no dilbertization, you're not talking about big companies and you're just working faster, without problems. I mean problems ... From what I hear, in bigger companies people have to do the same job several times, because a manager changed his mind, a company changed its mind, and so on.

A popular cartoon caricaturing corporations' absurd behaviour, the comic strip "Dilbert" was used as a catch-phrase for big companies. Again, even though the questions were focused on the companies programmers worked for, rather than on management, the figure of the boss was brought up spontaneously, as an important icon of the company. The manager was described as somebody who changes his mind, and makes "people" work not only more, but also for nothing. Such negative perception was intensified when I asked about managers directly (Wodan4):

[Q:] So, I have a final question. Could you give some advice to managers, people who lead IT projects? Some dos and don'ts? What would you advise?

[A:] Well, my view of a bad manager is somebody, who knows nothing but pretends to know a lot.

Software engineers question not only managerial competence in IT projects. They also challenge managerial knowledge *per se*. People who pursue a career in management do so because they cannot do anything else. As a result of their technical ignorance they make unrealistic demands, the most common complaint of programmers.
Many of the interviewees expressed frustration with the fact that their superiors perceived software as easily modifiable. A typical example is as follows from Wodan:

Tom, the project manager opens a meeting with the programming team. He refers to a discussion with a client and describes the list of changes the client asks for. Programmers take notes, sometimes asking questions for clarification. Occasionally they make remarks like "Will do" "OK, if THAT'S what they want ..." However, one of the changes leads to an astonished reaction. Peter, a member of the programming team, says "But it would require a major revision of the program". Somebody adds "It doesn't make any sense; we'll have to start all over again". Tom is very apologetic, he explains that the client is really important, and the change has already been approved by the company. He dismisses all doubts raised. By way of consolation he adds that he was able to negotiate a deadline extension, but two programmers shake their heads, even though they say nothing. Once the meeting is over the team stays in the room to decide what to do. Mark, one of the programmers, says "It's just software" and everybody laughs. Later I ask him what it meant. "Oh, you know. They never would ask an engineer to rebuild a bridge, or something physical, right? But they think software is like a couple of written lines, like a document or something, that you can alter whenever you want and how you want. But it's like rebuilding a bridge, or worse, as you don't really understand what one change will imply elsewhere. Once before we had a similar situation, and there was some ridiculous revision of the specification almost near the end of

> the whole project, and somebody said *Take it easy, it's just software*. It was so absurd, you know. So since then we say that and everybody knows, what it means".

The tension described had its roots in managerial misconception of programmers' work. Although interviewees did give some examples of knowledgeable managers, they faced misunderstanding and even disparaging of their own work on daily basis. But managers were criticized not only for their lack of skill. Another issue pointed out by more than a half of the interviewed was a fundamental difference in perception of software as a final product (Sand17):

> [Q:] What could you tell me from your own experience about relations between a software engineer and manager?
>
> [A:] Well, it's not too good, I'd say. I think there are two approaches. I mean, a manager always thinks about effectiveness. It doesn't matter how something works, it only has to work. And among programmers, there are two groups, as I said. Some do, excuse the expression, fart around, and follow managerial commands. But there are also programmers, who do something real, and they want to do it well, and doing this "well" is a bone of contention. For a manager doing something "well" means that something works. For a programmer it means that a program works, is stable, and has additional features, and so on.

Following managerial instructions was compared to "farting around". It was contrasted to a stance, in which a programmer really cares about the product, while a manager only wants it to be workable on basic level. Indeed, it is not unusual for managers to identify with the organization they work for, and for employees to identify more with the products they create (Sievers, 1990). Software engineers, however, expressed also a deep contempt for what managers do. Much of it was related to the way managers treated them. As one of the programmers put it (Visualprog3):

> [Q:] Could you, based on your own experience, give some advice to IT managers?
>
> [A:] I'd say...In general, a good manager does not watch over you the whole time you know. I mean if he knows what I have to do, and I know what I have to do, and we agree more or less when it can be done, what's the point? Maybe it's just me, some people like to be led by hand, but in my view the best managers I worked with understood, that ok, I can update them every 10 minutes or in real time, but then my job will only consist of describing why I can't do programming at all because of this reporting, you know? Not to mention that if you're in the middle of something and somebody comes and asks you to write something immediately, you drop everything, start doing this report, and can't return easily to what you did earlier. It's not that simple.

What was noticeable was that most of advice interviewees dispensed concerned the things that managers should avoid doing. In software engineers' view the best managers were those who intervened as little as possible, and simply followed a laissez-faire principle. In fact, the community of software engineers seemed to define the managers as redundant at best.

The best manager was one notable by his absence as managers were portrayed as stupid. However, it was recognized that managers were powerful and so programmers have to take part in their games, much as they tried to avoid this. The software engineers were unanimously sceptical and even hostile towards formality and procedures, which they associated with the companies they worked for, or with the general idea of how the firms operate. Whenever their organizations were described in a positive way, they were praised for being exceptions to the general rule. Although interviewees admitted the usefulness of organizing *per se*, they all criticized it first, and only then added disclaimers.

Interestingly, enough, these views were recognized and even accepted by managers. They often made an effort to minimize bureaucracy for programmers. The following scene from the company orientation program at Visualprog was particularly striking:

> Vera, the HRM officer for Visualprog is welcoming two new engineers. One of them will join the software group, the other will be supporting the business development department. Even though this is the first day at the company for the two of them, they both wear sweaters. Vera, on the other hand, wears a suit. She spends about 30 minutes describing the history of the company. Quite often she emphasizes the uniqueness of the environment they are going to work in (...) "You're part of a small company that was organizationally designed to be agile. People come here and say that everything here is a chaos. But it is just different. We're really not big on org charts ..." Both engineers nod and read the company's employees benefits folder. Vera speaks fluently and clearly, quite often repeating herself, probably on purpose. Maybe it's just an impression, but it seems as though she has delivered this speech many times before. She takes a piece of paper out of her folder. "You won't have to memorize this, it's funny we have this on the walls, but nobody really takes any notice of it" she says and recites the company's mission. Then she goes on to describe the reasons the company had for choosing ISO quality system several years ago. "Our quality system, all respect due to the creators, is not very user friendly" she adds with a smile. She hands a procedure chart to both of them. "You're welcome to read all of our quality procedures. If you do, let me know – I will be really impressed. But seriously, you won't have to do so much paperwork here; we're kind of sensitive enough to make sure you don't have to'".

The company's agility and "chaos" are presented as strong points for engineers. The ISO system, quite strict about procedures, is depicted as not necessarily inevitable (the engineers are ironically informed that they do not have to read all of the procedures) and as not entailing much paperwork. Caricatured and selective as this view of software engineers in the eyes of managers may be, it is still worth noticing that in the companies studied, managers quite often made allowances for programmers, commonly lowering bureaucratic expectations of them.

Deep-seated or just a veneer?

The hostility of software engineers to managers may be influenced by group stereotypes (Gill, 2003). Many of the symbolic gestures described have only surface meaning, and performing opposition may be in fact a sort of behavioral artifact. This would be particularly understandable when considered that, in contrast to the specialists described by Barley and Kunda (2004) in their study, interviewees were not contract workers and they stayed in the organizations by their own choice, in spite of available alternatives. This could mean that they were not exasperated enough to leave the organization, but also that they need to channel their negative reactions within the company. When knowing how well freelance workforce may be off, and while still being reluctant to leave the relatively safer payroll jobs, their aspirations and emotional reaction towards organizations could just as well be under the influence of those "hired guns". Itinerant programmers, reviving the Wild West archetype (and similarly to hackers, another important icon of this profession – Thomas, 2002) cultivate the idea of freedom and criticize "big corporations". Interviewees might have been simply borrowing this language and using it perfunctorily.

However, even surface actions and stereotypes show the ways in which organizational reality is constructed. They constitute nothing more, but also nothing less than a particular kind of stories by which actors organize and make sense of their workplace (Boje, 1991; Feldman and Skölberg, 2004).
Software engineers, similar to architects in this respect, have to define their profession in relation (or opposition) to the clients and organizations they work for (Larson, 1993, p. 144; Kociatkiewicz and Kostera, 2003). Organizations exist as speech communities, sharing an intersubjectively created system of meanings (Barley, 1983). In this light language of the analyzed group, forms reenactment of their roles. They reproduce sense and artifacts of daily work (Czarniawska-Joerges, 1992). Moreover, within the current study, the main focus is on actors' expressions, and the researcher should not judge which of them are deep, and which is a veneer. The power of naming is the power to shape reality. Additionally, within the methodology adopted, it may be quite difficult to discern the veneer from the essence, as the main purpose of the study is to analyze what the interviewees say, rather than guessing what they may mean (Rottenburg, 1994). Indeed, in this paper surface acting is regarded as equally important to deeply held beliefs. In the following paragraphs we interpret the programmers-managers hostility, and the unusual denouncement of traditional career and organizational values in the context of theories around professionalization.

Professionalization?
It may be useful to try to apply professionalization theory to the processes discussed. The tension between managers and software engineers could be understood as resulting from professionalization of engineers currently taking place. In some sense this tension is understandable in terms of antagonism between the managing and the managed, based on the old strategy of struggle, resulting in redefining the relations of organizational power (Foucault, 1982; Latour, 1986).

Originally professions were defined in terms of the altruistic provision of high standards of service to society (Carr-Saunders, 1966; Wilensky, 1964). Over the last 30 years it has been demonstrated, that it is also useful to use this term to describe a common trend among occupations in search of authority and recognition, and seeking to secure their own interests and privileges (Abbott, 1988; Alvesson, 1993). In this context, questioning managerial competence could be perceived then as a typical struggle regarding who should define the product and its standards. In more general terms it could be seen as an attempt of attaining social position and privileges. The extent to which a worker has control over the process of production, as well as to which vocational group is privileged to evaluate the outcome of work, have been emphasized as important for the advance of professionalization (Johnson, 1972; Trice, 1993). Viewed in this light, software engineers' dislike of their superiors would be nothing else but a manifestation of their subculture's struggle for legitimacy, quite typical for many occupations. In this sense the professionalization approach could be used then as an explanatory metaphor for understanding the interviews.

However, the traditional understanding of professions may be not fully applicable in case of software engineers. In order to be regarded as a profession, they would have to fulfill many criteria, including long formal and standardized education, barriers to entry, own code of ethics, peers' fraternities, client orientation, application of scientific methods, etc. (Abbott, 1988; Brante, 1988). Clearly, software engineers do not meet at least some of these requirements. For example, they very seldom unionize (Milton, 2003; Jaarsveld, 2004) and rarely belong to professional associations. Indeed, interviewees on many occasions showed that they not only did not fulfill the requirements of the model of professionalization, but also that they could not care less about the concepts it emphasized as important.

Something else that was at odds with the model of professionalization was that software engineering often calls for creativity, intuition and improvising, which are to some extent in conflict with a strict professional training.

Finally, professionalization theory refers to a process of formalizing the status and organization of some occupation, and this is not what software engineers favored. In this light it is also worth mentioning that the software engineers interviewed commonly categorize their managers, clients, but also their peers by referring to their knowledge, rather than to their formal status as in the following interview extract:

> [Q:] Can you give an example of a really good manager?
>
> [A:] Well, once I had a project leader, who was really outstanding. He was a former programmer, but he hadn't written a line of code for years, and thank God he knew he wasn't able to do this anymore. But he had a good grasp of what is possible and what is not, he really understood how things work. And he really knew something about projects, it was not that he went in for micro managing your work.

In describing other programmers the interviewees also often referred to their competences, irrespective of formal position or age ("he is quite new to our company, but he really knows the technology well ..." "It's ridiculous that people with such poor understanding of the environment are allowed to coordinate the whole team at all ..."). Even among the interviewees some programmers were perceived as "just coders" and their work was described as purely reproductive, just as a couple of project leaders, although high in the organizational hierarchy, were not respected because of their supposed ignorance.

What is certain is that interviewees in general did not particularly like managers showing up on the horizon at all. As the engineers said, they were able to control their own work to a large extent, irrespective of managerial interventions and checks. According to software engineers the managers also did not, and even were not able to delve into the process of software creation with any competence, being able only to judge outcomes. Moreover, other than simply communicating client's needs, managers rarely participated in discussions of how particular problems should be solved.

Despite this lack of understanding, management persisted in trying to exert its authority. For example, many programmers were subject to some forms of basic personal surveillance: in four of the companies researched the time spent at work was tracked (both by the clock and by computer logs). One of the stories recounted by interviewees at Sand was a particularly good example of how companies try to control software engineers, and at the same time how they resist it. The story was about a programmer, who supposedly:

> ... logged into the network in his cubicle, opened a couple of applications, left his jacket on the chair, put a cup of water on the table, and just drove with his wife to the mountains till Monday. The funniest thing is that nobody really noticed he wasn't there. I mean, probably

some people knew he’s not around, but nobody said anything and the manager thought he’s just working hard somewhere else.

Although perhaps apocryphal, this story describes both the resented surveillance, and the hero outwitting the hated system.

In fact, the use of a professionalization discourse may actually turn out to be a managerial device used to exert more control over the IT “professional” (Kraft, 1977; Greenbaum, 1979), a sort of ideological device validating a given rhetoric (Prasad and Prasad, 1994).

It has been commented that the importance of organization (and thus managers, in opposition to vocational groups) is historically relatively new (Whalley and Barley, 1997; Winter and Taylor, 2001). Thus, currently we can observe trends in the workplace that restore the former significance of particular occupations. The tension discussed is described then in terms of emerging “knowledge workers” organization or so-called intellectual capital (Bell, 1973, 1989; Mallet, 1975; Hippel, 1988; Senge, 1990; Drucker, 1993; Stewart, 1997; Styhre, 2003). Perhaps, a new culture is developing in relation to “new specialists” – its distinguishing marks being competence-based authority, horizontal flat structures, the demise of management and low formalization (Zabusky and Barley, 1996). These characteristics are true of many IT companies and may suggest that software engineers exist in opposition to traditional bureaucracies as their occupational identity is based mainly on knowledge. Thus, they denounce the bureaucracy, the vertical career structures, and formal authority of managers as reinforcing the old system they want to replace. In this sense software engineers could be labeled as “organizational professionals” (Freidson, 1986) in that they exercise their competence and power within, rather than over organizations, in distinction to the classic professions.

However, a totally different interpretation is possible, too. Software engineers may in fact be acting as the passive carriers of organizational roles, only agreeing to assume the role of a rebel. This view will be discussed in the conclusions.

Conclusions

Managerial power in many organizations can be understood in terms of their ability to impose their definition of their own cultural terrain on other groups and so to force the adoption of their own vocabulary (Rosen, 1991). Software engineers may constitute a major threat to managerial rhetoric, to managerial monophony (Ho¨pfl, 1995). As Zabusky (1997, p. 129) observes:

> [T]his conjunction does not represent a simple juxtaposition of two complementary forms; instead, it involves a contest for legitimacy, authority and autonomy within contemporary organizations. The contest is played out particularly among these technicians who are coming to work in bureaucratic organizations in increasing numbers.

Czarniawska-Joerges (1988) describes organizational ideology as a system of ideas that define the local reality, including the desired state of matters and ways to reach it. These ideas permeate the organization, imposing norms on the participating actors, who have to react towards them, one way or another. However, these ideas do not come from nowhere: organizations follow the ideas their stakeholders consider to be particularly valid. In this connection Meyer and Rowan (1977, p. 341) comment:

> ... the formal structures of many organizations in postindustrial society (Bell, 1973) dramatically reflect the myths of their institutional environments instead of the demands of their work activities.

According to this perspective, in order to be successful economically, organizations should abandon their traditional structure, decrease bureaucracy, and rely on knowledgeable gurus. Whether organizations with the "new specialists" playing major roles are indeed more effective is another issue.

According to Schön (1983), the myth of finally reaching the stage where organizations will be knowledge-driven, and knowledge industries will be as important for the economy as were formerly railroad or steel, is an old idea (Mallet, 1975). Indeed, for many years futurologists have proclaimed the era of the technical expert. Managerial functional literature has also gradually introduced the idea that specialists in general, and software engineers in particular, play a significant role in organizations. This role requires a particular approach and abandoning older management style, as programmers are too distinctive and important, to be treated like other employees (Licker, 1983; Prager, 1999). Brante (1988, p. 123) summarizes this point, when commenting that:

> In contrast to the old bureaucracy, their positions do not rest on legal authority but on argument, reason, and knowledge. Therefore, they stand in a politically contradictory relation to the "old guardians" which they regard as irrational and ignorant.

This view is arguably applicable to software engineers as they are "professionals" in that they are endowed with high esteem and authority. However, the ideal of the software engineer is that of the organizational rebel.

Managers treat programmers as being in the avant-garde of the future workplace, and this way the programmers fulfil this role. Indeed, their rejection of managerial culture is even expected and so required by the environment they work in. Ullman (1995) comments in this connection:

> The research is being funded through a chain of agencies and bodies which culminates in the Japan Board of Trade. The head of the sponsoring department comes with his underlings. They all wear blue suits. They sit at the conference table with their hands folded neatly in front of them. When they speak, it is with the utmost discretion; their voices are so soft, we have to lean forward to hear. Meanwhile, the research team behaves badly, bickers, has the audacity to ask when they'll get paid.
>
> The Japanese don't seem to mind. On the contrary, they appear delighted. They have received exactly what their money was intended to buy. They have purchased bizarre and brilliant Californians who can behave any way they like. The odd behavior reassures them: Ah! These must be real top-rate engineers!

Similar scenes have occurred in the companies studied. Vera from Visualprog when introducing new employees to the company, after describing the organizational structure and ISO policies to them, still on many occasions later emphasized they will not be "required to fill in the paperwork, as elsewhere" or "expected to spend most time on reporting, rather than actual work". Software engineers are expected to dislike procedures and resist bureaucracy. Indeed, the role they assume of "anarchistic professionals" is perhaps necessary for other reasons.

For example, their knowledge can be viewed as standardized, systematic, and impersonal. Only then it is justifiable and reasonable to provide them with "specifications" of the needed "construction" rather than engage them in communication/brainstorming with the client on what the program should actually do. In such a setting the knowledge associated with software development is more likely to be perceived as quantitative, codified, and explicit rather than qualitative, intuitive and tacit. Norm goes before creativity, standard education before genius. As a result organizations and managers tend to ignore individual aspects of coding, as well as its unique, extremely contextual character (in most cases advanced IT systems require lots of adjustments and modifications, if not revisions, for any additional client). For example, companies treat programmers as interchangeable and they follow a misleading man-month myth, believing that adding man-power to a project may help in finishing it more quickly (in fact usually it is just the opposite – Brooks, 1995). The same approach leads to the "it's just software" principle, described by the informants. In fact, these assumptions are, Bryant (2000) shows, a major drawback in understanding programming, and one of the main obstacles in successful communication between managers and programmers. This is exactly what interviewees remarked on in their stories on managerial incompetence.

Another reason is that their understanding of their own role calls for the creation of schedules, division of labor, as well as linear communication with the client. This inevitably results in perceiving software as transferable, with assignable responsibilities and tasks, and as designed according to an initial specification with just minor changes introduced later. It also goes without saying that such a definition of programmers' position is much more convenient for managers as well. On the one hand, software engineers are heroes of the future, fulfilling the myth of knowledge-intensiveness. On the other they have to write programs, which are extremely complex and contextual, as if they were easily replicable and modifiable; they are expected to (re)produce rather than to create. No wonder that being treated this way, and explicitly persuaded to renounce structures and procedures, they also do not particularly like their superiors whom they regard as lazy stupid careerists.

Note

1. The old debate on the sense or otherwise of defining the boundaries of an organization (Weick, 1979) and on the use of notions such as "outside" or "inside" does not constitute the subject of this paper.

References

Abbott, A.D. (1988), *The System of Professions: An Essay on the Division of Expert Labor*, University of Chicago Press, Chicago, IL.

Alvesson, M. (1993), "Organizations as rhetoric: knowledge-intensive firms and the struggle with ambiguity", *Journal of Management Studies*, Vol. 30, pp. 997-1018.

Barley, S. (1983), "Semantics and the study of occupational and organizational cultures", *Administrative Science Quarterly*, Vol. 28, pp. 393-431.

Barley, S.R. and Kunda, G. (2004), *Gurus, Hired Guns, and Warm Bodies: Itinerant Experts in a Knowledge Economy*, Princeton University Press, Princeton, NJ.

Bell, D. (1973), *The Coming of Post-Industrial Society: A Venture in Social Forecasting*, Basic Books, New York, NY.

Bell, D. (1989), "The third technological revolution and its possible socioeconomic consequences", *Dissent*, Vol. 36 No. 2, pp. 164-76.

Boje, D.M. (1991), “The storytelling organization: a study of story performance in an office-supply firm”, Administrative Science Quarterly, Vol. 36, pp. 106-26.

Brante, T. (1988), “Sociological approaches to the professions”, Acta Sociologica, Vol. 31, pp. 119-42.

Brooks, F.P. (1995), The Mythical Man-month: Essays on Software Engineering, Addison-Wesley Pub. Co., Reading, MA.

Bryant, A. (2000), “Metaphor, myth and mimicry: the bases of software engineering”, Annals of Software Engineering, Vol. 10, pp. 273-94.

Bucciarelli, L.L. (1988), “An ethnographic perspective on engineering design”, Design Studies, Vol. 9, pp. 159-78.

Carr-Saunders, A.M. (1966), “Professionalization in historical perspective”, in Vollmer, H.M. and Mills, D.L. (Eds), Professionalization, Prentice-Hall, Englewood Cliffs, NJ.

Cooper, M. (2000), “Being the ‘go-to guy’: fatherhood, masculinity, and the organization of work in Silicon Valley”, Qualitative Sociology, Vol. 23, pp. 379-408.

Coser, L.A. (1974), Greedy Institutions; Patterns of Undivided Commitment, The Free Press, New York, NY.

Crane, D. (1997), “Postmodernism and the avant-garde: stylistic change in fashion design”, Modernism/Modernity, Vol. 4, pp. 123-40.

Czarniawska-Joerges, B. (1992), Exploring Complex Organizations: A Cultural Perspective, Sage, Newbury Park, CA.

Czarniawska-Joerges, B. (1988), Ideological Control in Non-ideological Organizations, Praeger, New York, NY.

Drucker, P.F. (1993), The New Society: The Anatomy of Industrial Order, Transaction Publishers, New Brunswick, NJ.

Feldman, M.S. and Sko¨lberg, K. (2004), “Stories and the rhetoric of contrariety: subtexts of organizing (change)”, Culture and Organization, Vol. 8, pp. 275-92.

Foucault, M. (1982), “The subject and power”, in Dreyfus, H.L. and Rabinow, P. (Eds), Michel Foucault, beyond Structuralism and Hermeneutics, Harvester Wheatsheaf, London. Freidson, E. (1986), Professional Powers: A Study of the Institutionalization of Formal Knowledge, University of Chicago Press, Chicago, IL.

Garsten, C. (1994), Apple World: Core and Periphery in a Transnational Organizational Culture, Stockholm University Press, Stockholm.

Gephart, R.P. (2002), “Introduction to the brave new workplace: organizational behaviour in the electronic age”, Journal of Organizational Behavior, Vol. 23, pp. 327-44.

Gill, M.J. (2003), “Biased against ‘them’ more than ‘him’: stereotype use in group-directed and individual-directed judgments”, Social Cognition, Vol. 21, pp. 321-48.

Greenbaum, J.M. (1979), In the Name of Efficiency: Management Theory and Shopfloor Practice in Data-processing Work, Temple University Press, Philadelphia, PA.

Hall, R.H. (1986), Dimensions of Work, Sage, Beverly Hills, CA.

Hearn, L. (2005), “IT workers dubbed ‘worst dressed’”, The Sydney Morning Herald, Sydney, available at: www.smh.com.au/articles/2005/11/17/1132016909640.html

Hertzum, M. (2002), “The importance of trust in software engineers’ assessment and choice of information sources”, Information and Organization, Vol. 12, pp. 1-12.

Hippel, E.V. (1988), The Sources of Innovation, Oxford University Press, New York, NY.

Hochschild, A.R. (1997), The Time Bind: When Work becomes Home and Home becomes Work, Metropolitan Books, New York, NY.

Hofstede, G.H. (1980), Culture's Consequences: International Differences in Work-related Values, Sage, Beverly Hills, CA.

Höpfl, H. (1995), "Rhetoric and the threat of ambivalence", Studies in Cultures, Organizations and Societies, Vol. 1, pp. 175-87.

Jaarsveld, D.D.V. (2004), "Collective representation among high-tech workers at Microsoft and beyond: lessons from WashTech/CWA", Industrial Relations, Vol. 43, pp. 364-85.

Jackall, R. (1988), Moral Mazes: The World of Corporate Managers, Oxford University Press, New York, NY.

James, G. (1986), The Tao of Programming, Infobooks, Santa Monica, CA.

Jemielniak, D. (2005), "Time for IT: timing in software projects", in Khosrow-Pour, M. (Ed.), Managing Modern Organizations with Information Technology, Idea Group Inc., Hershey, PA.

Johnson, T.J. (1972), Professions and Power, Macmillan, London. Johnson, M.R. (1990),

Business Buzzwords: the Tough New Jargon of Modern Business, Blackwell, Oxford.

Kanter, R.M. (1977), Men and Women of the Corporation, Basic Books, New York, NY.

Kawasaki, G. (1990), The Macintosh Way, Scott, Foresman, Glenview, IL.

Kidder, T. (1981), The Soul of a New Machine, Little, Brown & Co., Boston, MA.

Klofas, J. and Toch, H. (1982), "The guard subculture myth", Journal of Research in Crime and Delinquency, Vol. 18, pp. 272-94.

Kociatkiewicz, J. and Kostera, M. (2003), "Shadows of silence", Ephemera, Vol. 4, pp. 305-13.

Kraft, P. (1977), Programmers and Managers: The Routinization of Computer Programming in the United States, Springer-Verlag, New York, NY.

Kunda, G. (1992), Engineering Culture: Control and Commitment in a High-tech Corporation, Temple University Press, Philadelphia, PA.

Kunda, G. and Van Maanen, J. (1999), "Changing scripts at work: managers and professionals", Annals of the American Academy of Political & Social Science, Vol. 561, pp. 64-80.

Kärreman, D. and Alvesson, M. (2004), "Cages in tandem: management control, social identity, and identification in a knowledge-intensive firm", Organization, Vol. 11, pp. 149-76.

Larson, M.S. (1993), Behind the Postmodern Facade. Architectural Change in Late Twentieth-century America, University of California Press, Berkeley, CA.

Latour, B. (1986), "The powers of association", in Law, J. (Ed.), Power, Action and Belief – A New Sociology of Knowledge?, Routledge&Kegan Paul, London.

Licker, P.S. (1983), "The Japanese approach: a better way to manage programmers", Communications of the ACM, Vol. 26, pp. 631-6.

Mallet, S. (1975), Essays on the New Working Class, Telos Press, St Louis, MO.

Martin, J. (1993), Cultures in Organizations: Three Perspectives, Oxford University Press, New York, NY.

Meyer, J.W. and Rowan, B. (1977), "Institutionalized organizations: formal structure as myth and ceremony", The American Journal of Sociology, Vol. 83, pp. 340-63.

Milton, L.P. (2003), "An identity perspective on the propensity of high-tech talent to unionize", Journal of Labor Research, Vol. 24, pp. 31-53.

Ouchi, W.G. (1981), Theory Z: How American Business can Meet the Japanese Challenge, Addison-Wesley, Reading, MA.

Perlow, L.A. (1997), Finding Time: How Corporations, Individuals, and Families can Benefit from New Work Practices, ILR Press, Ithaca, NY.

Perlow, L.A. (1998), "Boundary control: the social ordering of work and family time in a high-tech corporation", Administrative Science Quarterly, Vol. 43, pp. 328-57.

Peters, T.J. and Waterman, R.H. (1982), In Search of Excellence: Lessons from America's Best-run Companies, Harper & Row, New York, NY.

Pin˜eiro, E. (2003), "The aesthetics of code. On excellence in instrumental action", unpublished PhD thesis, available at: www.lib.kth.se/Fulltext/pineiro031128.pdf

Pondy, L.R. (1983), Organizational Symbolism, JAI Press, Greenwich, CT.

Prager, K.P. (1999), "Organizational culture and the IT professional", Information Systems Management, Vol. 16, pp. 12-18.

Prasad, P. and Prasad, A. (1994), "The ideology of professionalism and work computerization: an institutionalist study of organizational change", Human Relations, Vol. 47, pp. 1433-58.

Rafaelli, A. and Pratt, M.G. (1993), "Tailored meanings: on the meaning and impact of organizational dress", The Academy of Management Review, Vol. 18, pp. 32-55.

Rosen, M. (1991), "Breakfast at Spiro's: dramaturgy and dominance", in Frost, P.J., Moore, L.F., Louis, M.R., Lundberg, C.C. and Martin, J. (Eds), Reframing Organizational Culture, Sage, London.

Rottenburg, R. (1994), "From socialist realism to postmodern ambiguity: East German companies in transition", Industrial and Environmental Crisis Quarterly, Vol. 8, pp. 71-91.

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

Schön, D. (1983), The Reflexive Practitioner. How Professionals Think in Action, Basic Books, New York, NY.

Senge, P.M. (1990), The Fifth Discipline: The Art and Practice of the Learning Organization, Doubleday/Currency, New York, NY.

Sievers, B. (1990), "Zombies or people – what is the product of work", in Turner, B.A. (Ed.), Organizational Symbolism, De Gruyter, New York, NY.

Stewart, T.A. (1997), Intellectual Capital: The New Wealth of Organizations, Doubleday/Currency, New York, NY.

Styhre, A. (2003), Understanding Knowledge Management: Critical and Postmodern Perspectives, Copenhagen Business School Press, Copenhagen.

Thomas, D. (2002), Hacker Culture, University of Minnesota Press, Minneapolis, MN.

Trice, H.M. (1993), Occupational Subcultures in the Workplace, ILR Press, Ithaca, NY.

Trice, H.M. and Beyer, J.M. (1993), The Cultures of Work Organizations, Prentice-Hall, Englewood Cliffs, NJ.

Ullman, E. (1995), "Out of time: reflections on the programming life", in Brook, J. and Boal, I. (Eds), Resisting the Virtual Life: The Culture and Politics of Information, City Lights Publishers, San Francisco, CA.

Vallas, S.P. (2003), "The adventures of managerial hegemony: teamwork, ideology, and worker resistance", Social Problems, Vol. 50, pp. 204-25.

Van Maanen, J. (1991), "The smile factory: work at Disneyland", in Frost, P.J., Moore, L.F., Louis, M.R., Lundberg, C.C. and Martin, J. (Eds), Reframing Organizational Culture, Sage, London.

Weick, K.E. (1969), The Social Psychology of Organizing, Addison-Wesley, Reading, MA. Whalley, P. and Barley, S. (1997), “Technical work in the division of labor: stalking the wily anomaly”, in Barley, S. and Orr, J.E. (Eds), Between Craft and Science, Cornell University Press, London. Whyte, W.F. and Whyte, K.K. (1984), Learning from the Field: A Guide from Experience, Sage, Beverly Hills, CA. Wilensky, H.L. (1964), “The professionalization of everyone?”, The American Journal of Sociology, Vol. 70, pp. 137-58. Winter, S.J. and Taylor, L.S. (2001), “The role of information technology in the transformation of work”, in Yates, J. and Van Maanen, J. (Eds), Information, Technology and Organizational Transformation. History, Rhetoric, and Practice, Sage, London. Zabusky, S.E. (1997), “Computers, clients, and expertise: negotiating technical identities in a nontechnical world”, in Barley, S. and Orr, J.E. (Eds), Between Craft and Science, Cornell University Press, London. Zabusky, S.E. and Barley, S. (1996), “Redefining success: ethnographic observations on the careers of technicians”, in Osterman, P. (Ed.), White Collar Careers, Oxford University Press, Oxford.

Further reading

Burawoy, M. (1991), Ethnography Unbound: Power and Resistance in the Modern Metropolis, University of California Press, Berkeley, CA.

Clifford, J. and Marcus, G.E. (1986), Writing Culture: The Poetics and Politics of Ethnography: A School of American Research Advanced Seminar, University of California Press, Berkeley, CA.

About the authors
Dariusz Jemielniak is Assistant Professor of Management at Kozminski Business School, Warsaw, Poland. He was visiting researcher at Cornell University (2004-2005), is currently visiting researcher at Harvard University and will have the same appointment at the University of California Berkeley in 2008. He is a co-editor of The Handbook of Research on Knowledge-Intensive Organizations (2009) and of Management Practices in High Tech Environments (2008). He has received scholarships from The Fulbright Foundation, The Foundation for Polish Science and from The Collegium Invisibile. His areas of interest include organizational practices in knowledge-intensive environments and he has done qualitative studies of engineers including software engineers, and is currently conducting a follow-up study on lawyers. Dariusz is also starting to undertake action research. Dariusz Jemielniak can be contacted at: darekj@kozminski.edu.pl