BUSINESS, MANAGEMENT AND EDUCATION
ISSN 2029-7491 print / ISSN 2029-6169 online
2012, 10(1): 77–90
doi:10.3846/bme.2012.07

MANAGING IT WORKERS

Katarzyna Łubieńska[1], Jacek Woźniak[2]

University for Finance and Management, Pawia 55, 01-030 Warsaw, Poland
E-mails: [1]k.lubienska@aster.pl (corresponding author); [2]jkwozniak@tlen.pl

Received 07 October 2011; accepted 25 February 2012

Abstract. The text analyses the issue of motivation in software engineers. It bases on the experiences of one of the authors, and on preliminary results of qualitative and quantitative research gathered from 300 software engineers working in the IT financial service sector in Poland. It reviews main approaches to software specialist motivation found in the scientific and practical literature. It critiques some approaches to motivation of software specialists for the long-term consequences of using the motivators that they focus on.

It shows that research based on P. Glen's model (2003a) analyses only hygienic factors (as described by Herzberg) that can hinder the state of *flow* (Csikszentmihalyi 1975), which is characteristic of internally motivated software specialists. There are no analyses which would show how to secure long-term hygienic factors in the management of IT specialist motivation. Recommendations on long-term hygienic factors form a significant part of the text.

Keywords: motivation, IT worker, software specialist/engineer, geek, software development, knowledge work, management of IT team.

Reference to this paper should be made as follows: Lubieńska, K.; Woźniak, J. 2012. Managing IT workers, *Business, Management and Education* 10(1): 77–90. http://dx.doi.org/10.3846/bme.2012.07

JEL classification: L63, L86, M12, M15, M54.

1. Introduction

Over the past 50 years, IT technologies have permeated every sphere of our lives. Despite the fact that computer hardware is becoming cheaper by the year, outlays for the development of IT systems and technology are constantly on the increase. A persistent problem, however, is the effectiveness with which resources are used. Only ⅓ of IT projects are successfully concluded, and this rate has hardly changed since 1999, when only every fifth project terminated successfully. Even worse, over ¼ of the resources assigned for the development of information systems each year is irretrievably lost (ongoing projects are abandoned) and does not bring investors any profit; this state has remained almost without change over the years (Dominguez 2009).

The causes for this are sought in the work of IT specialists rather than in the quality of computer technologies. Motivation is also frequently considered to be the single most

important factor affecting IT specialists' productivity (Beecham *et al.* 2008: 861). The goal of this work is to briefly present the authors' perspective on efficient ways of managing teams of software specialists. We focus on motivating, as it is in this area that we consider that both theoreticians and practitioners have a mistaken perspective on where the core of the problem may lie; in fact their perspective may even be harmful to the work of software specialists.

Our analyses are grounded empirically in over twenty years of experience in management of software specialist teams on the part of one of the authors; and on preliminary results of research conducted since the end of 2010 into the motivation of software specialists working in the sector of financial services.

The article is organised as follows. A brief introduction describes the specifics of the profession of an IT specialist, providing a specific example of a knowledge worker. Next, two main trends in research on IT specialist motivation described in several recent reviews of this area of management (Beecham *et al.* 2008; Sharp *et al.* 2009) are presented. We then differentiate between two different approaches to motivating software specialists, which are the effect of two kinds of practical analyses. The first kind are analyses of IT specialists as a specific group of workers that differ from other employees (Glen 2003a, 2003b; Pfleging, Zetlin 2006); the second – research on motivation in open source type program development (Roberts *et al.* 2006). The fourth section presents a critique of these approaches, and shows their limitations and possible consequences. The final section presents our model, which attempts to combine the strong points of the approaches described, avoiding at the same time their weaknesses.

2. The IT specialist as a knowledge worker

Professionalisation – the development of professional cultures along the lines of the traditional free professions – is a trend frequently described in the literature (e.g. Jemielniak, Koźmiński 2008; Postuła 2010; Sikorski 1997; Woźniak 2010a, 2010b). Usually, emphasis is placed on three areas of differences between traditional employees, and knowledge workers whose jobs are evolving towards professionalisation:

1. Specific character of tasks. Defining and resolving a client's problems involves tasks rich in information processing, with non-standard criteria used to make decisions (Drucker 1999; Davenport 2007; Lillrank 2003). Some authors additionally draw attention to the role of information processing and communication, or emphasise the role of information-processing tools necessary for tasks defined in this way (Nogalski, Surowski 2008). The use of theoretical knowledge and many years of experience as a basis for solving the client's problems are always stressed (Postuła 2010; Woźniak 2010a).
2. Specific character of relationships between employers and knowledge workers. The latter do not link their future with the company as do traditional workers (Postuła 2010), and are even referred to as "freelance hirelings" (Koźmiński 2005). They are focused on their own career, its development and their place in the professional community, rather than on success within their employing company.

3. Specific attitude towards their work, to which they apply aesthetic criteria and which is proof of their individual value. They realise their ambitions at work through the development of excellent products, and this excellence obscures other functions of the product. As an effect of this attitude, they treat tasks as exciting challenges (Czarnkowska 2010; Jemielniak 2008a; Peters 2002; Woźniak 2010b). Professionalists constantly perfect their professional knowledge and skills, as this is the only way they can remain valued within their profession. Their value in the eyes of colleagues and exacting clients is dependent on these skills and talents (Postuła 2010; Rogoziński 1999).

Software developers are often referred to as archetypal knowledge workers (Czarnkowska 2010; Davenport 2007; Jemielniak 2008a; Nordenflycht 2010; Reich 1991; Scarborough 1999). However, there is a significant differentiation in competencies within this group, which is rarely accounted for (Jemielniak 2008a; Postuła 2010). Changes in this profession, brought about by computer companies becoming service-oriented and by offshoring of production and IT services, cause some to question whether software specialists are still to be considered typical knowledge workers (Marks, Scholarios 2007), i.e., "white collar" workers with high prestige, high income and autonomy in their work (Jemielniak 2008a).

Works based on research (Beecham et al. 2008; Sharp *et al.* 2009) as well as practical analyses of IT specialist motivation (Glen 2003a, 2003b) tend to approach most IT assignments as if they were creative and non-standardisable, requiring considerable autonomy on the part of those involved. This suggests that analyses based on the concept of knowledge workers are still useful for research into this profession (Enns *et al.* 2006). When speaking of IT specialists or software engineers in this article, we refer to people who create and maintain the functionality of IT systems and services. They usually have a university degree (in Poland, usually in IT sciences) (Czarnkowska 2010; LW; Postuła 2010), and usually work in teams in a project structure (which is variously placed in the hierarchic-functional structures of the organisation). They mainly communicate with other groups of employees via analysts and project managers (and sometimes managers from the hierarchic-functional structure) (LW; Postuła 2010). An IT specialist is required to solve complex problems – including problems of a technical nature – creatively, with the use of extremely diverse and rapidly evolving IT tools (Glen 2003a). Developing or modifying the typical IT solution requires the software specialist to cooperate with IT specialists from different fields and different countries, and with other specialists who frequently represent very specialised (mainly technological) fields (LW).

3. Two models of software specialist motivation

Frequently met within management literature is the thesis that software specialists are motivated above all internally, i.e., by the nature of the task itself (Beecham *et al.* 2008; Glen 2003a; Göran, Hanse 2011; Sharp *et al.* 2009). Maintaining high motivation levels requires a sense of autonomy while performing the job, which among others means that

the function of feedback from superiors is informational and not controlling (Göran, Hanse 2011). External rewards (such as remuneration or bonuses) may reinforce this kind of motivation only in the case of uninteresting tasks (Gagne, Deci 2005). The low effectiveness of using monetary rewards as motivators in the case of knowledge-based work is frequently emphasised.

Detailed research has shown that software specialists are strongly oriented towards end-product excellence, or in other words "a job well done", with remuneration serving no more than a hygienic function (Wallgren, Hanse 2010). They are strongly affected by interactive motivation (Alvesson 2004), which is related to social factors (group norms, reciprocity values and a sense of identity). Their readiness to work long hours may be interpreted sociologically, as the result not only of a passion for the task, but also of the norms prevalent in their social environment (Jemielniak 2008a; Postuła 2010). Specifically, software specialists are prone to succumb to organisational traditions defining how long work lasts (60 and not 80 hours a week or vice versa). Internally motivated software specialists are ready to work very long hours for long stretches, taking only short breaks for pizza or a candy bar (or more often than not, they work and eat at the same time) (Glen 2003a; LW).

A second model very frequently used for analysing motivation in empirical research (Beecham 2008: 869) is Hackman and Oldham's Job Characteristic Theory. According to this model, high job involvement, work satisfaction, internal motivation and positive behavioural effects are determined by three psychological states in the worker. These in turn can be affected by specific task structuralisation and working conditions that the Job Characteristic Theory describes (Woźniak 2010a). Among five significant factors in this structure, three determine the feeling that work is meaningful (which in itself is one of these three key psychological states): diversity of skills necessary for performance, character of the task as a whole, and importance of the task. The remaining two factors – a sense of autonomy (determining the second psychological state, i.e., responsibility for work outcome) and ongoing feedback concerning task progression (determining the third psychological state, i.e., a sense of being up to date with results) – determine an appropriate context for work (Woźniak 2010a).

Both these five factors describing work context and the three psychological states are easily operationalised. As a result, the theory forms a good point of departure for empirical analyses and possible conclusions for application.

In practice, however, these factors are rarely applied for motivating software specialists in large projects (LW). Access to ongoing results in IT projects is frequently impossible due to their sheer size and to delays in the implementation of final solutions. Sense of autonomy, comprehended as the freedom to think, is obvious in every kind of knowledge-based work. Attempts to modify the behaviour of IT specialists are therefore rarely undertaken. Project managers have little influence over work context; on the one hand because of the specifics of the project, and on the other because of a strong software specialist subculture (Jemielniak 2008a; Postuła 2010).

Naturally, the software specialist needs to feel that his/her work is meaningful. However, it is debatable whether meaningfulness can be achieved through structuralising tasks into mini-projects which make use of multiple skills, or through emphasizing the importance of these mini-projects. We may rather assume that most IT problems and the work environment in IT teams are of the kind described by Job Characteristic Theory (LW), so conclusions are practically of no significance from a management point of view. In the case of normal IT tasks they state: "do not interfere by bad management". In the case of tasks which are boring for IT specialists (i.e., do not depend on solving IT problems but involve documentation or codifying project conclusions), they provide no insightful advice at all.

4. Directives for software specialist motivation formulated by practitioners and analysts of IT project practice

P. Glen's (2003a, 2003b) term *geek*, a slang term for software specialists, has made a career because it refers especially aptly to this group's sense of professional identity. Irrespective of the work they do and their professional competence, software specialists have a sense of kinship with others who program computers, or more precisely – those who solve problems related to IT systems (Czarnkowska 2010; Postuła 2010). The word *geek* also denotes a person who is totally engrossed in his/her field of interest, at the expense of other areas of life (social skills, personal looks, social status). Software specialists were often outstanding students at school and at the same time outsiders, passionately absorbed by their hobby (Glen 2003a), and now in their professional lives, they are easily absorbed by long hours of work into solving a specific kind of problems.

Software specialists link knowledge workers' focus on developing their professional expertise, with a love of the technology with which they work. Rapidly changing technologies force them to learn new tools and fundamentally restructure customary practices, and hence to be on the constant lookout for new trends. Striving for successive diplomas, evidence that they have mastered numerous IT instruments, has its sources in the need to constantly be abreast of new developments in IT. Another reason is their love for new gadgets, which alongside letting them remain among the initiated few, also allow them to construct the best of technologically possible solutions. Software specialists' loyalty to technology, as described by Glen (2003b: 20) refers to their striving to deliver the best technical solutions that can create using the given technology, but also to a deep respect for the people who create such solutions.

Glen (2003a) postulates that managers should focus on the comfort of work of their IT teams. He suggests that small teams should be isolated from anything that could disturb their long hours of work; such an arrangement also builds a sense of competitiveness vis a vis the team's rivals. Isolation at work is not the equivalent of isolation from the organisation's goals: "one of the simple things that most of the motivational gurus get right is goal setting ... in geekwork, the best way to set a goal is to define a project to address it" (Glen 2003a: 111). With the help of standard management solutions – communicating

significance, selection of persons interested in the project, balanced access to resources (not too many and not too few) and building competitiveness – a focus on the goals of the project is achieved, and at the same time a sense of meaningfulness of the assignment is created.

Once focused on IT issues, the team will spontaneously apply itself to the job at hand for many hours without a break; if tasks are well allocated and an information flow is well structured, they will work towards a common goal. The manager's job is to motivate his/her team in the most traditional (though not classic) way – show appreciation for long hours of commitment by bringing them pizzas and free drinks. Acknowledging their work by satisfying physical needs so they do not need to interrupt it prolongs working hours and helps maximising the effort.

Glen's works evoked a strong response among IT managers, and showed the weakness of his proposal – software specialist isolation intensifies the "geek gap", the term B. Pfleging and M. Zetlin (2006) gave to the difference in how geeks and a business understand each other.

A second significant trend in research on the motivation of IT specialists is the analysis of communities developing open source programs (Roberts, Hann, Slaughter 2006). Software specialists active in the open source field develop whole systems or small and targeted solutions, which they make available free of charge to other IT workers in the internet community. These analyses differentiate between motives related to producing solutions for one's own projects (a specific kind of internal motivation), and an internalised external motivation related to social status in the IT community, as Roberts, Hann, Slaughter (2006) term aspiring to a higher status in the informal community of IT programmers. Additional motivation is gambling on the chance of a remunerated career within the community, for which one worked as a volunteer. Analyses of how these communities function have described not only relationships between internal and external motivation, but also the different effects of each of these categories for involvement in work (measured by scale of input into code development, and how the code evaluated).

All the instruments of external motivation – such as being given higher status or remuneration for writing a code – are correlated with an increased desire to participate in the project, but are not related to any increase in internal motivation (Roberts *et al.* 2006: 996), and even decrease interest in solving the specific, external problem (Roberts *et al*. 2006: 995). Rewards in the form of status in the community, considered to be very significant, strengthen bonds and increase involvement in future work. However, they do not focus involvement on any specific problems, nor do they modify the readiness to be interested in work itself.

We can take this result to indicate that social recognition does not positively modify intensity of internal motivation or interest in solving a specific technical or algorithmic problem. Appreciation within their community strengthens spontaneous activity, but does not give it a better direction. The authors of the study propose that feedback

accompanying appreciation should describe the elements of the significant input (Roberts *et al.* 2006: 996). This meets management's needs for giving specific direction to the motivation being reinforced.

Glen (2003a) indicates proven methods for motivating employees who spend a lot of their time at work, and describes the favourite toys of software specialists, which let them relax in their own company (table football, table tennis, etc.). Open source programming researchers draw attention to the fact that project stability requires using the activity of people who improve programs to solve their own problems, however this activity needs to be focused on specific targets. The core of the open source team has to be focused on solving problems significant for developing a given program.

In our opinion, the theoretical and practical motivational forces mentioned above do not address several issues specific to the management of software specialists:

1. Software specialists have a strong internal motivation to perfect their solutions and once in the state of *flow* (Csikszentmihalyi 1975) (long-term effort with breaks only for physiological purposes), do not require strong stimuli.
2. Software specialists are most creative in the first years of their professional careers, and immature personalities have a significant input in this environment.
3. Software specialists are very concerned about the success of their project and all factors which increase the chances of its success are motivating.

The IT manager's task is not only to maintain a state of inner motivation with the use of appropriate hygienic factors and through focusing work on areas that are of fundamental importance for the success of the project. The manager also needs to handle involvement, to protect motivated IT specialists against its negative consequences for themselves. This is the only way the manager can stabilise the operation of the team as a whole.

5. Hygienic factors in the motivation of IT specialists, or is essential changes to the IT specialist motivation theory

As open-source researchers emphasise, software specialists get involved in projects for many different reasons (Roberts *et al.* 2006: 985). However, at the start of the job, one motivation dominates – that of solving a technical or algorithmic problem.

The ease with which software specialists enter the state of flow has been frequently stressed (Glen 2003a, 2003b). The sources of this, however, have been sought in many different areas, e.g., socialisation at school, emotional aversion to external contacts, or focus on problem-solving and the love of competitive games (especially those where you compete against yourself). For software specialists, the essence of their work lies in solving an IT problem, while it is the role of people responsible for defining customer requirements to provide information about these needs in the terms of the framework and limitations of the project. Software specialists get drawn into creating and

developing an IT system not because solving the problem will be of use to the end user, but because it is a problem. As the old saying goes "mountains are there to be climbed".

The ongoing discussion on whether software specialists have stronger achievement needs than other groups of workers (Beecham *et al*. 2008: 861) cites disparate research results), ignores a specific kind of achievement need peculiar to IT specialists. Solving the technical or algorithmic problem and achieving success in this area (Beecham *et al.* 2008: 861) is a measure of the value of a person as a professional, both for him/herself, as for his/her environment (Glen 2003a). Doing a job well means finding an aesthetically beautiful ("elegant") solution to the problem, rather than meeting the end user's needs at the lowest possible cost. The task's meaningfulness has no relationship to its identity or importance, but to the ability to create excellent solutions, that meet parameters established in advance. If the project manager does not define the functionality to be achieved, and does not limit the freedom with which it can be redefined, the task will be modified to meet an aesthetic whole, regardless of the actual necessity of the additional functionality.

If one compares the work of software specialists to that of an artists (Glen 2003; Jemielniak 2008b), it becomes obvious why the need for external constraints, which enable them to work effectively, i.e. use their excessive internal motivation to implement the project within its predetermined limits. Like artists who, if unrestrained, never end their "endlessly imperfect" endeavour, so will an IT specialist constantly succumb to the wish to do "something really cool", and not just solve the client's problem. Just as artists in moments of doubt need external incentives to guide them back to the desire to work, so IT specialists in times of depletion need external stimuli supporting their motivation. In this sense, Glen is right – when software specialists are working, leave them alone and facilitate their work (e.g. by serving simple meals at the computer). In the flow phase, the software specialist does not need to be motivated, but care must be taken that the flow does threaten his/her existence.

We need to differentiate between three different motivational problems. The first concerns the actual motivation. How do you help "artists" experiencing a minor "creative crisis"? They cannot start working, suffer from a writer's block, or no longer understand what they are doing.

The second concerns hygienic factors, which help get on with the work when once immersed in the flow – when involvement and internal motivation are high. P. Glen discusses these kinds of hygienic factors in detail.

The third concerns long-lasting hygienic factors, creating working conditions that will allow the software specialist perform in his professional role in a long-term manner. IT specialists are exposed to multiple stress factors: long working hours, the need to engage in the race for technical competence, strong meritocratic competition within their professional group, and last but not least – focusing on success (solving the problem is proof of my value).

Building identity on professional competence, and the constant need for its confirmation through solving problems, hinders the development of support-style relationships based on family life. The readiness to give oneself up to one's work, or in other words excessive involvement, results not only in professional burnout (Nahrgang *et al.* 2010), but in an exhaustion of deep resources, not renewable with the use of standard methods of dealing with stress (MacEachen *et al.* 2008).

Research into motivating software specialists must include each of these three areas; otherwise it deals not with motivating, but with ways of "squeezing them dry".

6. Theses concerning management of software specialists

Among the external and internal motivators, Sharp *et al.* (2009: 223) distinguish in their review of the literature that the role of IT project management methodologies is not very clear. Methodologies include how to break the project up into coherent modules (sub-projects), which an employee or a group of employees can undertake, with a timetable for their implementation. Even when building participation, trust, autonomy and empowerment (to mention motivators related to formulating the goals of the sub-project) (Sharp *et al.* 2009: 223) team managers cannot forget that almost every IT specialist seeks to develop excellent solutions and overestimates the speed with which the problem can be solved. Setting objectives must take into account both conditions external to the team (e.g., required time limits), as those within the team, including the need to limit the software specialist's need to pursue excellence. Management by objectives requires analysts and project managers to develop schedules in a participatory manner, adapting them to external conditions and constraints in such a way that the schedules and products agreed upon are also acceptable from the professional perspective of those implementing them.

Thesis 0: Define the project objectives.

Set criteria for acknowledgement of milestones, specify in detail the material products at every stage (e.g., "a functioning IT system" is not a product; "results of a specific test" is a product).

Thesis 1: Agree upon a methodology for managing the project with all participants, and adhere to it.

Set project milestones as required by the client, but in such way that all members of the project team would also consider them implementable. If the deadlines and scope of the project exceed pessimistic estimates of what the team is capable of, adjust the scope of the project to deadlines (omit chosen functionalities which do not have to be implemented in the first stage or can be achieved by semi-automated tools, limit items that do not belong to the main process, etc.).

Thesis 2: Be flexible in reformulating expectations if they turn out to be unrealistic.

The software engineer is frequently unsure whether the given problem is at all solvable (within the adopted framework). Communication channels are usually adjusted to

the project methodology, but as team leader you must provide the means for quick and open informal communication, both technical as supportive in the event of difficulties. Proper working conditions in the case of IT professionals should ensure such relationships in which difficulties in completing tasks can be articulated in an open manner, as troubles if concealed can be a threat to the whole project.

Thesis 3: Tend to ongoing work hygiene; avoid demotivating your subordinates.

The software engineer does not need appreciation of his/her person – he/she needs to be judged on his merits, that of the product he/she creates. Such "externalised internal rewards" are standard hygienic factors in his work. Without them, internal motivation – the presence of which is treated as a normal state of affairs – does not function. Essential work factors are almost always secured in the case of software engineers, and Hackman and Oldham are correct in that not meeting them would be a source of demotivation. The authors of a review article (Sharp *et al.* 2009: 230) rightly emphasise that the "motivation (of IT workers) is heavily dependent on the context ... but the literature does not shed much light on how this influence works". Our article also lays stress on the hygienic effect of the context. We consider that in a well-organised project team, software engineers are internally motivated.

Thesis 4: Care for the hygiene of the software specialist's day to day life.

We wrote about the destructive role of permanent stress in the previous chapter. We know that the effectiveness of a mortally tired software specialist must be lower than that of the same software specialist who is rested. And this does not refer to the number of lines of code he/she writes, but to the quality of the end-product for solving the problem.

Maximizing problem solving efficiency requires the manager to take care that problems are tackled by IT specialists who are fresh and alert. In knowledge work, advances come in irregular spurts, as problem solving is the effect of making many breakthroughs, and not just of increment resulting from hours of assiduous work. Often, the results of a few good hours' work are more significant than of hundreds of hours of persistence at the desk.

Thesis 5: Take responsibility for the security of the project as a whole.

The role of the manager is to ensure the viability of the project, even if one of the software specialists drops out. Correcting others' work is often so difficult that it is easier to start anew. The manager is responsible for pairing people up in sub-teams so that everything that is worked on is understood by more than one person.

Thesis 6: Redefining project objectives should be a normal occurrence in the everyday life of the project.

Redefining project objectives or their components as a result of changes in the environment ("business changed its mind") should be an everyday matter in the life of the project, and should be an integral part of the project management methodology.

Change should by its very definition effect in a redefinition of timetables and project costs (regardless of the significance of the change). Software specialists we interviewed claim that changes which take place while the project lasts (see LW) are the most common cause of project failure (just after "unrealistic demands and deadlines" set at the start of the project). This means that in many settings, changes are made in an uncontrolled manner.

Thesis 7: Manage risk.

The business in which you work teaches you humility, and this is the case also for the most arrogant of software engineers. Name the risks to your project, check in an ongoing manner which risks begin to materialise, and be prepared to implement contingency plans. Notify both clients, as those who work on the project of the risks, especially those which lie within their sphere of competence.

7. Conclusions

The article analyses the main trends in research into software engineer motivation. We have noted that, although IT specialists are usually treated as prototypical knowledge workers, the changes that have taken place in this profession have made them increasingly dissimilar to professionals working on their own.

An analysis of reviews of research into the motivation of software specialists shows that research provides no management guidelines which go beyond good practice in the sector. Basing on two main trends in research on the management of IT team motivation, we hypothesised that researchers describe factors which are hygienic in nature, rather than motivational, as these factors create an environment in which software specialists can be subjected without interruption to the functioning of their own internal motivation. An analysis is made of the long-term consequences of focusing inordinately on solving a problem, and eight theses are formulated concerning software team management, which will be of help in implementing projects.

Our deliberations focus on only a chosen aspect of the problems of IT specialist management, i.e. on motivation. We have not broached the issues of communication between software specialists and their clients; of maintaining large IT systems in current conditions of IT team instability; of the specific counterculture they create; of disloyal software specialists; of managing multicultural and multinational projects; and many others. We only briefly mention the issue of leadership in IT teams.

The results of our deliberations are limited in at least two ways. Firstly, the empirical basis for our analyses is the practical experience of one of the authors, and the preliminary results of a survey carried out on a group of software specialists working in the financial services sector in Poland. Our analyses are based on projects that are of average difficulty and low market pressure.

Secondly, we have not made any intercultural comparisons; indeed we have not even taken into consideration differences in level of education or the service-orientation of IT workers.

Research considering these variables are a natural next step to make.

Another promising area of research would be on the negative effects of over-motivating software engineers. If our thesis concerning the over motivating of IT specialists is true, we should expect that the negative consequences of over motivation are partly responsible for the failure of IT projects. We need to find examples of project failures caused by over motivation to broaden the standards of analysing projects.

References

Alvesson, M. 2004. *Knowledge work and knowledge intensive firms*. Oxford.

Beecham, S.; Baddoo, N.; Hall, T.; Robinson, H.; Sharp, H. 2008. Motivation in Software Engineering: A systematic literature review, *Information and Software Technology* 50: 860–878. http://dx.doi.org/10.1016/j.infsof.2007.09.004

Csikszentmihalyi, M. 1975. *Beyond Boredom and Anxiety: Experiencing Flow in Work and Play*. San Francisco.

Czarkowska, L. D. 2010. *Nowy profesjonalizm: Kultura profesjonalna informatyków – antropologia organizacji*. Warszawa.

Davenport, T. H. 2007. *Zarządzanie pracownikami wiedzy*. Polish translation of Wolters Kluwer, Kraków.

Dominguez, J. 2009. *The Curious Case of the CHAOS Report 2009* [online], [cited 1-06-2011]. Available from Internet: http://www.projectsmart.co.uk/the-curious-case-of-the-chaos-report-2009.html

Glen, P. 2003a. *Leading Geeks. How to Manage and Lead People Who Deliver Technology*. Jossey-Bass, San Francisco.

Glen, P. 2003b. Leading Technical People, *Leader to Leader* 4: 19–24. http://dx.doi.org/10.1002/ltl.46

Drucker, P. F. 1999. *Społeczeństwo postkapitalistyczne*. Warszawa.

Göran, L.; Hanse, J. J. 2011. The Motivation of information Technology Consultants: The Struggle With Social Dimensions and Identity, *Human Factors and Ergonomics in Manufacturing & Service Industries* 21(6): 555–571.

Gagne, M.; Deci, E. L. 2005. Self-determination theory and work motivation, *Journal of Organizational Behavior* 26(4); 331–362.

Enns, H. G.; Ferratt, T. W.; Prasad, J. 2006. Beyond stereotypes of IT professionals Implications for IT HR practices, *Communications of the ACM* 49(4): 105–110. http://dx.doi.org/10.1145/1121949.1121956

Jemielniak, D.; Koźmiński, A. 2008. *Zarządzanie wiedzą. Podręcznik akademicki*. Warszawa.

Jemielniak, D. 2008a. *Praca oparta na wiedzy.* Warszawa.

Jemielniak, D. 2008b. Software engineer or an artist? Programmers identity choices, *Tamara Journal* 7(7.1): 20–36.

Koźmiński, A. 2005. *Zarządzanie w warunkach niepewności*. Warszawa.

Lillrank, P. 2003. The Quality of Standard, Routine and Nonroutine Processes, *Organization Studies* 24(2): 215–233. http://dx.doi.org/10.1177/0170840603024002344

Łubieńska, K.; Woźniak, J. 2012 (in preparation). *Managing IT workers*. Warszawa (cited as LW).

MacEachen, E.; Polzer, J.; Clarke, J. 2008. 'You are free to set your own hours': Governing worker productivity and health trough flexibility and resilience, *Social Science & Medicine*, 66 p.

Marks, A.; Scholarios, D. 2007. Revisiting technical workers: professional and organizational identities in the software industry, *New Technology, Work and Employment* 22(2): 98–117. http://dx.doi.org/10.1111/j.1468-005X.2007.00193.x

Nahrgang, J. D.; Morgeson, F. P.; Hofmann, D. 2010. Safety at Work: A Meta-Analytic Investigation of the Link Between Job Demands, Job Resources, Burnout, Engagement and Safety Outcomes, *Journal of Applied Psychology* 95(1): 71–94.

Nogalski, B.; Surowski, B. 2008. Pracownicy wiedzy. Problemy i dylematy badania pracy opartej na wiedzy, in Listwan, T.; Witkowski, S. A. (Eds.). *Kompetencje a sukces zarządzania organizacją.* Warszawa.

Nordenflycht, A. 2010. What is a Professional Service Firm?, *Academy of Management Review* 35/1.

Peters, T. 2002. *Profesjonalna firma usługowa*. Warszawa.

Pfleging, B.; Zetlin, M. 2006.*The Geek Gap*. New York.

Postuła, A. 2010. *Informatycy i organizacje*. Warszawa.

Pyöriä, P. 2005. The concept of knowledge work revisited, *Journal of Knowledge Management* 9(3): 116–127. http://dx.doi.org/10.1108/13673270510602818

Reich, R. 1991. *The work of nations*. New York.

Roberts, J. A.; Hann, I.-H.; Slaughter, S. A. 2006. Understanding the Motivations, Participation, and Performance of Open Source Software Developers: A Longitudinal Study of the Apache Projects, *Management Science* 52(7): 984–999. http://dx.doi.org/10.1287/mnsc.1060.0554

Rogoziński, K. 2003. Innowacyjność i nowa taksonomia usług w procesach globalizacji, *Working papers Katedry Usług AE Poznań* 2: 1–20.

Scarborough, H.1999. Knowledge as Work: Conflicts in the Management of Knowledge Workers, *Technology Analysis & Strategic Management* 11(1): 5–16. http://dx.doi.org/10.1080/095373299107546

Sharp, H.; Baddoo, N.; Beecham, S.; Hall, T.; Robinson, H. 2009. Models of motivation in software engineering, *Information and Software Technology* 51: 219–233. http://dx.doi.org/10.1016/j.infsof.2008.05.009

Sikorski, C. 1997. *Profesjonalizm.* Warszawa.

Wallgren, L. G.; Hanse, J. J. 2007. Job characteristics, motivator and stress among information technology consultant, *International Journal of Industrial Ergonomics* 37(1): 51–59. http://dx.doi.org/10.1016/j.ergon.2006.10.005

Woźniak, J. 2010a. *Systemy motywacyjne we współczesnej firmie*. Olsztyn.

Woźniak, J. 2010b. Systemy motywacyjne dla pracowników wiedzy różnych typów, in Kulawczuk, P.; Poszewiecki, A. (Eds.). *Behawioralne determinanty rozwoju przedsiębiorczości w Polsce.* Gdańsk, 224–239.

IT SEKTORIAUS DARBUOTOJŲ VADYBA

K. Łubieńska, J. Woźniak

Santrauka

Šiame straipsnyje analizuojama programinės įrangos inžinierių motyvacija. Jame remiamasi vieno iš autorių patirtimi ir preliminariais kokybinių ir kiekybinių tyrimų rezultatais, surinktais iš 300 programinės įrangos inžinierių, dirbančių Lenkijos IT finansinių paslaugų sektoriuje. Taip pat apžvelgiami mokslinėje ir praktinėje literatūroje rasti pagrindiniai metodai, taikomi specialistų motyvacijai. Programinės įrangos specialistų motyvacijai taikomi metodai kritikuojami už tai, kad buvo žiūrima pro pirštus į ilgalaikes motyvacijos veiksnių, kuriems jie skiria daugiausia dėmesio, naudojimo pasekmes.

Atskleidžiama, kad tyrimuose, grindžiamuose P. Gleno (2003a) modeliu, analizuojami tik higieniniai veiksniai (kaip rašo Herzbergas), o tai gali trukdyti *progresui* (Csikszentmihalyi 1975), kuris būdingas specialistams su vidine motyvacija. Nors straipsnyje neanalizuojama, kaip IT specialistų motyvacijos valdymo srityje reikėtų išsaugoti higieninius veiksnius, gana didelę jo dalį užima su ilgalaikiais higieniniais veiksniais susijusios rekomendacijos.

Reikšminiai žodžiai: motyvacija, IT darbuotojas, programinės įrangos specialistas (inžinierius), fanatikas, programinės įrangos kūrimas, IT komandos valdymas.

Katarzyna ŁUBIEŃSKA. MA in computer sciences, MBA; was an IT director and IT team leader in many financial companies. Her scientific interests focus on IT management.

Jacek WOŹNIAK, Dr Habil in management, works as a professor at the University of Finance and Management in Warsaw. His research is focused on training theory and management of professional service companies.