Editor: Forrest Shull ■ Fraunhofer Center for Experimental Software Engineering, Maryland ■ fshull@fc-md.umd.edu

What Do We Know about Developer Motivation?

Tracy Hall, Helen Sharp, Sarah Beecham, Nathan Baddoo, and Hugh Robinson

Software developers do better work and stay with one company if they're motivated—indeed, evidence exists that developer motivation affects project productivity,[1] software quality,[2] and a project's overall success.[3] By improving how they manage software developers' motivations, companies could significantly improve their ability to deliver good-quality software systems.

To explore what we know about what motivates developers, we systematically analyzed 92 studies on software developer motivation published between 1980 and 2006.[4] Our analysis shows that motivation influences many important aspects of project success. Timely project delivery, productivity, and adherence to budgets, as well as increases in staff retention and reduced absenteeism, are all related to motivation. Our analysis also shows that controlling these factors isn't straightforward.

Web Extra

Experimental method and data details at www.computer.org/portal/pages/software/content/WebExtra.html.

Study selection

To identify which studies to analyze, we searched eight popular publication portals as well as key software engineering conference proceedings, journals, and authors. We searched specifically for studies on motivation in a software engineering context, identifying more than 2,000 potential papers and rejecting approximately 1,500 of these (on the basis of titles and abstracts) as irrelevant to our focus. We read the remaining 519 papers in full to establish our final list. The 92 papers we chose were originally published in the *ACM SIGCPR Computer Personnel*, various IEEE proceedings, the *Communications of the ACM*, *MIS Quarterly*, and the *Journal of Systems and Software*. The studies varied hugely in how researchers measured motivation, the context in which they studied it, and the methods they used.

Who are software engineers?

In 1980, Daniel Cougar and Robert Zawacki were the first to report on computing personnel's motivations, extending the well-established but generic Job Characteristics Theory[5] to this group.[6] They reported that computing personnel need growth and learning and enjoy a challenge but have low needs for socializing. This work has significantly influenced subsequent thinking on developers' motivations.

Software engineering has evolved in various ways since this early work. The context in which developers work is more complex than ever. In addition, it's increasingly obvious that although software developers might generally have similarities as a professional group, individual developers vary.

Our analysis throws more light on how and why this is. In particular, it suggests two dimensions of motivation unrecognized in earlier work: *controllers* and *moderators*. Controllers are internal character-

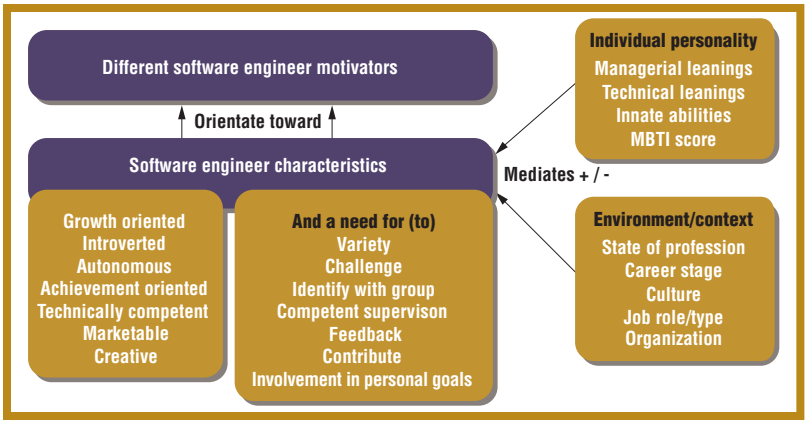

Figure 1. Motivating factors for software developers. Software engineers' personal characteristics are mediated by their personality and the environment in which they work. These personal characteristics orient software engineers toward responding to particular motivators.

0740-7459/08/$25.00 © 2008 IEEE

Table 1

General motivators for software developers

General motivators	Examples of study findings	No. of studies that report on motivation
Identify with the task	Clear goals, personal interest, knowing a task's purpose and how it fits with the whole, job satisfaction; producing an identifiable piece of quality work	20
Good management	Senior management support, team-building, good communication	16
Employee participation	Involvement in company, working with others	16
Career path	Opportunity for advancement, promotion prospects, career planning	15
Variety of work	Making good use of skills, being stretched	14
Sense of belonging	Supportive relationships	14
Rewards and incentives	Increased pay and benefits linked to performance	14
Recognition	Recognized for a high-quality, good job done based on objective criteria	12
Technically challenging work	Work isn't mundane and is technically challenging	11
Development needs addressed	Training opportunities exist to widen skills; opportunity to specialize	11
Feedback	Colleagues and managers provide feedback on work	10
Job security	Stable working environment and a basic level of job security	10
Autonomy	Freedom to carry out tasks, letting roles evolve	9
Work-life balance	Flexibility in work times, caring manager or employer, work location	7
Empowerment	Responsibility for the task is assigned to the (empowered) person	6
Appropriate working conditions	Working environment, good equipment, tools, physical space, quiet	6
Making a contribution/task significance	Degree to which the job has substantially affects others' lives or work	6
Trust	Trusting and respecting other people	4
Equity	People are treated and managed fairly	3
Sufficient resources	A basic level of resources exists for the job	2
Working in successful company	Financial stability	2

istics such as how introverted people are, their skill sets, and internal needs that control which motivators affect an individual developer.[7] Moderators are demographic considerations such as personality, career stage, role, and culture that moderate how strong each characteristic is for a given developer. For example, a young developer trying to buy a house or start a family will likely be more motivated by money and less by challenge, whereas a seasoned developer established and secure in his or her job will likely be more motivated by challenge and recognition for quality work. This developer will likely move to another workplace to gather more knowledge and experience if the work becomes mundane, whereas the young developer will likely stay or move for a higher salary. Figure 1 shows how sets of motivators are tailored to individual developers' characteristics, which are mediated by personality and environmental factors.[8] Our analysis suggests that before managers can focus on an effective set of motivators for an individual developer, they must consider contextual factors for that developer.

How we manage motivation

Having established that the way software developers respond to motivators varies according to context, let's consider their specific motivators. Tables 1 and 2 present the motivators reported in the 92 studies we analyzed. Table 1 shows general motivators relevant to software developers in their work; Table 2 shows motivators specific to software engineering work. In the first table, we can see that the most frequently reported motivator relates to work tasks. This suggests that to get the best work from developers, managers must carefully match tasks to individuals and comprehensively brief them on these tasks' context in terms of their purpose and the way individual tasks fit into the overall development. In the second table, we can see that software developers particularly respond to the challenging, dynamic, problem-solving aspects of software engineering tasks.

Table 1 also shows that good management and employee participation are often reported as general motivators. Also, many less-frequently reported motivators from Table 1 relate to good management. For example, providing feedback, recognition, autonomy, and resources are specific good management practices. This suggests that significant motivational benefits might arise from implementing good basic practices.

Many of the studies reported in the 92 papers don't interpret their findings in terms of existing motivation theory,[9] so their findings' wider implications aren't always clear. Classic motivation theory recognizes that some motivators are more likely than others to affect behavior. In particular, Frederick Herzberg defines the notion of *intrinsic* and *extrinsic* motivators.[10] Intrinsic motivators relate to the work itself

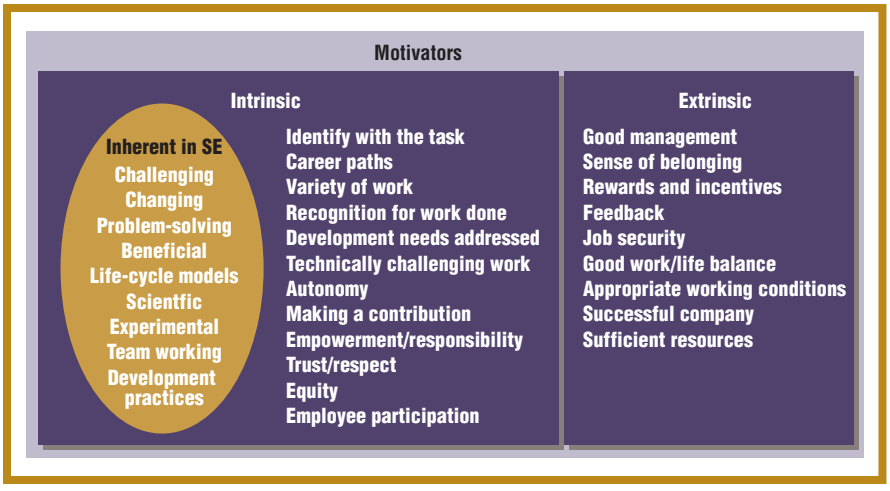

Figure 2. Intrinsic and extrinsic motivators for software developers. Software engineers' motivators are divided into those intrinsic to the work they do and extrinsic to that work. Also, intrinsic motivators are either specific to the work's software engineering aspect or related more generally to the work.

and primarily affect the behavior of people who, like developers, typically have high achievement needs. Extrinsic motivators are external to the work itself, such as working conditions and general good management—these have less influence on typical software developer behavior.

Figure 2 shows developers' intrinsic and extrinsic motivators. It suggests that the extrinsic nature of general good management practices means that these practices will likely have limited impact on behavior. However, the intrinsic nature of task-based issues means that managing developers' tasks effectively will have more long-term impact. In other words, managers must provide challenging problem-solving tasks, explicitly recognize quality work, and give developers autonomy to do their jobs. Managing these factors effectively will engage developers and excite them in their work.

Software engineering has evolved significantly in recent years. Radical new forms of working, including agile methods and globally distributed development, emphasize the importance of developers' human aspects. This software engineering evolution coupled with the increasing imperative to deliver successful software projects means that it's more important than ever that we manage developer motivation effectively. Our results are a first step toward being able to do so in today's complex and dynamic software engineering environments.

Acknowledgments

The UK's Engineering and Physical Science Research Council supported this research under grant EPSRC EP/D057272/1.

References

1. J. Procaccino and J.M. Verner, "What Do Software Practitioners Really Think about Project Success: An Exploratory Study," *J. Systems and Software*, vol. 78, no. 2, 2005, pp. 194–203.
2. B.W. Boehm, *Software Engineering Economics*, Prentice-Hall, 1981.
3. S.A. Frangos, "Motivated Humans for Reliable Software Products," *Microprocessors and Microsystems*, vol. 21, no. 10, 1997, pp. 605–610.
4. S. Beecham et al., "Motivation in Software Engineering: A Systematic Literature Review," to be published in *Information and Software Technology J.*, 2008; doi:10.1016/j.infsof.2007.09.004.
5. J.R. Hackman and G.R. Oldman, *Motivation through the Design of Work: Test of a Theory*, Academic Press, 1976.
6. D.J. Couger and R.A. Zawacki, *Motivating and Managing Computer Personnel*, John Wiley & Sons, 1980.
7. A. Maslow, *Motivation and Personality*, Harper & Row, 1954.
8. H. Sharp et al., "Models of Motivation in Software Engineering," to be published in *Information and Software Technology J.*, 2008; doi:10.1016/j.infsof.2008.05.009.
9. T. Hall et al., "A Systematic Review of Theory Use in Studies Investigating the Motivations of Software Engineers," to be published in *ACM Trans. Software Eng. and Methodology*, 2008; tosem.acm.org/Upcoming.php.
10. F. Herzberg, *The Motivation to Work*, 2nd ed., Chapman & Hall, 1959.

Tracy Hall is a reader at Brunel University. Contact her at tracy.hall@brunel.ac.uk.

Helen Sharp is a senior lecturer at the Open University. Contact her at h.c.sharp@open.ac.uk.

Sarah Beecham is a research fellow at Brunel University. Contact her at sarah.beecham@brunel.ac.uk.

Nathan Baddoo is a senior lecturer at the University of Hertfordshire. Contact him at n.baddoo@herts.ac.uk.

Hugh Robinson is a professor at the Open University. Contact him at h.m.robinson@open.ac.uk.

Table 2

Software-engineering-specific motivators

Software engineering motivators	Examples of study findings	No. of studies that report on motivation
Challenge	Software engineering is a challenging profession.	4
Change	Techniques and problems change constantly; work is dynamic.	4
Benefit	Developers create something to benefit others or enhance well-being	3
Problem solving	Developers understand and solve problems	3
Team work	Developers work in a team of other professionals rather than alone.	2
Science	Developers make observations, identify, describe, engineer, investigate and theorize, or explain a phenomenon.	2
Experiment	Developers try something new or use experimentation to gain experience.	2
Development practices	Developers use, for example, object-oriented, Extreme Programming, or prototyping practices.	2