Putting Human Aspects of Software Engineering in University Curricula

Orit Hazzan

People-related issues are central factors in determining the success of software projects\(^1 \text{-}^2\); indeed, many software projects fail in terms of meeting customer needs, on-time delivery, and budget.\(^3\) Since the importance of the human aspects of software engineering (SE) to the success of software projects has already been acknowledged, why do these aspects not receive more attention in the software industry in order to improve project results?

I suggest two barriers that block the attention that human aspects of SE deserve. These barriers originate in the academic status and positioning of SE and in the structure of SE programs, which, in many cases, are located in computer science (CS) departments or schools. The result of these two barriers is that CS and SE graduates, who in most cases eventually work in the software industry as software engineers, are unaware of the importance of human aspects. I also propose several suggestions regarding the professional development of software practitioners that address this gap between academia and industry.

Curricula Barriers

Software Engineering 2004 (SE2004), which presents curriculum guidelines for undergraduate SE degree programs, is part of the Computing Curricula Series developed by the Joint Task Force on Computing Curricula of the IEEE Computer Society and Association for Computing Machinery.\(^4\) The joint task force acknowledges the importance of the human aspects of SE, presenting the following as one of the core principles on which the curriculum was formulated:

SE2004 must include exposure to aspects of professional practice as an integral component of the undergraduate curriculum. The professional practice of software engineering encompasses a wide range of issues and activities, including problem solving, management, ethical and legal concerns, written and oral communication, working as part of a team, and remaining current in a rapidly changing discipline. (p. 14)

According to this volume, issues regarding human aspects of SE constitute 15 to 20 percent of the total number of hours in SE programs. However, these guidelines suggest only a recommended curriculum; SE programs can increase or decrease the number of hours devoted to any topic.

In addition, because of market forces, many CS graduates work as software engineers after graduation. These graduates do not have a general engineering education nor were they exposed to human aspects of SE, simply because they completed a scientific program rather than an engineering program. Still, their work requires engineering skills and an engineering perspec-
tive. Similarly, Peter Denning and Richard Riehle claim that “many of our software developers have been raised in a research tradition, not an engineering tradition.”

Research Barriers
In the second edition of The Mythical Man-Month,1 Brooks claimed this:

Most academic research on software engineering has concentrated on tools. I admire and covet sharp tools. Nevertheless, it is encouraging to see ongoing research efforts on the care, growing, and feeding of people, and on the dynamics of software management. (p. 276)

In the same spirit, 10 years later, Robert Glass, V. Ramesh and Iris Vessey indicated that SE research produces almost exclusively technical products.6 Specifically, they claimed, SE research focuses on topics related to systems and software concepts at technical levels of analysis by formulating processes, methods, and algorithms. Furthermore, according to Glass and his colleagues, most SE research studies (58 percent) use formulative approaches (for example, the formulation of an algorithm).

Why isn’t there more research on human aspects of SE?

I suggest that the fact that many SE programs are located in CS departments also helps determine the accepted research paradigm used to explore SE issues in general, and human aspects of SE in particular. Specifically, while most of the research methods employed in SE are quantitative (in my opinion, because SE programs are located within CS departments), qualitative research is more suitable for investigating many issues related to human aspects of SE.

Although the qualitative research approach is indeed used in SE,7 I suggest that its potential contribution in general, and to research on the human aspects of SE in particular, has not yet been fully exploited. The qualitative research approach is generally used to investigate situations in which people are involved and different kinds of processes take place—clearly, these are topics related to human aspects of SE. Indeed, qualitative data analysis can shed light on many aspects of such situations, and it may enable us to argue with a certain degree of generalization. However, the nature of quantitative research does not enable us to explore complex human-related situations in depth. By the same token, neither does the qualitative research approach enable us to present a full picture of complex situations. Qualitative methods, however, do let us highlight many angles of people-centered situations such as the ones that involve human aspects of SE.

Recently, we have witnessed an increase in awareness of the human aspects of SE. This attention emerged in the software industry, which suffers when human aspects of SE issues are insufficiently addressed. The emergence of agile software development is one example. The attention that the agile approach gives to human aspects of SE is reflected, for example, in the Agile Manifesto, whose first principle is “individuals and interactions over processes and tools.” I hope this increasing awareness in industry will reflect back on educational institutions, where future software engineers acquire their professional education.

In the meanwhile, software practitioners can boost their professional development by increasing their awareness of the human aspects of SE. This awareness, in turn, might improve their understanding of software development processes and, consequently, their ability to find better solutions to problems they encounter during development. To aid their developers, software organizations can increase the attention paid to human aspects of SE in several ways. First, they can develop professional programs that legitimize and elicit attention to human aspects of SE. Second, they can establish and encourage a culture in which human aspects of SE are cultivated, acknowledged, and considered. Finally, they can deliver the message about the importance of human aspects of SE to SE policy makers in academia and SE curriculum developers.

References

Software practitioners can boost their professional development by increasing their awareness of human aspects of software engineering.

Orit Hazzan is an associate professor in the Department of Education in Technology and Science at Technion—Israel Institute of Technology. Contact her at oritha@tx.technion.ac.il.