Is Emotion Relevant to Requirements Engineering?

I. Ramos[a] and D. M. Berry[b]

[a]Departamento de Sistemas de Informação, Universidade do Minho, Guimarães, Portugal; [b]School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada

This viewpoint argues that the introduction of most computer-based system to an organization transforms the organization and changes the work patterns of the system's users in the organization. These changes interact with the users' values and beliefs and trigger emotional responses which are sometimes directed against the software system and its proponents. A requirements engineer must be aware of these emotions.

1. A Bit of History[1]

Originally, computer-based systems (CBSs), actually just programs, were introduced merely to automate existing manual tasks for collecting, processing, or distributing information. A program was used for speed up, for error reduction, or for both in existing clerical tasks without changing the basic processes in which these tasks were carried out. If the original task produced a report on paper, the automated version produced the same report, so that the automated task could be carried out, albeit faster and with fewer errors, *in the original task's place*.

However, as observed by Lehman [2], the introduction of these CBSs began to affect the processes in which these automated tasks were embedded. Different and better processes were enabled by the automated tasks, even to the point of making the automated original tasks unnecessary, and even to the point of eliminating some peoples' jobs!

Correspondence and offprint requests to: Daniel M. Berry, School of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. Email: dberry@uwaterloo.ca

[1] Section 1 is reprinted with some minor changes from [1] by permission of the editor-in-chief of the *Journal of Information and Software Technology*.

New peripheral devices were introduced that allowed the computer to sense and control more than just data, e.g., to help operate production lines and to help fly aircraft. These devices allowed the introduction of CBSs to change more and more processes and to fundamentally alter the way things are done in many real-world, man-made systems not at all related to computing. For example, originally, commercial systems evolved in the presence of only paper and carbon paper. Thus, distributing multiple copies of a form containing all conceivably required information about a transaction became the way to distribute all needed information rapidly and in parallel to all involved in the transaction. Even so, these data were usually accurate only to the day or more. With the advent of computing, the production of the multiple copies was automated, with the basic workflow, and in particular the distribution of the copies, unaltered. Still, the data were usually accurate only to the day or more.

Later, as new high-resolution screens and networks became available, it was suddenly observed by creative software engineers that it was no longer necessary to print and distribute multiple paper copies of all possible needed information. Instead, it sufficed to put a network of computers, each with a high-resolution screen, on the desks of all persons involved in all transactions, and to allow each to directly access precisely the information he or she needed for his or her part of the transaction at the time he or she needed it. These data would be accurate to the moment. It became totally unnecessary to make any paper copy of the information. Fully automated enterprise resource planning (ERP) [3] had become possible.

On one hand, the operation of a transaction has improved. The information needed for the transaction is distributed or made available for access instantly to all involved in the transaction. A person involved in the transaction looks up only the information that he or she needs. Recall that a traditional form contains all information that might conceivably be needed in a transaction. With immediate electronic access, if it is known that a rarely needed item is readily accessible, it might be possible to forgo requesting this item during the filling of the electronic form, leaving it to be found if and when it is actually needed. Thus, the transaction is made more efficient.

On the other hand, there are some negative consequences. At the very least, because no paper copies are distributed, all those involved with the production, purchasing, distribution, and disposal of paper and printed reports find their responsibilities reduced and their jobs possibly eliminated altogether.

The organization has been transformed. The introduction of any CBS has the potential to transform any organization into which it is introduced. In general, the introduction of CBSs has become a driver for innovative work practices and new models of management and organization [4]. Furthermore, each employee that used to get his or her information from a particular place on one copy of all possible needed information now needs to work online on a computer with a windowing system with a program that allows querying for the needed information, but probably in a language derived from predicate calculus. One or more of these employees could rebel against this job change because he or she hates computers, hates mathematics or anything reeking of it, or refuses to learn any newfangled way to do something that he or she has done for years in a perfectly good simple way! “If it ain’t broke, don’t fix it!” says the employee. Clearly, emotional issues are coming into play in the use of CBSs and the organizational transformations (OTs) they breed.

Exacerbating these potential emotional problems is the fact that environments into which the CBSs are introduced are incredibly complex [5, 6]. Thus, it is next to impossible to predict the effect that the introduction of a new CBS will have on the organization and its users.

Let us consider the changing nature of requirements engineering (RE) for CBSs. Initially, RE needed to consider only the input–output behavior, i.e., the functionality, of programs. Certainly, nonfunctional issues, especially performance, were important too. If an automated task ended up taking too much time, space, or both, the task would not be effectively automated. As CBSs began to change the real-world systems in which they were embedded, RE needed to consider the effects of the changes, and in some cases, to predict and even alter likely effects. As CBSs began to transform organizations deploying them, RE needed to consider *how* the organization *should* be transformed, as opposed to just letting the organization be transformed any which way. RE needed to project organizational-level requirements onto the requirements for the CBSs that were to effect these transformations. As more and more technical issues in computing were solved, as computing technology became more stable, and as people began to be more educated about what computing technology could and could not do, emotional issues, which had been overshadowed by technical issues, percolated to the forefront. Finally, as these now more visible users’ emotions began to affect how well, and even if, the deployed CBSs would be used in the transformed organizations, RE needed more and more to consider emotional issues and to project them onto the requirements for the CBSs triggering the emotional responses.

This viewpoint attempts to describe the roles of emotions and of closely related values and beliefs in determining acceptance of deployed organization-transforming CBSs. It defines emotions and gives some examples of how they can impact the OT process. It reports published observations of CBS requirements affected by emotional issues. Finally, it considers whether the observed effects are just managerial problems.

The assumed domain for the CBSs in this viewpoint is any CBS for which there is a mandated or de facto RE process prior to building, buying, enhancing, or deploying the CBS. Thus, the domain includes contracted, bespoke CBSs; configurable, off-the-shelf CBSs; and in-house CBSs.

2. Emotions, OTs, and their Enabling CBSs

This section lays some background from the literature about emotions, OTs, and their enabling CBSs.

According to Damásio [7], there are three types of emotions, primary emotions, background emotions, and social emotions. Primary emotions include joy, sadness, fear, anger, surprise, and aversion. Background emotions include the sensations of well being and malaise, calmness and tension, pain and pleasure, enthusiasm and depression. Social emotions include shame, jealousy, guilt, and pride. Emotions are difficult to deal with in many situations, especially in situations in which one might not even expect emotions to be a factor, such as the deployment of new software.

OT and deployment of enabling CBSs, such as ERP systems, have a variety of impacts on the organization’s employees. These impacts are often ignored in the specification of the requirements for the CBSs [5, 8, pp. 41–44]. These impacts include:

- the CBS’s degradation of the employees’ quality of work life, by reducing job security and by increasing stress and uncertainty in pursuing task and career interests [8],
- the CBS’s impact on the informal communication responsible for friendship, trust, the feeling of belonging, and self respect [9],

- the power imbalances the CBS will cause [6, 10], and
- the employees' loss of work and life meaning, which can lead to depression [8, 11].

It is clear that we are talking implicitly in this viewpoint about negative emotions, against or confounding the construction or deployment of the CBS at hand. If the emotions of all stakeholders were positive, in favor of construction and deployment of the CBS, then emotional issues will not need to be considered in the requirements. In this case, the RE process would proceed as a so-called normal one, focusing strictly on the standard functional and nonfunctional requirements of the algorithmics, the technology, and the economics of the CBS.

3. Examples of Requirements Affected by Emotions

Perhaps the earliest, albeit implicit, recognition of the role of emotion in determining requirements for a CBS in the software engineering research literature was Boehm and Ross's 1989 Theory W [12]. Theory W and its follow-on Win-Win conditions [13] are methods of negotiating requirements so that each stakeholder for a CBS ends up winning in the sense that he or she has at least some of his or her requirements satisfied. The normal alternative has some stakeholders having none of their requirements satisfied in order that others have most of their requirements satisfied. This alternative is thus termed *Win-Lose*. The clearly understood reason for preferring Win-Win to Win-Lose is that when all stakeholders win, they all buy into the system, and there is less chance that some will reject the system as not meeting their requirements. There is less chance of sabotage by losing stakeholders.

Goguen observed in 1993 that [14] "It is not quite accurate to say that requirements are in the minds of clients; it would be more accurate to say that they are in the social system of the client organization. They have to be invented, not captured or elicited, and that invention has to be a cooperative venture involving the client, the users, and the developers. The difficulties are mainly social, political, and cultural, and not technical."[2]

Ramos, in carrying out case studies for her 1998 Ph.D. dissertation [16, 17], examined four organizations around Portugal, in businesses and universities, that were attempting CBS-enabled OTs. She found several examples of software features that raised fears in some stakeholders and of some CBS development processes that were affected by emotion-driven agendas of some stakeholders:

1. A software system that was supposed to store information about mistakes and who was responsible for them stressed out many potential users to the extent that the mistake-logging features had finally to be removed.
2. A library system that was to give all users access to more information so that they could participate more and could be more autonomous stressed out some potential users. They really preferred not to have access to information not specifically related to their own tasks and not to have more responsibility.
3. An off-the-shelf ERP system was shelved in favor of a nearly identical home-brewed system, just to give those who would develop the system a chance to learn and teach the system more gradually and to become indispensible.
4. In an university, a computer supported cooperative work (CSCW) [18] system was introduced to the classroom to allow the students of a team to work together and the faculty member to observe the students' progress. This CSCW system stressed out students who did not work well in teams; who did not trust others not to mess up their own work that was now accessible to others; who did not like the idea of instructors observing their work closely in real time; or who saw files they were editing being modified at the same time by others.

Krumbholz *et al* investigated the negative impact on user acceptance of ERP induced OT that results from a mismatch between the ERP system's actual and perceived functionalities and the users' requirements, including those motivated by their values and beliefs [19, 20]. They have found that users' emotions arising from their values and beliefs affect their acceptance of implemented software.

Bergman *et al* [6, p. 153] "examines [sic] the problem of stating and managing requirements for large system development in terms of heterogeneous engineering, focusing especially on issues of power and interest amongs principals involved in a project." They observe that RE is a political process. They note that there is extensive coverage of this view of RE in the Information Technology literature [10] and in the organizational theory literature [21]. They observe that more recently, the SE and RE literature has begun to explore this view as well [12, 13].

Huff [11] observes that the user orientation of educational and game software is thought to depend on the gender of the user. In particular, educational and game software writers follow gender stereotypes. Moreover, data show that users do indeed seem to have difficulty with software for the wrong gender when they use the software in public view, although not when they use the software in private. Clearly, the user interface of software is a requirements issue. Emotional issues, namely a fear of being

[2] This quotation is from a draft that preceded publication as a chapter in a book [15]. The quotation did not survive into the book chapter. However, by e-mailed personal communication, Joseph Goguen assures us that he still believes in the contents of the quotation and that he does not disown it.

perceived as behaving like a person of the opposite gender, are coming to play with these user-interface requirements.

Sickenius de Souza, Prates, and Barbosa [22] describe lessons learned from developing information technology to be used by volunteer social workers in Brazil. The developers recognized that volunteers are motivated not for money or advancement, but for their own satisfaction. The developers knew that they would have to understand such emotional factors in designing a new interactive system for the volunteers to use to do their work. The developers decided to use the underlying discourse unveiling method (UDUM), developed originally for clinical psychological research. UDUM uses open-ended interviews to allow people to talk freely. UDUM (p. 174) "focuses on grasping and analyzing hidden or implicit fears, desires, motivations, aspirations, conflicts, and other deep feelings experienced by individuals."

Boehm and Huang [23] describe a new method for tracking a project's adherence to its schedule and budget. They observe that the traditional earned-value management process performs well when tracking how closely a project is meeting its original plan. However, there are difficulties. (p. 36) "A project can be tremendously successful with respect to cost-oriented earned value, but an absolute disaster in terms of actual organizational value earned. This frequently happens when the product has flaws with respect to user acceptability, operational cost-effectiveness, or timely market entry." Thus, at least one of the possible budget, schedule, and value wrecking flaws is user acceptability. Whether a user likes and accepts a particular CBS can be as much an emotional issue as a question of available functionality. For example, like many, one of the authors of this viewpoint so dislikes the very idea of Microsoft that he refuses to use MS Windows and MS Office in any serious way, preferring to continue to use comfortable, old-fashioned UNIX software running on a Solaris. He admits that his aversion is not entirely rational. Indeed, Boehm and Huang cite another example: (p. 36) "the initiative to implement a new order-entry system to reduce the time required to process must convince the sales people that using the new system features will be good for their careers. For example, if the order-entry system is so efficiency-optimized, it doesn't track sales credits [which prove who sold what], the sales people will fight using it. The salespeople also must be trained to use the system effectively."

Finally, Johann Rost writes about political reasons for failed software projects [24]. In particular, with several concrete examples, he describes how emotions towards a CBS can lead to subversive behavior and how subversive behavior can sabotage software projects.

4. Are These Just Managerial Issues?

Some who have read our earlier papers accept that there indeed may be the kind of problems mentioned when deploying job-changing CBSs. However, they regard them as managerial problems and not as requirements problems. In one sense, these readers are right, in that the responses to these problems often require action by management, addressing social issues.

However, any problem that can prevent the successful deployment of a system, whether it be

- incorrect function,
- failure to notice tacit assumptions,
- or anything else

should be identified as early as possible so that dealing with it can permeate the entire system design and development process. Perhaps, a so-called managerial problem born of emotion can be solved by a simple change in functionality or user interface, e.g., by eliminating a hated feature entirely. Perhaps, managers and colleagues of the employees that hate the feature should clarify both the business strategy supported by the feature and the benefits of the feature to these employees. Delaying consideration of any problem drives up the cost of solving the problem once it is identified [25]. When viewed this way, all such problems become requirement problems, some of which may be solved by the software of the CBS. In the end, it may very well be that the decided-upon solution to an identified problem may be considered a managerial solution, e.g., educating users and their managers, providing incentives for adopting, etc. However, such solutions, especially that of educating users, may be applied also to what might appear to be a functional or user-interface issue. For example, NASA occasionally simply trains users to follow different steps during control of an unmanned deep-space vehicle rather than modify the on-board software as a solution to a detected failure of the software to meet its original requirements [26].

5. Conclusion

This viewpoint has argued that emotions are just as important and valid as factors affecting requirements as are any other traditional factor including function, performance, cost, and user interface. For a lengthier discussion of (1) the effects of emotions on deployment of CBSs, (2) case studies of deployments affected by emotions, and (3) *how* to elicit emotional issues during the standard RE effort, please see our paper with J.A. Carvalho [1].

Acknowledgments

The authors thank the anonymous referees of our previous papers for their helpful comments. Daniel Berry's work was supported in part by a Canadian NSERC grant, NSERC-RGPIN227055-00.

About the Authors

Daniel M. Berry got his Ph.D. in Computer Science from Brown University in 1974. He was on the faculty of the Computer Science Department at the University of California, Los Angeles, USA from 1972 until 1987. He was in the Computer Science Faculty at the Technion, Israel from 1987 until 1999. From 1990 until 1994, he worked for half of each year at the Software Engineering Institute at Carnegie Mellon University, USA, where he was part of a group that built CMU's Master of Software Engineering program. During the 1998-1999 academic year, he visited the Computer Systems Group at the University of Waterloo in Waterloo, Ontario, Canada. In 1999, Berry moved to the School of Computer Science at the University of Waterloo. Berry's current research interests are software engineering in general, and requirements engineering and electronic publishing in the specific.

Isabel Ramos got her doctor's degree in Information Technologies and Systems with a specialization in Information Systems Engineering and Management in 2001. She has also a master's degree in Informatics for Management. Ramos is an Assistant Professor in the Information Systems Department of the University of Minho, Portugal. She is a researcher in the Algoritmi Research Center of the University. She coordinates Knowledge Management interest group of her department. She is responsible for the Requirements Engineering modules in the Master's Course in Information Systems. She integrates the steering committee of a Master's Course in Business Information. Ramos's main areas of interest are requirements engineering, knowledge management, organizational theory, sociology of knowledge, history of science, and research methodology.

References

[1] Ramos, I., Berry, D.M., and Carvalho, J.A., "Requirements Engineering for Organizational Transformation", *Journal of Information and Software Technology* **47**(5), pp. 479–495 (May 2005).

[2] Lehman, M.M., "Programs, Life Cycles, and Laws of Software Evolution", *Proceedings of the IEEE* **68**(9), pp. 1060–1076 (September 1980).

[3] Esteves, J. and Pastor, J., "Enterprise Resource Planning Systems Research: An Annotated Bibliography", *Communications of the Association for Information Systems* **7**(8), pp. 1–51 (August 2001).

[4] Dickson, G.W. and DeSanctis, G., *Information Technology and the Future Enterprise: New Models for Managers*, Prentice-Hall, Englewood Cliffs, NJ (2000).

[5] Palmer, I. and Hardy, C., *Thinking about Management*, Sage, Thousand Oaks, CA (2000).

[6] Bergman, M.B., King, J.L., and Lyytinen, K., "Large-Scale Requirements Analysis Revisited: The need for Understanding the Political Ecology of Requirements Engineering", *Requirements Engineering Journal* **7**(3), pp. 152–171 (2002).

[7] Damásio, A., *The Feeling of What Happens: Body and Emotion in the Making of Consciousness*, Harcourt Brace, New York, NY (1999).

[8] Parker, S. and Wall, T., *Job and Work Design: Organizing Work to Promote Well-Being and Effectiveness*, Sage, Thousand Oaks, CA (1998).

[9] Snizek, W.E., "Virtual Offices: Some Neglected Considerations", *Communications of the ACM* **38**(9), pp. 15–17 (1995).

[10] Markus, M.L., "Power, Politics and MIS Implementation", *Communications of the ACM* **26**, pp. 430–444 (1983).

[11] Huff, C., "Gender, Software Design, and Occupational Equity", *Inroads, SIGCSE Bulletin* **34**(2), pp. 112–115 (June 2002).

[12] Boehm, B.W. and Ross, R., "Theory W Software Project Management: Principles and Examples", *IEEE Transactions on Software Engineering* **SE-15**(7), pp. 902–916 (July 1989).

[13] Boehm, B., Bose, P., Horowitz, E., and Lee, M. J., "Software Requirements As Negotiated Win Conditions", pp. 74–83 in *Proceedings of the First International Conference on Requirements Engineering(ICRE94)*, IEEE Computer Society, Colorado Springs, CO (April 1994).

[14] Goguen, J.A., "Requirements Engineering as the Reconciliation of Technical and Social Issues", Technical Report, Centre for Requirements and Foundations, Programming Research Group, Oxford University Computing Lab, modified version later published as [15], Oxford, U.K. (October 1993).

[15] Goguen, J.A., "Requirements Engineering as the Reconciliation of Technical and Social Issues", pp. 165–199 in *Requirements Engineering: Social and Technical Issues*, ed. J.A. Goguen and M. Jirotka, Academic Press, London, UK (1994).

[16] Ramos, I.M.P., "Aplicações das Tecnologias de Informação que Suportam as Dimensões Estrutural, Social, Política e Simbólica do Trabalho", Ph.D. Dissertation, Departamento de Informática, Universidade do Minho, Guimarães, Portugal (2000).

[17] Santos, I. and Carvalho, J.A., "Computer-Based Systems that Support the Structural, Social, Political and Symbolic Dimensions of Work", *Requirements Engineering* **3**(2), pp. 138–142 (1998).

[18] Easterbrook, S.M., *SCW: Co-operation or Conflict?*, Springer, London, UK (1992).
[19] Krumbholz, M., Galliers, J., Coulianos, N., and Maiden, N.A.M., "Implementing Enterprise Resource Planning Packages in Different Corporate and National Cultures", *Journal of Information Technology* **15**, pp. 267–279 (2000).
[20] Krumbholz, M. and Maiden, N.A.M., "The Implementing of ERP Packages in Different Organisational and National Cultures", *Information Systems Journal* **26**(3), pp. 185–204 (2000).
[21] Pfeffer, J. and Salancik, G., *The External Control of Organizations: a Resource Dependence Perspective*, Harper & Row, New York, NY (1978).
[22] Sickenius de Souza, C., Prates, R.O., and Barbosa, S.D.J., "Adopting Information Technology as a First Step in Design, Lessons Learned from Working with Brazilian Social Volunteers", *Interactions* **X**(2), pp. 72–79 (March+April 2003).
[23] Boehm, B.W. and Huang, L.G., "Value-Based Software Engineering: A Case Study", *IEEE Computer* **36**(3), pp. 33–41 (March 2003).
[24] Rost, J., "Political Reasons for Failed Software Projects", *IEEE Software* **21**(6), pp. 104, 102–103 (November+ December 2004).
[25] Boehm, B.W., *Software Engineering Economics*, Prentice-Hall, Englewood Cliffs, NJ (1981).
[26] Lutz, R.R. and Mikulski, I.C., "Empirical Analysis of Safety-Critical Anomalies During Operations", *IEEE Transaction on Software Engineering* **SE-30**(3), pp. 172–180 (March 2004).