GUEST EDITORIAL PREFACE

Special Issue on Emotions and People-Oriented Programming

Antonio A. Lopez-Lorca, Faculty of Science, Engineering, and Technology, Swinburne University of Technology, Melbourne, Australia

Leon Sterling, Faculty of Science, Engineering, and Technology, Swinburne University of Technology, Melbourne, Australia

Tim Miller, Department of Computing and Information Systems, University of Melbourne, Melbourne, Australia

...even if a design is elegant and functional, it will not have a place in our lives unless it can appeal at a deeper level, to our emotions. — Hartmut Esslinger (Sweet, 1999, p. 9).

There is a growing appreciation in software engineering research and practice that traditional software development and programming paradigms do not adequately consider social objectives (Baxter & Sommerville, 2010; Rahwan, Juan, & Sterling, 2006; Walenstein, 2003). For example, Baxter and Sommerville note the lack of modelling approaches that consider socio-technical considerations:

Modelling and abstraction is fundamental to software engineering, with models of different types being used by engineers to communicate. The practical use of socio-technical approaches has to acknowledge this by providing a means of modelling, and by integrating with existing approaches. [. . .] The abstractions currently used in technical system modelling (e.g., use-cases, objects, etc.) do not seem to us to be sufficient to represent socio-technical considerations. — Baxter and Sommerville (2010).

Recent work has investigated various modelling notations for capturing social objects (Miller, Pedell, Sterling, Vetere, & Howard, 2012; Yu, 2009), using concepts such as quality goals or soft goals to model classical non-functional requirements such as reliable and secure, but also less tangible non-functional requirements such as being fun, trustworthy, and flexible.

While these soft issues are important, one aspect that has not received sufficient attention from the software engineering community, and the IT community more broadly, is that of emotions. An emotion is a feeling that characterises a state of mind. Examples of emotions include feeling joy, terror, or safety.

It is apparent that interactions with software can elicit emotions. Game developers often pay special attention to emotions engendered when playing the game as part of their design. However, emotions are important outside of the context of games too. Bentley et al. (2002) note the example of two near-identical government websites that were commissioned from the same development company. In one project, the department desired a traditional website with static content, while on the other, the client expressed a need for their website to be stylish and fun. By explicitly considering these as user requirements, the second website was found to be more engaging to visitors, and further, visitors were encouraged to explore the site more than the other.

Ramos et al. (2005) present several other examples of projects that failed because they did not consider user emotions. They advocate identifying these issues as early as the requirements elicitation phase, and suggest techniques for identifying emotional issues when talking with stakeholders, including using a psychologist as part of the requirements team.

Pedell et al. (2014) present a study into emergency alarm systems for older people. They found that, while these systems are well engineered products that are highly reliable in raising alarms when enacted, they often failed in their overall objective, which is for older people to get help when in need. They found the common problem of older people failing to wear their emergency alarm pendant around their neck, due to the emotional stigmas attached to them, such as making them feel old and less independent. Pedell et al.'s conclusion is that by considering emotional issues during design, emergency alarm systems could be better aligned with their users' needs, and ultimately result in better social objectives.

While considering emotional issues can improve systems, to achieve repeatable results in such projects, we must embed these concepts throughout the entire software development lifecycle. This includes considerations such as how to model user emotions, how to design systems to considering emotional requirements, and how to implement and evaluate emotional requirements.

There is a small collection of papers investigating emotions in software engineering (Ramos, Berry, & Carvalho, 2005; Bentley, Johnston, & von Baggo, 2002; Miller, Pedell, Mendoza, Sterling, Kiernan, & Lopez-Lorca, 2014; Colomo-Palacios, Casado-Lumbreras, Soto-Acosta, & Garc´ıa-Crespo, 2011), however, this field is still in its infancy.

The aim of this special issue is to encourage and bring together work on how to incorporate emotions into software development and programming. In a literal sense, considering emotions is of necessity people-- oriented programming. There are three papers in this special issue, which were selected after a round of anonymous peer reviews.

The first paper of the special issue, Agent-Based Modelling of Emotional Goals in Digital Media Design Projects by James Marshall, outlines the author's experience in extending agent-based models to capture emotional needs of users in digital media design projects. Inspired by the use of emotions in product design, Marshall proposes the use of emotional goals to represent the emotional needs of users. These emotional goals are linked to specific functional goals within the system, and represent a desired emotional state of a user who is using that functionality. Marshall extends an existing agent-based modelling framework (Sterling & Taveter, 2009) to place these emotional goals in the context of a greater socio-technical system. Marshall describes his experience over two years in using these with students in a digital media design project at Swinburne University of Technology in Melbourne, Australia. In this project, students are divided into small teams and each team contributes to part of a greater project. The product, called Aspergion, is a game that aims to promote respect for people with Asperger's syndrome. The Aspergion project's functional, quality, and emotional goals were modelled using the approach described in Marshall's paper. As well as providing a shared understanding of the project goals, the students used the models as a project

management tool. Each week, the goals on the model were coloured depending on the progress of the project. This gave all participants, including both students and staff, a simple way to evaluate progress on the project. The models were printed on large rolls of paper and placed on a wall in the student lab. Overall, 160 students in two years used these models in their projects, demonstrating that the approach scales to large teams and large projects.

The second paper of the special issue, Design of an evaluation tool to measure emotional goals by Maheswaree Kissoon Curumsing, Sonja Pedell, and Rajesh Vasa, proposes a tool for measuring a user's perception of how well a software product fulfils an emotional goal. The authors describe a series of proposed tools that were considered and refined until they reached the final proposal. This proposal uses a straightforward questionnaire to ask users their perceptions. For each emotion, E, a question is presented in the form I feel E, and the user responds by marking an X on a continuous line, with the scale Always to Never. They evaluate the questionnaire on a prototype application used for older people to keep in touch with their relatives. The system has some key emotional requirements, such as making the older person feel In touch and Cared about. Further, their relatives, also users, can use the system to feel Reassured that the older person is safe and well. The authors trialled the prototype in the home of nine older people for a period of two weeks, and asked all participants to complete the questionnaire at the end of the trial period. Interestingly, as well as asking each participant (older person and relative) how well the system fulfilled the emotional goals, the authors asked the corresponding participants their perceptions of how well the other person's emotional goals were fulfilled.

For example, they asked each older person: I think that my relative feels reassured. The results indicate that their prototype achieves many of the emotional goals for which it was designed, but interestingly, that the perceptions that the older person and their relative had about each other were inconsistent on several of the emotional goals.

The third paper of the special issue, Understanding, Modeling and Exploiting User's Emotions for Brain-Driven Interface Design by Valeria Carofiglio and Fabio Abbattista, presents an approach to design 3D virtual environments that adapt their content based on the user's emotions. Real time data on the user's emotions is captured using brain electrical activity sensors. The authors conduct their experiment using a virtual environment to build scenes that show the life of prisoners in Nazi extermination camps during World War II. They chose this domain because it generates complex emotions in users. The authors assess the emotional impact of different elements of the scenes, i.e. 3D recreations and historical multimedia data. The authors describe an initial experiment where users visualise different scenes and then rate the scenes based on their subjective perception in terms of sadness, anger or fear. These perceptions are compared with the values captured by the brain computer interface worn by users during the experience to assess their consistency. Based on these results, the authors redesign the scenes and configure the 3D virtual environment. Their goal is to maintain an appropriate level of frustration in users throughout the whole experience, based on the principle that frustration helps to maintain engagement. However, too much frustration leads to a feeling of detachment from the content and may make users feel too distressed to continue. The adaptation mechanism designed by the authors monitors the brain electric activity of users while watching the scene. The system schedules content of high emotional intensity until it detects that the user feels too frustrated, then it switches to content with lower emotional intensity. Conversely, content of higher emotional intensity is scheduled when users' frustration levels fall too low. A second experiment shows promising results in maintaining the engagement in users of the 3D virtual environment.

The collection of three papers add to the literature. We have enjoyed preparing the special

issue. We hope that the special issue encourages more research into incorporating emotions in software development and people-oriented programming.

ACKNOWLEDGMENT

The authors are funded by the Australian Research Council Discovery Grant DP130102660.

REFERENCES

Baxter, G., & Sommerville, I. (2010). Socio-technical systems: From design methods to systems engineering. *Interacting with Computers*, *23*, 4–17. doi:10.1016/j.intcom.2010.07.003

Bentley, T., Johnston, L., & von Baggo, K. (2002). Putting some emotion into requirements engineering. In *Proceedings of the 7th Australian Workshop on Requirements Engineering* (pp. 227–244).

Colomo-Palacios, R., Casado-Lumbreras, C., & Soto-Acosta, P., & Garc´ıa-Crespo, A. (2011). Using the affect grid to measure emotions in software requirements engineering. *Journal of Universal Computer Science*, *17*(9), 1281–1298.

Miller, T., Pedell, S., Mendoza, A., Sterling, L., Kiernan, A., & Lopez-Lorca, A. (2014). *Emotionally-driven models for people-oriented software: The case study of emergency systems* (Under review).

Miller, T., Pedell, S., Sterling, L., Vetere, F., & Howard, S. (2012). Understanding socially oriented roles and goals through motivational modelling. *Journal of Systems and Software*, *85*(9), 2160–2170. doi:10.1016/j.jss.2012.04.049

Pedell, S., Lopez, A., Miller, T., & Sterling, L. (2014). *Don't leave me untouched: Considering emotions in personal alarm use and development* (Under review).

Rahwan, I., Juan, T., & Sterling, L. (2006). Integrating social modelling and agent interaction through goal-oriented analysis. *International Journal of Computer Systems Science & Engineering*, *21*(2), 87–98.

Ramos, I., Berry, D., & Carvalho, J. (2005). Requirements engineering for organizational transformation. *Information and Software Technology*, *47*(7), 479–495. doi:10.1016/j.infsof.2004.09.014

Sterling, L., & Taveter, K. (2009). *The art of agent-oriented modelling*. MIT Press.

Sweet, F. (1999). *Frog: Form follows emotion*. Thames & Hudson.

Walenstein, A. (2003). Finding boundary objects in SE and HCI: An approach through engineering-oriented design theories. Bridging the gaps between software engineering and human-computer interaction (pp. 92–99).

Yu, E. (2009). Social modeling and i*. Conceptual modeling: Foundations and applications (pp. 99-121).