Human Factors in Software Development Processes: Measuring System Quality

Chapter · November 2016

DOI: 10.1007/978-3-319-49094-6_57

Silvia Abrahão
Universitat Politècnica de València
143 PUBLICATIONS **1,113** CITATIONS
SEE PROFILE

Maria Teresa Baldassarre
Università degli Studi di Bari Aldo Moro
66 PUBLICATIONS **380** CITATIONS
SEE PROFILE

Yvonne Dittrich
IT University of Copenhagen
146 PUBLICATIONS **1,148** CITATIONS
SEE PROFILE

Antonio Piccinno
Università degli Studi di Bari Aldo Moro
92 PUBLICATIONS **759** CITATIONS
SEE PROFILE

Human Factors in Software Development Processes: Measuring System Quality

Silvia Abrahao[1], Maria Teresa Baldassarre[2], Danilo Caivano[2], Yvonne Dittrich[3], Rosa Lanzilotti[2], Antonio Piccinno[2]✉

[1]Universidad Politecnica de Valencia (UPV), Spain,
[2]Università degli Studi di Bari Aldo Moro, Italy
[3]IT University of Copenhagen, Denmark

[1]sabrahao@dsic.upv.es, [2]name.surname@uniba.it, [3]ydi@itu.dk

Abstract. Software Engineering and Human-Computer Interaction look at the development process from different perspectives. They apparently use very different approaches, are inspired by different principles and address different needs. But, they definitively have the same goal: develop high quality software in the most effective way. The second edition of the workshop puts particular attention on efforts of the two communities in enhancing system quality. The research question discussed is: who, what, where, when, why, and how should we evaluate?

Keywords: Human Computer Interaction, Software Engineering, Human factors, Development process

1 Introduction and Motivation

Software Engineering (SE) and Human-Computer Interaction (HCI) look at development processes of interactive software systems from different perspectives. Efforts to reduce the gap between the two communities for what concerns the introduction of human factors in software development processes have started to be discussed in the first edition of the workshop, held in Bolzano in 2015 [1]. One aspect that emerged from the discussion pointed out concerns on how system quality should be measured, in order to satisfy both communities. Indeed, the software product industry emphasizes how important it is to involve users and customers to evaluate quality in terms of functionality and usability of software products [2].

Although researchers and practitioners from the two communities share the same goal of developing high quality systems, the methodologies, methods and metrics they use to evaluate such quality are very different due to their background and expertise. The second edition of the workshop on Human Factors in Software Development Processes aims at providing a forum for discussing measuring system quality from both perspectives. In particular, the following research issues are addressed:

- key methods that allow to integrate human factors in the evaluation of the software quality;
- methodologies and techniques currently used in software development teams to engage users in the evaluation process;
- how the level of human factor involvement can be objectively verified during and after software development;
- how Software Engineering and HCI researchers and practitioners can overcome the communication gap when evaluating system quality.

Researchers and practitioners who face the problem of integrating human factors in software quality evaluation should have a place to discuss their experiences, lessons learned and future intentions to reach a common understanding on evaluation topics.

2 Filling the GAP between SE and HCI

SE and HCI apparently use very different approaches, are inspired by different principles and address different needs. Ultimately though they have the same goal: developing high quality software in the most effective way.

Based on the discussions of the previous edition and of the contributions received, the authors of this paper have classified some gaps between the two communities that can be seen as two sides of the same medal. In the following, we illustrate and discuss: (i) the main differences between SE and HCI approaches adopted, (ii) categorize the common wisdoms and (iii) explore possible ways to reduce the gap and converge.

2.1 The Differences

2.1.1 User vs. Market Oriented Systems

One of the most popular claims in the two communities is that they address different types of software products.

The HCI products are User Oriented Systems where typically there is some type of "users" (user, lead user, customer and so on) to refer to during the development. Here the source of requirements (functional and non functional) is primarily the "user" himself who is actively involved in the design, review and validation, i.e. ingrained in the entire development process right up until delivery. The systems are very focused on specific domains, they offer the functions that are specialized on users' needs.

The SE products are Market Oriented Systems that address a wide range of needs and are used by tons of users that differ for functions used, language, culture, ability, competences and skills. Here the source of requirements are the laws, domain and business rules, books, the already existing legacy systems, competitors and, in general, the so called stakeholders. The "users" are rarely involved and typically a "customer" doesn't exit because the developed product is addressed to an entire market segment with hopefully hundreds of customers. The systems are so big that there isn't a single “user” that knows all the requirements to develop and thus the “users” are simply not essential.

2.1.2 Vertical vs Layered or Horizontal architecture

An assertion emerged in the previous workshop is that the differences between User and Market Oriented product imply the use of different software architectures. The HCI architectures were perceived as “vertical” in that they address specific needs of specific users from specific domains. Due to these characteristics the system development starts from interface until database without particular attention to maintainability and reuse. The resulting systems are generally characterized by a high coupling between data functions and interfaces and a possible low internal cohesion.

The SE architectures are usually layered architectures inspired by principles such as modularization, separation of interest, information hiding, reuse etc. and the resulting systems are assumed to be maintainable and robust. The interface is only one of the system layers and it is often considered less important than others such as the data layer or business layer. Here the belief is that the obtained systems are so maintainable that their interfaces can be adjusted or completely changed in a short time and fashionable way without particular problems. Common architecture styles are Software Product Lines, Enterprise Architectures, Service Oriented Architectures etc. A software system is naturally perceived along the horizontal dimension because it covers a wide range of domains, from business to technical. A typical example is an Enterprise Resource Planning.

2.1.3 User Time - ex ante vs ex post

A crucial point in the SE-HCI debate is the decision on when the users should be involved. The HCI philosophy is “as soon as possible” while the SE face states that the “users are rarely involved”. This is the result of a numbers of convictions; for HCI: the interface and usability are crucial for system success and the nature and type of interface strongly influences the architecture style; The interface cannot be simply pushed into a system right after being developed; The source of requirements is also the “user” and thus drives system development from the start (User Driven Development) [7].

SE: in this community a Process/Product Driven Development is adopted. The source of requirements “can also” be the user but he/she is generally not considered particularly important, rather time consuming and misleading. If the system is developed according to the SE processes and principles the user interface can be easily pushed into the system at the end of development when the critical and most important layers are completed. The final users can be involved before software delivery to carry out pilot studies aimed at validating user satisfaction and product correctness. All the suggestions coming from users are then quickly and effectively incorporated.

2.2 Common wisdom

Following to discussions of the previous edition we have outlined some “common wisdom” points spread within the communities, just to mention a few:

- HCI systems are more usable than SE ones;
- HCI systems are less maintainable than SE ones
- HCI systems are less performing than SE ones

Obviously these are anecdotal assertions and with lack of any evidence with the only intent of soliciting and marking the differences and distance between the two communities.

2.3 Bridging the Gap

How do we reduce the gap? What approaches, techniques or processes should we use? Discussion and information collected among participants of both communities were trivial in some cases and original in others. They have been summarized in the following list:

- make use of short iterations, meetings and focus groups between “stakeholders” during system development in order to reduce the risks of omitted requirements and deliver unusable systems. Stakeholders obviously include among others, users, lead users and customers as well as developers and experts. To this end during the discussion of this point was clear that there is a misleading use of terms in the two communities and that most likely terms like user, lead user or customer as intended by HCI are included and considered by SE as stakeholders;
- use of lightweight processes, especially those ones proposed in HCI community that starting from the well known agile processes, such as SCRUM or XP, extend them by actively involving the “user” or “customer”;
- concepts such as experimentation and empirical evaluation can represent a common means for both communities. Action research, ethnographic studies, Cooperative Method Development, formal experiments, case studies [6], surveys, qualitative and quantitative evaluations etc. could become a shared platform of methods for a joint evaluation of what is done in the two communities;
- use of interdisciplinary teams that include both HCI and SE experts;
- definition and use of shared quality models able to objectively evaluate the quality of the system developed.

In the end we can observe that in the recent years the two communities are progressively converging towards a set of common practices, process and empirical techniques. The considerations above and the outcomes of the discussions enhanced during the first edition of the workshop suggest to refer to empirical approaches [5] and evaluation of software quality [3, 4] as a way for enhancing the convergence between communities and improves software quality overcoming skepticism and common wisdoms.

Despite the discussion, the research problem still remains: who, what, where, when, why, and how should we evaluate quality?

Nowadays software quality approaches are also converging towards product evaluation from different point of views: internal, external quality, quality in use. For example, if we refer to ISO/IEC 25000 we can observe that it standardizes the processes and models for product evaluation and provides useful guidelines for addressing system quality in a more objective way. This has been one of starting points for triggering further discussion in the current edition.

3 Audience and Expected Outcomes

The overall goal of this interdisciplinary workshop has been to raise the level of engagement and discussion about human factors in software quality measurements. A further goal of the workshop has been to identify opportunities to improve the synergy of the two communities on the scientific discourse and progress on human aspects of software evaluation, as well as to better identify opportunities able to educate practitioners and researchers about how to conduct sound human-centered evaluations in the context of software engineering. Indeed, the organizers of the workshop are a synergic composition of active researchers belonging to both communities. The expected outcome is a descriptive framework that helps to organize the current best practices and a set of recommendations for formalizing and verifying software system quality.

The workshop has received a positive response from both HCI and SE communities with several interesting and valuable contributions. The submissions were peer-reviewed by international committee members for their quality, topic relevance, innovation, and potentials to foster discussion. Finally, five papers were accepted.

In the first paper, "Gamification and Functional prototyping to Support Motivation towards Software Process Improvement" authors discuss commitment in software process improvement initiatives in the context of people-driven processes to help ensure software quality. Gamification and functional prototyping are used as a means to boost motivation and commitment.

The second paper "Exploring Mobile User Experience through Code Quality Metrics" presents a set of features for evaluating the code quality of Android applications. The discussion points out how user experience varies in mobile ecosystems and who developers should focus on software quality to assure usable applications from a user perspective.

In the third paper "Early-Usability in Model-Driven Game Development", authors propose a model that can be used to evaluate the usability of video games in early stages of development. Moreover, the method relies on a model that decomposes usability into measurable attributes and metrics specific to the video game domain, bridging de facto a gap between SE and HCI.

In the fourth paper "What aspects of context should be described in case studies about software teams? Preliminary results from a mapping study", authors illustrate the results of a mapping study aiming at addressing human-based factors that influence the selection and composition of software engineering teams and how these can influence and impact final quality.

Finally, in the fifth contribution "Miscommunication in Software Projects: Early recognition through tendency forecasts", authors address issues of team communication and how miscommunication can lead to delay of software releases and especially hamper customer satisfaction. Aspects related to team composition and their interaction with users is also addressed.

Acknowledgment. We would like to thank the organizers of PROFES 2016 for giving us the opportunity to organize this workshop. We are also grateful to our international program committee of experts in the field for their reviews and collaboration.

References

1. Abrahao, S., Baldassarre, M.T., Caivano, D., Dittrich, Y., Lanzilotti, R., Piccinno, A.: Human Factors in Software Development Processes. In: Abrahamsson, P., Corral, L., Oivo, M., Russo, B. (eds.) Product-Focused Software Process Improvement: 16th International Conference, PROFES 2015, Bolzano, Italy, December 2-4, 2015, Proceedings, pp. XIV-XVI. Springer International Publishing, Cham (2015)
2. Costabile, M.F., Fogli, D., Lanzilotti, R., Mussio, P., Piccinno, A.: Supporting Work Practice Through End-User Development Environments. Journal of Organizational and End User Computing 18(4)**,** 43-65 (2006)
3. Pardo, C., Pino, F.J., García, F., Piattini, M., Baldassarre, M.T. :A process for driving the harmonization of models. ACM International Conference Proceeding Series, pp. 51-54. DOI: 10.1145/1961258.1961271. (2010)
4. Pardo, C., Pino, F.J., García, F., Velthius, M.P., Baldassarre, M.T.: Trends in Harmonization of Multiple Reference Models. Communications in Computer and Information Science, 230, pp. 61-73. DOI: 10.1007/978-3-642-23391-3_5. (2011)
5. Ardimento, P., Caivano, D., Cimitile, M., Visaggio, G. Empirical investigation of the efficacy and efficiency of tools for transferring software engineering knowledge. Journal of Information and Knowledge Management, 7 (3), pp. 197-207 DOI: 10.1142/S0219649208002081 (2008)
6. Baldassarre, M.T., Bianchi, A., Caivano, D., Visaggio, G.: An industrial case study on reuse oriented development. IEEE International Conference on Software Maintenance, ICSM, 2005, art. no. 1510124, pp. 283-294. DOI: 10.1109/ICSM.2005.20 (2005)
7. Ardito C., Buono P., Caivano D., Costabile M. F., Lanzilotti R., Bruun A., Stage J.: Usability evaluation: a survey of software development organizations. International Conference on Software Engineering and Knowledge Engineering (SEKE 2011), Miami, Florida, USA, July 7-9, 2011. (pp. 282-287).