Information and Software Technology 74 (2016) 181–182

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Guest editorial for the special section with the extensions to the best papers from APSEC'13 and APSEC'14☆

Yann-Gaël Guéhéneuc[a,*], Gi-hwon Kwon[b], Pornsiri Muenchaisri[c]

[a] Department of Computer and Software Engineering Polytechnique Montréal C.P. 6079, succ. Centre-Ville Montreal, Quebec, Canada, H3C 3A7
[b] Department of Computer Science Kyonggi UniversitySan 94-6, Yiui-dong, Paldal-gu Suwon-si, Kyonggi-do, Korea, 442-760
[c] Department of Computer Engineering Chulalongkorn University Phyathai Road Bangkok, 10330, Thailand

ARTICLE INFO

Article history:
Received 13 January 2016
Revised 23 January 2016
Accepted 23 January 2016
Available online 3 March 2016

Keywords:
Software development
Requirements
Requirement ambiguities
Processes
Scrum
Scrum anti-patterns
Distributed software development
Git
Reviewing
pull-requests
Pull-request assignment

1. Introduction

It is now a platitude to say that software systems run our lives. Yet, software development remains a difficult endeavor. Among many studies of failed software projects, a recent study of 5400 large software projects, by McKinsey & Company in conjunction with the University of Oxford [1], reports that (1) 17% of the projects run into problems that threaten the very existence of the company and (2) 45% run over budget and 7% over time, while delivering 56% less value than predicted. These figures are worrisome and prompted both academia and the industry to study the software development process in details to understand the causes of these failures. Among many studies, both academia, e.g., [2,3] and industry, e.g., [4], recognize the importance of the *requirements* and of the *process* used to transform requirements into software artifacts.

Requirements are the basis on which customers and project members form a partnership to develop software systems. Consequently, requirements are both a contract between stakeholders and a blueprint for the software systems. They must have several qualities if the projects are to succeed. They must be correct, complete, clear, unambiguous, understandable, consistent, necessary, verifiable, traceable, feasible, independent of one another, of the design, and of the implementation [5,6]. However, achieving these qualities mostly remain difficult and dependent upon the abilities and expertise of stakeholders. In particular, building and maintaining unambiguous requirements are challenging tasks due to the very nature of natural languages, which for Janardan Misra proposes a tool based on syntactic and semantic aliasing of terms to help identifying and resolving ambiguous terms.

But, even with quality requirements, software projects are very much at the mercy of the software development process chosen by the stakeholders because software development is a non-linear process and, therefore, adaptation to changing requirements is more important than optimizing the software development process

☆ This special section contains five papers published at APSEC'13 and APSEC'14. The 20th Asia-Pacific Software Engineering Conference (APSEC'13, http://apsec2013.eng.chula.ac.th/) took place on 2-5 December 2013 in Bangkok, Thailand. The 21st Asia-Pacific Software Engineering Conference (APSEC'14, http://www.apsec2014.org/main/) took place on 1-4 December 2014 in Jeju, Korea.

* Corresponding author. Tel.:+66819895429.
E-mail addresses: yann-gael.gueheneuc@polymtl.ca, apsec1314@gmail.com (Y.-G. Guéhéneuc), khkwon@kyonggi.ac.kr (G.-h. Kwon), Pornsiri.Mu@Chula.ac.th (P. Muenchaisri).

http://dx.doi.org/10.1016/j.infsof.2016.01.015

per se. The Agile Manifesto [7] embraced changing requirements and asserted that stakeholders and their interactions are more important than the chosen process. It gave birth to different processes from the well-known Extreme Programming (XP), which made it popular, to Scrum, which is nowadays largely adopted by small and big companies likewise [8]. But agile processes such as Scrum are not without controversies [9] and their own sets of problems [10,11], which Veli-Pekka Eloranta, Kai Koskimies, and Tommi Mikkonen explore and codify in the form of 14 anti-patterns.

Agile processes became prominent coincidently at the time of the raise of the open-source movement and of on-line version control systems, issue tracking systems, and source-code hosting Web sites. After CVS (SourceForge) and SVN (SourceForge again), Git is now a de facto standard for many open and closed source software projects and (was) made popular (by) source-code hosting services like BitBucket and GitHub. These services integrate and make accessible in one place various tools supporting software projects, including but not limited to issue tracking, source code versioning, reviewing and commenting, quality assessment. These services foster collaboration around the code but, consequently, tend to be more taxing on core members of the projects for management [12], in particular reviews. Therefore, GitHub and other services promote a collaborative reviewing and Yue Yu, Huaimin Wang, Gang Yin, and Tao Wang propose a novel approach to support this collaborative effort by helping identifying the most appropriate reviewers.

Although software development remains a difficult endeavor, software systems now run our lives and they are nowadays very different from those systems of 10 or even 5 years ago. Mobile computing took over desktops computers and changed forever the ways that users interact with their systems [13]. Users now expect connectivity and services tailored to their particular needs and place. Tailoring services based on the users' contexts is therefore required to provide a quality user experience with relevant and timely information. It is so important that dedicated workshops and conferences sprung to existence in the past years, in particular the International Conference on Context-Aware Systems and Applications[1] supported by the European Alliance for Innovation and endorsed by major research centers and enterprises, such as Fraunhofer, Technion, IBM, and Microsoft[2]. However, contextual information is often noisy and partial and it must be checked against contextual rules to ensure its relevance and timeliness. We acknowledge the importance of contextual information with two articles verifying contextual information fast, using GPU by Jun Suia, Chang Xua, S.C. Cheung, Wang Xia, Yanyan Jianga, Chun Caoa, Xiaoxing Maa, and Jian Lu, while taking into account patterns of ephemeral and harmless inconsistencies by Wang Xi, Chang Xu, Wenhua Yang, Xiaoxing Ma, Ping Yu, and Jian Lu.

Thus, the articles in this Special Section covers a breadth and depth of relevant and timely topics related to software development, from processes to behavior. They showcase the state of the art on these topics and we are sure that they will provide the sound basis for new, future work that will further change the face of software development and of our daily uses of software systems.

—The guest editors:
Pornsiri Muenchaisri <pornsiri.mu@chula.ac.th> for APSEC'13
Yann-Gaël Guéhéneuc and Gi-hwon Kwon <yann-gael.gueheneuc@polymtl.ca> for APSEC'14

References

[1] Michael Bloch, Sven Blumberg, Jürgen Laartz, Delivering Large-Scale IT Projects on Time, on Budget, and on Value, Insights & Publications, McKinsey & Company, October 2012.Available at http://www.mckinsey.com/insights/business_technology/delivering_large-scale_it_projects_on_time_on_budget_and_on_value.
[2] Fabian de Bruijn, HansL. Dekkers, Ambiguity in natural language software requirements: a case study, Requirements Engineering: Foundation for Software Quality, LNCS, Springer, July 2010, pp. 233–247. Available at https://books.google.ca/books?id=fK3xJpqGhoUC&pg=PA233.
[3] SriFatimah Tjong, Avoiding Ambiguity in Requirements Specifications Ph.D. Thesis, University of Nottingham, February 2008.Available at https://cs.uwaterloo.ca/~dberry/FTP_SITE/students.theses/TjongThesis.pdf.
[4] Alliance - The Software Inside; The critical requirements for ambiguity testing; white paper. Available at http://www.allianceglobalservices.com/white-papers/the-critical-requirements-for-ambiguity-testing. Last accessed on 2016/01/10.
[5] OpenUP; Checklist: Qualities of Good Requirements; Eclipse Process Framework Composer. Available at http://epf.eclipse.org/wikis/openupsp/index.htm. Last accessed on 2016/01/10.
[6] Peter Zielczynski, Requirements Management Using IBM Rational RequisitePro, IBM Press, June 2008.Available at http://www.ibmpressbooks.com/articles/article.asp?p=1152528.
[7] The Agile Manifesto. Available at http://agilemanifesto.org/. Last accessed on 2016/01/10.
[8] Steve Denning, Scrum is a major management discovery; opinion piece, leadership, Forbes (April 2011) Available at http://www.forbes.com/sites/stevedenning/2011/04/29/scrum-is-a-major-management-discovery/.
[9] MichaelO. Church, Agile software development: why do some developers at strong companies like google consider agile development to be nonsense? Quora (March 2015) Available at https://www.quora.com/Agile-Software-Development-1/Why-do-some-developers-at-strong-companies-like-Google-consider-Agile-development-to-be-nonsense/answer/Michael-O-Church. Last accessed on 2016/01/10.
[10] MarkC. Paulk, On Empirical Research into Scrum Adoption; Institute for Software Research (August 2011) Available at http://www.cs.cmu.edu/~mcp/agile/oersa.pdf. Last accessed on 2016/01/10.
[11] Scrum.org; ScrumButs and Modifying Scrum. Available at https://www.scrum.org/ScrumBut. (Last accessed on 01.10.16).
[12] Peter Weimann, Christian Hinz, Elsje Scott, Michael Pollock, Changing the communication culture of distributed teams in a world where communication is neither perfect nor complete, Electron. J. Inf. Syst. Evaluat. 13 (2) (October 2010) 187–195.Available at www.ejise.com/issue/download.html?idArticle=676.
[13] David Cardinal; How the Smartphone Changed Everything; or, the Rise of BYOD in the Workplace; ArsTechnica UK, December 2010. Available at http://arstechnica.co.uk/information-technology/2015/12/how-the-smartphone-changed-everything-or-the-rise-of-byod-in-the-workplace/

[1] http://iccasa.org.
[2] http://eai.eu.