Editor: **Jeffrey C. Carver**
University of Alabama
carver@cs.ua.edu

Measuring Systems and Architectures: A Sustainability Perspective

Jordi Cabot, Rafael Capilla, Carlos Carrillo, Henry Muccini, and Birgit Penzenstadler

THIS ISSUE'S "PRACTITIONERS' Digest" department reports on the 2018 Measurement and Metrics for Green and Sustainable Software Systems Workshop (MeGSuS), the ACM/IEEE 21st International Conference on Model Driven Engineering Languages and Systems (MODELS), and the 12th European Conference on Software Architecture (ECSA). Feedback or suggestions are welcome. In addition, if you try or adopt any of the practices included in this article, please send me and the authors of the paper(s) a note about your experiences.

Using Green Metrics

"Towards Green Metrics Integration in the MEASURE platform," by Bagnato and Jérôme,[2] describes the inclusion of green metrics in the European MEASURE ITEA3, which uses a set of industry partners to develop a metrics framework bottom-up. The goal of MEASURE is to provide a systematic reference for companies to improve their assessment of green metrics. The MEASURE platform consists of a web application that allows companies to deploy, configure, collect, compute, store, combine, and visualize software measures defined according to the Object Management Group's Structured Metrics Metamodel. Their aim is to reuse green metrics, drawn from open source project unit tests, across software development projects. Three of the included metrics focus on energy consumption: the energy waste rate, the energy consumption of a software artifact, and energy efficiency. The authors identify three use cases for the tool suite: 1) monitoring energy consumption and/or power peaks at runtime during the software production phase, 2) comparing the energy consumption of similar software components when evaluating their energy efficiency, and 3) building a knowledge base about software energy consumption to help developers at earlier phases. In addition, researchers and practitioners can use the knowledge base as an input for building static analysis methods. The authors are currently soliciting additional metrics relevant to other dimensions of sustainability beyond energy efficiency. This paper appears in MeGSuS. Access it at http://bit.ly/PD_2019_May_1.

A Decade of Modeling Trends

"A Decade of Software Design and Modeling: A Survey to Uncover Trends of the Practice," by Badreddin and colleagues,[1] uses the results from two surveys (given 10 years apart) to report on the trends in adoption of software design and modeling in industry. The survey about modeling practices included the following topics: types of models and languages used, the typical modeling activities used in each software development phase, and the preferred platforms and tools. It also gathered respondents' opinions on the efficacy of modeling, the usability of the tools, and how model-centric approaches compare with code-centric ones. The results show a significant increase in the use of well-defined and formal modeling languages, especially in the initial development phases. Nevertheless, modeling tools do not seem to live up to the expectations of these new users because there is a significant decrease

Digital Object Identifier 10.1109/MS.2019.2897833
Date of publication: 16 April 2019

in participant satisfaction with the modeling tools. Specifically, the tools lacked adequate support for communication and collaboration, were complex, were inadequate for delivering executable artifacts, and had a high learning curve. Although the interest in software modeling and design and the consensus of its benefits is on the rise, the existing tools seem to be a major challenge for a more mainstream adoption. This paper appears in MODELS. Access it at http://bit.ly/PD_2019_May_2.

Architecture and Maintainability

"Abstraction Layered Architecture: Writing Maintainable Embedded Code," by Spray and Sinha,[3] describes the Abstraction Layered Architecture (ALA), a reference architecture for embedded systems. ALA is built based on more than two decades of experience designing embedded software and to improve long-term maintainability at Tru-Test, a New Zealand manufacturing company of embedded solutions. This paper reports on the impact of ALA on software maintainability when performing rearchitecting tasks that demand the addition of features. The authors investigated the role of 11 design principles for writing maintainable code and analyzed quality properties from the ISO/IEC 25010 standard. They evaluated ALA through two tasks on one of Tru-Test's products: 1) rearchitecting an embedded device for managing activities on a dairy farm and 2) adding a new feature. The evaluation results showed that the proposed approach produced more maintainable code. The programming effort for writing the functionality for 12 of the 50 classes took three months, with an estimate of one man-year to complete the functionality of the remainder of the 50 classes. This effort is much less than what would be required to develop these classes using the original conventionally written code. This paper appears in ECSA. Access it at http://bit.ly/PD_2019_May_3.

Continuous Software Engineering

"Enabling Continuous Software Engineering for Embedded Systems Architectures with Virtual Prototypes," by Antonino and colleagues, presents the II-Model, a continuous software engineering model that combines software architecture design and virtual prototypes. Continuous software engineering incorporates aspects intrinsically related to business strategy, development, and operations. Virtual prototypes correspond to executable architecture models that enable architecture simulations prior to system development. In this model, the specification and design phases run continuously with the integration and testing phases. Engineers use virtual prototypes to conduct continuous and integrated verification of different system properties at the architecture level. Companies such as Tesla, BMW, Jaguar, Land Rover, Brockwell Technologies, and Diagnostic Grifols are already using continuous software engineering. The authors envision a toolchain (potentially built on top of GitLab and Fraunhofer FERAL) enabling the automation of each step of the continuous-engineering cycle described in the II-Model.[4] This paper appears in ECSA. Access it at http://bit.ly/PD_2019_May_4.

> Continuous software engineering incorporates aspects intrinsically related to business strategy, development, and operations.

Measuring Software Architecture

"Software Architecture Measurement—Experiences From a Multinational Company" by Wu and colleagues[5] reports on the results from four years of creating and evolving an automated software architecture measurement system within Huawei. Because Huawei is always seeking to improve product quality and provide timely feature delivery to accommodate changing markets, it needed to adopt software architecture measurement practices based on the ISO/IEC 25010 quality standard. These practices help 1) support a quantitative comparison of projects and monitor architecture decay over time, 2) visualize architecture debt, and 3) demonstrate quality and productivity improvements. The authors used the Standard Architecture Index (SAI) to understand the priorities of Huawei. The SAI contains measures aimed at architectural issues related to maintainability, reliability, security, and performance. The authors used a tool called *UADP ArchGuarding* to support the measures in a six-product pilot study. The results showed an increase in product quality based on SAI 1.0. These products improved an average of 23.5%. In one product, they saw a 30% reduction in subsystem coupling from fixing more

ABOUT THE AUTHORS

JORDI CABOT is an ICREA research professor at the Universitat Oberta de Catalunya, Spain. He is a Member of the IEEE and ACM. Contact him at jordi.cabot@icrea.cat.

RAFAEL CAPILLA is an associate professor at the Rey Juan Carlos University, Spain. He is a Senior Member of the IEEE. Contact him at rafael.capilla@urjc.es.

CARLOS CARRILLO is an associate professor at the Technical University of Madrid, Spain. Contact him at carlos.carrillo@upm.es.

HENRY MUCCINI is an associate professor at the University of L'Aquila, Italy. He is the *IEEE Software* associate editor in chief and a Member of the IEEE and ACM Sigsoft. Contact him at henry.muccini@univaq.it.

BIRGIT PENZENSTADLER is an assistant professor at the California State University, Long Beach, and an adjunct professor at the Lappeenranta University of Technology, Finland. She is a Member of the IEEE and SWE. Contact her at birgit.penzenstadler@csulb.edu.

than 500 smells. In another product, they reduced coupling by 64% and reduced maintenance effort by 20%. In SAI 2.0, the authors included explicit mapping of architecture smells to quality attributes and broadened the scope of the smells. The primary benefit for software teams was more fine-grained information to improve in the architecture. As a result, the decoupling level metric showed a significant improvement in the modularity of the system, and the SAI of the subsystem improved up to 40%. The quantification of the architectural debt was key to conduct refactoring, and the studies revealed a positive impact of architecture measurement to ensure product quality. This paper appears in ECSA. Access it at http://bit.ly/PD_2019_May_5.

References

1. O. Badreddin, R. Khandoker, A. Forward, O. Masmali, and T. C. Lethbridge, "A decade of software design and modeling: A survey to uncover trends of the practice," in *Proc. 21th ACM/IEEE Int. Conf. Model Driven Engineering Languages and Systems*, Copenhagen, Denmark, 2018, pp. 245–255.
2. A. Bagnato and R. Jérôme, "Towards green metrics integration in the MEASURE platform," in *Proc. 4th Int. Workshop on Measurement and Metrics for Green and Sustainable Software Systems Collocated with Empirical Software Engineering Int. Week*, 2018, pp. 39–43.
3. J. Spray and R. Sinha, "Abstraction layered architecture: Writing maintainable embedded code," in *Proc. 12th European Conf. Software Architecture*, Madrid, Spain, 2018, pp. 131–146.
4. P. O. Antonino et al., "Enabling continuous software engineering for embedded systems architectures with virtual prototypes," in *Proc. 12th European Conf. Software Architecture*, Madrid, Spain, 2018, pp. 115–130.
5. W. Wu et al., "Software architecture measurement—Experiences from a multinational company," in *Proc. 12th European Conf. Software Architecture*, Madrid, Spain, 2018, pp. 303–319.