Requirements, Human Values, and the Development Technology Landscape

Jeffrey C. Carver, Leandro L. Minku, and Birgit Penzenstadler

THIS ISSUE’S COLUMN reports on papers from the 24th International Requirements Engineering Conference, 38th International Conference on Software Engineering, and 10th International Symposium on Empirical Software Engineering and Measurement. Feedback or suggestions are welcome. In addition, if you try or adopt any of the practices mentioned in the column, please send Jeffrey Carver and the paper authors a note about your experiences.

Toward Complete Requirements
“Challenging Incompleteness of Performance Requirements by Sentence Patterns,” by Jonas Eckhardt and his colleagues, describes a method to address a key requirements-engineering challenge—preventing incomplete requirements. To systematically address that challenge, Eckhardt and his colleagues use templates—for example, sentence patterns (see Figure 1). They focus on performance requirements.

On the basis of a study of requirements of 11 projects from five companies in various domains, Eckhardt and his colleagues developed a sentence template for performance requirements. They found that 86 percent of the requirements studied could be successfully translated into their framework. The requirements that couldn’t be translated were too vague. When they considered how well the requirements would drive system testing, the results showed that only 18 percent of the requirements included all the information necessary and that 32 percent required substantial interpretation. The remaining 68 percent were incomplete.

Requirements engineers can take the framework and sentence templates directly from the paper and apply them to elicit new performance requirements or analyze existing ones. Eckhardt and his colleagues are willing to provide assistance and answer any questions to facilitate their framework’s use. You can access this paper at goo.gl/H8Ngk9.

The Perils of Social Engineering
“A Serious Game for Eliciting Security Requirements,” by Kristian Beckers and Sebastian Pape, describes a game that helps employers teach employees about a vital issue related to company security—social-engineering threats. Social engineering is a nontechnical strategy to trick people into breaking security practices. The success of these techniques—
for example, phishing—depends on how well a cyberattacker can manipulate his or her victims into providing confidential information. Beckers and Pape’s game uses cards to educate players about social-engineering concepts and the attacks that can be constructed by applying social-engineering principles. To evaluate the game’s value, they tested it at two universities, with 27 university employees. Most participants held master’s degrees; a few held PhDs. They played the game for about two hours to learn about the security threats and then develop scenarios of the types of attacks likely in their work context. The participants identified a significant number of feasible scenarios (42 of the 49 proposed scenarios). Feedback surveys indicated that the participants had a positive sentiment about the experience.

To evaluate the game’s value, they tested it at two universities, with 27 university employees. Most participants held master’s degrees; a few held PhDs. They played the game for about two hours to learn about the security threats and then develop scenarios of the types of attacks likely in their work context. The participants identified a significant number of feasible scenarios (42 of the 49 proposed scenarios). Feedback surveys indicated that the participants had a positive sentiment about the experience.

The game is for use by chief security officers or IT administrators who are in charge of security, who can take the developed scenarios as input for eliciting security requirements. Beckers and Pape seek additional industry partners who are interested in evaluating the game in their settings. They’ll offer a free security analysis as part of the evaluation. You can access this paper at goo.gl/7xC33r.

Injecting Human Values into Software Engineering

“Values-First SE: Research Principles in Practice,” by Maria Angela Ferrario and her colleagues, shows that
embedding the principles of human-values research into software engineering decision making improves user-centered design decisions. The paper makes the interplay between values and software engineering choices visible to help reveal a system’s complex impact on politics, society, and the environment.

To illustrate this concept, the authors explain how using values helped inform design choices for a smart-health reporting device. During the project review, a value map guided the drafting of the project mission statement and teamwork principles. The mission statement was, “The driving values of this project are quintessentially human: respect, trust, empowerment, and mutual understanding. These values are, and must be, the drivers of our research and development process.”

The teamwork principles are included in the project documentation and embedded in day-to-day development practices. These practices include sharing the same office space and research tasks. For example, developers take part in ethnographic research while others familiarize themselves with the software code and hardware functionality. Supporting team building by engaging in cross-cutting tasks is a Speedplay practice and ties in with the value principle of helpfulness or mutual help.

Software can help with a broader adoption of value-conscious approaches. Software engineering can help by reusing and repurposing values-sensitive software, embedding values practices from other disciplines, and engaging with society to address current needs instead of creating new ones.

The practical takeaway is that value-first software engineering significantly improves the design choices for human-centered design. Consequently, software engineers will benefit from identifying the underlying values that are important to their stakeholders, which will result in improved overall design decisions and increased system acceptance. You can access this paper at goo.gl/EjhiwV.

**Mapping the Development Technology Landscape**

“Mining Technology Landscape from Stack Overflow,” by Chunyaang Chen and Zhenchang Xing, describes TechLand, a tool that automatically mines Stack Overflow question tags. TechLand aims to help developers determine the current development technology landscape. Knowledge of current trends in development technology is useful not only for professional development but also for choosing the best technology for a software project. However, the related information on the web typically is either outdated or too fragmented to be easily consolidated into the big picture.

TechLand uses association rule mining, community detection, and natural-language processing to create a graph of the development technology landscape. In the five months after TechLand was advertised on several programming-related websites, more than 1,000 users accessed it, with close to 300 using it in a meaningful way.

TechLand is available at www.graphofknowledge.appspot.com. You can access this paper at goo.gl/bz7p04.

**References**