From Storyboard to Videogame: A Content-Centric Development Process Model

pp. 24-30

Pablo Moreno-Ger, Iván Martínez-Ortiz, José Luis Sierra, and Baltasar Fernández-Manjón

Computer and videogames have a huge potential to facilitate learning because they easily capture students' attention. However, we can't just throw some math into a game and call it educational, nor can we call it entertainment when a talking animal gives the lesson while the student solves silly puzzles.

In the words of literature professor Henry Jenkins, we need to "move beyond the current state of edutainment products which combine the entertainment value of a bad lecture with the educational value of a bad game." Point-and-click adventure games like the *Monkey Island* or *King's Quest* sagas have all the ingredients needed to achieve this balance between content delivery and entertainment.

Making a Difference in the Software Century

pp. 32-38

Barry Boehm

In the 21st century software will be the main element that drives our necessary capabilities and quality of life. This will be very satisfying for software engineers, but it also will impose large responsibilities to provide excellence in software-intensive systems and the services they provide.

Gone are the days when software engineers could spend their entire careers doing the same thing in the same ways. The most significant challenges facing 21st-century organizations will likely be their ability to adapt to rapid and unpredictable change more quickly and appropriately than their competitors.

Determining the Impact of Software Engineering Research on Practice

pp. 39-49

Leon J. Osterweil, Carlo Ghezzi, Jeff Kramer, and Alexander L. Wolf

The enormous changes in software engineering practice make it prudent to consider the interplay between software engineering research and practice. Toward that end, the authors provide an overall view of the motivations behind the Impact Project, the research methodology followed, and the project's development plan. They also explore more specific questions that separate study groups within the project have investigated.

This research aims to determine what impact, if any, software engineering research has had upon practice and to substantiate any such impact with scholarly research appropriate to a scientific community.

The ASC-Alliance Projects: A Case Study of Large-Scale Parallel Scientific Code Development

pp. 50-58

Lorin Hochstein and Victor R. Basili

Computational scientists use computers to simulate physical phenomena in situations where experimentation would be prohibitively expensive or impossible. Advancing scientific research depends on developing software productively, which often must run on high-end computing systems, presenting unique software development challenges.

The authors studied five large projects that develop software for HEC systems. These projects face unique challenges both because of the problem's nature and the difficulties associated with programming the environment.

SSL/TLS Session-Aware User Authentication

pp. 59-65

Rolf Oppliger, Ralf Hause, and David Basin

Today's e-commerce applications often employ the Secure Sockets Layer or Transport Layer Security protocols to authenticate the server and cryptographically protect communications between client and server. While developers consider these protocols sound and secure, most SSL/TLS-based e-commerce applications employing user authentication at the application layer are vulnerable to phishing, Web spoofing, and, most importantly, man-in-the-middle attacks.

SSL/TLS session-aware user authentication has been proposed as a technological approach that provides a lightweight alternative to deploying and using public-key certificates on the client side.

Model Predictive Feedback Control for QoS Assurance in Web Servers

pp. 66-72

Cheng-Zhong Xu, Bojin Liu, and Jianbin Wei

In contrast to today's best-effort service model, the QoS differentiation-and-assurance architecture creates an opportunity to implement a differentiated pricing structure in the next generation of Internet services. By downgrading the quality of their requests, the architecture controls aggressive clients' behaviors and ensures fair sharing between clients, proxies, and even traffic from different domains.

Fairness assurance automatically builds a firewall against aggressive clients and protects the server from flash-crowd-like distributed denial-of-service attacks. The authors' *e*QoS framework represents a first step in a systematic approach to the provisioning of user-perceived page-view response-time differentiation and assurance.