Postprint of article in *Information and Software Technology* 46 (2004) 987–988

Editorial

1. Background

Software is playing an increasingly important role in our day-to-day life. However, software today — unlike automobiles, bridges or office towers — is produced without the benefit of established standards. It is well known that there are still unresolved errors in many of the software systems that we are using every day. The aim of the quality software conference series is to provide a forum to bring together researchers and practitioners working on improving the quality of software, to present new results and exchange ideas in this challenging area.

In 2003, the conference evolved from an Asian-Pacific based regional event to an International Conference on Quality Software (QSIC)[1]. The response to the call for papers was overwhelming: a total of 180 submissions from 28 countries were received. Among them, 40 regular papers, nine experience reports and five short papers were selected by the Program Committee for presentation at the conference held on November 6–7, 2003 in Dallas, Texas, USA.

After the conference, authors of seven regular papers were invited to submit revised and expanded versions for a special issue of *Information and Software Technology*. After a further reviewing process, five of them were accepted for publication in this special issue.

2. The papers

Reliability is an important aspect of software quality. How to assess software reliability remains a challenging problem for software developers, and is a hot topic for research. The paper entitled "How to Test for Optimal Software Reliability Assessment" by Cai, Li and Liu reports on an innovative approach based on controlled Markov chains. It is argued that, under the assumption that the code of the software under test is frozen and the number of tests is given a priori, software reliability assessment can be treated as a control problem. To minimize the resulting variance of software reliability estimate, the authors propose an adaptive software testing approach. Simulation data show that the adaptive approach works much better than the random test strategy, as the resulting variance from the former is much smaller than that of the latter.

As the most widely used software validation method, testing is a popular topic of the quality software conference. The paper "Mirror Adaptive Random Testing" by T.Y. Chen et al. proposes a new approach to random testing that is extended from adaptive random testing. It is achieved by choosing random test points in one subdomain and then mirroring them in other subdomains by simple translation functions, thus avoiding a large number of "distance" calculations that are necessary to ensure a good separation of test cases but can be expensive computationally. A thorough simulation study has been undertaken to determine the efficacy of the approach.

Program verification by theorem proving has a long history of almost 40 years, but has so far been less successful than other validation methods such as testing and model checking. To make the technique more applicable, the paper "Verifying Haskell Programs by Combining Testing, Model Checking and Theorem Proving" by Dybjer, Qiao and Takeyama pursues an integrated methodology. Agda/Alfa, an interactive proof assistant for dependent type theory, has been extended to include a random testing tool and a BDD-based model checker. Two non-trivial Haskell programs are validated using the combined approach, demonstrating how a good balance between the confidence in program correctness and the cost of verification can be achieved.

As model-driven development of .NET software has become popular, a tool supporting the validation of scenarios is invaluable to software developers. The paper "Instrumenting Scenarios in a Model-Driven Development Environment" by Grieskamp, Tillmann and Veanes presents such a tool — SpecExplorer. Scenarios are described in a modelling language called Spec#, which is similar in appearance to C#. A real application, CTAS Weather Control Logic, is used as a case study to illustrate several features of the tool: random execution, finite state machine generation and scenario conformance checking.

The lack of commonly accepted metrics for quality in software industry makes it difficult for users to compare software quality across products. The paper "The Software Evaluation Framework 'SEF' Extended" by Wong addresses this issue, reporting a quantitative study on stakeholders' understanding of software quality. The metrics used in the Software Evaluation Framework are described, and an analysis is made of how these metrics compare in the measurements of software characteristics. The model is empirically tested with quantitative techniques.

[1] The new acronym reminds us of the high quality and strength of Cu-SiC, a ceramic reinforced metal.

3. Looking ahead

After the success of three quality software conferences, the Fourth International Conference on Quality Software (QSIC 2004) is held in Braunschweig, Germany on September 8–9, 2004. Hans-Dieter Ehrich, Technical University of Braunschweig, Germany, and Klaus-Dieter Schewe, Massey University, New Zealand, serve as program co-chairs. Please visit http://qsic2004.tu-bs.de for more information.

Huimin Lin*
Laboratory for Computer Science
Institute of Software, Chinese Academy of Sciences
Beijing 100080, China
E-mail address: lhm@ios.ac.cn

Hans-Dieter Ehrich
Institute of Information Systems
Technical University of Braunschweig
Braunschweig, Germany
E-mail address: hd.ehrich@tu-bs.de

T.H. Tse
Department of Computer Science
The University of Hong Kong
Pokfulam, Hong Kong
E-mail address: thtse@cs.hku.hk

* Corresponding author.