Postprint of article in *International Journal of Software Engineering and Knowledge Engineering*
Vol. 11, No. 2 (2001) 1–4

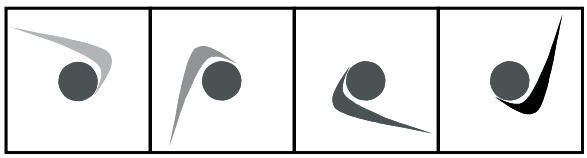
GUEST EDITOR'S INTRODUCTION

T. H. TSE
Department of Computer Science
The University of Hong Kong
Pokfulam, Hong Kong
Email: thtse@cs.hku.hk

About APAQS 2000

The quality of software has an important bearing on the financial and safety aspects of our daily lives. Unfortunately, software systems often fail to deliver according to promises. It is well known that there are still unresolved errors in many of the software systems that we are using every day. An Asia-Pacific Conference on Quality Software was held in Hong Kong in October 30–31, 2000, with a view to providing a forum to bring together researchers and practitioners from within and outside the Asia-Pacific region to seriously address this issue.

The acronym of the conference has chosen to be *APAQS*. According to the Oxford Dictionary, "apex" means the top or peak of a mountain or the highest or culminating point of time. The conference logo shows the stylized letters "a", "P", "Q" and "S". These letters are formed by ticks, the usual symbol for quality assurance. The logo also symbolizes the movement towards quality and integration.



We have received a total of 66 submissions from over twenty countries. Both the number of submissions and their quality were quite impressive for a new con-

ference. After a process of thorough peer reviews, 33 papers were selected. They covered a wide range of topics in quality software, including Web-based systems (2 papers), distributed, concurrent and real-time systems (3 papers), metrics and models (3 papers), software testing (9 papers), object-oriented models and design (3 papers), software quality assurance (3 papers), formal methods (3 papers), industrial experience (3 papers), prototyping (2 papers) and electronic commerce (2 papers). There were a total of 70 authors for the accepted papers, out of whom 37 were from the Asia-Pacific and 33 from outside the region. We also invited Professor Stephen Yau, Professor and Chair of Computer Science and Engineering Department, Arizona State University, and Mr K. H. Lau, Director of Information Technology Services of the Government of the Hong Kong Special Administrative Region, to be the keynote speakers. The conference proceedings were published by IEEE Computer Society Press.

About This Special Issue

We note that the software crisis is not restricted to the Asia-Pacific community. A Special Issue on Quality Software is therefore arranged in the International Journal of Software Engineering and Knowledge Engineering. The four articles have been selected for this issue after rigorous peer review.

Software testing was the most popular topic in the APAQS conference. It is also the most widely used method in the pursuit of software quality. It is very natural, therefore, that three papers in this special issue are related to software testing.

(a) Software testing techniques are usually classified into two categories: The black-box approach refers to testing based on software specifications, whereas the white-box approach refers to testing based on the source code of the software in question. Partition testing is one of the most important and practical techniques in the black-box approach. The input domain is divided into smaller subsets, known as subdomains. In theory, test data are allocated to each subdomain with a view to detecting possible failures. In practice, however, given a complex specification, the number of resulting subdomains may be very large and hence, given budgetary and time constraints, it may not be viable to allocate test cases to every subdomain.

In the paper entitled *A Study on a Path-Based Strategy for Selecting Black-Box Generated Test Cases* by Y. T. Yu, P. L. Poon, S. F. Tang and T. Y. Chen, the authors propose to solve this problem by supplementing the black-box approach by the white-box approach. Based on new concepts such as matching pairs and C_W-equivalence, they propose to produce an initial test suite using the classification tree methodology, which is a black box approach, and to select a proper subset of this test suite using white-box information. The proposed process is supported by algorithms. A case study illustrates that the integrated approach is effective and practical.

(b) In general, software testing methods attempt to ensure that test cases cover all components of a piece of software. In the white-box approach, the coverage criteria are based on the source code. Two of the most popular coverage methods are (i) branch and path coverage, which are based on the program control flows, and (ii) data-flow coverage, which are based on the relationships between the definition and use of program variables. Path coverage technique is addressed by the third paper and an enhanced data-flow technique is addressed by the fourth paper.

In the paper entitled *A Constraint Solver and its Application to Path Feasibility Analysis* by Jian Zhang and Xiaoxu Wang, the authors are concerned that some paths may not be feasible, and hence software testers may waste their time trying to make futile attempts to execute such paths. They tackle the problem of path feasibility by the use of a constraint solver, based on such constraint-based techniques as linear programming and Boolean satisfiability. They propose tools that are automatic and with a natural user-interface. Their approach compares favourably with other constraint-based techniques and tools.

(c) Internet applications are gaining huge momentum these days. The testing of Web-based systems, however, cannot be achieved via conventional techniques. Interactions between system components are no longer in the form of defined control procedures or function calls between two components. Data are stored in HTML documents and passed between clients and servers using the HTTP protocol. It is, therefore, insufficient to test Web-based software by means of conventional control flows or data flows.

In the paper entitled *An Object-Based Data Flow Testing Approach for Web Applications* by Chien-Hung Liu, David C. Kung, Pei Hsia and Chih-Tung Hsu, the authors apply a new data-flow technique for testing such applications. Different types of control flow graphs are used to model the data flow structure. From the intra-object, inter-object and inter-client perspectives, five levels of data-flow testing, including functional, function-cluster, object, object-cluster and application levels, are covered. The testing is therefore very comprehensive.

(d) Although software testing has remained the most popular method for quality assurance, formal verification should be applied for situations where it is safety-critical, or where the cost of errors would be enormous. Reactive systems, which are real-time systems that continuously react to stimuli from the environment, often fall into the former category.

In the paper entitled *Component-Based Verification in a Synchronous Setting* by Agathe Merceron and G. Michele Pinna, the authors propose to verify the refinement of component-based software. The properties of the refining components can be verified by means of model checking and observers. Although model checking in general may not be feasible when there are a large number of states, it can nevertheless be applied to software components, which are usually of reasonable sizes. A case study on protocols shows that the theoretical method can be applied in a

practical setting.

Looking Ahead

Following on the success of the first meeting, the second APAQS conference is scheduled on December 10–11, 2001 in Hong Kong. Professor C. V. Ramamoorthy of the University of California at Berkeley and Dr Ray Paul of the Office of the Assistant Secretary of Defense, USA, have agreed to be the keynote speakers. Please refer to `http://www.cs.cityu.edu.hk/%7Eapaqs` for more details.