# AN ENHANCEMENT OF SLICING TEST ALGORITHM FOR INTEGRATION TESTING OF EMBEDDED SYSTEM

AHMED SHEIKH ABDULLAHI MADEY

A dissertation report submitted in partial fulfillment of the
requirements for the award of the degree of
Master of Science (Computer Science)

Faculty of Computing
Universiti Teknologi Malaysia

MAY 2014

This dissertation is dedicated especially to my beloved parents and also not forgetting my beloved brothers and sisters for their endless supports and encouragements.

## ACKNOWLEDGEMENT

In preparing this dissertation report, I wish to express my sincere appreciation to my supervisor Assoc.Prof. Dr. Dayang Norhayati Binti Abang Jawawi for the guidance, advice and encouragement during my studying. The support and suggestion that Assoc.Prof. Dr. Dayang gives inspired me to going through in this dissertation.

Finally my special thanks to my parents for their love and care for their support and cheering me up at those difficult time.

# ABSTRACT

The complexity of testing the software of Component Based Software Development (CBD) for Embedded Real Time (ERT) software development highlight the challenges of designing, analyzing and testing ERT software. From this standpoint, the complexities of CBD for ERT in software testing require suitable software algorithms. Against these claims, a number of software testing algorithms have been formulated such as slicing algorithm, incremental algorithm, firewall algorithm, genetic algorithm as well as simulated annealing algorithm. Generally, not all of these algorithms support CBD and ERT software testing of the system. By applying slicing algorithm into ERT software testing, the complexity of ERT software development can be decreased and at the same time promote high degree of reuse through software testing based on component behavior. Currently, testing algorithm based on slicing does not directly support ERT software. In this research, the integration testing algorithm for CBD and ERT system has been proposed to represent a promising way to test ERT software in terms of algorithm refinement. The slicing algorithm called slicing architectures using service edges (SASE) has been enhanced to support a component oriented programming (COP) framework for CBD and ERT integrated system. The results shows that COP framework can be applied into SASE algorithm definitions and it has been mapped with the SASE algorithm based on the similarities and differences definitions. Thus, the quality of the enhanced SASE algorithm is better in terms of algorithm criteria based on Normative Information Model-based Systems Analysis and Design (NIMSAD) evaluation in support of ERT and CBD.

# ABSTRAK

Kerumitan yang terdapat semasa menguji perisian Komponen Berdasarkan Pembangunan Perisian (CBD) untuk Masa Nyata Terbenam (ERT) menunjukkan cabaran-cabaran dalam mereka-bentuk, menganalisa, dan menguji perisian ERT. Dari pandangan ini, kerumitan yang terdapat pada CBD untuk ERT memerlukan algoritma perisian yang sesuai. Pada tuntutan ini, beberapa algoritma ujian perisian telah dirumuskan seperti algoritma penghirisan, algoritma penambahan, algoritma firewall, algoritma genetik dan juga algoritma penyepuhlindapan. Pada amnya, tidak semua algoritma-algoritma ini menyokong sistem ujian perisian CBD dan ERT. Dengan mengaplikasikan algoritma penghirisan ke dalam ujian perisian ERT, kerumitan pada pembangunan perisian ERT boleh dikurangkan dan pada masa yang sama menggalakkan penggunaan semula pada tahap yang tinggi menerusi ujian perisian berasaskan perilaku komponen. Pada masa kini, algoritma ujian berasaskan penghirisan tidak menyokong perisian ERT secara langsung. Di dalam kajian ini, integrasi ujian algoritma untuk sistem CBD dan ERT telah dicadangkan untuk menunjukkan cara yang lebih berpotensi untuk menguji perisian ERT di dalam istilah penghalusan algoritma. Algoritma penghirisan telah ditingkatkan untuk menyokong rangka kerja pengaturcaraan berorientasikan komponen (COP) untuk sistem integrasi CBD dan ERT. Hasil menunjukkan bahawa COP boleh digunakan di dalam definisi algoritma SASE dan telah dipetakan dengan SASE algoritma berasaskan definisi persamaan dan perbezaan. Oleh itu, kualiti algoritma SASE yang ditingkatkan adalah lebih baik dari segi algoritma berasaskan kriteria dan Normatif Maklumat Sistem Analisis dan Reka bentuk berasaskan Model (NIMSAD) untuk menyokong ERT dan CBD.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

## 1.1 Overview

Software testing is a verification process in which an application of the software or the program meets the business requirements and technology that have dominated the design, development and works as expected. Testing of the software also identifies significant gaps and errors in the application code which must be corrected and fixed.

To fix errors is important because errors are classified according to severity. Testing requires planning, and during test planning we are able to decide which errors are important and the reasons for failure based on the requirements and documents of the design.

Generally, the defect is only important from the customer viewpoint because it affects the ease of use and operation of the application (Eun, 2009). In software testing there are three main objectives including verification, validation and fault finding as shown in Figure 1.1.

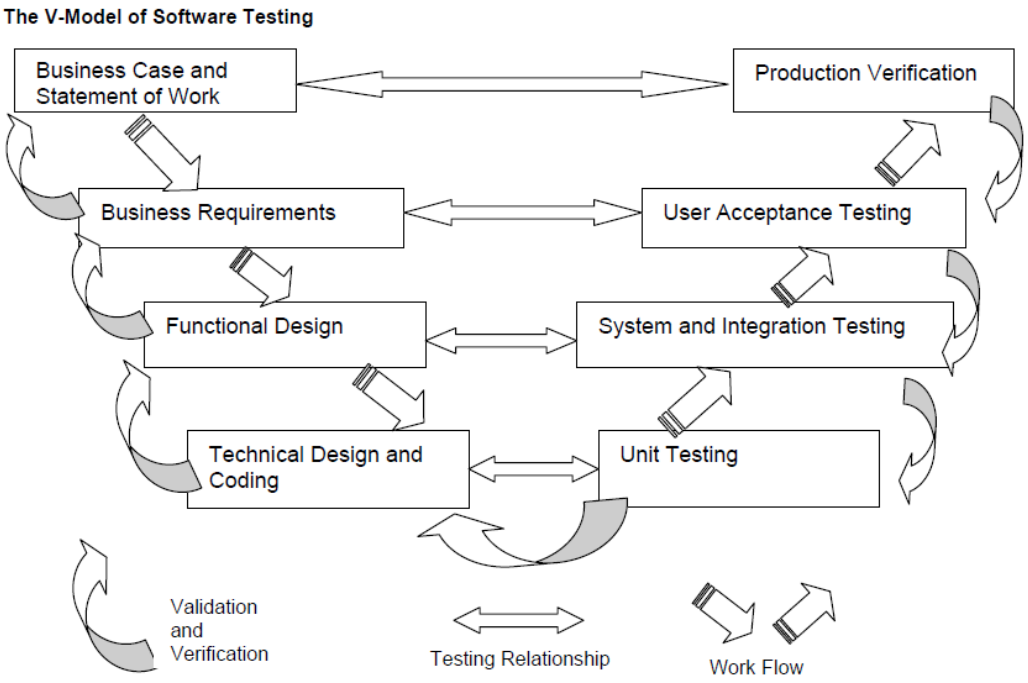


**Figure 1.1** Software Testing Objectives (Xia et al., 2000)

Following the software testing objectives shown in Figure 1.1, verification process confirms that the program meets the technical specification of the test. Furthermore, the validation process proves that the software or program meets the needs of the business. Fault finding is the difference between the real and the prospective result which can be traced from the source of the imbalance to an error in the design view and the development stages (coding).

Basically, an algorithm is as set of steps to solve a certain problem, and from this definition we can extract that test algorithm defines the testing procedures to achieve test objectives. The test algorithm therefore determines costs and effort of testing. Selecting a suitable test algorithm is one of the most significant planning tasks or decision the tester has.

The aim is to decide which test approach optimizes the relation between the costs of defects, the cost of testing as well as risk minimizing.

Testing algorithms state how the stakeholders' product risks are reduced at test level, which kinds of tests are to be performed, which exit and entry may be appropriate. Testing algorithms are established and created based on test improvement. System test is typically used and theoretical testing algorithms may be referred to. A testing algorithm indicates how the software functionality is to be a developed and settled in future release. For every stage of testing, an equivalent test algorithm should be created to test the new feature groups or sets.

Current systems of software are more complex and difficult to control, consequential in high cost of development, large scale, low production, uncontrollable quality of software and the additional risk of more new technology to change it.

On the other hand, Component Based Software Development (CBD) is able to meaningfully lessen test cost improvement, market time estimation and increase maintainability, reliability and generally software systems value. This attitude has generated a great amount of interest in both community research as well as in software trade, which is the focus of CBD.

Components of existing technologies have been implemented in various systems of software like: embedded system software, application based web projects and also object oriented circulated software component.

Embedded Real Time (ERT) is a system that combines software and hardware. The hardware component of the system exhorts the constraints of time while the software reduces the total cost and test flexibility. ERT software testing is crucial because of the combination of both hardware and software which can lead to complexity. So, all these events require a systematic process to reduce the time and cost of testing (Sabil and Jawawi, 2009).

Embedded systems are often used for many years and remain an integral part of system support. An integrated system is much larger than the speed of the tester of the original program and it is necessary to explain some good citations on integrated software systems especially since the source may affect its own value based on test performance ( Petricic, 2011).

Embedded systems are often limited tools of the test although some plans have to support a variety of programs and tools for software testing. Embedded software testing system is more limited and often used tools of basic research, and this is partly because embedded systems in many cases cannot be used as support tool as many integrated systems have their own tools for debugging and testing to reduce the number of tools for use within house tool.

With huge numbers of embedded software in application fields with great dependability and safety, embedded software testing is progressively facing various problems or challenges and attention must be paid to embedded software properties while conducting researches on new test software algorithms. Although research on software testing algorithms is detailed enough, there are still remarks on the software testing quality evaluation. Software testing assures the quality of software enabling people to go about their work with the knowledge that the software, which has been tested, will operate correctly.

## 1.2 Problem Background

Testing algorithms include slicing algorithm, incremental algorithm, firewall algorithm, genetic algorithm as well as simulated annealing algorithm. All these algorithms can be used in software testing but testing algorithms such as slicing and Genetic are used for CBD and ERT.

Slicing (Lalchandani, 2008): slicing explicitly traces the modification of the program or software which consists of all statements and segments of the software that may be infected. Applying slicing in software architecture in software testing can benefit in two main ways: the first concerns the maintenance of component based software by using slicing tools on an architectural description. In addition we can determine which components might be affected. Second, architectural reuse can be facilitated while code reuse is very important and reusing the software design is expected in software testing life cycle.

Genetic (Li and Chen, 2006): This is to extract two input statements into a new testing execution by inheriting the items from software parts to execute it. Software testing is the important instrument of assuring software quality and software testing includes the software life cycle. Genetic algorithms (Bing and ZiLi, 2006) are stochastic algorithms that use adaptive search methods for solving problems. They are very useful in optimization and problems of complex search. These kinds of algorithms are based on natural evolution and on Darwinian natural selection and the GA combines with automatic generation of testing case in ERT systems. The goal is to uncover as many faults as possible with a potent set of tests. Well tested data can cover complex designation paths of embedded software so desirable testing data is not easy to find.

Incremental (Baradhi, 1997): the test cases are selected from the outputs and this algorithm involves only the executed test by considering the statements of the program. Firewall: assists the testing integration by the tester. Simulated Annealing (Baradhi, 1997): This algorithm suggests the candidate solution that is represented by the testing execution and to be minimized by the function cost. These algorithms can be used for components but no proven studies have been found.

Several algorithms are possible to use for testing CBD software such as Slicing algorithm. The slicing algorithm indicates cases of test cross levels recognizing various levels of practical concepts for the test levels (Hao and Jiang, 2011). More useful particulars must be tested at minor levels of test than at advanced

ones. This changeability across test stages must bear in mind reuse approaches. Less test work provided by limited refinement of slicing as long as the improvement or generalization operations need to be performed substantially at each level of test.

Thus, slicing algorithm improvement is a straight forward algorithm that states useful cases of test alterations at levels of test for finding test cases to levels of minor test where the modification based on slicing algorithm plays a role from test phase to requirements to implementation.

### 1.2.1 CBD Software Integration Testing

One main problem in CBD integration testing is the instruction in which tested component are listed and this test order is referred to as component test order for several reasons. First, this test order concerns the order of tested components. Second, test order of component influences test component use and test case preparation. Third, test order of component decides the order in which faults of component are identified. Another significant problem when integration testing of CBD software is to make a decision concerning the component order integration. A number of studies have determined algorithms for integration test order from dependencies among components in the system component illustration.

The aim of all these test order steps is to minimize test step numbers to be shaped as this is supposed to be a main rate factor for integration testing. Indeed, steps are parts of software that have to be constructed in order to test software parts that are either not developed yet or have not yet been unit tested but want to test components that depend on them (Briand et al., 2003).

Slicing testing algorithm is a very important field research in software engineering and has been used in many applications such as maintenance of

software, software understanding, software analysis, inverse engineering, testing as well as debugging (Hao and Jiang, 2011). The method of CBD slicing has been investigated in many theses since the original definition by Mark Weiser in 1979. Weiser first proposed the idea of software slicing, and defined slicing of software as follows: software slicing is an executable part in terms of interest point variables and the executable part of software corresponding to the software in use.

Static software slice (Jia et al., 2010) concerns all sentences in a software that are related to the variable at the interest point. It analyses all possible software running tracks so it should simply contain unconnected points with greater idleness. Dynamic slicing (Jia et al., 2010) is established by all sentences which touch the variable at the interest sentence point in executable software path.

On the other hand, the component oriented programming (COP) context was initially planned for ERT software development for autonomous mobile robot (AMR). The target audience is the researchers in fields of mechatronics and robotics which are not from background of software engineering and do not have wide programming knowledge.

The framework of COP is a programming framework based on Pervasive Component Systems (PECOS) model. The proposed framework enables the idea in PECOS to be implemented optimally without requiring any support tools and proprietary runtime environment from the original PECOS project (Jawawi, 2007). Thus the COP used for requirement analysis, design and implementation are not included in the field of software testing.

## 1.3 Problem Statement

Slicing Testing Algorithm is to support thorough CBD software integration. Other algorithms are not well defined to be used in any COP framework because the other algorithms are not directly supported through CBD but some of the algorithms support ERT systems. Slicing algorithm is used to slice the program instruction and only contains the program statements but the slicing architectures using service edges (SASE) are used both in software architecture and software testing fields.

This study shows that slicing algorithm is much more closely related to the CBD for COP because it can support most COP elements in terms of software analysis and design. Hence, the main motivation of this work is to propose an enhanced Slicing Architecture using Service Edges (SASE) in COP framework to show application of the software components of an ERT system. Slicing algorithm is a commonly recognized technique for analyzing and testing CBD software to address test order of component problem and to recognize algorithm connected components.

## 1.4 Research Aim

The aim of this research is to propose testing algorithm to test CBD software for ERT system based on SASE testing algorithm which can support COP frameworks.

## 1.5 Objective of the Research

The aim of this research is supported by the following objectives:

i. To analyze the current testing algorithms and to study the test algorithm for component based software development of embedded systems.
ii. To propose the enhancement of slicing test algorithm of CBD for COP frameworks using in ERT system.
iii. To evaluate and validate the enhanced slicing test algorithm by comparing it with the original slicing test algorithm.

## 1.6 Scope of the Research

The scope of this research has been limited to the following:

i. This research focuses only on CBD testing algorithm for embedded real time system case study, and does not include other applications.
ii. This research applies CBD to verify the testing algorithm for Embedded Software Development.

## 1.7 Significance of the Research

The significance of this study is to promote the testing algorithm of CBD for embedded software development. The study concentrates on an in-depth understanding of CBD for ERT testing algorithm. Based on that knowledge, the advantages that can be derived from this study are to motivate the use of algorithms

based on algorithm much more closely related to CBD for ERT in the software testing.

## 1.8 Thesis organization

Chapter 2 discusses CBD algorithms for ERT among testing algorithms. In Chapter 3, the research methodology is conducted in achieving the research objectives and scopes. One case study is used involving ERT system. Chapter 4 discusses the results of the enhanced algorithm using COP frameworks. Chapter 5 discusses the validation phase. Finally, in Chapter 6, this research was concluded based on its objectives and future work was proposed.

## REFERENCES


Alkadi, I. S. & Alkadi, G. S. 2001, Algorithms that compute test drivers in object oriented testing. Aerospace Conference, IEEE Proceedings, 2001

Baradhi, G. & Mansour, N. 1997, A comparative study of five regression testing algorithms. Software Engineering Conference, 1997. Proceedings. 1997 Australian,. IEEE, 174-182.

Bo, Z. & Xiangheng, S. 2011, The effectiveness of real-time embedded software testing. Reliability, Maintainability and Safety (ICRMS), 2011 9th International Conference on, 2011. 661-664.

Bräunl, T. & Tay, N. 2001. Combining configuration space and occupancy grid for robot navigation. *Industrial Robot: An International Journal,* 28**,** 233-241.

Briand, L. C., Labiche, Y. & Yihong, W. 2003. An investigation of graph-based class integration test order strategies. *Software Engineering, IEEE Transactions on,* 29**,** 594-607.

Bing, L. & Zili, C. 2006, Pivotal techniques of embedded software testing case generation by Genetic algorithms. Computer-Aided Industrial Design and Conceptual Design, CAIDCD'06. 7th International Conference on,. IEEE, 1-5.

Bonail, B., Abascal, J. & Gardeazabal, L, 2009, Wheelchair-Based Open Robotic Platform And Its Performance Within The Ambiennet Project. Proceedings Of The 2nd International Conference On Pervasive Technologies Related To Assistive Environments, Acm, 63.

Castilho, O. & Trujilo, L. 2005. Multiple Objective Optimization Genetic Algorithms For Path Planning In Autonomous Mobile Robots. Int. J. Comput. Syst. Signal, 6**,** 48-63.

Cheein, F. A. A., De La Cruz, C., Bastos, T. F. & Carelli, R. 2009. Slam-Based Cross-A-Door Solution Approach For A Robotic Wheelchair. *International Journal Of Advanced Robotic Systems,* 6**,** 239-248.

El ariss, O., Dianxiang, X., Dandey, S., Vender, B., Mcclean, P. & Slator, B. 2010, A Systematic Capture and Replay Strategy for Testing Complex GUI Based Java Applications. Information Technology: New Generations (ITNG), 2010 Seventh International Conference on, 2010. 1038-1043.

Eun, J. 2009, A Test Process Improvement Model for Embedded Software Developments. Quality Software, 2009. QSIC '09. 9th International Conference on, 2009. 432-437.

Ha, X, Nsel, J., Rose, D., Herber, P. & Glesner, S. 2011, An Evolutionary Algorithm for the Generation of Timed Test Traces for Embedded Real-Time Systems. Software Testing, Verification and Validation (ICST), IEEE Fourth International Conference on, 2011. 170-179.

Hao, J. & Jiang, S.-J. 2011, An approach of slicing for Object-Oriented language with exception handling. Mechatronic Science, Electric Engineering and Computer (MEC), 2011 International Conference on, 2011. 883-886.

Iyenghar, P. 2011, Test Framework Generation for Model-Based Testing in Embedded Systems. Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on, 2011. 267-274.

Iyenghar, P., Pulvermueller, E., Westerkamp, C. & Wuebbelmann, J. 2011, Integrated model-based approach and test framework for embedded systems. Specification and Design Languages (FDL), Forum on, 2011. 1-8

Jia, L., Jiao, H. & Liu, J. 2010, A dynamic program slice algorithm based on simplified dependence. Advanced Computer Theory and Engineering (ICACTE), 3rd International Conference on, 2010. V4-356-V4-359.

Jia Limin, Jiaohongqiang & Liu Jie, 2010. A Dynamic Program Slice Algorithm Based on Simplified Dependence. 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE).

Jawawi, D. N., Mamat, R. & Deris, S. 2007. A Component-Oriented Programming for Embedded Mobile Robot Software. *International Journal of Advanced Robotic Systems,* 4.

Jawawi D. N. A., Suzila Sabil, Rosbi Mamat, Mohd Zulkifli Mohd Zaki, Mahmood Aghajani Siroos Talab, Radziah Mohamad, Norazian M. Hamdan & Khadijah

Kamal, 2011, “A Robotic Wheelchair Component-Based Software Development”, Book Chapter for Mobile Robots / Book 2, , Intech Open Access Publisher. ISBN 978-953-307-842-7, pp. 102-126.

Kuo-chung, T. & Daniels, F. J. 1997, Test order for inter-class integration testing of object-oriented software. Computer Software and Applications Conference, 1997. COMPSAC '97. Proceedings., The Twenty-First Annual International, 1997. 602-607.

Koskinen, J., Lintinen, H., Sivula, H. & Tilus, T. 2004. Evaluation of software modernization estimation methods using NIMSAD meta framework. *Publications of the Information Technology Research Institute,* 15.

Labiche, Y. 2011, Integration testing object-oriented software systems: An experiment-driven research approach. Electrical and Computer Engineering (CCECE), 2011 24th Canadian Conference on, 2011. 000652-000655.

Lalchandani, J. T. & Mall, R. 2008, Regression testing based-on slicing of component-based software architectures. proceedings of the 1st India software engineering conference,. ACM, 67-76.

Li, B. & Chen, Z. 2006, Pivotal techniques of embedded software testing case generation by genetic algorithms. Computer-Aided Industrial Design and Conceptual Design,. CAIDCD '06. 7th International Conference on, 2006. 1-5.

Loll, V. 2000, Developing and testing algorithms for stopping testing, screening, run-in of large systems or programs. Reliability and Maintainability Symposium, 2000. Proceedings. Annual, 124-130.

Ma, H., Dongfeng, W., Bastani, F., Yen, I. L. & Cooper, K. 2005, A model and methodology for composition QoS analysis of embedded systems. Real Time and Embedded Technology and Applications Symposium,. RTAS 2005. 11th IEEE,. 56-65.

Marrero perez, A. & Kaiser, S. 2009, Integrating Test Levels for Embedded Systems. Testing: Academic and Industrial Conference - Practice and Research Techniques, 2009. TAIC PART '09., 184-193.

Marrero perez, A. & Kaiser, S. 2009, Reusing Component Test Cases for Integration Testing of Retarding Embedded System Components. Advances in System Testing and Validation Lifecycle,. VALID '09. First International Conference on, 2009 1-6.

Man, K., Tang, K. & Kwong, S. 1996. Genetic algorithms: concepts and applications [in engineering design]. *Industrial Electronics, IEEE Transactions on,* 43**,** 519-534.

Petricic, 2011, A. Predictable dynamic deployment of components in embedded systems. Software Engineering (ICSE), 2011 33rd International Conference on, 2011. 1128-1129.

Sabil, S. & Jawawi, D. 2009, Integration of PECOS into MARMOT for Embedded Real Time Software Component-Based Development. Software Engineering Advances,. ICSEA '09. Fourth International Conference on, 2009. 265-270.

Sagarna, R., Arcuri, A. & Xin, Y. 2007, Estimation of distribution algorithms for testing object oriented software. Evolutionary Computation, 2007. CEC 2007. IEEE Congress on, 2007. 438-444.

Srivastava, P. R. & Baby, K. 2010, Automated Software Testing Using Metahurestic Technique Based on an Ant Colony Optimization. Electronic System Design (ISED), 2010 International Symposium on, 2010. 235-240.

Shahid, M., Ibrahim, S. & Mahrin, M. N. R. 2011. A study on test coverage in software testing. Advanced Informatics School (AIS), Universiti Teknologi Malaysia,International Campus, Jalan Semarak, Kuala Lumpur, Malaysia.

Staats, M., Whalen, M. W., Rajan, A. & Heimdahl, M. P. E. 2010, Coverage Metrics for Requirements-Based Testing: Evaluation of Effectiveness. NASA Formal Methods,. 161-170.

Truscan, D., Lindqvist, J. & Lilius, J. 2008, Testable Specifications of NoTA-based Modular Embedded Systems. Engineering of Computer Based Systems, 2008. ECBS 2008. 15th Annual IEEE International Conference and Workshop on the, 2008. 375-383.

Wei, W. & Zhao, Y. 2009, Comparison and Analysis of the Development in Grading Subjective Tests Algorithms. Intelligent Networks and Intelligent Systems,. ICINIS '09. Second International Conference on, 2009. 494-497.

Wieczorek, S., Stefanescu, A. & Roth, 2010, A. Model-Driven Service Integration Testing - A Case Study. Quality of Information and Communications Technology (QUATIC), 2010 Seventh International Conference on the, 2010. 292-297.

Xia, C., Lyu, M. R., Kam-fai, W. & Roy, K. 2000, Component-based software engineering: technologies, development frameworks, and quality assurance

schemes. Software Engineering Conference, 2000. APSEC. Proceedings. Seventh Asia-Pacific, 2000. 372-379.