An Architecture-based Approach to Support Alternative Design Decision in Component-Based System: A Case Study from Information System Domain

Adil A. Aziz [1], Wan M. N. Wan Kadir [1] and Adil Yousif [2]

[1] *Faculty of Computer Science & Information Systems, Universiti Teknologi Malaysia, Department of Software Engineering*

[2] *Department of Soft-Computing and Network Skudai, Johor 81310 UTM, Malaysia*

Adil_sudan@hotmail.com , wnasir@utm.my, adil_uofk@Gmail.com

Abstract

Component-Based System (CBS) is a promised approach to build applications from deployed components. It provides efficiency, reliability, maintainability. Interpreting the results of performance analysis and generating an alternative design to build system from component (Hardware/Software) is a great challenge in the software performance domain. There are so many options to compose the system. Span of design space hinders the selection of the appropriate design alternative. Currently, Meta-heuristics such as Genetic Algorithm (GA) methods are used to solve the problem. In recent investigations Particle Swarm Optimization (PSO), an alternative search technique, often outperforms GA when applied to various problems. In this paper, we describe performance prediction approach based on PSO for component-Based system development. PSO technique can be used to effectively generate alternatives design options in spanned design space and facilitate the design decision during the development process. The proposed approach aids developers to effectively trades-off between architectural designs alternatives, because it covers and generates all possible options and provides the best solution. To the best of our knowledge, we are the first who use PSO in software performance prediction, particularly in the context of CBS. To this end, outlines and an example are presented in the paper to describe the approach.

Keywords: *component; heuristics approach; model-based; performance; prediction; Particle Swarm Optimization*

1. Introduction

Recently component-based system (CBS) has become promised approach that positively impacts the area of developing large and complex systems. Component-Based System (CBS) is an approach to build applications from deployed components. It provides efficiency, reliability, maintainability. Furthermore, Component-based System Development (CBSD) enables the software architects to reason on the composed structure. This is not only essential for the functional properties but also non-functional properties and software quality as well. Performance, which is referring to how extend the system or component has satisfied the predefined requirements on restrictions of specific factors such as accuracy, available memory usage [1], is an important none-functional characteristic that must be deliberated when developing such applications. In the same time, it is an essential attribute for software

quality. The problems in software performance often leads to several difficulties such as; financial loss, costly development, and further than that, the damage of relationships with clients. The German police has developed a system called "Impol-Neu" [2] due to the late performance evaluation, the resulted performance did not satisfy the requirements. Then the system needed more two years to be implemented. Therefore, when performance issues are addressed at implementation or integration time, correction of problems will negatively impact the cost, schedule, and quality of the software [3].

The concept of Software Architecture (SA), which was matured in late 1980, has played an essential role in evaluating the design of complex systems. Since then a broad set of notations, tools, and techniques have been offered, therefore a genuine base for designing enterprise and complicated system was established [4]. However, SA is a suitable stage to cope with software qualities. Notable efforts have been committed to facilitate the evaluation of Software quality in the level of SA [5]. Software architects are employing Model-Driven Development (MDD) [6] to manage architectural models of the system under development. The architectural models transformed into simulation-based or analytical models (e.g. Stochastic process Algebra, QN, Petri nets). Then, the probable problems are derived from the resulted model. However, the interpreting of the results of performance analysis is quite essential in the software performance prediction. Besides, the analysis of software alternatives results is still lacks of automation and is based on the skills and experience of analysts [7]. Furthermore, although, many approaches have been proposed and were successfully applied to predict software performance, still span of design space is serious problem when attempting to select the best design alternative. Meta-heuristics such as Genetic Algorithms (GAs) methods have proven its usefulness to solve the problem even with multi-degree of freedom. In recent investigations, PSO, an alternative search technique, often performed better then GA when applied to various problems. In this paper, we describe performance prediction approach based on PSO for component-Based system development. The proposed approach aids developers to effectively trades-off between architectural designs alternatives. PSO can be used to provoke more efficient results. To the best of our knowledge, we are the first who employ PSO in software performance area particularly to CBS. Outlines of our approach are presented and a case applied using GA is described to be applied to our approach in order to compare between the two techniques.

This paper focuses on performance prediction to generate alternative design decision using particle swarm optimization in context of CBS. Our approach is based on model-driven architecture and completions [8], multi-criteria architecture optimization to consider bounds for quality requirements [9] Generally, model-based approaches rely on the Model Driven Development (MDD) technique which enables developer to efficiently evaluate and assess the system requirements and execution by using a set of models. On the other hand, Metaheuristics-based approaches encode the challenge of improving architectures as an optimization problem and use metaheuristic search techniques. In his review paper [10] Harman has described the use of search based optimization techniques applied software engineering activities.

The main target of our ongoing research is to develop a reasoning framework that supports the alternative design decision in the context of CBS development. In this paper, we present the outlines of our approach which uses quality of services bound to focus the search on an interesting region of design space. Quality of Service Modeling Language (QML) [11] is employed to facilitate the specifications statistically at the time of implementation, design, or dynamically at deployment or runtime.

The paper has been divided into six parts. The concepts of design space, metaheuristic using PSO presented in section two and three respectively. In the fourth part, we describe the

methodology. The related work stated in the fifth part. Finally, the conclusion found in part six.

2. Design Space

When an architect starts building a new CBS application, he/she has many options to do the job. Each probable solution is arranged from a mixture of distinctive components. All those possible alternatives are called Design Options. The combination that satisfied the performance requirements is the target of the architect. However, design options are proportional with the degree of freedom. The degrees of freedom are resulted due to the following [12];

- Components; the selection of one component from number of components with the same functionality but different performance specifications.
- Resource Allocation: due to the fact that, the selection of hardware does not impact the functional of components, its configuration could be changed during search. Therefore, hardware environment are modeled separately from the common assembly. This will enable the adjusting of resource size.
- Setting Parameters: performance of whole system may affected by parameters assigned by software component. Those parameters, often, do not have functional impact. The option of using RMI or SOAP network protocols can be configured.
- Usage Profile: different options related to number of users and their inputs could impact performance. If those variables are represented as parameters, trades-off between scalability and design options will be available.

Therefore, the design-space composes of various, but countable number of candidates. Obviously, those candidates are directly proportional with the degree of freedoms. In many cases, the design space become too large, this is reflecting the task of search for better solution. However, not all available solutions are acceptable to the system; therefore elimination of those options will reduce the design space and, in turn, increases the efficiency of the search. Several constraints are applied to perform such elimination. Noorshams [9] has proposed constraints such as contradiction and Software Architect and quality constrains. The area of prohibited design space DS could be identified as:

DesignSpaceConstrained DS := {c ∈ DS |c is invalid}

3. Optimization Algorithms

This section demonstrates the general formulation of optimization problems. Then we present the swarm intelligent and state its relationship to the family of evolutionary algorithms. Finally particle swarm optimization is described as it will be employed in our approach.

3.1 Problem Formulation

The objective of optimization is to find values of the variables that minimize or maximize the objective function while satisfying the constraints. Therefore, the main elements of any optimization problem are; objective/s function, design variables, constraints and set of

probable solutions. Let us assume an optimization problem denoted as P (S, f) and be identified as follow;

An objective function f to be minimized or maximized, the minimization problem is taken as general, where

f :DxD→IR

The design variables; is a set of unknowns or variables which affect the value of the objective function X.

X = {x1,…,xn}

Constraints: a set of constraints that let the unknowns to take on particular values but prohibit others, (the velocity of constraint could mitigated according to specific strategy such as delete, edit, modify, or penalty strategies) represented as

D1,…,Dn

The set of probable solutions with condition of satisfying all the constraints denoted as:

$$\mathcal{S} = \{s = \{(x_1, v_1), \ldots, (x_n, v_n)\} \mid v_i \in D_i, s$$

(under the constraints, where S is the area of the search or the Design Space).

The solution with minimum objective function values is the global optimal solution. This solution is linked to what known as; set of global optimal solution.

Assume the globally solution for P (S, f) is;

Satisfy

$$f(x^*) \leq f(s) \forall s \in S$$

$$where S^* \subseteq S$$

S* is the set of Global Optimal Solutions.

For any optimization, modeling the problem is essential and good mathematical model of the optimization problem is needed. Modeling means the process of identifying objective function, variables and constraints.

3.2 Intelligent Swarm Optimization

Meta-heuristics are originated and inspired by natural process and creature's behaviors to solve complex real world problems. Optimization is at the heart of many natural processes such as; Darwinian evolution, social group behavior and foraging strategies. The last two decades have witnessed notable increasing in the domain of nature-inspired search and optimization algorithms. Recently, these techniques are applied to variant problems. Evolutionary computing methods and the swarm intelligence algorithms are the main common groups of methods represent the field[13].

Meta-heuristics evolutionary techniques such as Genetic Algorithms (GAs) methods have proven their usefulness to solve the problem of spanned design space. In recent researches Swarm Intelligent (SI) techniques such as Firefly optimization algorithm [14] Particle Swarm Optimization (PSO) [15], [16], [17] an alternative search technique, often performed better than many evolutionary techniques such as GA when applied to various problems [18, 19]. Evolutionary techniques need to handle the population movement; therefore, they are less fast in discovering optimal solutions. Furthermore, evolutionary algorithms may have a memory

to store previous status; this may help in minimizing the number of individuals close to positions in candidate solutions that have been visited before, but it may also slow to converge since successive generations may die out. In recent researches Swarm Intelligent (SI) techniques such as. In this paper we introduce the use of particle swarm optimization to search design space. Our objectives are; to search the design space, automatically generate and evaluate the alternative's design. Then architects can reason on the provided options and choose the optimal solution that satisfies the specified quality requirements. Next section presents some details about PSO.

3.3 Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is one of the swarm intelligent (SI) optimization methods, it's an adaptive optimization method introduced in 1995 by Kennedy and Eberhart [20, 21]. The method inspired by social behavior of swarms such as bird flocking or fish schooling. In PSO, particles never die, and particles are considered as simple agents that move and interact through the search space and record the best solution that they have visited. Each particle represents a candidate solution in the solution search space and each particle has a position vector and velocity. The behavior of the particles is subjected to their capability to train from their past personal experience and from the success of their neighbor to adapt the flying speed and direction to the target. However, particles manage the current position, velocity and personal best position. Beside the personal best solution, the swarm is targeting the global best solution

Let us assume a swarm with N particles flying in design space consists of D-dimensional. Each particle i , re-position itself xi in the direction of the global optimum base on the following two factors;

The best position achieved by itself (pBest I) expressed by pi = (pi1, pi2, . . . , piD). And the best position achieved by the whole swarm (gBest), for a given subset of the swarm which expressed by pg = (pg1, pg2. . . pgD). The difference between the current position of the particle i and the best position of its neighborhood expressed by (pg − xi).

For simplification the two equations are :

V[i] = v[i] + c1 * rand() * (pbest[i] - present[]

v) + c2 * rand() * (gbest[i] - present[i]) (1)

present[i] = persent[i] + v[i] (2)

Where c1 and c2 are learning factors (weights)

The parameters used in algorithm are:

c1: acceleration factor related to gbest =3

c2: acceleration factor related to lbest =1

rand1 ( ): random number between 0 and 1

The following is the algorithm for PSO optimization.

For each particle
Initialize particle with feasible random number
END
Do
For each particle

Calculate the fitness value
If the fitness value is better than the best fitness value (pBest) in history
Set current value as the new pBest
End

Choose the particle with the best fitness value of all the particles as the gBest
For each particle
Calculate particle velocity according to velocity update equation

Update particle position according to position update equation
End

Until (maximum number of iterations) or (condition be satisfied).

rand2 ( ): random number between 0 and 1

According to [18], CL-PSO produces higher quality solutions and performs significantly better than other PSO algorithms. Therefore we employ CL-PSO in our work.

4. Methodology

1. According to the requirements, the system to be optimized is modeled with the PCM, and degrees of freedom specified before. See Figure 1.
2. QML is used to model the quality requirement of the system. And QML profile linked between PCM model and QML model.
3. The optimization starts with one or more initial candidates, in our approach we will use PSO algorithm or tool. Which did not yet finished.
4. From the resulted set of solution, the set of Pareto-optimal architecture configurations feasible with respect to the quality requirements is presented

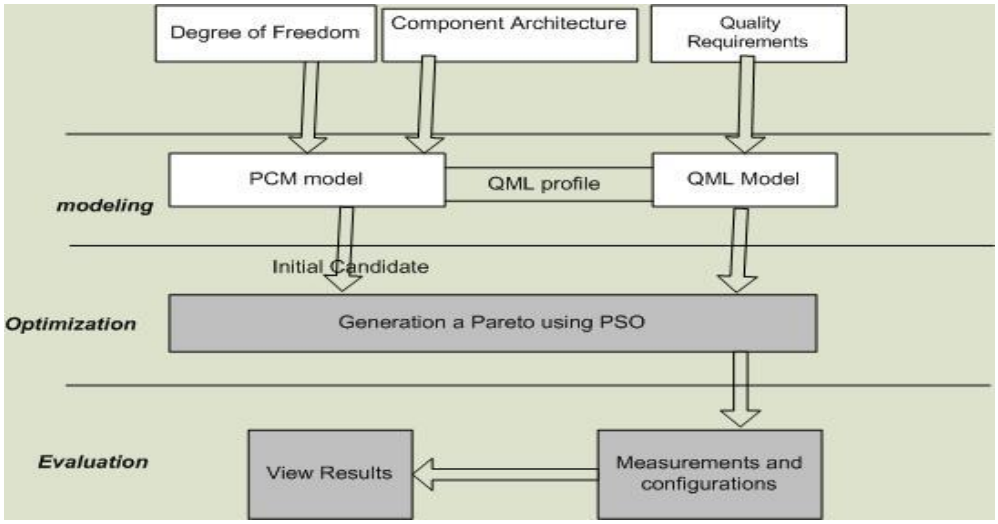

Figure 1: Process Overview of Our Approach

Finally, the software architect makes the trade-off decision and chooses one of the solutions.

5. Example Demonstration

The case study of Business Report System (BRS) is selected to evaluate the approach. The BRS provides statistical reports about business processes and is loosely based on a real system. The system composed of nine components and is allocated to four servers. One usage scenario, user interacts with the system every five seconds requesting a sequence of reports and views. For generating reports or viewing the plain data logged by the system, the web Server component deals with user requests. It delegates the requests to a Scheduler component, which in turn forwards the requests to the ReportingEngine component. The ReportingEngine accesses the Database component, sometimes using an indirectly cache component see [22] for more details.

Analysis Design Option

According to the requirements, the system to be optimized is modeled with the PCM, and degree of freedom is specified.

Search Setting

We use an array of double values for the encoding. Each design option is encoded in a single particle. For example, a system with the design options ("CPU processing speed server 1", "CPU processing speed server 2", "Component to provide service s"), could have an initial choice of (1.2GHz, 2.0GHz, b) and thus a particle of (1.2, 2.0, 1). The value in that place in the array thus represents the choice made for that design option. In the example, the first particle specifies the CPU processing speed of server 1 to be 1.2 GHz.

From the example description each component can be allocated to the one of the four servers with varying configuration. Server configuration and allocation are realized with joint degrees of freedom. Overall, 16 combinations were available for the four base servers Si and four configuration options Cj: S1_C1, S1_C2... S4_C3, S4_C4. The Web server can be realized using third party components. The software architect can choose among three functional equivalent implementations: Webserver2 with cost 4 and Webserver3 with cost 6. Both have less resource demand than the initial Web server. Webserver2 has better availability for the requests of type "view," while Webserver3 has better availability for the requests of type "report."

Parameters are stated based on the PSO Algorithm proposed in [18] (see Table 1). This algorithm is selected because it is employing a strategy of enabling other particles' previous best positions are exemplars to be learned from by any particle and each dimension. This means any particle can potentially learn from a different exemplar. Therefore this strategy makes the particles have more exemplars to learn from and a larger potential space to fly.

Table1: Parameters

	Parameters
1	No. of particles
2	Inertia weight ω Lin.
3	Acceleration coefficient c
4	Velocity bound V^{d}_{max}
5	Refreshing gap m
6	Boundary condition
7	Termination

Run Search

The optimization starts with one or more initial candidates, in our approach we will use PSO algorithm or tool. Which did not yet finished.

Table 1 illustrates the stages to generate optimum solutions. The first four cells allocated for the processing rate of the CPUs of the servers 1 to 4. Web-Servers are coded by WS, Servers are denoted by Ser. The first row represents the initial solution. The last two rows 25 and 26; are the optimal solution on which the software architect makes the trade-off decision and chooses one of the solutions. The results published in [23] will be used to validate our approach.

Table2: Sample of Results – Using Hybrid Algorithm [5]

0	(20.0, 20.0, 26.0, 8.0, WS , Ser1, Ser2, Ser3, Ser4, Ser4)
1	**(10.01, 21.64, 44.93, 5.37, WS2, Ser1, Ser2, Ser2, Ser4, Ser3)**
2	**(31.86, 23.68, 51.20, 9.88, WS, Ser1, Ser3, Ser3, Ser3, Ser1)**
3	**(12.52, 33.66, 37.92, 12.37, WS, Ser4, Ser2, Ser3, Ser3, Ser2)**
4	**(12.52, 33.66, 37.92, 12.37, WS, Ser2, Ser2, Ser3, Ser4, Ser2)**
25	**(29.48,31.26, 43.23, 8.0, WS1, Ser3, Ser3, Ser2, Ser4, Ser4)**
26	**(10.01, 21.57, 38.20, 11.60, WS4, Ser3, Ser3, Ser4, Ser2, Ser3)**

Results and Trade-off

From the resulted set of solution, the set of Pareto-optimal architecture configurations feasible with respect to the quality requirements is presented. Consequently, the software architect makes the trade-off decision and chooses one of the solutions. Thanks to availability of the trade-off prove, the approach can be used to achieve the architect's objectives since he/she can decide on design option according to the technical and financial preferences.

From our initial results applied to the domain of Embedded System domain, we present Figure 2 to show how architect can manage the conflict between Performance and reliability.

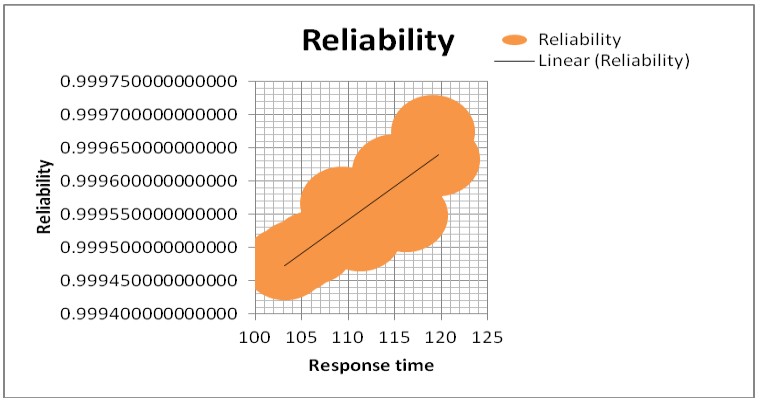

Figure 2: Trades-off between Reliability and Response-time

The figure shows that, the higher the reliability is provided, the slower in the response-time we obtained. The architect often, seeks for the fast response time and high reliability. The architect might decide base on worth case or within upper and lower limits for each quality criterion. In Table 3, we state the two close options, and it can be seen that; the higher and best reliability provided by option 1, but with a response time slower than second option. The second option has a faster response time, but reliability is slightly lower than first option. Consequently, the architect takes the decision that suit the system he/she is working on. In case of trade-off between cost and other quality attributes, architect can take the decision based on technical and financial preferences.

Table 1: Reliability against Response Time

Option No	Response Time	Reliability
1	119.11	0.999674741907323
2	114.89	0.999615464113025

6. Related Work

Many performance prediction approaches have suggested solving the problem of manual search by using Metaheuristic-based approaches. From literature, solutions can be grouped into three types that guide the search process; Anti-Patterns based solution, Rule-Based search, Scenario-based approaches, and search techniques.

Cortellessa and Frittella [24] have proposed an automated approach for the performance feedback generation process based on performance anti-patterns. By applying the approach in simple case study, they describe how anti-patterns are detected in an XML format. It takes as input the XML representation of the software system and gives in output a list of detected performance anti-patterns. No grantees to apply it in complex system. Since, it includes the problem (i.e. model properties that describe the anti-pattern) and the solution (i.e. actions to take for eliminate the problem), bad practice and unsolved problems may decrease the effectiveness of this method. Besides, human experience in several steps is needed. For example, the detection of antipatterns in a subsystem is a task whose complexity heavily depends on the structure of the subsystem and the definition of the antipatterns itself. Furthermore, there is no offer of new architecture candidates.

Rule-based [25] approaches try to identify problems in the model (e.g. bottlenecks) based on predefined rules and rules containing performance knowledge are applied to the detected problems. Rule-based approaches focus on performance analysis without considering other quality criteria. They operate on the performance model instead of the architectural model and are therefore difficult to use for regular architects not familiar with performance formalisms. These approaches cannot find solutions for which no rule exists, thus, they cannot cover all possible solutions and might result in locally optimal solutions.

McGregor et al [26] have developed, ArchE framework, a scenarios-based approach. ArchE framework assists the; creation of architectural models, collection of requirements as well as the information needed to analyze the extra-functional properties for the requirements. Also the approach provides the evaluation tools for modifiability or performance analysis, and suggests the needed improvements. ArchE features a simple performance model and the architecture model is not component based.

PSO has been applied in software testing namely unit testing [27]. However, the generation of appropriate test cases for a software unit impacts the quality of the overall testing of this

unit. GAs was the common search algorithms to find such cases. The author tried to answer the question of how PSO, as an alternative algorithm, contribute to the software testing in context of evolutionary structural testing. The researcher concludes that; GA features a slightly faster convergence for simple functions whereas PSO outperforms GA primarily for complex functions with big search spaces.

Metaheuristic-based approaches (e.g., [28] [29] [30], [31]) encode the challenge of improving architectures as an optimization problem and use metaheuristic search techniques [32] [10] (e.g., genetic algorithms, simulated annealing, etc) to find better design models. Noorshams et al [9] have extended the existing Quality of service Modeling Language to enable the specification of optimization goals and quality requirements. GA is used as optimization technique and the case study in PCM. The approach is base on identifying forbidden space, so the search focuses on specific design space. Our approach is based on this method, but we claim that, by using PSO after translation the QML requirements to constraints in an optimization problem, our approach will be produce better efficient results especially with large design space. Therefore the software architect can effectively and efficiently obtain better solution.

7. Conclusion

Designing architectures that provide a good trade-off between multiple quality criteria is quite difficult. Most of the proposed approaches do not provide substitution solutions; rather, they just recognize that performance is not satisfied. The architect needs to map the result back in the design model and manually develop new alternative. PSO has proved to be an efficient optimization method for single objective optimization, and more recently it has shown promising results for solving multi-objective optimization problems in different areas. PSO as search technique could be used to generate alternatives design automatically so it will contribute in improving the accuracy and decreasing the time of development. The use of PSO in the field of performance prediction will support the creation and evaluation of new architecture candidates. Therefore, a larger search space is explored and architects could be able to select between options that could improve the architecture of system. The improvement of architecture encompasses faster and more reliable software components, a more powerful hardware environment, or a different allocation of components to available hardware.

This paper has identified and delineated relevant concepts to optimization such as design space, degree of freedom, and metaheuristic algorithms, description of PSO, and methodology of our approach as well. Since improving one quality feature can weaken another, quality features cannot be individually improved. In the context of CBS system development, the generation and evaluation of the alternative architectures system based on performance involves the trade-off between multiple, potential conflicting criteria.

PSO technique can be used to effectively to deal with such a problem and facilitate the decision making during the development process. The proposed approach uses quality of service bound to evaluate upper /lower bound in the system performance and recognizes software performance bottleneck. QML is used to model the quality requirement of the system. PSO is similar in some ways to genetic algorithms and other evolutionary algorithms, however it requires less computational book keeping, and generally only a few lines of code are needed. Besides, PSO is very easy to understand and implement. Our objective to the next step is to apply our method using PSO to the case study of BPS, then comparing its results with the results obtained from the same case study applying GA such as the approach applied in [9].

Our ongoing research aims to propose multi-objective Particle Swarm Optimization (MOPSO) approach for information system and embedded system to support the right design decisions in developing software systems that have conflicting quality attributes. In this paper reliability, cost, and response time are considered, more quality attributes planned to be added such as availability. Furthermore, expected discount in the cost as the number of requested component increased will be considered as well.

Acknowledgement

The authors would like to thank Malaysian Ministry of Higher Education (MOHE) for their financial support under the Fundamental Research Grant Scheme (FRGS) Vot No. 78601.

References

[1] IEEE, Standard Glossary of Software Engineering Terminology, **(1991),** pp 610.12-1990.

[2] Jakob and Tretkowski, Das polizeiliche Informationssystem INPOL, **(2002)**

[3] Koziolek, H., Performance evaluation of component-based software systems: A survey. Performance Evaluation, **(2009),** 67(8): p. 634-658.

[4] Clements, P. and M. Shaw, "The Golden Age of Software Architecture" Revisited. Ieee Software, **(2009),** 26(4): p. 70-72.

[5] Martens, A., et al., A Hybrid Approach for Multi-attribute QoS Optimisation in Component Based Software Systems, in Research into Practice – Reality and Gaps, G. Heineman, J. Kofron, and F. Plasil, Editors. **(2010),** Springer Berlin / Heidelberg. p. 84-101.

[6] Fritzsche, M., et al., Towards Utilizing Model-Driven Engineering of Composite Applications for Business Performance Analysis, in Model Driven Architecture – Foundations and Applications, I.H. Schieferdecker, Alan, Editor. **(2008),** Springer Berlin / Heidelberg. p. 369-380.

[7] Cortellessa, V., A. Di Marco, and C. Trubiani. Performance Antipatterns as Logical Predicates. in Engineering of Complex Computer Systems (ICECCS), 2010 15th IEEE International Conference on, **(2010)**

[8] Happe, J., et al., Parametric performance completions for model-driven performance prediction. Performance Evaluation, **(2010),** 67(8): p. 694-716.

[9] Noorshams, Q., A. Martens, and R. Reussner. Using quality of service bounds for effective multi-objective software architecture optimization. 2011. Oslo, Norway: Association for Computing Machinery, **(2011)**

[10] Harman, M. The Current State and Future of Search Based Software Engineering. in Future of Software Engineering, 2007. FOSE '07. **(2007)**

[11] Frlund, S. and J. Koistinen, QML: A Language for Quality of Service Specification, **(1998)**

[12] Martens, A. and H. Koziolek, Automatic, Model-Based Software Performance Improvement for Component-based Software Designs. Electronic Notes in Theoretical Computer Science, **(2009),** 253(1): p. 77-93.

[13] website. Swarm and Evolutionary Computation. **(2011)** [cited 2011 13-11-2011]; Available from: http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/Swarm%20and%20Evolutionary%20Computation_2.pdf.

[14] Adil yousif, a.h.a., sulaiman mohd nor, adil ali abdelaziz Scheduling jobs on grid computing using firefly algorithm. Journal of Theoretical and Applied Information Technology , 14570 -JATIT, p 155 - 164 Vol 33. No. 2 -- 2011, **(2011)**

[15] Gudise, V.G. and G.K. Venayagamoorthy. Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks. in Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE. **(2003)**

[16] Boeringer, D.W. and D.H. Werner, Particle swarm optimization versus genetic algorithms for phased array synthesis. Antennas and Propagation, IEEE Transactions on, **(2004),** 52(3): p. 771-779.

[17] Eberhart, R. and Y. Shi, Comparison between genetic algorithms and particle swarm optimization, in Evolutionary Programming VII, V. Porto, et al., Editors. **(1998),** Springer Berlin / Heidelberg. p. 611-616.

[18] Liang, J.J., et al., Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. Evolutionary Computation, IEEE Transactions on, **(2006),** 10(3): p. 281-295.

[19] Qasem, S.N. and S.M. Shamsuddin, Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis. Applied Soft Computing, **(2011),** 11(1): p. 1427-1438.

[20] Kennedy, J. and R. Eberhart. Particle swarm optimization. in Neural Networks, **(1995)** Proceedings., IEEE International Conference on 1995

[21] Kennedy, J. and R.C. Eberhart. A discrete binary version of the particle swarm algorithm. in Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., **(1997)**, IEEE International Conference on (1997)

[22] A. Martens, D.A. PerOpteryx/Hybrid Optimisation Case Study. [cited **(2011)** 9-11-2011]; Available from: https://sdqweb.ipd.kit.edu/wiki/PerOpteryx/Hybrid.

[23] Martens, A., F. Brosch, and R. Reussner, Optimising multiple quality criteria of service-oriented software architectures, in Proceedings of the 1st international workshop on Quality of service-oriented software systems. **(2009),** ACM: Amsterdam, The Netherlands. p. 25-32.

[24] Cortellessa, V. and L. Frittella, A framework for automated generation of architectural feedback from software performance analysis, in Proceedings of the 4th European performance engineering conference on Formal methods and stochastic models for performance evaluation. **(2007),** Springer-Verlag: Berlin, Germany. p. 171-185.

[25] Xu, J., Rule-based automatic software performance diagnosis and improvement, in Proceedings of the 7th international workshop on Software and performance. **(2008)** ACM: Princeton, NJ, USA. p. 1-12.

[26] J. D. McGregor, F.B., L. Bass, P. Bianco, and M. Klein. , Using arche in the classroom: One experience. , in Technical Report CMU/SEI-2007-TN-001, Software Engineering Institute, Carnegie Mellon University, **(2007)**

[27] Windisch, A., S. Wappler, and J. Wegener, Applying particle swarm optimization to software testing, in Proceedings of the 9th annual conference on Genetic and evolutionary computation. **(2007),** ACM: London, England. p. 1121-1128.

[28] G. Canfora, M.D.P., R. Esposito, and M. L. Villani. , An approach for qoS-aware service composition based on genetic algorithms. In H.-G. Beyer and U.-M. O'Reilly, editors, Proc.of Genetic and Evolutionary Computation Conference 2005, pages 1069–1075. ACM, **(2005)**

[29] A. Aleti, S.B., L. Grunske, and I. Meedeniya, Archeopterix: An extendable tool for architecture optimization of AADL models. International Workshop on Model-Based Methodologies for Pervasive and Embedded Software (MOMPES), pages 61–71 **(2009)**

[30] Bouktif, S., H. Sahraoui, and G. Antoniol, Simulated annealing for improving software quality prediction, in Proceedings of the 8th annual conference on Genetic and evolutionary computation. **(2006),** ACM: Seattle, Washington, USA. p. 1893-1900.

[31] Khoshgoftaar, T.M., L. Yi, and N. Seliya, A multiobjective module-order model for software quality enhancement. Evolutionary Computation, IEEE Transactions on, **(2004),** 8(6): p. 593-608.

[32] Blum, C. and A. Roli, Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Comput. Surv. **(2003),** 35(3): p. 268-308.

Authors

Mr. Adil A. A Saed graduated from University of Khartoum. Currently he is a PhD student in software engineering, University Technology Malaysia. His research interests are component-Based system, model driven development, Optimization technique and in specific, Particle Swarm optimization - PSO.

Mr. Wan M. N. Wan Kadir is an Associate Professor in Software Engineering Department, University Technology Malaysia. His has a PhD degree in the field of Software Engineering from the University of Manchester. His research interest covers various SE knowledge areas based on the motivation to *reduce the cost* of development and maintenance as well as to *improve the quality* of large and complex software systems.

Mr. Adil Yousif Received the B.Sc. and M.Sc. from University of Khartoum, Currently he is a PhD candidate at Faculty of Computer Sciences & Information Systems, Universiti Teknologi Malaysia UTM. His research interests include computer network, grid computing and optimization techniques.