Vol. 6, No. 2 February 2015 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

Applying IWD Algorithm for Discrete Time, Cost and Quality Trade - off in Software Projects with Expressing Quality by Defects

[1] Abdulelah G.F. Saif, [2] Safia Abbas, [3] Zaki Fayed
[1] PhD. Student, Department of Computer Science, Ain Shams University, Egypt
[2] Dr, Department of Computer Science, Ain Shams University, Egypt
[3] Prof, Department of Computer Science, Ain Shams University, Egypt
[1] abdulelah_saif@yahoo.com, [2] Safia_abbas@yahoo.com, [3] ZFayed@hotmail.com

ABSTRACT

This research paper adapts intelligent water drops (IWD) algorithm to solve the discrete time- cost-quality trade-off problem (DTCQTP) in software projects which is never solved before by this algorithm. Project manager's goal is getting an optimal allocation of time, cost and quality of each task/activity in the software project such that total time and cost of the project is minimized while project quality is maximized. To achieve this goal, IWD algorithm decides the preferred modes of performing tasks where task quality is expressed quantitatively in terms of defects. An example is given at the end to show the trade-off analysis between project's cost, time and quality.

Keywords: *Time cost quality trade-off, Project scheduling, Meta-heuristic algorithms, Software defect origins, Software projects*

1. INTRODUCTION

One main task for project managers is to administrate projects under concern and achieve the required goals within the plan. Improving the resources allocation to guarantee minimum cost, time and high quality is an obligatory task for such administration [8, 14]. Accordingly, many researchers have devoted much effort to solve such riddle, on one hand, some of these researches considered continues mode for the time, cost and quality [15]. On the other hand, multiple modes for each activity depending on discrete models have been considered [10]. Accordingly, mathematical and meta-heuristic techniques are taken into account to solve such problems where meta- heuristic techniques were better to solve such problems [1, 2, 3, 5, 8, 15].

In the two cases, continues mode and discrete mode, task quality is measured based on managers' judgment which is expressed by values such as 90%, 80%, etc which don't reflect exactly number of defects in a task. Accordingly, this paper expresses each task quality in terms of number of it's defects and thus a total number of defects for all tasks reflect the quality of the software project quantitatively where most software organizations have databases to store times, costs and defects of software project tasks and also subcontractors often offer bids in terms of times, costs and defects to perform specific tasks in large software projects. To solve the problem in the discrete case (DTCQTP), a meta-heuristic algorithm, intelligent water drops (IWD), is used. The paper also introduces an example that shows the trade-off analysis between project's time, cost and quality.

The rest of this paper is organized as follows: in Section 2 related work, section 3 problem definition, section 4 software project time, cost and quality, section 5 DTCQTP representation and assumptions, section 6 IWD, section 7 example and section 8 concludes and discusses the paper.

2. RELATED WORK

DTCQTP is the problem of optimizing time, cost and quality based on discrete mathematical models; it is an extension of discrete time-cost trade-off problem (DTCTP) by taking quality into consideration [1]. The first work initiated by Babu and Suresh who claimed that the quality of a completed project may be affected by activities acceleration. Thus they developed a solution procedure which optimizes time, cost and quality in continuous mode [15]. Later on this procedure has been applied on real cement factory construction project in Thailand for evaluation. Although this procedure can assist managers in making trade-off decisions by providing valuable information, it disregards the multiple modes for different activities [13]. After that, several works considered DTCQTP by different authors such as:

Tareghian et al developed three inter related binary integer programming models for DTCQTP and used lingo software for optimization [4]. Afshar et al developed a new met heuristic, multi-colony ant algorithm, for optimizing time-cost-quality tradeoffs to generate optimal/near optimal solutions [8]. Because DTCQTP is NP-Hard, Iranmanesh et al proposed a meta-heuristic based on GA to solve such problem [7]. Refaat et al developed a practical software system using a multi-objective genetic algorithm (GA) for optimizing time-cost-quality tradeoffs simultaneously to help planers in decision making [5] and Shankar et al analysed project scheduling problem in terms of time, cost and quality [6]. Most of these mentioned works, task quality is measured based on manager's judgment. Shahsavari et al developed a mathematical model for discrete time, cost and quality trade-off problem using a novel hybrid genetic algorithm (NHGA) [2]. Moreover, in order to handle project quality uncertainty, NHGA has been applied associated with fuzzy logic by assuming time and cost as crisp variables, while quality as linguistic variable [3]. Roya et al estimated task quality based on its time and cost using fuzzy logic; however this is applicable when software organizations have not databases to store the parameters [14].

3. TIME, COST and QUALITY TRADEOFF PROBLEM DEFINITION

The discrete time, cost trade-off problem (DTCTP) [9]-[12], is a well known problem, in which activities durations are reduced by using more resources and overcomes the deadline problem. However, more resources lead to cost increasing. Recently, project managers' main consideration is to improve the project quality by reducing both the time needed and the cost leading to discrete time, cost, quality trade-off problem (DTCQTP) [1]-[8]. Accordingly, many met heuristic algorithms have been devoted to solve such problem such as genetic [2- 3], practical swarm optimization (PSO) [1] and Muliti-colony ant optimization [8] algorithms. DTCQTP has multiple efficient solutions, but in this work, a single solution is obtained in terms of minimum cost and time with maximum quality. Problem formulation in [2] is used in this problem with little modification.

4. SOFTWARE PROJECT TIME, COST AND QUALITY

Time: is time required to develop software. Cost includes hardware and software costs, travel and training costs, effort costs (the most dominant factor in most projects) and effort costs overheads; costs of building, heating, lighting, costs of networking and communications and costs of shared facilities (e.g library, staff restaurant, etc.) [16]. These costs are classified as direct cost which vary during project development such as travel costs and indirect cost which remain constant during time unit such as lighting costs [2]. Quality has been used in different contexts and has different definitions [17] which means different things to different people [18], but in this research, quality is defined as number of residual defects, with respect to any activity, offered in each bid (mode) from any subcontractor or number of residual defects in similar previously developed projects where their times, costs and defects were stored in organization database. The defect is defined as a divergent of actual results from desired results. According to [17], defects are classified based on their origin as requirements defects (e.g. leaving out a required cancel option in an input screen), design defects (e.g. error in the algorithm), and coding defects (e.g. looping 9 instead of 10 times), bad fixes (defects introduced during fixing defects) and documentation defects (e.g. incorrect instructions in user's manual to cancel an operation).

5. DTCQTP REPRESENTATION and MODELING

The following subsections show DTCQTP representation and assumptions as well as its mathematical model.

5.1 DTCQTP Representation and Assumptions

IWD algorithm assumes that the DTCQTP has the representation of activity-on-node network and also assumes that every node (activity) in a network has virtual edges to all its modes as in figure 1. Activities 1 and 5 have virtual edges to all their modes where M1 is the first mode and Mn is last mode of an activity and the other activities have virtual edges to their modes similar to activities 1 and 5. Activities S and F don not have modes and therefore do not have virtual edges to modes because these are dummy activities.

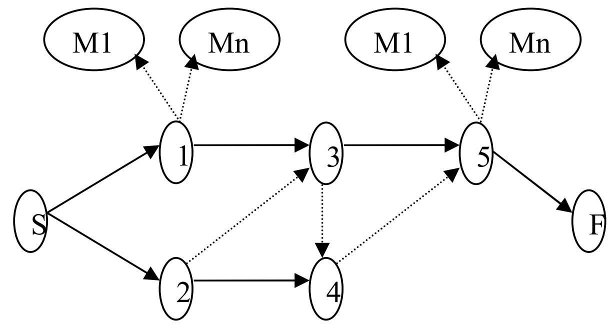

Fig 1: DTCQTP representation

5.2 Mathematical Modelling

The following notations are used to describe the DTCQTP:

Ct: Total cost of project (direct plus indirect), *Tt*: Total duration of project, *Qt*: Total quality of project, *Ic*: Project indirect cost per time unit, *Modes(i)*: Set of available execution modes for activity *i*, *Cik*: Direct cost of activity i when performed the kth execution mode, *tik*: Duration of activity *i* when performed the kth execution mode, *qik*: Quality of activity *i* when performed the kth execution mode, *yik*: Binary variable which is 1 when mode *k* is assigned to activity *i* and 0 otherwise, *Defects Allowed*: Upper bound for Project quality. *Tcpm* : Critical path duration obtained by critical bath method(CPM), if set of modes $\underline{K} = \{k1,k2,....,kn\}$ are assigned to activities.

Mixed integer programming is used for modeling DTCQTP:

$$Min\ Ct=\sum_{i=0}^{N+1} \sum_{k\in \mathrm{Modes(i)}} Cik\ .yik+ Ic\ .\ Tcpm \qquad (1)$$

$$Min\ Tt = Tcpm \qquad (2)$$

Subject to:

$$\sum_{i=0}^{N+1} \sum_{k\in \mathrm{Modes(i)}} qik.yik <=Defects\ Allowed \qquad (3)$$

$$\sum_{k\in \mathrm{Modes(i)}} yik=1. \qquad (4)$$

$$y_{ik} \in \{0,1\} \quad \forall\ i,k \qquad (5)$$

Objective functions (1) and (2) minimize the project's total costs and duration respectively. Constrain (3) enforces that the total quality of project does not bypass the desired level (upper bound). In (4) one and only one execution mode is assigned to each activity and equation (5) is sign constrains.

6. INTELLIGENT WATER DROPS (IWD) ALGORITHM

IWD is swarm-based optimization algorithm which has been inspired from natural rivers that find optimal/nearly optimal paths to their destination. IWD finds optimal/ nearly optimal solutions for optimization problems, by simulating the mechanisms that happen in the natural river system and implementing them in the form of algorithm [19],[20].

The IWD depicted the DTCQTP in the form of graph. The activities considered as nodes and the modes considered as virtual nodes connected to the activities through virtual edges figure 1. The algorithm has two kinds of parameters: static and dynamic. The static parameters include: termination criteria that terminates the algorithm execution such as maximum number of iterations *MaxIter*=100, number of water drops N_{IWD} =10, number of nodes (activities) *N*, number of modes of activities *Modes(i),i=1..N*, initial soil (*InitSoil*) and initial velocity (*InitVel*) that are set to 10000 and 100 respectively. To update the velocity, used parameters are $a_v = 1$, $b_v = 0.01$ and $c_v = 1$. To update the soil, used parameters are $a_s = 1$, $b_s = 0.01$ and $c_s = 1$. Local soil updating parameter P_n and global soil updating parameter P_{IWD}, which are chosen from [0, 1], here, are set to 0.9.

The other kind, dynamic parameters, includes: The initial soil for every virtual edge *(i, k)* is set by *soil (i, k) = InitSoil* and the total-best solution is denoted by $Ct=\infty$, $Tt=\infty$ and $Qt=-\infty$.

a. At beginning of each iteration of the algorithm, a visited modes list for each IWD is created and it is initialized to the empty list; *Vc_IWD_Modes*= { } and the velocity of each IWD is set to *InitVel* whereas the initial soil that each IWD carry is set to *zero*.
b. Each IWD is placed on a randomly chosen node (activity) and the visited modes list of each IWD is updated to include the mode just visited.
c. Each IWD choose mode *k* of activity *i* by the probability

$$p^{IWD}(i,k) = \frac{f(soil(i,k))}{\sum_{j \notin Vc\_IWD\_Modes(i)} f(soil(i,j))} \qquad (6)$$

Such that $f(soil(i,k)) = \dfrac{1}{e_s + g(soil(i,k))}$

$$g(soil(i,k)) = \begin{cases} soil(i,k) \text{ if } \min_{l \notin Vc\_IWD\_Modes(i)}(soil(i,l)) >= 0 \\ soil(i,k) - \min_{l \notin Vc\_IWD\_Modes(i)}(soil(i,l)) \quad else \end{cases}$$

e_s is small number to prevent a possible division by zero ,e_s here is set to 0 .0001.

d. For each IWD chose the mode *k* of activity *i*, it updates its velocity $Vel^{IWD}(t)$ by

$$Vel^{IWD}(t+1) = Vel^{IWD}(t) + \frac{a_v}{b_v + c_v . soil^2(i,k)} \qquad (7)$$

e. For each IWD chose the mode *k* of activity *i*, it computes the soil *Δsoil(i, k)* that the IWD load from the path (virtual edge) between *i* and *k* by

$$\Delta soil(i,k) = \frac{a_s}{b_s + c_s . Time^2(i;k;Vel^{IWD}(t+1))} \qquad (8)$$

Such that

$$Time(i,k;Vel^{IWD}(t+1)) = \frac{1}{Max(e_v, Vel^{IWD}(t+1))}, \quad e_v = 0.0001.$$

f. For each IWD chose mode *k* of activity *i*, it updates the soil *soil(i, k)* of the path(virtual edge) between activity *i* and mode *k* and also updates the soil that it carries $Soil^{IWD}$ by

$$Soil(i,k) = (1 - Pn) * soil(i,k) - Pn * \Delta soil(i,k) \qquad (9)$$

$$Soil^{IWD} = Soil^{IWD} + \Delta soil(i,k)$$

g. After every IWD completes choosing all modes to all activities, every IWD finds its solution.
h. Then, the algorithm finds iteration-best solution *IB* from all the solutions found by the IWDs.

$$IB = Min(C_{Sum}^{IWD} + IC * Tcpm^{IWD})$$

$$s.t\ (Q_{Sum}^{IWD} <= Defects\ Allowed) \qquad (10)$$

$Tcpm^{IWD}$ is a time computed using critical path method (CPM) from IWD chosen modes.

C_{Sum}^{IWD} , Q_{Sum}^{IWD} are the sum of direct costs and qualities extracted from chosen modes of IWDs respectively, *IC* is an indirect cost per time unit and *Defects Allowed* is a upper bound for quality.

i. After that, the algorithm updates the soils on the virtual edges that form the current iteration-best solution *IB* by

$$Soil(i,k) = (1 + P_{IWD}) * soil(i,k) - P_{IWD} * Soil^{IWD^{IB}} / Modes(i), \ \forall i,k \in IB, i=1,\dots,N \qquad (11)$$

Vol. 6, No. 2 February 2015 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

$Soil^{IWD^{IB}}$ is the soil that IWD with best iteration solution carries.

j. Before the end of each iteration of the algorithm, the total-best solution *Ct*, *Qt, Tt* is updated by the current iteration-best solution *IB* using

$$Ct = \begin{cases} IB & \text{if } Ct>IB \\ Ct & \text{Otherwise} \end{cases},$$

$$Qt = \begin{cases} Q_{Sum}^{IWD^{IB}} & \text{if } Ct>IB \\ Qt & \text{Otherwise} \end{cases}, \tag{12}$$

$$Tt = \begin{cases} Tcpm^{IWD^{IB}} & \text{if } Ct>IB \\ Tt & \text{Otherwise} \end{cases}$$

$Q_{Sum}^{IWD^{IB}}$, $Tcpm^{IWD^{IB}}$ are the sum of defects and a time obtained by IWD with best iteration solution respectively.

k. At the end of each iteration of the algorithm, if the total best solution is not improved after specified number of iterations,(here assume 20), soils of its virtual edges are reinitialize by

$InitSoil *0.1* \Gamma; \Gamma \in [0, 1]$ whereas the soils of others are reinitialized by *InitSoil* (13)

Step 11 is added to original IWD to improve it [19].

Figure 2 shows the formal steps of IWD algorithm.

1. Set parameters and read problem data
2. Initialize the soils of virtual edges between activities and thier modes.
3. While (termination condition not met) do
4. Initialize IWDs.
5. Construct solutions by IWDs.
6. Find the iteration best solution.
7. Update the soils of virtual edges that form current best solution.
8. Update the total best solution.
9. If the total best solution is not improved after 20 consecutive iterations then
10. Reinitialize soils of all virtual edges by *InitSoil.*
11. Reinitialize soils of the total best solution virtual edges by $InitSoil *0.1* \Gamma, \Gamma \in [0, 1]$.
12. End if
13. End while
14. Return the total best solution.

Fig 2: IWD algorithm for DTCQTP

7. EXAMPLE

Example with five task software programming project is considered figure 3, where activity 1 represents feasibility study and 2, 3, 4 and 5 represent requirements analysis, design, code and documentation activities respectively and S and F are dummy activities. If bids offered from different subcontractors to perform specific activities in this project or this project is similar to some projects developed previously, the modes take the forms as in table 1 below and *Indirect cost*=20. In real world, large projects have thousand of activities and thus more bids offered from many subcontractors to perform specific activities/tasks.

The algorithm is implemented using c# 2008 and executed only once for each quality bound and the solution is obtained. To get other solutions, run the algorithm more than once and take the best solution.

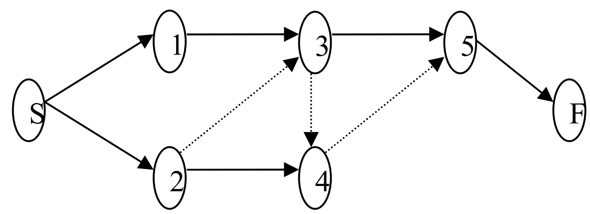

Fig 3: Project example

Table 1: Execution modes of activities

Activity ID	Modes)	Cost	Time	Quality(Defects)
1	1	60	100	4
	2	65	80	8
	3	90	70	10
2	1	45	90	5
	2	55	90	5
	3	80	65	15
	4	100	45	20
3	1	80	70	11
	2	100	100	3
4	1	35	100	30
	2	75	75	3
	3	95	100	4
	4	100	80	4
5	1	65	75	5
	2	50	50	32
	3	75	60	5

Table 2 shows the results in terms of total quality Qt, cost Ct and time Tt with Direct cost Cd of applying IWD to this project using different quality bounds (*Defects Allowed*). This project has about 5^4 solutions. Indirect cost is included in the first row of table 2.

Table 2: Final results

Indirect Cost=20					
Qt	Ct	Tt	Cd	Solution	Defects Allowed
49	5920	275	420	3 4 1 2 3	50
45	6025	280	425	3 3 1 4 3	45
34	6265	295	365	3 1 1 2 3	40
34	6265	295	365	3 1 1 2 3	35
28	6435	305	335	1 1 1 2 3	30
24	6870	325	370	2 2 2 2 3	25
20	7055	335	355	1 1 2 2 3	20

From table 2 and figure 4, the total cost and time are increased by minimizing quality bounds (minimizing defects) which mean maximizing project quality.

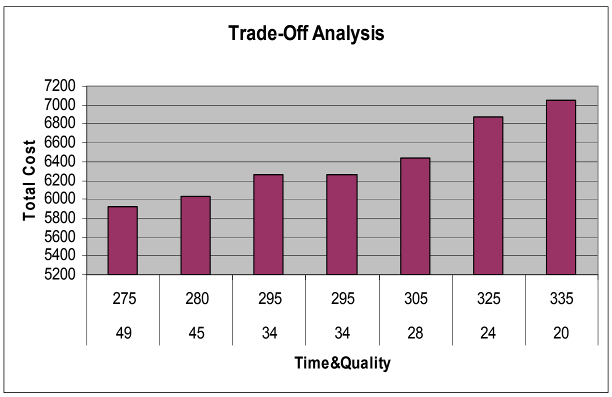

Fig 4: Time-cost-quality trade-off analysis

Figure 5 shows the processing time taken by IWD algorithm to reach the solutions under different quality bounds.

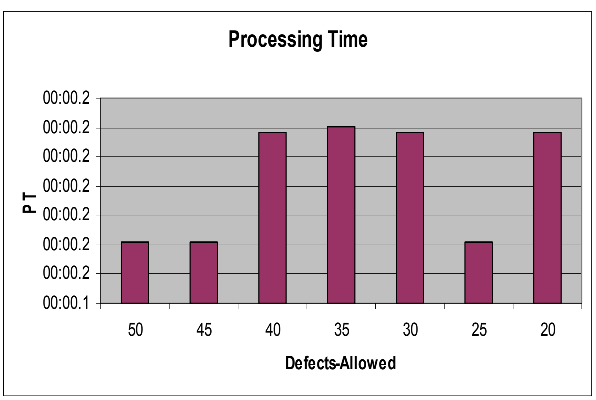

Fig 5: Processing time (TP)

8. CONCLUSIONS

In this paper, task quality is expressed by its defects and thus a total number of defects for all tasks reflect the quality of the software project quantitatively.

In DTCQTP, each project task/activity can be executed in one of several modes. The execution modes of any activity was assumed to be bids offered from different subcontractors or the software project to be developed is similar to previously developed projects repeatedly where each experience is one mode.

Solving the problem gave an optimal/nearly optimal solution in terms of time, cost, and quality of the project. By changing the allowable quality bound for the project and re-running the algorithm, other optimal solutions could be obtained .Having these optimal solutions and analyzing the environments needs, project managers could make decisions effectively.

To solve the problem, IWD algorithm was introduced, which takes somewhat less time to reach the optimal/nearly optimal solution under the allowable quality bound.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of Ain Shams University and Yemen government in supporting research authors.

REFERENCES

[1] Maryam Rahimi and Hossein Iranmanesh, “Multi Objective Particle Swarm Optimization for a Discrete Time, Cost and Quality Trade -off Problem”, World Applied Sciences Journal, vol.4, no.2, pp.270-276, 2008.

[2] N.Shahsavari Pour, M. Modarres, Mir B. Aryanejad, R. Tavakoli Moghadam, “The Discrete Time-Cost-Quality Trade-off Problem Using a Novel Hybrid Genetic Algorithm”, Applied Mathematical Sciences, Vol. 4, no. 42, pp. 2081 – 2094, 2010.

[3] Nasser Shahsavari Pour, Mohammad Modarres and Reza Tavakkoli-Moghaddam, “Time-Cost-Quality Trade-off in Project Scheduling with Linguistic Variables”, World Applied Sciences Journal 18 (3): 404-413, 2012.

[4] H. R. Tarighian and S. H. Taheri, “on the discrete time, cost and quality trade-off problem”, Applied Mathematics and Computation 181 (2006), 1305–1312.

[5] Refaat H. Abd El Razek, Ahmed M. Diab, Sherif M. Hafez, Remon F. Aziz, “Time-Cost-Quality Trade-off Software by using Simplified Genetic Algorithm for Typical repetitive Construction Projects”, World Academy of Science, Engineering and Technology 37, 2010.

[6] N. Ravi Shankar, M. M. K. Raju, G. Srikanth and P. Hima Bindu, “Time, Cost and Quality Trade-off Analysis in Construction of Projects”, Contemporary Engineering Sciences, Vol. 4, no. 6, pp.289 – 299, 2011.

[7] H. Iranmanesh, M. R. Skandari, and M. Allahverdiloo, “Finding Pareto Optimal Front for the Multi-Mode Time, Cost Quality Trade-off in Project Scheduling”, World Academy of Science, Engineering and Technology 40 2008.

[8] A. Afshar, A. Kaveh, O.R. Shoghli, “Multi-objective optimization of time-cost-quality using multi-colony ant algorithm”, Asian Journal of Civil Engineering (building and housing), vol. 8, NO. 2 (2007), Pages 113-124.

[9] E. Demeulemeester, B De Reyck, B Foubert, W Herroelen and M Vanhoucke, “New computational results on the discrete time/cost trade-off problem in project networks” , Journal of the Operational Research Society (1998) 49, 1153-1163

[10] Ahmed Baykal Hafizoglu, “Discrete time/cost tradeoff problem in project scheduling”, a thesis submitted to the graduate school of natural and applied sciences of Middle East technical university, June 2007.

[11] Mohamed H. Rasmy, Hesham M. E. Abdelsalam, Rania R. Hussein,“Multi-Objective Time-Cost Trade-Off Analysis in Critical Chain Project Networks Using Pareto Simulated Annealing”, March 2008.

[12] Amin Zeinalzadeh, “An Application of Mathematical Model to Time-cost Trade off Problem (Case Study)”, Australian Journal of Basic and Applied Sciences, pp. 208-214, 2011.

[13] Do Ba Khang and Yin Mon Myint, “Time, cost and quality trade-off in project management: a case study”, International Journal of Project Management, Vol. 17, No. 4, pp. 249-256, 1999.

[14] Roya M. Ahari and S. T. A. Niaki, “Fuzzy Optimization in Cost, Time and Quality Trade-off in Software Projects with Quality Obtained by Fuzzy Rule Base”, International Journal of Modeling and Optimization, Vol. 3, No. 2, April 2013.

[15] Babu, Suresh, “Project management with time, cost and quality considerations”, European Journal of Operational research, pp. 320-327, 1996.

[16] Ian Sommerville, “software engineering”, ninth edition, Addison-Wesley, 2011.

[17] Sunita Chulani, “results of Delphi for the defects introduction model (sub-model of the cost/quality model extension to COCOMO II), Center for software engineering, 1997.

[18] David Chappell, “the three aspects of software quality: functional, structural, and process”, available at http://www.davidchappell.com/writing/white_papers/The_Three_Aspects_of_Software_Quality_v1.0-Chappell.pdf (cited 9.11.2014).

[19] Hamed Shah-Hosseini, “The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm”, Int. J. Bio-Inspired Computation, Vol. 1, Nos. 1/2, 2009.

[20] Priti Aggarwal, Jaspreet Kaur Sidhu, Harish Kundra, “applications of intelligent water drops” , IISRO, Multi-Conference, Bangkok, 2013 .

AUTHOR PROFILES

Abdulelah Ghaleb Farhan Saif received BSc. in computer science from Thamar University (Thmar, Yemen), in 2004, and MSc. in computer science from Yarmouk University (Irbid, Jordan), in 2010. He is a research student of software engineering, evolutionary computation, and swarm intelligence. Currently, he is PhD student at Ain Shams University, Egypt.

Safia Abbas Mahmoed Abbas received BSc. in Pure Math and Computer Science from Ain Shams University (Cairo, Egyp), in 1998, and MSc. in computer science from Ain Shams University (Cairo, Egyp) in 2003 and Ph.D. in Computer Sciences from Niigata University (Niigata, Japan) in 2010. Her research interests in data mining. Currently, she is lecturer at Ain Shams University, Egypt.

Zaki Taha Ahmed Fayed received BSc. in 1976, Diploma in 1986 , MSc. in 1991 and Ph.D. in 1997. His research interests in software engineering and pattern recognition. Currently, he is Emeritus Professor at Ain Shams University, Egypt.