International Journal of the Physical Sciences Vol. 6(4), pp. 828-836, 18 February, 2011
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS10.623
ISSN 1992 - 1950 ©2011 Academic Journals

Full Length Research Paper

A genetic algorithms based approach for conflicts resolution in requirement

M. Ramzan[1], M. Qasim Khan[1], M. Amjad Iqbal[1], M. Aasem[1], Arfan Jaffar[2], Sajid Anwar[3]*, Awais Adnan[3], A. Tamleek[3], Muhammad Ali[3] and Masoom Alam[3]

[1]University Institute of Technology PMAS, University of Arid Agriculture, Rawalpindi, Pakistan.
[2]FAST-National University of Computer and Emerging Sciences, Islamabad, Pakistan.
[3]Institute of Management Sciences, Peshawar, Pakistan.

Accepted 28 January, 2011

With the improvements of technologies, software systems have been more complex then ever before. This requires new approaches to improve software development processes so that they sufficiently and efficiently meet these challenges. Requirement engineering in this regard facing many problems in which conflicts resolution gained very little popularity. In this paper we present motivation toward requirement conflicts resolution and discuss how resolution techniques from other domains can be applied in this area. We also proposed a new approach for conflicts resolution based on the idea of genetic algorithm. This model is applied to a case study to demonstrate how systematically resolution process can be optimized.

Key words: Genetic algorithm, conflict resolution, requirement engineering.

INTRODUCTION

A major problem in the requirement engineering is that it often faces immaturity and conflicts with each other (Lee et al. 1999; Robinson et al., 1990; Yen et al., 1999:4). Various stakeholders have different expectation and views about a system so they assign their own priorities to certain aspects. Development team also struggles to capture the right requirements which are critical and necessary for the project.

Robinson et al. (2003) describes three technical difficulties that lead to conflicts that is, voluminous requirements, changing requirements and analysts, complex requirement.

In the literature it is found that projects of different scope were failed due to poorly negotiated amongst stakeholders (Barry, 1996). The project team was unable to sort out the conflicts between stakeholders. Easterbrook et al., (1994) defines conflicts as something that occurs on the basis of goals and desires during project development. So it is necessary for users to negotiate at the right time with them and project team (Robinson et al., 1990) and find some solution for their conflicts (Nuseibeh, 1996).

In software engineering, the area of requirement engineering gets slow progress towards the automation. Many researchers try to automate its sub processes to take optimal results. Moreover the conflict resolution process still to be considered the research focus where a lot of work is needed.

The use of artificial techniques (AI) techniques in software engineering is an emerging area that will incorporate one domain in other (Srivastava and Kim, 2009). A lot of research is so far done in software testing using AI techniques and also evolutionary algorithms (Mansoor, 2004). But still, work is needed to apply evolutionary algorithms in requirements engineering. Genetic algorithm is considered as one of the significant and popular evolutionary computing technique from computational intelligence domain (Ramzan et al., 2010). In this paper, the approach of genetic algorithm (GA)

*Corresponding author. E-mail: sajid.anwar@imsciences.edu.pk.

were proposed to use it for the conflict resolution between stakeholders in requirement engineering, but the work still to be modified and need some automation.

So from the discussion it is observed that conflicts resolution is not a simple job amongst stakeholders. Different strategies were used for this purpose but these did not sufficiently provide statistical results that help the project managers in resolving conflicts. The aim of our study is that conflict resolution is still an optimization problem because from time to time the rate of conflicts must be reduced. So there is a need to use some optimizing techniques that reduces the number of iterations. A genetic algorithm based approach is our approach to reduce the time and costs that were spent in the conflict resolution process. Although genetic algorithms has several application in the field of software engineering but this will be the first attempt to use it for the purpose of negotiation and resolving conflicts. We will focus on the stakeholder's requirements that were supposed to be the conflicting requirements. Each conflicting requirement will be considered as chromosome. Random selecting from the set of population (chromosomes) and crossover and mutation process will be performed to create a new child (chromosome). These childs will be checked against a fitness function till termination condition.

Subsequently, a review of the study's literature which comprises the related material and resolving conflicts techniques and also some related work of GA is presented. Then the study's proposed model is given, after which a case study and findings on that study is given. Lastly, conclusions and future directions are suggested.

LITERATURE REVIEW

Here, we will first discuss the conflicts resolution techniques and its application and then genetic algorithm and its applications.

Conflicts resolution

One of the first methods that were used for negotiation and conflict resolution is known as theory X. In the theory it is understood that employees are lazy by nature and don't bother to work. As a result the management closely watched and supervised the workers at different levels. They are directed to work. Here the employees are given little control at each level. If a conflict arises between the employees then it is hard to find the solution. Managers are responsible to resolve the conflicts between them and decision making of what to follow is up to the management. Papa et al. (2008) describes that theory X managers has to force the employee to gain maximum output. The theory is management oriented and the main flaw here is that everything must be ended in the blame gaming. The employees have a negative point of view regarding this theory.

Due to its negative nature theory X was not so much popular. Another theory namely theory Y was introduced. In this theory the whole responsibilities were assigned to the employees. The management believes that those employees may be fully aware of their responsibilities and they can enjoy their work. In case of conflict between them on a certain problem, how to handle it is up to their own consideration. They were good enough trained to handle these situations. Papa et al. argues that work for them is as natural as to play. They have the capacity to solve the problems.

People make their own direction on their own choice. The theory Y manager believes that employees seek the understanding from time to time after given the right directions to them in the working environment. Employees considers positively about the theory Y.

Theory Z refers to Japans style management. The goals were set for them before they start working, when they were brought to their goals they were eager to perform. This theory is compatible with the Deming's 14 quality points. The loyalty of the employees provides a strong mechanism for achieving the project goals.

In theory W, conflicts resolution performs through mutual consideration. In the aforementioned theories one party has to lose his view after negotiation. In this theory everyone is considered to win. Different stakeholders are asked to bring their extreme points in the negotiating process and then a mediation take place. They all were brought to a common interest where each stakeholder and other participants were satisfied. Theory W makes everyone a winner due to its nature. Barry et al., (1996) discusses the problems faces by the project manager while resolving amongst different people of different fields that is, stakeholders, developers, customers etc. The problem is highlighted in Figure 1.

Pair wise comparison method (PCM)

This is a simple technique that is used for conflict resolution. This technique is mentioned for conflict solution in aspect oriented software development (Hameed, 2008). By nature we can illustrate it for the requirement conflicts. It is a formal technique. These steps are performed in the following technique (Rashid, 2004):

1. Draw a contribution matrix that contain list of requirements of the different stakeholders where each requirement relates to each other on the basis of positive and negative impact. Negative impact means that a certain requirement does not compliment any other requirement while positive impact is that both are complimenting each other.
2. Assign weights or values to the requirement according

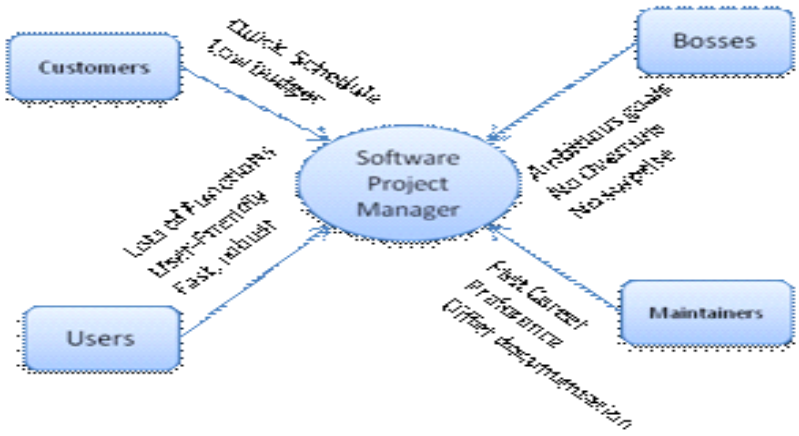


Figure 1. Software manager problems.

Table 1. Contribution Matrix.

Requirements	Req. 1	Req. 2	Req. 3	Req. 4	Req. 5
Req. 1		✓			✓
Req. 2	✓		✓	✓	
Req. 3	✓			✓	
Req. 4		✓		✓	✓
Req. 5		✓	✓		✓

to stakeholder's demands. These values may range from any numbers depending on the requirement needs. A matrix is obtained in which different stakeholders assign values in the matrix.

In the matrix the cell is marked with tick sign which indicates that the stakeholders relates its requirement with the cell and if the cell is empty it means that the stakeholders does not have any concern about the requirement (Table 1).

3. Solve the conflicts amongst different stakeholders giving priorities to the conflicts. If a specific requirement coincides with another requirement then the mark sign in the contribution matrix is replaced by the weights. These weights were initially assigned by the stakeholders. So give high priority on the basis of greater weights.

Another technique is Winwin Model it is used for resolving conflicts in requirements within stakeholders. It provides win conditions, issues that are raised in conflicts, options available and negotiating agreements (Ln et al., 2005). This model uses theory W (Barry et al., 1989) Make everyone a winner" for resolving conflicts between stakeholders through spiral model. They were brought to a common interest point where each party is satisfied. Winwin situation is occurred for everyone where they did not lose anything (Figure 2). In step 1 stakeholder, it starts with their win condition. If any conflicts are identified in step 2 then step 3 will occur. Here options for the conflicts and resolving strategies were discussed. The problems are evaluated with consensus. And in the step 4 the agreement is finalized between the stakeholders. So it is an incremental and spiral model. If after final agreement any conflict arises again then the step 1 is initiated (Figure 2).

Subsequently, we will discuss the genetic algorithm and its application in the field of software engineering.

Genetic algorithm

Genetic algorithms are naturally inspired algorithms adopted from Darwin's theory of evolution. The approach of GA comprises of several steps. First, is the initial selection of chromosome from population. A population consists of chromosomes. In the next step cross over on the selected chromosomes is performed and a slight change (mutation) after cross over in the child chromosome depicts genetic diversity. Each child is checked against a fitness function that is defined for the problems solution.

Mostly GA has been applied to the scheduling and optimization problems and for searching problems like TSP. optimization in this regard is an algorithmic sense that is not performed in most cases (Braune et al., 2008). In software engineering GA application are only up to the experimental results still theoretical proofs to be needed

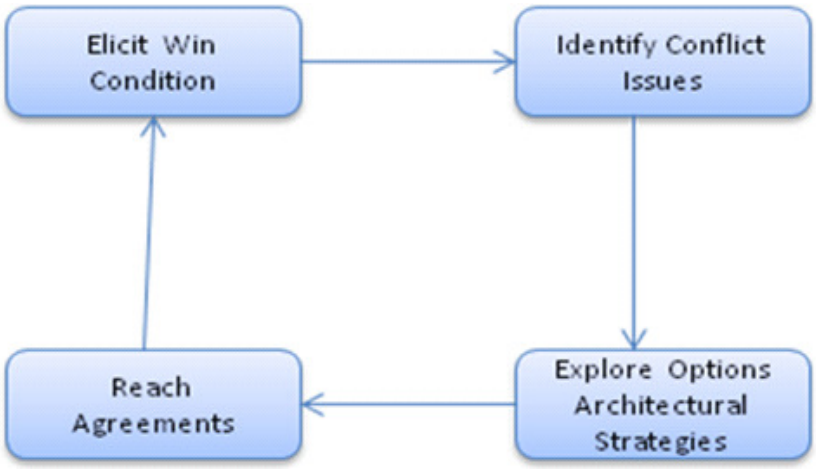


Figure 2. Winwin model.

(Jiang, 2006).

Genetic is applied for the school time table problem (Raghavjee et al., 2008). The GA refines implement initial population of timetables. Then cross over and mutation process evaluates the good quality feasible timetable. The result of the GA based time table was compared to performance parameters of neural networks, simulated annealing, tabu search and greedy algorithms. It is found that the outcome of the GA is most efficient. Software testing is a crucial part of the software development. The feasible test paths generation is a tired job. Several attempts were made to generate automated test paths. The use of GA in test data generation was exciting approach and uses GA in the generation of test data for the testing (Srivastava et al., 2009).

Genetic algorithms application in the field of testing and automatic test path generation is widely described in the research work (Jiang et al., 2006; Lee et al., 1999; Robinson, 1990; Yen et al., 1994). Some research work is also done in the project scheduling when different conflicts arise between actives of the project (Barry et al., 1989).

Frederic et al. (1996) describes conflict resolution in air traffic control which is also an optimization problem. It provides the mechanism to reduce the conflict between the air craft to land and take off. Same type of research work is also presented where air born conflict management is performed using the genetic algorithms (papa et al., 2008).

So from the discussion it is observed that the genetic algorithm is a useful tool for optimization and scheduling problems. Now the point of concern is that can we apply it for the software engineering problems. First of all we have point of those areas where we can found some type of optimization and still better results can be achieved. The possible factor of how GA can be a better choice for software engineering problems (Jiang, 2006). The authors discuss that work done in the area comprises of statistical results and still theoretical proofs are required. The author discusses the approaches that we have to identify and classify the software engineering problems so that we can provide the correct procedure for optimal solutions. In our previous work our emphasis was on the conflicts that were found after elicitation phase. Those conflicts were treated differently by different researchers, but from the study it is clearly observed that number of conflicts reduces after each negotiation process, so we can consider it also an optimization problem. Because the actual requirements increases to optimal point after resolution process. So we have to define and derive relation that conflict resolution is an optimization problem. We will discuss that in the subsequently.

A GA just starts its process by guessing the initial selection from the set of population. The process comprises of the following steps that is, initial selection, cross over and mutation operators and fitness function. A simple algorithm is presented below:

Initialize (population)
Evaluate (population)
While (stopping condition not satisfied) do

Selection (population)
Crossover (population)
Mutate (population)
Evaluate (population)

Each chromosome in the population is an element of data set. Randomly generates the initial population which evaluates its fitness function and applied crossover and mutation rules for the production of next generation.

The algorithm will perform its iteration until the chromosomes finds out the best solution for the problems or until a maximum number of iterations have taken place (suggesting that a solution is not going to be found given the resources available).

Table 2. Conflict identification table.

	A1	A2	A3	Aj-1	Aj	……	Am
R1	V(R1, A1)	V(R1,A2)					V(R1, Am)
R2	V(R2, A1)						
Ri-1				V'(Ri-1, Aj-1)			
Ri					V(Ri, Aj)		
Rn	V(Rn, A1)						V(Rn, Am)

```
Take initial activity V (Ri, Aj) as input
Take conflicting criteria CR
Repeat (until there is least number of conflicts
V (Ri, Aj) - V' (Ri, Aj) > minimum _No._ criteria)
Begin
        Initialize Conflict Identification
        Evaluate every activity against all others
        Identify conflicts and list Conflicts
        Initialize Conflict Resolution
        Evaluate each activity based on its conflicts
        Calculate fitness
                Select worst activities for regeneration
        Apply reproduction operators
        Apply Crossover
        Apply Mutation
        Evaluate new Activities
END
Terminate
```

Figure 3. Basic algorithm.

PROPOSED WORK

First of all we have to give statistical proof that conflict resolution in requirements engineering is still an optimization problem between the requirements those requirements.

Let us consider a set requirements R

$$\mathbf{R} = \{\mathbf{req.1}, \mathbf{req.2}, \mathbf{req.3}, \mathbf{req.4} \ldots\ldots\ldots\ldots\ldots\ldots \mathbf{req.n}\}$$

And also we have a set of conflicting requirements Cr,

$$\mathbf{Cr} = \{\mathbf{Cr.1}, \mathbf{Cr.2}, \mathbf{Cr.3} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{Cr.m}\}$$

The set of Cr will always reduced when we apply the conflict resolution process on it. The number of conflicting requirements will minimize with every iteration of the process.

So as Cr is subset of R
As Cr ≈ 0
Set R is called optimized when Cr is minimized.

Therefore we can say that conflict resolution is also an optimization problem.

Let us consider an example, we have a number of activities that are ranges from A1, A2, Am. These activities are different from one another according to the project scope. These activities will be performed on different number of requirements at different stages. These requirements list will range from R1, R2…Rn. A set of activities V (Ri, Aj) will be performed on different intervals. These sets of activities will be checked against the rule set or conflicting criteria. If any activity V (Ri, Aj) does satisfy the conflicting criteria (CR) then we will call it a conflict V' (Ri, Aj). The following scenario can be discussed as follows (Table 2):

Where; R1, R2,……., Rn are the requirements
A1, A2……An are the activities.
V (Ri, Aj) are set of activities performed on Ri using Aj
V' (Ri-1, Aj) are set of conflicting requirements that satisfies one are more conflicting criteria. (CR)
Where CR is a set of predefined rules that declares a V (Ri, Aj) a conflicting requirements V' (Ri, Aj).

Table 3. Crossover and mutation process.

Case	Req. No.	Client avg.	Crossover + Mutation	Client 1	Client 2	Client 3	Total	Fitness
A	3 7	8.5	Save record. Shall be deleted after n days and take the backup.	4	4	3	11	11>8.5 Accepted
B	4 8 12	10.6	Administrator should set the options for logins that is, some users can login many time at a time.	4	4	5	13	13>10.6 Accepted
C	9 14	7.5	Administrator should set the flag at the start of giving password for first time. The users should set their own option.	5	5	5	15	15>7.5 Accepted

Table 4. Fitness function.

Strongly rejected	1
Rejected conditionally	2
Partially accepted	3
Mostly accepted	4
Accepted completely	5

Table 5. Criteria for client's weight-age.

Req. No.	Description	Client 1	Client 2	Client 3	Total weight
3	Unsuccessful login shall be stored in database for security purpose.	5	1	3	9
7	Do not save the unsuccessful attempt in database for performance purpose.	1	5	2	8
4	No more than one person can login at one time.	5	1	3	9
8	More than one person can login at the same time.	2	5	4	11
12	Some users were allowed while some were not allowed.	3	4	5	12
9	Password shall be updated after specific period of time otherwise account will be locked.	2	5	1	8
14	It should be remained permanent and can be changed on request.	1	1	5	7

CASE STUDY

A case study has been given based on conflicting requirements that was identified. These conflicting requirements are given subsequently. Three clients have given weight-age according to the defined criteria. The weight-age criteria are given in the Table 5. The client can give their weight from 1 to 5 illustrated in Table 4. The weight of each client is added in as total weight against a certain requirement.

In the Table 3 the process of cross over and mutation process is performed for resolving conflicts. First of all we will take the average of the conflicts requirements. After the process of cross over and mutation once again the clients were asked to give their weight for the specific resolved conflict. The total of their sum must be greater than the client average that is our fitness function (Table 6). If the condition get false than we will repeat the process for suitable child (requirement). Our conflicting requirements are illustrated in Table 7 and the process is shown in Figures 4 and 5.

By plotting our values (Figure 6), we can easily understand that the required solution has a high percentage of acceptance.

CONCLUSION AND FUTURE WORK

At requirement engineering process, different stakeholder have their own point of view and expectation about the system. Conflicts arises when they present their demands. At this stage it is difficult to satisfy each stakeholder. Using genetic algorithm for resolving conflicts will reduce the cost of manual solution of the

Table 6. Fitness function condition.

A	If a > Cv ,	$a = \sum_{i=1}^{3} Ci = 11$ where 1<a>n, where n= No. of sum of values of requirements 11 > 8.5 --Accepted	**Equation 1**
B	If a > Cv ,	$a = \sum_{i=1}^{3} Ci = 13$ where 1<a>n, where n= No. of sum of values of requirements 13 > 10.6 --Accepted	**Equation 2**
C	If a > Cv ,	$a = \sum_{i=1}^{3} Ci = 15$ where 1<a>n , where n= No. of sum of values of requirements 15> 7.5 -- Accepted	**Equation 3**

Table 7. Conflict identification table.

Conflict ID	Req. #	Initiator	Action	Consequence	Dependency	Effect
1	3	User	Login	Not successful	-	Stored in DB
1	7	User	Login	Not successful	-	Do not save in DB
1	2	Administrator	customization in sw	Updated	-	Update the S/W
2	4	1 User	Login	Allowed	-	Successful Login
2	8	More than 1 user	Login	Allowed	-	Successful Login
2	12	Some Users	Login	Allowed	-	Successful Login
2	11	Some Users	Request	Granted	-	Report Generated
2	10	User	Request	Modification	-	Not Granted
3	9	System	update Password	New Password	-	New Password Generated
3	14	User	update Password	New Password	-	New Password Generated
3	15	System	update user profile	Profile updated	-	Updated the user profile
3	16	User	Entry	Record Entered	-	Record updated

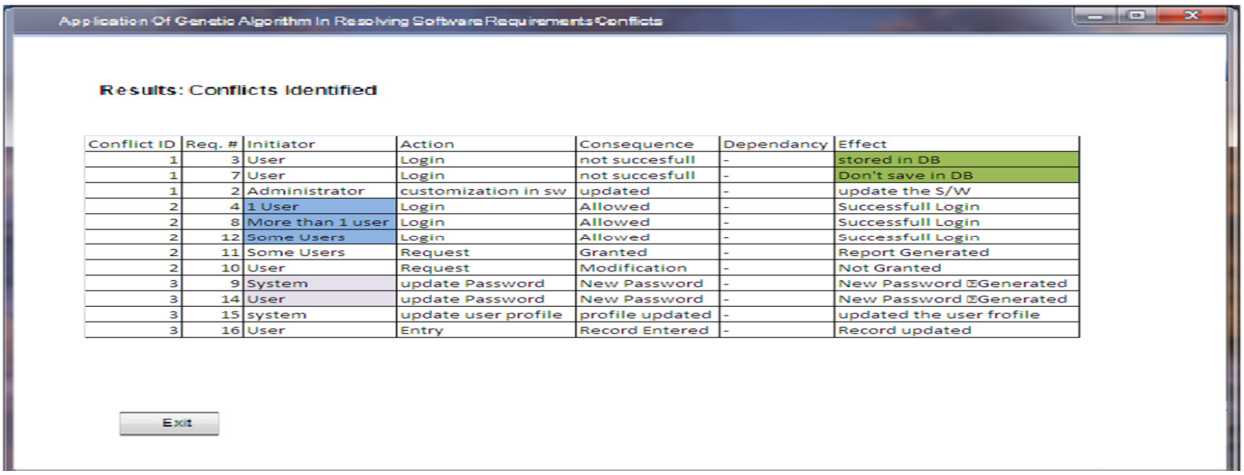
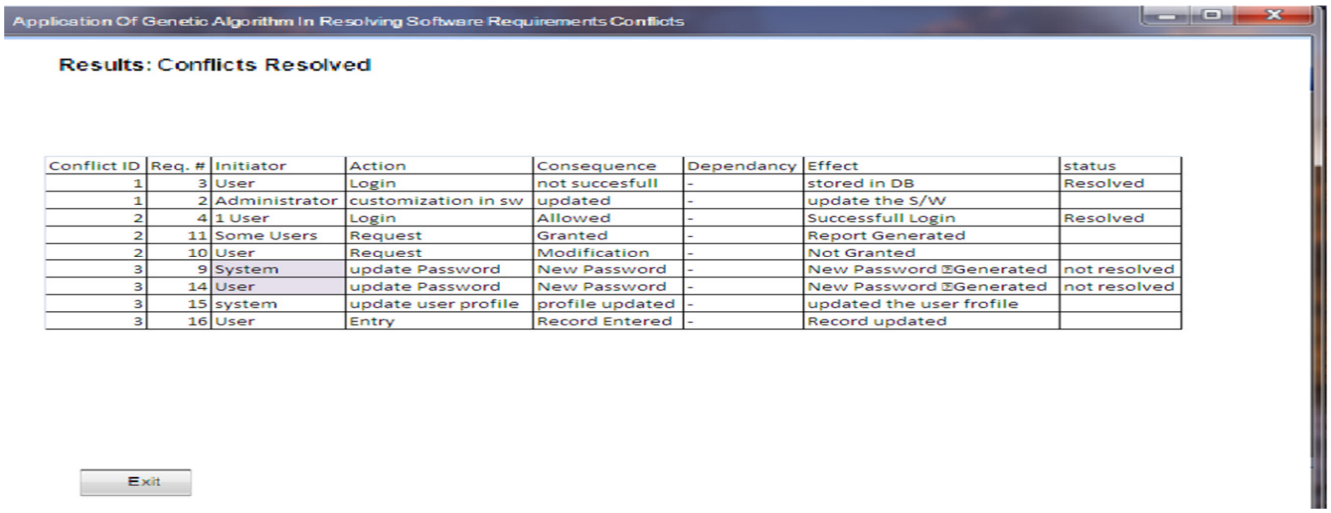


Figure 4. Conflicts identified.



Conflict ID	Req. #	Initiator	Action	Consequence	Dependancy	Effect	status
1	3	User	Login	not succesfull	-	stored in DB	Resolved
1	2	Administrator	customization in sw	updated	-	update the S/W	
2	4	1 User	Login	Allowed	-	Successfull Login	Resolved
2	11	Some Users	Request	Granted	-	Report Generated	
2	10	User	Request	Modification	-	Not Granted	
3	9	System	update Password	New Password	-	New Password Generated	not resolved
3	14	User	update Password	New Password	-	New Password Generated	not resolved
3	15	system	update user profile	profile updated	-	updated the user frofile	
3	16	User	Entry	Record Entered	-	Record updated	

Figure 5. Conflict resolved.

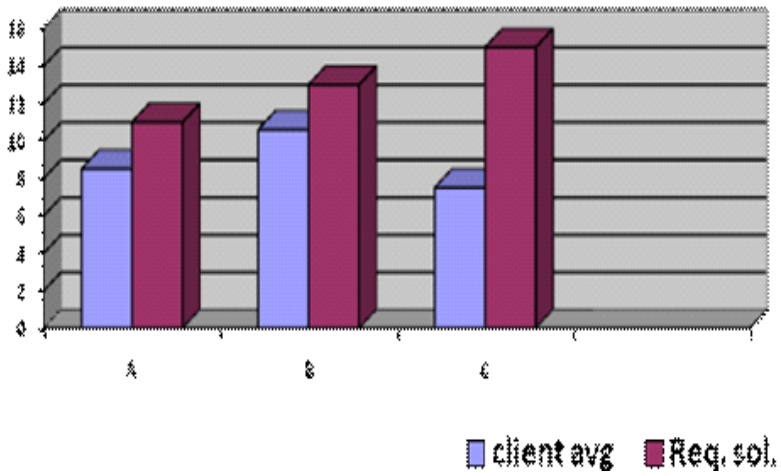


Figure 6. Graph between client average and required solution.

conflicts amongst different stakeholder and this process will become more systematic to some extent. It will also reduce the amount of risk. Resolving conflicts is a tedious job and our proposed model help project managers to sort out these conflicts. We use a GA approach to solve these conflicts. A case study is applied to validate our approach. We use the GA approach for resolving the conflicts. Our emphasis in this research work is to automate our previous work based on the idea of genetic algorithms. We transform the actual requirements into chromosomes and seeded it to our program. The results of the program were efficient and accurate. We have observed that the optimal results can be achieved if we further trained our GA. In the future work our emphasis will be to test our tools against large number of requirements and check its results accuracy.

REFERENCES

Barry B, Rony R (1989). Theory-W Software Project, Management and Examples. IEEE Trans. on soft. Eng., 15(7):

Barry B, In H (1996). Identifying Quality-Requirement Conflicts. IEEE Software, 13: 25-35.

Braune R, Wagner S, Affenzeller M (2008). Applying Genetic Algorithm for the optimization of production planning in real World manufacturing environment.

Rashid A, Moreira A, Araujo J (2004). Modularization and Composition of aspectual requirements. AOSD 2003, AOSD'03 Proceedings of the 2nd international conference on Aspect-Oriented software development, ACM 2003 1-58113-660.

Frederic M, Durand N, Alliot JM (1996). Air traffic conflict Resolution be genetic algorithm. Lecture Notes Comput. Sci., 1063: 370-383.

Easterbrook S (1994). Resolving Requirements Conflicts with Computer-Supported Negotiation, Academic Press, Requirements Eng., pp. 41-65.

Hameed F (2008). Model for conflict resolution in aspects within Aspect Oriented Requirement engineering. Master Thesis Software

Engineering thesis no: MSE-2008-11,BTH, Sweden.
Jiang H (2006). Can the Genetic Algorithm be a good tool for Software engineering searching problems? COMPSAC apos; 06. 30th Ann. Int., 17-21: 362–366.
In H, Kazman R, David O (2005). From requirements Negotiations to software architecture decisions. 47(8): 511-520.
Lee J, Kuo JY, Huang WT (1999). Classifying, Analyzing, And Representing Informal Requirements." Proc. Sixth Int. Fuzzy Syst. Assn. World Congr., 1: 645–648.
Mansoor N, Salame M (2004). Data Generation for Path Testing, Software Qual. J., 12: 121–136.
Nuseibeh B (1996). Conflicting Requirements: When the Customer Is Not Always Right. Requirements Eng., 1(1): 70-71.
Ramzan M, Iqbal A, Jaffer A, Rauf A, Sajid A, Shahid A (2010). Project scheduling Conflict Identification and Resolution using Genetic Algorithm. Inf. Sci. Appl. (ICISA)., pp. 1–6.
Raghavjee R, Pillay N (2008). An Application of Genetic Algorithms to the School Timetabling Problem.
Robinson WN (1990). Negotiation Behavior during Requirement Specification. Proc. Int. Conf. Software Eng., 268–276.
Robinson WN, Suzanne dP, Vecheslav V, (2003). Requirements Interaction Management. ACM Comput. Surv., 35(2): 132–190.
Srivastava P, Kim TH (2009). Application of Genetic Algorithms in Software testing. Int. J. Software Eng. Appl., 3(4): 87-95.
Srivastava P, Ranjan P, Gupta Y, Arrawatia SY (2009). Use of Genetic Algorithms in generation of feasible test data.
Papa MJ, Daniels TD, Spiker (2008). Organizational communication: Perspectives and trends. Thousand Oaks, CA: Sage.
Yen J, Liu X, Teh SH (1994). A Fuzzy Logic-Based Methodology for the Acquisition and Analysis of Imprecise Requirement." Concurr. Eng. Res. Appl., pp. 265–277.