A)

1. PRISONER'S DILEMMA IN SOFTWARE TESTING

BRIEF EXPLANATION :

ABSTRACT :

In this article, we will be considering the model of software engineering as a strategic game, which is essentially equivalent to Prisoner's dilemma in case of certain values of the productivity and reward parameters. This signifies that the game has an unique Nash equilibrium, which is not optimal for both players.

INTRODUCTION :

Software Testing is used to find errors so that they can be repaired and hopefully, the software quality improved. Here we will be considering software quality as an economic issue and will be applying rational methods. The problem of Software testing is modeled as a formal strategic game.

SOFTWARE TESTING :

Software Testing is used in serious attempts to improve the situation by detecting errors as soon as possible. There is an opposition of interests between the tester and the implementor. The goal of the tester is to find errors, whereas an implementor likes his program to be shown correct.

AN IDEALISED TESTING GAME :

The motivation for this game is The implementor is rewarded if he delivers an error-free software and the tester is rewarded if he performs a thorough testing job . Here p means 'poor'(bad) and q means 'quality' (good). So the implementor chooses between doing a poor job and doing a high quality job, delivering error-free software and the tester chooses between doing a poor job, performing a test with low error detection capability and performing a quality test, which takes more effort.

The payoff matrix of ITG is the payoff matrix of a Prisoner's Dilemma. There is one N.E., the action profile (q,q). In the N.E. both players choose to do a quality job.

	p	q
p	2,2	0,3
q	3,0	1,1

Adding an extra performance level :

As a model of an implementer's behavior and a tester's behavior is the binary choice with respect to the performance level. Perhaps a real implementer does not want to choose between two extreme values of 'poor' and 'quality' levels, or between Q = 20% and Q = 50%. But will the dilemma disappear if there is a third, intermediate level? In order to investigate this, we construct another game called ITG3 which is a 3 x 3 strategic two-player game. Therefore the refined payoff matrix is:

	p	pq	q
p	4,0 /0,4	4,–0.5 /0,3.5	4,–1/0,3
pq	3.5,0/–.5,4	3.5,–0.5/–0.5,3.5	3.5,–1/–0.5,3
q	3,0 /–1,4	3,–0.5 /–1,3.5	3,–1/–1,3

CONCLUSION :

The results of our investigations show that if the reward for passing tests for the programmer is high enough and the reward for finding an error for the tester is high enough, there is essentially a Prisoner's Dilemma game hidden in the interaction of a programmer and a tester.

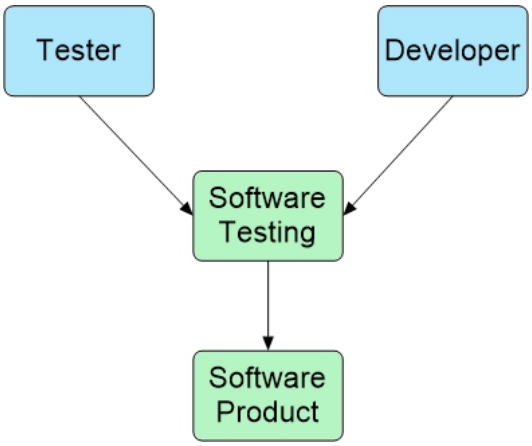
It seems wise for a software development lab manager to make sure that he separates the roles of programmers and testers and that he makes sure there is enough reward for a programmer to achieve PASSes and for a tester to achieve FAILs.

B)

POSSIBLE EXTENSIONS OF PAPER 1

Implementation in software testing :-

Software testing is a process which is carried out to give knowledge about the qualityof the software product to the stakeholder.



Even though testing cannot detect all the possible errors in the software as exhaustive testing is not possible it can check for error in major functionalities.
There are two members are involved which are developer and the tester in softwaretesting.

Problems faced in software testing :

Errors can occur at the time of requirement gathering, while creating or choosing architecture, selecting wrong development model this causes a huge amount in repairand maintenance job.

Players / Variables	Developer (D)	Tester (T)
Price	0	0
Speed	-1	-1
Total	1	1

Summary :

In the above table the values are -1, 0, and 1. When a player is interested to increaseavalue of some variable it is denoted by 1. When a player is not bothered about the value about the variable it is denoted by 0 and when the player wants to reduce the value of the variable it is denoted by -1.

According to game theory a game is non-cooperative if it satisfies two conditions, if there is no possibility of sharing the payoff. And all the players are making decisionsfor themselves.

Equilibrium and payoff :

In order to achieve the balance in equilibrium there should be balance in payoff. Thepayoff can be made in such a way that whenever a successful game outcome comes,all the players win in some way. This is easy to say but difficult to achieve scenario.All the credit of Successful outcome of Software testing game is not given to Developer but the blame is always given to Developer.

Player T should stop testing certain scenarios after certain successful test cases, which will lead to reduction in total development time of the Software testing game. Player D should try to keep the quality of work up to the mark as stated and acceptedin requirement documentation. This will help player D to maintain own image in market.

Conclusion :

When all the players are working for the success of the Software testing game alongwith own goals it will have more chances of successful outcome. So these results inmaximum balance between players which will in turn result into higher possibilities of getting a successful outcome from the Software testing game.

C)

2. ANALYZING SOFTWARE DEVELOPMENT AS A NONCOOPERATIVE GAME

Abstract :

In this paper we just analyzed the causes which occurs in the process of Software Development by the technique of game theory to uncover the failures of Software projects.

Problems faced :

The main failure of the software project is taking a wrong requirements gathering and making various error in the specification of the code and choosing a different types of the software architecture, so these type of failures or causes will takes wasteof time and spends more cost to buy the requirement and the completion of the project will not be successful. And also when we analyzing conflicts between the goals of individual team members and general software engineering goals, predictingthem, and developing a strategy to eliminate these conflicts. We offer an approach that solves this fundamental problem and enables us to predict software project failures and even avoid them.

Strategy played in software development :

So to overcome these type of problems we just introduced the concept of gametheory in which it plays a major role in the statement of software development. To make a software development successful they need M, C, D which refer as Management, Customers and Players.

Players involved in strategy :

So overall Management strategy comprises three goals: which refers to :

1. Decrease cost of the software product.
2. Increase its price.
3. Ship the product faster.

Customer strategy revolves around mainly three goals:

1. Decrease the price of the product.
2. Increase a number of features that it offers.
3. Purchase a higher quality product for a smaller price.

While players strategy is less than clear. While Management and Customer havewell-defined financial goals based on the successful outcome of the software development game,

1. Player goals are to satisfy job requirements.
2. Increase personal marketability.
3. Increase Management and Customer's dependence on the players.

Payoff :

Three columns specify SD game players, i.e. M, C, and D.

Value -1 means that a player is not interested in increasing the value of that variable.Value 0 means that a player is neutral on that variable.

Value 1 mean that a player is interested in increasing the value of that variable.

Player / Variable	M	C	D
Cost	-1	0	1
Price	1	-1	0
Speed	1	1	-1
Features	1	1	-1
Quality	0	1	-1
	2	2	-2

Summary :

Summing up the values of M,C & D.M

and C gets 2 while D gets -2.

This indicates that while M and D end up with positive payoffs, the payoff for D inthe same game is negative.

Equilibrium for SD Games :

Each player knows the unspoken rules for maximizing payoff for M and C. M knows that D cannot be relied upon especially if D knows how M maximizes its payoff. Managers try to hire mostly fresh college graduates for large-scale projects.As irrational as it seems this action is perfectly rational. Each player knows the unspoken rules for maximizing.

Software Complexity and SD Games :

Management was trying to maximize its payoff and wanted to reduce the cost of theproject by laying off a number of developers.

Managers first let go employees whose systems were less buggy and required less maintenance.

.

Conclusion :

By using these type of technique the software development will get succeed in the market. As well as the customer, management and the team workers who worked inthe software development will get succeed.

D)

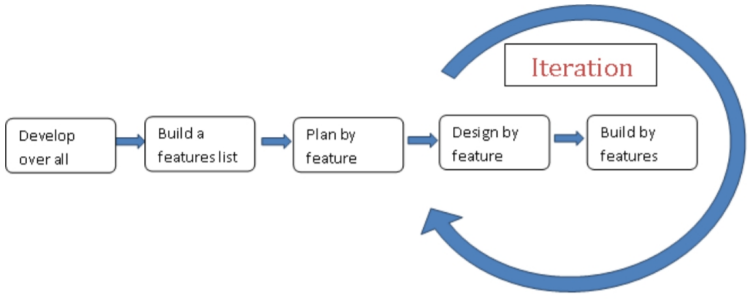
POSSIBLE EXTENSIONS OF PAPER 2

1) DEVELOPING USING AGILE METHODOLOGY :

Agile is software product development method to assemble a software product incrementally using short emphases so that the development is aligned with the changing industry demands.It has changed the landscape of modern software engineering and commercial software development. The Agile software development consists of characteristics like iterative, modularity, people oriented , time bound, no managers, open to change, satisfaction of customer and adaptive. Agile development contains different methods like programming at extreme level, scrum, crystal, (FDD) feature-driven-development.

Features Driven Development (FDD) :

The name suggests, features are main part of feature driven development. During project starting, one should focus and figure out the fundamental of the domain that the system has. Further next step is to find out the features of your system and make list of it, organizing them into associated sets and subject areas. Basic roles of FDD are Project manager, tester, Implementer or Developer.



BENEFITS AND COMPLICATIONS OF USING AGILE METHODOLOGIES :

- ✓ Agile makes groups increasingly productive at completing their work. This may give the manager a positive payoff as the product may reach the market soon.
- ✓ Agile development is iterative, it means that development is fast and early, a few iterations ensure a functional “ready to market” product.
- ✓ As it is making the process complex, implementer naturally maximizes the dependency of M on D.

- ✓ This greatly reduces the cost of production, which is exactly what the manager wants.
- ✓ It can be tough for bigger teams to be as compatible as smaller teams in concern with design and architectural changes. So, if the project itself is complex, then this method cannot be applicable.

2) MAKING IT PARETO OPTIMAL :

An **outcome** is said to be **Pareto optimal** if it cannot be dominated by any other **outcome**. To be specific on choosing a **Pareto outcome**, it is evident that no other **outcome** can prove to be better than this **outcome** for all the players.

In order to make it pareto optimal, the company should give shares or stocks equally to both the developer and the manager breaking the general rule that employee at an higher position gets higher payoff than the one at lower job role. Job security can be assured to the developers so that they can write less complicated and high quality program to complete their task.

Thus after making the non - cooperative game of software development pareto optimal, all three players gets a positive payoff.

CONCLUSION :

From this paper, we can come to a conclusion that giving equal company shares to both the manager and the developer, makes the game pareto optimal and thus every player gets satisfied with their respective payoffs. Following Agile methodology also helps both the manager and the developer in certain ways that helps both to get a positive payoff.