December 2005

Study to Secure Reliability of Measurement Data through Application of Mathematics Theory

Sang-Pok Ko
Mechatronics Samsung Electronics Co. Ltd

Byeong-Kap Choi
Mechatronics Samsung Electronics Co. Ltd

Hak-Yong Kim
Mechatronics Samsung Electronics Co. Ltd

Yong-Shik Kim
Mechatronics Samsung Electronics Co.

Yong-Shik Kim
Mechatronics Samsung Electronics Co. Ltd

See next page for additional authors

Authors
Sang-Pok Ko, Byeong-Kap Choi, Hak-Yong Kim, Yong-Shik Kim, Yong-Shik Kim, and Kyung-Whan Lee

Study to Secure Reliability of Measurement Data through Application of Mathematics Theory

Sang-Pok Ko, Byeong-Kap Choi,
Hak-Yong Kim, Yong-Shik Kim
Mechatronics Samsung Electronics Co., Ltd
416 Maetan-3Dong, Suwon City Korea
{sangpok.ko|bkchoi7|hakyong.kim|dys.kim}@samsung.com

Kyung-Whan Lee
Dept. of Computer Science
221 Huksuk-Dong, Seoul, Korea
kwlee@object.cau.ac.kr

Abstract

Recently, many corporations have applied CMM, SPICE or other software process maturity models to developing software products. These models are recommended to change from qualitative process management to quantitative management as they reach a certain level of maturity. Quantitative process management is possible only when reliable data exist. If quantitative process management are conducted when data's reliability are not ensured, adverse effects can be caused by the false management activities. That is why so many methods have been studied to analyze reliability of collected data. Most of them, however, are analysis methods for already collected data. Therefore, if the analysis finds that the existing data are not reliable, the data should be abandoned and new data be collected again, causing need to consume much time and effort. Therefore, as a way to secure reliability of data to be used for quantitative process management when they are input in the first place, this paper suggests a method applying the second level meta game of non-cooperative, bimatrix games to collecting data for quantitative measurement.

Keywords: S/W Measurement, S/W Metrics, S/W Quality, Game theory, Quantitative management, Software engineering.

1. Introduction

One object of software engineering is "producing high-quality products at minimum cost, within a short period."[1]. To this end, many methods and tools, models have been developed. Among them are CMM(Capability Maturity Model)[2] of SEI(Software Engineering Institute) and SPICE(Software Process Improvement & Capability determination)[3] of ISO/IEC 15504 TR2(Technical Report Type 2). Those models recommend that, if maturity of process reaches a certain level, metrics of development be quantitatively measured and used for quantitative management.

Quantitative management is a technique to perform a task by representing performance data in numbers and setting the quantitative goals based on the numbers. In quantitative management, most important is reliability of collected data. If reliability of the data is not guaranteed, the quantitative management based on the data is also meaningless.

Many methods have been studied to analyze reliability of collected data. Most of them, however, are just analysis methods for already collected data. Therefore, this paper suggests a technique applying a game theory to collecting input data as a way to secure reliability of data to be collected.

This paper consists of following parts: Chapter2 outlines of measure, game theory in S/W engineering, and business psychology; Chapter3 reviews obstacles to data collecting and

analysis of their weight values; Chapter4 deals with the technique applying a game theory to collecting data to be measured in order to get reliable data; Chapter5 demonstrates the technique described in the forth chapter through simulation; finally, Chapter 6 concludes this paper and provides study direction for the future.

2. Related Research

2.1 Measure in Software Engineering

In software engineering, measure is used for three purposes: for software process improvement; for project management involving estimation, quality management, productivity evaluation, and project control; as a data to make a technical decision when implementing a project.

Lord Kelvin once said, "When you can measure what you are speaking about, and express it in numbers, you know something about it. But when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind." [4] This means that software is hard to manage the development process, compared to hardware, because software development process, in much part, is highly dependant on people and thus, the measure is mainly qualitative. But if managed by quantitative measure, it can be managed in effective and systematic way.

2.2 Software Process Model

CMM and SPICE represent software process models. CMM consists of 5 maturity levels from 1 to 5: level 1 is Initial; level 2 is Managed; level 3 is Defined; level 4 is Quantitatively Managed; level 5 is Optimizing. On the other hand, SPICE have 6 maturity levels from 0 to 6: level 0 is Incomplete; level 1 is Performed; level 2 is Managed; level 3 is Established; level 4 is Predictable; level 5 is Optimizing.

The maturity levels of the two models are actually similar in meaning although they are referred to differently and SPICE has level 0 while CMM doesn't have. In both of the models, qualitative estimation by operating the process with certain special qualities is possible up to level 3. But from level 4, the process performance data should be measured to help understand the present state, improve the process through analysis of defects and removal of the causes, and facilitate quantitative estimation and future decision-making.

2.3 Game Theory

Game theory can be defined as a study of mathematical models about conflict and cooperation between rational decision-makers. Game theory using mathematical technique was designed to analyze situations when more than 2 parties should make a decision which will have an effect on their own interest under various possibilities of cooperation or conflict. This theory was first introduced in 1944 by a physicist and mathematician, Jon von Neumann and economist Oskar Morgenstern.

A game consists of player (at least 2 players), player's strategy, outcome of decision, and payoff, numerical value of the outcome.

Outcome is dictated by who selects which strategy. Therefore, each competitor should select a strategy which can maximize his or her own interest (performance) no matter what strategy other competitor(s) will employ.

According to number of competitors, games are classified into two types: two person game (e.g. chess) and multi-person game or n-person game (e.g. poker). The most common form of games is two person zero-sum game in which the sum of all payoffs to all players is zero because one player's winning necessarily means the other's loss if there is conflicting interest between players. According to number of strategies employed by players, games are also classified into two types: finite game and infinite game (for the purpose of continuing the play). Finite two person zero sum games[8] are most widely used in theory.

This paper applies the second level meta game of non-cooperative bimatrix games, to data collecting in order to secure reliability of data to be collected and measured in developing software.

2.4 Business Psychology

Psychological factors among developers also affect developing software. For leaders of software developing organizations, it is a difficult job to induce software developers to welcome additional work of collecting metrics, other than developing software. It is pointless to attribute all of this to young generation's individualism or selfishness. Leaders of organizations should understand the group psychology in the situation[11][12][13].

For instance, when a person hears the voice of somebody with epilepsy from the next room, if he is alone, the probability for him to give help is 85%. But, when he knows another 4 people, besides him, hear the voice, the probability will be reduced to 31%. In addition, longer time will be taken to take an action. This is a very well-known experiment in psychology field[7].

By the same principle, metrics collected in small group and in large group are somewhat different in terms of reliability. That is because individuals tend to feel less sense of responsibility in larger group due to diffusion of responsibility. One of psychologies fanning this trend is bystander effect in which people think, "Someone else will take responsibility" and want to stay low profile in the group. Therefore, it is needed to secure reliability of input data by motivating developers more actively and effectively, instead of just waiting until they input accurate metrics data to be measured.

3. Collecting Measurement Data

One of the biggest differences of software and hardware development is the level of visibility provided. For example, design and implementation of software are represented as conceptual modeling and logical codes. But in the case of hardware, it is designed intuitively using 3 dimensional CAD and implemented through combination of physical matters instead of logical codes, securing enough visibility.

Therefore, as a way to secure visibility and transparency in software development, it is strongly recommended to measure relevant data which are obtained on the course of development, and utilize the outcomes in project management.

3.1 Criteria of Data Collecting

For this paper, data were collected from 35 S/W developers of 5 organizations. Based on following 5 criteria, 27 metrics of the data were selected.

(1) Is data collecting possible?
(2) Is it contributing to improvement of Quality, Cost, Delivery, Productivity?
(3) Does it correspond to SPI (Software Process Improvement) strategy?
(4) Is the equality among developing organizations secured?
(5) Can reliability of the data be ensured?

Metrics as selected based on criteria above are shown in table 3-1(Metrics Table)

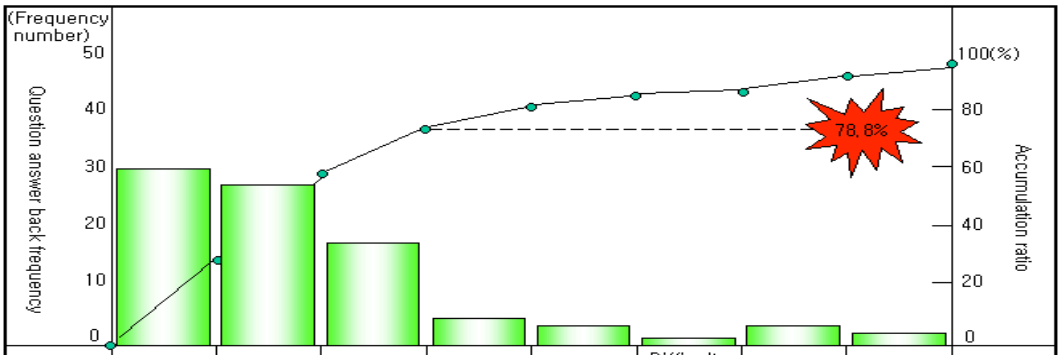
Table 3-1 Metrics Table of Measurement Data

Area	Factor	Metric	Calculation method
QUALITY	Defect	Defect density from design phase	Defect number of items of design / Number of page of the whole design document
		Defect density from module	Defect number of items of module / Total SLOC
		Defect management ratio	The management number of items which is completed / Total occurrence number of items
		Defect take out ratio	(Before release) The defect number of items which is discovered / Target discovery number of items
		Defect discovery ratio of user	(Release after 6 month ofr collections) Defect number of items / Total SLOC
	Requirement management	Requirement change ratio	Change, Add, Delete number of items / The requirement number of items which is decided
		Requirement implementation ratio	The requirement number of items where the implement is completed / The requirement number of items which is decided
		(customer)Unsatisfactory requirement number of items	The requirement number of items which is numsatisfactory / The requirement number of items which is decided(%)
		Changed number of line	Change, Add, Delete SLOC / Total SLOC(%)
	Process improvement	Design standard observance ratio	Design standard observancd number of items / Total design number of items
		Coding standard observance ratio	Sandard observance module(line) number of items / Total module(line) number of items
		Process observance ratio	Audit(checklist calculate) point
		Process improvement number of items	The improvement number of items which is reflected to a process standard
COST	Manpower cost	Manpower plan good hit ratio	The man month(MM) which becomes practice assign / The total man month(MM) which is planned
		Manpower plan good hit ratio each of phase	Each phase practice the total of the man month(MM) which is assigned / The total of the mon month(MM) which each phase is planned
	Failure cost	Defect correction time	Summation of defect correction hour(MH)
		Extra work time	Summation of over time(MH)
	Prevention cost	Teaching time	Summation of teaching time(MH)
		Review / Audit time	Until ready time summation of the hour when review/audit which is included(MH)
Delivery	Delivery observance	Delivery observance ratio	Actual accomplishment duration(month) / The accomplishment duration which is planned(month)
		Delivery observance ratio from each phase	Each phase actual accomplishment duration(month) / The accomplishment duration which is planned(month)
Productivity	Productivity	Each person productivity	KLOC / Man Month
		Each code conversion with document	Total document number of page / Total KLOC
		Each person document work	Total document number of page / Total assigned man month
	Re-use	Module re-use ratio	Re-used SLOC / Total SLOC
		Design document re-use ratio	Re-use design document number of page / Total design document number of page
	Size estimation	Size	Development the SLOC which is completed / Estimation SLOC

3.2 Obstacles to Secure Reliability of Input Data

For software developers who always feel stress of developing software, collecting metrics data is not easy. Therefore, if they are forced to collect data, data can be distorted when they

input the data. Therefore, it is important to know exactly which factors are burdensome in inputting data and remove them in order to get accurate input data from developers.
Following Figure 3-1 is burdensome factors to developers obtained from a survey.



Item	Additional effort	Unsasiness	Uncertainty	Failure to meet deadline	Sense of rejection	Difficulty in understanding	Vagueness	Complicatedness	Total
Frequency occurrence number	35	32	19	6	5	3	5	4	109 (Question answer back)
Prossession ratio	32.1	29.3	17.4	5.5	4.6	2.8	4.6	3.7	100%

Figure 3-1 Pareto Chart for analysis of burdensome factors to developers

Figure 3-1 shows the results obtained from a survey of 35 software developers. The surveyed items were developed through interview with developers. Description of each item is as follows.

Additional effort: additional effort required to input measurement data, besides software development work.

Uneasiness: sense of burden that the input data may be used as a data for evaluating an individual.

Uncertainty: uncertainty about whether the quantitative management will bring actual benefits to developers.

Failure to meet deadline: Uneasiness that additional work of data input can cause failure to meet deadline.

Sense of rejection: sense of rejection toward learning about new input system.

Difficulty in understanding: lack of understanding about input items

Vagueness: Vagueness of input range

Complicatedness: complicatedness caused by data input is conducted from too many sources.

The survey shows, as shown in Figure 3-1, that three major burdensome factors when developers input data are additional effort required to input measurement data besides software development work, sense of burden that the input data may be used as a data for evaluating an individual, uncertainty about the benefits quantitative management will bring. The three factors combined account for 78.8%.

The three factors can be reduced by introducing new tools, but unfortunately, cannot be completely removed, for collecting measurement data. Therefore, this paper suggests a way

to apply the second level meta game out of non-cooperative bimatrix games♣, in order to secure reliability of input data.

3.3 Analysis of Weight of Obstacles

The biggest obstacles to collecting accurate data are, as shown in Figure 3-1, boil down to three factors (additional effort, uneasiness, uncertainty). The third factor, uncertainty, is represented, here, as "profit of management" from qualitative management. After educating software developers enough about effects of quantitative management, another survey of the developers on the three factors was conducted again. As a result, weight values as shown in Table 3-2 were obtained.

Table 3-2 Weight Values for Major factors

	Additional effort	Unsasiness	profit of management
Weight Value of the data input which is reliability	-5	-3	8
Weight Value of the data input which is distorted	-1	0	-2

Table 3-3 is a matrix table of payoffs for developer A according to each strategy, based on these weight values.

Table 3-3 Distribution of Payoff according to strategy

		Strategy of developer B		
		α : Weight Value of the data input which is reliability	β : Weight Value of the data input which is distorted	δ : Data it does not input
Strategy of developer A	α : Weight Value of the data input which is reliability	0, 0	-10, 7	-5, 8
	β : Weight Value of the data input which is distorted	7, -10	-3, -3	-1, -2
	δ : Data it does not input	8, -5	-2, -1	0, 0

In the case that developer A selects α strategy, if developer B chooses α strategy too, ultimate payoff of A is 0. If B adopts β strategy, payoff for A is –10 due to false management, instead of plus gains from correct management. Finally, if B chooses δ strategy, data collecting is considered to fail because the size of the entire statistical population is too small. In this case, gains by A from quantitative management is 0, and there is no worry about being evaluated. Therefore, -5, value of additional effort for inputting data is the ultimate payoff of A. Table 3-3 shows payoffs calculated in that method.

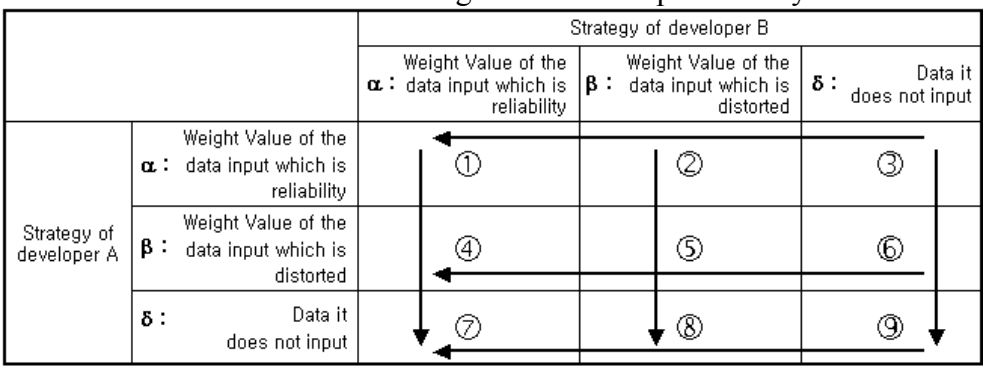
4. Application of Game Theory

4.1 Dominant Strategy and Pure Strategy Equilibrium Point

Software developers have three strategies available as shown in Table 3-3. For developer A's side, δ strategy out of the three strategies generates the largest payoff. In this case, δ

strategy of A dominates over others. Table 4-1 shows the movement diagram of developer A's payoffs in Table 3-3, based on dominant strategy.

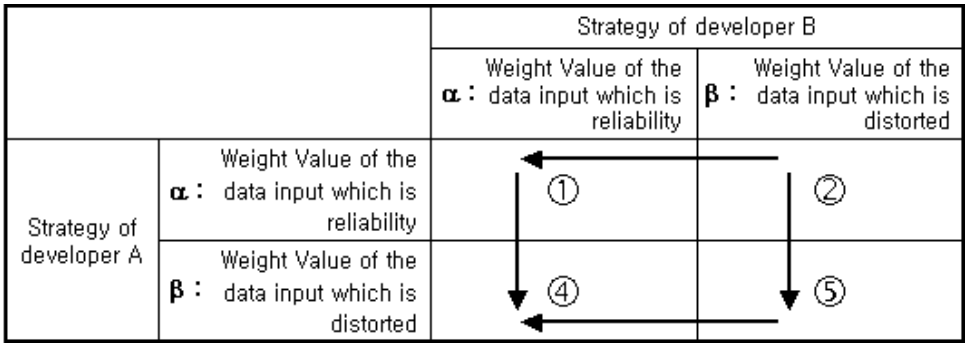
Table 4-1 Movement Diagram of Developer A's Payoffs



Analysis of the movement diagram in Table 4-1 shows us that arrows converge on number □, which means □ is equilibrium point of pure strategy. But for developer B's side, the equilibrium point is □. In other words, if all developers are allowed to head for equilibrium points of pure strategy with which they can maximize their own payoff, all developers will adopt δ strategy, and thus, number □ will become Nash equilibrium[10]. That is why collecting measurement data itself are not easy. If there is no input data, quantitative management is impossible. As a solution to this, this paper adopts a somewhat forceful way for measurement data entry.

The way involves inducing developers to input measurement data by imposing penalty on those who fail to input data. The movement diagram of this case corresponds to movement diagram of payoff when the third strategy, δ, of developer is excluded from Table 4-1. The movement diagram is shown in Table 4-2.

Table 4-2 Movement Diagram of Payoff when Strategy δ is Excluded



In Table 4-2, equilibrium point of pure strategy for developer A's payoff is number □. Like case in Table 4-1, all of developers can maximize their own payoff when they select strategy β. Therefore, if all of them pursue for strategy β, number of cases □ will become Nash equilibrium point and the ultimate payoff of developers will be □. But, number □ is non-Pareto optimal outcome as □ generates more payoff than □. In other words, in Table 4-2, non-Pareto optimal outcome □ is selected instead of Pareto optimal outcome □. This result shows that individual rationality based on principle of domination and group rationality based on Pareto principle do not correspond

4.2 Prisoner's Dilemma

Like the case in Table 4-2, non-zero sum game which has an equilibrium point, but non-Pareto optimal is called Prisoner's Dilemma[9]. It is referred as to the case in which players come to select non-Pareto optimal strategy if all players pursue their own maximum payoff, even though there is a Pareto optimal strategy available which provides maximum payoff to every one [5].

4.2.1 Exclusion of Strategy β

As a solution to this dilemma, strategy β can be excluded from Table 4-2, like from Table 4-1 in order to obtain Pareto optimal strategy. But exclusion of strategy β is impossible. Exclusion means here preventing developers from entering false data. This can be accomplished by examining reliability of all the input data and imposing penalty on developers who input false data. But it is difficult for developers to thrust away the temptation of strategy β because they know the fact that examining every input data is, in effect, impossible. Therefore, excluding strategy β is not a proper way.

4.2.2 Meta Game

In the case of the first level meta game where developer B decides sub-strategy according to strategy choice of developer A, A has 2 strategies(α, β) and B has following 4 strategies.

$\alpha\alpha$: To input reliable data regardless of developer A's strategy.
$\alpha\beta$: To select the same strategy as expected A's strategy.
$\beta\alpha$: To select the opposite strategy to expected A's strategy.
$\beta\beta$: To select false data regardless of A's strategy.

Table 4-3 is the matrix representing this situation.

Table 4-3 First level Meta Game

		Strategy of developer B			
		$\alpha\alpha$	$\alpha\beta$	$\beta\alpha$	$\beta\beta$
Strategy of developer A	α	0, 0	0, 0	-10, 7	-10, 7
	β	7, -10	-3 -3	7, -10	-3 -3

In the matrix above, strategy β of developer A does not dominate over strategy α anymore and strategy $\beta\beta$ of developer B dominates over the other three ones. Thus, strategy β of A, strategy $\beta\beta$ of B are the only equilibrium point. But this does not necessarily mean cooperation (input of reliable data) between the two developers. In other words, solution for the first level meta game is not cooperative strategy.

As Pareto optimal outcome is not obtained, developer A and B conduct the second level meta game in which developer B takes a sub-strategy and developer A selects a sub-strategy according to B's sub-strategy. In this case, developer A has 16 strategies available as shown in Table 4-4.

Table 4-4 Second level Meta Game

		DEVELOPER B			
		αα	αβ	βα	ββ
DEVELOPER A	αααα	0, 0	0, 0	−10, 7	−10, 7
	ααaβ	0, 0	0, 0	−10, 7	−3 −3
	ααβα	0, 0	0, 0	7, −10	−10, 7
	ααββ	0, 0	0, 0	7, −10	−3 −3
	αβαα	0, 0	−3 −3	−10, 7	−10, 7
	αβαβ	0, 0	−3 −3	−10, 7	−3 −3
	αββα	0, 0	−3 −3	7, −10	−10, 7
	αβββ	0, 0	−3 −3	7, −10	−3 −3
	βααα	7, −10	0, 0	−10, 7	−10, 7
	βααβ	7, −10	0, 0	−10, 7	−3 −3
	βαβα	7, −10	0, 0	7, −10	−10, 7
	βαββ	7, −10	0, 0	7, −10	−3 −3
	ββαα	7, −10	−3 −3	−10, 7	−10, 7
	ββαβ	7, −10	−3 −3	−10, 7	−3 −3
	βββα	7, −10	−3 −3	7, −10	−10, 7
	ββββ	7, −10	−3 −3	7, −10	−3 −3

In Table 4-4, there is no dominant strategy for developer B while strategy βαββ of developer A dominates over others. Therefore, developer B realizes that A will select βαββ, and thus selects strategy αβ favorable to B when A employs βαββ. Accordingly, βαββ of A, αβ of B becomes equilibrium point of the second level meta game, as shown in Table 4-4. So to speak, the best way for A is to believe that developer B will select the same strategy as him or her and take a cooperative strategy, and developer B knows the fact and selects the same strategy as A. Consequently, the players of the game have no choice but choosing a cooperative strategy in order to get maximum payoff. In this way, Pareto optimal solution for the second level meta game is obtained.

5. Simulation Study

Robert Axelrod maintained a natural evolution from competition to cooperation in his book 『The Evolution of Cooperation』[6]. The name of the game here is to demonstrate the question, "how can cooperation be developed in the world of fundamentally selfish agents?" This question involves following three questions: How can cooperation start? ; How can cooperative strategies survive better than non-cooperative ones? ; Which cooperative strategy can produce the best payoff and how can this dominate over the others [6]. He demonstrated these questions by applying prisoner's dilemma game.

This paper also applies the theory of Robert Axelrod to software development in order to change the developers behavior from competition to cooperation, and simulates this, using 『winpri』, a software developed by Philippe Mathieu and Fredderic Grignion.

Following figures are the outcomes obtained by simulating strategies selected by software developers through 『winpri』 after explaining 15 strategies of Robert Axelrod to them.

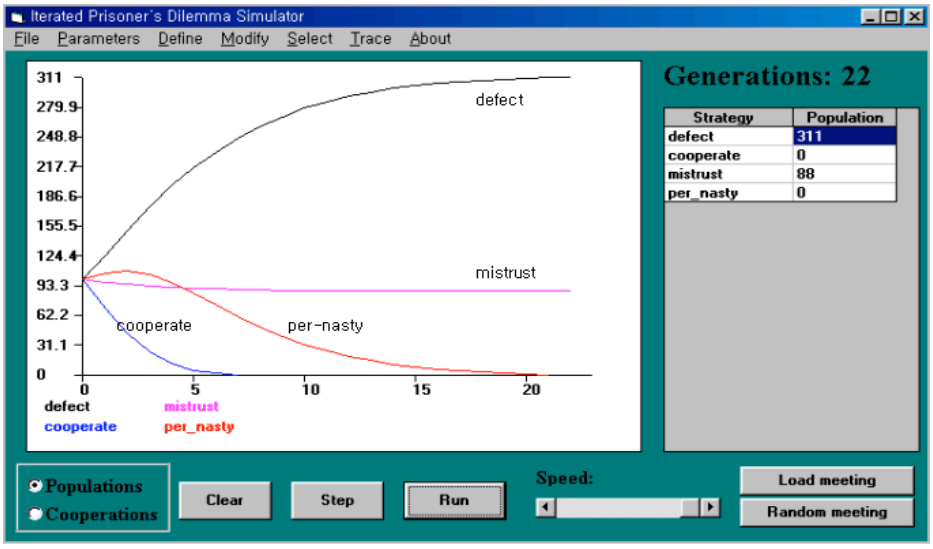


Figure 5-1 Cooperative Strategy among Defection Strategies

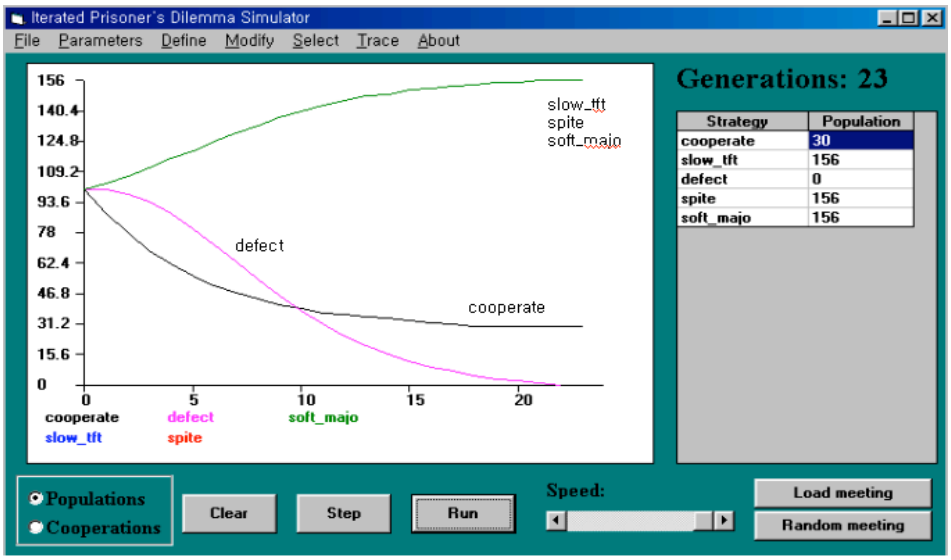


Figure 5-2 Defection Strategy among Cooperative Strategies

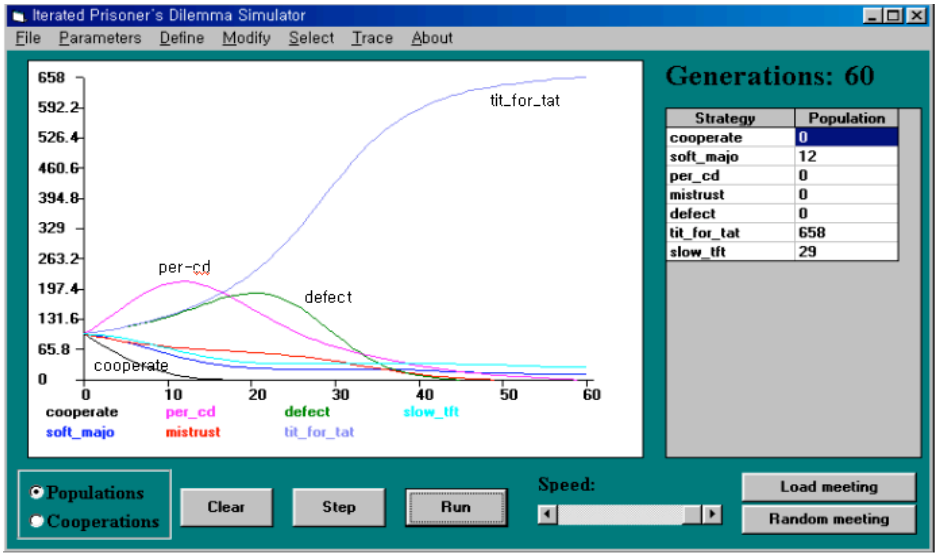


Figure 5-3 Dominant Strategy among Strategies

Figure 5-1 shows a cooperative strategy among defect strategies, and Figure 5-2 shows the payoff of a defection strategy among cooperative strategies. Finally, Figure 5-3 shows situations when cooperative strategies are mixed with defection strategies. In the early part of the game, defection strategies, “per-cd” and “defect” are seemed to dominate. But ultimately, "tit_for_tat" strategy (cooperating but retaliating partner’s defection with defection) produces the biggest payoff.

The simulation shows that cooperative strategy is dominant over selfish (defection) strategy. It also shows that reciprocal cooperation to respond the other’s defection with defection is more efficient than unconditional cooperation. As a result of educating these results to developers enough, and then collecting data again, it was found that the reliability of data is improved by 9.8%. This paper was analyzed through point estimation and interval estimation of statistical inference, for verification of its reliability.

6. Conclusion & Feature work

In software development, quantitative process management is one of the most common techniques to improve visibility (transparency). CMM and SPICE also recommend that quantitative process management be adopted when the maturity of the software process reaches a certain level. But if reliability of data is damaged, quantitative process management becomes pointless because every activity for quantitative process management is based wholly on metrics data. Therefore, it is necessary to verify reliability of data. Many ways to analyze the reliability have been studied. Most of them, however, are just analysis methods for already collected data.

Therefore, if the analysis finds that the existing data are not reliable, the data should be abandoned and new data be collected again, causing need to consume much time and effort. Therefore, as a way to improve reliability of measurement data to be collected, this paper deals with a technique to induce developers to input accurate data in the first place, instead of analyzing reliability of already collected data.

This paper analyzes psychological factors affecting data input (e.g. psychology that somebody else will take responsibility), additional efforts needed to input data, besides software development, and the concern that the data may be used to evaluate individuals, and studies on the solutions.

This paper finds the solution in the second level meta game of cooperative bimatrix games, applies the cooperation evolution theory of Robert Axelrod to the game, and demonstrates the solution to software developers by simulating the results of the application through 『winpri』 developed by Philippe Mathieu and Fredderic Grignion. As a result, reliability of measurement data is improved by 9.8%, compared to data collected with traditional methods.

The next study may well deal with organizational behavior theory defining the relation of project manager, software developer, and SPI for successful project management through application of game theory.

References

[1] Sang-Pok Ko, "A Study on the Measurement for Embedded Software", SERP. Vol. 2, pp. 635-638 June 2003.

[2] Paulk. M. C. et al. *The Capability Maturity Model : Guidelines for Improving the Software Process*. Addison-Wesley Pub Co., 1995.

[3] ISO/IEC 15504 TR2. *Software Process Assessment and Capability determination*. ISO/IEC 1998.

[4] Roser S. Pressman. *Software Engineering A Practitioner's Approach*. McGraw-Hill Pub Co., Fourth Edition 1998

[5] R.Axelrod. *The Evolution of Strategies in the Iterated Prisoner's Dilemma*. Pitman. London. 1987

[6] R.Axelrod. *The Evolution of Cooperation*. Basic Books. New York. 1984

[7] Ji-Hyun Ha(A psychiatrist). *The business psychology which is piquant.* Cheang-Lim Pub Co., Seoul. 2004

[8] Raghavan, T., "Zero-sum two-person games", *Handbook of Game Theory*, Volume 2, pp. 736-759, 1994.

[9] Rapoport, A., and A. Chammah, Prisoner's Dilemma, University of Michigan

[10] Nash, J., "Equilibrium points in n-person games", *Proceedings of the National Academy of Sciences USA*, 36, pp. 48-49, 1950

[11] Lucas, W., Game theory and Its Applications, American Mathematical Society, 1981.

[12] Milnor, J., "Games against nature", in Game Theory and Related Approaches to Social Behavior, Wiley, 1964.

[13] Rapoport, A., Mathematical Models in the Socialand Behavioral Sciences, Wiley-Interscience, 1983.