Journal of the Nigerian Association of Mathematical Physics
Volume **18 (May, 2011), pp 389 – 394**

Software Process Improvement Using Force Field Analysis

Francisca Egbokhare and Francis Imouokhome

Department of Computer Science
Faculty of Physical Sciences
University of Benin P.M.B 1154,
Benin City, Nigeria.

Abstract

Software process improvement is a necessity especially since the dynamic nature of today's hardware demands reciprocal improvements in the underlying software systems. Several process improvement models exist where organizations perform an introspective study of the current software development process and identify factors that require improvement. An improvement plan is then drawn and implemented. This paper studied the state of Nigerian software development organizations based on selected attributes. Force field analysis is used to partition the factors obtained into driving and restraining forces. An attempt was made to improve the software development process by transforming some of the restraining forces to driving forces.

Keywords: Software, Process improvement, driving forces, restraining forces, force field analysis.

1. Introduction:

The process of software development is dynamic and to achieve the quality objectives software development organizations, several software development and process improvement models exist [1,2,3,4,5]. A common feature of most software process models is that they encompass the total set of engineering activities needed to transform users' requirements into complete software system. [4] observed that there is a direct relationship between the quality of a software system and the quality of the processes used to produce and maintain it. To stay relevant and gain competitive advantage, most software development organizations now incorporate process improvement plans as part of their organizations' frameworks. Software process improvement involves an honest introspection and careful analysis of an organization's software development process to identify reasons for previous projects' successes and shortcomings. Any factors that give good results are consistently applied while process improvement frameworks are used to improve any factors that cause problems. [6] studied the state of the software development process in Nigerian software development organizations and discovered amongst other factors that lack of developers' motivation and the absence of improvement frameworks were some of the major causes of project failures. This paper proposed the use of force field analysis as a tool for software process improvement. Factors that influence software projects were partitioned into driving and restraining forces. Force Field diagrams were then used to model these factors and finally, we carried out some transformations (force field analysis) for software process improvement.

2. Methodology

A set of structured questionnaires with 8 (eight) software process attributes was the main tool used for data collection. The purpose of the questionnaire was to subjectively determine the presence/absence of certain qualitative attributes of a software process that could lead to successful software development. A homogeneous sample consisting of 20 randomly selected software development organizations from 4 (four) states in Nigeria, where there is high software development activity was used. This was to enable us to determine the state of the software development process in these organizations. A binary rating scale was used to code the data collected. For each attribute (A), a "Yes" response was assigned the value 1 (which indicated the presence of an attribute) and "No" was assigned the value 0 (which indicated the absence of an attribute). To determine the state of the software development process in an organization, we used the following formula for the state of an organization's software Process (O_s):

$$O_s = \sum_{i=1}^{N} A_i \{ \forall A_i = 'Yes' \} \quad (1)$$

Where:

N = the total number of attributes
A_i = attributes present

Corresponding authors: ***Francisca Egbokhare: E-mail:*** fegbokhare@yahoo.com, Tel: +2348037180057

A pass threshold is obtained when at least 80% of the attributes are present in an organization. A similar pass threshold was used to determine process maturity level [6].

3. Current State of Nigerian Software Development Organizations

An ideal software development process (Figure 1) involves the interaction of various components to produce the final system based on user requirements. The phases in software process models define the set of activities that are collectively performed by the interaction of these attributes.

Table 1 presents the findings from the 20 software development organizations studied. 3 (15%) of the organizations possessed at least 80% of the attributes for software project success, while a higher percentage (85%) could not obtain the pass threshold.

Percentage attributes Present	**Number of Organizations**
80 and above	03
70 – 79	05
61 – 69	02
50 – 59	00
40 – 49	07
30 – 39	01
Below 30	02
Total	20

When the base practices necessary for development of quality software systems are absent, chaos may arise in such organizations if no action is taken. Possible outcome (if no process improvement plans are implemented) is modelled in Figure 2:

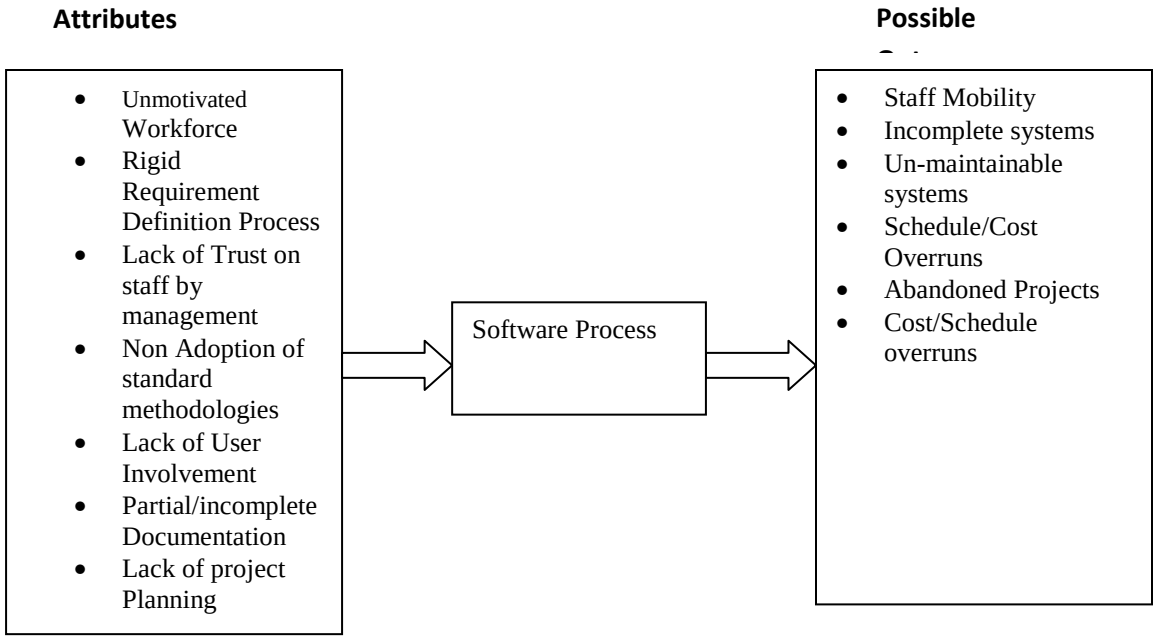

Figure 2: Possible Outcome when process improvement is not adopted

The factors modelled in Figures 1 and 2 can be viewed as two opposite forces at war that determine software project outcomes which are similar to the concepts described in Lewin's field theory.

3.2 Lewin's Field Theory (Force Field Analysis)

Force Field Analysis, a technique developed by Lewin [7], is useful for systematically reviewing the elements working for and against a proposed course of action. It assumes that in any situation there are both driving forces and restraining forces that influence implementation. In Lewin's field theory, organizations are viewed as systems in which the present situation is not static but a dynamic balance of forces working in opposite directions. In a force field analysis, an equilibrium is characterized

as a state of balance between driving forces (F_d) and restraining forces (F_r). Driving Forces are those forces that facilitate implementation while Restraining Forces impede the implementation process. To achieve change towards realization of a goal, it is essential to push on and overpower or immobilize the restraining forces or try to transform the restraining forces into driving forces [8]. A force field. diagram (Figure 3) is a picture that depicts the war between driving and restraining forces. The model is built on the idea that forces both drive and restrain change. In a force field diagram, driving forces are on the left column while restraining forces are on the right column. Arrows are drawn towards the middle with longer arrows indicating stronger forces.

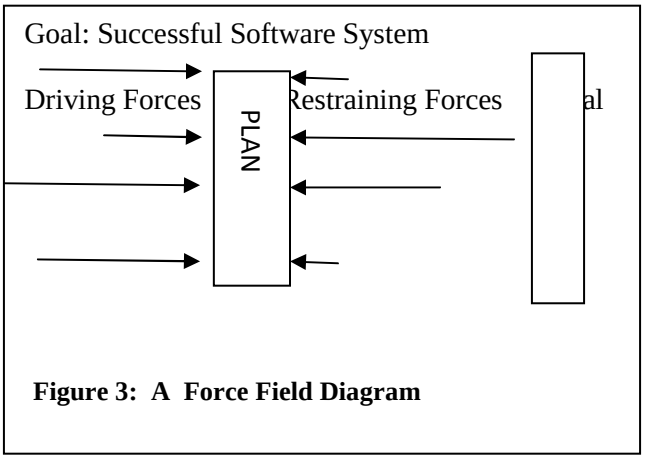

Figure 3: A Force Field Diagram

4. Application of Force Field Analysis to the Software Development Process.

Software process, (S_p) is viewed as a dynamic system, controlled by two opposing n forces, F_d and F_r represented as:

$$S_p = F_d + F_r \quad \text{.....} \quad \text{.....} \quad \text{.....} (2)$$

In an equilibrium state

$$F_d = F_r \quad \text{.....} \quad \text{.....} \quad \text{.....} \quad \text{...} (3)$$

Indicate that the organization have a defined software process but because the process is dynamic and mainly controlled by other external forces such as technology change, competition, change in user requirements, etc, a chaotic situation (S_c) may result in (4)

$$Fr > F_d \quad \text{.....} \quad \text{.....} \quad \text{.....} \quad \text{..} (4)$$

When (4) arises unless something is done very fast, the resultant effect can be catastrophic.

In this paper, we define a software process improvement plan as a strategy put in place by a software development organization to review its existing software process and to identify any factors that require improvement. During the review process, the factors obtained can be partitioned on a force field scale into two. Those that 'drive change' (which are retained) and those that 'resist change' which need to be improved by attempting to convert them into driving forces.

The findings from the organizations studied in this research are modelled as driving and restraining forces on a force field chart in Table 2.

Table 2: A force field chart showing driving and restraining forces

Driving Forces	**Restraining Forces**
Recruiting and Retaining Software developers	Lack of motivation of staff
Use of standard Software methodologies	Bureaucratic procedures
User Involvement throughout the Project	Unavailability of user representatives
Investment in current Technologies	Cost of staff training to use current technology
Flexibility of requirement definition process	Cost of changing requirements

Documentation	Staff mobility
Project Planning	Lack of Progress honesty
Trust	Cultural insensitivity to staff

To perform a force field analysis, weights are assigned to each factor using a numerical scale with high values indicating strong factors and vice versa. Using a 4-point scale, suppose we assigned values to each factor using the following weights:

i. Very Strong 4
ii. Strong 3
iii. Moderate 2
iv. Weak 1

This results in the following expressions for F_d and F_r .

$$F_d = \sum f_{d(i)}^{j} \ (i = 1,... \ N \ ; \ j = 1 \ ... \ 4).....(\quad 4a)$$

$$F_r = \sum f_{r(i)}^{j} \ (i = 1,... \ N \ ; \ j = 1 \ ... \ 4).....(\quad 4b)$$

Where f^{j} = assigned weights of f_d or f_r

The weights were assigned based on the ranking of the influence of each factor on overall project success/failure from literature [9],[10], [11],[12] as shown in Table 3.

Table 3: A force field chart with Weights assigned F_d and F_r

Driving Forces (F_d)	**Weights (F_d)**	**Restraining Forces (F_r)**	**Weight (F_r)**
Recruiting and Retaining Software developers	4	Lack of motivation of staff	4
Use of standard Software methodologies	4	Bureaucratic procedures	4
User Involvement throughout the Project	3	Unavailability of user representatives	4
Investment in current Technologies	2	Cost of staff training to use current technology	3
Flexibility of requirement definition process	3	Cost of changing requirements	3
Documentation	3	Staff mobility	4
Project Planning	2	Lack of Progress honesty	3
Trust	1	Cultural insensitivity to staff	1

The current ratio C_r between the driving and restraining factors from Figure 5 is:

$$C_r = F_d : F_r \quad \text{..... ....(5)}$$

This gives 22:26 which represents a chaotic situation. There is therefore need to improve the software process because the restraining forces are stronger (having weight = 26).

4.1 Performing Force Field Analysis

The next step is to analyse the restraining forces and attempt to transform some of them into driving forces. This varies between organizations and a defined plan for process improvement could become handy. Also, other external factors (non-process factors) can also play a major role in the transformation process. In this section, we attempted an analysis based on knowledge acquired from literature. It should be noted that in force field analysis, for every restraining force transformed one (1) additional point is added to the affected driving force.

1. Motivation is one of the strongest determinants of employee productivity[13]. This shows that increased motivation may have a strong influence on staff mobility. Therefore, when the two restraining forces — lack of motivation and staff mobility (resulting in reducing the strength of these forces by 2) — are eliminated from the restraining forces, these 2 points will be gained by the driving force , " recruiting and retaining staff" (ie. 4 (+2)).

2. [4] observed that when standard methodologies are not adopted in a software development organization, the only determinant of software project success is the heroic effort of some individuals (which cannot be guaranteed). When standard procedures are adopted, Bureaucratic procedures will be eliminated and this will add (+1) to the use of standard methodologies (4+(+1)).

3. When new technologies emerge, most organizations strive to invest in such technologies to remain relevant and gain competitive advantage. Staff Training may become necessary thus, the factor, "cost of training staff to use new technology" remains.

4. The requirements definition and analysis phase of software development is one of the most critical phases of software development. Users' needs change as they gain better understanding of the system in view. User involvement can be increased by investing on communication systems so that there can be constant interaction between developers and users. Rigidity may lead to un-usable or incomplete software systems. Hence, when flexibility is introduced, that factor will increase (+1). Resulting in (3 + (+1)) and unavailability of user representatives will be eliminated.
5. Project planning enables organizations to set milestones and define deliverables at each milestone. This enables the project to be focused and to strive towards the achievement of set milestones. This also guides the organizations to know the extent of work done and what is left. The clients can also receive concrete information on work progress. Project planning and cultural insensitivity are organizational issues and are based on set policies. Thus we did not attempt to improve them.

Table 4: A force field diagram for improved F_d and F_r

Driving Forces (F_d)	**Weights (F_d)**	**Restraining Forces (F_r)**	**Weight (F_r)**
Recruiting and Retaining Software developers	4 (+2)	Lack of motivation of staff (eliminated)	0
Use of standard Software methodologies	4 (+1)	Bureaucratic procedures (eliminated)	0
User Involvement throughout the Project	3 (+1)	Unavailability of user representatives (eliminated)	0
Investment in current Technologies	2	Cost of staff training to use new technology (improved)	3
Flexibility of requirement definition process	3	Cost of changing requirements	3
Documentation	3	Staff mobility (eliminated)	0
Project Planning	2	Lack of Progress honesty	3
Trust	1	Cultural insensitivity to staff	1

At the end of the analysis, the ratio of driving forces to restraining forces became 26:10 from Table 4 which implies a process improvement.

5.0 Conclusion

The software development process is dynamic and because changes in computer hardware in today's technology-driven society necessitate reciprocal changes in the supporting software, there is need for continuous software process improvement. This paper presents the software development process as a system with two opposing forces warring against each other. These forces were modelled using force field diagrams and force field analysis was used as a transformation tool to systematically improve the software development process. Continuous process improvement helps organizations to stay relevant and to gain competitive advantage. Force field analysis eases the task by providing the necessary tools to model and analyze the software process.

References

[1]. Sommerville, I.(2004). Software Engineering, 7th Edition. Addison–Wesley.

[2]. Peters J.F. & Petrycz W.(2000). Software Engineering: An Engineering Approach. John Wiley & Sons, 39 – 55.

[3]. Paulk, M., Weber C, Garcia S., Chrissis M. & Bush M.(1993a). Capability Maturity Model For Software, Version 1.1. SEI-93-TR-024. Software Engineering Institute, Pittburgh.

[4]. Humphrey, W.S.(1990). Managing The Software Process. Addison-Wesley. U.S.A.

[5]. Humphrey, W.S.(2000). The Team Software Process. CMM/SEI-2000-TR-023. Software Engineering Institute, Pittburgh.

[6]. Onibere, E.A. and Egbokhare, F.A.(2007) Software Development Process: A Case of Nigerian Software Development Organizations. The Information Technologist 4(1), 1-13

[7]. Yingxu and Graham (2000). Software Engineering Processes: Principles and Applications. CRC Press. 690pp.

[8]. Lewin K. (1951) 'Field Theory in Social Science', Harper and Row, New York.

[9]. Thomas J. (1985) 'Force Field Analysis: A New Way to Evaluate Your Strategy', Long Range Planning, Vol. 18, No. 6, pp. 54-59.

[10]. Lindner, J.R. (1998). Understanding Employee Motivation. Journal of Extension, 36(3), 7pp

[11]. Cockburn, A.(2003). Humans and Technology. Cockburn and Associates. Available online at: http//alistair.cockburn.us

[12]. Passova, S.(2005). The Need for Customer-Oriented Software Development. Sofea, Inc. e-Biz Development tools. Available online at: www.ebiz.net/topics/devtools/features/5577.html

[13]. Boehm, B.(1994). Software Engineering Economics. Englewood Cliffs, NJ.Prentice-Hall. 767pp