

Stochastically Simulating Low Morale - A Risk Factor Effect on Risk Management

¹P.K. Suri and ²Rachna Soni

¹Department of Computer Science and Applications, Kurukshetra University, Kurukshetra, Haryana, India

²Department of Computer Science and Applications, D.A.V. College for Girls Yamuna Nagar, Haryana, India.

Abstract: One of the proposed purposes for software process simulation is the management of software development risks, usually discussed within the category of project planning and management. However, modeling and simulation primarily for the purpose of software development risk management has not been explored and is quite limited. This study describes an approach to simulate Low Morale - a risk factor, to analyze its effect on certain software development risk management activities.

Key words: Software development risk, risk factors, software risk management, stochastic simulation

INTRODUCTION

One of the proposed purposes for software process simulation is the management of software development risks, usually discussed within the category of project planning/management^[1]. However, simulation primarily for the purpose of software development risk management has been quite limited. A notable exception is Madachy's model^[2], designed partially for the purpose of risk assessment. This study describes a different approach to simulation for managing software development risks.

Assessing uncertainty through risk factors: Uncertainty gives rise to risk, the potential of loss. The categories of (Gemmer, 1997, Table 1) are helpful for thinking about various uncertainties in managing software projects.

One means of managing the risks arising from uncertainty is to characterise risky scenarios and identify the risk factors in those scenarios. Each scenario can then be associated with a probability of occurrence, a potential cost and its value (utility) to the administrators. However, analysis of scenarios begins with identification and quantitative description of the factors composing scenarios. These risk factors can then be arranged in various scenarios and using a vehicle for propagating their uncertainties, can be related to system outcomes.

A simulation model supports risk management to the extent that it supports the generic process.

Software development risk factors (SDRFs): The software engineering literature on risk has focused on three topics:

- a. The appropriation of techniques from other disciplines

Table 1: Managerial categories for uncertainty

| Type of Uncertainty | Definition |
|---------------------|-----------------------------------------------------------------------------------|
| In time | Uncertainty about when certain events may occur or the ability to react to them |
| In control | Inadequate authority to make or influence decisions or inconsistency in processes |
| In information | Inadequate or inaccurate information on which decisions are based |

- b. The development of risk management approaches for software development
- c. The identification of software development risk factors and their relationship to project outcomes. Although, the second topic i.e. risk management, has received the most attention, considerable attention has been and is being given to understanding the sources of risk.

A variety of approaches have been used to investigate SDRFs. From them, a trend of collections of risk factors have emerged, including prioritized lists, taxonomies, questionnaires and matrices, for assessing software development risks. Some investigators have produced SDRFs lists numbering to the orders of 150 or more factors. Houston^[3] found twenty-nine of these factors were cited most often in studies intended to identify the most important SDRFs. The selected SDRFs for simulation in this study are shown in Table 2.

In context of Table 2, this can be stated that the effect of low morale on productivity can vary during a project and it can be continuously recalculated and the distributions for its variables are sampled continuously throughout each run.

Effect of low morale on Efficiency: Morale is modeled as a variable having an initial level set. This starting level, a value between 1 (lowest morale) and

Table 2: SDRF selected for simulation

| Risk factor | Potential effects modeled | Random variables |
|---------------------------|-----------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| *Lack of staff Commitment | Lower productivity, Increase in error generation, Attrition | *Multiplier to productivity for morale *Multiplier to error generation for morale |
| *Low Morale | Attrition, Morale change, Lower productivity due to loss of expertise | *Multiplier to attrition for morale Attrition, Replacement delay, Change to morale for attrition to make existing models dynamic |

11 (highest morale), typically reflects good morale, for example 7 (slightly “up”) or 8 (fairly satisfied team). The morale level can be affected by high schedule pressure and by attrition. It may, in turn, produce effects on productivity, error generation and attrition.

With regard to the effect of staff morale on productivity, the survey results showed that

- a. When low morale affected Efficiency, productivity decreased
- b. The probability of an effect increased from 0.46 to 0.95 as morale decreased from “slightly down “to “lowest: open rebellion”.

A typical software development project was considered for simulation experiment. In this case, morale is modeled.

INFORMAL DESCRIPTION OF THE ALGORITHM

An efficiency level of an employee in an organization may be defined as the measure of employee’s satisfaction. If the schedule pressure is more, the efficiency will naturally be low. No management would like to have a policy with poor efficiency level of an individual even if it is most economical operating policy. The efficiency here is expressed in percentage. An initial level of “Morale” is set. The “morale level” can be affected by “high schedule pressure” and by “attrition”. It may produce effect on efficiency and in turn, on productivity.

Terms used

| | |
|-------------------------------------|-----------|
| Morale level | ML |
| Cumulative morale level | CML |
| Incentive date due | DDUE |
| Incentive due | INSVD |
| Schedule pressure level | SHEDP |
| Cumulative Schedule pressure | CSHEDP |
| Shortage in morale level | SHORTAGE |
| Cumulative Shortage in Morale level | CSHORTAGE |

| | |
|-------------------------------|-------|
| Threshold Morale | THMO |
| Average morale level | AVML |
| Efficiency | EFFEN |
| Total weeks of simulation run | NRUNS |
| Moral boosting period | MBP |
| Moral level addition | MLA |

The program was written to simulate Fixed Threshold morale THMO & Morale level addition MLA combinations. The average morale level and the effective Efficiency level of a person was computed and printed.

The length of each simulation run was made NRUNS=500 weeks. We have assumed the initial morale level to be 11 units and no initial “incentive due” at the beginning of the simulation run.

ALGORITHM

STEP 1: Read input data. [Read the NRUNS i.e number of simulation runs, THMO (threshold morale, MLA(morale level addition))]

STEP 2: Perform Initialization.
[Set ML=11, RUN=1 CML=0, INSVD=0, DDUE =0
SHORTAGE =0
SHEDP=0 CSHEDP=0 CSHORTAGE=0]

STEP 3: ML –SHEDP
IF (ML<=0) ML=0
IF (DDUE=0) THEN { ML=ML+INSVD, INSVD=0}
CML=CML+ML,
IF (INSV=0){ IF (ML<=THMO)
{INSVD=MLA, DDUE=MBP} }

STEP 4: Generate Schedule Pressure level
[Evoke random number generator for generating random samples from empirical distribution. Use these samples to compute cumulative schedule pressure level and unfulfilled schedule pressure]

STEP 5: IF (SHEDO>ML)THEN
SHORTAGE=SHEDP-ML
ELSE SHEDP =0,
IF (DDUE >0) DDUE=DDUE-1

STEP 6: Compute Efficiency level
[The Efficiency level is [(CSHEDP – CSHORTAGE) *100] / CSHEDP

STEP 7: Compute average morale level

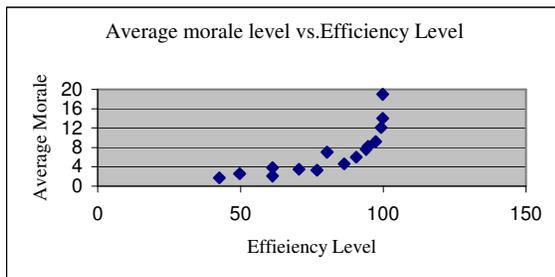
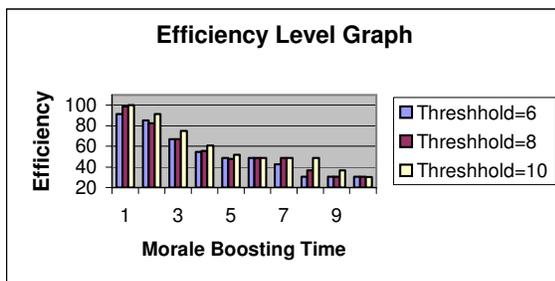
STEP 8: Update simulation run counter
[Increase run by 1 that keeps the record how many times a record has been executed]

STEP 9: If RUN < NRUN then go to step 3

STEP 10: Compute and print the Efficiency level of each run.

Table 3:

| Morale boosting period | Threshold moral=8 Avg moral level | Efficiency level (%) | Threshold moral=10 Avg moral level | Efficiency level (%) |
|------------------------|--------------------------------------|----------------------|---------------------------------------|----------------------|
| 1 | 11.70 | 98.97 | 13.58 | 99.96 |
| 2 | 8.23 | 82.22 | 9.48 | 91.42 |
| 3 | 6.66 | 66.80 | 7.48 | 74.84 |
| 4 | 5.54 | 55.67 | 6.07 | 60.77 |
| 5 | 4.77 | 47.86 | 5.16 | 51.88 |
| . | . | . | . | . |
| 9 | 2.90 | 30.49 | 2.9 | 30.39 |



The efficiency for different level of Morale threshold has been worked out and shown below.

From the above Table we can analyze that by reduction in morale boosting period the performance of the person improves, i.e. efficiency level but this is at increased average moral level.

Increasing moral-level -addition can increase the Efficiency level. But this also results in increased average moral level.

Again, a program was written to simulate all combinations of Threshold morale from 5 to 25 in steps of 2 and morale addition level (Incentive) from 5 to 15 in steps of 2 .For each of these different combinations ,the average morale level and Efficiency level computed and printed The length of each simulation run was 500 weeks.

From the Table 4, a plot of efficiency level vs average morale level was made. The policies lying on the curve are the best policies. Those lying above are inefficient and ineffective policies, because input in the

Table 4: Average morale level for given Efficiency level

| Efficiency Level | Average morale | Threshold morale | Morale addition |
|------------------|----------------|------------------|-----------------|
| 42.6 | 1.7 | 5 | 7 |
| 49.7 | 2.6 | 5 | 9 |
| 61.2 | 2.1 | 9 | 5 |
| 61.2 | 3.8 | 7 | 11 |
| 70.5 | 3.5 | 9 | 7 |
| 76.8 | 3.3 | 11 | 7 |
| 80.3 | 7.0 | 11 | 15 |
| 86.3 | 4.6 | 15 | 5 |
| 90.5 | 6.0 | 21 | 5 |
| 94.0 | 7.6 | 17 | 7 |
| 94.7 | 8.2 | 17 | 9 |
| 97.3 | 9.2 | 19 | 7 |
| 99.2 | 12.1 | 21 | 9 |
| 99.8 | 14.0 | 23 | 9 |
| 99.8 | 19.0 | 25 | 15 |

form of incentives do not result in improving efficiency level.

EFFICIENCY AND PRODUCTIVITY

Humphery^[4] has developed the personal software process to observe and improve the skills of an individual software engineer by reducing the errors affecting his productivity. It is commonly known that the improvement in efficiency level results in improvement in productivity. The industrial houses involved in software development business do observe their business interests in terms of “Productivity per employee”. Therefore, the productivity of an individual is an important measure which, in turn, is affected and observed through efficiency and morale level.

CONCLUSION

Software project managers can use such a model to study the potential impact of “low morale”- a risk factor, on project outcomes, then run simulation for risk mitigation plans, risk contingency plans and interventions, all as means of elucidating their experience and supporting project management decisions. Experimentation with risk- oriented simulators may provide insights into the relative influence of various risk factors and into best practices for project risk management.

REFERENCES

1. Kellner, M.I., J.R. Madachy and D.M. Raffo, 1999. Software process simulation modeling: Why? What? How? J. Systems and Software, 46: 91-105.
2. Madachy, R.J., 1994. A software project dynamics model for process cost, schedule and risk assessment. Ph.D. Thesis, University of Southern California.
3. Houston, D., 2000. Software project risk simulator. Ph.D. Thesis in progress. Jones, Capers (1994).
4. Humphrey, W., 1997. Introduction to the Personal Software Process. Addison-Wesley.