SOFTWARE PROCESS IMPROVEMENT PLANNING BY DESCRIPTIVE AHP

Satoru Takahashi and Kiyotoshi Komaya
Industrial Electronics and Systems Laboratory
Mitsubishi Electric Corporation
8-1-1 Tsukaguchi-honmachi, Amagasaki city, Hyogo 661-8661, JAPAN
takahasi@soc.sdl.melco.co.jp

Hiroyuki Tamura
Department of Systems and Human Science
Graduate School of Engineering Science, Osaka University
1-3 Machikaneyamacho, Toyonaka-city, Osaka, 560-8531, JAPAN
tamura@sys.es.osaka-u.ac.jp

Keywords: Software Process, Engineer Satisfaction, CMM, Descriptive AHP

Summary: *Software engineering organizations now tend to improve their processes according to such standards as CMM (Capability Maturity Model), a process evaluation model, in addition to improving their software products. This is because it is believed that mature engineering processes can develop high-quality products within the original schedule, and because customers want some objective measurements for their evaluations. This paper proposes a new model for improving software engineering processes, which can highly satisfy system engineers, by using a descriptive analytic hierarchy process (Descriptive AHP), a new AHP model for describing rank reversal phenomenon. Our model evaluates four engineering phases under six criteria defined at CMM Level 2, determining which phases should be improved first. As a result of having our system engineers use this method, we found that quality assurance activities in the manufacturing and examination phases need to be superior for improvement to other phases.*

1. Introduction

A lot of software engineering organizations now tend to evaluate and improve their own engineering processes, for the purpose of tracking projects objectively in detail and making decisions on how to proceed with projects efficiently (Humphrey, 1989). These movements are based on experiences which proves that organizations with better engineering processes can develop better products. ISO9000 and CMM (Paulk et al, 1993) are recognized as standard evaluation models for software engineering processes. Organizations which follow such models provide better environments for developing timely, cost-efficient and high-quality products. Customers are also able to measure the engineering ability objectively, rather than estimating it according to the brand name of the organizations as they have in the past. These process evaluations must be carried out by independent auditors in order to prevent arbitrary or biased interpretations. The auditors evaluate the process of the organizations and specify which phases should be improved.

We found, however, that the audit result did not always produce effective improvement of the engineering processes if the system engineers in the organizations were not satisfied with its improvement remedies. On the other hand, if the phases proposed to be improved correspond to the ones that the system engineers want to improve, the engineers' efforts can profoundly effect the process improvement (Yamamura, 1999). Engineering process evaluations which take engineer satisfaction into account have not been carried out in recent research, although their satisfaction is strongly related to activity results. On the contrary, engineer opinions have often been ignored by managers, who track the project from a different point of view, and this situation may well lead to insufficient results.

2. Software Process Improvement

CMM (Capability Maturity Model) (Paulk et al., 1993), which was proposed by Software Engineering Institute, Carnegie Melon University in the United States, is one of the standard evaluation models for software engineering processes. CMM divides the maturities of the engineering processes (how the process is defined/controlled/measured) into 5 levels (from Level 1 to 5). Project success in Level 1 organizations depends on the experiences and luck of the managers, and good products can be produced within the planned schedule by managers in higher level organizations. In each level, the conditions to be filled in order to advance the next level are defined as the KPA (Key Process Area) . Table 1 shows the KPA of CMM Level 2, which is the main target for lots of organizations to achieve at first.Goals and concrete activities are also defined there. The organizations can improve their maturity level by following these activities.

Table 1 KPA of CMM Level 2

Level	KPA	Abstract
2	Requirements Management	Establish common understanding with customers
	Software Project Planning	Make a reasonable project plan
	Software Project Tracking and Oversight	Give adequate visibility into actual progress
	Software Quality Assurance	Provide measurement to visualize the process
	Software Configuration Management	Maintain integrity of products in the life cycle
	Software Subcontract Management	Select qualified sub-contractors and manage them

In process improvement activities, the auditors and organizations evaluate how their current situation achieves the standard defined by the appropriate evaluation models, and propose activities based on the result, such as furthering engineer education or preparing effective tools. These improvement activities have to be targeted to one or two phases, since dramatic change in too many engineering phases can produce confusion for engineers, and activities need resources such as money, people and time.

Moreover, whether these activities succeed in improving largely depends on the system engineers' motivation, and large improvement can be expected when the proposed activities concentrate on the areas that are dissatisfied with. This means that conventional evaluation models, which evaluate engineering processes only by physical quantities, such as developing costs, or by absolute criteria, are not necessarily appropriate for choosing suitable phases to improve.
This paper proposes a new model which evaluates system engineer satisfaction with current engineering processes, in order to choose the best phase to improve the software engineering processes. Our new evaluation model focuses on the waterfall model engineering processes, which is a major process model in many software engineering organizations. The engineering activities in the waterfall model are carried out by the order of design, manufacturing, examination and maintenance, as shown in Table 2. Different engineering groups are often allocated to each phase, and only have responsibility for following their own procedures. Our model evaluates engineer satisfaction for each phase of the waterfall model processes by applying Descriptive AHP (Tamura et al., 1998), a new AHP model which improves the original analytic hierarchy process.

Table 2 Waterfall model Engineering Phases

Phase	Charge	Responsibilities
Design	Desing Dept.	Requirements acquisition and system design
Manufacturing	Manufacturing Dept.	Program design and coding
Examination	Quality Assurance Dept	System integration test
Maintenance	Maintenance Dept.	Installation and trouble-shooting

Prior to evaluating engineer satisfaction, we built a hierarchy for evaluating four engineering phases as alternatives under the KPA defined at CMM Level 2, as shown in Fig. 1.

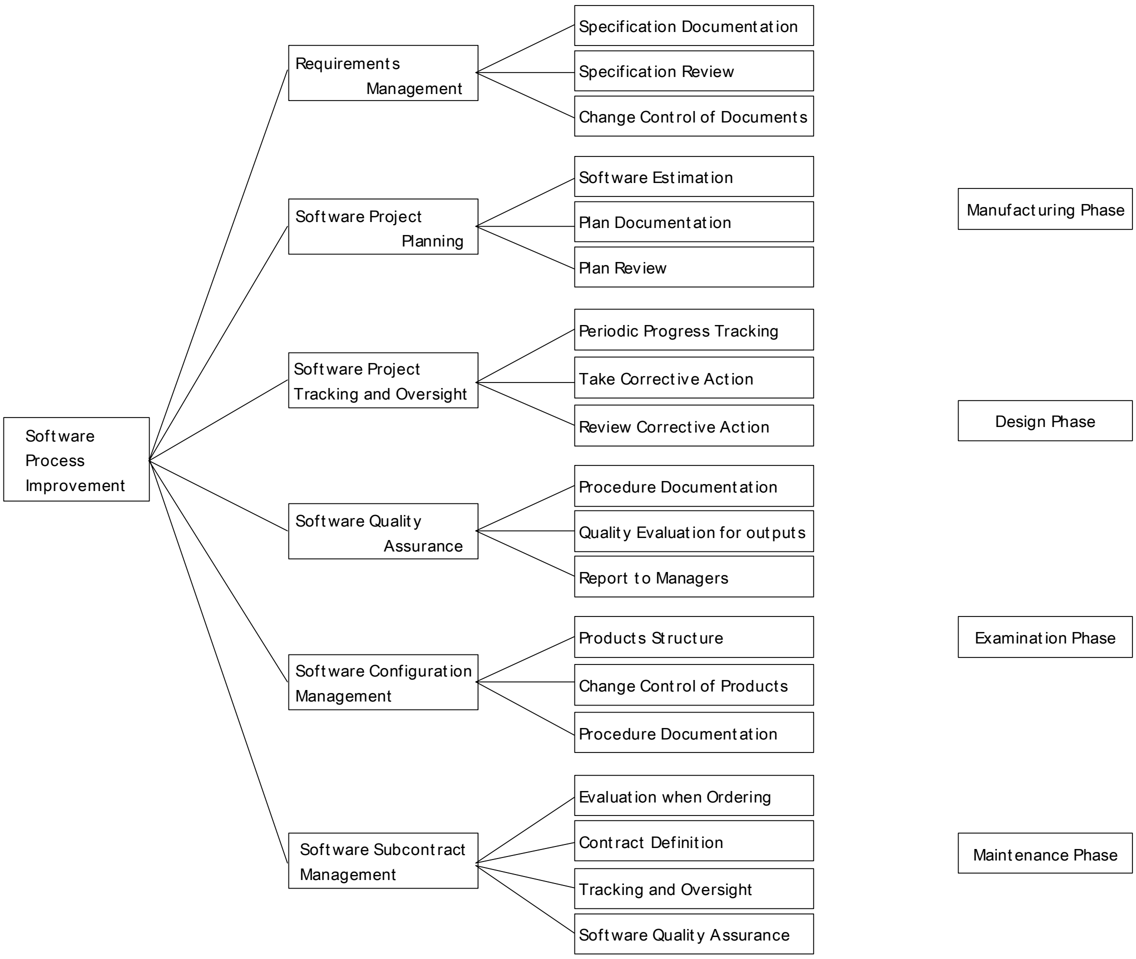

Fig. 1 Software Process Evaluation Hierarchy

3. Descriptive AHP

Descriptive AHP (Tamura et al., 1998) is a model to consistently describe rank reversal phenomenon, which is a major problem for the conventional AHP. Rank reversal is a phenomenon that occurs when the preference order of alternatives change by adding or deleting alternatives, even if the pairwise comparison results between the existing alternatives does not change.Previous research (Belton and Gear, 1983) has considered this phenomenon to be the contradiction of the conventional AHP method, and has made attempts to preserve the rank.

Descriptive AHP realizes that rank reversal can occur in the real world, and gives a meaningful interpretation to explain why this rank reversal occurs. This model requires an aspiration level to be set under each criterion. This aspiration level is a hypothetical alternative to at least satisfy evaluators. Evaluators add this aspiration level to a set of alternatives, and make a pairwise comparison matrix. The weights of alternatives are then obtained from the comparison matrix by the same method as the conventional AHP, and the weights of alternatives are normalized as that of aspiration level is 1.

In conventional AHP, rank reversal phenomenon occurs when the number of alternatives change, since this model normalizes weights of alternatives to sum up to 1, even if comparison results between alternatives do not change. In descriptive AHP, however, the weights of alternatives do not change without aspiration level changes, and rank reversal occurs only when aspiration levels are modified by the

change of situations around evaluators.
The merit of using Descriptive AHP is not only that it can describe a wider range of situations such as rank reversal phenomenon, but also that it is easier for evaluators to compare pairs of abstract alternatives. For example, it is very difficult to compare the design phase and the manufacturing phase under the criterion ``requirements management'', that is, to compare how much information is in the system design specification documents and software design specification documents, since they cannot be converted into numerical values. However, it is easier to imagine an aspiration level according to the Descriptive AHP procedure, since evaluators often require some goals to be satisfied when starting activities, and the aspiration level imagined will correspond to the lowest goal. Therefore, comparison between the real alternatives and the aspiration level is substantially the same as asking how one is satisfied with the real alternatives.

4. Software Process Improvement Model with Descriptive AHP

The software process evaluation hierarchy shown in Fig. 1 has four alternative phases under six main-criteria and 22 sub-criteria. The main-criteria are the KPA defined at CMM Level 2, and the sub-criteria are the concrete descriptions of the goals and activities in each KPA.

Our model requires evaluators to answer the importance ratio for improvement among any pair of criteria by selecting one choice out of Table 3. Then, the weights of alternatives can be calculated by applying the eigenvector method or geometric means method to the comparison matrix and normalizing them to sum up to 1.

Table 3 Choices for Pairwise Comparison Among Criteria

Choices	Importance ratio allocation
Much more important	7
More important	5
A little more important	3
Same	1
A little less important	1/3
Less important	1/5
Much less important	1/7

In the next stage, our model asks evaluators to compare pairs of alternatives for their importance under each sub-criterion. However, it is too difficult to consistently compare all pairs among a set of alternatives, including an aspiration level, since it requires responding to 220 questions.

In this paper, we ask evaluators only to compare each alternative with the aspiration level, instead of comparing all pairs. Our model is able to set the weight of each alternative by this simpler method because weights of alternatives can be determined only by setting an aspiration level. It only requires 84 questions for all.

Moreover, the questionnaire becomes easier to answer in our model, since comparing an alternative with the aspiration level is no more than asking how one is satisfied with the alternative. For example, all we have to do under the sub-criterion ``Specification documentation'' is ask how each engineer is satisfied with the contents of the specification documents after asking them imagine the aspiration level. Engineers select one of choices defined in Table 4 for measuring satisfaction with the alternative phase.

We also prepared a list which shows the criteria in detail, as shown in Table 5, in order to make it easier to imagine aspiration levels. The model calculates the weights of alternatives from the comparison results by normalizing the weight of the aspiration level to be 1 when the engineers finish the comparison.

Table 4 Choices for Pairwise Comparison Among Alternatives

Choices	Importance ratio allocation
Much dissatisfied	7
Dissatisfied	5
A little dissatisfied	3
Satisfied	1
Considerably satisfied	1/3
Much satisfied	1/5

Table 5 Standard for the Aspiration Level

Sub-criteria	Aspiration Levels to be imagined
Specification Documentation	have requirements for spec., deadline, environments etc.
Specification Review	review all specification documents
Documents Change Control	revise documents whenever specification changes
Software Estimation	estimate software scale, man-hour, development costs and so on
Plan Documentation	record development plan such as schedules, standards, etc.
Plan Review	review development plan
Periodic Progress Tracking	track project progress and problems periodically
Take Corrective Action	re-make a new plan and add resources for project delay
Review Corrective Action	review if a new plan is appropriate
SQA Procedures Documentation	confirm if SQA has appropriate procedures and members
Quality Evaluation for Outputs	confirm if enough review and examination are carried out
Report to Managers	report to managers about the project quality
Products Structure	write the whole structure of software and documents
Change Control of Products	control outputs versions and record change history
Configuration Procedures Doc.	record the procedures and standards for outputs configuration
Subcontract Evaluation	evaluate sub-contractor before ordering
Subcontract Definition	clarify the inputs and outputs of the subcontract
Subcontract Progress Tracking	confirm project progress of sub-contractors periodically
Subcontract Quality Assurance	examine the outputs of sub-contractors when accepting them

5. Application

We made a questionnaire of 84 questions following our model, and asked system engineers from among our real project members to answer. The results for two engineers are shown in Table 6 and Table 7. The first line of the tables shows the name of the criteria and its weights. The other lines show the weights of alternatives under each criteria, with the total weights on the right. The higher the value is, the more engineers feel dissatisfied with the phase and want improvement. Whether the total value of each alternative phase exceeds 1 or not can be an index of engineer satisfaction, because if engineering phases are considered to be at the aspiration level for all criteria, the total weight of the phases would be 1.

Table 6 Result for 1st evaluator

	Require. Manage. 0.227	PJ Plan 0.171	PJ Track 0.083	Quality Assur. 0.388	Config. Manag. 0.048	Subcon. Manag. 0.083	Total Weight 1.000
Design	1.312	1.423	4.376	2.249	1.000	2.400	2.024
Manufacture	3.000	2.278	1.566	4.896	4.376	2.259	3.498
Examination	2.364	3.796	5.220	4.651	2.688	3.077	3.808
Maintenance	1.624	2.694	1.266	2.249	2.688	0.722	1.996

Table 7 Result for 2nd evaluator

	Require. Manage. 0.109	PJ Plan 0.206	PJ Track 0.041	Quality Assur. 0.389	Config. Manag. 0.084	Subcon. Manag. 0.172	Total Weight 1.000
Design	1.000	1.572	0.333	1.217	1.516	0.333	1.105
Manufacture	2.717	1.572	0.333	1.773	1.516	3.284	2.016
Examination	4.147	0.715	0.333	2.603	0.794	2.047	2.044
Maintenance	1.859	2.717	0.600	1.587	1.004	0.809	1.628

These results show that, out of four phases, engineers in the organization are dissatisfied with the manufacturing and examination phases, and quality assurance is considered to have to be improved first among six main criteria. The engineers were satisfied with these results, and therefore agreed to develop new tools to measure and improve software qualities in the manufacturing and examination phases. We also found that some alternative phases were valued to be lower than 1, that is, that engineers were very much satisfied with the current situation of these phases. We can easily infer that changes in these phases might not be beneficial.

6. Conclusion

In this paper, we proposed a model to quantitatively evaluate engineer satisfaction for software engineering processes with Descriptive AHP. We also applied the model to our real engineering projects and found that quality assurance activities in software manufacturing and examination phases need to be improved first.

It is often said that software process improvement activities should be executed in a top-down style, since it is sometimes difficult to change the engineers' conservative positions without pressure from project managers. However, managers sometimes propose remedies that do not satisfy engineers, since managers view engineering processes from a different perspective. When both managers and engineers use our model, they will notice how much different sense they have even for the same engineering processes, and which will assist them in achieving the best solutions.

Group evaluation models will be desired for a large project, though our current model is targeted to single engineer. We are now considering how to integrate the opinions of several members, in order to put forth appropriate solutions for reaching mutual agreement.

When we apply our model to more projects, there may be cases where some members can not compare all pairs of alternatives or criteria, for instance, when a member who is engaged in a specific engineering phase can not answer questions about other phases. Calculating weights from an incomplete pairwise comparison matrix must also be considered in our future work.

Reference

Humphrey, W.S. (1989) *Managing the Software Process*, Addison-Wesley

Paulk, M.C., et al. (1993) *Key Practices of the Capability Maturity Model*, Ver. 1.1, SEI-93-TR025

Yamamura, G. (1999) “Process Improvement Satisfies Employees”, *IEEE Software*, 16- 5, pp. 83--85

Tamura, H., S. Takahashi, I. Hatono, and M. Umano (2000) “On a Behavioural Model of Analytic Hierarchy Process for Modelling the Legitimacy of Rank Reversal”, *Research and Practice in Multiple Criteria Decision Making*, pp. 173--184, Springer

Belton, V. and T. Gear (1983) “On a shortcoming of Saaty's Method of Analytic Hierarchies”, *OMEGA The International Journal of Management Science*, Vol. 11, No. 3, pp. 228--236