INVITED PAPER *Special Section on Concurrent Systems Technology*

Enhancing Software Project Simulator toward Risk Prediction with Cost Estimation Capability

Osamu MIZUNO[†a)], ***Member***, **Daisuke SHIMODA**[†b)], ***Nonmember***, **Tohru KIKUNO**[†c)], ***Member***, ***and*** **Yasunari TAKAGI**[††d)], ***Nonmember***

SUMMARY This paper presents an enhancement of a software project simulator to perform risk prediction with cost estimation capability.

So far, we have developed a software project simulator to simulate software development projects. In this simulator, a development process was described using Petri net model, and it was applied to some actual project data in a certain company successfully. On the other hand, we have also presented a risk predicting system to find "risky" projects by statistical analysis on risk questionnaire for project managers. In this approach, only the probability to be risky was calculated for a project. Thus, the managers in the company wanted to know a concrete proof why a software project becomes risky.

In this paper, to present the proof that a software project becomes risky, we try to enhance the previous project simulator so that the simulator can deal with risk factors. To consider the risk factors, we modify the previous simulator so that both the fluctuation of skill level and the deadline pressure can be represented by the parameters in the simulator. By using a case study, we confirm that the enhanced simulator can estimate the development cost under some typical risks. As a result, we can expect that the simulator shows how much the development cost of a risky project exceeds an estimate.

key words: *Software development project, Simulator, Risk factors, Cost estimation*

1. Introduction

Generally, a software development process includes concurrent execution of many activities such as design, coding, review, test, and debug. Thus, as software processes recently have become huge and complex, the management of software projects is difficult. As a result, various kinds of managemental problems such as cost excess, late delivery date, and low quality, have also become critical in industry. In order to solve such problems, software process improvement has been extensively performed[1].

In a certain company, the Software Engineering Process Group (SEPG) was established 8 years ago. The SEPG has tackled the process improvement activities in the company so far. We have already performed several joint works with the SEPG for software process improvement.

One of our successful joint works with the SEPG was the formal modeling of software processes by a Generalized Stochastic Petri Net(GSPN). Furthermore, in order to estimate the quality, cost, and duration of the software development process, we have developed a software project simulator based on the GSPN[2], [3]. The detailed parameters in the simulator were determined based on experience and data from software development processes in the company. Several experiments showed that software projects were simulated rather accurately according to the initial development plan.

On the other hand, we have also performed statistical analysis of project data collected from a number of projects. During the analysis, we have noticed that there were projects that appeared to be out of control from the project managers' viewpoint. We called such projects "risky projects." We then tried to develop a risk predicting system, which calculates the probability based on risk questionnaire for project managers and finds such risky projects according to the probability. An experimental application showed that most of the risky projects can be detected by the probability[4]. The project managers were, however, still skeptical because the probability does not provide a concrete proof why projects become risky. The managers in the company wanted to know such a concrete proof.

In this paper, we try to estimate development costs, which are an important metric to determine risky projects. To do so, the project simulator must have capability to deal with risk factors in the risk questionnaire and to estimate development costs.

For the previous simulator, it is impossible to simulate a project under risks, because the previous simulator cannot represent the risk factors. In the enhanced simulator, we implement a mechanism that adjusts parameters to deal with the influence of the risk factors. As a result, the following typical confusions of risky projects can be represented in the simulator: confusion caused by the fluctuation of the developer's skill level and the confusion caused by the deadline pressure[5].

Finally, we perform a case study to confirm whether a risky project can be simulated. The results

Manuscript received March 30, 2001.
Manuscript revised June 4, 2001.

[†]The authors are with Dept. of Informatics and Mathematical Science, Graduate School of Engineering Science, Osaka University.
[††]The author is with OMRON Corporation.
a) E-mail: o-mizuno@ics.es.osaka-u.ac.jp
b) E-mail: d-simoda@ics.es.osaka-u.ac.jp
c) E-mail: kikuno@ics.es.osaka-u.ac.jp
d) E-mail: taka@biwa.kusatsu.omron.co.jp

show that the enhanced simulator can estimate the development costs for both ideal case and risky case. As a result, we can expect that the simulator shows how much the development cost of a risky project exceeds an estimate.

2. Previous Works [2][4]

The Software Engineering Process Group (SEPG) has performed various activities to improve the software process in the company. We have cooperated with the SEPG and have made several empirical studies[2], [4].

2.1 Software project simulator

In order to estimate the quality and quantity of software development activities, we have developed a software project simulator based on Generalized Stochastic Petri Nets(GSPNs)[2]. The simulator consists of a Project model and a Process model. The Project model focuses 3 key components: activities, products, and developers (See Figure 1). The Process model includes a set of Activity models (See Figure 2), each of which specifies process activities in a software development.

Attributes of an activity
type
entry condition
exit condition
input products
output products
deadline
workload
developers

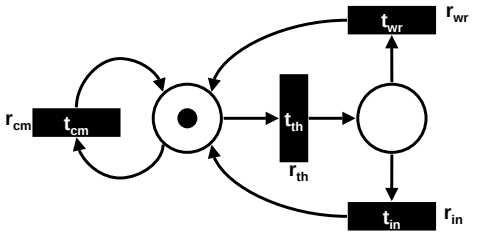
Attributes of a product
size
number of faults
completion rate

Attributes of a developer
skill level

Fig. 1 Template of Project model

A Project model describes the control flow of the components included in a project as shown in Figure 3. It also specifies resources and conditions for a project such as the assignment of developers, the schedule of development, and so on. An Activity model describes how to calculate development cost, duration, and the number of faults for each activity. Process model describes how to execute Activity models according to the specified conditions.

The Process model can take the influence of human factors into account by introducing the concept of "workload." The workload of an activity is defined as the total time needed for a developer with average capability to complete the activity. The workload can reflect the communication among the developers[6], the performance of CASE tools, and so on, and thus the



	t_{cm}	t_{th}	t_{wr}	t_{in}
Meaning	Communication	Think	Write	Write with a fault
Execution function	-	w = w + 1	s = s + 1	s = s + 1 f = f + 1

s : size of product
w : consumed workload
f : number of faults

Fig. 2 Design Activity model

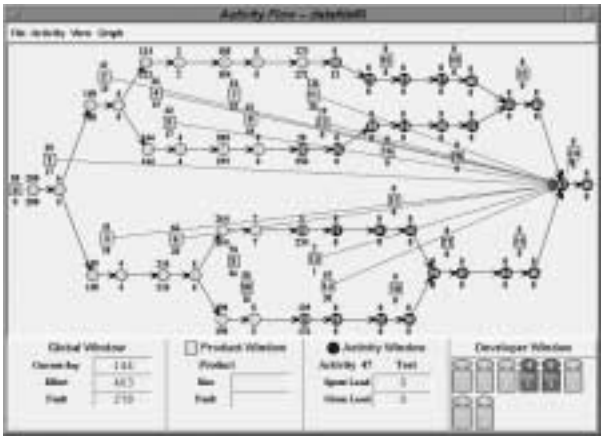


Fig. 3 Control flow of components

simulator can evaluate the dynamic aspects of software projects[7].

The Activity models in a Process model are described by GSPNs. The Activity model is prepared for each activity in a software development process, such as design, coding, review, test, and debug. An example of an Activity model for the design activity is shown in Figure 2.

Each transition of Figure 2 denotes the behavior of a developer in the activity. Transition t_{cm} represents the communication among developers, t_{th} represents the thinking of a developer, t_{wr} represents the writing of a developer without faults, and t_{in} represents the writing with injecting a fault. The firing delay of each transition is exponentially distributed and the firing rate of a transition is assigned as r_*. For example, the firing rate $r_{cm} = 0.2$ of transition t_{cm} means that the average firing delay of transition t_{cm} is $1/r_{cm} = 5$. Each firing rate is determined by the functions such as $r_{cm} = f(M, L, R)$ where M is the number of developers, L is experience level of developers, and R is a completion rate of an input product.

The token in the Activity model has some at-

tributes such as "consumed workload (w)," "number of injected faults (f)," and "size of output product (s)." When a transition is fired, a function attached to the transition is executed to change the attributes of the token. For example, in Figure 2, function $w = w + 1$ denotes the consumption of the workload.

The detail of the definitions of activities are described in the reference [8]. The prototype of the simulator was developed and several experiments have been done to show the effectiveness of the proposed simulator[3].

2.2 Risk predicting system

On the other hand, we have tried to predict the final status of software development projects by statistical methods[4]. The proposed method was based on a questionnaire for the risk factors in software development.

Table 1 Original questionnaire used in [4]

	Risk factors	Evaluation
Requirements	Required performance	High 3 - 0 Low
	Technical difficulty	Difficult 3 - 0 Easy
	Requirements specification review	Done | Not
	Number of requirements changes	#
Project Plan	Resultant products in the project plan	Described | Not
	Review plan in the project plan	Described | Not
	Project plan review	Done | Not
	Project plan review by the manager	Done | Not
Estimation	Optimism for the technical issues	Exists | Not
	Appropriate estimation	Done | Not
	Number of projects referred	#
	Needed time to make estimate	Days
Project Management	Unclear project management method	Unclear 3 - 0 Clear
	Possibility of size overcome	High 3 - 0 Low
	Possibility of cost overcome	High 3 - 0 Low
	Possibility of duration overcome	High 3 - 0 Low
	Taking track of the progress	Yes | No
	Frequency of progress report	#/Month
	Unresolved problems	#
	Review effort ratio	%
Developers	The morale of developers	Strong 3 - 0 Weak
	Number of leaders in the project	#
	Number of developers	#
	Average working time of developers	Hour/Day
	Organization chart	Made | Not
	Experience of the requirement describer	High 3 - 0 Low
	Average experience of the developer	High 3 - 0 Low
	Responsive person for each activity in the WBS	Specified | Not
Communication	Number of group meetings	#/Month
	Number of meetings with managers	#/Month
	Number of meetings with external groups	#/Month
External Risks	Untechnical pressure	Exists | Not
	Number of COTS packages	#
	Number of external companies cooperated with	#

First, we defined a "risky project" from the viewpoint of development cost and duration. We then designed the questionnaire that includes risk factors to occur in software development. We designed a questionnaire shown in Table 1 to be distributed to project managers of software projects[†]. The questionnaire includes 34 risk factors which are classified into 7 major categories: Requirements, Project plan, Estimation, Project management, Developers, Communication, and External risks. Then project managers return evaluations to each risk factor as shown in Table 1. The evaluations have the following three types: the binary evaluation (such as 'Yes' or 'No'), the ordinal evaluation (For example, for evaluations 'High', 'Relatively high', 'Relatively low', and 'Low', the values 3, 2, 1, and 0 are assigned, respectively), and the absolute value (such as the number of requirements changes).

Based on the responses to the questionnaire, we collected risk assessment data and applied the following logistic model to them:

$$P(Y|x_1, \cdots, x_n) = \frac{e^{b_0+b_1x_1+\cdots+b_nx_n}}{1 + e^{b_0+b_1x_1+\cdots+b_nx_n}}$$

where $x_1, \cdots, x_n$ are explanatory variables in the model, and Y is a binary dependent variable that represents whether a project is risky or not. P is the conditional probability that $Y = 1$(i.e. a project is risky) when the values of $x_1, \cdots, x_n$ are determined. We select the risk factors in the questionnaire as potential explanatory variables, and estimate the coefficients b_i's using the risk assessment data obtained from the responses to the questionnaire. The following is a logistic model constructed for the risk data at a certain company[4]:

$$P(Y|x_1, x_2) = \frac{e^{-5.251+2.727x_1+3.984x_2}}{1 + e^{-5.251+2.727x_1+3.984x_2}}$$

where x_1 and x_2 are the risk factors "Estimation" and "Project Plan," respectively[††].

We carried out an effectiveness analysis of the constructed model. The result showed that the constructed model can nicely predict risky projects based on the probability P for new data sets.

We also developed a WWW based system that distributes questionnaires to the developers, helps the statistical analysis by the SEPG, and predicts the final status of projects by applying the logistic model to the projects. The prototype of the predicting system has been developed, and applied to actual development projects in the company.

3. Needs for Enhancement

3.1 Cost estimation of risky project

As mentioned in subsection 2.2, the prediction of the final status of a project (risky or not) using the probability was successfully applied to actual projects. However, the project managers were still skeptical because they were provided with only the probabilities of risk. They actually wanted to know exactly what will happen in such risky projects.

We therefore tried to provide more concrete proof of risky projects. In more detail, we estimate the development cost of such risky projects, since the development cost is one of the important criteria to determine risky projects.

[†]To be exact, the questionnaire shown in Table 1 is slightly extended from that in reference [4].

[††]In [4], the risk factors were chosen from the categories in Table 1.

One of the possible ways to estimate the cost is to execute the project simulator under software risks. However, the previous simulator developed in [2] did not take the influence of software risks into account. Thus, in order to estimate the cost of risky projects, we enhanced the project simulator so that the software risks can be considered.

3.2 An approach for cost estimation

As a result of our investigation, we found that the following two confusions, which are closely related to the risk factors in Table 1, will have a major effect on the cost.

Confusion caused by a risky project: In a risky project, developers cannot bring their ability into full play[9]. That is, in a risky project, the skill level of a developer is reduced from his/her original level. It is known that the skill level of a developer significantly influences the cost of software projects[10], [11]. In the enhanced simulator, the developer's skill level can fluctuate according to the risks.

Confusion caused by deadline pressure: The deadline pressure causes many injected faults[5], [12], and thus increases the cost of the latter phases of a project. In a risky project, since schedule is frequently delayed[9], the influence of the deadline pressure increases greatly. If the development duration of an activity is 90% completed and the assigned workload is not completed 90%, then the developers feel deadline pressure. In the enhanced simulator, the deadline pressure is represented as a parameter in the fault injection rate.

As will be described in the next section, these situations are implemented by adjusting the parameters in the project simulator according to evaluation results for risk factors. As a result, we expect that the cost of risky projects can be estimated more correctly.

4. Enhancement of Project Simulator

In this section, we try to enhance the previous simulator so that the simulator can deal with the risk factors.

4.1 Selected risk factors

From Table 1, we selected risk factors to be added to the simulator. First, we identified and deleted 16 risk factors for which there was insufficient data for analysis. However, since important risk factors may be included in these deleted factors, we consider further analysis of them as an important future work. We then deleted 3 risk factors that can be represented directly as an initial input for the simulator. As a result, 15 risk factors in Table 2 remained to be selected and added to the enhanced simulator.

Table 2 Risk factors to be used in enhanced simulator

	Risk factors	Evaluation
Requirements	Requirements specification review	Done | Not
	Number of requirements changes	#
Project Plan	Resultant products in the project plan	Described | Not
	Project plan review	Done | Not
	Project plan review by the manager	Done | Not
Estimation	Optimism for the technical issues	Exists | Not
	Appropriate estimation	Done | Not
	Needed time to make estimate	Days
Project Management	Unclear project management method	Unclear 3 - 0 Clear
	Review effort ratio	%
Developers	The morale of developers	Strong 3 - 0 Weak
	Experience of the requirement describer	High 3 - 0 Low
	Average experience of the developer	High 3 - 0 Low
	Responsive person for each activity in the WBS	Specified | Not
External Risks	Untechnical pressure	Exists | Not

4.2 Parameters to represent the risk factors

In order to represent the risk factors in Table 2, we used the parameters in the project simulator. In the project simulator, the following parameters are used in firing rate functions:

- The skill level of developers (L)
- The mental stress caused by deadline pressure (D)
- The completion rate of the input products (R)
- The number of faults in the input products (F)
- The number of developers in an activity (M)
- The constants for each developer's behavior (K_*)
- The assigned workload by a development plan (WL)

For example, the firing rate functions transitions in Figure 2 are as follows[†]:

$$r_{cm} = K_{cm} \times \frac{M \times (M-1)}{L \times R}$$

$$r_{th} = K_{th} \times \frac{L}{D}$$

$$r_{wr} = K_{wr} \times L \times D$$

$$r_{in} = K_{in} \times \frac{F+1}{R \times D}$$

The initial values of these parameters must be given in a project description. An example is shown in Figure 4. It shows the specified values for the activities such as FD, FDR, PG, and so on (As shown in Figure 6). The number of developers(M) for Activity 0 is 2, the initial skill levels(L) of all developers in this project are 1, the completion rate(R) of Product 0 is 1.0, and the workload(WL) for the Activity 0 is 128.

In the enhanced simulator, the values of parameters are adjusted according to the given software risks. (The details for the adjusting parameters will be explained in Section 5.) In the following subsections 4.3 and 4.4, we will explain how the risky situations in subsection 3.2 are represented by adjusting the parameters.

[†]The definitions of firing rate functions are slightly extended from that of reference [2].

Activities' Definition

ID	Type	Assigned workload	Consumed Workload	Deadline	Injected faults	Detected faults	Input product(s)	Output product(s)	Developer(s)	Current condition	Start condition
0	FD	128	0	24	0	0	0	1	0,1	WAIT_START	0-NOT_START
1	FDR	16	0	0	0	0	1	1	0,1,2	NOT_START	0-FINISHED
2	PG	72	0	37	0	0	1	2	0,2	NOT_START	1-FINISHED
3	PGR	9	0	0	0	0	2	2	0,1,2	NOT_START	2-FINISHED
4	MT	48	0	45	0	0	1,2	3	0	NOT_START	3-FINISHED
5	MDB	0	0	45	0	0	3	1,2	0	NOT_START	4-FINISHED
6	FT	40	0	51	0	0	2	4	0	NOT_START	5-FINISHED
7	FDB	0	0	51	0	0	4	1,2	0	NOT_START	6-FINISHED

Products' Definition

ID	Size	Injected faults	Detected faults	Completion
0	10	0	0	1.0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

Developers' Definition

ID	Name	Skill	WorkTime
0	Willium	1	0
1	Richard	1	0
2	John	1	0

Fig. 4 Project description for the simulator

4.3 Confusion by fluctuating skill level

In risky projects, the skill of a developer can fluctuate (for instance, it falls down from 1.0 to 0.7). However, in the previous simulator, the value of the skill level L was a constant determined before a simulation begins. In the enhanced simulator, we adjust the value of L to represent a risky situation in a project.

Let us explain an example of a risky situation in a design activity. As shown in Figure 2, transitions t_{cm} and t_{th} fire selectively according to the firing rates r_{cm} and r_{th}. Consider the following two cases. Assume that $K_{cm} = 0.1$, $K_{th} = 0.2$, $M = 2$, $R = 1$, and $D = 1$. Assume that the assigned workload is 80 hours.

Case 1 (Ideal Case): Consider a case of no risk, in which the skill level $L = 1$. Firing rates r_{cm} and r_{th} are calculated as follows:

$$r_{cm} = 0.1 \times \frac{2 \times 1}{1 \times 1} = 0.2$$

$$r_{th} = 0.2 \times \frac{1}{1} = 0.2$$

These two equations simulate a situation where that communication occurs as frequently as the thinking ($r_{cm}/r_{th} = 0.2/0.2 = 1.0$).
In our simulator, the communication among developers is treated as a verbose behavior in the activity. So, communication is modeled as a wasting of time, that is, it does not consume any workload but only takes time. The cost of an activity increases according to the increase of time.
From the definition of Activity model in Figure 2, t_{th} fires 80 times during the activity, because the execution of an activity finishes when $w = 80$. Thus, t_{cm} also fires 80 times. Since the time needed for a communication behavior is 5 minutes, it takes 400 minutes (= 6.6 hours) more than expected.

Case 2 (Risky Case): Consider a case of a risky project where the skill level of developers becomes $L = 0.7$. r_{cm} and r_{th} are then calculated as follows:

$$r_{cm} = 0.1 \times \frac{2 \times 1}{0.7 \times 1} = 0.29$$

$$r_{th} = 0.2 \times \frac{0.7}{1} = 0.14$$

In this case, communication is about 2.1 times ($r_{cm}/r_{th} = 0.29/0.14 = 2.1$) more frequent than thinking. Thus, t_{cm} fires 168 times, and it takes 840 minutes (= 14 hours) more than expected.

As a result, since there are 2 developers($M = 2$) in this activity, the cost of Case 2 is $(14 - 6.6) \times 2 = 14.8$ person-hours more than Case 1.

4.4 Confusion by the deadline pressure

As mentioned earlier, in risky projects, the influence of deadline pressure may increase greatly. The value of D in the previous simulator was also a constant determined when a simulation begins. In the enhanced simulator, we adjust the value of D to represent deadline pressure.

Consider the following two cases in a design activity. Assume that $K_{wr} = 1.0$, $K_{in} = 0.1$, $R = 1$, $F = 0$, and $L = 1$. Assume that assigned workload WL is 80 hours, and scheduled duration to complete the activity is 10 days.

Case 1 (Ideal Case): Assume that 90% of workload(WL) is consumed when 9 days elapse. Since consumption of WL is on schedule, deadline pressure does not happen(See Case 1 in Figure 5). So, $D = 1.0$ holds during the activity.
In this case, firing rates for transitions t_{wr} and t_{in} are calculated as follows:

$$r_{wr} = 1.0 \times 1 \times 1 = 1.0$$

$$r_{in} = 0.1 \times \frac{0 + 1}{1 \times 1} = 0.1$$

According to Figure 2, the number of faults injected in the current product is calculated as follows:

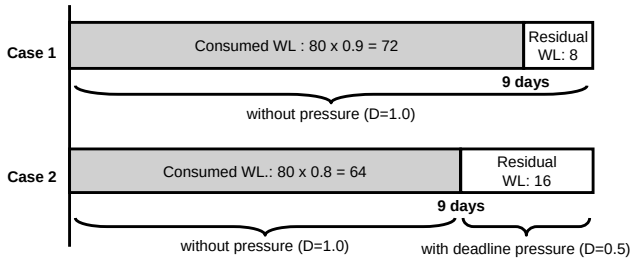


Fig. 5 Example of deadline pressure

$$f = 80 \times \frac{0.1}{0.1 + 1.0} = 7.3$$

Case 2 (Risky Case): Assume that 80% of WL is consumed when 9 days elapse. In this case, consumption of WL is behind schedule, and thus deadline pressure occurs† (See Case 2 in Figure 5). To represent the deadline pressure, the value of D is decreased from 1.0 to 0.5. During the consumption of the residual 20% of WL, the firing rates are changed as follows:

$$r_{wr} = 1.0 \times 1 \times 0.5 = 0.5$$

$$r_{in} = 0.1 \times \frac{0 + 1}{1 \times 0.5} = 0.2$$

The number of faults injected in the current product is calculated as follows:

$$f = (80 \times 0.8) \times \frac{0.1}{0.1 + 1.0} + (80 \times 0.2) \times \frac{0.2}{0.2 + 0.5} = 10.4$$

Faults will be removed in successive review or debug activities. If faults are detected in the review activity, it will take extra 30 minutes to remove each fault. That is, additional 1.5 person-hours are needed to remove the faults.

Furthermore, suppose that such faults are not detected in the review, and instead are detected in the test and debug activities. The activities are modeled to take 1 workload to detect a fault, and 1 workload to remove a fault. This implies that more than 6 person-hours (2 workloads × 3 faults) are needed to detect and remove such faults.

5. Implementation of Enhanced Simulator

5.1 Adjusting parameters for risk factors

In order to determine how to adjust the parameters, we interviewed the developers in the company. As a result of the interviews, we finally determined the parameters to represent the risk factors. Table 3 shows how the risk factors are implemented by the adjustment of parameters.

The left side of Table 3 shows the risk factors, and the right side shows the adjustment of parameters in the simulator to represent corresponding risk factors.

Most adjustments are specified by differences (plus or minus) with respect to the current values.

(1) For risk factors with a binary evaluation, we changed the values of the parameters according to one of the evaluations. For example, if the evaluation of "Requirement specification review" is "Not done", the parameters for 'deadline pressure' and 'completion rate' are decreased by 0.1 and 0.05, respectively, and 'injected fault' is increased by 2.

(2) For risk factors with ordinal evaluation, we changed the values of parameters according to the corresponding order. For example, if the evaluation of "The morale of developers" is ϵ ($0 \leq \epsilon \leq 3$), the skill level of developers are decreased by $0.03 \times (3 - \epsilon)$.

(3) For risk factors with evaluations using an absolute value, we changed the values according to step functions such as $w_1(\alpha)$, $d_1(\alpha)$ and so on. For example, if the evaluation of "Number of requirements changes" is α, the workload for the project is decreased by step function $w_1(\alpha)$ where $w_1(\alpha)$ generates values from 0 to 3 according to α.

The "review effort ratio" is the only exception. The value of the parameter "workload for the review activity" is set to χ% of the corresponding design/coding activity according to the evaluation. The value of a parameter is cumulatively increased or decreased if several risk factors influence the parameter.

Let us explain an implementation of a risk factor. If the risk factor "Requirement specification review" is "Not done," then the corresponding parameters in the simulator are changed. Because of the incompleteness of the requirements specification, the completion rate R is decreased by 0.05. The incompleteness of the requirements also influences the deadline pressure because such incompleteness is usually revealed in the final phase of an activity. The value of deadline pressure D is also decreased by 0.1. Since the lack of review implies several residual faults, the number of injected faults F in the initial product is increased by 2††.

5.2 Procedure of application

Here, we explain the procedure of application of the enhanced simulator to an actual project.

In order to estimate costs of projects, we prepared inputs for the enhanced simulator. The inputs for the simulator are a "project description" (See Figure 4) and

†The reason why such delay occurs is out of the scope of this example, but we can assume that a certain risk has already existed in the activity.

††Note that the values shown in Table 3 are heuristic ones based on our experience in the company. The generalization of such parameter setting remains as a future work.

Table 3 Adjustment of parameters according to evaluation result

	Risk factors		Adjustments of parameters
	Description of risk factors	Evaluation results	
Requirements	Requirements specification review	Not done	Deadline pressure(D): decrease by 0.1. Completion rate(R) for initial product: decrease by 0.05. Injected faults(F) in initial product: increase by 2.
	Number of requirements changes	# of changes (α)	Workload for the project: increase by $w_1(\alpha)$ [max. 10%]. Deadline pressure(D): decrease by $d_1(\alpha)$ [max. 0.1]. Completion rate(R) for initial product: decrease by $r_1(\alpha)$ [max. 0.05]. Injected faults(F) in initial product: increase by $f_1(\alpha)$ [max. 3].
Project Plan	Resultant products in the project plan	Not described	Completion rate(R) for initial product: decrease by 0.05.
	Project plan review	Not done	Completion rate(R) for initial product: decrease by 0.05.
	Project plan review by the manager	Not done	Completion rate(R) for initial product: decrease by 0.05.
Estimation	Optimism for the technical issues	Exists	Workload for the project: decrease by 5%. Deadline pressure(D): increase by 0.1.
	Appropriate estimation	Not done	Communication rate(K_{cm}) in the design and coding: increase by 0.05. Thinking rate(K_{th}): decrease by 0.05.
	Needed time to make estimate	days (β)	Workload for the project: decrease by $w_2(\beta)$ [max. 5%]. Communication rate(K_{cm}) in design & coding: increase by $r_2(\beta)$ [max. 0.05]. Deadline pressure(D): decrease by $d_2(\beta)$ [max. 0.1].
Project Management	Unclear project management method	Unclear 3 - 0 Clear (δ)	Communication rate (K_{cm}) in all activities: increase by 0.01 x δ. Thinking rate(K_{th}): decrease by 0.01 x δ.
	Review effort ratio	Ratio in % (χ)	Workload for the review: set to χ% of corresponded activity's effort.
Developers	The morale of developers	Strong 3- 0 Weak (ε)	Skill level(L) of all developers: decrease by 0.03 x (3-ε).
	Experience of the requirement describer	High 3 - 0 Low (ϕ)	Skill level(L) of requirement describer: decrease by 0.03 x (3-ϕ). Completion rate(K_{cm}) for initial product: decrease by 0.02 x (3-ϕ). Injected faults(F) in initial product: increase by 1 if ϕ = 3.
	Average experience of the developer	High 3 - 0 Low (γ)	Skill level(L) of all developers: decrease by 0.03 x (3-γ).
	Responsive person for each activity in the WBS	Not specified	Skill level(L) of all developers: decrease by 0.05.
External Risks	Untechnical pressure	Exists	Workload for the project: decrease by 10%. Deadline pressure(D): decrease by 0.2.

"evaluations of risk factors." The project description is constructed from the plan of the target project. It specifies the initial assignment of workload, assignment of developers, initial parameters in the simulator, and so on.

The evaluations of risk factors are collected from project managers of the target projects. The parameters in the simulator are adjusted by the evaluations of risk factors according to Table 3. The simulator then calculates the resultant cost for the project under the risk factors. By comparing the resulting cost in a risky situation with an initial estimate, we can see how much the cost exceeds the estimate under specified risks.

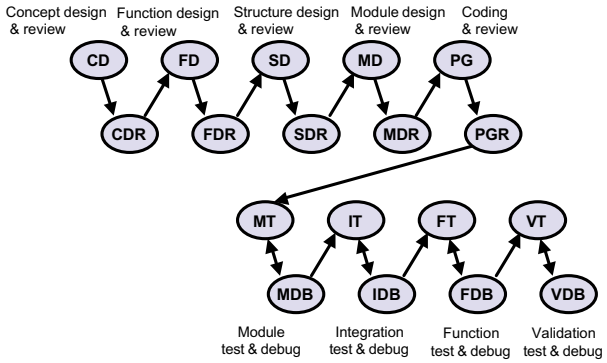


Fig. 6 Development process of target projects

6. Case Study

In order to show the effectiveness of the enhanced simulator, we made a case study. The objective of this case study is to show that the new simulator can estimate the cost according to the applied risk factors.

6.1 Target projects

In the case study, we use three projects (PR_1, PR_2, and PR_3), each of which is a typical development of embedded software for ticket vending machines in a certain company. They were executed in 1997. The development process of these projects was the standard waterfall model, as shown in Figure 6.

6.2 Estimation by previous simulator

First, we simulated the 3 projects with the previous simulator so that the estimated costs reflect the planned costs as correctly as possible. The estimated costs, calculated by the simulator, for the projects PR_1, PR_2, and PR_3 are shown in Table 4. For comparison, the planned cost and the actual cost for each project are also shown. By comparing the planned cost and estimated cost in Table 4, we can see that the simulator can estimate the development cost quite correctly in terms of the project plan for each project.

Table 4 Estimated costs of 3 projects (person-days)

		CD	FD–PG	MT–VDB	Total
PR_1	Planned cost	31	73	41	145
	Estimated cost	34	78	50	162
	Actual cost	31	84	46	161
PR_2	Planned cost	–	78	14	92
	Estimated cost	–	72	37	109
	Actual cost	–	99	19	118
PR_3	Planned cost	76	33	30	139
	Estimated cost	70	42	31	143
	Actual cost	108	81	57	246

6.3 Estimation by enhanced simulator

We regret to say that we do not have actual evaluations for the risk questionnaire for the 3 projects, PR_1, PR_2, and PR_3, since these projects had already been completed in 1997. We therefore tried to see the effect of the implemented risk factors by constructing virtual situations.

We constructed two cases of risks: "Ideal (no risk) Case" and "Risky Case." The evaluations of two cases are shown in Table 5. The evaluations for "Risky Case" are entirely bad, and for "Ideal Case" are entirely good. The probabilities of being risky are calculated by the risk prediction model[4] for the cases "Ideal Case" and "Risky Case." They are 0.8% and 95.6%, respectively.

Table 5 Two cases in the case study

	Risk factors	Ideal Case	Risky Case
Requirements	Requirements specification review	Done	Not done
	Number of requirements changes	0	10
Project Plan	Resultant products in the project plan	Described	Not described
	Project plan review	Done	Not done
	Project plan review by the manager	Done	Not done
Estimation	Optimism for the technical issues	Not Exists	Exists
	Appropriate estimation	Done	Not done
	Needed time to make estimate	5	0
Project Management	Unclear project management method	0 (Clear)	3 (Unclear)
	Review effort ratio	20	3
Developers	The morale of developers	3 (Strong)	0 (Weak)
	Experience of the requirement	3 (High)	0 (Low)
	Average experience of the developer	3 (High)	0 (Low)
	Responsive person for each activity in the WBS	Specified	Not specified
External Risks	Untechnical pressure	Not exists	Exists

6.4 Discussions

Table 6 shows the estimated costs calculated by the enhanced simulator. In Table 6, "Decision by [4]" shows the evaluation of being risky or not, based on the risky project's decision criteria in reference [4] for the resultant project documents. Since there are no serious problems in the documents for the projects PR_1 and PR_2, they are considered as "Non-risky." On the other hand, in the resulting documents of PR_3, the following descriptions remain:

Table 6 Comparison of development costs

	Decision by [4]	Actual cost	Estimated cost	
			Ideal Case	Risky Case
PR_1	Non-Risky	161	149	230
PR_2	Non-Risky	118	118	149
PR_3	Risky	246	123	207

- At the beginning, a certain part of the system was expected to be reused from other systems. However, it became clear during the functional design that the part must be modified.
- An experienced developer, who knows that part of the system well, was not available.
- As a result, the planned delivery date of the system was behind schedule.

The problem clearly corresponds to the risk factors in Table 2. From these facts, we consider that PR_3 was a "Risky" project.

The results for the "Ideal Case" for projects PR_1 and PR_2 show that they are close to the actual costs. On the other hand, estimated costs are relatively high in the case of the "Risky Case." Among them, the results for the "Risky Case" of PR_3 (that is, 207) is close to the actual cost (that is, 246).

As a result of this case study, we validated the execution of the enhanced simulator under risks at the company.

7. Conclusion

We have proposed an enhanced project simulator that can evaluate development costs under risks. In the new simulator, the parameters such as skill level of developers and the deadline pressure are adjusted according to evaluations for the risk factors. The experimental evaluation showed the validity of the new simulator.

The future work includes:

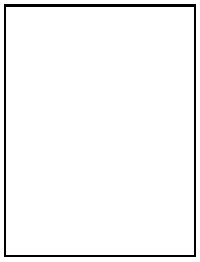
- A generalized method for determining values of parameters for risk factors should be developed.
- We must apply the new project simulator to actual development projects, from which we can collect data on actual risks.
- We have to investigate more risk factors that were not included in this study.
- We must develop a simulator based method for replanning risky projects.

Acknowledgment

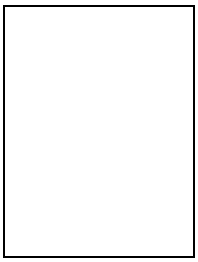
Authors would like to thank Associate Prof. Shinji Kusumoto and Assistant Prof. Tatsuhiro Tsuchiya of Osaka University for their discussions and advice to our analysis in this paper.

References

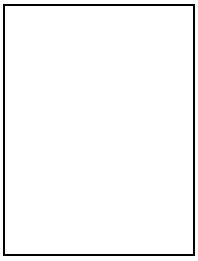
[1] W. S. Humphrey, Managing the Software Process, Addison Wesley, MA, 1989.
[2] S. Kusumoto, O. Mizuno, Y. Hirayama, T. Kikuno, Y. Takagi, and K. Sakamoto, "A new project simulator based on generalized stochastic Petri-Net," Proc. of 19th International Conference on Software Engineering(ICSE'97), pp.293–303, 1997.
[3] S. Kusumoto, O. Mizuno, T. Kikuno, Y. Hirayama, Y. Takagi, and K. Sakamoto, "Software project simulator for effective process improvement," Trans. of Information Processing Society of Japan, vol.42, no.3, pp.396–408, 2001.
[4] O. Mizuno, T. Kikuno, Y. Takagi, and K. Sakamoto, "Characterization of risky projects based on project managers' evaluation," Proc. of 22nd International Conference on Software Engineering(ICSE2000), pp.387–395, 2000.
[5] T. Furuyama, Y. Arai, and K. Iio, "Fault generation model and mental stress effect analysis," Journal of Systems and Software, vol.26, pp.31–42, 1994.
[6] F. P. Brooks, Jr., The Mythical Man-Month, Addison Wesley, MA, 1975.
[7] T. K. Abdel-Hamid, "The dynamics of software project staffing: A system dynamics based simulation approach," IEEE Trans. Software Eng., vol.15, no.2, pp.109–119, 1989.
[8] Y. Hirayama, "Quantitative evaluation of software process based on hierarchical project management model," Master dissertation, Osaka University, 1996.
[9] E. Yourdon, Death March : The Complete Software Developer's Guide to Surviving 'Mission Impossible' Projects, Prentice Hall Computer Books, 1997.
[10] K. Matsumoto, S. Kusumoto, T. Kikuno, and K. Torii, "An experimental evaluation of team performance in program development based on model – Extension of programmer performance model," Journal of Information Processing, vol.15, no.3, pp.466–473, 1992.
[11] H. Sackman, W. J. Erickson, and E. E. Grant, "Exploratory experimental studies comparing online and offline programming performance," Comm. ACM, vol.11, no.1, pp.3–11, 1968.
[12] M. V. Genuchten, "Why is software late? An empirical study of reason for delay in software development," IEEE Trans. Software Eng., vol.17, no.8, pp.582–590, 1991.



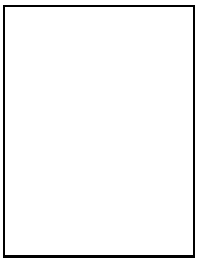
Osamu Mizuno received B.E. and M.E. degrees in information and computer sciences from Osaka University in 1996 and 1998, respectively. He has been working for Osaka University since 1999. He is currently a Research associate in the Department of Informatics and Mathematical Science at Osaka University. His research interests include the software process and the software quality assurance technique. He is a member of the IEEE.



Daisuke Shimoda received B.E. degree in information and computer sciences from Osaka University in 2001. He is currently a master course student in the Department of Informatics and Mathematical Science at Osaka University. His research interest includes the software process simulation.



Tohru Kikuno received M.S. and Ph.D. degrees from Osaka University in 1972 and 1975, respectively. He joined Hiroshima University from 1975 to 1987. Since 1990, he has been a Professor in the Department of Informatics and Mathematical Science at Osaka University. His research interests include the quantitative evaluation of software development processes and the analysis and design of fault-tolerant systems. He served as a program co-chair of the 1st International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'98) and of the 5th International Conference on Real-Time Computing Systems and Applications (RTCSA'98). He also served as a general co-chair of the 2nd International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC '99).



Yasunari Takagi received B.E. degree in information and computer science, from Nagoya Institute of Technology in 1985. He has been working for OMRON Corporation.