How does individual variability influence schedule risk?: A small simulation with experiment data

Lutz Prechelt (prechelt@ira.uka.de)
Barbara Unger (unger@ira.uka.de)
Fakultät für Informatik
Universität Karlsruhe
76128 Karlsruhe, Germany
http://wwwipd.ira.uka.de/EIR/

Technical Report 1999-12

September 1, 1999

Abstract

The present report is a follow-up to our report on an experiment for investigating the effects from Personal Software Process (PSP) training [1]. It uses the work time data from the experiment plus several simplifying assumptions in order to assess by stochastic simulation how much a reduction in the performance variability of individual programmers might reduce the uncertainty of project time requirements.

Contents

1 Introduction

In our experiment for investigating the effects of PSP training each member of a PSP-trained group of subjects and each member of a non-PSP-trained control group solved the same programming assignment. We will not iterate the details here and refer the reader to [1] instead.

In this experiment, the mean performance of the two groups did not differ much for most of the performance measures investigated (such as productivity in lines of code per hour or total working time in hours). However, we observed that the variability of these measures across the members of each group was often smaller among the members of the PSP-trained group.

Using several simplifying assumptions, we will now investigate the effects that this reduced variability might have on the probability with which a correctly devised project schedule will be overrun (or underrun).

2 Starting point

As a starting point, we use the actual data on productivity, measured in LOC/hour, observed in our experiment. This data is shown in Figure 1.

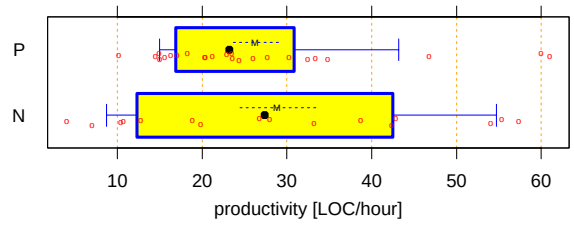

Figure 1: Productivity measured as the number of statement LOC written per hour. Top: PSP group. Bottom: non-PSP group. The box ranges across the middle half of the data; the whiskers indicate the top and bottom 10%. The fat dot is the median, the 'M' marks the arithmetic mean, the dashed line around it is plus/minus one standard error of the mean. For a more detailed explanation of this kind of box plots see [1].

The difference in the width of the box (the extension of the middle half of the data) is statistically significant as determined by a Bootstrapping test ($p = 0.04$. With a confidence of 80%, the difference in box width is larger than 7.4 LOC/hour).

3 Assumptions and definitions

1. Assume the software process consists of design, implementation, and test and these phases are performed strictly after one another.

   Let there be 2 independent design tasks (D_1, D_2), which are done in parallel, then 8 independent implementation tasks $(I_1, I_2, \ldots I_8)$, also done in parallel, and finally 3 independent test tasks (T_1, T_2, T_3), also done in parallel. The tasks of one phase all start at the same time; the next phase starts as soon as the longest task of the previous phase finishes.

2. Assume the manager has good knowledge of the expected productivity p_i for each member i of the team. The manager finds a work assignment that perfectly balances the size s of each subtask such that all expected work times are the same, e.g., $s(T_1)/p_j = s(T_2)/p_k = s(T_3)/p_l = t_{expect,T}$ etc.

   For simplicity, we assume that each phase has the same total size. The expected total project duration is therefore $t_{expect} := t_{expect,D} + t_{expect,I} + t_{expect,T}$.

   The p_i are drawn from historical productivity data for each team member, such as the data shown in Figure 1.

3. Assume the manager can estimate each individual productivity for a given particular project with an average absolute inaccuracy that is only as large as the uncertainty of the mean productivity of the team as a whole. In Figure 1, this inaccuracy is represented by the length d of the dashed line around the mean.

Note that (3) contains the crucial assumption that will produce the effect shown below: The given manager estimation inaccuracy implicitly assumes that the reduction in group variance that we observed in the experiment is also accompanied by a corresponding reduction in the task-to-task variance[1] of each individual in the group. We cannot observe the latter in the experiment because each individual performs only a single task.

4 Simulation and results

We can now simulate the risk of many projects by computing their actual duration as follows: For each phase P of each project select subteam members i at random and assume appropriate work assignments; member i works on a task of size s_i. Compute random productivity deviations d_i for each member.[2] Now determine the actual work time $t_i := t_{act,i} := s_i/(p_i + d_i)$ of each member.

The largest resulting actual time determines the phase duration. We compute 2+8+3 actual times t_i as above and then compute the total project duration as $t_{act} := \max(t_1, t_2) + max(t_3, t_4, \ldots, t_{10}) + max(t_{11}, t_{12}, t_{13})$. The deviation (in percent) from the plan for this

[1] more precisely: the task-to-task deviations from that person's mean performance

[2] Since d_i has a normal distribution, $p_i + d_i$ could become negative. Thus, we limit the deviation to factor 4, that is: $0.25 \leq p_i/(p_i + d_i) \leq 4$.

project is then $deviation := 100 \cdot (t_{act}/t_{expect} - 1)$. By simulating 100 projects in this manner for each of the groups, we obtain deviation distributions as shown in Figure 2.

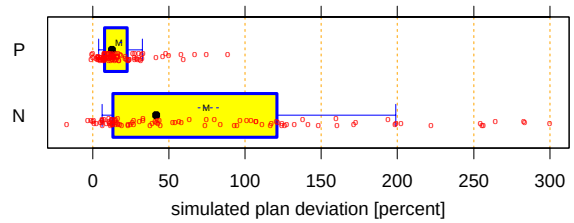

Figure 2: Deviations from the expected project time for 100 simulated projects in each group. Each expectation was computed from the expected productivity of each team member as predicted from Figure 1.

As we see, the consequence of the modest variability difference in Figure 1 is rather impressive: the mean deviation from the plan is four times as large for teams composed of N group members as for teams of P group members; catastrophic schedule overruns occur in the N group only.

5 Discussion and conclusions

A bold conclusion from this simulation would be the following: "A group of programmers whose group performance variability was reduced by giving them PSP training allows for development schedules with less slack time and puts much less burden on risk management."

However, one must obviously be quite careful before making such a statement (and therefore we do *not* make it). There are at least two major threats to the validity of the statement:

1. The model we applied for the partitioning, schedule planning, and execution of a software project is rather naïve. In particular, both the work partitioning and the performance of the individuals are not constant throughout a real project but rather are changing in a day-to-day feedback loop coupled with the risks of schedule overruns as observed by the project participants.
2. In the given context (i.e., our experiment), the assumption that reduced group variability is accompanied by reduced individual variability is just that: an assumption. Our experiment design was not suitable for substantiating it. Therefore, our simulation may not apply to PSP training (but might still be applicable to some other case of reduced individual performance variance).

Despite these problems, our simulation shows that it may be worthwhile to search for possibilities for reducing individual variability. It is quite plausible that PSP training has such effects, hence we deem it worth further investigation.

References

[1] Lutz Prechelt and Barbara Unger. A controlled experiment on the effects of PSP training: Detailed description and evaluation. Technical Report 1/1999, Fakultät für Informatik, Universität Karlsruhe, Germany, March 1999. ftp.ira.uka.de.