Cost Effectiveness of Programming Methods
– A Replication and Extension

Completed Research Paper

Wenying Sun
Computer Information Sciences
Washburn University
nan.sun@washburn.edu

Hee Seok Nam
Mathematics and Statistics
Washburn University
heeseok.nam@washburn.edu

Abstract
This paper is part of a comprehensive research project on the economics of pair programming. The purpose of the overall research is to thoroughly examine the economic advantages pair programming could gain over solo programming, identify situations where pair programming does not have economic advantages, and scenarios where solo programming is more cost effective than pair programming. This paper, due to space limitation, only reports results from one of the several studies we conducted. This paper focuses on one major previous publication, thoroughly examines its methodology and findings, and provides extensions we believe to be more robust and comprehensive. Our analysis results demonstrate pair programming and solo programming both have their own economic advantages and those advantages are dependent of a variety of factors.

Keywords
Pair programming, solo programming, net present value, breakeven unit value, breakeven unit value ratio.

Introduction

Solo programming is the traditional programming method where one programmer works on the programming task alone. Pair programming is a programming method where two programmers work on the same programming task side by side in front of one computer (Beck and Andres, 2004; Williams, Kessler, Cunningham, and Jeffries, 2000; Arisholm, Gallis, Dybå, and Sjøberg, 2007). In pair programming, one programmer is the driver, and the other is the navigator. The driver sits in front of the computer screen, types the code, and pays close attention to the coding details. The navigator sits beside the driver, reviews the code, and takes the lead in developing alternative strategies in the event of a problem. The programmers change roles periodically during the project to avoid role fatigue.

Despite growing interest in pair programming, issues remain that prevent the majority of organizations from adopting pair programming. The concern of increased overall project cost is a major obstacle. Previous studies, both anecdotal stories and academic research, are split in their conclusions regarding whether pair programming reduces the overall cost of a software development project compared to solo. Some suggest pair programming will reduce the overall cost of a project while others believe the benefits of pair programming do not justify the increased expense of the second programmer. Furthermore, two major studies, based on economic theories of software development, yielded markedly different conclusions. In one study pair programming was more cost effective than solo programming in all situations (Erdogmus and Williams, 2003), whereas in the other the economic benefit of pair programming depended on several factors (Padberg and Müller, 2003). Since the existing literature provides conflicting answers to the issue of cost, corporate decision makers do not have guidelines to follow to resolve this bottom line issue. Without clear cost benefits, given the fact that the majority of the software development companies are financially conscious, transition from solo to pair programming is difficult to argue for. To fill this gap, this research extends findings from prior studies and aims to answer the following research questions: 1) Is pair programming more cost effective than solo programming in all
situations? 2) If not, then in what situation is pair programming more cost effective than solo programming?

This paper reports results from one of the several studies the authors conducted. This paper focuses on one major publication - Erdogmus and Williams (2003), thoroughly examines its methodology and findings, and provides extensions we believe to be more robust and comprehensive. Despite its narrow focus, we believe this paper has several merits. First, it thoroughly addresses an important issue – cost effectiveness of pair and solo programming based on solid economic and mathematical terms. Second, unlike many previous publications which tend to champion a single programming approach, this study takes an objective stand and attempts to identify the variety of situations where one method has advantages over the other. Thirdly, practitioners could adopt our formulas, plug in their numbers, and gain ideas of which method they should adopt based on their conditions.

The paper is organized as follows. The first section provides theory and methods we adopted to conduct this study. Next, we report results from replications and extensions. Finally, we note some limitations of our approach and discuss directions for future research.

Theory and Methodology

This study is based on the economic theory of software development – the economic feasibility of a software project. Several methods are present to measure the economic feasibility. One method is breakeven analysis. Ergomus and Williams (2003) used the following formula to compare the two development methods: 

\[ \text{Breakeven Unit Value Ratio (BUVR)} = \frac{\text{BUV (solo)}}{\text{BUV (pair)}} \]

BUV is the threshold value of V above which NPV is positive; V is measured in $/LOC and represents the fixed increase in earned value per each additional unit of output produced. As the ratio increases, the advantage of pair over solo increases. In this study, we adopted the mathematical formulation developed by Erdogmus and Williams (2003). Our other studies (not reported here due to space constraints) investigate other alternatives.

We first examined and replicated the results from Erdogmus and Williams (2003). We then identified additional findings and applied other analysis processes we believed to be more comprehensive and robust. The following presents the procedures we followed. Please note even though they may not be familiar names in the IS field, both Erdogmus and Williams are leading authors on pair programming, and Erdogmus is a highly respected expert on economics of software development besides others.

Two value realization models were considered: single-point delivery, and continuous delivery. In single-point delivery model, rework immediately follows the development phase, and once all post-deployment defects are repaired, the project is complete (Figure 1). In incremental delivery model, new code is developed, deployed, and reworked in small increments; value is realized in small increments (Figure 1b).

![Figure 1. Two Delivery Models of Value Realization (Erdogmus and Williams 2003, p. 297)](image)

Their study relied on breakeven analysis (p. 285). NPV (net present value) varies as V varies from 5% to 30% of the unit labor cost C. V is measured in $/LOC (lines of code) and represents the fixed increase in earned value per each additional unit of output; the labor cost C is set to 50 and a work schedule of \( h_y = 1837.5 \text{ hours/year} \) is assumed; \( \omega \) is the output in KLOC (a thousand of lines of code) which varies from 1 to 16. (p. 300)
Below are the formulas for calculating NPV, BUV (Break-even Unit Value), and BUVR (Break-even Unit Value Ratio) in single and continuous delivery models. The formulas with s as the subscript are for the single delivery model, and the ones with c as the subscript are for the continuous delivery model.

\[
NPV_s = V \omega e^{-\frac{r\omega}{\pi\beta\gamma}} - \frac{n_{CN}}{r} \left\{ \frac{1}{1 + e^{-\frac{r\omega}{\pi\beta\gamma}}} + \varepsilon \right\}; \quad \varepsilon = \frac{\rho}{\rho + \beta\pi}
\]

\[
BUV_s = \frac{h_{CN}}{\omega r} e^{\frac{r\omega}{\pi\beta\gamma}} \left\{ \frac{1}{1 + e^{-\frac{r\omega}{\pi\beta\gamma}}} + \varepsilon \right\}
\]

\[
BUVR_s = \frac{BUV_{R,solo}}{BUV_{R,pair}}
\]

\[
BUV_c = \frac{cN}{\pi\epsilon}; \quad BUVR_c = \frac{BUV_{R,c,solo}}{BUV_{R,c,pair}}
\]

To calculate the NPV and BUVR, Erdogmus and Williams used the parameters in Table 1 below where π is productivity (LOC/hour), β is defect rate (defects/LOC), and ρ is rework speed (defects/hour) (p. 291). They stated most of the parameters were derived from previous studies (e.g. Jones 1997; Cockburn and Williams 2001; Williams et al. 2000), and the rework speed for a pair was based on the assumption that pairs could achieve rework productivity gains comparable to those reported for the initial development activities.

Solo = \{N = 1, \pi = 25.0, \beta = 0.00585, \rho = 0.0303\}

Pair = \{N = 2, \pi = 43.478, \beta = 0.00351, \rho = 0.0527\}

**Table 1. Parameters for Solo and Pair**

We first re-generated the graphs that demonstrated NPV/BUV as a function of unit value V and output ω. Next, we replicated the three sensitivity analyses the authors conducted. We then conducted two additional analyses. In Analysis 1, we applied the same percentage of improvement across all the three parameters (productivity, defect rate, and rework speed) at the same time. In Analysis 2, we changed % of improvement in one parameter while setting the values of the other two parameters equal between solo and pair. To better address the behavior of BUVR, we performed 10 million iterations for four scenarios. In scenario 1, productivity \( \pi_p = 2\pi_s/x \) where \( X \sim N(1.15, 0.1^2) \), the other two parameters used Utah data. Scenario 2 was on defect rate \( \beta_p = y\beta_s \) where \( Y \sim N(0.6, 0.1^2) \), the other two parameters used Utah data. The third one was on Rework speed \( \rho_p = 2\rho_s/z \) where \( Z \sim N(1.15, 0.1^2) \), the other two parameters used Utah data. In the fourth scenario, we applied Productivity \( \pi_p = 2\pi_s/x \) where \( X \sim N(1.15, 0.1^2) \), Defect rate \( \beta_p = y\beta_s \) where \( Y \sim N(0.6, 0.1^2) \), Rework speed \( \rho_p = 2\rho_s/z \) where \( Z \sim N(1.15, 0.1^2) \) at the same time.

Another question arises: what if some organizations cannot achieve the pair programming advantages as the ones reported in previous literature? What will be the behavior of BUVR’s? To answer this question, we repeated the above scenarios with the following changes: 1) pair has advantages in two parameters; 2) pair has an advantage in one parameter only; 3) pair does not have advantage in any of the three parameters.

Lastly, we attempted to solve the equations algebraically.

**Results**

The following are results from replication, extension, and additional analysis.

**Replication**
First, we replicated the graph that demonstrates NPV as a function of unit value \( V \) and output \( \omega \) (Figure 2a). BUV for pair using single delivery is given as Figure 2b.

![NPV and BUV as Function of V and \( \omega \)](image)

**Figure 2. NPV and BUV as Function of V and \( \omega \)**

In Figure 2a, the NPV = 0 plane splits the V-output space into feasible (NPV > 0) and infeasible (NPV < 0) regions. BUV is calculated as the threshold value of V above which the NPV is positive: \( \text{BUV} = \min \{V | \text{NPV} \geq 0\} \). Breakeven unit value ratio (BUVR) is \( \frac{\text{BUV}_{\text{solo}}}{\text{BUV}_{\text{pair}}} \). Values of BUVR greater than unity (unity being one) indicate an advantage for pairs; values smaller than unity indicate an advantage for soloists. As this ratio increases, the advantage of pairs over soloists also increases. (p. 300)

Erdogmus and Williams concluded the economic advantage of pairs over soloists is evident in both single-point delivery and continuous delivery models. BUVR is at least 2.24 when the discount rate is zero (p. 305). In the continuous delivery model, BUVR is 1.73, representing a 42% (1/1.73) advantage for pairs over soloists (p. 309). Our replication suggests BUVR being 1.5 for single-point delivery and 1.3 for continuous delivery. Therefore, both Erdogmus and Williams and our replication demonstrate an advantage for pairs over soloists when parameters presented in Table 1 are applied in the calculations.

We then replicated the three sensitivity analyses Erdogmus and Williams conducted. First, they changed the percentage of improvement in productivity (ranging from 1% to 400%) while holding the other two (defect rate and rework speed) constant. They found in single delivery, BUVR ranges from 1.3 to 1.5, and in continuous delivery, BUVR ranges from 1.1 to 1.5. Since BUVR is greater than 1, that means pair is better than solo. Figure 3a below is our replication of their first analysis.

Then, they varied the percentage of Improvement in defect rate (ranging from 1% to 100%) while holding the other two (productivity and rework speed) constant. When using benchmark (60%) from the studies conducted at the University of Utah, BUVR are 2.2 and 1.7 for single and continuous delivery respectively. Figure 3b is our replication of their second analysis.

Lastly, Erdogmus and Williams varied the percentage of Improvement in rework speed (ranging from 1% to 400%) while holding the other two (productivity and defect rate) constant. When using the benchmark (74%) from Utah studies, BUVR are 1.5 and 1.3 for single and continuous delivery respectively. Figure 3c is our replication of their third analysis.
All three analyses demonstrate there is an advantage of pairs over soloists when parameters from previous studies (presented in Table 1) are based for calculations. However, if we look into all the possible ranges of improvement on the parameters, we discover some new findings.

In the second analysis (the case of varying percentage of improvement in defect rate while having the other two constant), in single delivery, BUVR ranges from 0.8 to 4.4. BUVR is not always greater than 1, meaning pairs are not always better than soloists. With improvement of 12% or under, BUVR is less than 1, meaning soloists are better than pairs. With improvement of 13% to 19%, BUVR is around 1 (1.01 to 1.09), meaning soloists and pairs perform similarly. With improvement of 20% or above, BUVR is 1.1 to 4.4, meaning pairs are better than soloists.

In continuous delivery, BUVR ranges from 0.8 to 5.1. BUVR is not always greater than 1, meaning pairs are not always better than soloists. With improvement of 15% or under, BUVR is less than 1, meaning soloists are better than pairs. With improvement of 16% to 25%, BUVR is around 1 (1.00 to 1.09),
meaning soloists and pairs are similar. With improvement of 26% or above, BUVR is 1.1 to 5.1, meaning
pairs are better than soloists.

In the third analysis (the case of varying the percentage of Improvement in rework speed), in single
delivery, BUVR ranges from 0.8 to 3.6. BUVR is not always greater than 1, meaning pairs are not always
better than soloists. With improvement of 18% or under, BUVR is less than 1, meaning soloists are better
than pairs. With improvement of 19% to 29%, BUVR is around 1 (1.01 to 1.09), meaning soloists and
pairs are similar. With improvement of 30% or above, BUVR is 1.1 to 3.6, meaning pairs are better than
soloists.

In continuous delivery, BUVR ranges from 0.8 to 2.5. BUVR is not always greater than 1, meaning pairs
are not always better than soloists. With improvement of 23% or under, BUVR is less than 1, meaning
soloists are better than pairs. With improvement of 24% to 39%, BUVR is around 1 (1.00 to 1.09),
meaning soloists and pairs are similar. With improvement of 40% or above, BUVR is 1.1 to 2.5, meaning
pairs are better than soloists.

Table 2 below summarizes the situations where % of improvement in a given parameter leads to BUVR
value to be less than 1, around 1, and greater than 1 while fixing other parameters using the results from
the Utah studies. From the table, one can tell it takes about at least 20% improvement in defect rate in
pair in order for the pairs to be better than soloists, it takes at least 30% improvement in rework speed in
pair in order for the pairs to be better than soloists. Note that Utah values for defect rate and rework
speed are 40% and 73.9%, respectively. A joint effect on BUVR will be discussed in the next section.

<table>
<thead>
<tr>
<th>% improvement of pair over solo</th>
<th>BUVR &lt; 1</th>
<th>BUVR ≈ 1</th>
<th>BUVR &gt; 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect rate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single</td>
<td>12% or under</td>
<td>13% to 19%</td>
<td>20% or above</td>
</tr>
<tr>
<td>Continuous</td>
<td>15% or under</td>
<td>16% to 25%</td>
<td>25% or above</td>
</tr>
<tr>
<td>Rework speed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single</td>
<td>18% or under</td>
<td>19% to 29%</td>
<td>30% or above</td>
</tr>
<tr>
<td>Continuous</td>
<td>23% or under</td>
<td>24% to 39%</td>
<td>40% or above</td>
</tr>
</tbody>
</table>

Table 2. % Improvement vs. BUVR Values

**Extension**

For extension, we conducted two analyses, simulated millions of iterations on a variety of scenarios, and
also attempted to solve the equations algebraically, all with the hope to gain a more comprehensive
understanding of the effect of various ranges of parameters on BUVR.

In Analysis 1, we applied the same percentage of improvement across all the three parameters
(productivity, defect rate, and rework speed) at the same time. In Additional Analysis 2, we changed % of
improvement in one parameter while setting the values of the other two parameters equal between solo
and pair.

Figure 4 is the graph from Analysis 1. The left panel shows BUVR for percentage improvements from
single in productivity (x%), rework speed (x%), and defect rate (y%). If the graph is above the horizontal
plane BUVR = 1, then pair is better than solo for those x and y values. The right panel is a contour plot for
BUVR = K where K = 1, 1.1, 1.2, 1.3, 1.4, 1.5, 2, 3, 4, 5, 6, 7, 8, 9, 10 from lower left to upper right. The Utah
value is marked at (x,y) = (73.9, 40).
Figure 4: Analysis 1

Figure 5 below are the graphs from Analysis 2.
Assuming the same defect rate and rework speed for solos and pairs, increasing productivity in pairs doesn’t seem to improve BUVR to a level where pairs will be better than soloists. In single delivery, BUVR ranges from 0.4 to 0.50 decreasing with a lower bound 0.4329, and in continuous delivery, BUVR is from 0.50 to 0.60 increasing with an upper bound 0.6012. As productivity increases, there is a higher BUVR in continuous delivery but a lower BUVR in single delivery. However in either case BUVR is below 1 meaning that solo is better than pair. See Figure 5a.

Assuming the same productivity and rework speed for solos and pairs, reduction in defect rate by pair programming has the potential to make pairs better than soloists. If pairs can reduce the defect rate by at least 47% (in single delivery) and 60% (in continuous delivery), then BUVR is greater than 1, meaning pairs are better than soloists. Otherwise, BUVR is less than 1, meaning soloists are better than pairs. See Figure 5b.
Figure 5c demonstrates the same trend on rework speed. If pairs can increase the rework speed by at least 91\% (in single delivery) and 151\% (in continuous delivery), pairs are better than soloists. Otherwise, solos are better than pairs.

Now a natural question arises: If parameters for pair are fluctuating around the results from previous studies, then what will be the behavior of BUVR’s? To answer this question, we performed 10 million iterations for each of the scenarios.

In Scenario 1, productivity \( \pi_p = \frac{2 ne}{x^2} \) where \( X \sim N(1.15, 0.1^2) \), the other two parameters used data from previous studies. This simulation shows that the distribution of BUVR_\(s\) is skewed to the left due to an upper bound of BUVR_\(s\) (see Figure 3a) while BUVR_\(c\) has approximately symmetric and bell shaped distribution. Their standard deviations can be estimated using derivatives. In fact, if we take the derivative of BUVR_\(s\) with respect to \( x \) at 1.15, we get \(-0.06274382518\) which gives an estimation of the standard deviation of BUVR_\(s\) as \(0.06274382518 \times 0.1 = 0.006274382518\). Similarly, the derivative of BUVR_\(c\) at \( x=1.15 \) is \(0.2858386349\) which gives an estimation of the standard deviation of BUVR_\(c\) as \(0.2858386349 \times 0.1 = 0.02858386349\). Note that 0.1 is the standard deviation of \( X \).

\[
\begin{array}{|c|c|c|c|c|c|c|c|}
\hline
\text{Parameter} & \text{Min} & \text{Q1} & \text{Med} & \text{Mean} & \text{Q3} & \text{Max} & \text{St.Dev.} \\
\hline
\pi_p & 29.67 & 41.07 & 43.47 & 43.81 & 46.18 & 82.53 & 3.90126 \\
BUVR_\(s\) & 1.433 & 1.502 & 1.507 & 1.505 & 1.51 & 1.515 & 0.006558 \\
BUVR_\(c\) & 1.166 & 1.284 & 1.303 & 1.304 & 1.323 & 1.48 & 0.028654 \\
\hline
\end{array}
\]

Figure 6. Scenario 1
Our second scenario was on defect rate $\beta_p = y\beta_s$ where $Y \sim N(0.6, 0.1^2)$, the other two parameters used data from previous. If we take the derivative of BUVR with respect to $y$ at $y=0.6$, we get $2.631243876$ which gives an estimation of the standard deviation of BUVR as $2.631243876 \times 0.1 = 0.2631243876$. Similarly, the derivative of BUVR at $y=0.6$ is $1.623955855$ which gives an estimation of the standard deviation of BUVR as $1.623955855 \times 0.1 = 0.1623955855$. In addition, the distribution of BUVR is a little bit right skewed which can be explained by the fact that the derivative of BUVR with respect to $y$ is positive and increasing at $y=0.6$, i.e. they are more variable on the right hand side of 0.6 than the other side.

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>Q1</th>
<th>Med</th>
<th>Mean</th>
<th>Q3</th>
<th>Max</th>
<th>St.Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta_p$</td>
<td>0.000441</td>
<td>0.003115</td>
<td>0.00351</td>
<td>0.00351</td>
<td>0.003904</td>
<td>0.006561</td>
<td>0.000585</td>
</tr>
<tr>
<td>BUVR$_a$</td>
<td>0.7674</td>
<td>1.346</td>
<td>1.507</td>
<td>1.55</td>
<td>1.705</td>
<td>4.486</td>
<td>0.288618</td>
</tr>
<tr>
<td>BUVR$_c$</td>
<td>0.7897</td>
<td>1.202</td>
<td>1.303</td>
<td>1.324</td>
<td>1.423</td>
<td>3.761</td>
<td>0.173604</td>
</tr>
</tbody>
</table>

**Figure 7. Scenario 2**

The third one was on Rework speed $\rho_p = \frac{2\rho_s}{x}$ where $Z \sim N(1.15, 0.1^2)$, the other two parameters used data from previous. If we take the derivative of BUVR with respect to $z$ at $z=1.15$, we get $1.372871074$ which gives an estimation of the standard deviation of BUVR as $1.372871074 \times 0.1 = 0.1372871074$. Similarly, the derivative of BUVR at $z=1.15$ is $0.8473133737$ which gives an estimation of the standard deviation of BUVR as $0.8473133737 \times 0.1 = 0.08473133737$.\[\text{Twentieth Americas Conference on Information Systems, Savannah, 2014} \ 10\]
Figure 8. Scenario 3
In the fourth scenario, we applied Productivity $\pi_p = \frac{2\pi s}{x}$ where $X \sim N(1.15, 0.1^2)$, Defect rate $\beta_p = y\beta_s$ where $Y \sim N(0.6, 0.1^2)$, Rework speed $\rho_p = \frac{2\rho s}{z}$ where $Z \sim N(1.15, 0.1^2)$ at the same time.

As we can observe from the above simulations, a variation in parameters of pair may drop the value of BUVR down below 1. For standard deviations in the following range, we use 1 million random numbers for each normal distribution to estimate the probabilities of BUVR ≥ 1.

1) Productivity $\pi_p = x\%$ improvement from $\pi_s$; $X \sim N(73.9, \sigma_a)$, $5 \leq \sigma_a \leq 30$. Note that 73.9 is the value of Utah.

2) Defect rate $\beta_p = y\%$ improvement from $\beta_s$; $Y \sim N(40, \sigma_\beta)$, $5 \leq \sigma_\beta \leq 15$. Note that 40 is the value of Utah.

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>Q1</th>
<th>Med</th>
<th>Mean</th>
<th>Q3</th>
<th>Max</th>
<th>St.Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\pi_p$</td>
<td>30.37</td>
<td>41.07</td>
<td>43.48</td>
<td>43.82</td>
<td>46.19</td>
<td>78.89</td>
<td>3.901047</td>
</tr>
<tr>
<td>$\beta_p$</td>
<td>0.000444</td>
<td>0.003115</td>
<td>0.00351</td>
<td>0.00351</td>
<td>0.003905</td>
<td>0.0066</td>
<td>0.000585</td>
</tr>
<tr>
<td>$\rho_p$</td>
<td>0.03468</td>
<td>0.04868</td>
<td>0.05153</td>
<td>0.05193</td>
<td>0.05474</td>
<td>0.09053</td>
<td>0.004624</td>
</tr>
<tr>
<td>BUVR_s</td>
<td>0.6671</td>
<td>1.331</td>
<td>1.515</td>
<td>1.56</td>
<td>1.738</td>
<td>5.244</td>
<td>0.322276</td>
</tr>
<tr>
<td>BUVR_c</td>
<td>0.7054</td>
<td>1.192</td>
<td>1.309</td>
<td>1.33</td>
<td>1.445</td>
<td>4.049</td>
<td>0.197218</td>
</tr>
</tbody>
</table>

**Figure 9. Scenario 4**
3) Rework speed $\rho_p = z\%$ improvement from $\rho_s$; $Z \sim N(73.9, \sigma_p)$, $5 \leq \sigma_p \leq 30$. Note that 73.9 is the value of Utah. We assume $\sigma_p = \sigma_\pi$.

We can check that the probability $P(BUVR_s \geq 1)$ is greater than 0.9 for any standard deviations in the range above, while the probability $P(BUVR_c \geq 1)$ is bounded below by 0.8674. Note that $P(BUVR_s \geq 1)$ is always greater than or at least same to $P(BUVR_c \geq 1)$.

![Figure 10: Probability of BUVR](image)

Red: $P(BUVR_s \geq 1)$, Blue: $P(BUVR_c \geq 1)$

Our simulations suggest that when values data from previous are applied, the BUVR mean tends to be greater than 1 in both single and continuous models. However, what if some organizations cannot achieve the pair programming advantages as the ones reported in the previous studies? What will be the behavior of BUVR's? To answer this question, we repeated the above scenarios with the following changes: 1) pair has advantages in two parameters; 2) pair has an advantage in one parameter only; 3) pair does not have advantage in any of the three parameters.

In Change 1, there are three situations: a) pair has advantages in productivity and defect rate but not in rework speed; b) pair has advantages in productivity and rework speed but not in defect rate; c) pair has advantages in defect rate and rework speed but not in productivity. The following are the details.

In Situation a, we applied Productivity $\pi_p = x \pi_s$ where $X \sim N(1.739, 0.1^2)$, Defect rate $\beta_p = y\beta_s$ where $Y \sim N(0.6, 0.1^2)$, and Rework speed $\rho_p = z\rho_s$ where $Z \sim N(1, 0.1^2)$. In Situation b, we applied Productivity $\pi_p = x \pi_s$ where $X \sim N(1.739, 0.1^2)$, Rework speed $\rho_p = z\rho_s$ where $Z \sim N(1.739, 0.1^2)$, and Defect rate $\beta_p = y\beta_s$ where $Y \sim N(1, 0.1^2)$. In Situation c, we applied Defect rate $\beta_p = y\beta_s$ where $Y \sim N(0.6, 0.1^2)$, Rework speed $\rho_p = z\rho_s$ where $Z \sim N(1.739, 0.1^2)$, and Productivity $\pi_p = x \pi_s$ where $X \sim N(1, 0.1^2)$. Table 3 below is a summary of the above three simulations. The results suggest that for BUVR to be greater than 1, pairs need to have advantages on both defect rate and rework speed (Situation c). In other two situations, BUVR tends to be smaller than 1, meaning solos will be better than pairs.

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>Q1</th>
<th>Med</th>
<th>Mean</th>
<th>Q3</th>
<th>Max</th>
<th>St.Dev.</th>
</tr>
</thead>
</table>

Twentieth Americas Conference on Information Systems, Savannah, 2014
Table 3: Advantages in Two Parameters

In Change 2, pair has an advantage in one parameter only. We did the following three simulations: pair has an advantage in productivity only (Productivity $\pi = \pi_p$ where $X \sim N(1.739, 0.1^2)$, Defect rate $\beta = y\beta_s$ where $Y \sim N(1, 0.1^2)$, Rework speed $\rho = z\rho_s$ where $Z \sim N(1, 0.1^2)$); pair has an advantage in defect rate only (Defect rate $\beta = y\beta_s$ where $Y \sim N(0.6, 0.1^2)$, Productivity $\pi = x\pi_s$ where $X \sim N(1, 0.1^2)$, Rework speed $\rho = z\rho_s$ where $Z \sim N(1, 0.1^2)$); pair has an advantage in rework speed only (Rework speed $\rho = z\rho_s$ where $Z \sim N(1.739, 0.1^2)$, Productivity $\pi = x\pi_s$ where $X \sim N(1, 0.1^2)$, Defect rate $\beta = y\beta_s$ where $Y \sim N(1, 0.1^2)$). Table 4 is a summary of the simulations. The results suggest having an advantage in one parameter does not lead to a BUVR greater than 1.
Table 4. Advantage in One Parameter

It is not surprising to find out if pair does not have any advantage, then BUVR is around 0.5, which suggests solos are better than pair in terms of NPV. See Table 5. In this simulation, we applied Productivity $\pi_p = x \pi_s$, where $X \sim N(1, 0.1^2)$, Defect rate $\beta_p = y \beta_s$, where $Y \sim N(1, 0.1^2)$, and Rework speed $\rho_p = z \rho_s$, where $Z \sim N(1, 0.1^2)$.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Min</th>
<th>Q1</th>
<th>Med</th>
<th>Mean</th>
<th>Q3</th>
<th>Max</th>
<th>St.Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\pi_p$</td>
<td>11.56</td>
<td>23.31</td>
<td>25</td>
<td>25</td>
<td>26.68</td>
<td>37.48</td>
<td>2.499443</td>
</tr>
<tr>
<td>$\beta_p$</td>
<td>0.002791</td>
<td>0.005456</td>
<td>0.00585</td>
<td>0.00585</td>
<td>0.006245</td>
<td>0.008918</td>
<td>0.000585</td>
</tr>
<tr>
<td>$\rho_p$</td>
<td>0.03514</td>
<td>0.04953</td>
<td>0.05153</td>
<td>0.05153</td>
<td>0.05353</td>
<td>0.06909</td>
<td>0.008918</td>
</tr>
<tr>
<td>BUVR$_s$</td>
<td>0.5096</td>
<td>0.8355</td>
<td>0.9049</td>
<td>0.9133</td>
<td>0.9818</td>
<td>1.852</td>
<td>0.110812</td>
</tr>
<tr>
<td>BUVR$_c$</td>
<td>0.4903</td>
<td>0.7266</td>
<td>0.7712</td>
<td>0.7755</td>
<td>0.8197</td>
<td>1.339</td>
<td>0.070185</td>
</tr>
</tbody>
</table>

Table 5. No Advantages

We also attempted to algebraically solve the equations. In continuous delivery model, for $x\%$ improvement in the productivity, $y\%$ improvement in the defect rate, $z\%$ improvement in the rework speed, we can solve the inequality

$$BUVR_c \geq K$$

for $z$ as follows:

$$z \geq F(x, y) := 100 \cdot \left( \frac{2K\beta_p\pi_s}{\rho_s + \beta_s\pi_s} \left( \frac{1 + \frac{x}{100}}{1 + \frac{y}{100}} \right)^2 - 1 \right)$$

The left graph below shows the surface $z = F(x, y)$ for $0 \leq x \leq 400$, $0 \leq y \leq 100$ when $K=1$. If a point $(x, y, z)$ is above this surface and the horizontal plane $z = 0$, then BUVR is greater than 1 meaning that pair is better than solo.

In single delivery model, due to the complexity of the equation, $BUVR_s = 1$ cannot be solved algebraically, but an implicit plot of the solution can be obtained using software such as Maple. If a point $(x, y, z)$ is above the surface and the horizontal plane $z = 0$, then $BUVR_s$ is greater than 1.
Finally, we would like to suggest a modification to the model by replacing marginal costs as follows.

On page 303, the marginal cost was derived using \( mc_{\text{pre}} = \frac{E_{\text{E}_{\text{pre}}}}{\tau} \), but it is more reasonable to modify as

\[
mc_{\text{pre}} := \frac{E_{\text{E}_{\text{pre}}}}{\tau_{\text{pre}}},
\]

due to the fact that this cost only applies to the time period \([0, \tau_{\text{pre}}]\). It reduces to

\[
mc_{\text{pre}} = h_y NC_{\text{pre}}.
\]

Similarly, if we modify the definition of \( mc_{\text{post}} \) as

\[
mc_{\text{post}} := \frac{E(1-\varepsilon)_{\text{C}_{\text{post}}}}{\tau_{\text{post}}},
\]

then we get

\[
mc_{\text{post}} = h_y NC_{\text{post}}.
\]

Under the assumption \( C_{\text{pre}} = C_{\text{post}} = C, TDC_{s} \) on page 304 will be modified to

\[
TDC_{s} = \frac{h_y NC}{r} \left(1 - e^{-\frac{\tau_{\text{pre}}}{\tau_{\text{pre}}}}\right).
\]

It results in the following representations.

\[
NPV_s = V \omega e^{-\frac{\tau_{\text{pre}}}{\tau_{\text{pre}}}} - \frac{h_y N C}{r} \left(1 - e^{-\frac{\tau_{\text{pre}}}{\tau_{\text{pre}}}}\right)
\]

\[
BUV_s = \frac{h_y N C}{\omega r} \left(e^{\frac{\tau_{\text{pre}}}{\tau_{\text{pre}}}} - 1\right)
\]

Moreover, we can observe that
\[
\lim_{t \to 0} BUV_s = \lim_{\omega \to 0} BUV_s = \frac{NC}{\pi \epsilon}
\]

which is exactly \( BUV_c \). So the continuous delivery case can be interpreted as a limit case of single delivery as follows.

Since \( \tau = \frac{\omega}{\pi \epsilon \mathbf{h}_y} \) shows that \( \tau \) is a direct variation of \( \omega \), we can split the total work \( \omega \) into \( n \) equal parts \( \omega / n \) and the entire interval \( \tau \) can be divided into \( n \) consecutive subintervals with length \( \tau / n \). It means the first work \( \omega / n \) is completed at the time \( \tau / n \), and the next work \( \omega / n \) is completed at the time \( 2\tau / n \), and so on. A natural modification of Figure 2 (page 297) can be drawn below.

As a consequence, the \( NPV \) of the first work \( \omega / n \) determines the sign of the \( NPV \) of the total work \( \omega \). If \( n \) tends to infinity, then \( BUV_s \) converges to \( BUV_c \) for the continuous delivery of the paper. Some calculations show that \( BUV_s \) is always greater than \( BUV_c \), but the actual difference is very small (less than 0.0001).

**Limitations and Future Research**

In this paper, we reported the study we conducted based on the replication and extension of Erdogmus and Williams (2003). We demonstrated the fact that pair programming and solo programming both have their own economic advantages and those advantages are dependent of a variety of factors.

One major limitation of this paper is its narrow focus. We will address this limitation by creating a more comprehensive research paper that we hope to submit to a journal outlet. That comprehensive research paper will include other alternatives to examine the cost effectiveness of programming methods.

**References**


