Experimental Investigations of Programming Productivity Factors

Johnny Researcher and Suzie Important-Person

Abstract—This paper reports on a series of three experiments that explore factors that inhibit or contribute to coding productivity. We find strong empirical support that distractions and alcohol harm productivity but that design methods improve productivity (measured as lines of code per unit time). These findings should enable substantial opportunities for improvement by the software industry.

Index Terms—Empirical research, experiment, coding productivity.

1 Introduction

Since the software crisis was first identified almost 50 years ago [1] software engineers have been crying out to better understand the factors that enhance or hinder programmer productivity. For this reason we present the results from a series of experiments that provide useful insights and actionable findings for the practitioner community.

2 Related Work

As is customary here's a list of our work in order to boost our h-indices [2], [3], [4], [5], [6].

Some other researchers have also addressed the question of productivity. Their findings might be summarised as a productivity-enablers enhance productivity whilst hindering-factors have a negative impact. This situation motivates us to take the bold step of investigating exactly what these factors might be.

3 Theoretical Concerns

A major challenge is how to define coding productivity? An in depth review of the economics literature suggests that productivity, denoted p, might be modelled thus:

$$p = \frac{o}{i} \tag{1}$$

where o are units of output and i are units of input. Advanced algebraic manipulation[1] yields:

$$d = \frac{i}{p} \tag{2}$$

where d is delivery. Further extremely advanced manipulation allows us to compute delivery rate d^{-t} given as units of output per unit time.

$$d^{-t} = \frac{o}{m} \tag{3}$$

where m is the number of units of time length t to deliver o outputs.

As the astute reader will appreciate, thus far our analysis considers project productivity and delivery. Suppose the project comprises n individuals where $n > 1$? A major contribution of our research is to extend the above framework to handle such a situation. We believe this is likely to be of great value to practitioners. Taking Eqn. 1 we can re-express this:

$$p = \frac{\sum_{j=1}^{n} o_j}{\sum_{j=1}^{n} i_j} \tag{4}$$

where n is the number of programmers and j is the j^{th} programmer where $0 < j < n$.

4 Method

We conducted a series of independent experiments to investigate a set of factors that might impact coding productivity.

4.1 Experiment 1: The Impact of Distractions

Following conjectures arising from our previous studies such as [5] we pose the research question: do distractions impact the ability of a programmer to develop code? We simulated a distraction by means of loud and unpleasant music[2].

Our coders for this experiment were 20 newly enrolled undergraduate students. They were randomly allocated to loud noise or the control which was a less loud noise (since they were further from the source of the distraction, a single speaker) depending upon where they were seated themselves upon arrival in the lab. A simple test of the level of distraction suggested that 18 coders could be considered to receive the distraction treatment and 2 coders became the control.

The students were then asked to write code for a simple payroll system. In order to facilitate productivity they were allowed to make their own choice of programming language. The response variable was lines of code (LOC). The

- *J. Researcher is with the School of Hard Knocks, University of Life, UK*
- *S. Important-Person is Dean of Science, University of Life, UK*

Manuscript received April 7, 2016; accepted April 7, 2016; corrected June 8, 2016.

1. The full proof is available from the corresponding author.

2. For the purposes of our experiment this was deemed to be anything by Kate Bush.

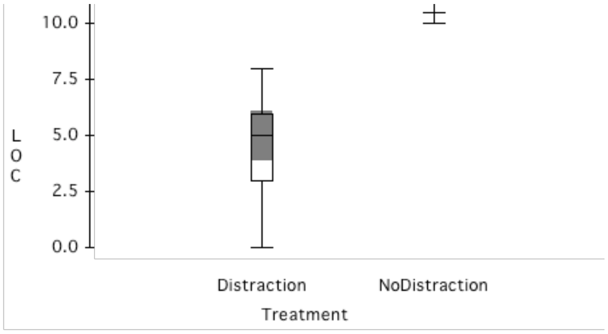

Fig. 1. Boxplots showing the negative impact of distractions on coding productivity

highly significant difference between treatments is shown in Figure 4.1.

This is confirmed by a 2-sample t-test (Difference Between Means = -6.11, t-Statistic = -8.379 with 4 df). Therefore we reject H_0 at $\alpha = 0.05, p = 0.0009$). This shows that the difference between those closer to the loud noise and those further away was statistically significant.

4.2 Experiment 2: The Impact of Alcohol

In the second of our series experiments we explored the impact of alcohol upon productivity. In this study we used a crossover design as we only 10 had participants. These were students from Experiment 1. Note due to the intervention of the University of Life Ethics Committee we were obliged to offer the students the option of not participating. Unfortunately 50% of the initial group took this option on the grounds that they were teetotal.

Using the crossover design, half the group were administered alcohol and the others were offered water. Then they performed a 30 minute coding task. Then the groups each received the other treatment and this was immediately followed by a second 30 minute coding task. This way we could compare the within subject differences for productivity in terms of LOC over the 30 minute period. Again, choice of language was unconstrained.

A paired t-Test of μ(Alcohol-Water) shows a mean of Paired Differences = -1.10; t-Statistic = -0.9052 with 9 df. Although this technically is a failure to reject H_0 at $\alpha = 0.05; p = 0.3889$ it still shows an almost 2 in 3 chance of significance. Note, the overall mean productivity is less for the alcohol condition. We therefore consider this experiment a success.

The raw data are given in Table 1. Note the crossover is indicated by the order, either alcohol-water (AW) or water-alcohol (WA).

4.3 Experiment 3: The Impact of Design Methods

In the third, and final, experiment we explored the impact of using a bespoke software design method known as the Researcher Design Method (RDM) after its inventor. This was compared to a control where the participants used no design method. Unfortunately, once again on the insistence of the Ethics Committee we were obliged to allow a further 5 students to withdraw, despite carefully explaining to them how this might harm their education prospects. Consequently, for the final study $n = 5$. Since RDM is a complex design method we selected the three participants who were most productive from previous experiment for this condition whilst the remaining two were assigned to the control condition.

TABLE 1
Raw Data for Impact of Alcohol and Water on Productivity in LOC

Subject	Order	Alcohol	Water
1	AW	5	3
2	AW	6	3
3	AW	5	4
4	AW	4	4
5	AW	7	3
6	WA	7	8
7	WA	7	12
8	WA	8	15
9	WA	5	11
10	WA	7	9
	mean	6.1	7.2

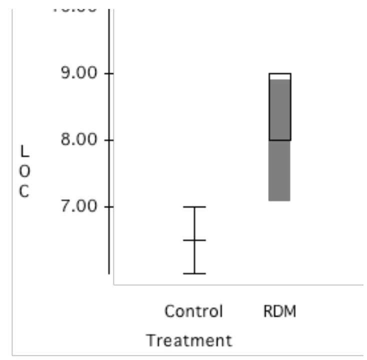

Fig. 2. Boxplots showing the positive impact of RDM on coding productivity

Figure 2 compares the coding productivity for the design method and the control. Immediately it is clear that RDM leads to superior productivity. This is confirmed by a 2-sample t-test where the difference between means = +1.83; t-Statistic = -3.051 with 1 df. Therefore we reject H_0 at $\alpha = 0.10; p = 0.0988$. This is strongly suggestive that the Researcher Design Method is a major contributor to coding productivity, again measured in LOC per unit time.

5 Conclusion

Our three experiments provide unequivocal proof that distractions and alcohol inhibit coding performance. In contrast the little known, but important, Researcher Design Method enhances productivity. We argue these are fundamental findings for practitioners and should have a substantial and positive impact upon the software industry and solving the “Software Crisis”.

Acknowledgment

This work is supported by Hard Working Tax Payers! Martin Shepperd is the evil mastermind behind this research travesty. Email him at martin.shepperd@brunel.ac.uk.

REFERENCES

[1] Naur, P. and Randall, B., *Software Engineering: Report on a Conference by the NATO Science Committee*, NATO Scientific Affairs Division, Brussels, 1968.

[2] Researcher, J., "My first pony", *Acta Infantilia*, 47(5), pp203–9, 2009.

[3] Researcher, J., "Schniedelgrüber transforms", *11th Undergraduate Workshop*, Neasden, UK, 2010.

[4] Researcher, J. et al. "Non-linearity in Stickman Space", *The Intl. J. of Egregious Results*, 23(11) pp830–1, 2012.

[5] Researcher, J. and Important-Person, S., "Misconjectures on the Riemann Hypothesis", *2nd International Pay to Publish Conference*, Bali, 2014.

[6] Researcher, J. and Important-Person, S., "An Empirical Study of How X47 is a Strong Determinant of Y", *IEEE Trans. on Sophistry*, 11(4) pp1, 2015.