

Ringkasan Paper

Nama : Agung Firmansyah (120500007X), Muhammad Ilman Akbar (1205000622)
 Kelompok : 316

Judul Paper	<i>Are Two Heads Better Than One for Software Development The Productivity Paradox of Pair Programming</i>
Penulis	VenuGopal Balijepally, RadhaKanta Mahapatra, Sridhar Nerur, Kenneth H. Price
Sumber	MIS Quarterly Vol. 33 No. 1, pp. 91-118/March 2009
Tahun	2009

Pendahuluan

Saat ini, *Extreme Programming* (XP) sedang naik daun dalam dunia pengembangan perangkat lunak. *Pair Programming*, memasang 2 orang programmer untuk bekerja secara kelompok (berpasangan), merupakan inti dari XP itu sendiri. Ekspektasi yang muncul dari penerapan *Pair Programming* ini adalah *Pair Programming* akan memberikan hasil yang lebih baik dari *Individual programming* dalam hal kecepatan pengembangan perangkat lunak, kualitas, dan lain-lain.

Penelitian ini bertujuan untuk mencari tahu tingkat keefektifan *Pair Programming* dibandingkan *individual programming* dengan jumlah programmer yang sama. Peneliti juga memperluas cakupan penelitian dengan mencari tahu bagaimana respon (kenyamanan dan kepercayaan diri) para programmer dengan pair dan para programmer individu. Sebenarnya, penelitian-penelitian sejenis sudah pernah dilakukan. Akan tetapi penelitian-penelitian tersebut memberikan hasil yang bervariasi.

Tabel 1. Hasil Penelitian tentang Efektivitas *Pair Programming* (Balijepally, V., dkk. 2009)

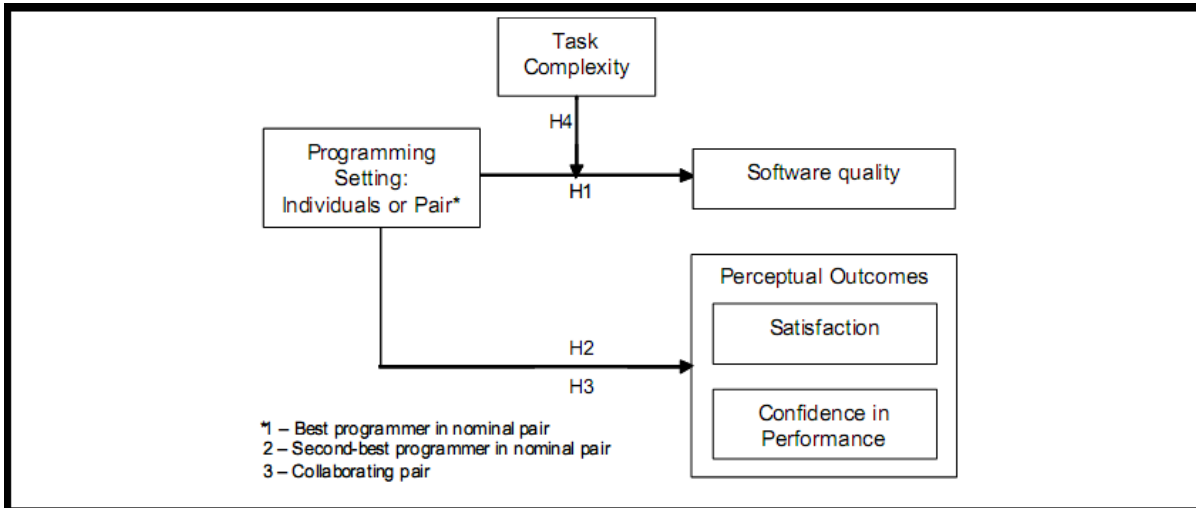
Positive Findings	Negative/Neutral Findings
Software Quality <ul style="list-style-type: none"> Enhanced quality (Nosek 1998; Williams 2000) Enhanced quality in large software projects (Cao et al. 2004) Enhanced quality of software designs (Canfora et al. 2007; Williams 2000) Reduced defects (Phongpaibul and Boehm 2006) 	<ul style="list-style-type: none"> No differences in the package level quality metrics (Madeyski 2006) No differences in proportion of correct solutions (Arisholm et al. 2007) No differences in percentage of correctly implemented test cases in a given time (Heiberg et al. 2003) Higher defects in pair solutions (Vanhanen and Lassenius 2005) No differences in errors uncovered in acceptance tests (Nawrocki and Wojciechowski 2001) No differences in defect density (Hulkko and Abrahamsson 2005) No effect on the thoroughness or effectiveness of testing when using test driven development (Madeyski 2007)
Development Effort <ul style="list-style-type: none"> Effort comparable if required to produce programs of similar level of correctness (Müller 2005, 2006) No productivity differences (Hulkko and Abrahamsson 2005) Reduced development effort in student 	<ul style="list-style-type: none"> Required more effort to perform the task correctly (Arisholm et al. 2007) Pair lower in productivity (sum of implemented use cases divided by effort) (Vanhanen and Lassenius 2005) Required more development effort in professional programmer group

<p>programmer group (Phongpaibul and Boehm 2006)</p> <ul style="list-style-type: none"> • Reduced effort (time spent) (Canfora et al. 2005) 	<p>(Phongpaibul and Boehm 2006)</p> <ul style="list-style-type: none"> • No differences in time to completion (Arisholm et al. 2007; Nawrocki and Wojciechowski 2001; Rostaher and Hericko 2002) • <i>Pair Programming</i> as in XP less efficient (Nawrocki and Wojciechowski 2001)
<p>Task Complexity</p> <ul style="list-style-type: none"> • Reduced time to completion on low complexity tasks for experienced programmers (Arisholm et al. 2007) • Flexible <i>Pair Programming</i> beneficial for large scale projects with complex software development (Cao et al. 2004) • Increased correctness of solution on complex tasks for junior programmers (Arisholm et al. 2007) • Most useful for learning and complex tasks (Hulkko and Abrahamsson 2005) 	<ul style="list-style-type: none"> • No reduction in time taken to solve more complex tasks correctly (Arisholm et al. 2007) • No effect of task complexity on the effort between pairs and individuals (Vanhanen and Lassenius 2005)
<p>Others</p> <ul style="list-style-type: none"> • Higher adherence to coding standards due to peer pressure (Cockburn and Williams 2001) • Higher enjoyment with problem solving process (Nosek 1998; Vanhanen and Lassenius 2005; Williams 2000) • Higher confidence in solutions (Nosek 1998; Williams et al. 2000) • Enhanced learning due to cross training (Williams 2000; Williams et al. 2000) • Improved team building (Williams 2000) • Collaborative and supportive environment fostered (Cao et al. 2004) • Higher problem solving skills (Williams 2000) • Enhanced predictability of development time and program size (Nawrocki and Wojciechowski 2001) 	<ul style="list-style-type: none"> • Lower adherence to coding standards but has higher comment ratio (Hulkko and Abrahamsson 2005)

Model Penelitian

Penelitian ini membandingkan *Pair Programming* dengan gabungan individual programming. Berikut ini batasan penelitiannya:

1. Ada dua jenis programmer: lapis 1(better) dan lapis 2.
2. Programmer akan dipasangkan (paired) secara acak.
3. Pengelompokan programmer individu dilakukan secara acak dalam lapis yang sama.
4. Ada 3 pengelompokan programmer: paired, lapis 1, dan lapis 2.
5. Tugas pemrograman adalah *maintenance* perangkat lunak dengan dua tingkat kesulitan: tinggi dan rendah.
6. Fokus di perbandingan antara pair programmer dan gabungan individual programmer (kualitas produk dan tingkat kesulitan tugas).
7. Tambahan: menghitung respon para programmer terhadap pekerjaan mereka (kenyamanan dan kepercayaan diri).



Gambar 1. Model Penelitian (Balijepally, V., dkk. 2009)

Table 3. Demographic Details of Experimental Subjects		
Demographic Variable	Number of Subjects	Percentage
Gender		
Male	89	74.2
Female	31	25.8
Subject status		
Undergraduate	99	82.5
Graduate	21	17.5
Subject Citizenship		
U.S.	81	69.2
Other countries	36	30.8
Programming experience		
0 – 1 year	44	37.9
1 – 2 years	30	25.9
2 – 4 years	20	17.2
> 4 years	22	19.0
Java experience		
0 – 1 year	85	73.3
1 – 2 years	24	20.7
2 – 4 years	6	5.1
> 4 years	1	0.9

Gambar 2. Rata-rata Margin dari "Software Quality" (Balijepally, V., dkk. 2009)

Hipotesis

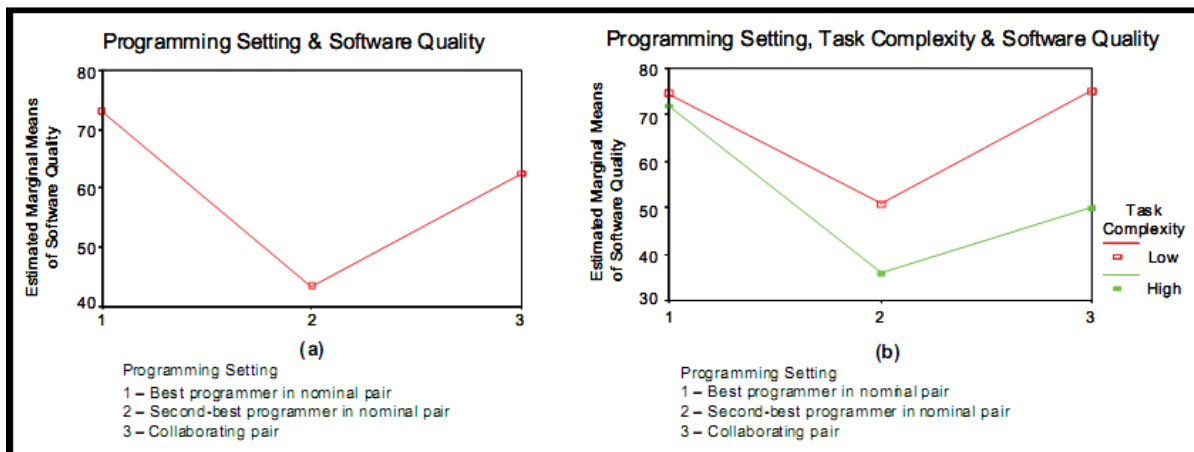
Hipotesis 1 : Kualitas produk *Pair Programming* lebih baik daripada gabungan individual programmer lapis 2.

Hipotesis 2A : Kenyamanan dalam bekerja para paired programmer lebih tinggi daripada gabungan individual programmer lapis 1.

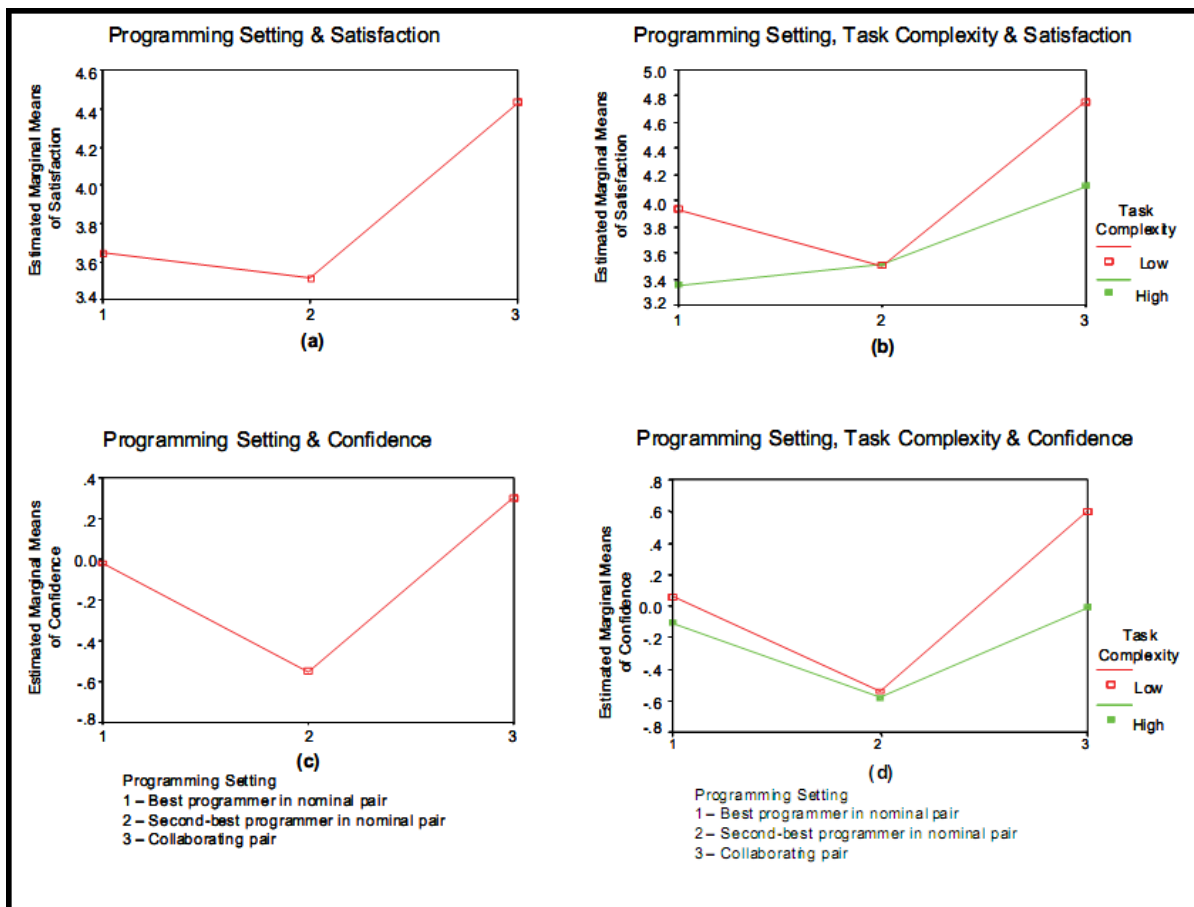
Hipotesis 2B : Kenyamanan dalam bekerja para paired programmer lebih tinggi daripada gabungan individual programmer lapis 2.
 Hipotesis 3A : Kepercayaan diri dalam bekerja para paired programmer lebih tinggi daripada gabungan individual programmer lapis 1.
 Hipotesis 3B : Kepercayaan diri dalam bekerja para paired programmer lebih tinggi daripada gabungan individual programmer lapis 2.

Hasil Penelitian

Hasil penelitian dapat dilihat melalui grafik-grafik di bawah ini.



Gambar 3. Rata-rata Margin dari "Software Quality" (Balijepally, V., dkk. 2009)



Gambar 4. Rata-rata Margin dari "Satisfaction" dan "Confidence in Performance" (Balijepally, V., dkk. 2009)

Kesimpulan

Hasil penelitian menunjukkan bahwa secara umum *Pair Programming* lebih baik daripada *individual programming*. Selain itu penelitian kali ini memberikan kontribusi pada bidang IS, meliputi:

1. Memperkenalkan penelitian perbandingan gabungan individu dengan kelompok.
2. Menggunakan pendekatan dari banyak sisi: kenyamanan dalam bekerja, kepercayaan diri, dan kualitas produk.
3. Membuktikan bahwa tingkat kesulitan tugas mempengaruhi keefektifan pengerjaan tugas oleh programmer individu.

Penelitian Selanjutnya

Penelitian yang dilakukan kali ini terbatas pada tugas *maintenance* perangkat lunak. Tugas tersebut cenderung membutuhkan lebih sedikit kreativitas. Penulis menyarankan agar penelitian selanjutnya juga melingkupi tahap-tahap lain dari proses pengembangan perangkat lunak seperti desain perangkat lunak, implementasi, dan testing.

Daftar Pustaka

Balijepally, V., dkk. 2009. "Are Two Heads Better Than One for Software Development The Productivity Paradox of Pair Programming," MIS Quarterly Vol. 33 No. 1, pp. 91-118/March 2009

Komentar Penulis

Ada beberapa komentar dari penulis ringkasan:

1. Paper ini sangat mudah dimengerti.
2. Peneliti memaparkan tahap demi tahap proses penelitiannya sehingga memudahkan pembaca memahami alur penelitian.
3. Peneliti telah memaparkan keseluruhan paper dalam abstraknya. Hal ini memudahkan studi literatur bagi para pembaca.
4. Hasil riset yang ditampilkan sangat mudah dimengerti. Meskipun menggunakan perhitungan statistika yang rumit, peneliti memaparkan hasilnya dengan grafik yang mudah dimengerti.