Θεσσαλονίκη 2011

ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ

Βιβλιογραφική έρευνα στις εμπειρικές μελέτες του Προγραμματισμού ανά ζεύγη

Του φοιτητή Δούκα Κλεάνθη

ΑΜ: 04/2630

Επιβλέπων Καθηγητής: Σφέτσος Παναγιώτης

Abstract:

[Ο προγραμματισμός ανά ζεύγη είναι μια από τις πρωταρχικές και θεμελιώδες τεχνικές των ευέλικτων μεθόδων. Τα αποτελέσματα που παρουσιάζουν οι ερευνητές ορισμένες φορές είναι αμφισβητούμενα και αμφιλεγόμενα. Μια μερίδα επιστημόνων πιστεύει πως ο προγραμματισμός ανά ζεύγη προσδίδει οφέλη κατά την διαδικασία ανάπτυξης λογισμικού μια άλλη πιστεύει πως όχι. Η συγκεκριμένη βιβλιογραφική έρευνα συγκέντρωσε γνώση και θα προσπαθήσει να δώσει απαντήσεις σε ορισμένα ερωτήματα. Ποια είναι η απήχηση του προγραμματισμού ανά ζεύγη στην διδασκαλία των φοιτητών, ποιες είναι οι επιπτώσεις στην ανάπτυξη του λογισμικού και ποιος είναι ο αντίκτυπος του ανθρώπινου παράγοντα στην διαδικασία.]

1. Εισαγωγή

Ο προγραμματισμός ανά ζεύγη είναι μια συνεργασιακή προσέγγιση ανάπτυξης λογισμικού με τους Williams και Kesler[1] να την ορίζουν ως εξής:

Προγραμματισμός ανά ζεύγη είναι το στυλ προγραμματισμού στο οποίο δυο προγραμματιστές δουλεύουν δίπλα δίπλα σε έναν υπολογιστή, συνεργάζονται συνεχώς πάνω στο ίδιο σχέδιο(program design), αλγόριθμο, κώδικα ή τέστ. Ένα μέλος του ζεύγους, ο οποίος ονομάζεται οδηγός ή Driver, πληκτρολογεί στον υπολογιστή ή γράφει σχέδια προγράμματος. Το άλλο μέλος του ζευγαριού, ο οποίος ονομάζεται καθοδηγητής ή Navigator, είναι υπεύθυνο για πολλές εργασίες, μια από αυτές είναι να παρατηρεί την εργασία του οδηγού, να βρίσκει τυχόν λάθη στην τακτική και την στρατηγική. Λάθη τακτικής είναι τα λάθη συντακτικού, τυπογραφικά σφάλματα, κάλεσμα λάθος μεθόδων κτλ. Λάθη στρατηγικής προκύπτουν όταν ο οδηγός κατευθύνεται σε λάθος μονοπάτι, δηλαδή όταν η εφαρμογή υλοποίησης δεν θα εκπληρώσει αυτό που πρέπει να εκπληρώσει. Ο καθοδηγητής είναι αυτός που σκέφτεται μακροπρόθεσμα και καθορίζει την στρατηγική.

Οι δυο προγραμματιστές είναι σαν ένα ενοποιημένο πανέξυπνο πλάσμα το οποίο δουλεύει με ένα μυαλό και είναι υπεύθυνο για την οποιαδήποτε πτυχή του αναπτυσσόμενου τεχνουργήματος. Και οι δύο είναι ισότιμοι, ενεργοί συμμετέχοντες στην διαδικασία όλη την ώρα και μοιράζονται την κυριότητα του δημιουργήματος, είτε πρόκειται για μια μικρή εργασία ενός πρωινού είτε πρόκειται για ένα ολόκληρο προτζεκτ [2].

Έχουν περάσει αρκετά χρόνια από το 2001, τότε που έγινε η ιδρυτική διακήρυξη των ευέλικτων μεθόδων (Agile Methods Manifesto)[3], αλλά ακόμη και τώρα γίνονται συζητήσεις και διαμάχες υπέρ και κατά των μεθόδων αυτών. Μια θεμελιώδες τεχνική των ευέλικτων μεθόδων , ορισμένοι θα την παρουσίαζαν ως επιτακτική, είναι αυτή του προγραμματισμού ανά ζεύγη. Θεωρείται έτσι γιατί μπορεί να χρησιμοποιηθεί συνδυαστικά με άλλες τεχνικές των ευέλικτων μεθόδων όπως Extreme Programming, Feature Driven Development, Scrum, Lean Software Development, Crystal και Dynamic Systems Development Method.

Η συγκεκριμένη βιβλιογραφική μελέτη χωρίζεται σε 8 στάδια. Μετά την εισαγωγή ακολουθεί μια σύντομη περιγραφή της τεχνικής του προγραμματισμού ανά ζεύγη και ακολουθεί η μεθοδολογία της έρευνας. Στο επόμενο στάδιο γίνεται μια σύντομη περιγραφή των συμπεριλαμβανομένων μελετών που χρησιμοποίησε η έρευνα. Ακολουθούν τα συμπεράσματα με τους περιορισμοί και τέλος οι παραπομπές της έρευνας.

2. Θεωρία

Στον προγραμματισμό ανά ζεύγη δυο προγραμματιστές δουλεύουν μαζί σε έναν υπολογιστή για την δημιουργία ενός τεχνουργήματος(κώδικας, σχέδιο, αλγόριθμος). Ο ένας από αυτούς ονομάζεται οδηγός και είναι αυτός που οδηγεί του πληκτρολόγιο, το ποντίκι, το μολύβι και είναι υπεύθυνος για την συγγραφή κώδικα και την δημιουργία σχεδίων. Ο δεύτερος προγραμματιστής που ονομάζεται καθοδηγητής είναι αυτός που ενεργά εποπτεύει την δουλειά του οδηγού. Παρατηρεί τυχών συντακτικά λάθη, ανατρέχει σε πηγές για πληροφορίες, σκέφτεται εναλλακτικούς τρόπους επίλυσης, καθορίζει την στρατηγική και προτείνει λύσεις.

Πολλές φορές ο προγραμματισμός ανά ζεύγη δεν είναι δυνατόν να πραγματοποιηθεί στον ίδιο χώρο. Έτσι με την βοήθεια οπτικοακουστικών μέσων, προγραμμάτων διαμοιρασμού επιφάνειας εργασίας αλλά και ειδικών προγραμμάτων ανεπτυγμένων για τον σκοπό αυτό μπορούμε να επιτύχουμε αυτή την απομακρυσμένη σύζευξη. Όταν οι προγραμματιστές από χωροταξική άποψη δεν βρίσκονται στην ίδια τοποθεσία τότε έχουμε Απομακρυσμένο Προγραμματισμό Ανά Ζεύγη η αλλιώς Distributed Pair Programming (DPP).

Ορισμένες παρατηρήσεις για την τεχνική όπως περιγράφονται στην διαδικτυακή τοποθεσία http://www.wikipedia.org/ :

- Ποιότητα σχεδίασης: Ο κώδικας ελέγχεται από 2 προγραμματιστές όπου ο ένας είναι ειδικά επιφορτισμένος να παρατηρεί τυχών λάθη καθώς ο άλλος γράφει κώδικα
- Μείωση κόστους ανάπτυξης: Λιγότερα λάθη σημαίνει χαμηλότερο κόστος, ειδικά όταν εντοπιστούν στα αρχικά στάδια που είναι πιο εύκολο να διορθωθούν είναι λιγότερο δαπανηρά.
- Αυξημένη αυτοπεποίθηση: Ορισμένοι προγραμματιστές αναφέρουν ικανοποίηση όταν εργάζονται σε ζεύγη.
- Αντιμετώπιση δύσκολων προβλημάτων: Τα προβλήματα λύνονται γρηγορότερα και με μεγαλύτερη ευκολία καθώς οι προγραμματιστές δουλεύουν και αναζητούν την λύση σε ζεύγη.
- Επιμόρφωση και εκπαίδευση: Επειδή οι προγραμματιστές δουλεύουν σε ζεύγη η γνώση είναι πολύ εύκολο να μοιραστεί. Ένας αρχάριος προγραμματιστής μπορεί να μάθει από έναν πιο έμπειρο πολυπλοκότερες τεχνικές και μεθόδους.
- Ελαχιστοποίηση ρίσκου μάνατζμεντ: Εάν ένας προγραμματιστής αποχωρήσει από την ομάδα τότε οι υπόλοιποι μπορούν εύκολα να προφτάσουν καθώς όλοι μοιράζονται εργασία πάνω στον ίδιο κώδικα.
- Αυξημένη πειθαρχία: Όταν οι προγραμματιστές δουλεύουν σε ζευγάρια είναι λιγότερο πιθανό να αφαιρεθούν και να ασχοληθούν με κάτι άλλο.
- Λιγότερες «διακοπές»: Όταν οι προγραμματιστές δουλεύουν σε ζεύγη δεν διακόπτονται τόσο εύκολα από άλλους ανθρώπους σε σχέση με το αν δούλευαν μόνοι.
- Μειωμένο ρίσκο RSI: Το ρίσκο της φυσικής κούρασης των προγραμματιστών που δουλεύουν αρκετές ώρες μειώνεται καθώς με την εναλλαγή θέσεων ο ένας ξεκουράζεται καθώς ο άλλος συνεχίζει το γράψιμο.
- Προτιμήσεις στην εργασία: Ορισμένοι άνθρωποι δεν μπορούν να εργαστούν με άλλους ανθρώπους προτιμούν να εργάζονται μόνοι.
- Αίσθηση φόβου: Ένας λιγότερο ικανός προγραμματιστής μπορεί να νιώθει την αυτοπεποίθησή του να μειώνεται καθώς εργάζεται με έναν αρκετά ικανό προγραμματιστή και έτσι να μην συνεισφέρει αρκετά στον κώδικα και στο ζεύγος.
- Διδακτική: Οι ικανοί προγραμματιστές μπορεί να βρίσκουν αρκετά βαρετό να διδάσκουν νέες τεχνικές σε αρχάριους προγραμματιστές.
- Συγκρούσεις προσωπικότητας: Τα ζεύγη ορισμένες φορές έχουν αρκετές δυσκολίες στην συνεργασία τους και μπορεί να γίνουν λιγότερο παραγωγικά απ' ότι όταν δούλευαν μόνοι.
- Ωράρια: Είναι αρκετά δύσκολο να βρεθούν κοινά χρονοδιαγράμματα έτσι ώστε να συναντηθεί το ζεύγος και να δουλέψει μαζί.

3. Review Method

Η συστηματική βιβλιογραφική έρευνα(SLR-Systematic Literature Review) αναφέρεται στην διαδικασία εύρεσης, αξιολόγησης και ερμηνείας όλης της διαθέσιμης βιβλιογραφίας ώστε να μπορέσουν να βρεθούν απαντήσεις για συγκεκριμένες ερευνητικές ερωτήσεις. Η συγκεκριμένη βιβλιογραφική έρευνα περαιώθηκε σε 3 φάσεις: Σχεδιασμός, διεξαγωγή και συγγραφή[4][11]. Κάθε φάση αποτελείται από περισσότερα απλά βήματα.

Φάση σχεδιασμού:

- Ορισμός ερευνητικών ερωτήσεων
- Ανάπτυξη μεθοδολογίας έρευνας
- Επικύρωση μεθοδολογίας έρευνας

Φάση διεξαγωγής:

- Εντοπισμός σχετικής βιβλιογραφίας
- Επιλογή πρωτογενών μελετών
- Αξιολόγηση βάση κριτηρίων ποιότητας
- Εξαγωγή δεδομένων
- Σύνθεση δεδομένων

Φάση συγγραφής:

- Συγγραφή της έρευνας
- Επικύρωση της έρευνας

Σύμφωνα με [5][6][11], ο σχεδιασμός μιας έρευνας περιγράφεται από τα εξής παρακάτω βήματα

- Ορισμός ερευνητικών ερωτήσεων
- Ορισμός τακτικής αναζήτησης
- Ορισμός κριτηρίων συμπερίληψης και απόρριψης
- Ορισμός στοιχείων ποιότητας
- Ορισμός μεθόδων συλλογής δεδομένων
- Σύνθεση δεδομένων

3.1 Ορισμός ερευνητικών ερωτήσεων

Η συγκεκριμένη βιβλιογραφική μελέτη θα ερευνήσει την απήχηση που έχει ο προγραμματισμός ανά ζεύγη στην διδασκαλία φοιτητών. Πως το αντιλαμβάνονται οι ίδιοι οι φοιτητές αλλά και πως οι καθηγητές μπορούν να επωφεληθούν από την διαδικασία. Σε ένα δεύτερο επίπεδο η έρευνα θα ασχοληθεί με την επίπτωση που έχει ο προγραμματισμός ανά ζεύγη στην ανάπτυξη του λογισμικού και ποιος είναι ο αντίκτυπος που έχει ο ανθρώπινος παράγοντας στην διαδικασία. Συγκεκριμένα οι κύριες ερωτήσεις τις έρευνας είναι:

- [Q1]Ποιες είναι οι επιπτώσεις που έχει ο προγραμματισμός ανά ζεύγη όταν αυτός χρησιμοποιείται σαν εκπαιδευτικό εργαλείο;
- [Q2]Ποιες είναι οι επιπτώσεις που έχει ο προγραμματισμός ανά ζεύγη στην παραγωγικότητα των δημιουργών;
- [Q3]Ποιες είναι οι επιπτώσεις που έχει ο προγραμματισμός ανά ζεύγη στον χρόνο ανάπτυξης του προγράμματος;
- [Q4]Ποιες είναι οι επιπτώσεις που έχει ο προγραμματισμός ανά ζεύγη στην ποιότητα του αναπτυσσόμενου προγράμματος;
- [Q5]Πως ο προγραμματισμός ανά ζεύγη επηρεάζεται από την προσωπικότητα των δημιουργών;

3.2 Ορισμός τακτικής αναζήτησης

Ο εντοπισμός της σχετικής βιβλιογραφίας που θα χρησιμοποιηθεί στην συγκεκριμένη μελέτη θα βρεθεί μέσω δημοσιεύσεων που έχουν αναρτηθεί σε αναγνωρισμένα Journal και Conferences σχετικά με την «μηχανική στο λογισμικό» έως τον Φεβρουάριο του 2011. Τα Journals επιλέχθηκαν γιατί είναι γνωστό ότι εμπεριέχουν είτε εμπειρικές μελέτες, είτε surveys οι οποίες χρησιμοποιήθηκαν σαν πηγές για διάφορες συστηματικές έρευνες(SLR) σχετιζόμενες με την μηχανική λογισμικού. Το πρώτο μέλημα είναι η δημιουργία του κατάλληλου string αναζήτησης το οποίο θα χρησιμοποιηθεί στις διαδικτυακές ψηφιακές βιβλιοθήκες. Οι όροι αναζήτησης που χρησιμοποιήθηκαν ήταν οι εξής: *Pair programming, experiment, empirical, survey, evaluation, case study, field study, review, measurement.* Με την βοήθεια των δυαδικών τελεστών AND και OR και των παραπάνω όρων αναζήτησης δημιουργήθηκε το παρακάτω String:

(*Pair programming AND experiment)OR(Pair programming AND empirical)OR(Pair programming AND survey)OR(Pair programming AND evaluation)OR(Pair programming AND case study)OR(Pair programming AND field study)OR(Pair programming AND review)OR(Pair programming AND measurement)*

Αυτό το string αναζήτησης χρησιμοποιήθηκε στις πιο κάτω διαδικτυακές βιβλιοθήκες έτσι ώστε να επιστραφούν υποψήφιες μελέτες για την συγκεκριμένη βιβλιογραφική έρευνα. Όπου στάθηκε δυνατό χρησιμοποιήθηκαν και τα φίλτρα αναζήτησης της μηχανής για να περιοριστούν τα αποτελέσματα τα οποία είναι σχετικά μόνο με την επιστήμη των υπολογιστών(Πχ. SpringerLink select filter “computer engineer”).

- ACM Digital library
- IEEExplore
- ScienceDirect
- SpringerLink
- Wiley InterScience
- CiteseerX Library

Σε ορισμένες ηλεκτρονικές βιβλιοθήκες ήταν αδύνατο να γίνει χρήση δυαδικών τελεστών και έτσι έγινε αναζήτηση της κάθε φράσης ξεχωριστά.

3.3 Ορισμός κριτηρίων συμπερίληψης και απόρριψης

Έπειτα από την αναζήτηση στις ηλεκτρονικές βιβλιοθήκες οι επιστρεφόμενες μελέτες αξιολογήθηκαν σε πρώτη φάση βάση του τίτλου και των abstract τους. Σε δεύτερη φάση αξιολογήθηκαν μέσα από το περιεχόμενό τους. Μελέτες οι οποίες δεν είχαν εμπειρικά δεδομένα που θα μπορούσαν να τις υποστηρίξουν απορρίφθηκαν από την επιλογή. Όσες μελέτες έγιναν με την μέθοδο της προσομοίωσης, μελέτες που ανέλυαν οικονομικά και μαθηματικά μοντέλα, άλλες που περιέγραφαν τον προγραμματισμό ανά ζεύγη σε θεωρητικό επίπεδο όλες εξαιρέθηκαν από την επιλογή. Επιλέχθηκαν μελέτες όπου το κύριο αντικείμενό τους ήταν ο προγραμματισμός ανά ζεύγη αφού πρώτα διαχωρίστηκαν σε πειράματα, surveys,case/field studies και reviews και είχαν απαραιτήτως εμπειρικά δεδομένα για να μπορέσουν να τις υποστηρίξουν. Μελέτες ανεξαρτήτως αν έγιναν από σπουδαστές η επαγγελματίες επιλέχτηκαν για την συγκεκριμένη έρευνα.

Κριτήρια απόρριψης

- Μελέτες οι οποίες περιέγραφαν τον προγραμματισμό ανά ζεύγη σε θεωρητικό επίπεδο.
- Μελέτες με «γνώμες ειδικών» χωρίς αποδείξεις.
- Μελέτες που ασχολήθηκαν με τα εργαλεία υποστήριξης του προγραμματισμού ανά ζεύγη(software και hardware).
- Μελέτες που έγιναν με την μέθοδο της προσομοίωσης αλλά και μελέτες που ανέλυαν διάφορα οικονομικά και μαθηματικά μοντέλα.
- Μελέτες που δεν έχουν δημοσιευθεί σε κάποιο ευρέως αναγνωρισμένο συνέδριο η journal.
- Μελέτες οι οποίες δεν γράφτηκαν στα αγγλικά.

Κριτήρια συμπερίληψης

- [Q1] Systematic literature Reviews, case or field studies, Surveys και πειράματα που είχαν αντικείμενο τον προγραμματισμό ανά ζεύγη και απαραιτήτως είχαν εμπειρικά δεδομένα για να τις υποστηρίξουν.
- [Q2] Μελέτες που θα βοηθήσουν στο να απαντηθούν οι ερωτήσεις της συγκεκριμένης έρευνας.
- Μελέτες που διεξήχθηκαν από επαγγελματίες αλλά και σπουδαστές.
- Μελέτες που έχουν περάσει το κατώφλι στοιχείων ποιότητας.

3.4 Ορισμός στοιχείων ποιότητας

Οι μελέτες που πέρασαν τον αρχικό παραπάνω έλεγχο αξιολογήθηκαν επιμελώς χρησιμοποιώντας τις τεχνικές και τα κριτήρια που περιγράφονται από την μέθοδο Critical Appraisal Skills Progamme(CASP)[7] και τις διαδικασίες που προτείνονται από την μελέτη της Kitchenham [8], Dyba[9] και Sfetsos[10]. Αυτά τα κριτήρια συνοψίζονται στις 3 παρακάτω κατηγορίες

- *Rigour:* Είναι ορθή και κατάλληλη η τεχνική που εφαρμόστηκε για να προσεγγίσει τις κύριες ερευνητικές μεθόδους της μελέτης;

- *Credibility:* Είναι τα ευρήματα της έρευνας ορθά, έχουν νόημα και είναι σωστά παρουσιασμένα;

- *Relevance:* Προσφέρουν κάτι στην γνώση που υπάρχει ήδη και μπορούν να εφαρμοστούν στην βιομηχανία;

Αναλύοντας τις 3 παραπάνω κατηγορίες προκύπτουν τα στοιχεία ποιότητας. Η κατηγορία αυστηρότητας (Rigour) έχει τα εξής κριτήρια ποιότητας:

- [Q3]Αν δικαιολογεί ο ερευνητής γιατί επέλεξε την συγκεκριμένη μέθοδο για την έρευνά του
- [Q4]Δίνει κάποια εξήγηση γιατί χρησιμοποιήθηκε το συγκεκριμένο δείγμα και ποια ήταν τα κριτήρια επιλογής του

- [Q5]Υπήρχε κάποιο control group για σύγκριση αποτελεσμάτων
- [Q6]Υπήρχε περιγραφή για το πώς συγκεντρώθηκαν τα δεδομένα και γιατί χρησιμοποιήθηκε η συγκεκριμένη μέθοδος
- [Q7]Υπήρχε περιγραφή για το πώς αναλύθηκαν τα δεδομένα και αν καταφέρνουν τα δεδομένα αυτά να απαντήσουν τα ερωτήματα της έρευνας

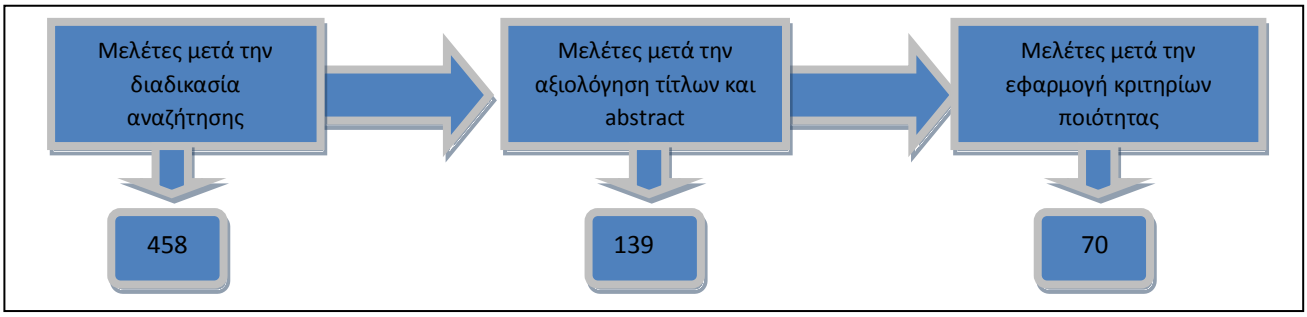
Η κατηγορία αξιοπιστίας (credibility) έχει και αυτή τα δικά της κριτήρια που είναι:

- [Q8]Ο ερευνητής συζητά την οποιαδήποτε προκατάληψη(bias), τον ρόλο του που μπορεί να επηρεάζουν τα αποτελέσματα αλλά και τυχών περιορισμούς που προκύπτουν
- [Q9]Το πώς καταλήγουν οι ερευνητές δικαιολογείται από τα ευρήματα τους

Τελικά και η κατηγορία της συνάφειας (relevance) με το μοναδικό κριτήριο ποιότητας:

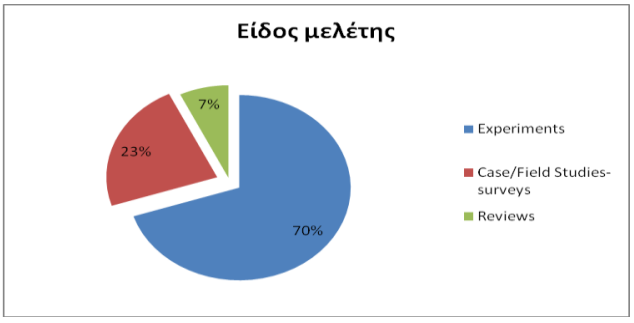
- [Q10]Είναι τα ευρήματα της μελέτης χρήσιμα για την έρευνα ή για την βιομηχανία της πληροφορικής

Κάθε ένα από αυτά τα κριτήρια βαθμολογήθηκε με κλίμακα 0/1 ή αλλιώς ναι/όχι. Όλες οι μελέτες που συμπεριλήφθηκαν στην συγκεκριμένη έρευνα αξιολογήθηκαν μέσω των κριτηρίων ποιότητας έτσι ώστε να διασφαλισθεί ότι θα έπαιζαν σημαντικό ρόλο στην εξαγωγή των απαντήσεων. Εάν μια μελέτη απαντούσε θετικά σε κάποιο κριτήριο τότε έπαιρνε 1 βαθμό. Εάν δεν κατάφερνε να απαντήσει θετικά τότε έπαιρνε 0 βαθμούς. Αθροίζοντας τους βαθμούς που παίρνει μια μελέτη για κάθε κριτήριο έχουμε την συνολική ποιότητα της κάθε μελέτης. Τα κριτήρια Q1 και Q2 είναι τα μόνα που καθορίζουν εάν μια μελέτη θα συμπεριλαμβανόταν στην έρευνα ή όχι και ορίζονται ως το κατώφλι ποιότητας. (Πίνακας ποιότητας στο appendix D)



3.6 Ορισμός μεθόδων συλλογής δεδομένων

Στις μελέτες που πέρασαν το κατώφλι ποιότητας έγινε εξαγωγή δεδομένων. Ο τίτλος, ο συγγραφέας, ο τύπος μελέτης(πείραμα, survey, case study κτλ.), το περιβάλλον μελέτης(ακαδημαϊκό, βιομηχανικό), ο πληθυσμός του δείγματος και τα κύρια συμπεράσματα είναι αυτά που εξήχθησαν και δημιούργησαν την συνοπτική φόρμα εξαγόμενων δεδομένων. Σε ορισμένες περιπτώσεις στάθηκε αδύνατο να συμπληρωθούν κάποια στοιχεία στην φόρμα γιατί δεν ήταν ξεκάθαρο στην εκάστοτε μελέτη και έτσι αφέθηκαν κενά. Συνολικά συγκεντρώθηκαν 49 πειράματα, 16 case/field studies-surveys και 5 reviews όπως φαίνεται από στην διπλανή πίτα.



3.7 Σύνθεση δεδομένων

Τα παραπάνω δεδομένα που είχαν εξαχθεί στο προηγούμενο στάδιο τώρα συντέθηκαν και εξήχθησαν τα συμπεράσματα που καταλήγουν την έρευνα. Τα συνοπτικά διαγράμματα δεδομένων ακολουθούν στο τέλος της έρευνας στο Appendix A.

4. Αποτελέσματα

Παρακάτω ακολουθεί μια σύντομη περιγραφή για κάθε μία μελέτη που συμπεριλήφθηκε στην συγκεκριμένη έρευνα. Αρχικά περιγράφονται τα πειράματα, έπειτα τα case/field studies – surveys και τέλος τα reviews.

4.1 Experiments

Muller et. al. στο [S1][**An Empirical Study about the Feelgood Factor in Pair Programming**] διεξήγαγε 2 ξεχωριστά πειράματα. Στα πειράματα αυτά προσπάθησε να εξηγήσει τις επιπτώσεις που έχει στην απόδοση των προγραμματιστών, η εμπειρία αλλά και το πόσο καλά αισθάνονται (χρησιμοποιεί τον όρο "feelgood factor") κατά την διάρκεια του προγραμματισμού ανά ζεύγη. Στα πειράματα αυτά έλαβαν μέρος 38 μαθητές. Κατά την διάρκεια των πειραμάτων οι μαθητές έπρεπε να λύσουν συγκεκριμένα προβλήματα προγραμματισμού σε Java και για να θεωρηθούν ότι έχουν λυθεί έπρεπε να περάσουν 95 από τα 100 test cases. Κατέληξαν ότι το πόσο καλά αισθάνεται το ζευγάρι είναι ένας υποψήφιος παράγοντας για την απόδοσή. Αντίθετα η εμπειρία δεν παίζει καμία σημασία. Παρόλα αυτά μας εξηγούν ότι το δείγμα τους ήταν μικρό και τα αποτελέσματά τους δεν ήταν ισορροπημένα. Γι αυτό περισσότερες έρευνες με μεγαλύτερο δείγμα και περισσότερες μεταβλητές είναι απαραίτητο να γίνουν για εξαγωγή ασφαλών συμπερασμάτων.

Norsaremah Salleh et. al. στο [S2] [**An Empirical Study of the Effects of Personality in Pair Programming using the Five-Factor Model**] πείραμα που έκαναν θέλησαν να μελετήσουν την επιρροή των διαφορετικών προσωπικοτήτων στην διαδικασία του PP. Το πείραμα τους χρησιμοποίησε το μοντέλο των 5 προσωπικοτήτων(ανοιχτός σε εμπειρίες, ευσυνειδησία, εξωστρέφεια, τερπνότητα, νευρωτισμός) και υποβλήθηκαν στο τεστ προσωπικότητας IPIP για να γίνει σαφής προσδιορισμός . Επιλέγονταν φοιτητές για κάθε μάθημα έτσι ώστε να δημιουργούνται ζευγάρια είτε με όμοια προσωπικότητα είτε με διαφορετική. Το πείραμα έδειξε ότι ανεξάρτητα από την βαθμολογία που πέτυχαν οι φοιτητές τα επίπεδα της ικανοποίησης αλλά και της αυτοπεποίθησης ήταν αρκετά υψηλά. Συσχέτισαν παράλληλα την επιτυχία μεγάλων βαθμών στα τεστ με την υψηλή ευσυνειδησία αλλά και την ανοικτότητα σε νέες εμπειρίες.

Rafael Duque et. al. [S3] [**Analyzing Work Productivity and Program Quality in Collaborative Programming**] σε μια σειρά πειραμάτων που έκαναν προσπάθησαν να δείξουν την υπεροχή του απομακρυσμένου προγραμματισμού ανά ζεύγη έναντι των παραδοσιακών μεθόδων των solo προγραμματιστών. Στα πειράματά τους χρησιμοποίησαν το COLLECE ένα πρόγραμμα ειδικά αναπτυγμένο σε java για να τους βοηθήσει στον απομακρυσμένο PP. Τα πειράματα τους επικεντρώθηκαν σε φοιτητές πληροφορικής ίδιου εξαμήνου με εμπειρία τουλάχιστον ενός χρόνου. Κινήθηκαν γύρω από την αποδοτικότητά τους αλλά και την ποιότητα του παραγόμενου προγράμματος. Με την ανάλυση των αποτελεσμάτων κατέληξαν ότι οι προγραμματιστές ανά ζεύγη χρειάζονται περισσότερο χρόνο για την περαίωση των εργασιών τους. Αυτό συμβαίνει λόγω του χρόνου που χάνεται στην επικοινωνία μεταξύ του ζευγαριού. Παρόλα αυτά οι προγραμματιστές ανά ζεύγη γενικά παράγουν καλύτερα ποιοτικά προγράμματα με λιγότερα λάθη.

Laurie Williams et. al. [S4][**Building Pair Programming Knowledge through a Family of Experiments**] σε μια σειρά πειραμάτων προσπάθησαν να κατανοήσουν την αποδοτικότητα του PP σαν μια εναλλακτική μορφή διδασκαλίας. Τα πειράματα έγιναν σε 2 πανεπιστήμια των ΗΠΑ και έλαβαν μέρος 1200 φοιτητές. Έκαναν τα πειράματα τους έχοντας στο μυαλό τις εξής έξι υποθέσεις.

1) Ίσο ή μεγαλύτερο ποσοστό προγραμματιστών ανά ζεύγη θα περάσει το μάθημα με βαθμό C σε σχέση με τους solo προγραμματιστές
2) Προγραμματιστές ανά ζεύγη θα πετύχουν ίσο ή μεγαλύτερο βαθμό στις τελικές εξετάσεις από τους solo προγραμματιστές.
3) Οι προγραμματιστές ανά ζεύγη θα παράγουν καλύτερα προγράμματα σε σχέση με τους solo.
4) Οι προγραμματιστές ανά ζεύγη θα διασκεδάσουν κατά την διάρκεια του PP και θα αποκτήσουν μια θετική στάση απέναντι στις τεχνικές προγραμματισμού με συνεργασία
5) Ο προγραμματισμός ανά ζεύγη σε αυτό το εισαγωγικό μάθημα δεν θα επηρεάσει αρνητικά την μελλοντική τους απόδοσή σαν solo προγραμματιστές.

6) Οι φοιτητές που ήταν προγραμματιστές ανά ζεύγη ένα χρόνο αργότερα θα επιδιώξουν κάποια ειδικότητα σχετική με την επιστήμη των υπολογιστών

Με τα δεδομένα που συνέλεξαν κατάφεραν να υποστηρίξουν και τις 6 παραπάνω υποθέσεις. Μελλοντικά θέλουν να επαναλάβουν παρόμοια πειράματα και να εξετάσουν την αποδοτικότητα και την συμβατότητα των ζευγαριών μέσω τέστ προσωπικότητας .

Joseph Chao et. al. [S5] [**Critical Personality Traits in Successful Pair Programming**] διεξήγαγαν πειράματα για να εντοπίσουν ποια είναι τα κρίσιμα γνωρίσματα προσωπικότητας που είναι απαραίτητα για την επιτυχία του PP. Σε μια on-line δημοσκόπηση που συμμετείχαν επαγγελματίες προγραμματιστές κατηγοριοποιήθηκαν 15 διαφορετικά γνωρίσματα προσωπικότητας. Τα σημαντικότερα από αυτά ήταν η ανοικτομυαλοσύνη (open minded), η λογική, η προσήλωση, η δημιουργικότητα, η υπευθυνότητα και η ευελιξία. Με αυτή την γνώση που απέκτησαν από τους επαγγελματίες έκαναν πειράματα πάνω σε φοιτητές πληροφορικής. Αφού έκαναν τεστ προσωπικότητας στους φοιτητές τους χώρισαν σε ζευγάρια σύμφωνα με τα παραπάνω ευρήματα. Τους δόθηκαν προβλήματα να λυθούν και έπειτα απαντούσαν σε ερωτηματολόγια για να αξιολογηθεί η συμβατότητα με το ζεύγος τους. Η ποιότητα των προγραμμάτων κρίθηκε από το αν θα έβγαζαν σωστά αποτελέσματα, αν χρησιμοποίησαν σωστά αντικείμενα και interface και εάν είχαν σωστό προγραμματιστικό στυλ. Κατέληξαν ότι ζευγάρια με μεγαλύτερη «ανοικτομυαλοσύνη» και υπευθυνότητα ήταν αυτά με την καλύτερη ποιότητα κώδικα.

Jari Vanhanen και Casper Lassenius [S6] [**Effects of Pair Programming at the Development Team Level: An Experiment**] σε πείραμα που έκαναν σε φοιτητές μελέτησαν τις επιπτώσεις του PP στην παραγωγικότητα, στα ελαττώματα, στην ποιότητα του σχεδιασμού, στην μεταβίβαση της γνώσης και της ευχαρίστησης στην δουλειά προσομοιώνοντας πραγματικές συνθήκες εργασίας. Γι αυτό τον λόγο οι φοιτητές που επιλέχτηκαν έπρεπε να πληρούν κάποιες υψηλές προϋποθέσεις και κρητίρια. Οι προγραμματιστές ανά ζεύγη αλλά και οι Solo προγραμματιστές έπρεπε να φέρουν εις πέρας ένα ολοκληρωμένο project μέσα σε 400 ώρες. Οι προγραμματιστές ανά ζεύγη τελικά παρουσίασαν 29% μειωμένη παραγωγικότητα σε σχέση με τους σόλο γιατί χρειάστηκαν χρόνο να γνωρίσουν και να οικειοποιηθούν το ζεύγος τους. Αυτό είναι όμως κόστος σε μια εταιρία που θα υπάρξει μια φορά και θα αποσβεστεί κατά την διάρκεια του χρόνου. Οι σόλο προγραμματιστές είχαν μεγαλύτερη επιτυχία στο να βρίσκουν λάθη στον κώδικα ενώ αντίθετα οι PP είχαν καλύτερη σχεδίαση αλλά μικρότερη λειτουργικότητα και αυξημένο αριθμό λαθών στο τελικό πρόγραμμα. Γενικά οι PP ασχολήθηκαν και κατανόησαν περισσότερα πακέτα(java) σε σχέση με τους σόλο κάτι που μας λέει ότι είχαν μια καλύτερη μετάδοση γνώσης.

Jo E. Hannay et. al. [S7] [**Effects of Personality on Pair Programming**] έκαναν μια μελέτη για τις επιπτώσεις στον PP των πέντε χαρακτηριστικών της προσωπικότητας (ανοιχτός σε εμπειρίες, ευσυνειδησία, εξωστρέφεια, τερπνότητα, νευρωτισμός) μαζί με την εξειδίκευση και την πολυπλοκότητα. Το πείραμα είχε 196 επαγγελματίες προγραμματιστές από 3 χώρες. Τα αποτελέσματα της έρευνάς τους δεν επιβεβαιώνουν τα μοντέλα που δείχνουν ότι ορισμένα χαρακτηριστικά της προσωπικότητας έχουν επίπτωση στην αποδοτικότητα. Η προσωπικότητα είναι ένα χαρακτηριστικό που είναι αμετάβλητο. Γι αυτό τον λόγο καταλήγοντας μας λένε ότι οι μελέτες πρέπει να δώσουν μεγαλύτερο βάρος σε άλλους παράγοντες αποδοτικότητας. Σε παράγοντες οι οποίοι μπορούν να αναπτυχθούν και να βελτιωθούν όπως είναι η εξειδίκευση, οι προγραμματιστικές ικανότητες, η μάθηση αλλά και το κίνητρο.

Laurie Williams et. al. [S8] [**Examining the Compatibility of Student Pair Programmers**] κατά την διάρκεια ενός πειράματος που έγινε από το 2002 έως το 2005 σε 1350 φοιτητές του πανεπιστημίου της Βόρειας Καρολίνας, μελέτησαν το πόσο συμβατά είναι τα ζευγάρια που όριζαν οι καθηγητές βάσει του τύπου προσωπικότητας, τον τρόπο εκμάθησης, το επίπεδο ικανότητας, την αυτοεκτίμηση τους ως προγραμματιστές, τα ήθη στην εργασία και την διαχείριση χρόνου στην εργασία. Τα αποτελέσματα έδειξαν ότι το 93% των ζευγαριών είναι συμβατά μεταξύ τους. Τα ζευγάρια μάλιστα είναι ακόμη πιο συμβατά εάν διαλέξουμε τα μέλη τους τυχαία χωρίς να λαμβάνουμε υπ ‘όψιν τα παραπάνω αναφερόμενα χαρακτηριστικά. Οι φοιτητές προτιμούν να είναι σε ζευγάρι όπου και οι 2 θα έχουν το ίδιο επίπεδο ικανότητας. Αυτό πιθανότατα οδηγεί στο φαινόμενο ο ένας φοιτητής να συμπληρώνει τις γνώσεις του άλλου φοιτητή. Τα πιο ασύμβατα ζευγάρια είναι αυτά που έχουν διαφορετικά ήθη επάνω στην εργασία(πχ κάποιος δουλεύει πρωί κάποιος αργά το βράδυ).

Raymund Sison [S9] [**Investigating the Effect of Pair Programming and Software Size on Software Quality and Programmer Productivity]** έκανε 2 πειράματα με την βοήθεια του PP για να εξακριβώσει εμπειρικά την αύξηση της ποιότητας των παραγόμενων προγραμμάτων έναντι των παραδοσιακών τεχνικών . Το πρώτο πείραμα αφορούσε την ανάπτυξη ενός μικρού συστήματος 10,000 έως 100,000 γραμμών κώδικα. Το δεύτερο την επίλυση μικρών προγραμματιστικών προγραμμάτων μέχρι 10,000 γραμμές κώδικα. Στα πειράματα έλαβαν μέρος 48 τριτοετής φοιτητές που έρχονταν για πρώτη φορά σε επαφή με τον PP. Οι φοιτητές κατά την διάρκεια των πειραμάτων χρησιμοποίησαν την τεχνική PSP(Personal Software Process) και μέσω ενός εξειδικευμένου εργαλείου κατέγραφαν κάθε σφάλμα αλλά και τον χρόνο που χρειάστηκε κατά την διάρκεια της ανάλυσης, του σχεδιασμού, της συγγραφής και των δοκιμών. Τα αποτελέσματα έδειξαν πως στα μικρά και πολύπλοκα συστήματα υπήρχε σημαντική μείωση στα λάθη που γίνονταν και αύξηση της ποιότητας από τους PP έναντι των solo προγραμματιστών παρόλο που δεν υπήρχε αξιοσημείωτη διαφορά στην παραγωγικότητα των 2 ομάδων. Αντίθετα στα πολύ μικρά προγράμματα τα σφάλματα που έγιναν ήταν περίπου τα ίδια αλλά οι PP είχαν μείωση στην παραγωγικότητα τους.

Nick Z. Zacharis [S10][**Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course**] σε πείραμα που έκανε στο πανεπιστήμιο του Πειραιά μελέτησε τον «εικονικό» (απομακρυσμένο) προγραμματισμό ανά ζεύγη(VPP). Το πείραμα έγινε σε 129 φοιτητές χωρίς προηγούμενη προγραμματιστική εμπειρία. Ένας από τους σκοπούς του ήταν να μεταφέρει το μάθημα σε μια πιο μαθησιο-κεντρική και συνεργασιακή κατεύθυνση. Θέλησε να απαντήσει σε 3 κύριες ερωτήσεις. Φοιτητές που θα χρησιμοποιήσουν τον VPP θα παράγουν καλύτερα προγράμματα σε μικρότερο χρόνο σε σχέση με τους σόλο προγραμματιστές. Φοιτητές που χρησιμοποιούν τον VPP θα έχουν ίδιους ή μεγαλύτερους τελικούς βαθμούς στο μάθημα. Φοιτητές που χρησιμοποιούν τον VPP θα διασκεδάσουν κατά την διάρκεια του πειράματος και θα έχουν μελλοντικά μια θετική άποψη στο θέμα της συνεργασίας στον προγραμματισμό. Τα αποτελέσματα του πειράματος επιβεβαίωσαν και τις 3 παραπάνω υποθέσεις. Οι προγραμματιστές VPP σε ποσοστό περίπου 50% είχαν καλύτερη ποιότητα κώδικα με λιγότερα λάθη και μεγαλύτερη παραγωγικότητα.

Jari Vanhanen et. al. [S11][**Perceived Effects of Pair Programming in an Industrial Context**] έκαναν ένα πείραμα για την υπεροχή του PP σε σχέση με τους solo προγραμματιστές. Το πείραμα αυτό έλαβε μέρος σε μια μεσαίου μεγέθους εταιρία ανάπτυξης προγραμμάτων στην Φιλανδία και συμμετείχαν 28 επαγγελματίες προγραμματιστές. Με την έρευνά τους κατάφεραν να δείξουν πως προηγούμενες μελέτες που περιοριζόταν σε μικρό περιβάλλον μπορούν να γενικοποιηθούν σε πραγματικές συνθήκες εργασίας στην βιομηχανία. Τα αποτελέσματά του PP είναι ιδιαιτέρως θετικά στην ποιότητα των προγραμμάτων, στην τήρηση χρονοδιαγραμμάτων, στην εκπαίδευση, στην διασκέδαση στην δουλειά και στην πειθαρχεία. Πέρα από τα θετικά στοιχεία ανακάλυψαν και 2 αρνητικά. Οι προγραμματιστές ανά ζεύγη χρειάστηκε να καταβάλουν μεγαλύτερη προσπάθεια για την ίδια εργασία σε σχέση με τους solo προγραμματιστές. Επίσης οι προγραμματιστές σε ζεύγη βρήκαν την εργασία στα αρχικά στάδια περισσότερο κουραστική. Όσο όμως προχωρούσε ο χρόνος η κούραση έπεσε στα ίδια επίπεδα με τους solo.

Brian F. Hanks [S12][**Distributed Pair Programming: An Empirical Study**] μελέτησε τον απομακρυσμένο προγραμματισμό ανά ζεύγη. Στο πείραμα που έκανε θέλησε να βρει την αποτελεσματικότητα που θα έχει ένα απομακρυσμένο ζεύγος σε σχέση με αυτούς που βρισκόταν στον ίδιο χώρο. Ανέπτυξε ένα εργαλείο διαμοιρασμού επιφάνειας εργασίας μέσω δικτύου(VNC) που θα βοηθούσε τους απομακρυσμένους προγραμματιστές. Ταυτόχρονα με χρήση προγραμμάτων επικοινωνίας μέσω ήχου οι απομακρυσμένοι προγραμματιστές «ένιωθαν» ότι βρισκόταν στον ίδιο χώρο. 76 φοιτητές έλαβαν μέρος στο πείραμα. Τα αποτελέσματα της έρευνας έδειξαν ότι οι φοιτητές που χρησιμοποίησαν το εργαλείο είχαν ίδια απόδοση με τους φοιτητές που ήταν στον ίδιο χώρο και είχαν τα ίδια επίπεδα αυτοπεποίθησης. Καταλήγοντας μας λέει πως θα χρειαστεί μεγάλη προσπάθεια για να δημιουργηθεί ένα κατάλληλο εμπορικό σύστημα που θα υποστηρίξει τον DPP.

Gerardo Canfora et. al. [S13][**Empirical Study on the Productivity of the Pair Programming**] έκανε ένα πείραμα για να εξακριβώσει την παραγωγικότητα του PP σε σχέση με τους solo προγραμματιστές. Το πειραμά τους έγινε με την βοήθεια φοιτητών. Κατέληξαν στο συμπέρασμα πως ο ίδιος προγραμματιστής εάν εμπλακεί σε κατάσταση προγραμματισμού ανά ζεύγη θα χρειαστεί λιγότερο χρόνο να ολοκληρώσει μια εργασία απ ότι αν ήταν solo προγραμματιστής. Μελλοντικά σχεδιάζουν να επαναλάβουν το πείραμα τους για να ισχυροποιήσουν τα αποτελέσματα τους.

Irina Diana Coman et. al. [S14][**Investigating the Usefulness of Pair-Programming in a Mature Agile Team**] στο πείραμα που έκαναν ερεύνησαν τον τρόπο που λειτουργεί ο PP σε μια ομάδα που συνεχώς χρησιμοποιούσε τεχνικές XP. Το πείραμα έλαβε χώρα σε μια Ιταλική εταιρία για 84 ημέρες με 16 προγραμματιστές. Κατά τον πρώτο μήνα του project ο PP χρησιμοποιούνταν 60% από τους «βετεράνους» προγραμματιστές και λίγο περισσότερο από τους νεοφερμένους στην εταιρεία. Αυτό τους βοήθησε να διαπιστώσουν πως ο PP βοηθά σημαντικά στην εκπαίδευση των μελλών της επιχείρησης αλλά και στην μετάδοση γνώσης σε ολόκληρη την ομάδα για το αναπτυσσόμενο project.

Nachiappan Nagappan et. al. [S15][**Pair Learning: With an Eye Toward Future Success**] διεξήγαγαν ένα πείραμα για μπορέσουν να διαπιστώσουν την αποτελεσματικότητα του προγραμματισμού ανά ζεύγη ως εισαγωγικό μάθημα σε πανεπιστήμια που ασχολούνται με την επιστήμη των υπολογιστών. Σκοπός τους ήταν να απαντήσουν της εξής ερωτήσεις:

1. Ίσο ή μεγαλύτερο ποσοστό προγραμματιστών ανά ζεύγη θα περάσει το μάθημα με βαθμό C η μεγαλύτερο σε σχέση με τους solo προγραμματιστές.
2. Οι προγραμματιστές ανά ζεύγη θα έχουν μια θετική στάση απέναντι στις συνεργασιακές προγραμματιστικές τεχνικές.
3. Κάνοντας ένα εισαγωγικό μάθημα σχετιζόμενο με προγραμματισμό ανά ζεύγη δεν θα έχει επίπτωση στα μελλοντικά solo προγραμματιστικά τους μαθήματα
4. Οι καθηγητές/επιβλέποντες των προγραμματιστών ανά ζεύγη θα έχουν λιγότερο φόρτο εργασίας αναφορικά με βαθμολόγηση, λιγότερες ερωτήσεις προς απάντηση, λιγότερη προσπάθεια διδασκαλίας σε σχέση με τους καθηγητές των solo προγραμματιστών.

Στα 2 χρόνια που διήρκησε το πείραμα σε ορισμένα εξάμηνα υπήρξε μια αύξηση της επιτυχίας των μαθητών μέχρι και 85% σε σχέση με τους σόλο. Σε ποσοστό 59% έως 64% οι φοιτητές είχαν μια πολύ θετική στάση απέναντι στον προγραμματισμό ανά ζεύγη παρόλο που ορισμένοι από αυτούς παραπονέθηκαν ότι το ζεύγος τους «καθυστερούσε» και άλλοι ότι δεν είχαν προηγουμένη solo εμπειρία. Επίσης τα αποτελέσματα τους έδειξαν ότι ο PP δεν επηρέασε σε σημαντικό ποσοστό την απόδοσή τους ως solo προγραμματιστές. Οι προγραμματιστές ανά ζεύγη κατά την διάρκεια του μαθήματος έτειναν να κάνουν πιο ουσιώδες ερωτήσεις (βελτίωση αλγορίθμου, χρήση διαφορετικών σεναρίων) σε σχέση με τους solo οι οποίοι ασχολούνταν συνεχώς με βασικές ερωτήσεις όπως η σύνταξη.

Emilio Bellini et. al. [S16][**The Impact of Educational Background on Design Knowledge Sharing During Pair Programming: An Empirical Study**] έκαναν ένα πείραμα για να εξακριβώσουν την σχέση μεταξύ του εκπαιδευτικού υποβάθρου και την μεταφορά γνώσης ανάμεσα στους προγραμματιστές ανά ζεύγη που σχεδιάζουν ένα πρόγραμμα. Το πείραμα έγινε σε μεταπτυχιακούς φοιτητές διάφορων επιστημών όπως οι οικονομολόγοι, φιλόλογοι, φιλόσοφοι (MUTEGS-non scientific) και φυσικοί, μαθηματικοί, μηχανικοί(MUTS-scientific) όπου όλοι είχαν την ίδια γνώση πάνω στην επιστήμη των υπολογιστών. Χωρίστηκαν τυχαία σε 4 ζευγάρια με έναν φοιτητή MUTS και έναν MUTEGS, σε 5 ζευγάρια με 2 φοιτητές MUTS, σε 5 ζευγάρια με 2 φοιτητές MUTEGS, και 16 solo προγραμματιστές MUTS και MUTEGS. Τα αποτελέσματα έδειξαν πως οι προγραμματιστές με το ίδιο εκπαιδευτικό υπόβαθρο ωφελήθηκαν τα μέγιστα του προγραμματισμού ανά ζεύγη. Η σύζευξη μεταξύ scientific και non-scientific φαίνεται να είναι η χειρότερη με άσχημα αποτελέσματα και για τους δυο. Η σιωπηρή γνώση που χρειάζεται για τον σχεδιασμό είναι μεγαλύτερη σε σχέση με τον προγραμματισμό άρα και γνώση που μοιράζεται μεταξύ των σχεδιαστών είναι μεγαλύτερη.

Matthias M. Muller [S17][**Do programmer pairs make different mistakes than solo programmers?**] έκανε ένα πείραμα για να εξακριβώσει τα λάθη που γίνονται από προγραμματιστές σε ζεύγη και solo προγραμματιστές. Στο πείραμα έλαβαν μέρος 38 φοιτητές. Τα αποτελέσματα έδειξαν πως οι προγραμματιστές σε ζεύγη κάνουν τα ίδια αλγοριθμικά λάθη με τους solo προγραμματιστές. Επίσης οι προγραμματιστές ανά ζεύγη έχουν λιγότερα εκφραστικά λάθη στον κώδικα και παρουσιάζουν ίδια περίπου ικανότητα επίλυσης με τους solo σε προβλήματα τύπου eureka. Σε δύσκολα και πολύπλοκα προβλήματα όπου η λύση μπορεί εύκολα να εντοπιστεί σαν σωστή, φαίνεται ότι οι προγραμματιστές ανά ζεύγη έχουν καλύτερες πιθανότητες επίλυσης σε σχέση με τους solo.

Kyungsub Choi et. al. [S18][**Pair dynamics in team collaboration**] στο πείραμα που έκαναν μελέτησαν πως η προσωπικότητα, οι ικανότητες επικοινωνίας και το φύλο έχουν επίπτωση στον προγραμματισμό ανά ζεύγη σε σχέση με τα επίπεδα ικανοποίησης, συμβατότητας, επικοινωνίας και αυτοπεποίθησης. Οι φοιτητές που έλαβαν μέρος στο πείραμα αξιολογήθηκαν για την προσωπικότητα τους με την μέθοδο MBTI και για την ικανότητα επικοινωνίας με την μέθοδο CSRS. Τα αποτελέσματα της έρευνας έδειξαν πως η προσωπικότητα δεν έπαιξε κανέναν σημαντικό ρόλο στην ικανοποίηση, συμβατότητα, επικοινωνία και αυτοπεποίθηση των προγραμματιστών σε ζεύγη. Αξίες όπως η μετριοφροσύνη, η αξιοπρέπεια και οι τρόποι συμπεριφοράς χαρακτηρίζουν περισσότερο την επιτυχή συνεργασία απ ότι ο τύπος της προσωπικότητας. Τα ζεύγη ομοίου φύλου έδειξαν πως έχουν μεγαλύτερη συνεκτικότητα. Σαν αποτέλεσμα αυτού είχαν καλύτερη επικοινωνία, συμβατότητα και ικανοποίηση.

David Stotts et. al. [S19][**Virtual Teaming: Experiments and Experiences withDistributed Pair Programming]** στα 2 πειράματα που έκαναν μελέτησαν τον απομακρυσμένο προγραμματισμό ανά ζεύγη και τα οφέλη του σε σχέση με τον προγραμματισμό ανά ζεύγη στον ίδιο χώρο. Η έρευνα τους έγινε σε φοιτητές 2 πανεπιστήμιων με την βοήθεια οπτικοακουστικών προγραμμάτων επικοινωνίας και προγραμμάτων διαμοιρασμού επιφάνειας εργασίας. Τα αποτελέσματα έδειξαν πως ο απομακρυσμένος προγραμματισμός ανά ζεύγη είναι εφικτός τρόπος για την δημιουργία προγραμμάτων. Τα προγράμματα είναι ίσης ποιότητας με αυτά των προγραμματιστών ανά ζεύγη στον ίδιο χώρο και είναι εφικτή η απομακρυσμένη συνεργασία με απλά και ευρέως διαδεδομένα προγράμματα. Επίσης οι απομακρυσμένοι προγραμματιστές διατηρούν τα οφέλη που έχουν οι προγραμματιστές στον ίδιο χώρο(pair pressure, pair learning).

Panagiotis Sfetsos et. al. [S20][**An experimental investigation of personality types impact on pair effectiveness in pair programming**] στο πείραμα που έκανε μελέτησε την επίπτωση που έχει η προσωπικότητα στην αποτελεσματικότητα και την συνεργασίας στον προγραμματισμό ανά ζεύγη. Στο πείραμα έλαβαν μέρος 70 φοιτητές αφού υποβλήθηκαν στα τεστ MBTI και KTS για τον προσδιορισμό της προσωπικότητας και της ιδιοσυγκρασίας. Τα αποτελέσματα του πειράματος αποκαλύπτουν πως ζεύγη φοιτητών ανομοιογενούς προσωπικότητας είχαν μεγαλύτερη αποτελεσματικότητα και συνεργασία σε σχέση με τα ομοιογενή ζεύγη. Η επικοινωνία ήταν επίσης πολύ υψηλή στα ανομοιογενή ζευγάρια κάτι που οδήγησε στην σωστότερη συγγραφή κώδικα και στον καλύτερο σχεδιασμό προγραμμάτων. Κατέληξαν πως ίσως χρειάζονται περισσότερα πειράματα για την επιβεβαίωση των αποτελεσμάτων τους.

Gerardo Canfora et. al. [S21][**Evaluating performances of pair designing in industry**] σε πείραμα που έκαναν μελέτησαν τον προγραμματισμό ανά ζεύγη κατά την διαδικασία του σχεδιασμού προγραμμάτων στην βιομηχανία. Προσπάθησαν να απαντήσουν στην ερώτηση εάν ο σχεδιασμός ανά ζεύγη χρειάζεται μικρότερη προσπάθεια σε σχέση με τον solo και εάν θα παράγουν καλύτερα σχέδια ποιοτικώς. Στο πείραμα τους έλαβαν μέρος μερικοί από τους εργαζόμενους της εταιρίας Soluziona Software Factory. Τα αποτελέσματα της έρευνας δείχνουν ότι η απόδοση των σχεδιαστών σε ζεύγη είναι χαμηλότερη σε σχέση με τον προγραμματισμό ανά ζεύγη. Επίσης βρήκαν πως τα σχέδια των ζευγαριών είναι ποιοτικότερα από αυτά των solo χωρίς να υπάρχει κάποιο όφελος ως προς τον χρόνο. Συμπερασματικά μας επισημαίνουν πως ο προγραμματισμός ανά ζεύγη μπορεί επιτυχώς να χρησιμοποιηθεί και σε άλλες φάσεις της ανάπτυξης του λογισμικού όπως είναι αυτή του σχεδιασμού.

Erik Arisholm et. al. [S22][**Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise]** διεξήγαγε ένα μεγάλο πείραμα μεταξύ 295 προγραμματιστών, όλων των επιπέδων από 3 χώρες. Σκοπός του πειράματος ήταν να μελετήσουν τον προγραμματισμό ανά ζεύγη σε σχέση με την πολυπλοκότητα του συστήματος και την εμπειρία. Κατά την διάρκεια του πειράματος έγιναν διάφορες εργασίες συντήρησης ρουτινών σε 2 ξεχωριστά συστήματα java με διαβαθμισμένη πολυπλοκότητα. Τα αποτελέσματα έδειξαν πως δεν υπάρχει ιδιαίτερη υπεροχή των PP αναφορικά με τον χρόνο που χρειάζεται να βρεθεί μια σωστή λύση σε σχέση με τους Solo. Αντίθετα υπάρχει 84% αύξηση στην καταβαλλόμενη προσπάθεια από τους PP ώστε να βρεθεί μια σωστή λύση. Στα πολύπλοκα συστήματα υπάρχει αύξηση 48% στην εύρεση σωστών λύσεων αλλά όχι σημαντική διαφορά στον χρόνο εύρεσης αυτών των λύσεων. Για τα απλά συστήματα υπήρξε μείωση του χρόνου κατά 20% αλλά όχι σημαντικές διαφορές στην εύρεση σωστών λύσεων. Κλείνοντας μας λένε πως είναι καλύτερο σε πολύπλοκες διαδικασίες συντήρησης να χρησιμοποιούνται μη έμπειροι προγραμματιστές σε ζεύγη αφού τα καταφέρνουν το ίδιο καλά σε σχέση με τους έμπειρους.

Kyungsub S. Choi et. al. [S23][**Exploring the underlying aspects of pair programming:The impact of personality**] στο πείραμά τους μελέτησαν την επίπτωση που έχει η προσωπικότητα στον προγραμματισμό ανά ζεύγη. 128 συνολικά από προπτυχιακούς και μεταπτυχιακούς φοιτητές έλαβαν μέρος στο πείραμα αφού υποβλήθηκαν στο τεστ προσωπικότητας MBTI. Με τα αποτελέσματα αυτού του τεστ χωρίστηκαν σε 3 ομάδες. Την εντελώς όμοια προσωπικότητα, των αντίθετων προσωπικοτήτων και αυτή των αναμεμειγμένων. Τα αποτελέσματα έδειξαν πως η ομάδα με τις αναμεμειγμένες προσωπικότητες είχε την μεγαλύτερη παραγωγικότητα κώδικα με αμέσως επόμενη αυτή των αντιθέτων προσωπικοτήτων.

Tanja Bipp et. al. [S24][**Pair programming in software development teams – An empirical study of its benefits**] διεξήγαγαν ένα πείραμα για να βρουν τα προτερήματα και τα μειονεκτήματα που έχει ο PP στην διδασκαλία φοιτητών πληροφορικής. Συνολικά 95 φοιτητές έλαβαν μέρος σε 2 ξεχωριστά πειράματα. Στο πρώτο πείραμα χωρίστηκαν σε 4 ομάδες των 6 ή 7 ατόμων με τις 2 ομάδες να δουλεύουν σαν PP. Στο δεύτερο πείραμα χωρίστηκαν σε 9 ισομελής ομάδες με τις 5 από αυτές να δουλεύουν σαν PP. Και στις 2 περιπτώσεις οι προγραμματιστές σε ζεύγη ενάλλασσαν τον συνεργάτη τους με άλλον σε τακτά χρονικά διαστήματα. Αυτό είχε σαν αποτέλεσμα οι φοιτητές των ομάδων να έχουν μια καλύτερη οπτική για όλο πρόβλημα συνολικά αλλά και να ανταλλάσουν γνώση με όλα τα μέλη της ομάδας. Τα προγράμματα των PP είναι ποιοτικότερα, λιγότερο σύνθετα και ευκολοδιάβαστα. Αυτό βοήθησε σημαντικά στον εντοπισμό διάφορων λαθών και την διόρθωσή τους. Το μόνο αρνητικό που εντόπισαν για τους PP είναι ότι υπάρχει μια μικρή μείωση της αποδοτικότητας σε σχέση με τους solo.

Kim Man Lui et. al. [S25][**Pair programming productivity: Novice–novice vs. expert–expert**] διεξήγαγαν μια ακολουθία πειραμάτων για να διαπιστώσουν την σχέση μεταξύ εμπειρίας και παραγωγικότητας στον προγραμματισμό ανά ζεύγη σε σχέση με τους solo. Στα πειράματα που έγιναν οι προγραμματιστές έλυναν ένα συγκεκριμένο πρόβλημα πληροφορικής ξανά και ξανά. Αυτή η μεθοδολογία «μεταμόρφωσε» τους αρχάριους προγραμματιστές σε έμπειρους, αφού δούλευαν συνεχώς στο ίδιο θέμα. Επίσης τα προγράμματα περνούσαν από συγκεκριμένα test cases έτσι ώστε η ποιότητα να θεωρηθεί σταθερή και όχι υποκειμενική στην ανθρώπινη κρίση. Οι PP είχαν μεγαλύτερη παραγωγικότητα σε πολύπλοκα προγραμματιστικά προβλήματα όσο το πρόβλημα τους ήταν άγνωστο. Αντίθετα όταν έλυσαν το πρόβλημα αρκετές φορές και απέκτησαν κάποια εμπειρία η παραγωγικότητά τους έπεσε χαμηλότερα σε σχέση με αυτή των solo.

Charlie McDowell et. al. [S26][**The Impact of Pair Programming on Student Performance,Perception and Persistence**] έκαναν ένα πείραμα για να εξετάσουν πως ο προγραμματισμός ανά ζεύγη επηρεάζει την απόδοση των φοιτητών στα αντίστοιχα μαθήματα και εάν είναι ένας από τους παράγοντες που οδηγούν τους φοιτητές να ακολουθήσουν κάποιον κλάδο της επιστήμης των υπολογιστών. Στο πείραμα έλαβαν μέρος 555 νεοεισερχόμενοι φοιτητές. Τα αποτελέσματα έδειξαν πως οι προγραμματιστές ανά ζεύγη είχαν μεγαλύτερη επιτυχία στο μάθημα και ότι ήταν περισσότερο πιθανό να ακολουθήσουν κάποιο κλάδο της επιστήμης των υπολογιστών σε σχέση με τους solo. Συνολικά οι PP είχαν μεγαλύτερη αυτοπεποίθηση για τις λύσεις που παρέδιδαν σε σχέση με τους solo όπως επίσης μεγαλύτερη απόλαυση και ικανοποίηση.

O Matthias Muller [S27][**Two controlled experiments concerning the comparison of pair programming to peer review**] έκανε ένα πείραμα και εξέτασε το κόστος ανάπτυξης που έχει ο προγραμματισμός ανά ζεύγη σε σχέση με τους solo προγραμματιστές όταν αυτοί έχουν την επιπλέον βοήθεια των reviews. Τα αποτελέσματα έδειξαν πως τα προγράμματα των PP και των solo προγραμματιστών έχουν το ίδιο κόστος ανάπτυξης αν υπάρχει η υποχρέωση να έχουν τα ίδια επίπεδα ορθότητας στον κώδικα. Στην περίπτωση που υπάρχουν διαφορετικά επίπεδα ορθότητας στον κώδικα οι PP παράγουν ποιοτικότερα προγράμματα σε σχέση με τους solo αλλά με μεγαλύτερο κόστος. Καταλήγοντας μας επισημαίνει πως πρέπει να γίνουν περισσότερες έρευνες που να καλύπτουν περισσότερους παραμέτρους όπως η ικανότητα, η παραγωγικότητα αλλά και ένα μεγαλύτερο μέγεθος δείγματος.

Monvorath Phongpaibul et. al. [S28][**A Replicate Empirical Comparison between Pair Development and Software Development with Inspection**] σε μία σειρά πειραμάτων (ακαδημαϊκά αλλά και στην βιομηχανία)μελέτησαν την επίπτωση που έχει η καταβαλλόμενη προσπάθεια και η ποιότητα του προγράμματος μεταξύ ομάδων προγραμματιστών ανά ζεύγη και ομάδων που χρησιμοποιούσαν την τεχνική

του Inspection για την ανάπτυξη λογισμικού. Για να συνδυάσουν την προσπάθεια με την ποιότητα χρησιμοποίησαν την μέθοδο CoSQ(Cost of Software Quality). Τα αποτελέσματα έδειξαν πως οι PP χρειάστηκαν να καταβάλουν μικρότερη προσπάθεια για να επιτύχουν ίδια η ανώτερη ποιότητα προγραμμάτων σε σχέση με τους προγραμματιστές της τεχνικής του Inspection. Επίσης το συνολικό κόστος ανάπτυξης μειώνεται για τους PP μέχρι και 22%. Στον βιομηχανικό τομέα όπου τα προβλήματα που αντιμετωπίζονται είναι πιο πολύπλοκα υπάρχει μια αύξηση της προσπάθειας κατά 4% αλλά μείωση των σφαλμάτων κατά 40% προς όφελος των PP. Στα πειράματα έλαβαν μέρος άνθρωποι παρόμοιων πολιτισμών και έτσι στάθηκε αδύνατο να μελετηθούν οι επιπτώσεις που θα είχαν οι πολιτιστικές διαφορές.

Η Laurie Williams [S29][**But, Isn't That Cheating?**] έκανε ένα από τα πρώτα πειράματα για τον προγραμματισμό ανά ζεύγη σε ακαδημαϊκό επίπεδο στις αρχές διάδοσης της τεχνικής. 20 αρχάριοι φοιτητές έλαβαν μέρος στο πείραμα. Σκοπός του πειράματος ήταν να δουλέψουν όλοι μαζί σε ένα προϊόν που αφορούσε μάθημα σχετικό με διαδικτυακό προγραμματισμό. 84% των φοιτητών συμφώνησαν ότι διασκέδασαν περισσότερο κάνοντας την εργασία σαν ζεύγη προγραμματιστών. 84% των φοιτητών πιστεύουν πως έμαθαν την γλώσσα ASP γρηγορότερα και καλύτερα και το 95% ένιωθαν μεγαλύτερη αυτοπεποίθηση. Το ότι δούλευαν σε ζευγάρια τους έδωσε ώθηση ώστε να δουλέψουν πιο έντονα, πιο αποδοτικά και να μοιραστούν γνώση.

Lynda Thomas et. al. [S30][**Code Warriors and Code-a-Phobes: A Study in Attitude and Pair Programming**] ερεύνησαν το πώς η αυτογνωσία ενός προγραμματιστή επιδρά στον προγραμματισμό ανά ζεύγη. Στο πείραμα έλαβαν μέρος 60 νεοεισερχόμενοι φοιτητές οι οποίοι μέσω ενός ερωτηματολογίου αξιολόγησαν την αντίληψη που έχουν πάνω στον προγραμματισμό. Η αξιολόγηση αυτή τους χώριζε σε μια κλίμακα 9 κατηγοριών. Στη πρώτη κατηγορία μπήκαν οι φοιτητές που δεν είχαν κανένα απολύτως πρόβλημα στην συγγραφή κώδικα, αγαπούσαν τον προγραμματισμό και πίστευαν ότι δεν θα έχουν καμία δυσκολία με το μάθημα(Code-Warriors). Στην τελευταία κατηγορία μπήκαν οι φοιτητές που είχαν δυσκολία να λύσουν προγραμματιστικά ακόμη και τα πιο απλά προβλήματα, δεν θεωρούσαν τους εαυτούς τους καλούς προγραμματιστές και δεν τους άρεσε καθόλου ο προγραμματισμός(Code-Phobe). Τα αποτελέσματα της έρευνας μας δείχνουν πως γενικά στους φοιτητές άρεσε ο προγραμματισμός ανά ζεύγη και ότι τους βοήθησε να επιτύχουν καλύτερες λύσεις στα προβλήματα. Οι φοιτητές με την λιγότερη αυτοεκτίμηση απόλαυσαν την διαδικασία περισσότερο. Αντίθετα αυτοί που βρέθηκαν στην πρώτη κλίμακα(code warriors) ήταν αυτοί που δεν τους άρεσε τόσο ο προγραμματισμός ανά ζεύγη. Επίσης τους code warriors τείνει να τους αρέσει ακόμη λιγότερο ο προγραμματισμός ανά ζεύγη όταν τους βάλουμε σε ομάδα με τα μέλη της τελευταίας κλίμακας(code phobe). Γενικά οι φοιτητές αποδίδουν τα μέγιστα όταν οργανώνονται σε ζεύγη ίδιων επιπέδων.

Nachiappan Nagappan et. al. [S31][**Improving the CS1 Experience with Pair Programming**] στο πείραμα που διεξήγαγαν μελέτησαν την αποτελεσματικότητα του προγραμματισμού ανά ζεύγη σε ένα εισαγωγικό μάθημα πληροφορικής στο πανεπιστήμιο NCSU. Στο 1 ακαδημαϊκό έτος που κράτησε το πείραμα έλαβαν συνολικά μέρος περίπου 495 φοιτητές et. al. προσπάθησε να απαντήσει στις εξής ερωτήσεις:

1. Μεγαλύτερο ποσοστό προγραμματιστών ανά ζεύγη θα περάσει το μάθημα με βαθμό C η μεγαλύτερο σε σχέση με τους solo προγραμματιστές
2. Μεγαλύτερο ποσοστό προγραμματιστών ανά ζεύγη θα έχει καλύτερο τελικό βαθμό στο μάθημα σε σχέση με τους solo προγραμματιστές.
3. Οι προγραμματιστές ανά ζεύγη θα έχουν καλύτερους βαθμούς και απόδοση στις εργαστηριακές ασκήσεις σε σχέση με τους solo προγραμματιστές.
4. Οι καθηγητές/επιβλέποντες των προγραμματιστών ανά ζεύγη θα έχουν λιγότερο φόρτο εργασίας αναφορικά με βαθμολόγηση, λιγότερες ερωτήσεις προς απάντηση, λιγότερη προσπάθεια διδασκαλίας σε σχέση με τους καθηγητές των solo προγραμματιστών.
5. Οι προγραμματιστές σε ζεύγη θα αποκτήσουν μια πιο θετική στάση απέναντι στις συνεργασιακές τακτικές προγραμματισμού σε σχέση με τους solo προγραμματιστές.

Πολλές φορές οι νέοι προγραμματιστές έχουν αρκετές απλές απορίες(πχ συντακτικό) και χρειάζεται να περιμένουν αρκετά τον καθηγητή ώστε να τους λυθούν, κάτι που οδηγεί σε εκνευρισμό αφού δεν μπορούν να ολοκληρώσουν την εργασία τους. Αντίθετα οι προγραμματιστές ανά ζεύγη έχουν συνεχή επικοινωνία μεταξύ τους ανταλλάσουν γνώση και έτσι τα μικρά αυτά προβλήματα λύνονται εύκολα. Οι καθηγητές των PP ασχολούνται με ερωτήσεις πιο προηγμένου επιπέδου (πχ διαφορετικοί τρόποι επίλυσης, εξειδικευμένοι

αλγόριθμοι) και όχι με τα «βασικά» κάτι που τους κάνει να νιώθουν πιο εποικοδομητικούς και να έχουν μεγαλύτερη ικανοποίηση για την δουλειά τους. Ο προγραμματισμός ανά ζεύγη βοηθά καλύτερα στην απόδοση των περισσότερων φοιτητών και τους κάνει να έχουν μια καλύτερη στάση απέναντι στις συνεργασιακές τακτικές προγραμματισμού. Η έρευνα κατάφερε να αποδείξει πως οι φοιτητές σε ζεύγη έχουν ίδιους και σε πολλές φορές καλύτερους βαθμούς στις εργαστηριακές ασκήσεις αλλά απέτυχε να δείξει πως οι προγραμματιστές ανά ζεύγη έχουν καλύτερους τελικούς βαθμούς στο μάθημα.

Emilia Mendes et. al. [S32][**Investigating Pair-Programming in a 2nd-year Software Development and Design Computer Science Course**] έκαναν ένα πείραμα για να ενισχύσουν το κύρος των ερευνών που υποστήριζαν πως ο προγραμματισμός ανά ζεύγη βοηθά την εκπαιδευτική διαδικασία και την ευχαρίστηση των φοιτητών. Στο πείραμα αυτό που έγινε το 2004 έλαβαν μέρος 300 φοιτητές οι οποίοι χωρίστηκαν σε προγραμματιστές ανά ζεύγη αλλά και σε solo προγραμματιστές. Τα αποτελέσματα της έρευνας τους έδειξαν πως τα προγράμματα των PP ήταν περισσότερο ποιοτικά, είχαν καλύτερους βαθμούς (67%) στην τελική εξέταση και μεγαλύτερα ποσοστά επιτυχίας(91%) σε σχέση με τους solo. Στους περισσότερους φοιτητές άρεσε ο PP και δεν είχαν προβλήματα που σχετιζόταν με το φύλο, την εθνικότητα και την ικανότητα επικοινωνίας.

Neha Katira et. al. [S33][**On Understanding Compatibility of Student Pair Programmers**] διεξήγαγαν ένα πείραμα το οποίο μελετούσε τους πιθανούς λόγους που επηρέαζαν την συμβατότητα των προγραμματιστών ανά ζεύγη. Στο πείραμα αυτό έλαβαν μέρος συνολικά 564. Οι φοιτητές αρχικά διαχωριστήκαν ανάλογα με το επίπεδο τους σε πρωτοετής/νεοεισερχόμενους φοιτητές(CS1), σε μη πτυχιούχους μηχανικούς λογισμικού(SE) και σε πτυχιούχους φοιτητές εξειδικευμένους στον object-oriented προγραμματισμό(OO). Οι φοιτητές υποβλήθηκαν σε τεστ MBTI για τον προσδιορισμό προσωπικότητας και της αυτοεκτίμησης αλλά και σε τεστ για να εξακριβωθεί η προγραμματιστική τους ικανότητα. Οι φοιτητές χωρίστηκαν σε ομάδες με solo προγραμματιστές και προγραμματιστές ανά ζεύγη που είχαν όλες τις μίξεις από προσωπικότητα, ικανότητα και αυτοεκτίμηση. Τα αποτελέσματα έδειξαν πως το 90% των ζευγών είναι συμβατά εάν επιλέξουμε τυχαία μέλη να τα απαρτίζουν. Οι φοιτητές όμως με διαφορετική προσωπικότητα έχουν καλύτερη συνεργασία μεταξύ τους χωρίς αυτό να επηρεάζεται από την αυτοεκτίμηση. Επίσης οι φοιτητές προτιμούν να είναι σε ζεύγη όπου και οι δυο θα έχουν τα ίδια επίπεδα ικανότητας.

Linda L. Werner et. al. [S34][**Pair-Programming Helps Female Computer Science Students**] έκαναν μία έρευνα για τις επιπτώσεις που έχει ο προγραμματισμός ανά ζεύγη και τα οφέλη που έχουν από αυτόν οι γυναίκες. Τα δεδομένα που συνέλλεξαν ήταν από 2 ξεχωριστά πειράματα που είχαν κάνει οι συγγραφείς σε προηγούμενο χρόνο και συμμετείχαν 141 και 24 γυναίκες αντίστοιχα. Τα αποτελέσματα έδειξαν πως ο PP βοήθησε σημαντικά τις γυναίκες να κατανοήσουν πως η επιστήμη των υπολογιστών δεν είναι μια ανταγωνιστική και κοινωνικά απομονωτική διαδικασία κάτι το οποίο πίστευαν. Επίσης τους ενθάρρυνε να επιλέξουν κάποια ειδικότητα σε σχέση με τους υπολογιστές στο μέλλον.

Η Laurie Williams et. al.[S35][**Strengthening the Case for Pair Programming**] σε ένα από τα πρώτα πειράματα που συνέβησαν στον ακαδημαϊκό χώρο μελέτησαν τις επιπτώσεις που έχει ο προγραμματισμός ανά ζεύγη στους φοιτητές. Το πείραμα έγινε στο πανεπιστήμιο της UTAH και έλαβαν μέρος 41 φοιτητές. Από τα αποτελέσματα βρέθηκε πως τα προγράμματα των PP είχαν μεγαλύτερη επιτυχία στο πέρασμα των Test Cases απ' ότι τα προγράμματα των solo. Επίσης βρέθηκε πως ολοκλήρωσαν τις εργασίες τους από 40% έως 50% γρηγορότερα σε σχέση με τους solo. Τα προγράμματα τους ήταν ποιοτικότερα και με λιγότερα σφάλματα. Οι PP έλυναν τις περισσότερες τεχνικές απορίες μεταξύ τους και δεν χρειαζόταν να περιμένουν οδηγίες από τον καθηγητή. Οι PP δεν σπαταλούσαν χρόνο σερφάροντας, βλέποντας τα mail τους ή κάνοντας άλλες εργασίες πέρα από το μάθημα αφού αισθανόταν «πίεση» από το ζεύγος (pair pressure effect) του για την ολοκλήρωση της εργασίας. Το 90% των φοιτητών δήλωσαν πως διασκέδασαν και τους άρεσε περισσότερο ο προγραμματισμός ανά ζεύγη σε σχέση με τον solo.

Ο John T. Nosek [S36][**The Case for Collaborative Programming**] έκανε ένα πείραμα για να μελετήσει τις επιπτώσεις του προγραμματισμού ανά ζεύγη στον βιομηχανικό τομέα. Στο πείραμα αυτό έλαβαν μέρος 15 επαγγελματίες προγραμματιστές οι οποίοι χωρίστηκαν σε ζεύγη και σε solo προγραμματιστές. Σκοπός τους ήταν να αναπτύξουν ένα script διαδικασιών ρουτίνας σε μια βάση δεδομένων. Το πείραμα προσπάθησε να απαντήσει στο αν οι PP θα παρήγαγαν πιο ευκολοδιάβαστες και πιο λειτουργικές

λύσεις, αν θα είχαν μεγαλύτερη αυτοπεποίθηση και αν θα το απολάμβαναν περισσότερο, αν θα σπαταλούσαν λιγότερο χρόνο σε σχέση με τους solo και αν οι πιο έμπειροι προγραμματιστές θα τα πήγαιναν καλύτερα από τους λιγότερο έμπειρους. Ο ερευνητής κατάφερε να αποδείξει πως οι προγραμματιστές σε ζεύγη είχαν λύσεις λειτουργικότερες και πιο ευκολοδιάβαστες με πιο χαρούμενους και με μεγαλύτερη αυτοπεποίθηση δημιουργούς. Αποδείχτηκε πως οι προγραμματιστές σε ζεύγη χρειάστηκαν 40% λιγότερο χρόνο για να ολοκληρώσουν μια εργασία και βρέθηκε πως οι εμπειρότεροι προγραμματιστές τα πήγαν καλύτερα είτε σαν ζεύγη είτε σαν solo.

Neha Katira et. al. [S37][**Towards Increasing the Compatibility of Student PairProgrammers**] έκαναν ένα πείραμα για την συμβατότητα μεταξύ των ζευγών του προγραμματισμού ανά ζεύγη. Σκοπός τους ήταν να κατανοήσουν αλλά και να προβλέψουν τους παράγοντες που θα όριζαν μία επιτυχημένη σύζευξη. Στο πείραμα έλαβαν μέρος 361 φοιτητές πληροφορικής του πανεπιστημίου NCSU. Κατέληξαν ότι αυτό που παίζει σημαντικό ρόλο στην συμβατότητα των ζευγαριών είναι η αντίληψη της ικανότητας που έχουν οι προγραμματιστές για τον εαυτό του αλλά και για τον συνεργάτη τους. Γενικά δυο φοιτητές οι οποίοι πιστεύουν ότι έχουν ίδιου επιπέδου ικανότητες αποδίδουν καλύτερα. Η προσωπικότητα όπως και η αυτοεκτίμηση δεν είναι καθοριστικοί παράγοντες για την συμβατότητα του ζευγαριού. Τέλος ζεύγη διαφορετικών φύλων είναι πιθανότερο να μην είναι συμβατά.

Laurie A. Williams et. al. [S38][**The Effects of "Pair-Pressure" and "Pair-Learning" on Software Engineering Education**] έκανε ένα πείραμα για να μελετήσει την επίπτωση του προγραμματισμού ανά ζεύγη στην εκπαιδευτική διαδικασία. Στο πείραμα αυτό έλαβαν μέρος 20 φοιτητές του πανεπιστημίου της Utah στα πλαίσια ενός μαθήματος διαδικτυακού προγραμματισμού. Τα αποτελέσματα έδειξαν πως η «πίεση» που ασκούσε το ένα ζεύγος στο άλλο ήταν εξαιρετικά εποικοδομητική. Κανείς δεν ήθελε να απογοητεύσει τον συνεργάτη του. Οι PP πέτυχαν καλύτερους βαθμούς με μέσο όρο 98%. Επίσης ένιωθαν μεγαλύτερη αυτοπεποίθηση και είχαν καλύτερη ποιότητα στον κώδικά τους. Στο 84% των συμμετεχόντων άρεσε ο PP και νιώθουν πως τους βοήθησε σημαντικά στο να μάθουν καλύτερα τις γλώσσες ανάπτυξης διαδικτυακών προγραμμάτων. Το διδακτικό προσωπικό επίσης είχε λιγότερο φόρτο εργασίας και λιγότερες απορίες προς επίλυση.

Gerardo Canfora et. al. [S39][**Working in pairs as a means for design knowledge building: an empirical study**] έκαναν ένα πείραμα που επικεντρώθηκε στο πως γίνεται η μεταφορά της γνώσης στον προγραμματισμό ανά ζεύγη κατά την διάρκεια του σχεδιασμού του προγράμματος. Κατά την σχεδιαστική φάση οι συμμετέχοντες χρειάζεται να έχουν αρκετή παρατηρητικότητα, να καταστρώνουν προσωπικά μοντέλα και γενικά υπάρχει μια σιωπηρή γνώση η οποία δεν μπορεί να αποτυπωθεί παρά μόνο με τον διάλογο. Στο πείραμά τους προσπάθησαν να απαντήσουν στην εξής ερώτηση: Δεν υπάρχουν σημαντικές διαφορές ανάμεσα στην γνώση που χτίζεται από solo προγραμματιστές και σε αυτή των ζευγών. Στο πείραμα έλαβαν μέρος 45 φοιτητές. Τα αποτελέσματα έδειξαν πως τα ζεύγη έχουν μια πιο γρήγορη, μεγάλη και σταθερή ανάπτυξη γνώσης. Οι ερευνητές επισημαίνουν πως τα προβλήματα που αντιμετώπισαν οι φοιτητές ήταν σχετικά εύκολα και δεν έχουν καμία σχέση με την βιομηχανία.

Lech Madeyski et. al. [S40][**Impact of pair programming on thoroughness and fault detection effectiveness of unit test suites**] έκανε ένα πείραμα για να εξετάσει την αποδοτικότητα του προγραμματισμού ανά ζεύγη στην επιμέλεια και την εύρεση λαθών στα unit test suites. Χρησιμοποιώντας τις τεχνικές του branch coverage(bc) και του mutation score indicator(msi) εξέτασε προγραμματιστές ανά ζεύγη και solo προγραμματιστές σε ακαδημαϊκό περιβάλλον. Τα αποτελέσματα έδειξαν πως με τις δυο τεχνικές που χρησιμοποιήθηκαν δεν υπήρχε κάποια αξιοσημείωτη διαφορά στην απόδοση. Αποφασίστηκε να γίνει μια πιο ενδελεχής μελέτη από προσεκτικά επιλεγμένες εργασίες. Τα αποτελέσματα ήταν τα ίδια. Καταλήγουν πως τα συμπεράσματα τους δεν μπορούν να γενικοποιηθούν και πρέπει να γίνουν περισσότερες μελέτες σε μεγαλύτερα και πολυπλοκότερα προβλήματα για την εξαγωγή πιο ασφαλών συμπερασμάτων.

Jerzy Nawrocki et. al. [S41][**Experimental Evaluation of Pair Programming**] έκαναν ένα πείραμα σε ακαδημαϊκό επίπεδο με τεταρτοετής φοιτητές. 21 φοιτητές έλαβαν μέρος στο πείραμα οι οποίοι χωρίστηκαν σε 3 ομάδες. Στην πρώτη ομάδα είχαμε φοιτητές που χωρίστηκαν σε 5 ζευγάρια, στην δεύτερη ομάδα είχαμε 5 solo προγραμματιστές οι οποίοι χρησιμοποιούσαν τακτικές του extreme programming (XP) και στην τρίτη ομάδα solo προγραμματιστές που χρησιμοποίησαν τακτικές Personal Software Process(PSP).

Τους ζητήθηκε να γραφτούν τέσσερα προγράμματα έτσι ώστε να μετρηθούν LOC και χρόνος ολοκλήρωσης. Τα αποτελέσματα έδειξαν πως solo και PP χρειάστηκαν περίπου τον ίδιο χρόνο για την ολοκλήρωση των εργασιών τους. Επίσης βρέθηκε πως οι PP χρειάστηκαν λιγότερες φορές να επανεξετάσουν τα προγράμματα τους και να τα ξανά υποβάλλουν λόγω των λιγότερων λαθών. Καταλήγοντας μας λένε πως ο PP είναι αποδοτικός αλλά όχι στα επίπεδα που παρουσιάστηκαν από προηγούμενες έρευνες και χρειάζονται περισσότερα και μεγαλύτερα πειράματα από το δικό τους για εξαγωγή ασφαλών συμπερασμάτων.

Maree Mujeeb-u-Rehman et. al. [S42][**Heterogeneous and homogenous pairs in pair programming: an empirical analysis**] διεξήγαγαν ένα πείραμα για τις επιπτώσεις που έχει η προσωπικότητα στον προγραμματισμό ανά ζεύγη. Πιο συγκεκριμένα ποιος είναι ο αντίκτυπος των ομογενών και ετερογενών ζευγαριών στην απόδοση, την ακρίβεια και την ποιότητα, τον αριθμό των λαθών στον κώδικα, την έμπνευση(brainstorming) αλλά και την ενδεχόμενη δημιουργικότητα. Οι φοιτητές που συμμετείχαν στο πείραμα χωρίστηκαν σε 2 ζεύγη ανδρών, σε 2 γυναικών(ομογενή ζεύγη) και σε 2 μεικτά ζευγάρια(ετερογενή ζεύγη). Βρέθηκε πως τα ετερογενή ζευγάρια είχαν καλύτερη απόδοση, λιγότερα λάθη, καλύτερη έμπνευση και δημιουργικότητα. Επίσης τα ομογενή ζεύγη των γυναικών ήρθαν στην $2^{η}$ θέση της αποδοτικότητας και τελευταία αυτά των ομογενών ανδρών.

Raymund Sison [S43][**Investigating Pair Programming in a Software Engineering Course in an Asian Setting**] πραγματοποίησε ένα πείραμα για να διαπιστώσει την επίπτωση που έχει ο προγραμματισμός ανά ζεύγη στους τριτοετής φοιτητές πληροφορικής σε ένα πανεπιστήμιο των Φιλιππινών. Στο πείραμα αυτό έλαβαν μέρος 24 φοιτητές οι οποίοι χωρίστηκαν σε προγραμματιστές ανά ζεύγη και solo προγραμματιστές. Οι ερευνητές προσπάθησαν να απαντήσουν στο αν υπάρχουν διαφορές στην συχνότητα/πυκνότητα των λαθών κώδικα μεταξύ PP και solo προγραμματιστών αλλά και στην παραγωγικότητα(LOC/hour) μεταξύ τους. Τα αποτελέσματα έδειξαν πως οι PP είχαν σημαντικά μικρότερη συχνότητα λαθών σε σχέση με τους solo προγραμματιστές αλλά τα επίπεδα παραγωγικότητας ήταν όμοια και στις 2 ομάδες. Επίσης παρατηρήθηκαν και άλλα θετικά στοιχεία υπέρ του PP όπως η μετάδοση γνώσης, κοινωνικές αλληλεπιδράσεις, δεν υπήρχε χάσιμο χρόνου σε άσκοπες και άσχετες με το αντικείμενο εργασίες, μεγαλύτερη ευχαρίστηση και λιγότερη κούραση. Επιπρόσθετα τα προγράμματα των PP ήταν πιο ευκολοδιάβαστα και συμπαγή με τους προγραμματιστές να βρίσκουν γρηγορότερα τα λάθη στον κώδικά τους.

Charlie McDowell et. al. [S44][**The Effects of Pair-Programming on Performance in an Introductory Programming Course**] διεκπεραίωσαν ένα πείραμα για να μελετήσουν την επίπτωση του προγραμματισμού ανά ζεύγη στις επιδόσεις των φοιτητών ενός εισαγωγικού μαθήματος προγραμματισμού. Στο πείραμα αυτό έλαβαν μέρος 313 φοιτητές χωρισμένοι σε ζευγάρια ή σε solo προγραμματιστές. Τα αποτελέσματα εμφάνισαν πως οι PP είχαν καλύτερους βαθμούς στις παραδοτέες εργασίες κατά 86% έναντι 77% των solo έχοντας αυξημένη ποιότητα στον κώδικά τους. Στους τελικούς βαθμούς των μαθημάτων δεν βρέθηκαν αξιοσημείωτες διαφορές ανάμεσα στις 2 ομάδες αλλά αποτυπώθηκε πως το 92% των PP κατάφερε να περάσει το μάθημα και μόνο το 77% των solo είχε την ίδια επιτυχία.

Brian Hanks [S45][**Problems Encountered by Novice Pair Programmers**] έκανε ένα πείραμα σε ακαδημαϊκό επίπεδο για να μπορέσει να εξακριβώσει εάν οι προγραμματιστές ανά ζεύγη κάνουν διαφορετικά λάθη από αυτά που κάνουν οι solo. Τα αποτελέσματα έδειξαν πως και οι 2 ομάδες κάνουν παρόμοια λάθη. Οι PP όμως φαίνεται ότι μπορούν να επιλύσουν μόνοι τους αυτά τα λάθη χωρίς να χρειάζονται περεταίρω βοήθεια από το εκπαιδευτικό προσωπικό παρά μόνο για τα πιο ουσιώδη και δύσκολα θέματα. Αυτό έχει ως αποτέλεσμα την αύξηση της αυτοπεποίθησης και της αυτοεκτίμησης.

Lech Madeyski [S46][**The Impact of Pair Programming and Test-Driven Development on Package Dependencies in Object-Oriented Design — An Experiment**] έκανε ένα πείραμα για να εξακριβώσει την επίπτωση που θα έχει ο προγραμματισμός ανά ζεύγη και η Test-Driven τεχνική ανάπτυξης πάνω στην ποιότητα. Σαν ένδειξη ποιότητας ο ερευνητής χρησιμοποιεί τους δείκτες ποιότητας σχεδιασμού σε επίπεδο πακέτου*(«package dependenciesin an object-oriented design»* - όπως χαρακτηριστικά αναφέρει). Στο πείραμα αυτό έλαβαν μέρος 188 φοιτητές με τον μελετητή να προσπαθεί να εξομοιώσει παρόμοιες καταστάσεις που θα αντιμετωπιστούν στην βιομηχανία. Οι φοιτητές χωρίστηκαν σε ζεύγη και σε solo προγραμματιστές και θα χρησιμοποιούσαν την TDD τεχνική ή την Test last παραδοσιακή τεχνική. Τα

αποτελέσματα της έρευνας έδειξαν πως η ποιότητα δεν επηρεάστηκε σημαντικά από την τεχνική ανάπτυξης και ότι αυτή περιορίζεται στο επίπεδο της ακαδημαϊκής τάξης.

Lucas Layman [S47][**Changing Students' Perceptions: An Analysis of the Supplementary Benefits of Collaborative Software Development**] διενέργησε ένα πείραμα για να μελετήσει τον αντίκτυπο που έχει η προσωπικότητα στον προγραμματισμό ανά ζεύγη. Το πείραμα αυτό έγινε στο πανεπιστήμιο NCSU και έλαβαν μέρος συνολικά 78 φοιτητές. Ο ερευνητής για να βοηθηθεί στην έρευνά του χρησιμοποίησε την τεχνική MBTI για την αξιολόγηση της προσωπικότητας, και την τεχνική Felder - Silverman για να διαπιστώσει το μαθησιακό στυλ που ακολουθεί ο κάθε φοιτητής. Οι φοιτητές μετά την εμπειρία του προγραμματισμού ανά ζεύγη φαίνεται ότι προτιμούν αυτή την μέθοδο αφού τους βοηθά να ολοκληρώσουν τις εργασίες τους σε λιγότερο χρόνο και τους κάνει να νιώθουν πιο οργανωμένοι. Αυτή η θετική στάση τους βοηθά να αποκτήσουν γνώση η οποία θα διατηρηθεί και στο μέλλον. Φοιτητές με «διαισθητικό» στυλ μάθησης και εξωστρεφή χαρακτήρα φαίνεται ότι προτιμούν τον PP περισσότερο. Αντίθετα «στοχαστικοί» φοιτητές, εσωστρεφής που ήταν δυνατοί προγραμματιστές είναι αυτοί που δεν τους αρέσει ο PP. Ένα πρόβλημα που αποτελεί παράλληλα και σημαντικό λόγο για τον οποίο στους φοιτητές δεν άρεσε ο PP είναι η δυσκολία εύρεσης κοινού χρονοδιαγράμματος με τον συνεργάτη τους ώστε να μπορέσουν να δουλέψουν μαζί. Κλείνοντας μας λέει πως η προσωπικότητα αλλά και το στυλ μάθησης δεν έχουν σημαντική επίπτωση στην συνεργασία που αναπτύσουν οι PP.

Jerzy R. Nawrocki et. al. [S48][**Pair Programming vs. Side-by-Side Programming**] έκαναν ένα πείραμα για να συγκρίνουν τον κλασικό προγραμματισμό ανά ζεύγη που περιγράφεται από την βιβλιογραφία του XP και τον side by side προγραμματισμό(2 υπολογιστές ανά ζευγάρι). Στο πείραμα έλαβαν μέρος 30 μεταπτυχιακοί φοιτητές και χωρίστηκαν σε PP, σε side by side και solo προγραμματιστές. Τα αποτελέσματα της έρευνας έδειξαν πως οι SbS προγραμματιστές παρήγαγαν κώδικα που περνούσε τα τεστ γρηγορότερα από τους PP. Επίσης PP και SbS ήταν γρηγορότεροι από τους solo προγραμματιστές. Παρατηρήθηκε πως η γνώση για τον παραγόμενο κώδικα διαδίδεται με πιο αργό ρυθμό στους SbS απ' ότι στους προγραμματιστές ανά ζεύγη κάτι που φαίνεται από την 20% παραπάνω προσπάθεια που κατέβαλε η μεριά των SbS. Καταλήγουν πως ο SbS προγραμματισμός είναι μια ενδιαφέρουσα εναλλακτική του παραδοσιακού προγραμματισμού ανά ζεύγη.

Matthias M. Muller [S49][**A preliminary study on the impact of a pair design phase on pair programming and solo programming**] έκανε ένα πείραμα για να εξετάσει το κόστος που έχει ο σχεδιασμός σε ζεύγη στον προγραμματισμό ανά ζεύγη και στον solo προγραμματισμό. 18 φοιτητές πληροφορικής έλαβαν μέρος στο πείραμα. Το πείραμα χωρίζεται σε 3 στάδια, ένα προκαταρκτικό τεστ(για να εντοπιστεί η ικανότητα των φοιτητών), τον σχεδιασμό και την υλοποίηση. Οι φοιτητές κατά την διάρκεια του σχεδιασμού χωρίστηκαν σε ζευγάρια. Τα ζευγάρια αυτά διασπάστηκαν την περίοδο της συγγραφής κώδικα και έγιναν είτε solo προγραμματιστές είτε προγραμματιστές ανά ζεύγη. Τα αποτελέσματα έδειξαν πως όταν υπάρχουν ίδια επίπεδα ορθότητας(correction level) στα προγράμματα τότε το κόστος είναι ίδιο για τους solo αλλά και για τους προγραμματιστές ανά ζεύγη. Επίσης τα αποτελέσματα έδειξαν πως οι solo και οι προγραμματιστές σε ζεύγη κάνουν ίδιο αριθμό λαθών αλλά διαφορετικά μεταξύ τους λάθη. Ο συγγραφέας όμως επιφυλάσσεται λόγο του μικρού δείγματος . Καταλήγει πως ο σχεδιασμός σε ζεύγη θα μπορούσε να είναι μια εναλλακτική του προγραμματισμού ανά ζεύγη.

4.2 Case/field studies – surveys

Laurie Williams et. al. [S50][**An Initial Exploration of the Relationship Between Pair Programming and Brooks' Law**] με την έρευνά τους βρήκαν την σχέση μεταξύ του PP και του νόμου του Brooke. Ο νόμος του Brooke μας λέει το εξής:«βάζοντας περισσότερο ανθρώπινο δυναμικό σε ένα καθυστερημένο χρονικά προγραμματιστικό πρότζεκτ το κάνει ακόμη πιο καθυστερημένο». Ο χρόνος και το κόστος που χρειάζεται για την εκπαίδευση, την αφομοίωση και την αλληλεπικοινωνία των νέων μελλών υπερκαλύπτει την αποδοτικότητα τους βραχυπρόθεσμα. Πολλοί οργανισμοί σε ανεπίσημες αναφορές έλεγαν πως αν οι νέοι εργαζόμενοι μπουν σε καθεστώς PP τότε αντισταθμίζεται θετικά η παραγωγικότητα σε σχέση με το κόστος και τον χρόνο που αναφέρονται στο νόμο του Brooke. Δημιούργησαν 2 δημοσκοπήσεις και τις έστειλαν σε επαγγελματίες προγραμματιστές που δούλευαν σε αληθινές συνθήκες. Έλαβαν και ανέλυσαν 30 απαντήσεις χρησιμοποιώντας την μαθηματική μέθοδο του Stutzke. Κατέληξαν στο συμπέρασμα πως εάν μια

ομάδα χρειάζεται άμεσα περισσότερο ανθρώπινο δυναμικό αυτό θα πρέπει να αποκτηθεί μέσα από τις τεχνικές του PP. Ταυτόχρονα θα πρέπει να γίνεται εναλλαγή ζευγαριών στα καινούρια μέλη για να μπορέσουν να εκπαιδευτούν καλύτερα αλλά και για να αποκτήσουν μια πιο γενική άποψη για το πρότζεκτ.

Sami Pietinen et. al. [S51][**Productivity of Pair Programming in a Distributed Environment – Results from Two Controlled Case Studies**] ερεύνησαν την πιθανότητα αύξησης παραγωγικότητας του προγραμματισμού ανά ζεύγη σε σχέση με τον solo προγραμματισμό. Καταλήγουν στο συμπέρασμα ότι ο PP αυξάνει την αυτοπεποίθηση στην δουλειά, την επικοινωνία την μεταβίβαση γνώσης αλλά και την ευχαρίστηση στην εργασία. Όλα αυτά τα οφέλη έρχονται όμως με ένα κόστος. Αυτό είναι το κόστος της προσπάθειας. Χρειάζεται 13% περισσότερη προσπάθεια από τους solo προγραμματιστές για να ολοκληρώσουν ένα έργο σε συγκεκριμένο dead line. Επίσης βρήκαν πως ο PP δεν είναι τόσο αποδοτικός όταν χρησιμοποιείται συνεχώς σε ένα πρότζεκτ. Ο PP είναι εξαιρετικά αποδοτικός όταν χρησιμοποιείται παράλληλα με άλλες τεχνικές XP όπως τα Peer Reviews.

Sallyann Bryant et. al. [S52][**The Collaborative Nature of Pair Programming**] μελέτησαν την ομιλία μεταξύ των προγραμματιστών σε ζεύγη σαν μια ένδειξη συνεργασίας. 36 ζευγάρια επαγγελματιών προγραμματιστών μελετήθηκαν και αναλύθηκαν για την επικοινωνία που είχαν μέσα σε μια εβδομάδα. Αφού ηχογράφησαν τις συνομιλίες τις ανέλυσαν με συγκεκριμένη μέθοδο. Τα αποτελέσματά τους έδειξαν πως κάθε ζευγάρι είχε περισσότερες από 250 λεκτικές επικοινωνίες μέσα σε μία ώρα. Υπήρχαν πολύ μικρές περιόδους σιωπής. Ακόμη και όταν τέσταραν κάποιο αντικείμενο δεν έχαναν την ευκαιρία για προσωπική κουβεντούλα. Ο περισσότερος χρόνος σπαταλήθηκε στην κατανόηση του προβλήματος, ακολουθεί ο χρόνος συγγραφής κώδικα, μετά η δοκιμή του και λιγότερο χρόνο χρειάστηκαν για τα σχόλια στον κώδικα και στον σχολιασμό του IDE. Μεγαλύτερο ποσοστό συνεργασίας παρατηρήθηκε στην διάρκεια του refactoring και στην συγγραφή νέου κώδικα με ποσοστό πάνω από το 80%. Επίσης παρατηρήθηκε μια μικρή διαφορά στην συμβολή(για την ολοκλήρωση της εργασίας) του driver σε σχέση με τον navigator.

Andrew Begel et. al. [S53][**Pair Programming: What's in it for Me?**] έκαναν μια έρευνα για την χρήση και την αποδοχή που έχει ο προγραμματισμός ανά ζεύγη μεταξύ των εργαζομένων της Microsoft. Έστειλαν ερωτηματολόγια σε τυχαίο δείγμα εργαζομένων του τεχνικού τομέα και έλαβαν 487 έγκυρες απαντήσεις. Τα αποτελέσματα έδειξαν πως το 64% των εργαζομένων πιστεύουν πως ο PP δουλεύει αρκετά καλά γι αυτούς. Όσο όμως ανεβαίνουμε στην ιεραρχία των εργαζομένων το ποσοστό αυτό μειώνεται. Επίσης μόνο το 25% πιστεύει ότι δεν υπάρχει μείωση στον χρόνο ολοκλήρωσης μιας εργασίας. Το 65% νιώθει ότι υπάρχει καλύτερη ποιότητα κώδικα και λιγότερα bugs. Παρόλα τα καλά όμως υπάρχουν και κάποια προβλήματα. Έχουμε αύξηση του κόστους λόγω των περισσότερων προγραμματιστών που χρειάζονται, πολλές φορές υπάρχουν προβλήματα λόγω των διαφορετικών ωράρια των εργαζομένων όπως επίσης και ασυμβατότητα μεταξύ των ζευγών λόγω προσωπικότητας. Καταλήγοντας μας λένε πως τα αποτελέσματα δεν πρέπει να γενικοποιηθούν και πρέπει να υπάρξουν και άλλες έρευνες σε διαφορετικές εταιρείες ανάπτυξης λογισμικού.

Jeffrey C. Carver et. al. [S54] [**Increased Retention of Early Computer Science and Software Engineering Students using Pair Programming**] στην μελέτη που έκαναν θέλησαν να αποδείξουν την σχέση μεταξύ του PP και την θέληση των φοιτητών να ακολουθήσουν την ειδικότητα του μηχανικού υπολογιστών, του μηχανικού προγραμμάτων και της επιστήμης των υπολογιστών. Κατά την διάρκεια της μελέτης παράλληλα θα παρατηρούσαν και τους παράγοντες που θα οδηγούσαν στην επιλογή της ειδικότητας. Τα αποτελέσματα έδειξαν πως οι περισσότεροι φοιτητές που επέλεξαν να παρακολουθήσουν εργαστήρια PP συνέχισαν σε κάποια ειδικότητα της επιστήμης των υπολογιστών(προγραμματιστές, μηχανικοί) σε σχέση με τους solo. Μερικοί πιθανοί παράγοντες γι αυτή τους την απόφαση είναι:
Έγινε σωστός καταμερισμός εργασίας και στα 2 άτομα και ότι νιώθουν ότι προσέφεραν το ίδιο.
Είχαν μια πολύ καλή συνεργασία με το ζεύγος τους.
Οι φοιτητές πιστεύουν ότι απέκτησαν μια πολύ καλή γνώσης των θεμελιωδών αρχών της επιστήμης των υπολογιστών μέσω της συνεργασίας με το ζεύγος τους.

Lucas Layman et. al. [S55][**How and Why Collaborative Software Development Impacts the Software Engineering Course**] ερεύνησαν την επίπτωση που έχει ο προγραμματισμός σε ακαδημαϊκό επίπεδο. Οι έρευνες αυτές έγιναν στο πανεπιστήμιο NSCU με συμμετέχοντες 192 φοιτητές σε

μια ολόκληρη ακαδημαϊκή περίοδο. Οι φοιτητές μέσω ερωτηματολογίων αξιολογήθηκαν για την αντίληψη που έχουν ως προγραμματιστές, τις συνήθειες τους όταν εργάζονται αλλά και την άποψη τους για την συνεργασία στις διάφορες φάσεις του προγραμματισμού. Μετά από στατιστική ανάλυση κατέληξαν πως οι κυριότεροι παράγοντες που επηρεάζουν τον προγραμματισμό ανά ζεύγη είναι η προγραμματιστική ικανότητα, η αυτοπεποίθηση, η κωλυσιεργία και η υπευθυνότητα. Φοιτητές με χαμηλή αυτοπεποίθηση προτιμούν PP και οι οκνηροί φοιτητές πιστεύουν πως ο PP εξοικονομεί χρόνο. Η προσωπική αντίληψη αλλά και οι συνήθειες στην δουλειά είναι αυτά που επηρεάζουν τον γενικό μέσο όρο βαθμολογίας σε ένα μάθημα. Υπάρχουν όμως και φοιτητές ανεξαρτήτως μεγέθους project και των θετικών του PP που προτιμούν να εργάζονται μόνοι. Κλείνοντας παρατηρούν πως τα ευρήματα τους επιβεβαιώνουν προηγούμενες έρευνες που προτείνουν τον προγραμματισμό ανά ζεύγη για αύξηση αυτοπεποίθησης και βελτίωση των εργασιακών συνηθειών.

Jari Vanhanen et. al.[S56][**Issues and Tactics when Adopting Pair Programming: A Longitudinal Case Study**] έκαναν μια έρευνα για να παρατηρήσουν την υιοθέτηση του προγραμματισμού ανά ζεύγη στο περιβάλλον εργασίας μιας Φιλανδικής εταιρίας. Σκοπός τους ήταν να εξακριβώσουν τα προβλήματα που θα προκύψουν και ποιες τακτικές θα υλοποιηθούν για να ξεπεραστούν αυτά τα προβλήματα. Στην μελέτη έλαβε μέρος μια ομάδα 35 προγραμματιστών με την εταιρεία να στοχεύει στην αύξηση της ποιότητας των προγραμμάτων αλλά και στη διάδοση γνώσης σε όλους τους συμμετέχοντες. Ο αρχηγός αυτής της ομάδας καθορίστηκε να κάνει την προώθηση του προγραμματισμού ανά ζεύγη προτείνοντας απλές στρατηγικές για την υλοποίηση του οι οποίες συνοψίζονται στις εξής:

- Εθελοντική χρήση.
- Πρέπει να χρησιμοποιείται όταν ένα νέο μέλος έρχεται στην ομάδα, όταν χρειάζεται να διαδοθεί η γνώση και σε δύσκολα προγραμματιστικά προβλήματα.
- Οι συμμετέχοντες είναι αυτοί που αποφασίζουν σε ποια φάση της εξέλιξης του προγράμματος θα την χρησιμοποιήσουν. Ανάμεσα στις συνεδρίες δεν θα πρέπει να γίνεται solo προγραμματισμός που να υπερβαίνει τις 10 ώρες.
- Οτιδήποτε παραδίδει το ζεύγος υπεύθυνο γι αυτό είναι πάντα ένα άτομο.
- Οι συμμετέχοντες καθορίζουν μόνοι τους τα ζεύγη**.**
- Οι συνεδρίες πρέπει να διαρκούν από μισή έως τρείς ώρες.

Οι ερευνητές μέσω των ερωτηματολογίων θεωρούν πως οι παραπάνω στρατηγικές βοήθησαν σημαντικά στην ενσωμάτωση του PP. Η ενθάρρυνση έναντι της επιβολής είναι το ένα από αυτά που βοήθησαν στην επικράτηση καλού κλίματος στην ομάδα και μια θετική στάση των συμμετεχόντων απέναντι στον PP. Το κυριότερο πρόβλημα που αντιμετώπισε η ομάδα ήταν αυτό των υποδομών. Κατά την διάρκεια των συνεδριών υπήρχαν αρκετές συζητήσεις έτσι αυξανόταν ο θόρυβος με αποτέλεσμα να επηρεάζονται τα υπόλοιπα μέλη της εταιρείας. Το πρόβλημα αυτό ξεπεράστηκε από την εταιρεία όταν δημιουργήθηκε ένα δωμάτιο αποκλειστικά για PP. Η έρευνα αυτή θεωρείται κλειστή και βλέπει μόνο προς συγκεκριμένες κατευθύνσεις έτσι οι ερευνητές κλείνοντας μας λένε ότι θέλουν να επαναλάβουν την έρευνα σε περισσότερες εταιρείες για να μπορέσουν να την γενικοποιήσουν.

Hema Srikanth et. al. [S57][**On Pair Rotation in the Computer Science Course**] έκαναν ένα πείραμα πάνω στον προγραμματισμό ανά ζεύγη και συγκεκριμένα μελέτησαν τις επιπτώσεις που έχει η εναλλαγή ζευγαριών πάνω στους φοιτητές. Τα αποτελέσματα έδειξαν πως υπάρχουν θετικά και αρνητικά στοιχεία. Η εναλλαγή ζευγαριών βοήθησε τους φοιτητές να ανταλλάξουν γνώση μεταξύ τους και να μάθουν πως οι συμφοιτητές τους αντιμετώπιζαν τα διάφορα προγραμματιστικά προβλήματα. Επίσης η εναλλαγή βοήθησε τα ζευγάρια σε μελλοντικές εργασίες όταν αυτά δεν ήταν συμβατά (πχ διαφορετικά προγράμματα μέσα στην ημέρα, διαφορετικές προσωπικότητες). Ορισμένες φορές υπήρχαν και προβληματικοί φοιτητές οι οποίοι ήταν αμελείς και δεν συνεισέφεραν καθόλου στις εργασίες. Με την εναλλαγή οι φοιτητές αυτοί ήταν ευκολότερο να εντοπιστούν και να μεταχειριστούν κατάλληλα αφού οι καθηγητές έπαιρναν feedback από περισσότερους φοιτητές. Ένα αρνητικό στοιχείο είναι ότι οι φοιτητές με κάθε αλλαγή ζεύγους χρειαζόταν να ξαναπροσαρμόσουν το προγραμματιστικό τους στυλ. Υπήρχαν ζευγάρια που ήταν αρκετά συμβατά και αποδοτικά. Όταν όμως άλλαζαν ταίρι υπήρχε το ρίσκο ότι το νέο ζεύγος δεν θα ήταν το ίδιο συμβατό. Η αξιολόγηση του ζεύγους είναι αρκετά πολύπλοκη διαδικασία σε σχέση με την αξιολόγηση ενός μεμονωμένου φοιτητή που θεωρείται σχετικά απλοϊκή.

Shaochun Xu et. al. του [S58] [**Empirical Validation of Test-Driven Pair Programming in Game Development**] μελέτησαν μερικές τεχνικές του XP στην δημιουργία παιχνιδιών για H/Y. Ανάμεσα σε αυτές τις τεχνικές ήταν και ο PP. Οι 12 φοιτητές που συμμετείχαν στο πείραμα δούλεψαν είτε σαν ζευγάρια είτε σαν solo προγραμματιστές και σκοπός τους ήταν να φτιάξουν ένα παιχνίδι γραμμένο σε java. Κατέληξαν ότι οι προγραμματιστές σε ζεύγη χρειάστηκαν πολύ λιγότερο χρόνο για να ολοκληρώσουν το πρόγραμμα. Επίσης τα προγράμματα ήταν πιο ποιοτικά και μπορούσαν να ολοκληρώσουν με επιτυχία περισσότερα test cases. Οι PP είχαν μεγαλύτερη αποδοτικότητα γράφοντας περισσότερες γραμμές κώδικα ανά ώρα με μεγαλύτερη συνοχή. Όλοι οι φοιτητές στο τέλος του πειράματος ήταν απόλυτα ικανοποιημένοι από την συμμετοχή τους και είναι διατεθειμένοι να χρησιμοποιήσουν την τεχνική του PP στο μέλλον.

Andrea Janes et. al. [S59][**An Empirical Analysis on the Discontinuous Use of Pair Programming**] σε πείραμα που έκαναν μελέτησαν την αποτελεσματικότητα του PP σαν εργαλείο για την μεταβίβαση γνώσης και εμπειριών σε ένα περιβάλλον όπου οι προγραμματιστές(φοιτητές) συναντιόνταν περιστασιακά . Το αποτέλεσμα της έρευνας ήταν να διαπιστώσει πως οι φοιτητές σε ποσοστό 92% είχαν αύξηση στην επικοινωνία μεταξύ τους. Επίσης οι ικανότητες των φοιτητών για επίλυση προβλημάτων, διαχείριση χρόνου και απόκτηση γνώσης βελτιώθηκαν εξίσου. Τα αποτελέσματά τους επιβεβαιώνουν προηγούμενες έρευνες και τονίζουν πως είναι ιδιαίτερα χρήσιμα σε περιπτώσεις distributed pair programming(Απομακρυσμένος προγραμματισμός ανά ζεύγη).

Jari Vanhanen et. al. [S60][**Experiences of Using Pair Programming in an Agile Project**] στο πείραμα που διεξήγαγαν προσπάθησαν να διαπιστώσουν πως μια ομάδα προγραμματιστών εφαρμόζει τον προγραμματισμό ανά ζεύγη σε ένα project και ποιες είναι οι επιπτώσεις του. Η έρευνα έγινε σε μια μεγάλη εταιρεία τηλεπικοινωνιών στην Φιλανδία και πήραν μέρος 4 επαγγελματίες προγραμματιστές. Ο PP τους βοήθησε στις πολύπλοκες προγραμματιστικές εργασίες αυξάνοντας την ποιότητα του κώδικα και μειώνοντας την προσπάθεια που χρειαζόταν να καταβάλουν αλλά και τα λάθη(bugs) στον κώδικα. Αντίθετα στις απλές εργασίες η προσπάθεια αυξανόταν σε σχέση με τον solo προγραμματισμό. Υπήρχε μια καθημερινή εναλλαγή στα μέλη του ζεύγους κάτι που αύξησε την ανταλλαγή γνώσης ανάμεσα στους προγραμματιστές όλης της ομάδας. Στο τέλος κάθε μέλος της ομάδας γνώριζε ακριβώς το όλο σύστημα.

Timothy H. DeClue [S61][**PAIR PROGRAMMING AND PAIR TRADING EFFECTS ON LEARNING AND MOTIVATION IN A CS2 COURSE**] έκανε μια έρευνα για να διαπιστώσει εάν μπορούμε επιτυχώς να ενσωματώσουμε τον προγραμματισμό ανά ζεύγη στην εκπαιδευτική διαδικασία. Στην έρευνα έλαβαν μέρος 24 μαθητές χωρισμένοι σε ζευγάρια και εναλλάσσονταν μεταξύ τους κάθε 2 εβδομάδες. Οι μαθητές στις εργασίες τους φαίνεται να έχουν μικρότερο χρόνο ολοκλήρωσης, λιγότερα λάθη και μεγαλύτερη κατανόηση για το πώς αναπτύσσεται το λογισμικό. Η εναλλαγή ζευγών δεν ήταν δύσκολη για τους μαθητές και τους βοήθησε στο να γράψουν πιο ευκολοδιάβαστο κώδικα με πιο συμπαγή σχεδιασμό και καλύτερη χρήση της βιβλιογραφίας.

Brian Hanks [S62][**Student Attitudes toward Pair Programming**] έκανε μια έρευνα σε ακαδημαϊκό επίπεδο για τις απόψεις των φοιτητών προς τον προγραμματισμό ανά ζεύγη. Η έρευνα αυτή βασίστηκε σε τρία προηγούμενα πειράματα που είχε κάνει και θέλησε να δει γιατί υπήρξε ασυνέπεια στα αποτελέσματα διαφορετικών τάξεων με διαφορετικούς καθηγητές. Τα αποτελέσματα της έρευνας έδειξαν πως στους φοιτητές άρεσε ο προγραμματισμός ανά ζεύγη, πιστεύουν ότι τους βοήθησε να μάθουν περισσότερο και έκανε το μάθημα πιο διασκεδαστικό. Σε μια τάξη υπήρχαν ενδείξεις πως η αρνητική στάση ενός καθηγητή απέναντι στον PP θα είχε αντίκτυπο και στην στάση των μαθητών. Αυτή την ασυνέπεια ο ερευνητής δεν κατάφερε να την αποδείξει. Παρόλα αυτά επισημαίνει πως όταν ανακαλύπτονται νέες τεχνικές αυτές θα πρέπει να εφαρμόζονται σωστά και όχι με προκατειλημμένη στάση. Αυτό θα βοηθήσει την βελτίωση των τεχνικών αυτών γενικότερα, ώστε να ωφεληθούν και άλλοι φοιτητές.

Sarah B. Berenson et. al. [S63][**VOICES OF WOMEN IN A SOFTWARE ENGINEERING COURSE: REFLECTIONS ON COLLABORATION**] μελέτησαν την συμπεριφορά των γυναικών κατά την διαδικασία του προγραμματισμού ανά ζεύγη σε πανεπιστημιακό περιβάλλον. Συγκεκριμένα προσπάθησαν να χαρακτηρίσουν τα συνεργασιακά περιβάλλοντα όταν σε αυτά συμμετέχουν γυναίκες και τι είναι αυτό που υπάρχει σε αυτά και αυξάνει το ενδιαφέρον των γυναικών για την επιστήμη των υπολογιστών. Τα αποτελέσματα των συνεντεύξεων αποκάλυψαν πως αυτό που βοηθά τις γυναίκες περισσότερο είναι η

αίσθηση της συνεργασίας, η παραγωγικότητα αλλά και η αυξημένη αυτοπεποίθηση. Η συνάντηση των μελών πρόσωπο με πρόσωπο, η αυξημένη ποιότητα κώδικα και ο μικρότερος χρόνος ολοκλήρωσης των εργασιών έδιναν στις γυναίκες μεγαλύτερη ικανοποίηση και αυτοπεποίθηση. Κλείνοντας μας λένε πως τα ευρήματα δεν είναι σωστό να γενικοποιηθούν λόγω του μικρού δείγματος αλλά και το ότι αναφέρονται σε προπτυχιακούς φοιτητές.

Jan Chong et. al. [S64][**The Social Dynamics of Pair Programming**] έκαναν μία έρευνα στον βιομηχανικό τομέα και μελέτησαν ποιοι είναι οι παράγοντες και πως επηρεάζουν την διαδικασία του προγραμματισμού ανά ζεύγη. Στην έρευνα αυτή έλαβαν μέρος 2 εταιρείες λογισμικού. Η πρώτη παρατήρηση που έγινε είναι ότι οι ρόλοι του Driver και του Navigator δεν ακολουθούνται ακριβώς όπως περιγράφεται στην βιβλιογραφία. Ο driver και ο navigator δεν κινούνταν με διαφορετικά επίπεδα αφαιρετικότητας αλλά σκεφτόταν συνεχώς με την ίδια λογική και χαράσσουν όμοια στρατηγική για την επίλυση του προβλήματος. Ο Navigator είναι πιο επιρρεπής στο να του αποσπαστεί η προσοχή και να αφαιρεθεί, με την συχνή εναλλαγή ρόλων όμως αυτό το πρόβλημα μειώνεται και αυξάνεται η αποδοτικότητα και των 2 μελών. Ένα άλλο σημαντικό στοιχείο που παρατηρήθηκε είναι ότι οι προγραμματιστές προτιμούν να έχουν ζεύγος κάποιον με όμοιες προγραμματιστικές δυνατότητες και γνώση. Αυτό όμως άλλαζε όταν νέα μέλη με λιγότερες γνώσεις ερχόταν στην ομάδα. Τα παραπάνω θα πρέπει να λαμβάνονται υπ' όψιν κατά την διαδικασία επιλογής ζευγαριών. Η εναλλαγή μεταξύ των ζευγών θα πρέπει να γίνεται στα αρχικά στάδια της ανάπτυξης για να μπορέσει να διαδοθεί η γνώση του κώδικα αλλά και να υπάρξει ολική γνώση του συστήματος από όλα τα μέλη της ομάδας. Αυτό βέβαια θα πρέπει να αποφεύγεται στην μετέπειτα ανάπτυξη του έργου γιατί μπορεί να διασπάσει κάποιο αποδοτικό ζευγάρι και να φέρει δυσλειτουργίες.

Beth Simon et. al. [S65][**First Year Students' Impressions of Pair Programming in CS1**] πραγματοποίησαν μια έρευνα για φοιτητές όπου στο εισαγωγικό μάθημα προγραμματισμού χρησιμοποίησαν την τεχνική του προγραμματισμού ανά ζεύγη και στην συνέχεια την επόμενη χρονιά στο επόμενο μάθημα συνέχισαν σαν solo προγραμματιστές. Οι απαντήσεις των φοιτητών μας λένε πως βρήκαν αρκετά ευκολότερο να κατανοήσουν ορισμένες πτυχές του προγραμματισμού όταν βρισκόταν σε ζεύγη. Στον προγραμματισμό πολλές φορές τα προβλήματα είναι δύσκολα και δυσνόητα και προκαλούν άγχος στους προγραμματιστές. Έχοντας κάποιον δίπλα σου βοηθά σε αυτές τις καταστάσεις, αφού μοιράζοντας την γνώση μπορείς να βρεις περισσότερες από μια λύσεις για ένα πρόβλημα ευκολότερα. Με τον PP υπήρχαν πολλές κοινωνικές αλληλεπιδράσεις με τους φοιτητές να νιώθουν το αποτέλεσμα της παραγωγής κώδικα περισσότερο σαν ομαδική προσπάθεια. Ένα σημαντικό πρόβλημα που παρατηρήθηκε είναι αυτό της εύρεσης κοινού χρονοδιαγράμματος μεταξύ των ζευγών. Οι solo προγραμματιστές έβρισκαν πιο συναρπαστικό να λύνουν τα προβλήματα μόνοι τους και νιώθουν ότι γνωρίζουν καλύτερα τις απαντήσεις. Οι solo προγραμματιστές παραδέχτηκαν ότι όταν χρειαζόταν βοήθεια την ζητούσαν πρώτα από κάποιον συμφοιτητή τους και μετά από τον επιβλέπων καθηγητή.

4.3 Reviews

Hanna Hulkko et. al. [S66][**A Multiple Case Study on the Impact of Pair Programming on Product Quality**] έκαναν μια συγκεντρωτική ανάλυση για τέσσερις μεγάλες έρευνες που είχαν δημοσιοποιηθεί μέχρι κάποια συγκεκριμένη χρονολογία. Βρήκαν πως πλέον υπάρχουν χειροπιαστά στοιχεία τα οποία μπορούν να χρησιμοποιηθούν στην βιομηχανία. Για παράδειγμα μπορούν να εκτιμήσουν και να επικεντρώσουν την προσπάθεια των PP στοχευόμενα σε συγκεκριμένες δραστηριότητες, εργασίες και φάσεις της ανάπτυξης. Επιπλέον βρέθηκαν πραγματικά ποσοτικά στοιχεία που υποστηρίζουν τα αποτελέσματα της ποιότητας του λογισμικού. Μελλοντικά σχεδιάζουν να επαναλάβουν την έρευνά τους με μεγαλύτερη λεπτομέρεια και πληθώρα στις μεταβλητές .

Tore Dybå et. al. [S67][**Are Two Heads Better than One? On the Effectiveness of Pair Programming**] έκαναν ένα review 15 ερευνών που έγιναν από το 1998 μέχρι το 2007. Οι ιδιότητες των ερευνών που μελέτησαν ήταν ο χρόνος, η προσπάθεια που καταβλήθηκε ανάμεσα στους solo και τους PP και η ποιότητα του προγράμματος. Η μετά-ανάλυση που έκαναν έδειξε ενδιαφέροντα αποτελέσματα. Ο PP είναι αρκετά αποτελεσματικός όταν υπάρχουν πολύπλοκα προγραμματιστικά προβλήματα. Επίσης μπορούν να κερδίσουν χρόνο σε πιο απλοϊκές εργασίες. Καταλήγοντας επισημαίνουν πως χρειάζονται περισσότερες έρευνες σε διαφορετικές κατευθύνσεις για να καταλάβουμε τι δουλεύει και τι όχι.

Jo E. Hannay et. al. [S68][**The effectiveness of pair programming: A meta-analysis**] συγκέντρωσαν και ανέλυσαν 57 μελέτες πάνω στον προγραμματισμό ανά ζεύγη. Ύστερα από μελέτη και στατιστική ανάλυση με διάφορα μαθηματικά μοντέλα κατέληξαν σε ενδιαφέροντα συμπεράσματα. Οι μέχρι τώρα μελέτες και γνώση τους αποκάλυψαν πως ο προγραμματισμός ανά ζεύγη είναι αρκετά επωφελής στα πολύπλοκα προγραμματιστικά προβλήματα αφού φέρνει υψηλά επίπεδα ακρίβειας/ορθότητας στην εύρεση των λύσεων. Επίσης στις απλούστερες εργασίες μπορεί να υπάρξει εξοικονόμηση χρόνου. Η υψηλή ποιότητα στις λύσεις των πολύπλοκων προβλημάτων έρχεται με ένα σημαντικό κόστος, αυτό του χρόνου και της μεγαλύτερης προσπάθειας. Αντίθετα ο μικρότερος χρόνος υλοποίησης των απλούστερων εργασιών οδηγεί σε μειωμένη ποιότητα. Από τα παραπάνω συμπεραίνουν πως ο προγραμματισμός ανά ζεύγη είναι ιδανικό να χρησιμοποιείται ή όταν υπάρχουν χαμηλά επίπεδα πολυπλοκότητας και ο χρόνος είναι η ουσία ή όταν υπάρχουν υψηλά επίπεδα πολυπλοκότητας και η ορθότητα παίζει σημαντικό ρόλο.

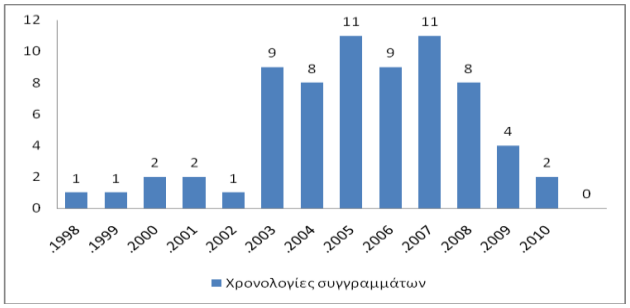
Charlie McDowell et. al. [S69][**Experimenting with Pair Programming in the Classroom**] ανέλυσαν 3 μελέτες του πανεπιστημίου UCSC θέλοντας να δείξουν τα θετικά του προγραμματισμού ανά ζεύγη στην διδασκαλία των φοιτητών. Τα αποτελέσματα έδειξαν πως οι φοιτητές που χρησιμοποιούσαν τον PP είχαν καλύτερους βαθμούς στις εξετάσεις και αρκετά βελτιωμένη ικανότητα προγραμματισμού σε σχέση με τους solo. Οι περισσότεροι φοιτητές που είναι σε ζευγάρια αποκτούν γνώσεις ευκολότερα και τα προγράμματα που δημιουργούν είναι ποιητικότερα. Κλείνοντας μας λένε πως χρειάζεται να γίνουν περισσότερες έρευνες για τον χρόνο που απαιτείται για την ολοκλήρωση μιας εργασίες γιατί στα πειράματα του UCSC δεν χρησιμοποιήθηκαν τα σωστά εργαλεία και τεχνικές και τα αποτελέσματα ήταν αντικρουόμενα μεταξύ τους. Επίσης και άλλες μελέτες θα πρέπει να γίνουν με παραμέτρους όπως η αξιολόγηση, η εποπτεία και η εναλλαγή ζευγών.

Laurie Williams et. al. [S70][**In Support of Student Pair-Programming**] έκαναν μια συγκεντρωτική έρευνα επιστημονικών συγγραμμάτων και θέλησαν να δείξουν πως ο προγραμματισμός ανά ζεύγη μπορεί να βελτιώσει την εκπαιδευτική διαδικασία. Τα αποτελέσματα της εργασίας τους δείχνουν πως οι φοιτητές μπορούν να ολοκληρώνουν τις εργασίες τους γρηγορότερα και σε υψηλότερο επίπεδο. Η επικοινωνία που αναπτύσσεται μεταξύ των φοιτητών τους οδηγεί να είναι περισσότερο χαρούμενοι, να μαθαίνουν γρηγορότερα και να είναι λιγότερο πιεσμένοι. Η επιθυμία των φοιτητών να «αντιγράψουν» έχει μειωθεί όπως έχει μειωθεί και το φόρτο εργασίας των καθηγητών.

5. *Περιορισμοί έρευνας*

Υπάρχουν ορισμένοι παράγοντες που πρέπει να συζητηθούν και αναφέρονται στους περιορισμούς που μπορεί κάποιος να συναντήσει στην έρευνα. Ο πρώτος περιορισμός αφορά την επάρκεια του String αναζήτησης. Είναι πιθανό αυτό το string αναζήτησης να μην είναι αρκετά ικανό για την εύρεση όλων των διαθέσιμων και σημαντικών ερευνών που πρέπει να συμπεριληφθούν στην συγκεκριμένη βιβλιογραφική μελέτη. Στην διάρκεια αναζήτησης βρέθηκαν πολλές και ενδιαφέρουσες έρευνες οι οποίες δεν είχαν εμπειρικά ή και στατιστικά δεδομένα για να τις υποστηρίξουν. Εάν αυτές οι έρευνες συμπεριλαμβανόταν και δεν απορριπτόταν τα αποτελέσματα αυτής της μελέτης μπορεί να ήταν διαφορετικά.

6. *Discussion*



Στις επόμενες ενότητες γίνεται η σύνθεση των αποτελεσμάτων και η συζήτησή τους. Αρχικά θα συζητηθούν τα πειράματα, έπειτα τα case/field studies και surveys και στο τέλος τα reviews. Από την ανάλυση στις χρονολογίες δημοσίευσης των συγγραμμάτων παρατηρούμε πως από το 2003 έως και το 2008 υπάρχει μια «έξαρση» και ένα τεράστιο ενδιαφέρον των ερευνητών για την τεχνική του προγραμματισμού ανά ζεύγη και ανάγκη για κάλυψη όσο το δυνατόν ευρύτερης θεματολογίας.

6.1 Πειράματα

Από τις συνολικά 70 μελέτες της έρευνας οι 49 από αυτές ήταν πειράματα. Σε όλα τα πειράματα γενικά επικρατεί θετική στάση απέναντι στον προγραμματισμό ανά ζεύγη και στις συνεργασιακές τεχνικές στον προγραμματισμό. Τα πειράματα [S4][S15][S26][S32][S38] δείχνουν ξεκάθαρα την υπεροχή που έχουν οι προγραμματιστές ανά ζεύγη στις ακαδημαϊκές τους επιδόσεις. Είτε αναφερόμαστε σε τελικές εξετάσεις, είτε σε διαγωνίσματα εξαμήνου, project και εργασίες οι προγραμματιστές ανά ζεύγη φαίνεται πως τα καταφέρνουν καλύτερα έχοντας μεγαλύτερα ποσοστά επιτυχίας και υψηλότερες βαθμολογίες. Η παρουσία των φοιτητών σε κλειστές ομάδες προγραμματιστών ανά ζεύγη δεν είχε καμία επίπτωση στην μετέπειτα πορεία τους σαν solo προγραμματιστές[S4][S15]. Σε ορισμένες περιπτώσεις[S4][S26] ο προγραμματισμός ανά ζεύγη έδωσε μια επιπλέον ώθηση στους φοιτητές να κυνηγήσουν μια «ειδικότητα» σχετιζόμενη με την επιστήμη των υπολογιστών. Τα πειράματα δείχνουν πως οι προγραμματιστές ανά ζεύγη τείνουν να συμπληρώνουν ο ένας τις γνώσεις του άλλου. Αυτό έχει ως αποτέλεσμα το διδακτικό προσωπικό να ασχολείται με ουσιώδη προβλήματα(πχ. τεχνικές και τακτικές προηγμένων και διαφορετικών μεθόδων επίλυσης) και όχι με απλά ερωτήματα (πχ. συντακτικού) που έχουν οι solo προγραμματιστές[S15][S31][S35]. Επίσης αυτή η αλληλοσυμπλήρωση βοηθά στην καλύτερη διάδοση της γνώσης και σε μια πιο ολοκληρωμένη άποψη για το αναπτυσσόμενο σύστημα[S6][S24][S29]. Τα ζεύγη έχουν μια πιο γρήγορη, μεγάλη και σταθερή αφομοίωση γνώσης σε σχέση με τους solo προγραμματιστές[S39]. Στα πειράματα [S10] και [S31] δεν διακρίθηκαν διαφορές στην ακαδημαϊκή επίδοση μεταξύ των δυο ομάδων.

Τα πειράματα [S1][S10][S13][S35][S36][S39][S55][S58][S63] μας αποκαλύπτουν πως υπάρχει αύξηση στην παραγωγικότητα και μείωση στον χρόνο ολοκλήρωσης των προγραμμάτων. Κατά την διάρκεια σύζευξης με άλλους ανθρώπους υπάρχει το κοινό αίσθημα ότι κανένας δεν θέλει να απογοητεύσει τον συνεργάτη του(Pair Pressure) για να μην θεωρηθεί υπεύθυνος για την αποτυχία. Στα πειράματα [S35][S38][S45] μελετήθηκε αυτό το φαινόμενο. Τα ζεύγη έπαψαν να σπαταλούν τον χρόνο τους άσκοπα(βλέποντας mail, σερφάροντας στο διαδίκτυο κτλ.) και αφοσιώθηκαν αποκλειστικά στην εργασία τους. Το ιδανικό σενάριο σε μία επιχείρηση είναι οι εργαζόμενοι να γνωρίζει ο ένας τον άλλον έτσι ώστε να μην σπαταληθεί χρόνος στην οικειοποίηση μεταξύ των μελών του ζεύγους. Αυτό όμως είναι μια ιδανική κατάσταση και δεν συμβαίνει πάντα. Αυτός ο παραπάνω χρόνος που χρειάζεται το ζεύγος για να «δέσει» (Pair Jelling) ορισμένες φορές οδηγεί στην αύξηση του χρόνου ολοκλήρωσης, μείωση στην παραγωγικότητα, κούραση από την μεριά των προγραμματιστών και παραπάνω προσπάθεια[S3][S6][S11][S24][S48]. Αυτό όμως είναι ένα κόστος που η εταιρεία θα το πληρώσει μία φορά και θα αποσβεστεί στην διάρκεια του χρόνου[S6]. Όταν ένας νέος προγραμματιστής γίνει μέλος σε μια ομάδα ανάπτυξης άγνωστη προς αυτόν θα πρέπει να ενσωματώνεται μέσω των τεχνικών του προγραμματισμού ανά ζεύγη. Έτσι θα βοηθηθεί στην εκπαίδευση του αλλά και στην γνωριμία με το αναπτυσσόμενο project[S14]. Όταν έρχονται αντιμέτωποι με άγνωστα και πολύπλοκα προγραμματιστικά προβλήματα οι προγραμματιστές ανά ζεύγη έχουν μεγαλύτερη παραγωγικότητα σε σχέση με τους solo προγραμματιστές[S25]. Αναλύοντας δείκτες BC και MSI βρέθηκε ότι δεν υπάρχει αξιοσημείωτη διαφορά στην απόδοση[S40].Στην ίδια κατεύθυνση κινούνται και οι μελέτες [S21][S41][S43][S49] στις οποίες δεν παρατηρείται διαφορά στην απόδοση μεταξύ προγραμματιστών ανά ζεύγη και προγραμματιστών της παραδοσιακής solo τεχνικής.

Η συντριπτική πλειοψηφία των πειραμάτων δείχνει πως ο προγραμματισμός ανά ζεύγη οδηγεί στην αύξηση της ποιότητας και στην μείωση των παραγόμενων λαθών με μόνο ένα πείραμα να έχει ανάμεικτα αποτελέσματα[S17]. Στα πειράματα για να εξακριβωθεί η ποιότητα χρησιμοποιούνται κυρίως τα εσωτερικά χαρακτηριστικά (πχ. Lines Of Code – LOC κτλ.) ή εξωτερικά χαρακτηριστικά (πχ. επιτυχία σε test cases, ακαδημαϊκή βαθμολογία κτλ.). Τα πειράματα [S3][S9][S10][S21][S24][S25][S26][S35][S36][S38] δείχνουν μια σημαντική αύξηση στην ποιότητα των παραγόμενων προγραμμάτων. Στα πειράματα [S9][S17][S22][S28] φαίνεται πως στα πολύπλοκα προγραμματιστικά προβλήματα υπάρχει μείωση των παραγόμενων λαθών και αυξημένες πιθανότητες σωστής επίλυσης. Σε διαφορετικά επίπεδα ορθότητας οι προγραμματιστές ανά ζεύγη

παράγουν ποιοτικότερα προγράμματα σε σχέση με τους solo [S27] αλλά με μεγαλύτερο κόστος. Παρατηρήθηκε επίσης συγγραφή προγραμμάτων με πιο ευκολοδιάβαστα και συμπαγή χαρακτηριστικά[S4][S43]. Σε ορισμένες περιπτώσεις βέβαια δεν μπορεί να διακριθεί σημαντική διαφορά ανάμεσα στην ποιότητα που παρήγαγαν προγραμματιστές σε ζεύγη και solo προγραμματιστές[S46].

Ο ανθρώπινος παράγοντας από την μεριά του παίζει σημαντικό ρόλο στην διαδικασία του προγραμματισμού ανά ζεύγη αλλά πολλές φορές τα πειράματα έχουν αντιφατικά αποτελέσματα μεταξύ τους. Σε ένα κοινό σημείο που συμφωνούν τα περισσότερα από αυτά [S8][S10][S15][S26][S29] [S32][S35][S36][S45] είναι ότι υπάρχει αυξημένη αυτοπεποίθηση και ικανοποίηση από την πλευρά των δημιουργών για τις παραγόμενες λύσεις. Μια μερίδα ερευνητών πιστεύει πως για να υπάρξει συμβατότητα μεταξύ των ζευγαριών έτσι ώστε να έχουμε μια επιτυχή σύζευξη, πρέπει τα μέλη του ζευγαριού να είναι αντίθετων[S23] ή ετερογενών[S20][S33] προσωπικοτήτων. Μία άλλη μερίδα πιστεύει πως η προσωπικότητα[S47] δεν παίζει σημαντικό ρόλο, με χαρακτηριστικά όπως η «ανοιχτομυαλοσύνη»[S2][S5], η ευσυνειδησία[S2], η υπευθυνότητα[S5], η μετριοφροσύνη, η αξιοπρέπεια, οι τρόποι συμπεριφοράς, ικανότητα επικοινωνίας[S18] να είναι τα χαρακτηριστικά που θα κρίνουν την επιτυχή σύζευξη με πιο ασύμβατα ζευγάρια αυτά που έχουν διαφορετικά ήθη επάνω στην εργασία[S8]. Επίσης παρατηρήθηκε πως οι προγραμματιστές προτιμούν να συνεργάζονται με κάποιον στα ίδια επίπεδα ικανότητας[S8][S30][S33][S37] και προγραμματιστές με υψηλή ικανότητα και αυξημένη αυτοπεποίθηση είναι αυτοί που τους άρεσε ο προγραμματισμός ανά ζεύγη λιγότερο[S30][S47]. Το πόσο καλά αισθάνεται ένα ζευγάρι(Feelgood Factor) είναι ένας υποψήφιος παράγοντας για υψηλή απόδοση[S1]. Αν υποθέσουμε πως η προσωπικότητα είναι ένας παράγοντας που δεν μεταβάλλεται θα πρέπει να δοθεί ιδιαίτερη προσοχή σε αυτούς που μπορούν να αναπτυχθούν και να βελτιστοποιηθούν όπως είναι η εξειδίκευση, οι προγραμματιστικές ικανότητες, η μάθηση αλλά και το κίνητρο[S7].

Διάφορα επιπλέον σημαντικά χαρακτηριστικά που παρατηρήθηκαν από τα πειράματα είναι πως ο απομακρυσμένος προγραμματισμός ανά ζεύγη μπορεί να πραγματοποιηθεί και να έχουμε παρόμοια οφέλη με τον συμβατικό[S3][S10][S12][S19] και ότι ο προγραμματισμός ανά ζεύγη μπορεί να χρησιμοποιηθεί και σε άλλα στάδια της ανάπτυξης λογισμικού(πχ. σχεδιασμός)[S16][S21][S49]. Ο προγραμματισμός ανά ζεύγη βοήθησε σημαντικά τις γυναίκες να κατανοήσουν πως η επιστήμη των υπολογιστών δεν είναι μια ανταγωνιστική και κοινωνικά απομονωτική διαδικασία και τις ενθάρρυνε να επιλέξουν κάποια ειδικότητα σε σχέση με τους υπολογιστές στο μέλλον[S34].

6.2 case/field studies – surveys

Σειρά στην συζήτηση έχουν τα 16 case/field studies – surveys. Όπως και στα πειράματα έτσι και στην συγκεκριμένη κατηγορία συγγραμμάτων παρατηρούμε πως υπάρχει μια θετική στάση απέναντι στις τεχνικές του προγραμματισμού ανά ζεύγη. Οι φοιτητές στις εργασίες τους φαίνεται να έχουν μικρότερο χρόνο ολοκλήρωσης, λιγότερα λάθη και μεγαλύτερη κατανόηση για το πώς αναπτύσσεται το λογισμικό[S61]. Επιπροσθέτως βελτίωσαν την ικανότητα τους για επίλυση προβλημάτων και την διαχείριση του διαθέσιμου χρόνου τους[S59]. Ο σωστός καταμερισμός εργασίας σε συνδυασμό με την αυξημένη κατανόηση θεμελιωδών αρχών της επιστήμης των υπολογιστών που επιτεύθηκαν μέσω του προγραμματισμού ανά ζεύγη είναι πιθανοί παράγοντες που οδήγησαν τους φοιτητές να ακολουθήσουν κάποια «ειδικότητα» στον κλάδο των υπολογιστών[S54]. Η διάδοση της γνώσης βοηθήθηκε σημαντικά από την συχνή εναλλαγή ζευγαριών. Οι φοιτητές μέσω της εναλλαγής ήρθαν σε επαφή με διαφορετικούς τρόπους σκέψης κατανοώντας περισσότερες τακτικές αντιμετώπισης προγραμματιστικών προβλημάτων αλλά συχνά έπρεπε να αναπροσαρμόσουν το προγραμματιστικό τους στυλ[S57]. Επίσης αντιμετωπίστηκαν φαινόμενα οκνηρών φοιτητών και ασύμβατων ζευγαριών[S57]. Όταν νέες τεχνικές εφαρμόζονται στην εκπαίδευση αυτές θα πρέπει να χρησιμοποιούνται με σύνεση και σωστό προγραμματισμό και όχι με προκατειλημμένη στάση έτσι ώστε να βελτιώνονται οι ίδιες και να επωφελούνται οι φοιτητές τα μέγιστα[S62].

Από την άποψη της παραγωγικότητας και του χρόνου ολοκλήρωσης έχουμε πολλά παραγόμενα συμπεράσματα. Ορισμένα από αυτά είναι αντικρουόμενα μεταξύ τους. Υπάρχουν περιπτώσεις όπου έχουμε μια αύξηση της παραγωγικότητας[S58] και μείωση στον χρόνο ολοκλήρωσης[S53][S55][S58][S61]. Σε πολύπλοκες προγραμματίστηκες εργασίες παρατηρήθηκε μείωση της καταβαλλόμενης προσπάθειας. Αντίθετα στα πιο απλά προβλήματα έχουμε αύξηση στον χρόνο ολοκλήρωσης[S60]. Η αντίθετη άποψη έρχεται από την μελέτη [S51] που παρατηρεί μείωση στην παραγωγικότητα. Επίσης παρατηρούνται και ορισμένα προβλήματα όπως διαφορετικά ωράρια εργασίας στα ζεύγη, ασυμβατότητες προσωπικότητας και αυξημένο κόστος[S53]. Για να μην υπάρξουν έντονα φαινόμενα του νόμου του Brooke βάζοντας νέο προσωπικό σε ένα ήδη αργοπορημένο project το προσωπικό αυτό θα πρέπει να αποκτάται μέσω των τεχνικών του προγραμματισμού ανά ζεύγη[S50]. Ο περισσότερος χρόνος κατά την δημιουργία ενός προγράμματος σπαταλιέται κατά την διαδικασία κατανόησης του προβλήματος με τον driver να συμβάλει ελάχιστα περισσότερο σε σχέση με τον navigator[S52].

Δεν μπορούμε εύκολα να εξακριβώσουμε αν υπάρχει αύξηση η μείωση στην παραγωγικότητα και στον χρόνο ολοκλήρωσης. Αυτό που μπορούμε να πούμε χωρίς να είμαστε τόσο επιφυλακτικοί είναι πως ο προγραμματισμός ανά ζεύγη βοήθησε στην παραγωγή ποιοτικότερων λύσεων με λιγότερα bugs[S53][S58][S60][S61][S63]. Όπως στα πειράματα έτσι και εδώ παρατηρήθηκε πως η εναλλαγή ζευγαριών βοήθησε τους δημιουργούς να έχουν μια καλύτερη άποψη και γνώση για το αναπτυσσόμενο σύστημα[S50][S51][S60][S64].

Ομοίως με τα πειράματα βρέθηκε πως οι προγραμματιστές προτιμούν να είναι σε ζεύγη όπου τα μέλη θα είναι το ίδιο τεχνικά καταρτισμένοι[S64][S65]. Σημαντικότεροι παράγοντες που παίζουν ρόλο στον προγραμματισμό ανά ζεύγη από άποψη προσωπικότητας είναι η προγραμματιστική ικανότητα, η αυτοπεποίθηση, η κωλυσιεργία η υπευθυνότητα και τα ήθη επάνω στην εργασία[S55]. Απλές στην εύρεση και στην υλοποίηση τακτικές με σημαντικότερη αυτήν της ενθάρρυνσης έναντι της επιβολής βοήθησαν στην σωστή ενσωμάτωση του προγραμματισμού ανά ζεύγη σε μια εταιρεία[S56]. Με τον προγραμματισμό ανά ζεύγη μπορούμε να επιτύχουμε μείωση του άγχους[S65],διασκέδαση[S62],μεγαλύτερη ικανοποίηση[S63] και αυξημένη επικοινωνία μεταξύ των δημιουργών[S52][S65] ακόμη και σε προσωπικό επίπεδο.

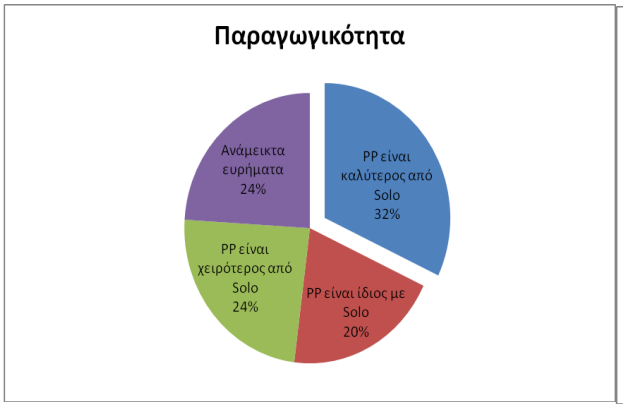
6.3Reviews

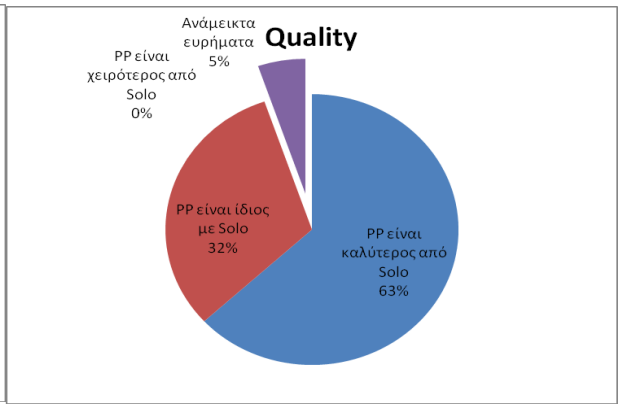
Τέλος οι υπόλοιπες 5 έρευνες είναι Reviews. Οι φοιτητές επωφελούνται από τον προγραμματισμό ανά ζεύγη αφού μπορούν να αποκτήσουν περισσότερες γνώσεις γρηγορότερα[S69]. Αυτό έχει ως αποτέλεσμα καλύτερους βαθμούς, ποιοτικότερη και γρηγορότερη συγγραφή προγραμμάτων[S69][S70]. Ο προγραμματισμός ανά ζεύγη είναι ιδανικό να χρησιμοποιείται ή όταν υπάρχουν χαμηλά επίπεδα πολυπλοκότητας και ο χρόνος είναι η ουσία ή όταν υπάρχουν υψηλά επίπεδα πολυπλοκότητας και η ορθότητα παίζει σημαντικό ρόλο[S67][S68]. Ο προγραμματισμός ανά ζεύγη μπορεί να χρησιμοποιηθεί πέρα από την φάση του προγραμματισμού και σε άλλες περιόδους ανάπτυξης του προγράμματος [S66]. Όλες οι έρευνες επισημαίνουν πως παρόλο που πέρασαν αρκετά χρόνια από την πρώτη εφαρμογή της τεχνικής ακόμη χρειάζονται αρκετές μελέτες για να μπορέσουμε να καταλάβουμε τι ακριβώς δουλεύει και τι όχι.

6.4 Γενικές παρατηρήσεις

Από τις συνολικά 70 έρευνες που περιέχει η μελέτη οι 25 από αυτές στην θεματολογία τους αναλύουν και συγκρίνουν την παραγωγικότητα του προγραμματισμού ανά ζεύγη σε σχέση με τον solo προγραμματισμό. Οι 19 από τις 70 συγκρίνουν την ποιότητα. Οι 8 συγκρίνουν την απόδοση στην εκπαίδευση και οι 13 από αυτές την ικανοποίηση των δημιουργών. Τα αποτελέσματα παρουσιάζονται συνοπτικά στον παρακάτω πίνακα και στα 2 διαγράμματα για την απόδοση και την ποιότητα. Στο appendix B υπάρχουν ομοίως τα διαγράμματα για την απόδοση στην εκπαίδευση και την ικανοποίηση των δημιουργών.

	Παραγωγικότητα	Quality	Απόδοση στην εκπαίδευση	Ικανοποίηση
PP είναι καλύτερος από Solo	S1,S10,S13,S35, S39,S55,S58,S63	S3,S9,S10, S21,S24,S25, S26,S35,S36, S38,S58,S60	S4,S15,S26,S32, S38	S8,S10,S15, S26,S29,S32, S35,S36,S45, S55,S61
PP είναι ίδιος με Solo	S21,S40,S41,S43, S49	S11,S27,S40, S41,S46,S49	S10,S31	S4
PP είναι χειρότερος από Solo	S3,S6,S11,S24,S48, S51			
Ανάμεικτα ευρήματα	S9,S16,S22,S25, S45,S60	S17	S44	S65





Στο appendix C υπάρχει πίνακας με τα ποσοστά αύξησης ή μείωσης της παραγωγικότητας, της ποιότητας και ακαδημαϊκής επίδοσης αλλά και ο αντίκτυπος που επιφέρουν. Τα ποσοστά αυτά εξήχθησαν μόνο από τις έρευνες που τα παρουσίαζαν και όπου αυτό ήταν δυνατό.

7. *Συμπεράσματα*

Η γνώση που προέρχεται από τα πειράματα, τα case/field studies, τα surveys και τα reviews φαίνεται πως συμπληρώνει η μια την άλλη. Ωστόσο υπάρχουν φορές που τα αποτελέσματα έρχονται σε σύγκρουση μεταξύ τους και δεν είναι εύκολη η εξαγωγή άμεσων συμπερασμάτων. Πολλές έρευνες συγκλίνουν και υποστηρίζουν πως ο προγραμματισμός ανά ζεύγη βοήθησε σημαντικά τους φοιτητές στις ακαδημαϊκές τους επιδόσεις [S4][S15][S26][S32][S38]. Βελτίωση των ποσοστών των επιτυχόντων από 7% έως 28% αλλά και υψηλότερες βαθμολογίες για όλες τις δραστηριότητες του εξαμήνου. Πέρα από τους βαθμούς υπήρξε βελτίωση και στην εκπαίδευση των φοιτητών χτίζοντας σωστές βάσεις στις θεμελιώδες αρχές της επιστήμης

των υπολογιστών. Με την σύζευξη οι φοιτητές έρχονται σε επαφή με πολλούς διαφορετικούς τρόπους σκέψης και έχουν μια άμεση ανταλλαγή γνώσης με τους συμφοιτητές τους[S57]. Αυτή η επαφή οδηγεί σε μια πιο σφαιρική γνώση για το αναπτυσσόμενο σύστημα [S6][S24][S29] αλλά και σε αλληλοσυμπλήρωση γνώσεων κάτι που επιφέρει άμεση μείωση στον φόρτο εργασίας των καθηγητών. Αυτή η διάδοση γνώσης μεταξύ των ομάδων των φοιτητών τείνει να είναι γρήγορη, μεγάλη και σταθερή[S39]. Η παραγωγικότητα σε ορισμένες περιπτώσεις αυξήθηκε [S1][S10][S13][S35][S36][S39][S55][S58][S63][S58] και μειώθηκε ο χρόνος ολοκλήρωσης[S53][S55][S58][S61] έως και 61% όμως το πιο πιθανό είναι να αυξηθεί και η καταβαλλόμενη προσπάθεια[S3][S6][S11][S24][S48] από 4% έως 57%. Οι προγραμματιστές ανά ζεύγη παρουσιάζουν μια πειθαρχία στην εργασία και αφού ξεπεραστούν τα αρχικά στάδια της οικειοποίησης, του δεσίματος, της χημείας μεταξύ των μελών(*pair jelling*) [S3][S6][S11][S24][S48] τότε κανείς δεν θέλει να απογοητεύσει τον συνεργάτη του(*pair pressure*) [S35][S38][S45]. Τα προγράμματα που αναπτύσσονται είναι περισσότερο ποιοτικά [S3][S9][S10][S21][S24][S25][S26][S35][S36][S38][S58][S60] με μείωση από 8% έως 50% του ποσοστού των παραγόμενων λαθών. Παρόλο που υπάρχει μια τόσο σημαντική αύξηση στην ποιότητα των προγραμμάτων αυτό δεν θα πρέπει να θεωρείται δεδομένο αλλά θα πρέπει να προσεγγίζεται με μία πιο συντηρητική ματιά. Στις περισσότερες έρευνες δεν ακολουθείται κάποια ευρέως διαδεδομένη, έγκυρη και αναγνωρίσιμη μέθοδος μέτρησης της ποιότητας(πχ ISO) παρά μόνο γνώμες ειδικών, ακαδημαϊκή βαθμολογία, επιτυχία σε test cases, αριθμός σφαλμάτων κτλ. Η ποιότητα του κώδικα επίσης φαίνεται πως είναι ανακριβής, λόγω των ποικίλων επιπέδων λειτουργικότητας σε διαφορετικά προγράμματα[S48]. Ωστόσο εξακολουθούμε να πιστεύουμε πως ο προγραμματισμός ανά ζεύγη είναι κατάλληλος όταν υπάρχουν χαμηλά επίπεδα πολυπλοκότητας και ο χρόνος είναι η ουσία, ή όταν υπάρχουν υψηλά επίπεδα πολυπλοκότητας[S9][S17][S22][S28] και η ορθότητα παίζει σημαντικό ρόλο[S67][S68] παρόλο που σε ορισμένες περιπτώσεις έχουμε αύξηση του κόστους[S27] έως 13%. Η εκπαίδευση των εργαζομένων που επιτυγχάνεται οδηγεί σε μείωση του φαινομένου του νόμου του Brooke[S50] αλλά μερικές φορές ο ρόλος του οδηγού και του καθοδηγητή δεν ακολουθούνται πιστά όπως περιγράφεται στην βιβλιογραφία[S64]. Όλα τα είδη ερευνών συμφωνούν πως τα μέλη επιλέγουν να είναι σε ζεύγη ίδιων ή παρόμοιων επιπέδων [S8][S30] [S33][S37][S64][S65] γνώσης και τεχνικής κατάρτισης. Μια άλλη σημαντική παρατήρηση όπου πειράματα, case/field studies-surveys και reviews συμφωνούν είναι πως υπάρχει αύξηση της αυτοπεποίθησης και ικανοποίησης των συμμετεχόντων στον προγραμματισμό ανά ζεύγη[S8][S10][S15][S26][S29][S32][S35] [S36][S45][S55][S61] από 81% έως 95%. Αντιθέτως οι ερευνητές παρουσιάζουν αντικρουόμενα ευρήματα και διαφωνούν μεταξύ τους για το αν η προσωπικότητα παίζει σημαντικό ρόλο η όχι στην επιτυχημένη σύζευξη. Ορισμένοι πιστεύουν πως ο προγραμματισμός ανά ζεύγη είναι αποτελεσματικότερος όταν οι προγραμματιστές που τον απαρτίζουν έχουν προσωπικότητες αντίθετες[S23] ή ετερογενής[S20][S33]. Η αντίθετη άποψη μας παρουσιάζει πως η προσωπικότητα δεν παίζει κανένα απολύτως ρόλο στην διαδικασία και πως άλλοι είναι οι παράγοντες που μας οδηγούν σε επιτυχή σύζευξη («ανοιχτομυαλοσύνη»[S2][S5], ευσυνειδησία[S2], υπευθυνότητα[S5][S55], μετριοφροσύνη, αξιοπρέπεια, τρόποι συμπεριφοράς, ικανότητα επικοινωνίας[S18], τα ήθη επάνω στην εργασία[S55]) και γι αυτό θα πρέπει να επικεντρωθούμε μόνο σε ανθρώπινα γνωρίσματα που μπορούν να αναπτυχθούν και να βελτιωθούν(εξειδίκευση, προγραμματιστικές ικανότητες, μάθηση, κίνητρο[S7]). Σε ορισμένες περιπτώσεις η κοινή εύρεση χρόνου λόγω διαφορετικών ωραρίων ήταν ένα επιπλέον πρόβλημα[S47][S53][S65] αλλά αυτό αντισταθμίζεται με την αποβολή άγχους που καταφέρνουν να έχουν οι προγραμματιστές ανά ζεύγη[S65].

Από την πρώτη υλοποίηση της τεχνικής μέχρι σήμερα έχουν περάσει αρκετά έτη. Καταλήγοντας η συγκεκριμένη έρευνα θέλει να επισημάνει πως θα πρέπει να γίνουν και άλλες μελέτες σε διαφορετικές κατευθύνσεις για να ισχυροποιήσει το κύρος των ευρημάτων της αλλά και να μπορέσουμε με ακρίβεια να αποτυπώσουμε το σαφές πλαίσιο αποτελεσματικής εφαρμογής της τεχνικής.

8. References

[1] L. Williams and R. Kesler, "Pair programming illuminated" Addison Wesley, 2002

[2] L. Williams et al. "Building pair programming knowledge through a family of experiments", Proceedings International Symposium Empirical Software Engineering, 30 Sept.-1 Oct. 2003, pages 143 – 152, 2003

[3] Beck K., Cockburn A., Jeffries R., Highsmith J., (2001) "Agile manifesto", http://www.agilemanifesto.org, 2-2010

[4]P. Brereton et al. "Lessons from applying the systematic literature review process within the software engineering domain", Journal of Systems and Software 80 (4) (2007) 571–583

[5] B. Kitchenham, "Procedures for Undertaking Systematic Literature Reviews", Joint Technical Report, Computer Science Department, Keele University

[6] B. Kitchenham et al. "Systematic literature reviews in software engineering – a systematic literature review", Information and Software Technology 51 (1) (2009) 7–15.

[7] Greenhalgh T, "How to Read a Paper"

[8] B.Kitchenham et al.(2002), "Preliminary guidelines for empirical research in software engineering", IEEE Transactions on Software Engineering 28 (8) 721–734.

[9] Tore Dyba˚ and Torgeir Dingsøyr, "Empirical studies of agile software development: A systematic review" 2008 Elsevier B.V. All rights reserved. doi:10.1016/j.infsof.2008.01.006

[10] Panagiotis Sfetsos et al. "An experimental investigation of personality types impact on pair effectiveness in pair programming" Empir Software Eng (2009) 14:187–226

[11] Apostolos Ampatzoglou and Ioannis Stamelos, "Software engineering research for computer games: A systematic review" 2010 Elsevier B.V. All rights reserved. doi:10.1016/j.infsof.2010.05.004

8.1 Sources

[S1] M. M. Muller and F. Padberg, "An Empirical Study about the Feelgood Factor in Pair Programming". Proceedings of the 10th International Symposium on Software Metrics (METRICS'04) 1530-1435/04

[S2] N. Salleh et al. "An Empirical Study of the Effects of Conscientiousness in Pair Programming using the Five-Factor Personality Model". ICSE'10, May 2-8, 2010, Cape Town, South Africa

[S3] R. Duque and C. Bravo, "Analyzing Work Productivity and Program Quality in Collaborative Programming" 978-0-7695-3372-8/08 The Third International Conference on Software Engineering Advances

[S4] L. Williams et al. "Building Pair Programming Knowledge through a Family of Experiments" Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE'03)0-7695-2002-2/03

[S5] J. Chao and G. Atli, "Critical Personality Traits in Successful Pair Programming" Proceedings of AGILE 2006 Conference (AGILE'06)

[S6] J. Vanhanen and C. Lassenius, “Effects of Pair Programming at the Development Team Level: An Experiment” 0-7803-9508-5/05

[S7] J. E. Hannay et al. “Effects of Personality on Pair Programming” IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 36, NO. 1, JANUARY/FEBRUARY 2010

[S8] L. Williams et al. “Examining the Compatibility of Student Pair Programmers” Proceedings of AGILE 2006 Conference (AGILE'06)

[S9] R. Sison, “Investigating the Effect of Pair Programming and Software Size on Software Quality and Programmer Productivity” 2009 16th Asia-Pacific Software Engineering Conference 1530-1362/09

[S10] N. Z. Zacharis, “Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course” 10.1109/TE.2010.2048328

[S11] J. Vanhanen and C. Lassenius, “Perceived Effects of Pair Programming in an Industrial Context” 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO 2007)

[S12] B.F. Hanks, “Distributed Pair Programming: An Empirical Study” C. Zannier et al. (Eds.): XP/Agile Universe 2004, LNCS 3134, pp. 81–91, 2004

[S13] G. Canfora et al. “Empirical Study on the Productivity of the Pair Programming” H. Baumeister et al. (Eds.): XP 2005, LNCS 3556, pp. 92–99, 2005

[S14] I. D. Coman et al. “Investigating the Usefulness of Pair-Programming in a Mature Agile Team” P. Abrahamsson et al. (Eds.): XP 2008, LNBIP 9, pp. 127–136, 2008

[S15] N. Nagappan et al. “Pair Learning: With an Eye Toward Future Success” F. Maurer and D. Wells (Eds.): XP/Agile Universe 2003, LNCS 2753, pp. 185–198, 2003

[S16] E. Bellini et al. “The Impact of Educational Background on Design Knowledge Sharing During Pair Programming: An Empirical Study” K-D Althoff et al. (Eds.): WM 2005, LNAI 3782, pp. 455 – 465, 2005.

[S17] M.M. Muller, “Do programmer pairs make different mistakes than solo programmers?” 0164-1212/$ - see front matter 2006 Elsevier Inc. All rights reserved.

[S18] K.S. Choi et al. “Pair dynamics in team collaboration” Computers in Human Behavior 25 (2009) 844–852

[S19] D. Stotts et al. “Virtual Teaming: Experiments and Experiences with Distributed Pair Programming” F. Maurer and D. Wells (Eds.): XP/Agile Universe 2003, LNCS 2753, pp. 129–141, 2003.

[S20] Sfetsos, P., I. Stamelos, et al. “An experimental investigation of personality types impact on pair effectiveness in pair programming.”Empirical Software Engineering 14(2): 187-226.

[S21] G. Canfora et al. “Evaluating performances of pair designing in industry” The Journal of Systems and Software 80 (2007) 1317–1327

[S22] E. Arisholm et al. “Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise” IEEE Transactions on Software Engineering Volume 33 Issue 2, February 2007

[S23] K.S. Choi et al. “Exploring the underlying aspects of pair programming: The impact of personality” Information and Software Technology Volume 50 Issue 11, October, 2008

[S24] T. Bipp et al. “Pair programming in software development teams – An empirical study of its benefits” Information and Software Technology Volume 50, Issue 3, February 2008, Pages 231-240

[S25] K. M. Lui and K.C. Chan , “Pair programming productivity: Novice–novice vs. expert–expert” International Journal of Human-Computer Studies - Human-computer interaction research in the managemant information systems discipline Volume 64 Issue 9, September 2006

[S26] C. McDowell et al. “The Impact of Pair Programming on Student Performance, Perception and Persistence” Proceeding ICSE '03 Proceedings of the 25th International Conference on Software Engineering

[S27] M.M. Muller, “Two controlled experiments concerning the comparison of pair programming to peer review” Journal of Systems and Software Volume 78 Issue 2, November 2005

[S28] M. Phongpaibul and B. Boehm, “A Replicate Empirical Comparison between Pair Development and Software Development with Inspection” First International Symposium on Empirical Software Engineering and Measurement 0-7695-2886-4/07

[S29] L. Williams, “But, Isn’t That Cheating?” November 10 - 13,1999 San Juan, Puerto Rico 29‘h ASEEREEE Frontiers in Education Conference 12b9-26

[S30] L. Thomas et al. “Code Warriors and Code-a-Phobes: A Study in Attitude and Pair Programming” SIGCSE’03, February 19-23, 2003, Reno, Nevada, USA. Copyright 2003 ACM 1-58113-648-X/03/0002

[S31]N. Nagappan et al. “Improving the CS1 Experience with Pair Programming” Proceeding SIGCSE '03 Proceedings of the 34th SIGCSE technical symposium on Computer science education

[S32] E. Medes et al. “Investigating Pair-Programming in a 2nd-year Software Development and Design Computer Science Course” ITiCSE’05, June 27–29, 2005, Monte de Caparica, Portugal.Copyright 2005 ACM 1-59593-024-8/05/0006

[S33] N. Katira et al. “On Understanding Compatibility of Student Pair Programmers” SIGCSE’04, March 3–7, 2004, Norfolk, Virginia, USA. Copyright 2004 ACM 1-58113-798-2/04/0003

[S34] L. L. Werner et al. “Pair-Programming Helps Female Computer Science Students” ACM Journal of Educational Resources in Computing, Vol. 4, No. 1, March 2004, Article 3.

[S35] L. Williams et al. “Strengthening the Case for Pair Programming” IEEE Software, vol. 17, no. 4, pp. 19-25, July/Aug. 2000

[S36] J.T. Nosek, “The Case for Collaborative Programming” COMMUNICATIONS OF THE ACM March 1998/Vol. 41, No. 3

[S37]N. Katira et al. “Towards Increasing the Compatibility of Student Pair Programmers” ICSE’05, May 15-21, 2005, St. Louis, Missouri, USA ACM 1-58113-963-2/05/0005

[S38] L. Williams and R. Kessler, “The Effects of “Pair-Pressure” and “Pair-Learning” on Software Engineering Education” Proceeding CSEET '00 Proceedings of the 13th Conference on Software Engineering Education & Training

[S39] G. Canfora et al. “Working in pairs as a means for design knowledge building: an empirical study” Proceedings of the 12th IEEE International Workshop on Program Comprehension (IWPC’04) 1092-8138/04

[S40]L. Madeyski, “Impact of Pair Programming on Thoroughness and Fault Detection effectiveness of Unit Test Suites” SOFTWARE PROCESS IMPROVEMENT AND PRACTICE Softw. Process Improve. Pract. 2008; 13: 281–295

[S41] J. Nawrocki and A. Wojciechowski “Experimental Evaluation of Pair Programming” roceedings of the 12th European Software Control and Metrics Conference, ESCOM

[S42]M. Rehman et al. “HETEROGENEOUS AND HOMOGENOUS PAIRS IN PAIR PROGRAMMING: AN EMPIRICAL ANALYSIS” 0-7803-8886-0/05 IEEE CCECE/CCGEI, Saskatoon, May 2005

[S43] R. Sison, “Investigating Pair Programming in a Software Engineering Course in an Asian Setting” 2008 15th Asia-Pacific Software Engineering Conference

[S44] C. McDowell et al. “The Effects of Pair-Programming on Performance in an Introductory Programming Course” Presented at The 33rd ACM Technical Symposium on Computer Science Education, February 7-March 3, 2002

[S45] B. Hanks, “Problems Encountered by Novice Pair Programmers” ACM J. Educ. Resour. Comput. 7, 4, Article 2 (January 2008), 13 pages

[S46] L. Madeyski, “The Impact of Pair Programming and Test-Driven Development on package Dependencies in Object-Oriented Design — An Experiment” J. M¨unch and M. Vierimaa (Eds.): PROFES 2006, LNCS 4034, pp. 278–289, 2006.

[S47] L. Layman, “Changing Students’ Perceptions: An Analysis of the Supplementary Benefits of Collaborative Software Development” Proceedings of the 19th Conference on Software Engineering Education & Training (CSEET’06)

[S48] J. Nawrocki et al. “Pair Programming vs. Side-by-Side Programming” I. Richardson et al. (Eds.): EuroSPI 2005, LNCS 3792, pp. 28 – 38, 2005

[S49]M. M. Muller, “A preliminary study on the impact of a pair design phase on pair programming and solo programming”Information and Software Technology 48 (2006) 335–344

[S50] L. Williams et al. “An Initial Exploration of the Relationship Between Pair Programming and Brooks’ Law” Proceedings of the Agile Development Conference (ADC’04)

[S51] S. Pietinen et al. “Productivity of Pair Programming in a Distributed Environment – Results from Two Controlled Case Studies” R.V. O’Connor et al. (Eds.): EuroSPI 2008, CCIS 16, pp. 47–58, 2008

[S52]S. Bryant et al. “The Collaborative Nature of Pair Programming” P. Abrahamsson, M. Marchesi, and G. Succi (Eds.): XP 2006, LNCS 4044, pp. 53 – 64, 2006

[S53]A. Begel and N. Nagappan “Pair Programming: What’s in it for Me?” ESEM’08, October 9-10, 2008, Kaiserslautern, Germany. Copyright 2008 ACM 978-1-59593-971-5/08/10

[S54] J. Carver et al. “Increased Retention of Early Computer Science and Software Engineering Students using Pair Programming” 20th Conference on Software Engineering Education & Training (CSEET'07)

[S55]L. Layman et al. “How and Why Collaborative Software Development Impacts the Software Engineering Course” October 19 – 22, 2005, Indianapolis, IN 35th ASEE/IEEE Frontiers in Education Conference T4C-9

[S56]J. Vanhanen et al. “Issues and Tactics when Adopting Pair Programming:A Longitudinal Case Study” International Conference on Software Engineering Advances(ICSEA 2007)

[S57] H. Srikanth et al. “On Pair Rotation in the Computer Science Course” Proceedings of the 17th Conference on Software Engineering Education and Training (CSEET’04)

[S58]S. Xu and V. Rajlich, “ Empirical Validation of Test-Driven Pair Programming in Game Development” Proceedings of the 5th IEEE/ACIS International Conference on Computer and Information Science and 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse (ICIS-COMSAR’06)

[S59] A. Janes et al. “An Empirical Analysis on the Discontinuous Use of Pair Programming” M. Marchesi and G. Succi (Eds.): XP 2003, LNCS 2675, pp. 205–214, 2003

[S60] J. Vanhanen and H. Korpi, “Experiences of Using Pair Programming in an Agile Project” Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS'07)

[S61]T. DeClue, “PAIR PROGRAMMING AND PAIR TRADING: EFFECTS ON LEARNING AND MOTIVATION IN A CS2 COURSE” Journal of Computing Sciences in Colleges (2003) Volume: 18, Issue: 5, Publisher: Consortium for Computing Sciences in Colleges, Pages: 49–56

[S62] B. Hanks, “Student Attitudes toward Pair Programming” ITiCSE’06, June 26–28, 2006, Bologna, Italy. Copyright 2006 ACM 1-59593-055-8/06/0006

[S63] S. Berenson et al. “VOICES OF WOMEN IN A SOFTWARE ENGINEERING COURSE: REFLECTIONS ON COLLABORATION” ACM Journal of Educational Resurces in Computing, Vol. 4, No. 1, March 2004, Article 2.

[S64] J. Chong and T. Hurlbutt, “The Social Dynamics of Pair Programming” 29th International Conference on Software Engineering (ICSE'07)

[S65]B. Simon and B. Hanks, “First Year Students’ Impressions of Pair Programming in CS1” ICER’07, September 15–16, 2007, Atlanta, Georgia, USA. Copyright 2007 ACM 978-1-59593-841-1/07/0009

[S66] H. Hulkko and P. Abrahamsson, “A Multiple Case Study on the Impact of Pair Programming on Product Quality” ICSE'05, May 15-21, 2005, St. Louis, Missouri, USA. Copyright 2005 ACM 1-581 13-963-2/05/0005.

[S67]T. Dyba et al. “Are Two Heads Better than One?” Published by the IEEE Computer Society 0740-7459 /07

[S68]J. Hannay et al. “The effectiveness of pair programming: A meta-analysis” Information and Software Technology 51 (2009) 1110–1122

[S69] C. McDowell et al. “Experimenting with Pair Programming in the Classroom” ITiCSE'03, June 30-July 2, 2003, Thessaloniki, Greece. Copyright 2003 ACM 1-58113-672-2/03/000

[S70]L. Williams and R. Upchurch , “Support of Student Pair-Programming” SIGCSE 2001 2/01 Charlotte. NC, USA © 2001 ACM ISBN *1-58113-329-4/011000*

Appendix A

Συνοπτική φόρμα εξαγόμενων δεδομένων

ID	Name	Name	Type	Research environment	Population size	Main conclusions
.S1	An Empirical Study about the Feelgood Factor in Pair Programming	Matthias M. Muller	experiment	academic	38 Students	Το πόσο καλά αισθάνεται ένα ζευγάρι είναι υποψήφιος παράγοντας για υψηλή απόδοση. Η εμπειρία δεν παίζει σημαντικό ρόλο στην απόδοση.
.S2	An Empirical Study of the Effects of Personality in Pair Programming using the Five-Factor Model	Norsaremah Salleh	experiment	academic	218 Students	Υπάρχει σχέση μεταξύ της υψηλής βαθμολογίας και της ευσυνειδησίας και ανοικτότητας σε νέες εμπειρίες
.S3	Analyzing Work Productivity and Program Quality in Collaborative Programming	Rafael Duque, Crescencio Bravo	experiment	Academic	51 Students	Οι PP ποιοτικότερα προγράμματα με λιγότερα λάθη αλλά χρειάζονται περισσότερο χρόνο
.S4	Building Pair Programming Knowledge through a Family of Experiments	Laurie Williams	experiment	Academic	1200 Students	Οι PP πέτυχαν καλύτερες ακαδημαϊκές επιδόσεις από τους solo παράγοντας ποιοτικότερα προγράμματα
.S5	Critical Personality Traits in Successful Pair Programming	Joseph Chao	experiment	Academic	58 Students	Η υπευθυνότητα και η ανοικτομυαλοσύνη είναι αυτές που θα μπορούσαν να οδηγήσουν σε υψηλή ποιότητα κώδικα
.S6	Effects of Pair Programming at the Development Team Level: An Experiment	Jari Vanhanen	experiment	Academic	20 Students	Οι PP είχαν μειωμένη παραγωγικότητα, περισσότερα λάθη αλλά αυξημένη μετάδοση γνώσης
.S7	Effects of Personality on Pair Programming	Jo E. Hannay	experiment	Industrial	196 Professionals	Παράγοντες όπως η προσωπικότητα παραμένουν σταθεροί, γι αυτό πρέπει να δοθεί βάρος σε αυτούς που μπορούν να βελτιωθούν(Εξειδίκευση, μάθηση, ικανότητες, κίνητρο)
.S8	Examining the Compatibility of Student Pair Programmers	Laurie Williams	experiment	Academic	1350 Students	Εάν επιλέξουμε τυχαία τα μέλη του ζεύγους των προγραμματιστών έχουμε τις μεγαλύτερες πιθανότητες αυτά να είναι συμβατά ανεξαρτήτως προσωπικότητας. Τα πιο ασύμβατα μέλη είναι αυτά με τα διαφορετικά ήθη επάνω στην εργασία
.S9	Investigating the Effect of Pair Programming and Software Size on Software Quality and Programmer Productivity	Raymund Sison	experiment	Academic	48 Students	Οι PP στα μικρά και πολύπλοκα συστήματα είχαν αύξηση στην ποιότητα ενώ στα πολύ μικρά συστήματα μείωση στην παραγωγικότητα
.S10	Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course	Nick Z. Zacharis	experiment	Academic	129 Students	Οι προγραμματιστές VPP σε ποσοστό περίπου 50% είχαν καλύτερη ποιότητα κώδικα με λιγότερα λάθη και μεγαλύτερη παραγωγικότητα
.S11	Perceived Effects of Pair Programming in an Industrial Context	Jari Vanhanen	experiment	Industrial	28 Professionals	Ο PP έχει θετικές επιπτώσεις στην μάθηση, ποιότητα, στην τήρηση των χρονοδιαγραμμάτων, στις διαπροσωπικές σχέσεις, στην ομαδικότητα και στην πειθαρχία. Στα αρνητικά είναι η περισσότερη καταβαλλόμενη προσπάθεια και η εξάντληση των προγραμματιστών
.S12	Distributed Pair Programming: An Empirical Study	Brian F. Hanks	experiment	Academic	76 Students	PP στον ίδιο χώρο αλλά και απομακρυσμένοι έχουν τα ίδια επίπεδα αυτοπεποίθησης.
.S13	Empirical Study on the Productivity of the Pair Programming	Gerardo Canfora	experiment	Academic	24 Students	Ένας προγραμματιστής εάν εμπλακεί σε κατάσταση προγραμματισμού ανά ζεύγη θα χρειαστεί λιγότερο χρόνο να ολοκληρώσει μια εργασία απ ότι αν ήταν solo προγραμματιστής
.S14	Investigating the Usefulness of Pair-Programming in a Mature Agile Team	Irina Diana Coman	experiment	Industrial	16 Professionals	Σε ένα προγραμματιστικό project ο PP χρησιμοποιείται κάτω από 30% του χρόνου ανάπτυξης. Ο PP είναι ιδιαίτερα χρήσιμος για την εκπαίδευση και την μετάδοση γνώσης
.S15	Pair Learning: With an Eye Toward Future Success	Nachiappan Nagappan	experiment	Academic	N/A	Οι PP είχαν μέχρι και 85% μεγαλύτερη επιτυχία στις εξετάσεις σε σχέση με τους solo. Ο PP δεν επηρεάζει την solo ικανότητα. Μικρότερο φόρτο εργασίας για το εκπαιδευτικό προσωπικό
.S16	The Impact of Educational Background on Design Knowledge Sharing During Pair Programming: An Empirical Study	Emilio Bellini	experiment	Academic	44 Students	Υπάρχει μεγάλη μεταφορά σιωπηρής γνώσης μεταξύ του ζευγαριού. Η σύζευξη μεταξύ scientific και non-scientific δεν αποδίδει
.S17	Do programmer pairs make different mistakes than solo programmers?	Matthias M. Muller	experiment	Academic	38 Students	Οι PP έχουν λιγότερα εκφραστικά λάθη στον κώδικα. Ο PP είναι κατάλληλος για πολύπλοκα προγραμματιστικά προβλήματα
.S18	Pair dynamics in team collaboration	Kyungsub S. Choi	experiment	Academic	128 Students	Αξίες όπως η μετριοφροσύνη, η αξιοπρέπεια και οι τρόποι συμπεριφοράς χαρακτηρίζουν την επιτυχή συνεργασία του ζεύγους. Ζεύγη όμοιου φύλου έχουν μεγαλύτερη συνεκτικότητα

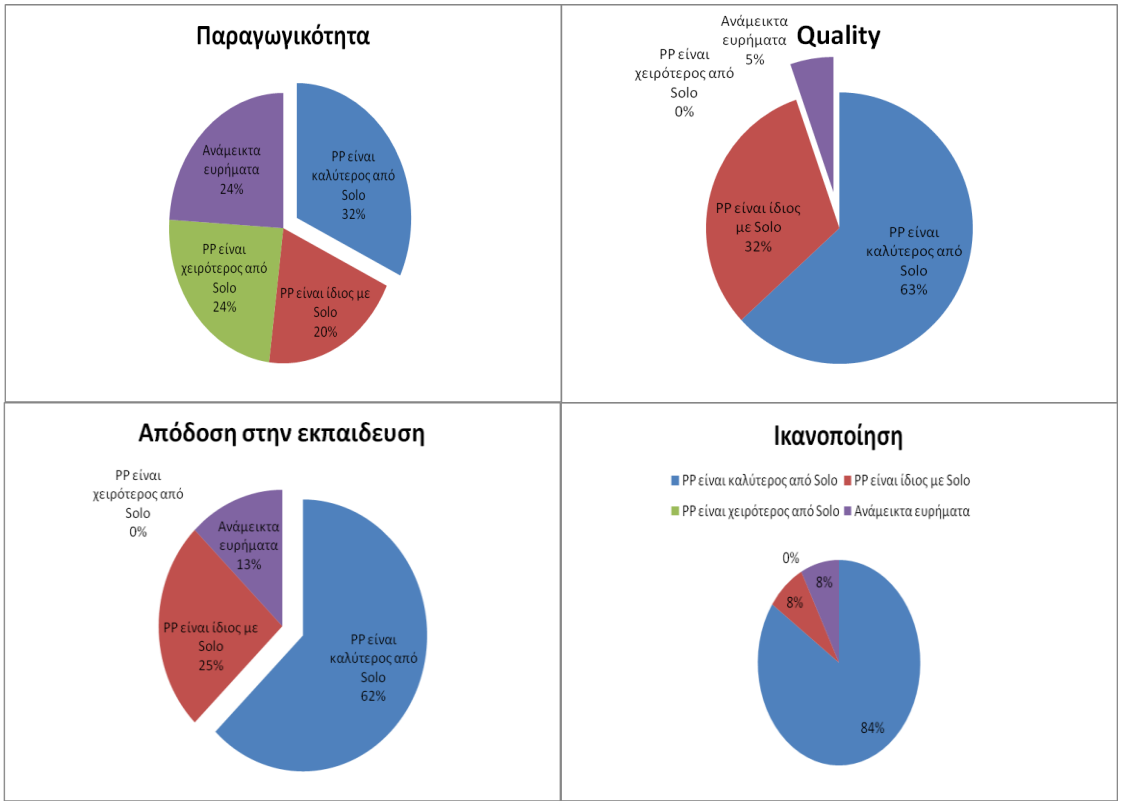
.S19	Virtual Teaming: Experiments and Experiences with Distributed Pair Programming	David Stotts	experiment	Academic	N/A	Τα προγράμματα των απομακρυσμένων PP έχουν ίδια ποιότητα με αυτών στον ίδιο χώρο. Οι απομακρυσμένοι PP επωφελούνται και αυτοί από τα φαινόμενα Pair learning και Pair pressure
.S20	An experimental investigation of personality types impact on pair effectiveness in pair programming	Panagiotis Sfetsos	experiment	Academic	70 Students	Ζεύγη με ετερογενή χαρακτηριστικά προσωπικότητας και ταπεραμέντου έχουν μεγαλύτερη αποτελεσματικότητα και συνεργασία
.S21	Evaluating performances of pair designing in industry	Gerardo Canfora	experiment	Industrial	18 Professionals	Ο Προγραμματισμός ανά ζεύγη μπορεί επιτυχώς να χρησιμοποιηθεί στην φάση του σχεδιασμού ενός προγράμματος με ικανοποιητικά αποτελέσματα
.S22	Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise	Erik Arisholm	experiment	Industrial	295 Professionals	Σε πολύπλοκες διαδικασίες συντήρησης μη έμπειροι προγραμματιστές σε ζεύγη τα καταφέρνουν το ίδιο καλά με τους πιο έμπειρους
.S23	Exploring the underlying aspects of pair programming:The impact of personality	Kyungsub S. Choi	experiment	Academic	128 Students	Οι ομάδες με ανάμεικτα MBTI χαρακτηριστικά είναι κατάλληλες για σχηματισμό ζευγών PP
.S24	Pair programming in software development teams – An empirical study of its benefits	Tanja Bipp	experiment	Academic	95 Students	Οι PP έχουν ποιοτικότερα και ευκολοδιάβαστα προγράμματα και επωφελούνται από την ανταλλαγή γνώσης αλλά με μια μικρή μείωση της απόδοσης σε σχέση με τους solo
.S25	Pair programming productivity: Novice–novice vs. expert–expert	Kim Man Lui	experiment	Academic	40 Students	Ο PP είναι το κατάλληλο εργαλείο για την επίλυση ασυνήθιστων προγραμματιστικών προβλημάτων
.S26	The Impact of Pair Programming on Student Performance, Perception and Persistence	Charlie McDowell	experiment	Academic	555 Students	Ο PP βοηθά τους μαθητές να επιτύχουν καλύτερους βαθμούς, ποιοτικότερα προγράμματα, μεγαλύτερη ευχαρίστηση και οδηγεί τους μαθητές σε αναζήτηση "ειδικότητας" σχετική με την επιστήμη των υπολογιστών
.S27	Two controlled experiments concerning the comparison of pair programming to peer review	Matthias M. Muller	experiment	Academic	38 Students	Σε ίδια επίπεδα ορθότητας PP και solo προγραμματισές έχουν το ίδιο κόστος. Για διαφορετικά επίπεδα οι PP παράγουν ποιοτικότερα προγράμματα με υψηλότερο κόστος
.S28	A Replicate Empirical Comparison between Pair Development and Software Development with Inspection	Monvorath Phongpaibul	experiment	Academic	56 Students	οι PP χρειάζεται να καταβάλουν μικρότερη προσπάθεια για να επιτύχουν ίδια η ανώτερη ποιότητα προγραμμάτων σε σχέση με τους προγραμματιστές της τεχνικής του Inspection με κόστος μικρότερο εως 22%
.S29	But, Isn't That Cheating?	Laurie Williams	experiment	Academic	20 Students	Μεγαλύτερη ικανοποίηση, αυτοπεποίθηση και γρηγορότερη εκμάθηση για τους PP
.S30	Code Warriors and Code-a-Phobes: A Study in Attitude and Pair Programming	Lynda Thomas	experiment	Academic	60 Students	Ο PP αρέσει γενικά στους φοιτητές και τους βοηθά στην σωστότερη επίλυση προβλημάτων. Ο PP άρεσε στους φοιτητές με μικρή αυτοπεποίθηση και δεν άρεσε σε αυτούς που θεωρούσαν τον εαυτό τους "εξαιρετικό" προγραμματιστή
.S31	Improving the CS1 Experience with Pair Programming	Nachiappan Nagappan	experiment	Academic	495 Students	Ο PP βοήθησε τους φοιτητές να αναπτύξουν αισθήματα υπερ των συνεργασιακών τεχνικών και να επιτύχουν ίσους ή καλύτερους βαθμούς στις εργαστηριακές ασκήσεις σε σχέση με τους solo
.S32	Investigating Pair-Programming in a 2nd-year Software Development and Design Computer Science Course	Emilia Mendes	experiment	Academic	300 Students	Τα προγράμματα των PP ήταν περισσότερο ποιοτικά, είχαν καλύτερους βαθμούς στην τελική εξέταση και μεγαλύτερα ποσοστά επιτυχίας σε σχέση με τους solo
.S33	On Understanding Compatibility of Student Pair Programmers	Neha Katira	experiment	Academic	564 Students	Φοιτητές με διαφορετικές προσωπικότητες έχουν καλύτερη συνεργασία μεταξύ τους. Οι φοιτητές προτιμούν να είναι σε ζεύγη όπου και τα 2 μέλη είναι το ίδιο τεχνικά καταρτισμένα
.S34	Pair-Programming Helps Female Computer Science Students	Linda L. Werner	experiment	Academic	141 + 24 Female Students	Ο PP βοήθησε τις γυναίκες να κατανοήσουν πως η ανάπτυξη λογισμικού δεν είναι μια ανταγωνιστική και κοινωνικά απομονωτική διαδικασία
.S35	Strengthening the Case for Pair Programming	Laurie Williams	experiment	Academic	41 Students	Υψηλή ποιότητα κώδικα σε λιγότερο χρόνο για τους PP. Το φαινόμενο του "Pair pressure" έχει θετική επίπτωση στην απόδοση
.S36	The Case for Collaborative Programming	John T. Nosek	experiment	Industrial	15 Professionals	Οι PP σε λιγότερο χρόνο παρήγαγαν ποιοτικότερες λύσεις
.S37	Towards Increasing the Compatibility of Student Pair Programmers	Neha Katira	experiment	Academic	361 Students	Φοιτητές οι οποίοι πιστεύουν ότι έχουν ίδιου επιπέδου ικανότητες αποδίδουν καλύτερα με την προσωπικότητα να μην παίζει σημαντικό ρόλο
.S38	The Effects of "Pair-Pressure" and "Pair-Learning" on Software Engineering Education	Laurie Williams	experiment	Academic	20 Students	Η «πίεση» που ασκεί το ένα ζεύγος στο άλλο είναι εξαιρετικά εποικοδομητική
.S39	Working in pairs as a means for design knowledge building: an empirical study	Gerardo Canfora	experiment	Academic	45 Students	Οι PP έχουν μια πιο γρήγορη, μεγάλη και σταθερή ανάπτυξη γνώσης
.S40	Impact of Pair Programming on Thoroughness and Fault Detection Effectiveness of Unit Test Suites	Lech Madeyski	experiment	Academic	N/A	Δεν υπάρχει διαφορά στους δείκτες MSI και BC

.S41	Experimental Evaluation of Pair Programming	Jerzy Nawrocki	experiment	Academic	21 Students	PP και Solo προγραμματιστές χρειάζονται τον ίδιο χρόνο για την ολοκλήρωση των εργασιών με τους PP να κάνουν λιγότερα λάθη. Ο PP είναι αποδοτικός αλλά όχι στα επίπεδα που παρουσιάστηκε από προηγούμενες μελέτες.
.S42	Heterogeneous and homogenous pairs in pair programmnig: An empirical analysis	Maree Mujeeb-u-Rehman	experiment	Academic	6 Students	Τα ετερογενή ζευγάρια έχουν καλύτερη απόδοση, λιγότερα λάθη, καλύτερη έμπνευση και δημιουργικότητα
.S43	Investigating Pair Programming in a Software Engineering Course in an Asian Setting	Raymund Sison	experiment	Academic	24 Students	Οι PP και solo προγραμματιστές είχαν ίδια παραγωγικότητα αλλά οι PP είχαν μικρότερη συχνότητα λαθών
.S44	The Effects of Pair-Programming on Performance in an Introductory Programming Course	Charlie McDowell	experiment	Academic	313 Students	92% PP και 77% solo προγραμματιστές κατάφεραν να περάσουν το μάθημα χωρίς αξιοσημείωτη διαφορά στους τελικούς βαθμούς
.S45	Problems Encountered by Novice Pair Programmers	Brian F. Hanks	experiment	Academic	30 Students	PP και Solo προγραμματιστές κάνουν τα ίδια λάθη. Οι PP όμως έχουν την ικανότητα να διορθώνουν τα περισσότερα από αυτά χωρίς βοήθεια
.S46	The Impact of Pair Programming and Test-Driven Development on Package Dependencies in Object-Oriented Design — An Experiment	Lech Madeyski	experiment	Academic	188 Students	PP και DPP είχαν τα ίδια αποτελεσματα στην ποιότητα του λογισμικού
.S47	Changing Students' Perceptions: An Analysis of the Supplementary Benefits of Collaborative Software Development	Lucas Layman	experiment	Academic	78 Studends	Η προσωπικότητα αλλά και το στυλ μάθησης δεν έχουν σημαντική επίπτωση στην συνεργασία που αναπτύσουν οι PP μεταξύ τους
.S48	Pair Programming vs. Side-by-Side Programming	Jerzy Nawrocki	experiment	Academic	30 Students	ο SbS προγραμματισμός είναι μια ενδιαφέρουσα εναλλακτική του παραδοσιακού προγραμματισμού ανά ζεύγη
.S49	A preliminary study on the impact of a pair design phase on pair programming and solo programming	Matthias M. Muller	experiment	Academic	18 Students	Ίδιος αριθμός λαθών για solo και PP. Ο σχεδιασμός σε ζεύγη είναι εφικτός χωρίς επιπλέον αύξηση κόστους
.S50	An Initial Exploration of the Relationship Between Pair Programming and Brooks' Law	Laurie Williams	survey-case/field study	Industrial	30 Professioals	Εάν μια ομάδα χρειάζεται άμεσα περισσότερο ανθρώπινο δυναμικό αυτό θα πρέπει να αποκτηθεί μέσω του PP
.S51	Productivity of Pair Programming in a Distributed Environment – Results from Two Controlled Case Studies	Sami Pietinen	survey-case/field study	Industrial	N/A	Ο PP καθώς αυξάνει την αυτοπεποίθηση στην δουλειά, την επικοινωνία, την μεταβίβαση γνώσης έχει το κόστος της αυξημένης καταβαλόμενης προσπάθειας
.S52	The Collaborative Nature of Pair Programming	Sallyann Bryant	survey-case/field study	Industrial	72 Professionals	Η συνεργασία μεταξύ PP είναι συνεχώς πάνω από 80% και είναι μεγαλύτερη στο στάδιο του refactoring και της συγγραφής κώδικα
.S53	Pair Programming: What's in it for Me?	Andrew Begel	survey-case/field study	Industrial	487 Answers	Το 64% των εργαζομένων πιστεύουν πως ο PP δουλεύει γι αυτούς και όσο ανεβαίνουμε στην ιεραρχία των εργαζομένων το ποσοστό πέφτει. Η ασυμβατότητα των ζευγών, το αυξημένο κόστος και τα ωράρια είναι τα προβλήματα που συναντήθηκαν
.S54	Increased Retention of Early Computer Science and Software Engineering Students using Pair Programming	Jeffrey C. Carver	survey-case/field study	Academic	95 Students	Ο PP βοήθησε στο να επιλέξουν οι φοιτητές μια "ειδικότητα" σχετική με την επιστήμη των υπολογιστών
.S55	How and Why Collaborative Software Development Impacts the Software Engineering Course	Lucas Layman	survey-case/field study	Academic	192 Students	Φοιτητές με χαμηλή αυτοπεποίθηση προτιμούν τον PP. Ο PP προτείνεται για αύξηση της αυτοπεποίθησης και βελτίωση των εργασιακών συνθηκών
.S56	Issues and Tactics when Adopting Pair Programming: A Longitudinal Case Study	Jari Vanhanen	survey-case/field study	Industrial	35 Professioals	Η ενθάρρυνση έναντι της επιβολής είναι αυτό που βοήθησε στην επικράτηση καλού κλίματος και της θετικής στάσης των συμμετεχόντων απέναντι στον PP
.S57	On Pair Rotation in the Computer Science Course	Hema Srikanth	survey-case/field study	Academic	17 Students	Οι φοιτητές έχουν την δυνατότητα να μοιραστούν γνώση με περισσότερους συμφοιτητές τους.
.S58	Empirical Validation of Test-Driven Pair Programming in Game Development	Shaochun Xu	survey-case/field study	Academic	12 Students	Οι PP χρειάστηκαν λιγότερο χρόνο για την ολοκλήρωση των εργασιών τους έχοντας καλύτερη ποιότητα και συνοχή στα προγράμματά τους σε σχέση με τους solo
.S59	An Empirical Analysis on the Discontinuous Use of Pair Programming	Andrea Janes	survey-case/field study	Academic	15 Students	Αύξηση της ικανότητας επικοινωνίας κατά 92%
.S60	Experiences of Using Pair Programming in an Agile Project	Jari Vanhanen	survey-case/field study	Industrial	4 Professionals	Ο PP βοήθησε στις πολύπλοκες προγραμματιστικές εργασίες αυξάνοντας την ποιότητα του κώδικα και μειώνοντας την προσπάθεια και τα bugs στον κώδικα. Αντίθετα στις απλές εργασίες η προσπάθεια αυξανόταν σε σχέση με τον solo προγραμματισμό
.S61	Pair programming and pair trading Effects on learning and motivation in a cs2 course	Timothy H. DeClue	survey-case/field study	Academic	24 Students	Ο PP βοήθησε να υπάρξει μικρότερος χρόνος ολοκλήρωσης, λιγότερα λάθη και μεγαλύτερη κατανόηση για το πώς αναπτύσσεται το λογισμικό

.S62	Student Attitudes toward Pair Programming	Brian F. Hanks	survey-case/field study	Academic	N/A	Με την βοήθεια του PP οι φοιτητές πιστεύουν ότι έμαθαν περισσότερα μέσω ενός διασκεδαστικού μαθήματος. Υπάρχουν ενδείξεις πως η στάση των φοιτητών απέναντι σε νέες τεχνικές μπορεί να επηρεάζεται από την γνώμη του καθηγητή
.S63	VOICES OF WOMEN IN A SOFTWARE ENGINEERING COURSE: REFLECTIONS ON COLLABORATION	Sarah B. Berenson	survey-case/field study	Academic	3 Students	Η συνάντηση των μελών πρόσωπο με πρόσωπο, η αυξημένη ποιότητα κώδικα και ο μικρότερος χρόνος ολοκλήρωσης των εργασιών έδιναν στις γυναίκες μεγαλύτερη ικανοποίηση και αυτοπεποίθηση
.S64	The Social Dynamics of Pair Programming	Jan Chong	survey-case/field study	Industrial	2 Corporations	Υπάρχουν ασυνέπειες μεταξύ της εφαρμοζόμενης τεχνικής του PP στην βιομηχανία και σε αυτή που περιγράφεται από την βιβλιογραφία
.S65	First Year Students' Impressions of Pair Programming in CS1	Beth Simon	survey-case/field study	Academic	13 Students	Οι solo προγραμματιστές βίωναν πιο έντονο άγχος όταν υπήρχαν προβλήματα στην ανάπτυξη. Γρηγορότερη εύρεση λύσεων για τους PP
.S66	A Multiple Case Study on the Ipact of Pair Programming on Product Quality	Hanna Hulkko	review	Literature	4 Articles	Ο PP μπορεί στοχευμένα να χρησιμοποιηθεί σε συγκεκριμένες φάσεις, δραστηριότητες και εργασίες κατά την διάρκεια της ανάπτυξης
.S67	Are Two Heads Better than One?	Tore Dybå	review	Literature	15 Articles	Ο PP είναι αρκετά αποτελεσματικός όταν υπάρχουν πολύπλοκα προγραμματιστικά προβλήματα και υπάρχει κέρδος χρόνου στα πιο απλοϊκά
.S68	The effectiveness of pair programming: A meta-analysis	Jo E. Hannay	review	Literature	57 Articles	Ο PP είναι ιδανικό να χρησιμοποιείται όταν υπάρχουν χαμηλά επίπεδα πολυπλοκότητας και ο χρόνος είναι η ουσία ή όταν υπάρχουν υψηλά επίπεδα πολυπλοκότητας και η ορθότητα παίζει σημαντικό ρόλο
.S69	Experimenting with Pair Programming in the Classroom	Charlie McDowell	review	Literature	3 Articles	Οι PP έχουν καλύτερη απόδοση στα μαθήματα ποιοτικότερα προγράμματα και αυξημένη απορρόφηση γνώσεων
.S70	In Support of Student Pair-Programming	Laurie Williams	review	Literature	N/A	Οι φοιτητές μέσω του PP ολοκληρώνουν τις εργασίες τους γρηγορότερα με λιγότερο αγχος και περισσότερη χαρά και ικανοποίηση

Appendix B

Διαγράμματα σύγκρισης



Appendix C

Έρευνα	Παραγωγικότητα		Quality		Απόδοση στην εκπαίδευση	
	Possitive Effect	Negative effect	Possitive Effect	Negative effect	Possitive Effect	Negative effect
S3		μειωμένη κατά 13%	αυξημένη κατά 48%			
S4					Από 7% έως 11% αύξηση στο ποσοστό των επιτυχόντων	
S6		μειωμένη κατά 29%	8% λιγότερα σφάλματα			
S9			έως 42% λιγότερα σφάλματα			
S10	Από 2% έως 6% αυξημένη παραγωγικότητα	57% μεγαλύτερη προσπάθεια	50% λιγότερα σφάλματα			
S13	Από 3% έως 61% λιγότερος χρόνος					
S15					από 20% έως 28% αύξηση στο ποσοστό των επιτυχόντων	
S21	20% μείωση του χρόνου στα απλά προβλήματα		48% αύξηση στην εύρεση σωστών λύσεων στα πολύπλοκα προβλήματα			
S24		μειωμένη κατά 13,6%				
S25	29% μείωση του χρόνου ολοκληρωσης όσο το πρόβλημα παράμενε άγνωστο					
S26					10% αύξηση στο ποσοστό των επιτυχόντων	
S27			29% σωστότερες λύσεις με αυξημένο κόστος κατά 13%			
S28		αυξημένη προσπάθεια κατά 4%	40% λιγότερα σφάλματα			
S31					11% αύξηση στο ποσοστό των επιτυχόντων	
S32					12% αύξηση στο ποσοστό των επιτυχόντων	
S35	Από 40% έως 50% λιγότερος χρόνος		Από 13% έως 17% μεγαλύτερη επιτυχία στα test cases			
S36	40% λιγότερος χρόνος					
S38					20% υψηλότερους μέσους όρους στην βαθμολογία	
S43			28% λιγότερα σφάλματα			
S44					15% αύξηση στο ποσοστό των επιτυχόντων	
S51		αυξημένη προσπάθεια κατά 13%				

Appendix D

Πίνακας ποιότητας

ID	Name	Name	Research Design	Sampling	control group	Data collection methods	Data analysis methods	Threats/Bias Discussion	Discussion	Valuable	Sum
.S1	An Empirical Study about the Feelgood Factor in Pair Programming	Matthias M. Muller	1	1	1	1	1	1	1	1	8
.S2	An Empirical Study of the Effects of Personality in Pair Programming using the Five-Factor Model	Norsaremah Salleh	1	1	0	1	1	1	1	1	7
.S3	Analyzing Work Productivity and Program Quality in Collaborative Programming	Rafael Duque, Crescencio Bravo	1	0	1	1	1	0	1	0	5
.S4	Building Pair Programming Knowledge through a Family of Experiments	Laurie Williams	1	1	1	1	1	0	1	1	7
.S5	Critical Personality Traits in Successful Pair Programming	Joseph Chao	1	0	0	1	1	0	1	0	4
.S6	Effects of Pair Programming at the Development Team Level: An Experiment	Jari Vanhanen	1	1	1	0	1	1	1	1	7
.S7	Effects of Personality on Pair Programming	Jo E. Hannay	1	1	1	1	1	1	1	1	8
.S8	Examining the Compatibility of Student Pair Programmers	Laurie Williams	1	1	1	1	1	0	1	1	7
.S9	Investigating the Effect of Pair Programming and Software Size on Software Quality and Programmer Productivity	Raymund Sison	1	1	1	1	1	0	1	1	7
.S10	Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course	Nick Z. Zacharis	1	0	1	0	1	0	1	1	5
.S11	Perceived Effects of Pair Programming in an Industrial Context	Jari Vanhanen	1	1	0	1	1	1	1	1	7
.S12	Distributed Pair Programming: An Empirical Study	Brian F. Hanks	1	0	1	0	0	1	1	0	4
.S13	Empirical Study on the Productivity of the Pair Programming	Gerardo Canfora	1	1	1	0	1	1	1	1	7
.S14	Investigating the Usefulness of Pair-Programming in a Mature Agile Team	Irina Diana Coman	1	1	0	1	1	1	1	1	7
.S15	Pair Learning: With an Eye Toward Future Success	Nachiappan Nagappan	1	0	1	0	1	0	1	1	5
.S16	The Impact of Educational Background on Design Knowledge Sharing During Pair Programming: An Empirical Study	Emilio Bellini	1	0	1	1	0	1	1	1	6
.S17	Do programmer pairs make different mistakes than solo programmers?	Matthias M. Muller	1	0	1	1	1	1	1	1	7
.S18	Pair dynamics in team collaboration	Kyungsub S. Choi	1	0	0	1	1	1	1	1	6
.S19	Virtual Teaming: Experiments and Experiences with Distributed Pair Programming	David Stotts	1	0	1	0	0	1	1	0	4
.S20	An experimental investigation of personality types impact on pair effectiveness in pair programming	Panagiotis Sfetsos	1	1	1	1	1	1	1	1	8
.S21	Evaluating performances of pair designing in industry	Gerardo Canfora	1	1	1	0	1	0	1	1	6
.S22	Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise	Erik Arisholm	1	1	1	1	1	1	1	1	8
.S23	Exploring the underlying aspects of pair programming:The impact of personality	Kyungsub S. Choi	1	1	1	1	1	1	1	1	8
.S24	Pair programming in software development teams – An empirical study of its benefits	Tanja Bipp	1	0	1	1	1	0	1	1	6
.S25	Pair programming productivity: Novice–novice vs. expert–expert	Kim Man Lui	1	1	1	0	0	0	1	1	5
.S26	The Impact of Pair Programming on Student Performance, Perception and Persistence	Charlie McDowell	1	0	1	0	1	0	1	1	5
.S27	Two controlled experiments concerning the comparison of pair programming to peer review	Matthias M. Muller	1	1	1	1	1	0	1	1	7
.S28	A Replicate Empirical Comparison between Pair Development and Software Development with Inspection	Monvorath Phongpaibul	1	0	1	1	1	1	1	1	7
.S29	But, Isn't That Cheating?	Laurie Williams	1	0	1	0	0	0	1	0	3
.S30	Code Warriors and Code-a-Phobes: A Study in Attitude and Pair Programming	Lynda Thomas	1	0	1	0	1	0	1	1	5
.S31	Improving the CS1 Experience with Pair Programming	Nachiappan Nagappan	1	0	1	0	1	1	1	1	6
.S32	Investigating Pair-Programming in a 2nd-year Software Development and Design Computer Science Course	Emilia Mendes	1	0	1	1	1	0	1	1	6
.S33	On Understanding Compatibility of Student Pair Programmers	Neha Katira	1	0	1	1	1	0	1	1	6
.S34	Pair-Programming Helps Female Computer Science Students	Linda L. Werner	1	0	1	0	0	0	1	0	3
.S35	Strengthening the Case for Pair Programming	Laurie Williams	1	0	1	0	0	0	1	1	4

.S36	The Case for Collaborative Programming	John T. Nosek	1	0	1	0	1	0	1	1	5
.S37	Towards Increasing the Compatibility of Student Pair Programmers	Neha Katira	1	1	0	0	0	0	1	1	4
.S38	The Effects of "Pair-Pressure" and "Pair-Learning" on Software Engineering Education	Laurie Williams	1	0	1	0	0	0	1	1	4
.S39	Working in pairs as a means for design knowledge building: an empirical study	Gerardo Canfora	1	1	1	1	1	0	1	1	7
.S40	Impact of Pair Programming on Thoroughness and Fault Detection Effectiveness of Unit Test Suites	Lech Madeyski	1	1	1	1	1	1	1	1	8
.S41	Experimental Evaluation of Pair Programming	Jerzy Nawrocki	1	0	1	1	0	0	1	1	5
.S42	Heterogeneous and homogenous pairs in pair programmnig: An empirical analysis	Maree Mujeeb-u-Rehman	1	0	0	0	1	0	1	0	3
.S43	Investigating Pair Programming in a Software Engineering Course in an Asian Setting	Raymund Sison	1	0	1	0	1	0	1	1	5
.S44	The Effects of Pair-Programming on Performance in an Introductory Programming Course	Charlie McDowell	1	0	1	0	1	0	1	1	5
.S45	Problems Encountered by Novice Pair Programmers	Brian F. Hanks	1	0	1	0	1	1	1	1	6
.S46	The Impact of Pair Programming and Test-Driven Development on Package Dependencies in Object-Oriented Design — An Experiment	Lech Madeyski	1	1	1	1	1	1	1	1	8
.S47	Changing Students' Perceptions: An Analysis of the Supplementary Benefits of Collaborative Software Development	Lucas Layman	1	0	1	1	1	0	1	1	6
.S48	Pair Programming vs. Side-by-Side Programming	Jerzy Nawrocki	1	1	1	1	1	0	1	1	7
.S49	A preliminary study on the impact of a pair design phase on pair programming and solo programming	Matthias M. Muller	1	0	1	1	1	0	1	1	6
.S50	An Initial Exploration of the Relationship Between Pair Programming and Brooks' Law	Laurie Williams	1	1	0	1	1	0	1	1	6
.S51	Productivity of Pair Programming in a Distributed Environment – Results from Two Controlled Case Studies	Sami Pietinen	1	1	1	1	1	1	1	1	8
.S52	The Collaborative Nature of Pair Programming	Sallyann Bryant	1	1	0	1	1	0	1	1	6
.S53	Pair Programming: What's in it for Me?	Andrew Begel	1	1	0	1	1	1	1	1	7
.S54	Increased Retention of Early Computer Science and Software Engineering Students using Pair Programming	Jeffrey C. Carver	1	1	1	1	1	0	1	1	7
.S55	How and Why Collaborative Software Development Impacts the Software Engineering Course	Lucas Layman	1	0	1	1	1	0	1	1	6
.S56	Issues and Tactics when Adopting Pair Programming: A Longitudinal Case Study	Jari Vanhanen	1	0	0	1	1	1	1	1	6
.S57	On Pair Rotation in the Computer Science Course	Hema Srikanth	1	0	0	1	1	1	1	0	5
.S58	Empirical Validation of Test-Driven Pair Programming in Game Development	Shaochun Xu	1	0	1	0	1	1	1	0	5
.S59	An Empirical Analysis on the Discontinuous Use of Pair Programming	Andrea Janes	1	1	0	1	1	0	1	1	6
.S60	Experiences of Using Pair Programming in an Agile Project	Jari Vanhanen	1	0	1	1	0	1	1	1	6
.S61	Pair programming and pair trading Effects on learning and motivation in a cs2 course	Timothy H. DeClue	1	0	0	1	1	0	1	0	4
.S62	Student Attitudes toward Pair Programming	Brian F. Hanks	1	1	0	1	1	0	1	1	6
.S63	VOICES OF WOMEN IN A SOFTWARE ENGINEERING COURSE: REFLECTIONS ON COLLABORATION	Sarah B. Berenson	1	0	1	1	0	0	1	1	5
.S64	The Social Dynamics of Pair Programming	Jan Chong	1	0	0	1	1	0	1	1	5
.S65	First Year Students' Impressions of Pair Programming in CS1	Beth Simon	1	0	0	1	1	1	1	1	6
.S66	A Multiple Case Study on the Ipact of Pair Programming on Product Quality	Hanna Hulkko	1	1	0	1	1	0	1	1	6
.S67	Are Two Heads Better than One?	Tore Dybå	1	0	1	1	1	0	1	1	6
.S68	The effectiveness of pair programming: A meta-analysis	Jo E. Hannay	1	1	0	1	1	0	1	1	6
.S69	Experimenting with Pair Programming in the Classroom	Charlie McDowell	1	0	1	0	1	0	1	1	5
.S70	In Support of Student Pair-Programming	Laurie Williams	1	0	0	0	0	0	1	1	3
											120