Συνέδρια της Ελληνικής Επιστημονικής Ένωσης Τεχνολογιών Πληροφορίας & Επικοινωνιών στην Εκπαίδευση

Τόμ. 1 (2010)

5ο Συνέδριο Διδακτική της Πληροφορικής

Συγκριτική Μελέτη των Εκπαιδευτικών Προγραμματιστικών Περιβαλλόντων BlueJ & jGRASP

Μ. Σατρατζέμη, Σ. Ξυνόγαλος

Βιβλιογραφική αναφορά:

Σατρατζέμη Μ., & Ξυνόγαλος Σ. (2023). Συγκριτική Μελέτη των Εκπαιδευτικών Προγραμματιστικών Περιβαλλόντων BlueJ & jGRASP . *Συνέδρια της Ελληνικής Επιστημονικής Ένωσης Τεχνολογιών Πληροφορίας & Επικοινωνιών στην Εκπαίδευση*, *1*, 429–438. ανακτήθηκε από https://eproceedings.epublishing.ekt.gr/index.php/cetpe/article/view/5173

Συγκριτική Μελέτη των Εκπαιδευτικών Προγραμματιστικών Περιβαλλόντων BlueJ & jGRASP

Μ. Σατρατζέμη[1], Σ. Ξυνόγαλος[2]

[1]Τμήμα Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας,
[2]Τμήμα Διοίκηση Τεχνολογίας, Πανεπιστήμιο Μακεδονίας
{maya, stelios}@uom.gr

Περίληψη

Πολλοί καθηγητές αντιμετωπίζουν σοβαρά προβλήματα όταν διδάσκουν Αντικειμενοστρεφή Προγραμματισμό (ΑΠ). Πολλά από αυτά τα προβλήματα θα μπορούσαν να υπερνικηθούν ή να περιοριστούν με τη χρήση κατάλληλων εργαλείων. Μερικές από τις κύριες απαιτήσεις για ένα περιβάλλον ΑΠ είναι παρόμοιες με εκείνες για τη γλώσσα: πρέπει να παρουσιάζει τις κύριες έννοιες και τα εργαλεία με τρόπο συνεπή, και πρέπει να είναι εύκολο στη χρήση. Αυτό σημαίνει ότι χρειαζόμαστε ένα αντικειμενοστρεφές περιβάλλον που είναι αρκετά απλό για να χρησιμοποιηθεί στη διδασκαλία. Για το σκοπό αυτό αναπτύχθηκε πλήθος Εκπαιδευτικών Προγραμματιστικών Περιβαλλόντων (ΕΠΠ). Δυστυχώς, πολύ λίγα είναι γνωστά για τον εκπαιδευτικό τους αντίκτυπο, ή ακόμη και για το πώς να αξιολογήσουμε αυτόν τον αντίκτυπο. Στην εργασία αυτή παρουσιάζουμε δυο ΕΠΠ, το BlueJ & jGRASP, με βάση τα απαιτούμενα χαρακτηριστικά ενός περιβάλλοντος για τον ΑΠ. Επίσης συνοψίζουμε και αναλύουμε μια αντιπροσωπευτική συλλογή αξιολογήσεων αυτών των περιβαλλόντων, με σκοπό να συνεισφέρουμε σε επόμενες αξιολογήσεις τους.
Λέξεις κλειδιά: *Αντικειμενοστρεφής Προγραμματισμός, Εκπαιδευτικό περιβάλλον, Αξιολόγηση*

Abstract

Many teachers experience serious problems when teaching object-oriented programming. Many of these problems could be overcome or reduced through the use of appropriate tools. Some of the main requirements for an object-oriented environment are similar to those for the language. An OOP environment should present the underlying concepts and tools in a consistent manner, and it should not be overly complicated to use. This means that we need an object-oriented environment which is simple enough to be used for teaching. For this purpose a lot of Educational Programming Environments (EPE) have been developed. Unfortunately, too little is known about the educational impact of these environments, or even how to assess this impact. In this paper we present two EPE, BlueJ & jGRASP, based on the requirements for an environment appropriate for teaching OOP. Also, we summarize a representative collection of the assessments of these two environments, aiming to contribute to their future assessments.
Keywords: *Object-Oriented Programming, Educational Environment, Evaluation*

1. Εισαγωγή

Οι περισσότερες έρευνες σχετικά με τη διδασκαλία και μάθηση του Αντικειμενοστρεφούς Προγραμματισμού (ΑΠ) εστίασαν στη καταγραφή των λανθασμένων αντιλήψεων και δυσκολιών που αντιμετωπίζουν οι αρχάριοι, καθώς και σε προτάσεις προσέγγισης της διδασκαλίας του ΑΠ για την αντιμετώπισή τους. Πολύ λιγότερες ασχολήθηκαν με το χρησιμοποιούμενο προγραμματιστικό περιβάλλον. Ο Kölling (Kölling, 1999) επισημαίνει ότι η επιλογή του κατάλληλου περιβάλλοντος αποτελεί κρίσιμο παράγοντα για την επιτυχία ενός εισαγωγικού μαθήματος. Παρόλα αυτά τα περιβάλλοντα για τη διδασκαλία και μάθηση του προγραμματισμού καθώς και οι απαιτήσεις για τέτοια περιβάλλοντα έχουν πολύ λιγότερο διερευνηθεί απ' ότι οι γλώσσες προγραμματισμού (McIver, 2002). Από την βιβλιογραφία φαίνεται ότι οι ερευνητές ενδιαφέρονται περισσότερο για την εξέταση των γλωσσικών χαρακτηριστικών γνωρισμάτων και πολύ λιγότερο για τη διαπίστωση των πλεονεκτημάτων ή των μειονεκτημάτων των προγραμματιστικών περιβαλλόντων. Τη τελευταία δεκαετία όμως χρησιμοποιούνται οι αντικειμενοστρεφείς γλώσσες προγραμματισμού και η διδασκαλία στρέφεται επιπλέον και σε θέματα όπως: δοκιμή, διόρθωση και επαναχρησιμοποίηση κώδικα. Αυτό δημιουργεί την ανάγκη, αρκετά νωρίς, να εξεταστούν πολλαπλά αρχεία κώδικα και ποικιλία εργαλείων για την ανάπτυξη του προγράμματος. Για να δώσουμε σε έναν σπουδαστή εξ αρχής τη δυνατότητα να αντιμετωπίσει αυτήν την αυξανόμενη πολυπλοκότητα, κρίνεται απαραίτητη

η υποστήριξη του από το κατάλληλο περιβάλλον. Αρκετά νωρίς επισημάνθηκε η ανάγκη για την ανάπτυξη Εκπαιδευτικών Προγραμματιστών Περιβαλλόντων (ΕΠΠ) για τον ΑΠ.

Στην εργασία αυτή γίνεται μια συγκριτική μελέτη δυο ΕΠΠ, του BlueJ και του jGRASP. Η επιλογή αυτών των ΕΠΠ έγινε για τους εξής λόγους: Το BlueJ σήμερα ίσως είναι το πλέον χρησιμοποιούμενο ΕΠΠ (σύμφωνα με τον επίσημο ιστότοπο, www.blue.org, χρησιμοποιείται σε 922 ιδρύματα), οι δημιουργοί του υποστηρίζουν τη διδασκαλία του ΑΠ με ένα βιβλίο μέσω του οποίου προτάθηκε μια νέα προσέγγιση της διδασκαλίας του ΑΠ, δημοσιεύονται άρθρα σχετικά με τη χρήση και αξιολόγηση του, είναι ένα ενεργό project και από το 2009 είναι λογισμικό ελεύθερου και ανοικτού κώδικα. Το jGRASP χρησιμοποιείται (σύμφωνα με τον επίσημο ιστότοπο, www.jrasp.org), σε 319 ιδρύματα-σχολεία, κολλέγια, πανεπιστήμια, δημοσιεύονται άρθρα σχετικά με τη χρήση και αξιολόγηση του, είναι ενεργό project, διατίθεται δωρεάν και υποστηρίζεται ως ερευνητικό project από το NFS. Η εργασία ολοκληρώνεται συνοψίζοντας και αναλύοντας μια αντιπροσωπευτική συλλογή αξιολογήσεων αυτών των περιβαλλόντων, με σκοπό να συνεισφέρουμε σε επόμενες αξιολογήσεις τους.

2. Δυο Ολοκληρωμένα Εκπαιδευτικά Προγραμματιστικά Περιβάλλοντα

Οι απαιτήσεις για ένα ΕΠΠ για τον ΑΠ διατυπώθηκαν από τον Kölling (Kölling, 1999) και είναι:

Ευκολία χρήσης (Ease of use), Ολοκληρωμένα εργαλεία (Integrated tool), υποστήριξη «πρώτα αντικείμενα» (Object-support), υποστήριξη για την επαναχρησιμοποίηση κώδικα (Support for code reuse), υποστήριξη μάθησης προγραμματιστικών εννοιών (Learning support), Διαθεσιμότητα (Availability). Στη συνέχεια αναλύουμε τις κυριότερες απ' αυτές.

1. ***Ευκολία χρήσης****:* Το σύστημα χρειάζεται να είναι αρκετά απλό, για να μπορούν να το χρησιμοποιούν οι αρχάριοι μετά από ένα πολύ σύντομο χρονικό διάστημα, αλλά και αρκετά ισχυρό για να καλύπτει πολλούς από τους στόχους της ανάπτυξης λογισμικού εύκολα ή αυτόματα.
2. ***Υποστήριξη «πρώτα αντικείμενα****:* Το αντικειμενοστρεφές παράδειγμα είναι βασισμένο στην ιδέα ότι τα αντικείμενα υπάρχουν ανεξάρτητα το ένα από το άλλο, και διάφορες λειτουργίες μπορούν να εκτελεσθούν σ' αυτά. Συνεπώς, ένας χρήστης σε ένα πραγματικά αντικειμενοστρεφές περιβάλλον ανάπτυξης πρέπει να είναι σε θέση να δημιουργήσει με αλληλεπίδραση αντικείμενα οποιασδήποτε διαθέσιμης κλάσης, να τα χειριστεί και να καλέσει τις μεθόδους τους. Η σύνθεση των αντικειμένων στο επίπεδο χρηστών πρέπει να είναι δυνατή. Αυτή η δυνατότητα είναι γνωστή και *ως περιβάλλον προσανατολισμένο στο στιγμιότυπο* (*instance-centred environment)*. Μια τέτοια δυνατότητα, εάν παρέχεται, οδηγεί στη δυνατότητα της σταδιακής ανάπτυξης εφαρμογών. Οποιαδήποτε μεμονωμένη κλάση μπορεί να εξεταστεί ανεξάρτητα μόλις ολοκληρώνεται η υλοποίηση της. Έτσι, ο έλεγχος γίνεται πιο εύκολος απ' ότι στα διαδικαστικά συστήματα. Δυστυχώς, στα περισσότερα επαγγελματικά αντικειμενοστρεφή περιβάλλοντα, τα αντικείμενα πρέπει να δημιουργούνται μέσω μιας μη-αντικειμενοστρεφούς λειτουργίας (μέθοδος main). Εν τέλει ο προγραμματιστής πρέπει να χρησιμοποιήσει το διαδικαστικό παράδειγμα προγραμματισμού για την υλοποίηση αλλά και τον έλεγχο ενός προγράμματος. Κάτι τέτοιο απαιτεί δύο τρόπους σκέψεις: μια σκέψη για το μοντέλο της εφαρμογής και μια για τις αλληλεπιδράσεις με το περιβάλλον. *Για να αποφευχθεί αυτό το πρόβλημα, οι κύριες αφαιρέσεις, που θα χρησιμοποιηθούν κατά την αλληλεπίδραση του χρήστη με το περιβάλλον, πρέπει να είναι οι κλάσεις και τα αντικείμενα και θα πρέπει να αντιμετωπιστούν ως αντικείμενα σε επίπεδο χρήστη, πάνω στα οποία θα μπορεί να εκτελέσει λειτουργίες αλληλεπιδρώντας με τις διεπαφές τους.*
3. ***Υποστήριξη μάθησης προγραμματιστικών εννοιών (Learning support)***
   3.1 Αλληλεπίδραση/πειραματισμός (Interaction/experimentation) Να διευκολύνει την αλληλεπίδραση με τις κλάσεις και τα αντικείμενα ώστε να υποστηρίζει την κατανόηση των αντικειμενοστρεφών εννοιών. Η δυνατότητα για παράδειγμα να δημιουργούνται πολλά αντικείμενα από μια κλάση, με αλληλεπίδραση, και να διαπιστώνεται ότι αυτά τα αντικείμενα συμπεριφέρονται με τον ίδιο τρόπο αν και έχουν διαφορετική υπόσταση και κατάσταση, διευκολύνει τη κατανόηση της διαφοράς ανάμεσα στην κλάση και στα αντικείμενα αυτής.
   3.2 Οπτικοποίηση (Visualisation) Οι έννοιες κλάση και αντικείμενο πρέπει να απεικονίζονται στην οθόνη. Σε πολλές έρευνες διαπιστώθηκε ότι οι σπουδαστές συναντούν δυσκολίες με την

κατανόηση αυτών των εννοιών και της σχέσης που τις συνδέει. Ο λόγος είναι ότι το μόνο που βλέπουν στην οθόνη είναι γραμμές κώδικα. Εάν θέλουμε οι σπουδαστές να σκέφτονται και να μιλούν για τις κλάσεις, και τις σχέσεις μεταξύ αυτών, πρέπει το περιβάλλον να τις αναπαριστά οπτικά. Το ίδιο μπορεί να ειπωθεί και για τα αντικείμενα κατά το χρόνο εκτέλεσης: οι σχέσεις μεταξύ των αντικειμένων γίνονται καλύτερα κατανοητές όταν είναι ορατές. Ένα περιβάλλον πρέπει να χρησιμοποιεί για τις αναπαραστάσεις των εννοιών αυτών τόσο το σχήμα όσο και το κείμενο. Η διαχείριση της γραφικής ή της αναπαράστασης με κείμενο μιας κλάσης ή ενός αντικειμένου πρέπει να είναι δυνατή εναλλακτικά.

Σήμερα, αν και αναπτύχθηκαν αρκετά ΕΠΠ για τον ΑΠ, όπως το BlueJ, jGRASP, DrJava κ.ά. (Georgantaki & Retalis, 2007) εν τούτοις, πολύ λίγα έχουν γίνει για την μελέτη της εκπαιδευτικής τους επίδρασης στη διδασκαλία και μάθηση του ΑΠ. Αυτό που είναι γνωστό για τον πραγματικό αντίκτυπο αυτών των περιβαλλόντων είναι ιδιαίτερα περιορισμένο. Όπως αναφέρει ο Guzdial (Guzdial, 2004) «*η μέγιστη συνεισφορά σ' αυτόν τον τομέα δεν είναι να αναπτυχθούν και άλλα ΕΠΠ αλλά να μελετήσουμε αυτά που έχουμε*». Οι Gross, Powers στην εργασία τους (Gross & Powers, 2005), παρουσιάζουν μελέτες που έχουν διενεργηθεί για την αξιολόγηση 5 ΕΠΠ. Στην εργασία τους γίνεται φανερό ότι οι διάφορες έρευνες δε χρησιμοποιούν την ίδια μεθοδολογία αξιολόγησης των ΕΠΠ και γι' αυτό προτείνουν ένα πλαίσιο τυποποίησης των κριτηρίων που πρέπει να χρησιμοποιούνται για να αποτιμάται η ποιότητα των αξιολογήσεων των ΕΠΠ.

Στην ενότητα αυτή γίνεται μια συγκριτική παρουσίαση των ΕΠΠ BlueJ και jGRASP, με βάσει τα χαρακτηριστικά που πρέπει να διαθέτουν τα ΕΠΠ και τα οποία αναλύθηκαν παραπάνω. Η παρουσίαση γίνεται με αναφορές σε δυο στιγμιότυπα (Σχήμα 1 και Σχήμα 2) που προέρχονται από το ίδιο εκπαιδευτικό σενάριο, την εφαρμογή dome_v2, από το βιβλίο των (Barnes & Kölling, 2006), την οποία χρησιμοποιούν για τη διδασκαλία της κληρονομικότητας και του πολυμορφισμού.

2.1 BlueJ

Το BlueJ σήμερα θεωρείται κάτι περισσότερο από ΕΠΠ για την Java, αφού αναφέρεται σε ένα σύνολο από παραδείγματα και πηγές (www.bluej.org), ένα βιβλίο (Barnes & Kölling, 2006) για τη διδασκαλία και μάθηση του ΑΠ με τη Java, ένα πλαίσιο παιδαγωγικών αρχών και μια ιδιαίτερη παιδαγωγική προσέγγιση για την εισαγωγή στον ΑΠ (Kölling & Rosenberg, 2001). Το γραφικό ενδιάμεσο βοηθά τη διδασκαλία και μάθηση της αντικειμενοστρεφούς σχεδίασης και υποστηρίζει την προσέγγιση «πρώτα-αντικείμενα».

Στο βασικό παράθυρο του BlueJ (BlueJ Version 2.5.3, 8 Oct 2009) εμφανίζεται ένα απλοποιημένο UML διάγραμμα κλάσεων του ανοικτού project (Σχήμα 1, Β1) με τη βοήθεια του οποίου καταδεικνύεται εξ αρχής ότι μια εφαρμογή είναι ένα σύνολο από κλάσεις, ενώ στο *object bench*, κάτω τμήμα του βασικού παραθύρου, εμφανίζονται, ως εικονίδια, τα αντικείμενα που δημιουργούνται από κάθε κλάση. Αυτή η οπτικοποίηση επιδιώκει να δημιουργήσει μια νοερή (mental) έμφαση της σχέσης ανάμεσα στην κλάση και στο αντικείμενο. Με άμεση διαχείριση επάνω στο εικονίδιο της κάθε κλάσης (*διπλό κλικ ή δεξί κλικ του ποντικιού και επιλογή από αναδυόμενο μενού*): εμφανίζει τον κώδικα της αντίστοιχης κλάσης (Σχήμα 1, Β7), μεταγλωττίζει τη κλάση, δημιουργεί αντικείμενο της κλάσης το οποίο εμφανίζεται στο *object bench*, στο κάτω τμήμα του βασικού παραθύρου (Σχήμα 1, Β2). Με άμεση διαχείριση επάνω στο εικονίδιο του κάθε αντικειμένου επιτρέπει: την κλήση οποιασδήποτε από τις δημόσιες μεθόδους που ορίστηκαν στη κλάση του αντικειμένου, την εμφάνιση της κατάστασης του αντικειμένου μέσω της επιλογής *Inspect,* (Σχήμα 1, Β3).

2.2 jGRASP

Στο βασικό παράθυρο του jGRASP (έκδοση: jGRASP 1.8.7_07, December 10, 2009) εμφανίζεται το UML διάγραμμα των κλάσεων του ανοικτού project (Σχήμα 2, J1). Στο αριστερό τμήμα του βασικού παραθύρου, ανάλογα με την καρτέλα που είναι ορατή, εμφανίζεται είτε η δομή του ανοικτού project (*καρτέλα Browse*), είτε πληροφορίες για την αποσφαλμάτωση ή τη βηματική εκτέλεση μιας εφαρμογής (*καρτέλα Debug*), είτε τα δημιουργημένα αντικείμενα (*καρτέλα Workbench*). Στο τμήμα όπου εμφανίζεται το UML διάγραμμα ο χρήστης έχει άμεση διαχείριση αφού σύροντας και αποθέτοντας το αρχείο μιας κλάσης (από τη *καρτέλα Browse*) μπορεί να την προσθέσει σ' αυτό. Μετά

από κάθε μεταγλώττιση δημιουργούνται οι εξαρτήσεις των κλάσεων και αναπαρίστανται γραφικά (κληρονομικότητα, συσχέτιση, διασύνδεση).

Με άμεση διαχείριση επάνω στο εικονίδιο της κάθε κλάσης *(διπλό κλικ ή δεξί κλικ του ποντικιού και επιλογή από αναδυόμενο μενού*): εμφανίζει πληροφορίες για τη δομή της κάθε κλάσης (πεδία, κατασκευαστές, μέθοδοι) στην καρτέλα *UML info*, εμφανίζει τον κώδικα της αντίστοιχης κλάσης (Σχήμα 2, J7), μεταγλωττίζει τη κλάση, δημιουργεί αντικείμενο της κλάσης και το εμφανίζει σε δυο μορφές, με τη μορφή εικονιδίου (μικρό ορθογώνιο) και με τη μορφή κειμένου, στην *καρτέλα Workbench* η οποία γίνεται αυτόματα ορατή (Σχήμα 2, J2 & J3). Με άμεση διαχείριση επάνω στο εικονίδιο του κάθε αντικειμένου επιτρέπει: την κλήση οποιασδήποτε από τις δημόσιες μεθόδους που ορίστηκαν στη κλάση του αντικειμένου, την εμφάνιση της κατάστασης του αντικειμένου στην καρτέλα *Workbench* αυτόματα όταν αυτό δημιουργείται και σε ξεχωριστό παράθυρο (*Viewer*) (Σχήμα 2, J3), όπου επίσης εμφανίζει οπτικοποιημένη μια σύνθετη δομή, όπως αντικείμενα της κλάσης-συλλογής ArrayList (Σχήμα J4.1), τη δυναμική διαχείριση των αντικειμένων (παράθυρο *Viewer*). Ο χρήστης μπορεί: (1) να αλλάξει το τύπο του αντικειμένου σε οποιοδήποτε συμβατό τύπο (επιλογή *Type*, Σχήμα 2, J8) και, (2) να αλλάξει το πλαίσιο πρόσβασης του αντικειμένου (επιλογή *Accessibility*, Σχήμα 2, J8). Αυτές οι επιλογές επιτρέπουν στο χρήστη να ερευνήσει τις σχέσεις πρόσβασης και ορατότητας των αντικειμένων πειραματιζόμενος με οποιοδήποτε αντικείμενο που εμφανίζεται στη καρτέλα *Workbench* ή *Debug*. Οι πειραματισμοί αυτοί βοηθούν στην κατανόηση των εννοιών της κληρονομικότητας και του πολυμορφισμού (Cross et al., 2008).

2.3 Συγκριτική παρουσίαση

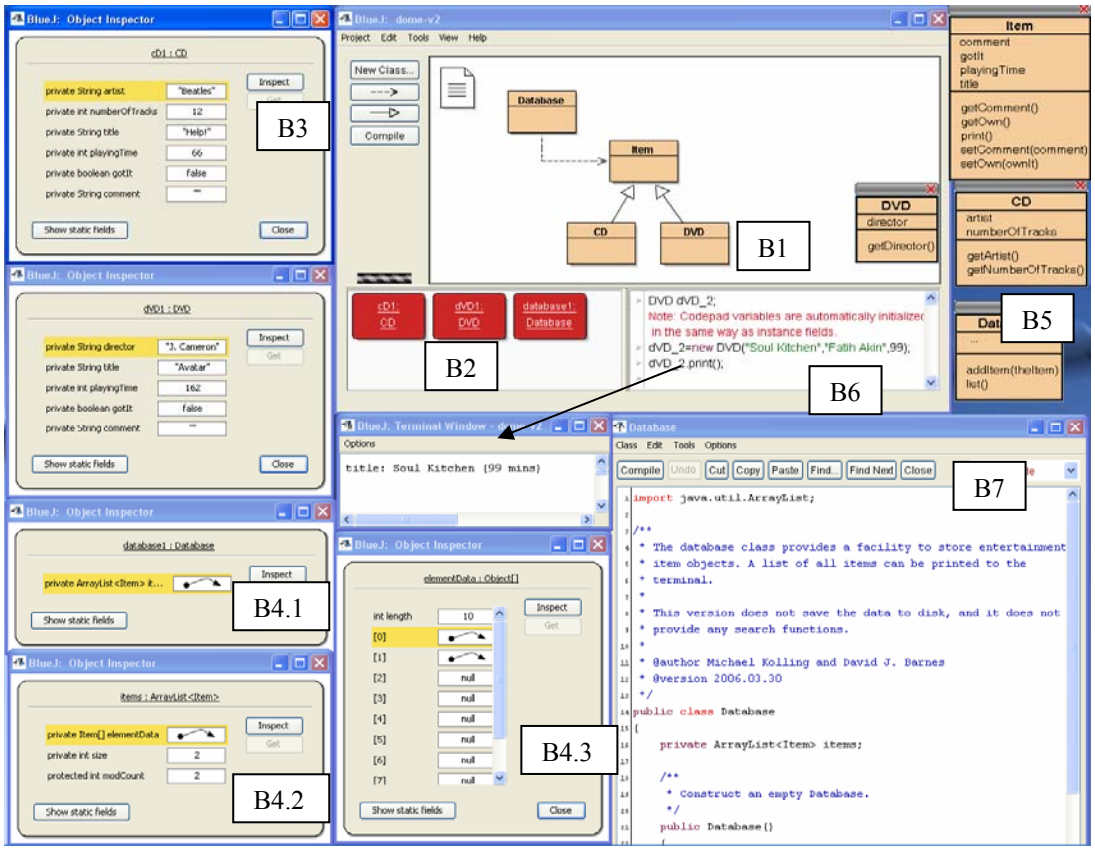
Στον Πίνακα 1 έχουν συγκεντρωθεί οι λειτουργίες εκείνες που υποστηρίζουν το αντικειμενοστρεφές παράδειγμα προγραμματισμού (Object support) και βοηθούν στην μάθηση του (learning support). Για κάθε λειτουργία δίνεται και ο αριθμός, μέσα σε παρένθεση, της απαίτησης, όπως ορίσθηκε από τον Kölling και παρουσιάσθηκε παραπάνω. Στη 3η και 4η στήλη του Πίνακα 1 δίνονται τα σημεία, σε σχέση με τα Σχήματα 1 και 2, όπου φαίνεται το κάθε περιβάλλον πως υποστηρίζει την αντίστοιχη λειτουργία.

Πίνακας 1. *Συγκριτική παρουσίαση λειτουργιών του BlueJ & jGRASP*

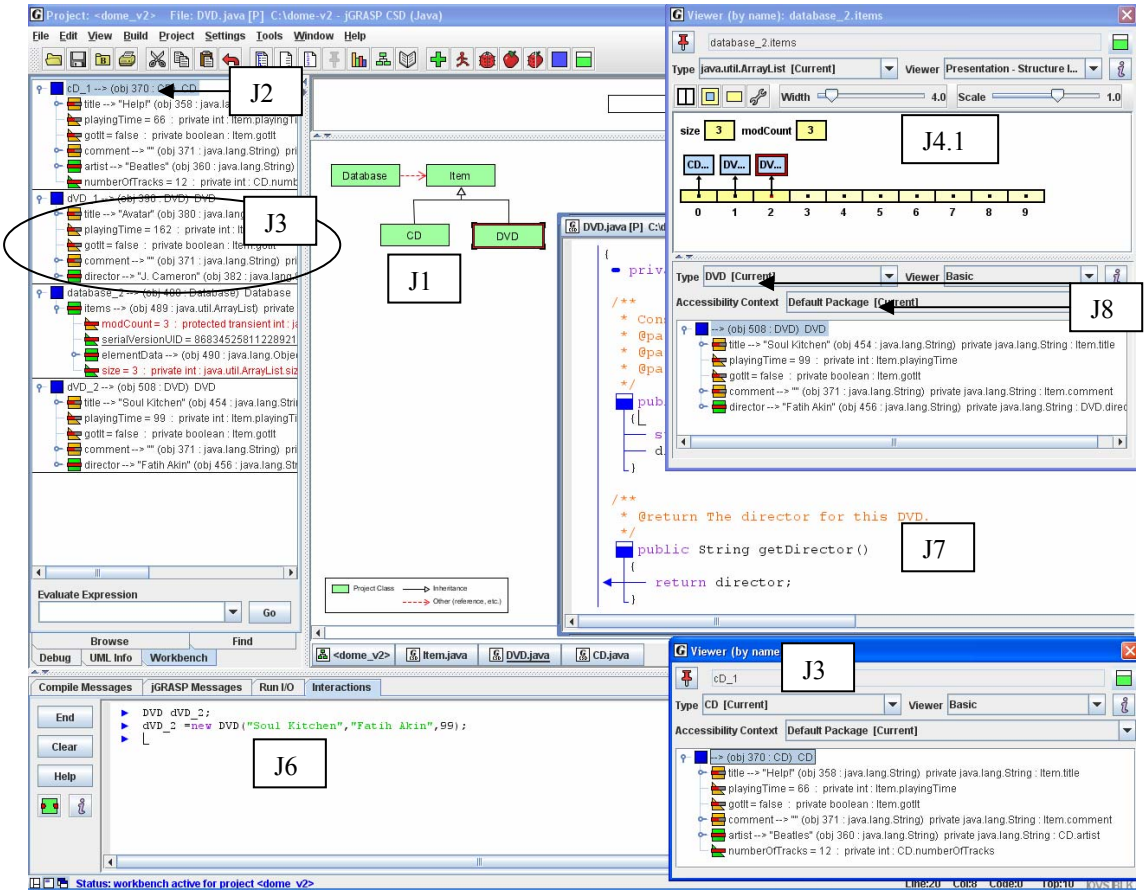
	Λειτουργίες	**BlueJ**	**jGRASP**
1	Οπτικοποίηση κλάσης – αντικειμένου (2)	B1 – B2	J1 – J2
2	Οπτικοποίηση κατάστασης αντικειμένου (3.2)	B3	J3
3	Οπτικοποίηση κατάστασης αντικειμένου-συλλογή (πχ. ArrayList) (3.2)	B4.1, B4.2, B4.3	J4.1
4	Αναπαράσταση δομής κλάσης (3)	- (από πρόσθετο-*Class card*) B5	καρτέλα *UML info**
5	Εκτέλεση μεμονωμένης εντολής/ έκφρασης (java interpreter) (3.1)	B6 (*Code Pad)*	J6 (καρτέλα *Interactions*)
6	Διάγραμμα δομής κώδικα (CSD) (3)	-	J7
7	Κλήση της μεθόδου main με άμεση διαχείριση (2)	- (από πρόσθετο)	Ναι
8	Δυναμική διαχείριση αντικειμένων (κατά την εκτέλεση, αλλαγή του τύπου του αντικειμένου, ή/και του τροποποίητη πρόσβασης), (3.2)	-	J8

* λόγω έλλειψης χώρου δεν δίνεται στιγμιότυπο

Στη συνέχεια αναλύονται οι σημαντικότερες λειτουργίες, όπως οι 2, 3 & 5 καθώς και οι λειτουργίες σχετικά με την ευκολία χρήσης και τη διεπαφή των δυο περιβαλλόντων:



Σχήμα 1: *Το γραφικό ενδιάμεσο του BlueJ*



Σχήμα 2: *Το γραφικό ενδιάμεσο του jGRASP*

Οπτικοποίηση κατάστασης αντικείμενου & αντικειμένου-συλλογή (πχ. ArrayList)

BlueJ

Όταν δημιουργείται ένα αντικείμενο η κατάσταση του δεν γίνεται άμεσα ορατή. Χρειάζεται να κληθεί η λειτουργία Inspect με άμεση διαχείριση του εικονιδίου του αντικειμένου. (Μειονέκτημα)

Όταν μεταβάλλεται η τιμή ενός πεδίου αντικειμένου ενημερώνεται δυναμικά και αντιστοίχως μεταβάλλεται η τιμή του πεδίου στο αντίστοιχο παράθυρο Object Inspector (εφόσον το παράθυρο Object Inspector είναι ανοικτό).

Για την εμφάνιση της κατάστασης ενός αντικειμένου-συλλογή (πχ αντικείμενο της κλάσης ArrayList) θα χρειαστεί ο χρήστης να ανοίξει 3 παράθυρα (βλέπε Σχήμα 1, σημεία B4.1, B4.2, B4.3) και η οπτικοποίηση του αντικειμένου-συλλογή, πχ αντικείμενο της κλάσης ArrayList, δεν είναι αντίστοιχη με τη νοερή αναπαράσταση των πινάκων ή των λιστών. (*Μειονέκτημα*)

jGRASP

Η δημιουργία ενός αντικειμένου προκαλεί και την αυτόματη εμφάνιση της καρτέλας Workbench και την εμφάνιση του αντικειμένου σ' αυτήν χωρίς επιπλέον ενέργειες. (*Πλεονέκτημα*)

Όταν μεταβάλλεται η τιμή ενός πεδίου αντικειμένου ενημερώνεται δυναμικά και αντιστοίχως μεταβάλλεται η τιμή του πεδίου στο παράθυρο του Workbench ή στο αντίστοιχο παράθυρο Viewer και η μεταβολή σημειώνεται με κόκκινο χρώμα. (*Πλεονέκτημα*)

Για την εμφάνιση της κατάστασης ενός αντικειμένου-συλλογή (πχ αντικείμενο της κλάσης ArrayList) θα χρειαστεί ο χρήστης να ανοίξει ένα παράθυρο, σύροντας το εικονίδιο του αντίστοιχου πεδίου του αντικειμένου (εμφάνιση του παραθύρου Viewer, βλέπε Σχήμα 2, J4.1). Η οπτικοποίηση είναι αντίστοιχη με τη νοερή αναπαράσταση των πινάκων ή των λιστών. (*Πλεονέκτημα*)

Αλληλεπίδραση/Πειραματισμός

BlueJ

Επιτρέπει τη σύνταξη μεμονωμένης εντολής, μέσω του Code Pad, χωρίς να ενημερώνεται το γραφικό ενδιάμεσο, πχ. η δημιουργία αντικειμένου με σύνταξη εντολής στο Code Pad, δεν εμφανίζει το δημιουργημένο αντικείμενο στο Object bench (Σχήμα 1, B6). (*Μειονέκτημα*)

jGRASP

Επιτρέπει τη σύνταξη μεμονωμένης εντολής, καρτέλα Interactions, και ενημερώνεται άμεσα και δυναμικά το γραφικό ενδιάμεσο, πχ. η δημιουργία αντικειμένου με σύνταξη εντολής στη καρτέλα Interactions εμφανίζει το δημιουργημένο αντικείμενο στο Workbench ή στο αντίστοιχο παράθυρο Viewer (Σχήμα 2, J6). (*Πλεονέκτημα*)

Ευκολία χρήσης - Γραφικό ενδιάμεσο

BlueJ

Σε διαφορετικά παράθυρα εμφανίζεται το βασικό παράθυρο με το διάγραμμα κλάσεων και το object bench, το αρχείο κώδικα κάθε κλάσης, η κατάσταση κάθε αντικειμένου (σε ένα η περισσότερα ανάλογα αν αναφέρονται σε αντικείμενα άλλης κλάσης), Debugger. Ενδεχόμενη δυσκολία για την ταυτόχρονη εμφάνιση διαφορετικών αναπαραστάσεων διαφορετικών εννοιών, αφού απαιτείται η διαχείριση από το χρήστη της διευθέτησης των διαφορετικών παραθύρων. (*Μειονέκτημα*)

Η κατασκευή του UML διαγράμματος κλάσεων και των σχέσεων μεταξύ αυτών (κληρονομικότητα, διασύνδεση, ή εξάρτηση) γίνεται με άμεση διαχείριση (τοποθέτηση τόξων μεταξύ εικονιδίων κλάσεων) και άμεση παραγωγή/ενημέρωση του κώδικα. (*Πλεονέκτημα*)

Η εμφάνιση της δομής μιας κλάσης (κατασκευαστές, πεδία, μέθοδοι) δεν συμπεριλαμβάνεται στο βασικό πακέτο του BlueJ, αλλά σε πρόσθετο εμφανίζοντας τις πληροφορίες κάθε κλάσης σε διαφορετικό παράθυρο. (Μειονέκτημα)

jGRASP

Στο ίδιο παράθυρο είναι δυνατόν να εμφανίζονται 3 διαφορετικές όψεις εννοιών. πχ το διάγραμμα κλάσεων, το Workbench και το αρχείο κώδικα, λόγω της χρήσης καρτελών που αναπτύσσονται στα 3 διαφορετικά τμήματα του βασικού παραθύρου. Διευκολύνει στην μάθηση των εννοιών του ΑΠ αφού επιτρέπει την εύκολη και ταυτόχρονη εμφάνιση της έννοιας σε μορφή εικόνας και σε μορφή κειμένου και δεν απαιτεί την διευθέτηση παραθύρων. (*Πλεονέκτημα*)

Υποστηρίζει άμεσα την εμφάνιση της δομής μιας κλάσης (κατασκευαστές, πεδία, μέθοδοι) στη καρτέλα UML info. (*Πλεονέκτημα*)

Η κατασκευή του UML διαγράμματος κλάσεων και των σχέσεων μεταξύ αυτών (κληρονομικότητα, διασύνδεση, ή εξάρτηση) γίνεται με άμεση διαχείριση και αυτόματα από το σύστημα αλλά χωρίς την άμεση ενημέρωση του κώδικα. (*Μειονέκτημα*)
Εμφανίζει με 3 διαφορετικούς τρόπους τη μορφή του UML διαγράμματος. Εμφανίζει με διαφορετικά χρώματα τις κλάσεις του χρήστη από τις JDK κλάσεις. Η οπτικοποίηση του UML διαγράμματος είναι παραμετροποιημένη, ώστε να εμφανίζει σε απλοποιημένη μορφή τις σχέσεις μεταξύ των κλάσεων (πχ μόνο κληρονομικότητα) για αρχάριους χρήστες ή σε εμπλουτισμένη μορφή για προχωρημένους χρήστες. (*Πλεονέκτημα*)

3. Αξιολογήσεις

3.1 Αξιολογήσεις του BlueJ

Οι Haaster και Hagan (Haaster & Hagan, 2004) πραγματοποίησαν έρευνα αξιολόγησης της αποτελεσματικότητας του BlueJ στην υποστήριξη των σπουδαστών στην μάθηση του ΑΠ σε φοιτητές του Πανεπιστημίου Monash που είχαν συμπληρώσει 2 εξάμηνα διδασκαλίας του ΑΠ με Java. Συμμετείχαν 115 φοιτητές και τα δεδομένα συγκεντρώθηκαν από τις απαντήσεις των φοιτητών σε ερωτηματολόγια. Τα βασικά κριτήρια που τέθηκαν για την αξιολόγηση του BlueJ ήταν: Ευχρηστία: αν είναι διαθέσιμο και εύκολο στην εγκατάσταση, αν είναι σταθερό και προβλέψιμη η συμπεριφορά του, αν επιτρέπει την εξατομίκευση, αν υποστηρίζει τόσο τους αρχάριους όσο και τους ειδικούς χρήστες Υποστήριξη του Παραδείγματος προγραμματισμού: αν χρησιμοποιεί μια τυποποιημένη έκδοση της γλώσσας προγραμματισμού, αν χρησιμοποιεί τους τυποποιημένους συμβολισμούς και ορολογία, αν συνδέει επιτυχώς το κώδικα και την οπτικοποίηση του. Από τους σπουδαστές ζητήθηκε να αξιολογήσουν τις διάφορες οπτικοποιήσεις του BlueJ. Το 91% βρήκε χρήσιμο το τονισμό των λέξεων στη σύνταξη του προγράμματος και πάνω από το 60% βρήκε χρήσιμο καθένα από τα υπόλοιπα χαρακτηριστικά.

Οι (Ragonis & Ben-Ari, 2005) πραγματοποίησαν έρευνα για να διαπιστώσουν αν είναι εφικτό να διδαχθεί ο ΑΠ σε μαθητές λυκείου. Η έρευνα διενεργήθηκε σε 18 μαθητές το 2000-1 και σε 29 μαθητές το 2001-2 και χρησιμοποιήθηκε το BlueJ. Το αντικείμενο της μελέτης τους ήταν να ερευνηθεί η διαδικασία της μάθησης και της κατανόησης του μαθησιακού υλικού, παρά να αξιολογηθεί το συγκεκριμένο μάθημα ή το επίπεδο της απόδοσης των μαθητών. Οι Ragonis & Ben-Ari αναφέρουν σχετικά για το BlueJ: «*Ένα κύριο χαρακτηριστικό γνώρισμα του BlueJ είναι η στατική οπτικοποίηση των κλάσεων και των αντικειμένων, και η άμεση αλληλεπίδραση με τα εικονίδια αυτά, διευκολύνοντας την κατανόηση των βασικών εννοιών του ΑΠ. Επιπλέον, όμως, οι σπουδαστές πρέπει να κατανοήσουν και τη ροή εκτέλεσης ενός προγράμματος, η οποία στον ΑΠ είναι αρκετά σύνθετη. Διδάσκοντας ΑΠ χρησιμοποιώντας το BlueJ, εντοπίσαμε σοβαρές μαθησιακές δυσκολίες σχετικά με τη δυναμική ενός προγράμματος, τις οποίες το BlueJ επιδείνωσε.*» Δίνουμε αποσπάσματα από τα προβλήματα που κατέγραψαν για κάθε τύπο δυσκολίας. Δυσκολία 1-Κατάσταση αντικειμένου Δυσκολεύονται να αντιληφθούν ότι η κλήση μιας μεθόδου επιδρά στη κατάσταση του αντικείμενου. *Αν και υπάρχει η δυνατότητα να χρησιμοποιηθεί η επιλογή «inspect» πριν και μετά από την κλήση μιας μεθόδου η δυνατότητα αυτή δεν βοήθησε τους σπουδαστές να κατανοήσουν την έννοια της αλλαγής της κατάστασης ενός αντικειμένου η οποία μεταβάλλεται κατά τη διάρκεια της εκτέλεσης μιας μεθόδου.*» Δυσκολία 2 & 3-Κλήση μεθόδου & Παράμετροι «*Κατά την κλήση μιας μεθόδου με παραμέτρους, το BlueJ εμφανίζει ένα πλαίσιο στο οποίο ο χρήστης μπορεί να δώσει τιμές στις παραμέτρους. Η επιλογή αυτή δημιουργεί προβλήματα στους σπουδαστές.* Επίσης δυσκολεύονται να κατανοήσουν από που προέρχονται οι τιμές των παραμέτρων. *Εάν οι σπουδαστές βλέπουν συχνά σενάρια στα οποία οι παράμετροι παίρνουν τις τιμές από το περιβάλλον, ξεχνούν ή και δεν αντιλαμβάνονται το σενάριο της εκτέλεσης μιας ακολουθίας κλήσεων μεθόδων με τις παραμέτρους τους.*» Δυσκολία 4-Επιστρεφόμενες τιμές Δεν κατανοούν που επιστρέφονται οι τιμές μεθόδων. *Η κλήση μιας non-void μεθόδου μέσω αναδυόμενου μενού προκαλεί την εμφάνιση ενός πλαισίου όπου εμφανίζεται η επιστρεφόμενη τιμή. Αυτή η δυνατότητα προκάλεσε σύγχυση στην κατανόηση της ροής δεδομένων κατά την εκτέλεση μιας ακολουθίας εντολών. Οι σπουδαστές κατανόησαν κατά τρόπο τοπικό ότι «η μέθοδος επιστρέφει μια τιμή» αλλά δεν κατανόησαν τη χρήση μιας επιστρεφόμενης τιμής από μια non-void μέθοδο. Επιπλέον, η εμφάνιση της επιστρεφόμενης τιμής στο πλαίσιο ερμηνεύθηκε ως μια εντολή ανάλογη με αυτήν της εμφάνισης τιμής.*» Δυσκολία 6-Κατασκευαστές: «*Όταν καλείται ο κατασκευαστής για να δημιουργηθεί*

ένα αντικείμενο του οποίου οι παράμετροι είναι αντικείμενα, το BlueJ επιτρέπει τη διαλογική ανάθεση υπαρχόντων αντικειμένων στα πεδία, κλικάροντας το εικονίδιο του αντικειμένου. Κατά συνέπεια, οι σπουδαστές είδαν «τη μισή» από την παραπάνω εντολή όταν χρησιμοποίησαν το περιβάλλον για να δημιουργήσουν ένα αντικείμενο, και συγχέουν τη «χειρωνακτική» δημιουργία αντικειμένου (με τη βοήθεια του BlueJ) με τη «πραγματική» δημιουργία αντικειμένου.»

Οι (Xinogalos, Satratzemi, & Dagdilelis, 2006; Xinogalos, Satratzemi, & Dagdilelis, 2008) πραγματοποίησαν διάφορες έρευνες σχετικά με τη διδασκαλία του ΑΠ και τη χρήση του BlueJ. Η 1^{η} έρευνα διενεργήθηκε σε 45 δευτεροετείς φοιτητές του τμήματος Διοίκησης Τεχνολογίας, με σκοπό να εκτιμηθεί η επίδρασης της διδακτικής προσέγγισης των Kolling & Barnes, όπως παρουσιάζεται στο βιβλίο τους (Barnes & Kölling, 2006), σε συνδυασμό με τη χρήση του BlueJ, στη διδασκαλία του ΑΠ. Μεταξύ των δυσκολιών που κατεγράφησαν ήταν και αυτές που αποδόθηκαν στο BlueJ. Παρατήρησαν ότι οι σπουδαστές αντιμετώπισαν δυσκολίες στην εξαγωγή πληροφοριών από τον κώδικα ενός προγράμματος – ενώ αντίθετα εξήγαγαν εύκολα τις πληροφορίες από τα διαγράμματα που δημιουργήθηκαν στο BlueJ. Ακόμη, παρατηρήθηκαν παρανοήσεις σχετικά με τη δημιουργία αντικειμένου, αφού σε σχετική ερώτηση, οι σπουδαστές απαντούσαν για το πώς δημιουργείται ένα αντικείμενο στο BlueJ, δηλαδή περιέγραφαν τις ενέργειες πάνω στο εικονίδιο και όχι τη σχετική εντολή. Δηλαδή, παρατηρήθηκε η ίδια συμπεριφορά μ' αυτή που αναφέρουν οι (Ragonis & Ben-Ari, 2005), (Δυσκολία 6). Επίσης παρατηρήθηκε ότι οι σπουδαστές τις περισσότερες φορές έγραφαν την εντολή δημιουργίας του αντικειμένου χωρίς προηγούμενη δήλωση του. Πράγμα που αποδόθηκε στο γεγονός ότι μέσω της διεπαφής του BlueJ, επιτρέπεται η κατασκευή αντικειμένου με άμεση διαχείριση του εικονιδίου και η αντίστοιχη δήλωση του αντικειμένου δημιουργείται αυτόματα και «σιωπηλά» από το ίδιο το σύστημα. Οι περισσότεροι από τους σπουδαστές αντιμετώπισαν δυσκολίες στον ορισμό μεθόδων που επέστρεφαν αντικείμενο της κλάσης ArrayList καθώς και με τη κλήση τους από την main(): (i) η εντολή return έλειπε (ii) η μέθοδος κλήθηκε από τη main ως μέθοδος void. Αντίθετα, όταν οι αντίστοιχες μέθοδοι κλήθηκαν από το BlueJ οι σπουδαστές περιέγραφαν ορθά τι συνέβαινε. Πράγμα που δείχνει και πάλι, ότι, όταν το σύστημα αναλαμβάνει «σιωπηλά» ενέργειες σε πρώτη φάση βοήθα στην κατανόηση των εννοιών ενέχει όμως και κίνδυνο παρανοήσεων αν η δραστηριότητα της άμεσης διαχείρισης δεν συνοδευτεί με αντίστοιχη δραστηριότητα σύνταξης κώδικα. Ο Kölling στην περιγραφή των απαιτήσεων για οπτικοποίηση αναφέρει «*Η διαχείριση της γραφικής ή της αναπαράστασης με κείμενο μιας κλάσης ή ενός αντικειμένου πρέπει να είναι δυνατή εναλλακτικά*». Στις επόμενες έρευνες που διεξήγαγαν οι Xinogalos et al. έλαβαν υπόψη τα προβλήματα που παρουσιάστηκαν από τη μονομερή χρήση διαφόρων χαρακτηριστικών του BlueJ και της σειράς μαθημάτων, όπως περιγράφεται στο βιβλίο, και επανασχέδιασαν το μάθημα και τις δραστηριότητες με εμφανή τα αποτελέσματα βελτίωσης και μείωσης των προβλημάτων τα οποία είχαν αποδοθεί στο BlueJ.

Οι Georgantaki & Retalis (Georgantaki & Retalis, 2007) πραγματοποίησαν το 2006 έρευνα στο πλαίσιο διαδικτυακού μεταπτυχιακού μαθήματος, στην οποία συμμετείχαν 18 σπουδαστές. Ανάμεσα στα θέματα που διερευνήθηκαν στη πιλοτική μελέτη ήταν η αξιολόγηση ευχρηστίας του BlueJ. Οι σπουδαστές, όπως ανέμεναν και οι ερευνητές, θεώρησαν ως σημαντικότερα χαρακτηριστικά αυτά που αναφέρονται στην οπτικοποίηση των κλάσεων και των αντικειμένων, την απλότητα και την αλληλεπίδραση που προσφέρει.

3.2 Αξιολογήσεις του jGRASP

Η ομάδα ανάπτυξης του jGRASP διενεργεί προγραμματισμένα πειράματα προκειμένου να εκτιμήσει την επίδραση των διαφόρων χαρακτηριστικών του jGRASP στην κατανόηση εννοιών του ΑΠ. Η πρώτη αξιολόγηση πραγματοποιήθηκε το 2000 (Hendrix et al., 2000) προκειμένου να ελεγχθεί η εκπαιδευτική αξία των διαγραμμάτων δομών ελέγχου (CSD) στη κατανόηση προγράμματος. Η πειραματική αξιολόγηση έγινε με 39 φοιτητές οι οποίοι χωρίστηκαν σε 2 ομάδες, και παρακολουθούσαν μάθημα προχωρημένων εννοιών του ΑΠ. Η ομάδα που χρησιμοποίησε το CSD και η ομάδα ελέγχου (CG-Control Group) που εργάστηκε χωρίς το CSD. Η ανάλυση της απόδοσης των 2 ομάδων έδειξε ότι η επίδραση των CSD ήταν θετική. Ιδιαίτερα το 45% των απαντήσεων της ομάδας CSD ήταν σωστές, ενώ μόνο το 26% των απαντήσεων της ομάδας CG ήταν σωστές. Η δεύτερη αξιολόγηση πραγματοποιήθηκε το 2006 (Jain et al., 2006) και περιελάμβανε 2 πειράματα

προκειμένου να ελεγχθεί η εκπαιδευτική αξία των object viewers. Δημιουργήθηκαν 2 ομάδες από 26 σπουδαστές η καθεμία. Οι σπουδαστές της ομάδας 1 (G1) ήταν εξοικειωμένοι με τον debugger του jGRASP και οι σπουδαστές της ομάδας 2 (G2) ήταν εξοικειωμένοι με τον debugger και τους object viewers του jGRASP. Συγκεκριμένα ελέγχτηκαν οι παρακάτω υποθέσεις: *«Υ1: Οι σπουδαστές είναι σε θέση να κωδικοποιήσουν ακριβέστερα (με λιγότερα λάθη) χρησιμοποιώντας τους object viewers του jGRASP»*. Η ανάλυση των αποτελεσμάτων έδειξε ότι ο μέσος χρόνος για την ομάδα G2 ήταν 109 λεπτά ενώ ο μέσος χρόνος για την ομάδα G1 ήταν 112 λεπτά. Επίσης η μέση ακρίβεια της G2 ήταν 6.34, ενώ η μέση ακρίβεια της ομάδας G1 ήταν 4.48. Γενικώς διαπιστώθηκε ότι για το 95% όλων των περιπτώσεων, οι object viewers βοήθησαν τους σπουδαστές να αυξήσουν την ακρίβεια και να μειώσουν το χρόνο για προγράμματα που υλοποιούν δομές δεδομένων. *«Υ2: Οι σπουδαστές είναι σε θέση να εντοπίσουν και να διορθώσουν τα «μη-συντακτικά» λάθη ακριβέστερα, χρησιμοποιώντας τους object viewers του jGRASP»*. Η ανάλυση των αποτελεσμάτων έδειξε ότι ο μέσος χρόνος για την ομάδα G2 ήταν 88.23 λεπτά ενώ ο μέσος χρόνος για την ομάδα G1 ήταν 87.6 λεπτά. Επίσης η G2 εντόπισε και διόρθωσε περισσότερα λάθη. Επιπλέον η ομάδα αυτή εισήγαγε λιγότερα λάθη. Γενικώς διαπιστώθηκε ότι για το 95% όλων των περιπτώσεων, οι object viewers βοήθησαν τους σπουδαστές να αυξήσουν την ακρίβεια, αλλά ο χρόνος που χρειάστηκαν για να αναπτύξουν τα προγράμματα τους ήταν ελαφρά μεγαλύτερος.

4. Συμπεράσματα

Στην εργασία αυτή έγινε μια συγκριτική παρουσίαση δυο ΕΠΠ, του BlueJ και jGRASP καθώς και των αξιολογήσεων που έχουν πραγματοποιηθεί από διάφορες ομάδες ερευνητών. Και τα δυο περιβάλλοντα ακολουθούν τις απαιτήσεις που διατύπωσε ο Kölling για τα χαρακτηριστικά που πρέπει να έχει ένα ΕΠΠ για τη διδασκαλία και μάθηση του ΑΠ. Το BlueJ συνεχίζει να διατηρεί την αρχική του απλότητα. Το jGRASP χρησιμοποιεί σε μεγαλύτερο βαθμό απεικονίσεις εννοιών με γραφικό τρόπο, όπως σχηματική και χρωματική αναπαράσταση αντικειμένων, πεδίων και μεθόδων με στόχο την υποστήριξη της κληρονομικότητας και του πολυμορφισμού. Επίσης επιτρέπει τη δυναμική διαχείριση των αντικειμένων κατά την εκτέλεση της εφαρμογής δίνοντας τη δυνατότητα για *what if* σενάρια. Χρησιμοποιεί την οπτικοποίηση όχι μόνο για να υποστηρίξει την κατανόηση των εννοιών κλάση, αντικείμενο αλλά και για να βοηθήσει στην κατανόηση της δομής του κώδικα με τα CSD. Το BlueJ ευνοεί τη δημιουργία αντικειμένου και τη κλήση μεθόδου με άμεση διαχείριση των γραφικών αναπαραστάσεων τους, και αν και επιτρέπει τη σύνταξη μεμονωμένης εντολής, αυτή δε συνοδεύεται με δυναμική ενημέρωση του γραφικού ενδιάμεσου, Αντίθετα, το jGRASP, προσφέρει και τις δυο δυνατότητες, δηλαδή οι ίδιες ενέργειες γίνονται με άμεση διαχείριση της γραφικής αναπαράστασης της έννοιας αλλά και με σύνταξη κειμένου (εντολής). Τα χαρακτηριστικά αυτά επιτρέπουν μια ισορροπημένη και «δίκαιη» χρήση τόσο της εκτέλεσης εντολών με άμεση διαχείριση των γραφικών αναπαραστάσεων των εννοιών κλάση και αντικείμενο, όσο και της εκτέλεσης εντολών με τη σύνταξη κειμένου. Επίσης, επιτρέπουν στο jGRASP να μπορεί να χρησιμοποιηθεί όχι μόνο σε ένα εισαγωγικό μάθημα για τον ΑΠ αλλά και σε πιο προχωρημένα, όπως δομές δεδομένων με java.

Όσον αφορά τις έρευνες που έχουν διεξαχθεί για την αξιολόγηση τους, φαίνεται ότι, αν και τα δυο περιβάλλοντα χρησιμοποιούνται από ένα μεγάλο αριθμό ιδρυμάτων, εντούτοις οι αναφορές σχετικά με την αξιολόγηση της εκπαιδευτικής τους αξίας είναι περιορισμένες, και μερικές λιγότερο συστηματικές, επιβεβαιώνοντας τα όσα διατύπωσαν οι Guzdial και Gross. Αν και υπάρχουν έρευνες που επισημαίνουν προβλήματα από τη χρήση του BlueJ παρ' όλα αυτά ο σκοπός τους δεν ήταν να ερευνήσουν την εκπαιδευτική του αξία, όπως η έρευνα των (Ragonis & Ben-Ari, 2005) διενεργήθηκε με σκοπό να διερευνήσουν αν μπορεί να διδαχθεί ο ΑΠ σε νεαρούς μαθητές. Επίσης και οι έρευνες των (Xinogalos, Satratzemi, & Dagdilelis, 2006; Xinogalos, Satratzemi, & Dagdilelis, 2008) αξιολογούν κατά βάση τη διδακτική προσέγγιση των (Barnes & Kölling, 2006) και έμμεσα και το BlueJ. Στα πλαίσιο τέτοιων ερευνών καταγράφηκαν και δυσκολίες ή παρανοήσεις των σπουδαστών που οφείλονταν στο περιβάλλον. Αν και οι έρευνες αυτές δεν διενεργήθηκαν με στόχο να αξιολογήσουν το BlueJ και κατά συνέπεια δεν μπορούν να αξιολογηθούν ως έγκυρες και αντιπροσωπευτικές, παρ' όλα αυτά συνεισέφεραν στην ερευνητική και εκπαιδευτική κοινότητα στο ότι επισημάνθηκαν προβλήματα που οφείλονται περισσότερο στην υπέρμετρη χρήση των διεπαφών του περιβάλλοντος οι οποίες ως ένα βαθμό βοηθούν τους σπουδαστές να κατανοήσουν τις έννοιες κλάση, αντικείμενο, όταν έρχονται σε πρώτη επαφή μ' αυτές, αλλά στη συνέχεια θα πρέπει να

εξισορροπείται η χρήση των γραφικών χαρακτηριστικών του περιβάλλοντος με τη σύνταξη εντολών. Ίσως σ' αυτό το σημείο η δυνατότητα για σύνταξη και εκτέλεση μεμονωμένων εντολών (*καρτέλα Interactions*) με ταυτόχρονη δυναμική οπτικοποίηση των ενεργειών να κάνει το jGRASP πιο ισχυρό.

Όσον αφορά τις έρευνες αξιολόγησης του jGRASP είναι πιο στοχευόμενες, χρησιμοποιήθηκαν 2 ομάδες χρηστών για την αξιολόγηση της επίδρασης συγκεκριμένων χαρακτηριστικών του και κατά συνέπεια τα αποτελέσματα φαίνονται εγκυρότερα. Βέβαια θα πρέπει να επισημανθεί ότι το επίπεδο γνώσεων των χρηστών ήταν προχωρημένο, και ενδεχόμενα οι αρχάριοι σπουδαστές να επισημάνουν προβλήματα σχετικά με την ευκολία χρήσης του, καθώς το γραφικό του ενδιάμεσο δεν φαίνεται το ίδιο λυτό και απλό με αυτό του BlueJ.

Ενδιαφέρον έχει η διενέργεια έρευνας για την αποτίμηση της επίδρασης των χαρακτηριστικών του BlueJ και αντίστοιχα του jGRASP στην μάθηση του ΑΠ, στην παραγωγικότητα, στην ικανοποίηση ή και στην απογοήτευση, δυο ομάδων αρχάριων χρηστών με αντίστοιχα χαρακτηριστικά (ίδιο επίπεδο γνώσεων), τον ίδιο διδάσκοντα, και την ίδια διδακτική προσέγγιση. Στα άμεσα σχέδια μας είναι η διενέργεια μιας τέτοιας έρευνας υιοθετώντας τη διδακτική προσέγγιση διδασκαλίας που προτείνουν οι (Barnes & Kölling, 2006), η οποία είναι εφαρμόσιμη χρησιμοποιώντας είτε το BlueJ είτε το jGRASP καθώς και τα δυο διαθέτουν αντίστοιχα χαρακτηριστικά και υποστηρίζουν την προσέγγιση «πρώτα αντικείμενα» και ακολουθώντας τη μεθοδολογία αξιολόγησης που προτείνεται από τους (Gross & Powers, 2005).

Βιβλιογραφία

Barnes, D. & Kölling, M. (2006). *Objects First with Java: A practical introduction using BlueJ*, 3rd edition, Prentice Hall /Pearson Education, Harlow, England.

Cross II J., Hendrix T., Umphress D. & Barowski L. (2008). Exploring accessibility and visibility relationships in java. *In ITiCSE 2008*, pp. 103-108.

Cross II J., Hendrix T. & Barowski L., (2002). Using the Debugger as an Integral Part of Teaching CS1. *In 32nd ASEE/IEEE Frontiers in Education Conference 2002*, pp. FIG-1-FIG6.

Georgantaki, S. & Retalis, S. (2007). Using Educational Tools for Teaching Object Oriented Design and Programming. *Journal of Information Technology Impact*, 7(2), 111-130.

Gross, P. & Powers, K., (2005). Evaluating assessments of novice programming environments. *In First international workshop on Computing education research*, Seattle, WA, USA, 2005, pp. 99-108.

Guzdial, M., (2004). Programming environments for novices. *In Computer Science Education Research*. S. Fincher and M. Petre (Eds.). Swets and Zeitlinger. Chapter 3.

Hendrix, T., Cross II, J., Maghsoodloo, S. & McKinney M. (2000). Do visualizations improve program comprehensibility? experiments with control structure diagrams for Java. *In SIGCSE 2000*, pp. 382-386.

Jain, J., Cross II, J., Hendrix, T., Umphress, D. & Barowski L. (2006). Experimental evaluation of animated-verifying object viewers for Java. *In SOFTVIS 2006*, pp. 27-36.

Kölling, M. & Rosenberg, J. (2001). Guidelines for Teaching Object Orientation with Java. *ACM SIGCSE Bulletin*, 33(3), 33-36.

Kölling, M. (1999). The Problem of Teaching Object-Oriented Programming, Part 2: Environments. *Journal of Object-Oriented Programming*, 11(9), 6-12.

Kölling, M., Quig, B., Patterson, A. & Rosenberg, J. (2003). The BlueJ system and its pedagogy, *Journal of Computer Science Education*, 13(4), 249-268.

McIver, L. (2002). Evaluating Languages and Environments for Novice Programmers. *In Fourteenth Annual Workshop of the Psychology of Programming Interest Group (PPIG 2002)*, Middlesex 2002, UK, pp. 100-110.

Ragonis, N. & Ben-Ari, M. (2005). A long-Term Investigation of the Comprehension of OOP Concepts by Novices, *Computer Science Education*, 15(3), 203-221.

Van Haaster, K. & Hagan, D. (2004). Teaching and Learning with BlueJ: an Evaluation of a Pedagogical Tool. *In Information Science + Information Technology Education Joint Conference*, Rockhampton, QLD, Australia, June, 2004, pp. 455-470.

Xinogalos, S., Satratzemi, M. & Dagdilelis, V. (2006). Studying Students' Difficulties in an OOP Course Based on BlueJ. *In 9th IASTED International Conference on Computers and Advanced Technology in Education (CATE 2006)*, Lima, Peru, 4-6 October 2006, pp. 82-87.

Xinogalos, S., Satratzemi, M. & Dagdilelis, V. (2008). An analysis of students' difficulties with ArrayList object collections and proposals for supporting the learning process. *In 8th IEEE International Conference on Advanced Learning Technologies (IEEE ICALT 2008)*, Niigata, Japan, 18-20 July 2007, pp. 180-182.