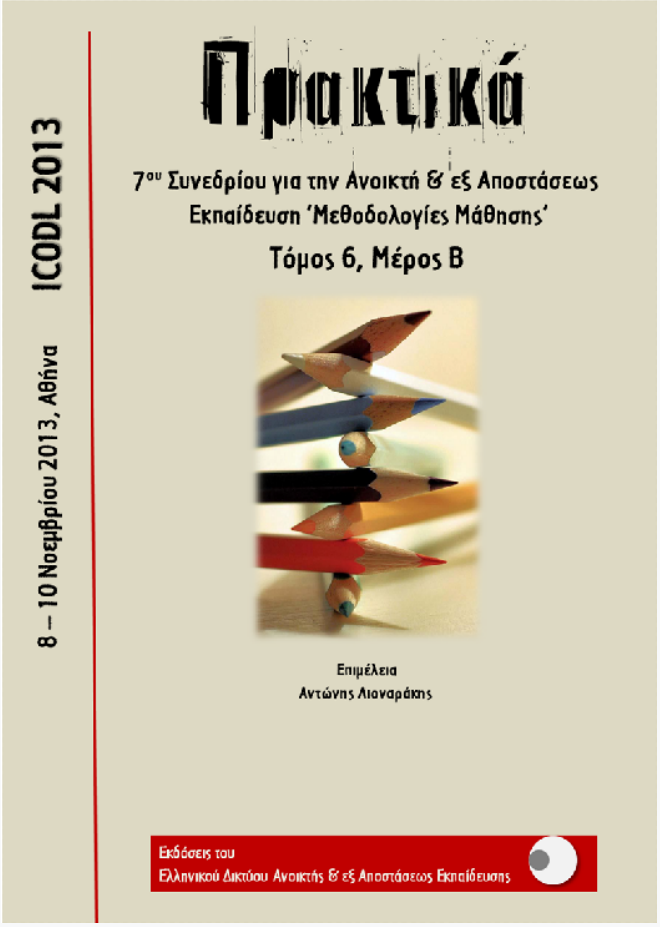
Διεθνές Συνέδριο για την Ανοικτή & εξ Αποστάσεως Εκπαίδευση

Τόμ. 7, Αρ. 6Β (2013)

Μεθοδολογίες Μάθησης



Αυτόματη αξιολόγηση ασκήσεων προγραμματισμού, με χρήση της πλατφόρμας DOMjudge

Χαράλαμπος Τσιμπούρης, Κυριάκος Σγάρμπας

doi: 10.12681/icodl.589

Αυτόματη αξιολόγηση ασκήσεων προγραμματισμού, με χρήση της πλατφόρμας DOMjudge

Unsupervised assessment of programming exercises based on DOMjudge

Χαράλαμπος Τσιμπούρης
Υπ. Διδάκτορας
Εργαστήριο Ενσύρματης Τηλ.
Τμήμα Ηλ. Μηχ. & Τεχ. Υπολ.
xtsimpouris@upatras.gr

Κυριάκος Σγάρμπας
Επίκουρος Καθηγητής
Εργαστήριο Ενσύρματης Τηλ.
Τμήμα Ηλ. Μηχ. & Τεχ. Υπολ.
sgarbas@upatras.gr

Abstract

Programming courses, especially during freshmen years, require correcting a significant number of exercises (source code) which multiplied by the large number of students create a significant workload and requires a large team of teachers. However, evaluation should be completed in a quite short time, so that students know their performance throughout the semester. In this paper, we present a solution to the above problem, based on the modification of the DOMjudge platform (normally used for programming contests) to assess within seconds automatically programming exercises. In the modified system presented, teachers may provide departments and individual lessons, open or restricted access, which introduce programming exercises to solve. Students can solve the exercises through a web browser, without even installing any application regardless of operating system, using a visually interactive editor. We analyse all the individual changes made to the original version of the platform, both in functionality of extra features, and the database structure that supports it. The modified system is aimed at remote and uninterrupted assistance of students in learning various programming languages while, in addition, it can be respectively used for automatic and objective grading of final examinations.

Keywords: *autonomous assessment, e-Learning, programming exercises*

Περίληψη

Η διδασκαλία μαθημάτων προγραμματισμού, ιδιαίτερα στα μικρά έτη των πανεπιστημιακών ιδρυμάτων, απαιτεί τη διόρθωση σημαντικού αριθμού ασκήσεων (κώδικα) που πολλαπλασιαζόμενος με το μεγάλο πλήθος φοιτητών δημιουργεί έναν σημαντικό φόρτο εργασίας και απαιτεί πολυμελείς ομάδες διδασκόντων για τη διόρθωση. Μάλιστα, συχνά η αξιολόγηση των ασκήσεων πρέπει να γίνει σε ιδιαίτερα περιορισμένο χρόνο, ώστε οι φοιτητές να γνωρίζουν την επίδοσή τους σε μια άσκηση πριν ασχοληθούν με την επόμενη. Στο παρόν κείμενο παρουσιάζεται μια λύση στο παραπάνω πρόβλημα, βασισμένη στην τροποποίηση της πλατφόρμας DOMjudge (που κανονικά χρησιμοποιείται για διαγωνισμούς προγραμματισμού) ώστε να διορθώνει και να αξιολογεί αυτόματα (ακόμη και σε πραγματικό χρόνο) ασκήσεις προγραμματισμού. Στο τροποποιημένο σύστημα που παρουσιάζεται στην εργασία, οι διδάσκοντες μπορούν να ορίζουν τμήματα και επί μέρους μαθήματα, ανοικτής ή περιορισμένης πρόσβασης, στα οποία εισάγουν ασκήσεις προγραμματισμού προς

επίλυση. Οι φοιτητές μπορούν να λύσουν τις ασκήσεις μέσα από έναν φυλλομετρητή (web-browser), χωρίς να εγκαταστήσουν οποιαδήποτε εφαρμογή και ανεξαρτήτως λειτουργικού συστήματος, χρησιμοποιώντας έναν διαδραστικό κειμενογράφο, ο οποίος υποστηρίζει μεγάλη ποικιλία γλωσσών προγραμματισμού. Στην παρούσα εργασία, αναλύονται όλες οι επιμέρους τροποποιήσεις που έγιναν στην αρχική έκδοση της πλατφόρμας DOMjudge, τόσο σε λειτουργικότητα των επιπλέον δυνατοτήτων, όσο και στη δομή της βάσης δεδομένων που την υποστηρίζει. Το τροποποιημένο σύστημα στοχεύει στην απομακρυσμένη και αδιάλειπτη υποβοήθηση φοιτητών για την εκμάθηση ποικίλων γλωσσών προγραμματισμού και επιπροσθέτως μπορεί να χρησιμοποιηθεί για την αυτόματη και αντικειμενική βαθμολόγηση αντίστοιχων εξετάσεων.

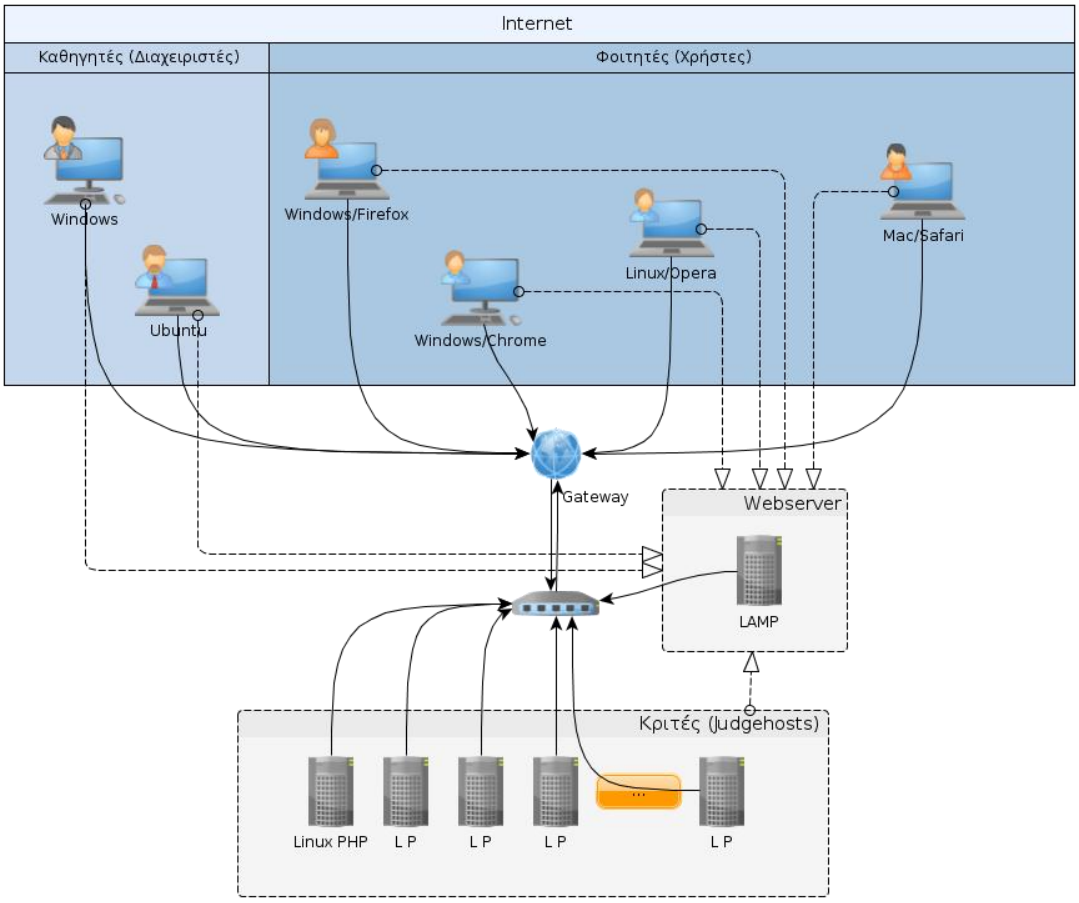
Λέξεις-κλειδιά: *αυτόματη αξιολόγηση, απομακρυσμένη μάθηση, ασκήσεις προγραμματισμού*

1. Εισαγωγή

Στα μαθήματα προγραμματισμού, οι φοιτητές μαθαίνουν να επιλύουν προβλήματα παράγοντας κατάλληλο και αποδοτικό κώδικα. Η αξιολόγηση πρέπει να συμμορφώνεται με αυτούς τους στόχους ώστε να ελεγχθεί η ικανότητα των φοιτητών να κατασκευάσουν τον σωστό κώδικα. Συνήθως οι φοιτητές λαμβάνουν μια σειρά από προβλήματα (εργαστηριακές ή/και φροντιστηριακές ασκήσεις) κατά τη διάρκεια του ακαδημαϊκού εξαμήνου με σκοπό να αναπτύξουν συγκεκριμένες δεξιότητες: ανάλυση σε υπο-προβλήματα, εξεύρεση λύσεων, εφαρμογή, αξιολόγηση και τελικός έλεγχος λύσεων. Παρ' όλα αυτά, η ανάθεση πολλών εργασιών προκαλεί σημαντική δυσκολία διόρθωσης στους διδάσκοντες του μαθήματος και στους βοηθούς εργαστηρίων καθώς ο φόρτος εργασίας αυξάνεται δυσανάλογα για κάθε επιπλέον εργασία που ανατίθενται, συχνά σε εκατοντάδες φοιτητές. Ενδεικτικά, ένα μάθημα προγραμματισμού στο Α' έτος ενός πανεπιστημιακού τμήματος μπορεί να έχει περί τους 300 εγγεγραμμένους φοιτητές. Σε προγράμματα σπουδών που εφαρμόζουν Ανοικτή και εξ' Αποστάσεως Εκπαίδευση, όπως για παράδειγμα στο Ελληνικό Ανοικτό Πανεπιστήμιο, αυτός ο αριθμός εύκολα μπορεί να ξεπεράσει το 1000.

Στην παρούσα εργασία παρουσιάζεται μια τροποποίηση (ανακατασκευή) της πλατφόρμας ανοικτού κώδικα DOMjudge (http://domjudge.sourceforge.net/) για την αυτόματη και απομακρυσμένη βαθμολόγηση κώδικα με στόχο την υποβοήθηση της εκπαιδευτικής διαδικασίας. Η πλατφόρμα DOMjudge βασίζεται στο ευρέως διαδεδομένο δωρεάν πακέτο λογισμικού L.A.M.P. (Linux, Apache, MySQL, PHP) και παραδοσιακά χρησιμοποιείται στην διεξαγωγή διαγωνισμών προγραμματισμού. Η συγκεκριμένη πλατφόρμα έχει κατασκευαστεί στο Utrecht University (Eldering, Kinkhorst, Warken, 2010) και έχει χρησιμοποιηθεί επιτυχώς σε πολλούς διαγωνισμούς πληροφορικής όπως τον ACM ICPC (http://icpc.baylor.edu/). Επίσης, έχει μελετηθεί για χρήση "σοβαρών παιγνίων" (serious games) σε συνδυασμό με τη μηχανή δημιουργίας γραφικών Unity3D (Coelho, Kato1, Xavier & Gonçalves, 2011). Τα πλεονεκτήματά του DOMjudge βασίζονται στο γεγονός πως δεν χρειάζεται εξειδικευμένο υλικό (hardware) αλλά μπορεί να εγκατασταθεί σε υπολογιστές με λειτουργικό σύστημα Debian GNU/Linux μαζί με το L.A.M.P (Pacheco, 2010). Αρκεί ένας μόνο διακομιστής (*webserver*) για να εξυπηρετεί όλες τις αιτήσεις αρχείων κώδικα και να τις καταχωρεί στην βάση δεδομένων του (MySQL) από την οποία όλα τα μηχανήματα *κριτές* (*judgehosts*) αξιολογούν τον κώδικα των χρηστών και τους βαθμολογούν με χρήση γλώσσας προγραμματισμού PHP (Εικόνα 1). Εν' αντιθέσει

άλλων αντίστοιχων συστημάτων όπως τα CourseMarker (Higgins, Hegazy, Symeonidis & Tsintsifas, 2003), SAC (Auffarth, Lopez-Sonchez, Campos & Puig,
άλλων αντίστοιχων συστημάτων όπως τα CourseMarker (Higgins, Hegazy, Symeonidis & Tsintsifas, 2003), SAC (Auffarth, Lopez-Sonchez, Campos & Puig,



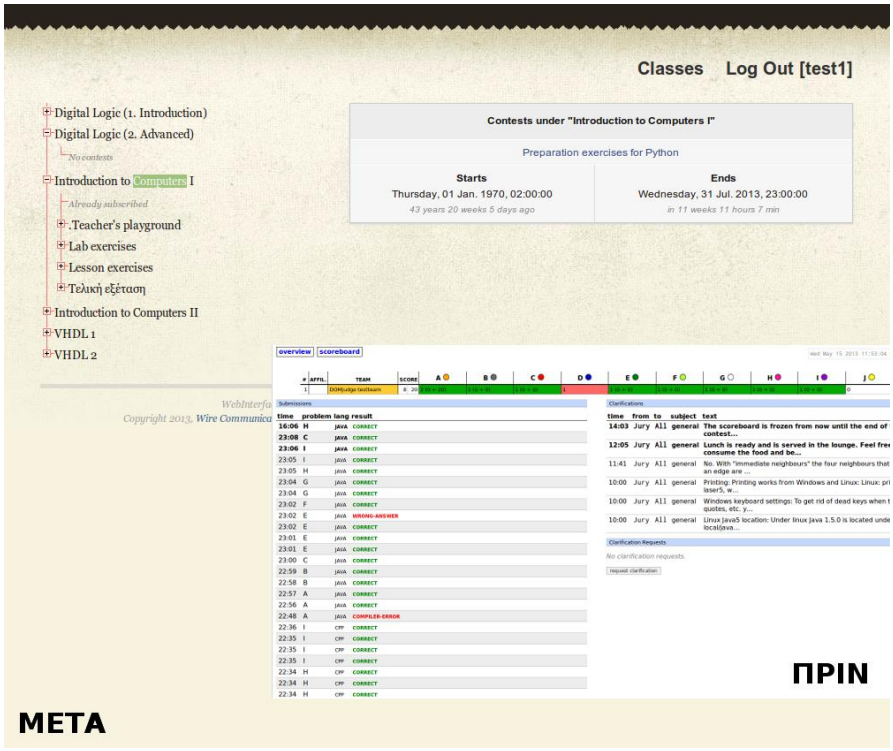
Εικόνα 1: Τοπολογία δικτύου πλατφόρμας DOMjudge

2004) και BOSS (Luck & Joy, 1999), η συγκεκριμένη πλατφόρμα δεν απαιτεί κάποια εγκατάσταση από την πλευρά του χρήστη (φοιτητή), καθώς όλη η λειτουργικότητα, υποστηρίζεται από τους ευρέως διαδεδομένους φυλλομετρητές Chrome, Firefox, Intenet Explorer και Opera. Επίσης, έχει κατασκευαστεί με έμφαση στην ασφάλεια του συστήματος ενάντια σε επιθέσεις υλικού και λογισμικού ενώ η δομή του βοηθάει στην εύκολη επεκτασιμότητα σε περίπτωση που το απαιτούν οι συνθήκες, όπως για παράδειγμα για την υποστήριξη ακόμα περισσότερων φοιτητών χωρίς επιπλέον καθυστέρηση, σύμφωνα με τις αναλυτικές τεχνικές οδηγίες των κατασκευαστών (Eldering κ.ά., 2010). Από την άλλη πλευρά όμως, το DOMjudge στοχεύει εξ' ολοκλήρου στην υποστήριξη διαγωνισμών πληροφορικής και ως εκ τούτου στερείται βασικών δυνατοτήτων εκπαιδευτικής υποστήριξης όπως – ενδεικτικά – δυνατότητα διόρθωσης παράλληλων ασκήσεων που δεν ανήκουν στο ίδιο μάθημα, προβολή σφαλμάτων χρήστη, ύπαρξη διαδραστικού κειμενογράφου και δυνατότητα εγγραφής νέων χρηστών χωρίς τη διαμεσολάβηση διαχειριστή. Η τροποποιημένη έκδοση που παρουσιάζουμε σε αυτήν την εργασία προσθέτει αυτές τις δυνατότητες και ολοκληρώνεται σε ένα σύστημα που επιτυγχάνει την αυτόματη αξιολόγηση ασκήσεων προγραμματισμού σε μεγάλη ποικιλία γλωσσών για πολλά μαθήματα ταυτόχρονα και για μεγάλο πλήθος φοιτητών. Στην ενότητα 2 αυτής της εργασίας παρουσιάζονται οι τεχνικές κατασκευαστικές λεπτομέρειες του συστήματος. Στην

ενότητα 3 εξηγείται ο τρόπος χρήσης και οι αρχές λειτουργίας του. Στην ενότητα 4 παρουσιάζονται συμπεράσματα και μελλοντικές βελτιώσεις.

2. Τροποποίηση-Ανακατασκευή

Η τροποποίηση-ανακατασκευή της πλατφόρμας DOMjudge για εκπαιδευτική χρήση (υποστήριξη μαθημάτων προγραμματισμού), συμπεριέλαβε πολλές βελτιώσεις, όχι μόνο σε κώδικα, αλλά και στη δομή της βάσης δεδομένων. Συγκεκριμένα, οι αλλαγές



Εικόνα 2: Αρχική σελίδα χρήστη, πριν και μετά τις αλλαγές

που έγιναν διακρίνονται σε τρεις κατηγορίες: (α) Τροποποιήσεις στον Εξυπηρετητή (Webserver), (β) Τροποποιήσεις στη Βάση Δεδομένων (MySQL Database), (γ) Τροποποιήσεις στο Δαίμονα κριτή (Judgehost daemon).

2.1 Τροποποιήσεις στον Εξυπηρετητή (Webserver)

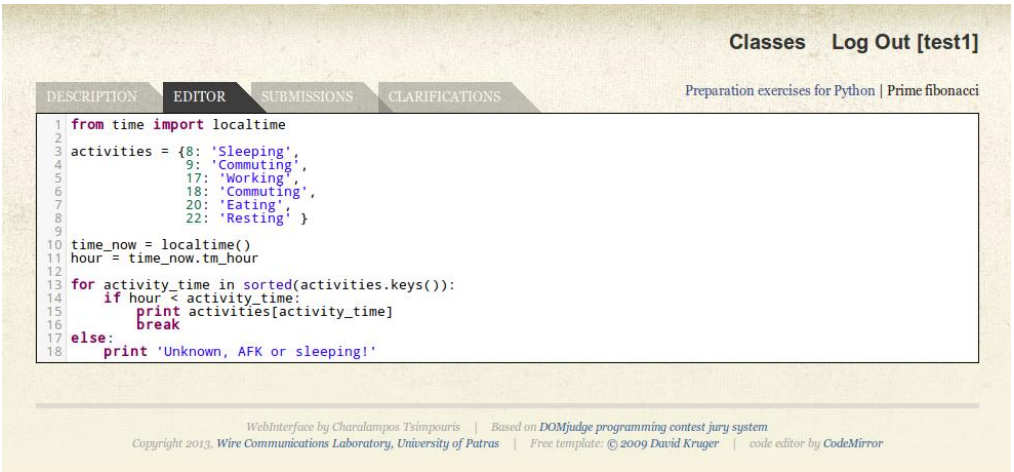
Η πρόσβαση όλων των χρηστών, φοιτητών και διδασκόντων, γίνεται μέσω διαδικτυακής (*web-based)* ιστοσελίδας η οποία υποστηρίζεται από όλους τους γνωστούς φυλλομετρητές, χωρίς να μας απασχολεί το λειτουργικό σύστημα του χρήστη. Η αναβάθμιση μας οδήγησε στην πλήρη αντικατάσταση του υπάρχοντος κώδικα στον εξυπηρετητή με σκοπό την βελτίωσή του με πιο σύγχρονα πρότυπα συγγραφής κώδικα. Στην εικόνα 2 γίνεται εμφανής η μετάβαση από την υπάρχουσα πλατφόρμα (“*ΠΡΙΝ”*), στην νέα (“*ΜΕΤΑ”*). Οι αλλαγές που έγιναν στον εξυπηρετητή ήταν:

- Με χρήση αρχείων προτύπων και της μηχανής Rain TPL (http://www.raintpl.com/) έγινε πλήρης διαχωρισμός του κώδικα ελέγχου και αποφάσεων σε σχέση με την εμφάνιση της σελίδας στον τελικό χρήστη. Ως εκ τούτου, ακόμα και εκ των υστέρων, η εμφάνιση μπορεί να κατασκευασθεί εκ νέου χωρίς να επηρεαστεί η λειτουργικότητα της υπηρεσίας. Επίσης, έγινε πλήρης διαχωρισμός των λεκτικών εκφράσεων, οι οποίες μπορούν να μεταφραστούν μέσω συστήματος διαχείρισης σε οποιαδήποτε άλλη γλώσσα

χωρίς να επηρεαστεί ο υπάρχον κώδικας, η λειτουργικότητα του συστήματος ή η τελική παρουσίαση. Εδώ, πρέπει να τονίσουμε το γεγονός, πως στα πλαίσια της αναβάθμισης, προχωρήσαμε σε πλήρη εξελληνισμό όλης της πλατφόρμας.

- Η αρχική σελίδα περιλαμβάνει πλέον τμήματα με δυνατότητα ιεράρχησης και επί μέρους μαθήματα, στα οποία οι φοιτητές μπορούν να συμμετέχουν και να επιλύουν σειρές ασκήσεων (Εικόνα 2).

- Η μέθοδος υποβολής αρχείων κώδικα, αντικαταστάθηκε με τον γνωστό online επεξεργαστή κώδικα CodeMirror (http://codemirror.net/) ο οποίος βασίζεται σε JavaScript και εκτελείται πλήρως στον υπολογιστή του χρήστη (client-side). Πλέον δεν υπάρχει η απαίτηση ο φοιτητής να εγκαταστήσει οποιοδήποτε πρόγραμμα ή μεταγλωττιστή στον υπολογιστή του (βλ. Εικόνα 3). Μέσα από την ιστοσελίδα μπορεί να γράψει κώδικα, σε οπτικά υποβοηθούμενο κειμενογράφο που προσαρμόζεται ανάλογα με τη γλώσσα προγραμματισμού του εκάστοτε μαθήματος (προς το παρόν υποστηρίζονται οι γλώσσες C, C++, Python, PHP, AWK, Bash, C#, Java, Pascal, Perl, POSIX, Haskell με δυνατότητα προσθήκης νέων).
- Καθώς η συγγραφή κώδικα μπορεί να διακοπεί ανά πάσα στιγμή, λόγω προβλημάτων δικτύου, οι φοιτητές έχουν τη δυνατότητα να αποθηκεύουν προσωρινά στον απομακρυσμένο διακομιστή και σε τακτά χρονικά διαστήματα τον κώδικά τους, κατά βούληση, πριν την τελική υποβολή.
- Ανάλογα με τις ρυθμίσεις κάθε άσκησης, ο φοιτητής είναι σε θέση να δει τα αποτελέσματα του κώδικά του σε κάθε περίπτωση εισόδου, καθώς επίσης και τα μηνύματα για πιθανά σφάλματα, όπως ακριβώς θα μεταγλώττιζε και εκτελούσε το πρόγραμμα τοπικά στον υπολογιστή του.
- Οι διδάσκοντες μπορούν να προβάλουν συγκεντρωτικά τις βαθμολογίες για όλους τους εγγεγραμμένους φοιτητές και να τις κατεβάσουν τοπικά στον υπολογιστή τους για περαιτέρω μελέτη. Επίσης, μπορούν να μεταφορτώσουν μαζικά και τον τελικό κώδικα που κατέθεσε κάθε φοιτητής, για τον οποίο και αξιολογήθηκε με την τελική του βαθμολογία από το σύστημα.



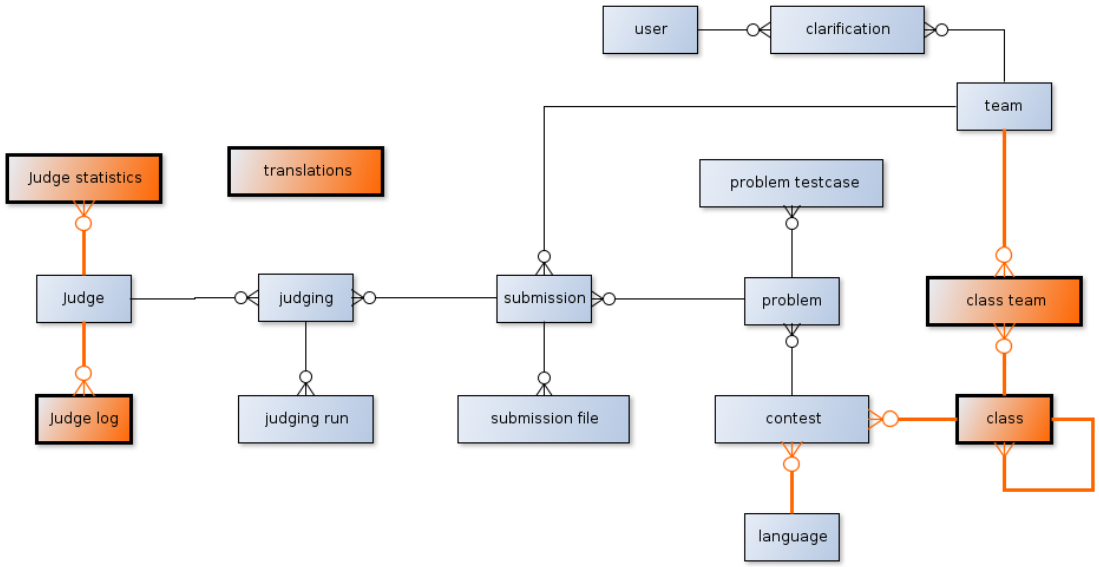
Εικόνα 3: Online κειμενογράφος, οπτικά υποβοηθούμενος

2.2 Τροποποιήσεις στη Βάση Δεδομένων (MySQL Database)

Η βελτίωση της πλατφόρμας μας οδήγησε στο να αλλάξουμε τη δομή της βάσης δεδομένων σε πολλούς πίνακες, καθώς χρειάστηκε να υποστηρίξουν νέες υπηρεσίες

και δυνατότητες. Στην εικόνα 4 εμφανίζεται το απλοποιημένο διάγραμμα οντοτήτων, στο οποίο έχουν σημειωθεί με πορτοκαλί χρώμα και έντονη γραμμή οι νέοι πίνακες και οι αντίστοιχες συνδέσεις. Στην συνέχεια, αναφέρονται επιγραμματικά οι κυριότερες αλλαγές στον τρόπο λειτουργίας του συστήματος που οδήγησαν σε αλλαγές στη δομή της βάσης:

- Κάθε αίτηση κώδικα (καταχώρηση στον πίνακα *submission*) φοιτητή θα πρέπει να μπορεί να διαφοροποιηθεί σε περίπτωση που δεν είναι τελική υποβολή αλλά προσωρινή αποθήκευση, ούτως ώστε το σύστημα να μην προχωρά σε βαθμολόγηση. Ως εκ τούτου, τροποποιήθηκε κατάλληλα ο αντίστοιχος πίνακας με την πληροφορία της τελικής υποβολής ή της προσωρινής αποθήκευσης.
- Δημιουργήθηκε κατάλληλος πίνακας (*class team*) για διασταύρωση και τήρηση αρχείου για το ποιος φοιτητής είναι εγγεγραμμένος σε ποιο μάθημα (*class*). Πλέον, υπάρχουν σειρές ασκήσεων που είναι "κλειστές", και χρειάζεται ειδική άδεια καθηγητή (όπως για παράδειγμα εξέταση εργαστηρίου) ή "ανοιχτές" και μπορεί να έχει πρόσβαση όποιος φοιτητής το επιθυμεί (όπως για παράδειγμα ασκήσεις για εξάσκηση φοιτητών με την πλατφόρμα).
- Δημιουργήθηκε κατάλληλος πίνακας για την ιεράρχηση μαθημάτων, και κατ' επέκταση τμημάτων (πίνακας *class*).
- Δημιουργήθηκε κατάλληλος πίνακας για την τήρηση σφαλμάτων και προβλημάτων από τα μηχανήματα κριτές (*judge log*, *judge statistics*). Πλέον, η προβολή σφαλμάτων μπορεί να γίνει και συγκεντρωτικά μέσα από την



Εικόνα 4: Απλοποιημένο διάγραμμα βάσης δεδομένων

ιστοσελίδα, χωρίς να είναι απαραίτητη η πρόσβαση του διαχειριστή στον εκάστοτε υπολογιστή.

2.3 Τροποποιήσεις στο Δαίμονα Κριτή (Judgehost daemon)

Σε κάθε υπολογιστή που λειτουργεί ως *κριτής* (διορθωτής κώδικα), εκτελείται ειδικό πρόγραμμα που συνδέεται με την κεντρική βάση δεδομένων, διαβάζει νέες αιτήσεις, και προχωρά με προκαθορισμένα βήματα (που εξαρτώνται από τη γλώσσα προγραμματισμού που εξετάζεται) προτού εμφανίσει την τελική βαθμολογία. Στη συγκεκριμένη περίπτωση, όλες οι αλλαγές ολοκληρώθηκαν ώστε να συμφωνούν με

την νέα δομή της βάσης και κυρίως με τον τρόπο βαθμολόγησης. Συγκεκριμένα, η βαθμολόγηση άλλαξε από «Επιτυχία/Αποτυχία σε όλη την άσκηση», σε διαβάθμιση από 0 έως 10, ανάλογα με το ποσοστό σωστά λυμένων περιπτώσεων εισόδου/εξόδου (*correct testcases*), ως προς το πλήθος όλων των περιπτώσεων εισόδου/εξόδου (*all testcases*).
Επιπλέον, κατασκευάστηκε αυτοματοποιημένο πρόγραμμα εγκατάστασης «judge daemon» σε bash script που μπορεί να τρέξει απευθείας από USB memory stick και να εντάξει προσωρινά οποιονδήποτε υπολογιστή στο δίκτυο των κριτών, εύκολα και γρήγορα ακόμα κι αν είναι σε χρήση για άλλη εργασία. Σκοπός είναι να υπάρχει πρόβλεψη ώστε το σύστημα να μπορεί εύκολα να ανταπεξέλθει σε μελλοντικές αυξημένες απαιτήσεις ως προς το πλήθος των φοιτητών που το χρησιμοποιούν ταυτόχρονα, αν αυτό κριθεί απαραίτητο.

3. Λειτουργία Συστήματος

Η σωστή χρήση της πλατφόρμας από τους διδάσκοντες αποτελεί τον βασικό κορμό εγκατάστασης και λειτουργίας του συστήματος, και ξεκινά από τον ορισμό μαθημάτων, με δυνατότητα ιεραρχικής σύνδεσης μεταξύ τους (για παράδειγμα «Εισαγωγή στον Προγραμματισμό / Γλώσσα Python» και «Εισαγωγή στον Προγραμματισμό / Γλώσσα C»). Στην συνέχεια, και για κάθε μάθημα, ο διδάσκων καλείται να δημιουργήσει ομαδοποιημένες σειρές ασκήσεων (που μπορούν να ονομαστούν, για παράδειγμα: «Φροντιστηριακές ασκήσεις», «Ασκήσεις εργαστηρίου», κλπ) και πλέον μπορεί να φορτώσει εύκολα και γρήγορα ασκήσεις από αντίγραφα ασφαλείας παλαιότερων σειρών ή να δημιουργήσει νέες και να εισάγει όλες τις απαραίτητες πληροφορίες μέσα από τη διαδραστική ιστοσελίδα. Κάθε άσκηση, στην οποία ο φοιτητής καλείται να επιλύσει ένα συγκεκριμένο πρόβλημα καταθέτοντάς τον κώδικά του, απαρτίζεται από την εκφώνηση και από ζευγάρια αρχείων εισόδου/εξόδου που ονομάζονται «testcases». Κάθε ζευγάρι περιγράφει την είσοδο στο πρόγραμμα της άσκησης και την αντίστοιχη σωστή έξοδο. Συνεπώς, με ικανό πλήθος testcases το σύστημα μπορεί να ελέγξει την ορθότητα του προγράμματος του φοιτητή χωρίς να βασίζεται σε έναν πρότυπο (σωστό) κώδικα, επιτρέποντας έτσι μεγάλη ελευθερία στον φοιτητή ως προς τον τρόπο υλοποίησης. αυτόν καθεαυτόν Για κάθε testcase ο κώδικας του φοιτητή βαθμολογείται δυαδικά («Επιτυχία» ή «Αποτυχία»). Το άθροισμα των επιτυχιών προς το πλήθος των testcases, απαρτίζουν τη τελική βαθμολογία του φοιτητή για την συγκεκριμένη άσκηση και ανάλογα με τις ρυθμίσεις του συστήματος, ο φοιτητής μπορεί να ενημερωθεί για τον βαθμό του είτε άμεσα (εντός δευτερολέπτων από την στιγμή της υποβολής) ή μετά από κάποια συγκεκριμένη ημέρα/ώρα που έχει καθορίσει ο διδάσκων. Καθώς η διαδικασία βαθμολόγησης είναι πλήρως αυτοματοποιημένη, υπάρχει η δυνατότητα προσθήκης απεριόριστων testcases για κάθε πρόβλημα, με επιθυμητή διαβάθμιση, ούτως ώστε όλοι οι φοιτητές να μπορούν να ανταπεξέλθουν βαθμιαία ενώ παράλληλα να εκπαιδεύονται σταδιακά σε πιο δύσκολες περιπτώσεις του προβλήματος. Η κατασκευή των testcases δεν χρειάζεται να καλύπτει κάποια συγκεκριμένα κριτήρια, και αποφασίζεται αυθαίρετα από τον διδάκσων ξεχωριστά για κάθε άσκηση, ως προς την δομή των αρχείων εισόδου και εξόδου.
Το γεγονός πως δεν γίνεται σύγκριση με έτοιμο κώδικα για κάθε πρόβλημα, αλλά εκτέλεση για προκαθορισμένη ομάδα από testcases, σε συνδυασμό με την πλήρη αποστασιοποίηση του διδάσκοντος από την διαδικασία, καθιστά το σύστημα πλήρως αντικειμενικό ως προς τη βαθμολόγηση των φοιτητών.

4. Συμπεράσματα και Βελτιώσεις

Το σύστημα που περιγράψαμε βρίσκεται ακόμη σε δοκιμαστική έκδοση alpha και θα παραμείνει έτσι μέχρι το τέλος του 2013, ώσπου να επιβεβαιωθεί η σταθερότητά του στις νέες αλλαγές και η απόκρισή του σε πραγματικές συνθήκες. Πάντως είναι ενθαρρυντικές οι πρώτες εντυπώσεις από νέους δοκιμαστικούς χρήστες ως προς την ευκολία χρήσης του, αλλά και η πληρότητά του ως προς τα λειτουργικά χαρακτηριστικά σε σχέση με ρεαλιστικές εκπαιδευτικές ανάγκες, όπως καταγράφονται από διδάσκοντες. Σύμφωνα με το χρονοδιάγραμμα που έχουμε θέσει, από τις αρχές του 2014 η τροποποιημένη πλατφόρμα θα είναι διαθέσιμη ως δοκιμαστική έκδοση beta.

Το σύστημα, εκτός από τη λίστα γλωσσών που αναφέρθηκε στο κεφάλαιο 2.1, θα μπορεί να υποστηρίξει στο μέλλον και ακόμα περισσότερες γλώσσες προγραμματισμού όπως Prolog, LISP και άλλες.

Επιπλέον, για επόμενη έκδοση σχεδιάζεται η ενσωμάτωση υποστήριξης μαθημάτων που περιλαμβάνουν σχεδίαση υλικού (hardware). Συγκεκριμένα, θα κατασκευαστεί και θα ενσωματωθεί στην πλατφόρμα ένας διαδραστικός επεξεργαστής λογικών ψηφιακών κυκλωμάτων, με τη χρήση του οποίου ο φοιτητής θα καλείται να σχεδιάσει ακολουθιακά ή συνδυαστικά λογικά κυκλώματα σε VHDL. Σε περίπτωση σφάλματος, ο φοιτητής θα μπορεί να βλέπει την έξοδο του κυκλώματος που σχεδίασε (ως πίνακα αληθείας) σε αντιπαράθεση με την σωστή λύση.

Σε επόμενο στάδιο, θα γίνει επίσης σύνδεση της διαδικασίας εγγραφής νέων χρηστών με βάση δεδομένων LDAP για την ενοποίηση με συστήματα πρόσβασης φοιτητών σε άλλες υπηρεσίες, όπως πρόσβαση σε ηλεκτρονικό ταχυδρομείο αλλά και στην Πλατφόρμα Ασύγχρονης Τηλεκπαίδευσης - Open eClass, που ήδη υποστηρίζεται από τα περισσότερα πανεπιστημιακά ιδρύματα.

Τέλος, υπάρχει η δυνατότητα να ενσωματωθεί στην πλατφόρμα υποσύστημα εντοπισμού αντιγραφών (Lancaster & Culwin, 2004) μεταξύ αρχείων κώδικα που έχουν κατατεθεί από διαφορετικούς φοιτητές. Η χρήση του υποσυστήματος μπορεί να ρυθμιστεί ώστε να μη λειτουργεί αποτρεπτικά, δηλαδή να μην απορρίπτει τον κώδικα τη στιγμή της κατάθεσης, αλλά υποστηρικτικά προς τον διδάσκοντα, ο οποίος και θα είναι υπεύθυνος της τελικής απόφασης αποδοχής ή απόρριψης κώδικα.

Βιβλιογραφικές Αναφορές

Auffarth B., Lopez-Sonchez M., Campos i Miralles J., & Puig A. (2004) *System for Automated Assistance in Correction of Programming Exercises (SAC).* InL Proceedings of the Fifth CIDUI-V International Congress of University Teaching and Innovation.

Coelho A., Kato1 E., Xavier J., & Gonçalves R. (2011) *Serious Game for Introductory Programming*, LNCS, 6944, 61-71, 2011.

Eldering, J., Kinkhorst, T. & Warken, P. (2010) *DOMjudge Administrators Manual*.

Pacheco, P. (2010) *Computer-based assessment system for e-Learning applied to programming education,* Masters thesis.

Higgins C., Hegazy T., Symeonidis P., & Tsintsifas A. (2003) *The coursemarker cba system: Improvements over ceilidh,* Education and Information Technologies, vol. 8, no. 3, pp. 287–304.

Lancaster T. & Culwin, F. (2004) *A Comparison of Source Code Plagiarism Detection Engines, Computer Science Education,* 14:2, 101-112.

Luck M. & Joy M. (1999) *A secure on-line submission system,* Software Practice and Experience, vol. 29, no. 8, pp. 721–740.