



Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Σχεδίαση και Υλοποίηση μιας Καταπληκτικής Γλώσσας Προγραμματισμού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΓΕΡΑΣΙΜΟΣ Τ. ΙΩΑΝΝΟΥ

Επιβλέπων : Νικόλαος Σ. Παπασπύρου
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2014



Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Σχεδίαση και Υλοποίηση μιας Καταπληκτικής Γλώσσας Προγραμματισμού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΓΕΡΑΣΙΜΟΣ Τ. ΙΩΑΝΝΟΥ

Επιβλέπων : Νικόλαος Σ. Παπασπύρου
Αν. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 17η Ιουλίου 2014.

.....
Νικόλαος Σ. Παπασπύρου
Αν. Καθηγητής Ε.Μ.Π.

.....
Πέτρος Χ. Παπαδόπουλος
Επικ. Καθηγητής Ε.Μ.Π.

.....
Γεώργιος Χ. Νικολάου
Καθηγητής Ε.Κ.Π.Α.

Αθήνα, Ιούλιος 2014

.....
Γεράσιμος Τ. Ιωάννου

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Γεράσιμος Τ. Ιωάννου, 2014.

Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Σκοπός της παρούσας εργασίας είναι αφενός η σχεδίαση μίας απλής γλώσσας υψηλού επιπέδου με υποστήριξη για προγραμματισμό με αποδείξεις, αφετέρου η υλοποίηση ενός μεταγλωττιστή για τη γλώσσα αυτή που θα παράγει κώδικα για μία γλώσσα ενδιάμεσου επιπέδου κατάλληλη για δημιουργία πιστοποιημένων εκτελέσιμων.

Στη σημερινή εποχή, η ανάγκη για αξιόπιστο και πιστοποιημένα ασφαλή κώδικα γίνεται διαρκώς ευρύτερα αντιληπτή. Τόσο κατά το παρελθόν όσο και πρόσφατα έχουν γίνει γνωστά προβλήματα ασφάλειας και συμβατότητας προγραμμάτων που είχαν ως αποτέλεσμα προβλήματα στην λειτουργία μεγάλων συστημάτων και συνεπώς οικονομικές επιπτώσεις στους οργανισμούς που τα χρησιμοποιούσαν. Τα προβλήματα αυτά οφείλονται σε μεγάλο βαθμό στην έλλειψη δυνατότητας προδιαγραφής και απόδειξης της ορθότητας των προγραμμάτων που χαρακτηρίζει τις σύγχρονες γλώσσες προγραμματισμού. Για το σκοπό αυτό, έχουν προταθεί συστήματα πιστοποιημένων εκτελέσιμων, στα οποία έχουμε τη δυνατότητα να προδιαγράψουμε την ορθότητα των προγραμμάτων, και να παρέχουμε μία τυπική απόδειξη αυτής, η οποία μπορεί να ελεγχθεί μηχανιστικά πριν το χρόνο εκτέλεσης.

Τα συστήματα που έχουν προταθεί είναι ενδιάμεσου επιπέδου οπότε η διαδικασία προγραμματισμού σε αυτά είναι ιδιαίτερα πολύπλοκη. Οι γλώσσες υψηλού επιπέδου που συνοδεύουν αυτά τα συστήματα, ενώ είναι ιδιαίτερα εκφραστικές, παραμένουν δύσκολες στον προγραμματισμό. Μία απλούστερη γλώσσα υψηλού επιπέδου, όπως αυτή που προτείνουμε σε αυτή την εργασία, θα επέτρεπε ευρύτερη εξάπλωση του συγκεκριμένου ιδιώματος προγραμματισμού.

Στη γλώσσα που προτείνουμε, ο προγραμματιστής προδιαγράφει τη μερική ορθότητα του προγράμματος, δίνοντας προσυνθήκες και μετασυνθήκες για τις παραμέτρους και τα αποτελέσματα των συναρτήσεων που ορίζει. Επίσης δίνει ένα σύνολο θεωρημάτων βάσει του οποίου κατασκευάζονται αποδείξεις της ορθής υλοποίησης και χρήσης των συναρτήσεων αυτών. Ως μέρος της εργασίας, έχουμε υλοποιήσει σε γλώσσα OCaml ένα μεταφραστή αυτής της γλώσσας στο σύστημα πιστοποιημένων εκτελέσιμων NFLINT.

Επιτύχαμε να διατηρήσουμε τη γλώσσα κοντά στο ύφος των ευρέως διαδεδομένων συναρτησιακών γλωσσών, καθώς και να διαχωρίσουμε τη φάση προγραμματισμού από τη φάση απόδειξης της ορθότητας των προγραμμάτων. Έτσι ένας μέσος προγραμματιστής μπορεί εύκολα να προγραμματίζει στη γλώσσα που προτείνουμε με τον τρόπο που ήδη γνωρίζει, και ένας γνώστης μαθηματικής λογικής να αποδεικνύει σε επόμενη φάση την μερική ορθότητα των προγραμμάτων. Ως απόδειξη της πρακτικότητας της προσέγγισης αυτής, παραθέτουμε ένα σύνολο παραδειγμάτων στη γλώσσα με απόδειξη μερικής ορθότητας.

Λέξεις κλειδιά

Γλώσσες προγραμματισμού, Προγραμματισμός με αποδείξεις, Ασφαλείς γλώσσες προγραμματισμού, Πιστοποιημένος κώδικας.

Abstract

The purpose of this diploma dissertation is on one hand the design of a simple high-level language that supports programming with proofs, and on the other hand the implementation of a compiler for this language. This compiler will produce code for an intermediate-level language suitable for creating certified binaries.

The need for reliable and certifiably secure code is even more pressing today than it was in the past. In many cases, security and software compatibility issues put in danger the operation of large systems, with substantial financial consequences. The lack of a formal way of specifying and proving the correctness of programs that characterizes current programming languages is one of the main reasons why these issues exist. In order to address this problem, a number of frameworks with support for certified binaries have recently been proposed. These frameworks offer the possibility of specifying and providing a formal proof of the correctness of programs. Such a proof can easily be checked for validity before running the program.

The frameworks that have been proposed are intermediate-level in nature, thus the process of programming in these is rather cumbersome. The high-level languages that accompany some of these frameworks, while very expressive, are hard to use. A simpler high-level language, like the one proposed in this dissertation, would enable further use of this programming idiom.

In the language we propose, the programmer specifies the partial correctness of a program by annotating function definitions with pre- and post-conditions that must hold for their parameters and results. The programmer also provides a set of theorems, based on which proofs of the proper implementation and use of the functions are constructed. An implementation in OCaml of a compiler from this language to the NFLINT certified binaries framework was also completed as part of this dissertation.

We managed to keep the language close to the feel of the current widespread functional languages, and also to fully separate the programming stage from the correctness-proving stage. Thus an average programmer can program in a familiar way in our language, and later an expert on formal logic can prove the semi-correctness of a program. As evidence of the practicality of our design, we provide a number of examples in our language with full semi-correctness proofs.

Key words

Programming languages, Programming with proofs, Secure programming languages, Certified code.

Ευχαριστίες

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή αυτής της διατριβής, κ. Γιάννη Παπαδάκη, για τη συνεχή καθοδήγηση και εμπιστοσύνη του. Ευχαριστώ επίσης τα μέλη της συμβουλευτικής επιτροπής, κ.κ. Νίκο Παπαδόπουλο και Γιώργο Νικολάου για την πρόθυμη και πάντα αποτελεσματική βοήθειά τους, τις πολύτιμες συμβουλές και τις χρήσιμες συζητήσεις που είχαμε. Θέλω να ευχαριστήσω ακόμα τον συμφοιτητή και φίλο Πέτρο Πετρόπουλο, ο οποίος με βοήθησε σε διάφορα στάδια αυτής της εργασίας. Θα ήθελα τέλος να ευχαριστήσω την οικογένειά μου και κυρίως τους γονείς μου, οι οποίοι με υποστήριξαν και έκαναν δυνατή την απερίσπαστη ενασχόλησή μου τόσο με την εκπόνηση της διπλωματικής μου, όσο και συνολικά με τις σπουδές μου.

Γεράσιμος Τ. Ιωάννου,
Αθήνα, 17η Ιουλίου 2014

Η εργασία αυτή είναι επίσης διαθέσιμη ως Τεχνική Αναφορά CSD-SW-TR-42-14, Εθνικό Μετσόβιο Πολυτεχνείο, Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών, Εργαστήριο Τεχνολογίας Λογισμικού, Ιούλιος 2014.

URL: <http://www.softlab.ntua.gr/techrep/>
FTP: <ftp://ftp.softlab.ntua.gr/pub/techrep/>

Περιεχόμενα

Περίληψη	5
Abstract	7
Ευχαριστίες	9
Περιεχόμενα	11
Κατάλογος πινάκων	13
Κατάλογος σχημάτων	15
1. Εισαγωγή	17
1.1 Η γλώσσα προγραμματισμού C	17
1.2 Σημασιολογία γλωσσών προγραμματισμού	18
1.3 Θεωρία πεδίων	18
Παράρτημα	21
A. Ευρετήριο συμβολισμών	21
B. Ευρετήριο γλωσσών	23
C. Ευρετήριο αριθμών	25

Κατάλογος πινάκων

Κατάλογος σχημάτων

Εισαγωγή

1.1 Η γλώσσα προγραμματισμού C

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα
μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα,
μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα
μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα
μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα,
μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα

[illegible]

Παράρτημα Α

Ευρετήριο συμβολισμών

$A \rightarrow B$: συνάρτηση από το πεδίο A στο πεδίο B .

Παράρτημα Β

Ευρετήριο γλωσσών

Haskell : η γλώσσα της ζωής μου.

Παράρτημα C

Ευρετήριο αριθμών

42 : life, the universe and everything.