

Λογοτεχνικός προγραμματισμός: Ή πώς να δημιουργείτε προγράμματα καθώς περιγράφετε το τι θέλετε να κάνει ο Η/Υ

Απόστολος Συρόπουλος

28ης Οκτωβρίου 366

671 00 ΞΑΝΘΗ

apostolo@obelix.ee.duth.gr

Abstract

In this article we present the literate programming methodology developed by Donald E. Knuth. In particular, we present the methodology and its advantages. Moreover, we present noweb, the literate programming tool developed by Norman Ramsey.

Κάνοντας τα πρώτα μου βήματα στο χώρο του προγραμματισμού έμαθα ότι ένα καλογραμμένο πρόγραμμα θα πρέπει πάντα να συνοδεύεται από μια καλή επεξήγηση της λειτουργίας του. Αυτό, συνήθως, γίνεται με την προσθήκη σχολίων κατά τη διάρκεια γραφής του κώδικα ή, στη χειρότερη των περιπτώσεων, αφού τελειώσει η διαδικασία αυτή. Επιπλέον, κάποιος αναλαμβάνει να γράψει ένα τεχνικό κείμενο το οποίο περιγράφει γενικά τον τρόπο λειτουργίας του προγράμματος και το οποίο αφορά μόνο προγραμματιστές. Προφανώς θα ήταν ευχής έργον να υπάρχει ένας τρόπος ανάπτυξης προγράμματος όπου όλες οι παραπάνω διαδικασίες θα γινόταν ταυτόχρονα και με τον καλύτερο δυνατό τρόπο. Αν πιστεύετε ότι αυτό δεν είναι δυνατό, τότε σίγουρα δεν έχετε ακούσει τίποτα για τον λογοτεχνικό προγραμματισμό.¹

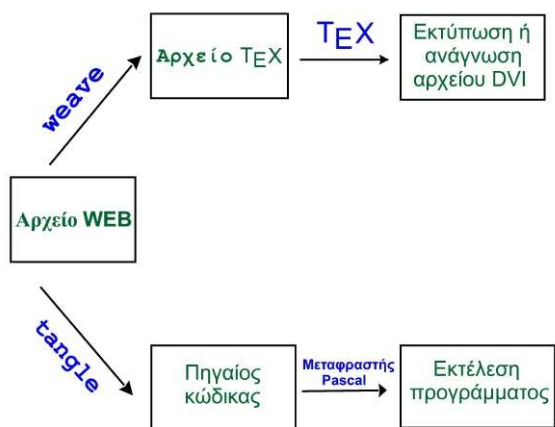
¹ Ο όρος *λογοτεχνικός προγραμματισμός* αποτελεί μετάφραση του αγγλικού όρου *literate programming*. Στην ορολογία των ανθρωπιστικών σπουδών, ο όρος *literate* μεταφράζεται ως *εγγράμματος* (-η, -ο) και ο όρος *literacy* ως *εγγραμματισμός*. Όμως το να μιλούμε για *εγγράμματος προγραμματισμό* θα ακουγόταν κάπως παράξενο στα αυτιά των προγραμματιστών, γι' αυτό και επικράτησε ο όρος *λογοτεχνικός προγραμματισμός*. Αν κάποιος άλλος έχει να προτείνει έναν καλύτερο όρο, ευχαρίστως να τον συζητήσουμε.

Ο λογοτεχνικός προγραμματισμός είναι μια μεθοδολογία ανάπτυξης προγράμματος. Η καινοτομία του λογοτεχνικού προγραμματισμού βρίσκεται στον τρόπο αντιμετώπισης του προγραμματισμού ως καθημερινή δραστηριότητα. Έτσι, αντί να φανταζόμαστε ότι η βασική μας αποστολή είναι να καθοδηγήσουμε τον Η/Υ για το τι πρέπει να κάνει, επικεντρώνουμε στο να επεξηγήσουμε σε ανθρώπους το τι θέλουμε να κάνει ο Η/Υ. Κατά αυτό τον τρόπο ο προγραμματιστής γίνεται ένας χρονογράφος, του οποίου η βασική επιδίωξη είναι η άψογη επεξήγηση και η τελειότητα του στυλ. Ο λογοτεχνικός προγραμματιστής διαλέγει προσεκτικά τα ονόματα των μεταβλητών και εξηγεί την χρήση κάθε μιας. Ο λογοτεχνικός προγραμματιστής μοχθεί για την δημιουργία ενός προγράμματος το οποίο είναι κατανοητό, εξαιτίας του γεγονότος ότι οι διάφορες έννοιες έχουν εισαχθεί κατά τέτοιο τρόπο ώστε να γίνονται κατανοητές από τους αναγνώστες του.

Η μεθοδολογία του λογοτεχνικού προγραμματισμού αποτελεί εφεύρεση του γνωστού καθηγητή της επιστήμης των Η/Υ Donald Knuth. Ο λογοτεχνικός προγραμματισμός εφευρέθηκε κατά την διάρκεια σχεδιασμού της δεύτερης έκδοσης του T_EX και του METAFONT, δύο πασίγνωστων προγραμμάτων τα οποία σχεδίασε ο Κνυτη. Ο εφευρέτης αυτής της μεθοδολογίας προγραμματισμού, πίστευε ότι επειδή ο προγραμματισμός αποτελεί μια καθαρά προσωπική δραστηριότητα, αυτό που για αυτόν είχε τρομακτικά αποτελέσματα, ίσως για κάποιους άλλους να μην είχε τα προσδοκώμενα αποτελέσματα. Ο χρόνος έδειξε ότι η μεθοδολογία αυτή επιφέρει τα προσδοκώμενα αποτελέσματα, ανεξάρτητα των όποιων ιδιαίτερων χαρακτηριστικών οποιοδήποτε προγραμματιστή. Μόλις κανείς καταπιαστεί με τον σχεδιασμό ενός προγράμματος χρησιμοποιώντας τη μεθοδολογία του λογοτεχνικού προγραμματισμού, αργά ή γρήγορα ανακαλύπτει πτυχές του προβλήματος αλλά και της λύσης που αρχικά ούτε καν είχαν περάσει από το μυαλό του. Αυτό το γεγονός κάνει το λογοτεχνικό πρόγραμμα ασύγκριτα καλύτερο από κάποιο αντίστοιχο που αναπτύχθηκε με κάποια συμβατική μέθοδο. Τούτη η διαπίστωση κάνει και τους πιο σκεπτικούς να υιοθετήσουν αυτή τη μεθοδολογία για την ανάπτυξη των προγραμμάτων των.

Στην αρχική του μορφή ένα λογοτεχνικό πρόγραμμα αποτελείται από «ανάμιχτες» ενότητες κώδικα στη γλώσσα προγραμματισμού Pascal και χειμένου στα αγγλικά. Η κάθε ενότητα περιέχει τον κώδικα που λύνει κάποιο επιμέρους πρόβλημα αλλά και την επεξήγηση της λειτουργίας του. Όλες μαζί οι ενότητες κώδικα συνθέτουν το τελικό πρόγραμμα. Ο αναγνώστης ίσως ανακαλύπτει κάποια σχέση μεταξύ του λογοτεχνικού προγραμματισμού και του δομημένου προγραμματισμού. Η σχέση δεν είναι τυχαία, αλλά είναι γεγονός ότι ο λογοτεχνικός προγραμματισμός είναι μια πιο γενική μεθοδολογία, αφού μπορεί να εφαρμοστεί σε προγραμματιστικά στυλ όπου αποτυγχάνει ο δομημένος προγραμματισμός. Κάθε λογοτεχνικό πρόγραμμα ονομάζεται WEB (ιστός) και φυλάσσεται σε κάποιο αρχείο με προέκταση web. Ένα ειδικό πρόγραμμα, το tangle (ξεμπλέκω), αναλαμβάνει να εξάγει τον κώδικα Pascal από το WEB, ενώ ένα άλλο πρόγραμμα,

Σχήμα 1



Διττή χρήση αρχείου WEB

το weave (υφαίνω), παράγει ένα αρχείο TEX το οποίο περιέχει την τεκμηρίωση (documentation) του προγράμματος.

Στο Σχήμα 1 φαίνεται η διττή χρήση του ενός αρχείου WEB. Έτσι λοιπόν κανείς γράφει σε ένα αρχείο, π.χ., foo.web, το λογοτεχνικό του πρόγραμμα και με την εντολή tangle foo.web παίρνει το αρχείο foo.p το οποίο περιέχει τον κώδικα Pascal. Στη συνέχεια το πρόγραμμα αυτό μπορεί να μεταφραστεί σε γλώσσα μηχανής με τον αγαπημένο σας compiler (αποτελεί γνώμη μου ότι ο όρος «μεταφραστής» δεν αποδίδει το ακριβές νόημα της λέξεως compiler, ενώ δυστυχώς δεν έχει πέσει στην αντίληψη μου κάποια άλλη σωστότερη απόδοσή της). Από την άλλη με την εντολή weave foo.web παράγεται το αρχείο foo.tex το οποίο μπορούμε να τροφοδοτήσουμε στο TEX και στην συνέχεια μπορούμε να τυπώσουμε το στοιχειοθετημένο αποτέλεσμα που περιέχεται στο αρχείο foo.dvi. Γίνεται λοιπόν κατανοητό ότι γενικά το πρόγραμμα tangle παράγει πηγαίο κώδικα σε κάποια γλώσσα προγραμματισμού, ενώ το πρόγραμμα weave παράγει ένα αρχείο σε κάποιο γλώσσα μορφοποίησης κειμένου. Γιατί όμως ο Knuth διάλεξε την Pascal και το TEX; Όμως ο λόγος για τον οποίο ο Knuth διάλεξε την Pascal είναι προφανής, ενώ ο λόγος για τον οποίο διάλεξε την Pascal είναι ότι την εποχή εκείνη (δεκαετία του 1980) η γλώσσα αυτή διδάσκονταν σε όλα σχεδόν τα πανεπιστήμια του κόσμου (εκτός των ελληνικών...) και έτσι ο πολύς κόσμος θα μπορούσε να χρησιμοποιήσει τα νέα εργαλεία. Αξίζει βέβαια να σημειώσουμε ότι η Pascal δεν αποτέλεσε ποτέ την αγαπημένη γλώσσα προγραμματισμού του Knuth, απλά γιατί δεν παρείχε εκείνες τις δυνατότητες που κάνουν τον προγραμματισμό συστημάτων απλή υπόθεση. Ενώ η μη υποστήριξη χρήσης αλλά και δημιουργίας μακροεντο-

λών αντισταθμίστηκε από την δυνατότητα του προγράμματος `tangle` να χειρίζεται μακροεντολές.

Αν και το σύστημα WEB και γενικότερα ο λογοτεχνικός προγραμματισμός βρήκε γρήγορα πολλούς και πιστούς φίλους, εντούτοις σύντομα έγινε κατανοητό ότι η μεγάλη εξάρτηση του συστήματος από την Pascal έκανε τουλάχιστον δυσχερές την εξάπλωση της ιδέας του λογοτεχνικού προγραμματισμού. Έτσι πολύ σύντομα έκαναν την εμφάνιση τους αρκετά συστήματα WEB για διάφορες γλώσσες προγραμματισμού, όπως η Ada, η Modula-2, η FORTRAN, η APL, η Scheme, κ.λπ. Ειδικά για την γλώσσα προγραμματισμού C (αλλά και την C++), ο Silvio Levy σε συνεργασία με το Donald Knuth δημιούργησαν το σύστημα CWEB, το οποίο αποτελεί κατά τον Knuth το τέλειο προγραμματιστικό εργαλείο. Όμως ακόμη και το CWEB είναι ένα εργαλείο εξαρτημένο από μία γλώσσα προγραμματισμού αλλά και μία γλώσσα μορφοποίησης κειμένου. Για τον λόγο αυτό ο Norman Ramsey σχεδίασε το σύστημα SpiderWEB το οποίο επιτρέπει στον χρήστη του να δημιουργήσει το δικό του WEB για την γλώσσα προγραμματισμού της επιλογής του (ως γλώσσα μορφοποίησης παραμένει το `TEX`). Δηλαδή, για κάθε γλώσσα προγραμματισμού μπορούμε να δημιουργήσουμε τα αντίστοιχα προγράμματα `weave` και `tangle`. Επειδή όμως και το SpiderWEB είχε τα μειονεκτήματά του, σχεδόν ταυτόχρονα ο Preston Briggs και ο Norman Ramsey δημιούργησαν τα συστήματα `nuweb` και `noweb` αντίστοιχα. Το `nuweb` χρησιμοποιεί το `LATEX` ως γλώσσα μορφοποίησης κειμένου, ενώ λειτουργεί με οποιαδήποτε γλώσσα προγραμματισμού. Το `noweb` κάνει ένα βήμα παραπέρα αφού μεν λειτουργεί με οποιαδήποτε γλώσσα προγραμματισμού, ενώ μπορεί και παράγει έξοδο στις γλώσσες μορφοποίησης κειμένου `TEX`, `LATEX`, `troff` και `HTML`. Επειδή το `noweb` είναι το εργαλείο που έχει τραβήξει την περισσότερη προσοχή των μελών της κοινότητας των λογοτεχνικών προγραμματιστών, θα περιγράψουμε τον τρόπο χρήσης του εργαλείου.

Ένα αρχείο `noweb` αποτελείται από «ενότητες» οι οποίες μπορούν να εμφανίζονται με όποια σειρά εμείς θέλουμε και μπορούν να περιέχουν τεκμηρίωση ή/και κώδικα. Το τμήμα της τεκμηρίωσης πρέπει να ξεκινάει με το σύμβολο `@` στην πρώτη στήλη και να ακολουθείτε από ένα τουλάχιστον κενό ή μια αλλαγή γραμμής. Το τμήμα κώδικα ξεκινάει με `<<'onoma tm'hmatos>>=` το οποίο θα πρέπει να βρίσκεται σε μια γραμμή από μόνο του και να ξεκινάει από την πρώτη στήλη. Ως όνομα τμήματος μπορεί να είναι μια λέξη ή και μία ολόκληρη φράση. Μια ενότητα τερματίζεται με την αρχή μιας νέας ενότητας.

Τα τμήματα τεκμηρίωσης περιέχουν κείμενο το οποίο αγνοείται από το αντίστοιχο πρόγραμμα `tangle` του `noweb` που ονομάζεται `notangle`, ενώ το αντίστοιχο πρόγραμμα `weave` του `noweb` που ονομάζεται `noweave` το εμφανίζει στην οθόνη του Η/Υ αφού προηγουμένα το περάσει από ένα φίλτρο ώστε να το πάρουμε στη μορφή που επιθυμούμε. Το `noweb`, όπως αναφέραμε, μπορεί να συνεργαστεί με το `LATEX` το `plain TEX`, το `troff` ή την `HTML`. Με το `plain TEX` εισαγάγει μια αναφορά σε ένα μακροπακέτο `TEX`, το `nwmac`, το οποίο ορίζει εντολές όπως η

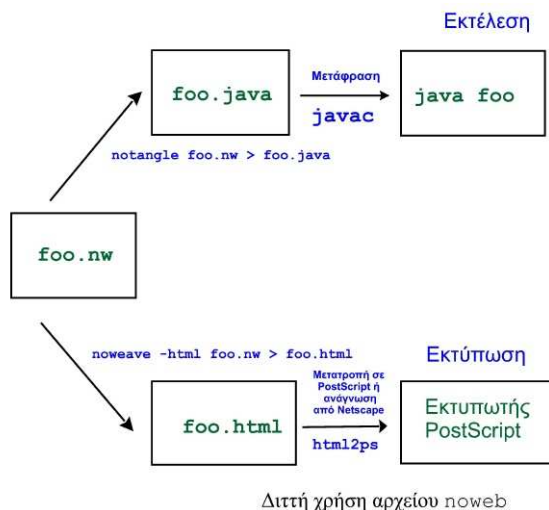
`\chapter`, για κεφάλαια, και η `\section`, για ενότητες. Με το L^AT_EX χρησιμοποιεί το πακέτο `noweb`, ενώ με την HTML τα πράγματα είναι απλά αφού παράγει ένα ολοκληρωμένο αρχείο χωρίς αναφορές σε τίποτα.

Τα τμήματα κώδικα περιέχουν πηγαίο κώδικα και αναφορές σε άλλα τμήματα κώδικα άλλων ενοτήτων. Η συγγραφή ενός τμήματος κώδικα είναι κάτι ανάλογο με τη δημιουργία μιας μακροεντολής: το `notangle` εξ ορισμού εξάγει ένα πρόγραμμα με το να αναπτύξει ένα βασικό τμήμα κώδικα, συνήθως με όνομα `<<*>>`. Αυτό περιέχει αναφορές σε άλλα τμήματα κώδικα, τα οποία με την σειρά τους περιέχουν άλλες αναφορές· όλες αυτές οι αναφορές αναπτύσσονται πλήρως και έτσι τελικά παράγεται και εκτυπώνεται στην οθόνη του Η/Υ ο κώδικας του προγράμματος μας. Το καλό με το `notangle` είναι ότι δεν αλλοιώνει τη μορφή της γραφής του κώδικα, όπως κάνουν τα παραδοσιακά συστήματα. Αν σε ένα τμήμα τεκμηρίωσης θέλουμε να γράψουμε ένα μικρό τμήμα κώδικα, θα πρέπει να το γράψουμε μεταξύ των συμβόλων `[[και]]`, π.χ.: `[[i++]]` ή `[[a[6]]]`. Αν θέλουμε να έχουμε το σύμβολο `@` στη πρώτη στήλη μιας γραμμής κώδικα, τότε απλά γράφουμε δύο φορές το σύμβολο, δηλ. `@@`. Αν θέλουμε τα σύμβολα `<< και >>` να χάσουν την προκαθορισμένη σημασία των, απλά βάζουμε μπροστά των το σύμβολο `@` (π.χ., `@<<`). Αν μία γραμμή έχει τη μορφή `@ %def ονόματα`, αυτό σημαίνει ότι η προηγούμενη ενότητα όριζε τα ονόματα που εμφανίζονται στη λίστα των ονομάτων. Τα ονόματα χωρίζονται με κενούς χαρακτήρες.

Τα δύο προγράμματα, δηλαδή το `noweave` και το `notangle`, δέχονται μια σειρά από ορίσματα γραμμής εντολών. Αναφέρουμε τα δύο πιο σπουδαία για το `notangle` πρώτα.

1. Αν θέλουμε να μετατρέψουμε τα τμήματα τεκμηρίωσης σε σχόλια, τότε απλά γράφουμε τον τύπο σχολίων που υποστηρίζει η γλώσσα μας. Το σύστημα αναγνωρίζει σχόλια της μορφής των παρακάτω γλωσσών (ορισμάτων): AWK (`-awk`), C (`-c`), ICON (`-icon`), SML (`-ml`), Modula-3 (`-m3`), Pascal (`-pascal`), FORTRAN77 (`-f77`) και FORTRAN90 (`-f90`). Εξ ορισμού θεωρεί ότι τα σχόλια είναι της μορφής που αναγνωρίζει η C.
2. Αν για κάποιο λόγο έχουμε κάποιο λάθος καλό είναι να μπορούμε βλέποντας το αρχείο του κώδικα να διορθώνουμε αμέσως το λογοτεχνικό πρόγραμμα. Η σύνδεση αυτή γίνεται δυνατή με το όρισμα `-L` και την παράμετρό της, η οποία καθορίζει την μορφή ενός σχολίου, που όταν εμφανίζεται στο αρχείο κώδικα δηλώνει την αντίστοιχη γραμμή στο αρχείο του λογοτεχνικού προγράμματος. Η παράμετρος είναι ένα κορδόνι στο οποίο τα σύμβολα `%F`, `%L` και `%N` δηλώνουν το όνομα του πηγαίου αρχείου, τον αριθμό γραμμής και αλλαγή γραμμής αντίστοιχα. Το σύμβολο `%%` δηλώνει απλά το σύμβολο «επί τοις εκατό». Για παράδειγμα για την Java η παράμετρος μπορεί να είναι `//line %L "%F" %N`, ενώ για την Pascal μπορεί να είναι `(* line %L "%F" *) %N`.

Σχήμα 2



Στην περίπτωση του `noweave` τα πιο σημαντικά ορίσματα καθορίζουν το είδος του παραγόμενου αρχείου, καθώς και κάποια επιπλέον χαρακτηριστικά του. Αν δεν προκαθορίσουμε το είδος του αρχείου που θέλουμε να δημιουργήσει το πρόγραμμα, αυτό δημιουργεί αρχείο \LaTeX . Αλλιώς, μπορούμε να δηλώσουμε ότι επιθυμούμε τη παραγωγή αρχείου plain \TeX με την παράμετρο `-tex`, αρχείου HTML με την παράμετρο `-html` ή αρχείου troff με την παράμετρο `-troff`. Αν επιθυμούμε τα τμήματα κώδικα να είναι σε HTML και τα τμήματα κειμένου σε \LaTeX ώστε να μπορούμε να δημιουργήσουμε άψογα αισθητικά αρχεία HTML με το πρόγραμμα `\LaTeX2html`, χρησιμοποιούμε την παράμετρο `-\LaTeX+html`. Τέλος, είναι δυνατή η δημιουργία ευρετηρίου ενοτήτων και μεταβλητών με την παράμετρο `-index`.

Στο Σχήμα 2 φαίνεται ένα τυπικό παράδειγμα της διττής χρήσης ενός αρχείου `noweb`.

Ο λογοτεχνικός προγραμματισμός αναμφισβήτητα είναι μια καινοτόμα μεθοδολογία προγραμματισμού η οποία αυξάνει την παραγωγικότητα μιας εταιρείας, αφού τεκμηρίωση και κώδικας αναπτύσσονται ταυτόχρονα με τον καλύτερο δυνατό τρόπο. Παράλληλα, έχει αποδειχθεί ότι ένας προγραμματιστής που αναπτύσσει προγράμματα με την μέθοδο του λογοτεχνικού προγραμματισμού, είναι πολύ πιο παραγωγικός από τον παραδοσιακό συνάδελφό του. Επιπλέον, ο λογοτεχνικός προγραμματισμός ως προγραμματιστική μεθοδολογία έχει βρει τη θέση του στα προγράμματα σπουδών πολλών τμημάτων Πληροφορικής, με πολύ ενθαρρυντικά αποτελέσματα. Λόγω του ότι ακόμη τα προγράμματα δεν αναπτύσσονται ως λογοτεχνικά προγράμματα, κυρίως λόγω άγνοιας, κάποιος ίσως αναρωτηθεί για το

αν είναι δυνατό να μετατραπεί ένα απλό πρόγραμμα σε λογοτεχνικό. Μια απάντηση στο πρόβλημα αυτό δίνει μια απλή και συνάμα αποδοτική μέθοδος, η μέθοδος «Spaniel», προτάθηκε από τον γράφοντα και τον Νίκο Χατζηγεωργίου/²

Υπάρχουν πολλά παραδείγματα χρήσης του λογοτεχνικού προγραμματισμού, για την ανάπτυξη πολύπλοκων προγραμμάτων. Αρχικά, ο λογοτεχνικός προγραμματισμός χρησιμοποιήθηκε για την ανάπτυξη της έκδοσης 2 του METAFONT και της έκδοσης 3 του T_EX, δύο πολύ γνωστών προγραμμάτων. Παράλληλα, πολλά συστήματα και εργαλεία, σχετικά με την ψηφιακή στοιχειοθεσία αναπτύχθηκαν ως λογοτεχνικά προγράμματα, όπως για παράδειγμα το METAPOST, το Ωμέγα, κ.ά. Επίσης, ο Norman Ramsey ανέπτυξε ένα λογοτεχνικό πρόγραμμα 33000 περίπου γραμμών, εκ των οποίων οι 15000 γραμμές είναι τεκμηρίωσή (ως γλώσσα προγραμματισμού χρησιμοποιήθηκε η Ada). Επίσης ένας πολύ γνωστός μεταφραστής της γλώσσας C, ο lcc, αποτελεί ένα ακόμη τεράστιο λογοτεχνικό πρόγραμμα το οποίο μάλιστα εκδόθηκε σε μορφή βιβλίου από την Addison-Wesley (βλέπε <http://www.cs.princeton.edu/software/lcc/>). Ακόμη, ο μεταφραστής μιας ειδικής γλώσσας σχεδιασμένης να εκφράζει αλγόριθμους νευρωνικών δικτύων που μαθαίνουν, της CuPit, είναι ένα εξαιρετικό λογοτεχνικό πρόγραμμα (βλέπε <http://www.ipd.ira.uka.de/~hopp/cupit.html>). Ακόμη μπορείτε να βρείτε δεκάδες προγράμματα σε άρθρα αλλά και ιστοσελίδες, π.χ. <http://obelix.ee.duth.gr/~apostolo>.

Αν κάποιος επιθυμεί να μάθει σχετικά πράγματα για τον λογοτεχνικό προγραμματισμό γενικότερα, αλλά και το noweb ειδικότερα, μπορεί να συμβουλευθεί κάποιο από τα παρακάτω URL. Αν ενδιαφέρεστε για κάποιο βιβλίο σχετικά με τον λογοτεχνικό προγραμματισμό, τότε θα πρέπει να ρίξετε μια ματιά στην παρακάτω σελίδα στο URL: <http://www.loria.fr/services/tex/english/texbib.html> Αν απλά ενδιαφέρεστε να μάθετε για άρθρα σχετικά με τον λογοτεχνικό προγραμματισμό, τότε αξίζει να ρίξετε μια ματιά στην σελίδα που ετοίμασε ο Nelson Bebe στο URL <http://www.math.utah.edu/pub/tex/bib/litprog.html> Τέλος, μπορείτε να μάθετε ότι θέλετε σχετικά με το noweb από την σελίδα του στο URL: <http://www.cs.virginia.edu/~nr/noweb/>.

Δημοσιεύτηκε στο τεύχος
Ιουλίου-Αυγούστου 1999 του
περιοδικού RAM.

² N. Hatzigeorgiu, A. Syropoulos (1998), "Literate programming and the 'Spaniel' Method." *SIGPLAN Notices*, vol. 33, no. 12, pp. 52-56.