SEPDS: ΕΝΑ ΠΕΡΙΒΑΛΛΟΝ ΓΙΑ ΤΗΝ ΠΡΩΤΟΤΥΠΟΠΟΙΗΣΗ ΚΑΤΑΝΕΜΗΜΕΝΩΝ ΣΥΣΤΗΜΑΤΩΝ

Αχιλλέας Καμέας Στέργιος Παπαδημητρίου Κώστας Αποστολάς Γιώργος Παυλίδης

Τμήμα Μηχ/κων Ηλεκτρονικών Υπολογιστών & Πληροφορικής, Παν/μιο Πατρών

Abstract

Στην εργασία αυτή παρουσιάζεται το SEPDS, ένα περιβάλλον που υποστηρίζει τον σχεδιασμό και την πρωτοτυποποίηση διαλεκτικών (interactive) κατανεμημένων εφαρμογών. Οι εφαρμογές σχεδιάζονται χρησιμοποιώντας δύο μοντέλα βασισμένα σε Data–Flow Graphs: το EDFG (για τον καθορισμό της λειτουργικότητας) και το IDFG (για τον καθορισμό του τρόπου επικοινωνίας με τον τελικό χρήστη). Πέρα από τα εργαλεία σχεδιασμού της λειτουργικότητας και του user interface της εφαρμογής, το SEPDS υποστηρίζει ακόμα την εκτέλεση και την μέτρηση της απόδοσης πρωτοτύπων της κατανεμημένης εφαρμογής, ώστε να είναι δυνατό να ανακαλυφθούν τα λάθη σχεδιασμού, οι ατέλειες και οι παρανοήσεις νωρίς στην διαδικασία ανάπτυξης, με αποτέλεσμα να μειώνεται το κόστος ανάπτυξης εφαρμογών. Η πρόσβαση στα εργαλεία του συστήματος υποστηρίζεται από ένα σύγχρονο WIMP–type user interface, το οποίο επιτρέπει αποτελεσματική χρήση του συστήματος και παρέχει έξυπνη βοήθεια σε ορισμένες φάσεις της διαδικασίας σχεδιασμού.

1 Εισαγωγή

Η αύξηση του μεγέθους και της πολυπλοκότητας των software projects στα τελευταία χρόνια, έχει οδηγήσει στην ανάγκη κατασκευής περιβαλλόντων ανάπτυξης software (Software Development Enviroments) [4]. Τα περιβάλλοντα αυτά (SDEs) στοχεύουν στην παροχή βοήθειας, και άρα διευκολύνουν την ανάπτυξη μεγάλων συστημάτων. Επίσης υποστηρίζουν βελτιωμένη ποιότητα προγραμμάτων και αυξημένη παραγωγικότητα καθώς σχετίζονται με (και αυτοματοποιούν) ολοένα περισσότερα τμήματα της διαδικασίας παραγωγής software. Ορισμένα από τα SDEs είναι προσανατολισμένα στο να παρέχουν τεχνική κάλυψη σε όλα τα στάδια του μοντέλου ζωής για τα προγράμματα (software life cycle model) [15]. Σκοπός των περιβαλλόντων αυτών είναι να παρέχουν στους χρήστες όλα τα απαραίτητα εργαλεία για το σχεδιασμό, το χρονοπρογραμματισμό και τον έλεγχο κατά τη διάρκεια ολόκληρης της ζωής ενός software πακέτου. Βέβαια άλλα συστήματα είναι προσανατολισμένα σε συγκεκριμένου τύπου δραστηριότητες όπως είναι ο σχεδιασμός ενός συστήματος software ή ο προγραμματισμός και η ανάλυση ενός συστήματος.

Το SEPDS [8] ανήκει στη δεύτερη κατηγορία των SDEs[1] και προσανατολίζεται στη γρήγορη πρωτοτυποποίηση κατά την διαδικασία ανάπτυξης software. Σαν πρωτότυπο θεωρείται ένα συγκεκριμένο εκτελέσιμο μοντέλο από επιλεγμένα τμήματα ενός ευρύτερου προτεινόμενου συστήματος. Ετσι, γρήγορη πρωτοτυποποίηση είναι η διαδικασία του σύντομου κτισίματος και δοκιμής μιας σειράς πρωτοτύπων.

[1]Το SEPDS σχεδιάστηκε και αναπτύσσεται σε συνεργασία TU Delft (Ολλανδία) και Παν/μιου Πατρών. Η ανάπτυξη γίνεται κυρίως με τη χρηση Ada σε Sun workstations.

Τα προβλήματα της ανάπτυξης software είναι ιδιαίτερα εμφανή κατά τη διαδικασία της πρωτοτυποποίησης γιατί τα πρωτότυπα γίνονται αντικείμενα συχνών και επαναλαμβανομένων αλλαγών. Τα πλεονεκτήματα των πρωτοτύπων οφείλονται σημαντικά στην ικανότητα μετατροπής του πρωτοτύπου με πολλή λιγότερη προσπάθεια και κόστος από την αλλαγή του ίδιου του παραγόμενου software. Η πρωτοτυποποίηση με τη βοήθεια του υπολογιστή παρέχει αυτόματη βοήθεια και αυξάνει την ποιότητα και την απόδοση κατά τη διαδικασία ανάπτυξης, γιατί εφοδιάζει με software εργαλεία που βοηθούν το χρήστη στην κατασκευή και την εκτέλεση των πρωτοτύπων γρήγορα και συστηματικά [10]. Με τη χρήση τέτοιων εργαλείων η χρήση των πρωτοτύπων γίνεται ιδιαίτερα ελκυστική, τόσο για να γίνουν αλλαγές ακόμα και αν έχει παραχθεί κάποια έκδοση του software συστήματος, όσο και για να υλοποιηθεί η πρώτη του έκδοση. Η πρωτοτυποποίηση που βασίζεται σε αντικείμενα (object based prototyping) παρέχει απλότητα, ευλυγισία και διευκολύνει την αυτοματοποίηση της διαδικασίας αυτής.

Οι κυριώτερες δυνατότητες που ξεχωρίζουν το SEPDS από άλλα συστήματα πρωτοτυπόποιησης [11,9,5] είναι:

- Η διαδικασία στηρίζεται στη μοντελοποίηση με βάση το EDFG μοντέλο (περιγράφεται παρακάτω) και αυτό αποτελεί την καλύτερη δυνατή ενδιάμεση λύση ανάμεσα στα προτεινόμενα τυπικά [5] και γραφικά [9] μοντέλα.

- Το ίδιο το σύστημα παρέχει στο χρήστη τη δυνατότητα να λύνει τα προβλήματα που αφορούν στο διαμοιρασμό (partitioning) ενός συστήματος σε διεργασίες (processes), στο βαθμό παραλληλοποίησης των διεργασιών και στο μηχανισμό δέσμευσης πόρων.

Για να υποστηρίξει τις δυνατότητες αυτές το SEPDS περιέχει εργαλεία για την κατασκευή, την εξομείωση και τη μέτρηση (profiling) ενός πρωτοτύπου κατανεμημένου συστήματος. Με τη χρήση των εργαλείων αυτών, ο σχεδιαστής μπορεί να κτίσει ένα πρωτότυπο προχωρώντας από γενικές σε ειδικότερες περιγραφές (top down) και στη συνέχεια να το διαμοιράσει ώστε να επιτύχει τον επιθυμητό βαθμό παραλληλισμού. Οι βασικές έννοιες στις οποίες στηρίζονται ο σχεδιασμός των εργαλείων αυτών αλλά και η υλοποίησή τους θα παρουσιαστούν αμέσως παρακάτω. Επειτα θα παρουσιαστεί η αρχιτεκτονική του συστήματος και τα διάφορα εργαλεία του, και θα κλείσουμε με τις μελλοντικές ερευνητικές κατευθύνσεις.

2 Βασικές έννοιες

Το SEPDS χρησιμοποιεί δύο μοντέλα βασισμένα σε data–flow graphs [6] για την περιγραφή των προδιαγραφών του συστήματος προς πρωτοτυποποίηση. Για την περιγραφή του τρόπου λειτουργίας χρησιμοποιείται το μοντελο EDFG (Extended Data–Flow Graph), ενώ για την περιγραφή της επικοινωνίας χρήστη–συστήματος (user–system interaction) χρησιμοποιείται το μοντέλο IDFG (Interactive Data–Flow Graph). Ετσι το SEPDS πετυχαίνει τον κατάλληλο βαθμό εξάρτησης μοντέλου συστήματος και μοντελου επικοινωνίας [2] χωρίς να επιβαρύνει τον σχεδιαστή με κάποιο διαφορετικό τρόπο προγραμματισμού. Υπενθυμίζεται ότι η διαδικασία σχεδίασης του συστήματος είναι η διαδοχική, από πάνω προς τα κάτω εκλέπτυνση (top down refinement), και ο στόχος είναι να επιτευχθεί ο βέλτιστος διαμοιρασμός των διεργασιών του.

2.1 Το μοντέλο EDFG

Ενα EDFG είναι ένας διμερής κατευθυνόμενος γράφος που περιέχει δύο τύπους κόμβων. Ο ένας τύπος αναφέρεται με το όνομα link και ο άλλος αναγνωρίζεται με το όνομα actor. Οι actors περιγράφουν λειτουργίες ενώ τα links λαμβάνουν δεδομένα από έναν actor και τα στέλνουν σε έναν ή περισσότερους actors, μέσω συνδετικών ακμών που καλούνται edges. Γενικά ένας actor μπορεί να εκτελεστεί όταν οι κατάλληλες είσοδοι περιέχουν tokens[2]. Ενας κόμβος που έχει άδεια εκτέλεσης αποσπά και καταναλώνει tokens από τις ακμές εισόδου. Παράλληλα δημιουργεί και εμφανίζει tokens στις γραμμές εξόδου του. Το σύνολο των links εισόδου ενός actor ορίζει το *σύνολο πυροδότησης εισόδου actor (Input Firing semantic Set)* και αντίστοιχα το σύνολο των links εξόδου ορίζει το *σύνολο πυροδότησης εξόδου actor (Output Firing semantic Set)*. Για να ορίσουμε έναν actor α σε ένα EDFG πρέπει να ορίσουμε την πεντάδα $(\delta(\alpha), PRE(\alpha), POST(\alpha), FUN(\alpha), TYPE(\alpha))$ όπου :

1. δ(α) είναι ο χρόνος πυροδότησης (firing time) και αναφέρεται στο χρόνο που σχετίζεται με την εκτέλεση του actor. Για να χρησιμοποιηθεί το EDFG σαν εργαλείο ανάλυσης απόδοσης, είναι απαραίτητο να αντιστοιχίσουμε σε κάθε γεγονός το χρόνο που χρειάζεται για να ολοκληρωθεί. Ο χρόνος αυτός μπορεί να είναι σταθερός για οποιαδήποτε πυροδότηση ενός actor, αλλά μπορεί και να εξαρτάται και από τις τιμές ορισμένων δεδομένων που χρησιμοποιεί ο actor. Λέμε ότι ο actor α είναι σε φάση πυροδότησης για μία περίοδο ίση με το χρονικό διάστημα δ(α) που ακολουθεί την πυροδότηση του actor. Η παράμετρος δ(α) μπορεί να χρησιμοποιηθεί για τη μοντελοποίηση συστημάτων πραγματικού χρόνου (real time systems) που χρειάζονται πεπερασμένης διάρκειας χρόνο για να ολοκληρώσουν την εκτέλεσή τους [12,13].

2. Η παράμετρος $PRE(\alpha)$ καθορίζει τις συνθήκες που απαιτούνται για να πυροδοτηθεί ένας actor. Οι συνθήκες εκφράζονται με τη μορφή λογικής (boolean) συνάρτησης των συνόλων πυροδότησης ειδόδου και εξόδου του actor και έτσι υπάρχει η δυνατότητα να εκφράζεται το στοιχείο IFS σα μεταβλητή.

3. Η παράμετρος $POST(\alpha)$ αναφέρεται στις συνθήκες που προκύπτουν σαν αποτέλεσμα της πυροδότησης του actor. Δηλαδή περιγράφει το τρόπο που τα tokens κατανέμονται στα links εξόδου του actor μετά την ολοκλήρωση της εκτέλεσής του. Η παράμετρος αυτή μοντελοποιεί τις περιπτώσεις εκείνες, που tokens παρουσιάζονται σε ένα υποσύνολο του συνόλου πυροδότησης εξόδου του actor. Και εδώ χρησιμοποιείται λογική έκφραση που συντάσσεται ανάλογα με την παράμετρο $PRE(\alpha)$.

4. Το πεδίο $FUN(\alpha)$ περιγράφει τη συνάρτηση που εκτελεί ο actor.

5. Το πεδίο $TYPE(\alpha)$ καθορίζει αν ο actor είναι στοιχειώδης ή όχι. Χρησιμεύει στην επιλογή περισσοτέρων από ένα επιπέδων περιγραφής (άρα και λεπτομέρειας) ενός actor. Αν ένας actor δεν είναι στοιχειώδης γράφος στο EDFG, τότε έχει μία εσωτερική δομή που με τη σειρά της είναι ένας νέος EDFG.

Το κύριο πλεονέκτημα της χρησιμοποίησης των συναρτήσεων PRE και $POST$ είναι ότι μπορούμε να ορίζουμε links που περιέχονται στο σύνολο πυροδότησης εισόδου, τα οποία διατηρούν

[2]Token είναι μια αφηρημένη δομή δεδομένων

τα tokens τους όταν ο actor πυροδοτείται. Τα links αυτά καλούνται links κατάστασης (state links) επειδή κατέχουν και διατηρούν πληροφορία που σχετίζεται με τη τρέχουσα κατάσταση του όλου υπολογισμού. Κατά τη διάρκεια της διαδικασίας περιγραφής ενός actor σε βαθύτερα επίπεδά του, υπάρχουν links κατάστασης που χρειάζεται να συνδεθούν με περισσότερους από έναν actors, γιατί όλοι αυτοί οι actors χρειάζονται τα tokens των state links που αναφέραμε. Ετσι η αντιστοίχιση ονομάτων (labeling) των links κατάστασης είναι ένας τρόπος για να αναγνωρίζεται η εξάρτηση των δεδομένων (data dependency) ανάμεσα σε actors όταν χρειάζεται κάτι τέτοιο. Η παραδοχή αυτή επιτρέπει την ενσωμάτωση ή συγχώνευση ενός actor και ορισμένων links, είτε αυτά είναι κατάστασης είτε ειδόδου ή εξόδου, σε ένα ανεξάρτητο αντικείμενο (object) του οποίου τα links κατάστασης αναπαριστούν την εσωτερική κατάσταση του actor, ενώ τα υπόλοιπα links ορίζουν τον τρόπο επικοινωνίας με τον actor αυτό.

Σύμφωνα, λοιπόν, με την προσέγγιση αυτή, ένα κατανεμημένο σύστημα περιγράφεται ως ένα σύνολο από γράφους που επικοινωνούν μεταξύ τους, έτσι ώστε κάθε EDFG να αναπαριστά ένα αντικείμενο που ανήκει σε μία από δύο δυνατές κλάσεις [14]. Η πρώτη ονομάζεται κλάση εξυπηρετητή (server) και η δεύτερη κλάση πελάτη (client). Γενικά σε τέτοια αντικείμενα τα links κατάστασης είναι άγνωστα για τα άλλα αντικείμενα του συστήματος και τα διαχειρίζεται μόνο η συνάρτηση του αντικειμένου στο οποίο ανήκουν. Επίσης, κάθε link εισόδου αντιστοιχίζεται σε ένα συγκεκριμένο τύπο ώστε να δέχεται μόνο tokens του τύπου αυτού. Ο actor, σε ένα αντικείμενο, αναπαριστά το ενεργό μέρος που μπορεί να είναι ένα σύνολο από συναρτήσεις (functions) ή διεργασίες (procedures).

2.2 Το μοντέλο IDFG

Αντίστοιχα περιγράφεται και η επικοινωνία του συστήματος με τον χρήστη, χρησιμοποιώντας το μοντέλο IDFG. Ενα IDFG είναι πάλι ένας διμερής κατευθυνόμενος γράφος, με τα ίδια χαρακτηριστικά, όπως στο μοντέλο EDFG. Κάθε actor έχει δύο τμήματα: το τμήμα συμπεριφοράς (behavioural part) και το τμήμα λειτουργίας (functional part). Στο πρώτο περιλαμβάνονται κανόνες (rules) που περιγράφουν πώς μετασχηματίζονται οι συνθήκες εισόδου του actor (περιγράφονται με την συνάρτηση PRE του actor) σε συνθήκες εξόδου (συνάρτηση $POST$ του actor), ενώ στο δεύτερο περιέχονται τμήματα κώδικα που υλοποιούν την συνάρτηση FUN του actor. Ακόμα, στη θέση της παραμέτρου $TYPE$ χρησιμοποιείται η παράμετρος $INHERITS$, για να απεικονίσει θέματα υλοποίησης (θα αναλυθεί στα επόμενα).

Υπάρχουν έξι τύποι links: user action, system action, object condition, goal, incommunication, outcommunication. Οι δύο πρώτοι χρησιμοποιούνται για να περιγράψουν τα γεγονότα (events) που συμβαίνουν στο σύστημα. Ενας βασικός περιορισμός είναι ότι μετάβαση κατάστασης έχουμε μόνο σαν αποτέλεσμα κάποιου γεγονότος. Ο τύπος object condition χρησιμοποιείται για να απεικονίσει την κατάσταση των αντικειμένων της οθόνης κάθε φορά. Ο τύπος goal χρησιμοποιείται για να περιγράψει το περιβάλλον (context) μέσα στο οποίο λαμβάνει χώρα το αντίστοιχο γεγονός. Οι δύο τελευταίοι τύποι απεικονίζουν την δυνατότητα επικοινωνίας ανάμεσα σε ανεξάρτητους γράφους που περιγράφουν το σύστημα.

Ετσι μπορούν να περιγραφούν όλα τα γεγονότα που λαμβάνουν χώρα στο σύστημα (είτε προέρχονται από τον χρήστη, είτε από άλλο τμήμα του συστήματος), το περιβάλλον μέσα στο οποίο αυτά συμβαίνουν (και κατ'επέκταση μπορεί να γίνουν υποθέσεις για τους στόχους του χρήστη) και τα ορατά αποτελέσματα που έχουν τα διάφορα γεγονότα. Πρόκειται λοιπόν για ένα state–based μοντέλο [3], το οποίο αναπαριστά καλά τις προθέσεις του χρήστη (user–centered)

[1]

Υπάρχουν δύο είδη actors: action actors και context actors. Οι πρώτοι αναπαριστούν τις ενέργειες που υποστηρίζονται κατευθείαν από το σύστημα που σχεδιάζεται (application commands), ενώ οι δεύτεροι αναπαριστούν τις εντολές που υποστηρίζονται από το user interface του συστήματος (user interface commands). Οι context actors σχηματίζονται με την σύνθεση action actors και context actors. Η σύνθεση είναι μια ιδιότητα του μοντέλου που μας επιτρέπει να ορίσουμε υπογράφους που αντιστοιχούν σε "στόχους" του χρήστη. Με βάση τους στόχους και τις ενέργειες του χρήστη, οι στόχοι αναλύονται σε "επιμέρους στόχους", μέχρι να φτάσουμε σε επίπεδο application command. Ετσι, ανάλογα με το είδος της ανάλυσης, διακρίνουμε τέσσερεις τύπους context actors:

- Ακολουθίας, όπου ένας στόχος αναλύεται σε επιμέρους στόχους, οι οποίοι πρέπει να επιτευχθούν ακολουθιακά
- Ενας–από–πολλούς, όπου ένας στόχος αναλύεται σε επιμέρους στόχους, από τους οποίους αρκεί η επιτυχία ενός
- Ολοι–ανεξάρτητα–από–σειρά, όπου όλοι οι επιμέρους στόχοι πρέπει να επιτευχθούν ανεξάρτητα όμως από σειρά
- Παράλληλα, όπου ενώ όλοι οι επιμέρους στόχοι πρέπει να επιτευχθούν, αυτό μπορεί να γίνει παράλληλα.

Αλλη μια ιδιότητα του μοντέλου είναι η κληρονομικότητα (inheritance) ανάμεσα σε τύπους (actors). Η ιδιότητα αυτή επιτρέπει στο σχεδιαστή να επαναχρησιμοποιεί actors που έχει ήδη ορίσει και αποθηκεύσει σε κάποια βιβλιοθήκη, ώστε το SEPDS να παράγει αυτόματα κομμάτια κώδικα. Η ιεραρχία κληρονομικότητας αναπαρίσταται με την παράμετρο $INHERITS$ ενός actor.

Κάθε στιγμή, ένα σύνολο από actors είναι έτοιμο προς εκτέλεση (ready–to–fire). Αυτοί οι actors περιέχουν tokens σε όλους τους links εκτός από τους event links. Ενας από αυτούς θα εκτελεστεί ανάλογα με το επόμενο event, το οποίο μπορεί να είναι είτε system είτε user action. Ετσι, μια μετάβαση κατάστασης γίνεται μόνο εξαιτίας κάποιου event.

2.3 Ανάπτυξη από πάνω προς τα κάτω

Οταν χρειάζεται να γίνει ανάλυση ενός αντικειμένου προς τα κάτω, να περιγραφούν δηλαδή με μεγαλύτερη λεπτομέρεια οι λειτουργίες του σε "χαμηλότερο επίπεδο", ένα πρωτότυπο μοντελοποιημένο χρησιμοποιώντας το SEPDS, αρχικά αναπαριστά το σύστημα σε ένα υψηλό, επομένως και γενικό επίπεδο αφαίρεσης. Τα αντικείμενα στο επίπεδο αυτό, περιέχουν σύνθετες δομές δεδομένων και υπολογισμών. Στη διαδικασία ανάλυσης τα αντικείμενα πρέπει να αναλυθούν σε πιο σύνθετους γράφους που τα περιγράφουν με μεγαλύτερη λεπτομέρεια. Η διδικασία αυτή πρέπει να επαναληφθεί μέχρι να επιτευχθεί ο επιθυμητός βαθμός λεπτομέρειας ή μέχρι όλοι οι actors να έχουν γίνει στοιχειώδεις γράφοι ροής δεδομένων (data flow graphs). Η διαδικασία ανάλυσης αποτελείται από δύο στάδια:

1. την ανάλυση του αντικειμένου σαν γράφος (object refinement)
2. την ανάλυση του αντικειμένου σε ότι αφορά τις λειτουργίες του (functional refinement)

Το πρώτο στάδιο απαιτεί καταρχήν την εσωτερική διάσπαση του τρέχοντος actor σύμφωνα με τις προδιαγραφές του (διατηρώντας όμως το interface του), έπειτα την πρόσθεση νέων actors στο γράφο, και την αντιστοίχιση των links κατάστασης σε περισσότερους από έναν actor αν χρειαστεί. Αυτό γίνεται αν πολλοί actors απαιτούν οπωσδήποτε το ίδιο link και διευκολύνεται σημαντικά με την αντιστοίχιση ονομάτων στα links. Το δεύτερο στάδιο περιλαμβάνει τη διάσπαση της συνάρτησης του actor όπως αυτή περιγράφεται από το πεδίο FUN σε συναρτήσεις ή διεργασίες που μπορούν να αντιστοιχιστούν στους νέους actors που προέκυψαν στο πρώτο στάδιο της διαδικασίας ανάλυσης.

2.4 Διαμοιρασμός του πρωτοτύπου

Ο σκοπός του διαμοιρασμού του πρωτοτύπου είναι η εύρεση του πιο κατάλληλου στοιχειώδους μεγέθους για τα αντικείμενα και έχει σαν αποτέλεσμα την αύξηση του βαθμού παραλληλίας στο σύστημα. Οι βασικές ιδέες για το διαμοιρασμό του πρωτοτύπου συνοψίζονται στα εξής:

- κάθε κανάλι εισόδου αντιστοιχίζεται σε απαιτήσεις του ίδιου τύπου
- κάθε τύπος απαίτησης ή γεγονός διεγείρει μία ακολουθία πυροδοτήσεων (Firing Sequence) που ξεκινά στο σχετικό link εισόδου και τερματίζεται σε κάποιο link εξόδου
- ένα αντικείμενο μπορεί να διαμοιραστεί σύμφωνα με τις διαθέσιμες ακολουθίες πυροδότησης και το βαθμό της εξάρτησης των δεδομένων όπως αυτή εκφράζεται από τα links κατάστασης

3 Αρχιτεκτονική του SEPDS

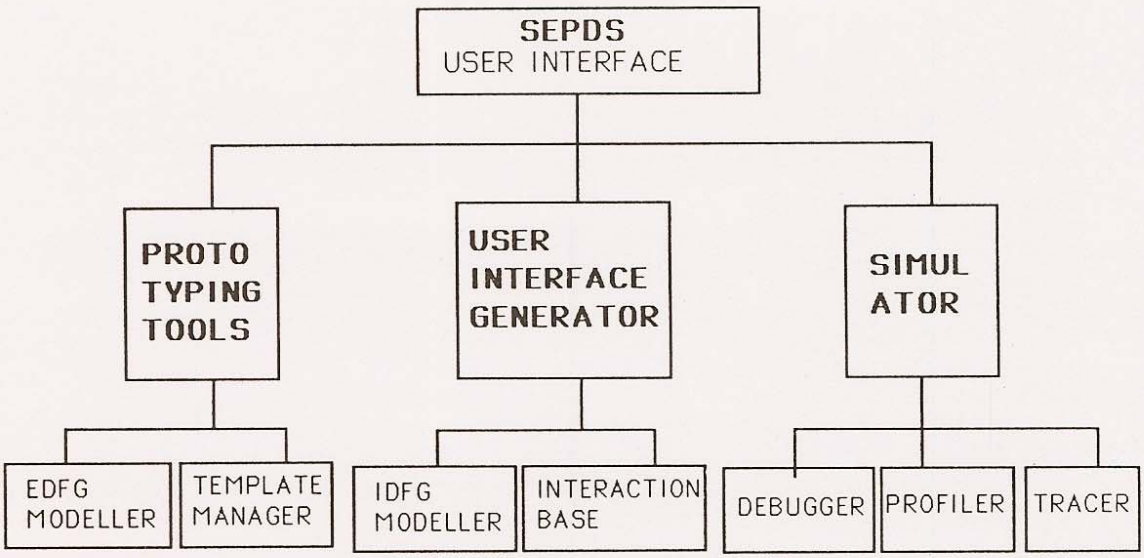
Το SEPDS σχεδιάστηκε για να υποστηρίζει την πρωτοτυποποίηση κατανεμημένων συστημάτων. Οι απαιτήσεις για τέτοια συστήματα είναι δύσκολο να καθοριστούν. Δύσκολο επίσης είναι και να συμπεράνουμε αν τα συστήματα αυτά είναι και εφικτά, εκτός και αν δημιουργήσουμε κάποιο εκτελέσιμο μοντέλο τους. Τα κατανεμημένα συστήματα δημιουργούν επιπρόσθετα προβλήματα στην έρευνα της υπολογιστικής δύναμης όλων των διαθέσιμων μονάδων επεξεργασίας. Για το λόγο αυτό ένα σύστημα που υποστηρίζει το χρήστη πρέπει να παρέχει δυνατότητες για τον καθορισμό της μεθόδου δέσμευσης πόρων, για τον έλεγχο του φόρτου στο σύστημα και για τον ορισμό του στοιχειώδους μεγέθους των αντικειμένων.

Με βάση τα παραπάνω το SEPDS αποτελείται από τρία βασικά υποσυστήματα (Σχήμα 1):

1. το User Interface Generator που παρέχει τα εργαλεία που επιτρέπουν τον σχεδιασμό του user interface της εφαρμογής,
2. τα εργαλεία πρωτοτυποποίησης για τις μετατροπές του πρωτοτύπου και την ανάλυση από πάνω προς τα κάτω των αντικειμένων του, και
3. τον εξομοιωτή (simulator) για την εκτέλεση, τον έλεγχο του πρωτοτύπου και τη μέτρηση ορισμένων παραμέτρων του.

Την πρόσβαση του σχεδιαστή σε όλα αυτά τα εργαλεία διαχειρίζεται το user interface του SEPDS, το οποίο του παρέχει πολλές δυνατοτητες, όπως εισαγωγή πληροφορίας που σχετίζεται

με τις απαιτήσεις του σχεδιασμού, έλεγχος των αποτελεσμάτων της εξομοίωσης, επιλογή των κομματιών του πρωτότυπου που τον ενδιαφέρουν και σχεδιασμό του user interface της εφαρμογής που έκτισε.



Σχήμα 1. Η αρχιτεκτονική του SEPDS

3.1 Το user interface generator υποσύστημα

Τα εργαλεία του υποσυστήματος αυτού επιτρέπουν στον σχεδιαστή να κατασκευάσει το user interface της εφαμογής χρησιμοποιώντας το IDFG μοντέλο. Ο IDFG modeler, πέρα από την υποστήριξη της εισαγωγής των προδιαγραφών επικοινωνίας (interaction specifications), το υποσύστημα αυτό υποστηρίζει την σύνδεση του IDFG γράφου με τον EDFG γράφο, ώστε να επιτυγχάνεται σωστή ολοκλήρωση της εφαρμογής με το user interface της.

Το user interface generator υποσύστημα περιλαμβάνει μια βάση γνώσης με οδηγίες σχεδιασμού διαλεκτικών εφαρμογών (interaction base). Ετσι, βοηθείται ο σχεδιαστής, και υποστηρίζεται μια ημι–αυτόματη διαδικασία κατασκευής user interface, με το σύστημα να προτείνει τον κατάλληλο σχεδιασμό, και τον σχεδιαστή να τον τελειοποιεί. Κατά τον σχεδιασμό, παρέχεται πλήρης υποστήριξη στον χρήστη (έλεγχος συνέπειας, επαναχρησιμοποίηση κλπ), όπως και κατά τον σχεδιασμό της λειτουργικότητας της εφαρμογής. Το συστημα που παράγεται, μπορεί επίσης να εισαχθεί στο υποσύστημα πρωτοτυποποίησης για την παραγωγή διαλογικού πρωτοτύπου, ή στο υποσύστημα εξομοίωσης γαι την παραγωγή μετρήσεων, όπως συμβαίνει με κάθε γράφο εφαρμογής που σχεδιάστηκε με το SEPDS.

3.2 Εργαλεία πρωτοτυποποίησης

Το υποσύστημα με τα εργαλεία πρωτοτυποποίησης αποτελείται από δύο βασικά τμήματα, τον EDFG modeler και τον Template Manager.

Ο EDFG modeler παρέχει μία σειρά από εργαλεία για τη δημιουργία και τη διατήρηση των EDFGs. Ο Graph Editor υποστηρίζει την με γραφικό τρόπο εισαγωγή του μοντέλου και παρέχει στο χρήστη πληροφορίες που τον βοηθούν κατά το διαμοιρασμό του πρωτοτύπου. Ο Top Down Modeler βοηθά το σχεδιαστή να διασπάσει έναν actor ή ένα ευρύτερο μέρος ενός EDFG σε απλούστερους actors και στη συνέχεια να τους συνδέσει μέσω στοιχειωδών actors. Ο Actor Base Manager συντηρεί και διαχειρίζεται μία βάση δεδομένων από διαθέσιμους actors που μπορούν να χρησιμοποιηθούν στη διαδικασία κατασκευής του EDFG για το πρωτότυπο. Για καθένα από αυτούς τους actors ο χρήστης πρέπει να παρέχει μία επιπρόσθετη πληροφορία, πέρα από την πεντάδα που ορίσαμε παραπάνω, για να είναι δυνατή η εξαγωγή από τη βάση του αναγκαίου actor. Τότε κατά τη διάρκεια του σχεδιασμού θα είναι διαθέσιμη μία λίστα με όλους τους actors που μπορεί να ζητήσει ο χρήστης μέσω του Graph Editor ώστε να συντομεύσει η διαδικασία της μοντελοποίησης.

Ο template manager περιέχει τα εργαλεία για τη διατήρηση του απαραίτητου software και το κτίσιμο του πρωτοτύπου. Το software περιέχεται σε ειδική βάση δεδομένων με τη μορφή ανεξάρτητων συναρτήσεων (templates) σε γλώσσα προγραμματισμού Ada, έτσι ώστε να είναι επαναχρησιμοποιήσιμο και να αντιστοιχίζεται σε συνιστώσες του EDFG. Εδώ το κύριο πρόβλημα είναι πως θα ορισθεί αυτή η συστηματική κατηγοριοποίηση των templates ώστε να επιτευχθεί ο επιθυμητός βαθμός επαναχρησιμοποίησης.

Ο Template Library Manager υποστηρίζει την οργάνωση των templates σε μία βάση δεδομένων σύμφωνα με κάποια προκαθορισμένη πολιτική κατηγοριοποίησης και ένα ορισμένο σύνολο κανόνων για την αρχικοποίηση γενικών παραμέτρων ή τη σύνθεση οντοτήτων από απλούστερες και διαθέσιμες. Το εργαλείο αυτό είναι ο αρχικός σχεδιασμός μιας βάσης γνώσης για ένα σύστημα που υποστηρίζει τη διαδικασία πρωτοτυποποίησης. Σκοπός είναι να διαχωριστεί η μοντελοποίηση του πρωτοτύπου από το πραγματικό κτίσιμο του εκτελέσιμου μέρους του. Το βασικό πλεονέκτημα είναι ότι οι καταχωρήσεις λειτουργιών σε actors μπορούν να γίνουν μόνο κατά τη διάρκεια του κτισίματος. Επομένως ο προγραμματιστής μπορεί να δημιουργήσει το μοντέλο ανεξάρτητα από τις προδιαγραφές και το σχεδιασμό των templates. Ο Prototype Builder αποτελείται από δύο μέρη. Το ένα μέρος αξιοποιείται για την επιλογή ενός template σύμφωνα με τα μοντέλα και την αρχικοποίησή του σύμφωνα με τις οδηγίες του χρήστη. Για το λόγο αυτό υπάρχουν τρεις κύριοι μηχανισμοί:

- Επιλογή του κατάλληλου template από την αντίστοιχη βάση με τη βοήθεια του Template Library Manager. Οι γενικές παράμετροι του template θα καθοριστούν στην διάρκεια της επιλογής του.

- Αν ένα template δεν υπάρχει ή δε μπορεί να κατασκευαστεί, ο αντίστοιχος actor θα πρέπει να διασπαστεί σε απλούστερους. Γενικά μία τέτοια διάσπαση πρέπει να γίνει όποτε ένα template δεν υπάρχει στη βιβλιοθήκη ή όποτε οι λειτουργίες που σχετίζονται με τον actor είναι εξαιρετικά πολύπλοκες ώστε να συμφέρει η διασπασή τους σε περισσότερες και απλούστερες.

- Υλοποίηση σε γλώσσα προγραμματισμού Ada. Ο προγραμματιστής υποχρεώνεται να γράψει μόνος του τον κώδικα για κάποιο template όταν αυτό δεν περιέχεται στη βάση των

templates και δεν υπάρχει τρόπος να εκφραστεί με τη βοήθεια άλλων.

Το δεύτερο μέρος του Prototype Builder χρησιμοποιείται για την κατασκευή stubs[3]. Επειδή ένα πρωτότυπο είναι ένα εκτελέσιμο μοντέλο που δεν απεικονίζει ολόκληρο το σύστημα, χρειάζεται να προστεθούν πολλά stubs. Έτσι εγγυάται η σωστή εκτέλεση του πρωτοτύπου και αντικαθιστώνται τμήματά του που δεν είναι ακόμα διαθέσιμα στο τρέχον επίπεδο της διαδικασίας μοντελοποίησης.

3.3 Το υποσύστημα εξομοίωσης

Σε πραγματικές υλοποιήσεις, σπάνια γεγονότα όπως η απώλεια της διάταξης σε ένα link είναι φυσικό να συμβαίνουν. Προκειμένου λοιπόν να διορθωθεί το πρωτότυπο πρέπει να γίνει έρευνα και διόρθωση των λαθών και αυτό είναι εξαιρετικά δύσκολο. Η ανάλυση της πιθανής συμπεριφοράς ενός πρωτοτύπου απαιτεί μία διαδικασία πειραματισμού, γι'αυτό και η εξομοίωση της λειτουργίας του πρωτοτύπου αποτελεί την καλύτερη λύση.

Η βασική απαίτηση από τον simulator είναι να μπορεί να αναλύει την απόδοση των κατανεμημένων συστημάτων. Θεωρούμε ότι υπάρχει μία ολοκληρωμένη εφαρμογή που έχει υποστεί τον απαραίτητο διαμοιρασμό και το αποτέλεσμα είναι ένας γράφος που αναπαριστά το σύστημα. Αν το κατανεμημένο πρωτότυπο τρέχει γρήγορα και αξιοποιεί στον επιθυμητό βαθμό τους διαθέσιμους επεξεργαστές, τότε η βελτίωση της παραλληλίας έχει επιτευχθεί. Παρ'όλα αυτά στις περισσότερες περιπτώσεις το πρωτότυπο δεν τρέχει όπως επιθυμούμε και δεν μπορεί να χρησιμοποιεί τους διαθέσιμους επεξεργαστές. Τότε πρέπει να ξεκινήσει ένα δύσκολο έργο που περιλαμβάνει τη μετατροπή των προγραμμάτων ή και τον επανασχεδιασμό το όλου συστήματος ώστε να επιτευχθεί η καλύτερη δυνατή αξιοποίηση του hardware.

Το υποσύστημα εξομοίωσης αποτελείται από τρία εργαλεία, τα debugger, tracer και profiler. Ο debugger επιτρέπει στο σχεδιαστή να επιτηρεί τη συμπεριφορά της υλοποίησης. Εχει δυνατότητες για την έναρξη της εκτέλεσης του πρωτότυπου και την απεικόνιση αποτελεσμάτων. Επειδή οι απαιτήσεις για τους tracer και debugger καλύπτονται από το πρόγραμμα Verdix Ada, προς το παρόν χρησιποιείται αυτή η εφαρμογή στο κομμάτι του simulator.

Ο profiler του SEPDS είναι ένα σχετικά απλό εργαλείο μέτρησης που και αυτό βασίζεται στα μοντέλα EDFG και IDFG. Με τη χρήση της πληροφορίας που περιέχεται στο γράφο του πρωτότυπου είναι δυνατό να εκτελεστεί κάθε actor ξεχωριστά χωρίς την μετατροπή του πρωτοτύπου ή τη σύνδεση του κώδικά του με το λειτουργικό σύστημα.

Ο profiler περιέχει δύο εφαρμογές. Ο πυρήνας (kernel) περιέχει εφαρμογές για την κίνηση και την αποστολή tokens ανάμεσα στα αντικείμενα, τον ορισμό, τη λήψη και την απεικόνιση της πληροφορίας που σχετίζεται actors, links και tokens, την εξέλιξη του χρόνου εκτέλεσης, την αποθήκευση πληροφοριών και τη ρύθμιση του πρωτοτύπου. Ο δρομολογητής των actors (που βρίσκεται στην εφαρμογή kernel) εκτελεί τους actors και καλεί τις απαραίτητες ρουτίνες για να κινεί και να στέλνει tokens από τα κανάλια εισόδου στα κανάλια εξόδου των actors. Μετά την εκτέλεση ενός actor η κατάλληλη πληροφορία με τις μετρήσεις που του αντιστοιχούν αποθηκεύεται. Υπάρχουν διάφορα είδη μετρήσεων που αφορούν στη διαδικασία εξομοίωσης και που πρέπει να γίνουν από την εφαρμογή profiling, όπως parallelism profile που δείχνει το βαθμό παραλληλίας στο τρέχον πρωτότυπο, actor profile που δείχνει τη συχνότητα εκτέλεσης των actors, synchronization profile για το χρόνο που δαπανήθηκε για το συγχρονισμό ανάμεσα στα αντικείμενα, και

[3]Stub είναι ένας μηδενικός γράφος, ίως με ενδεικτικές λειτουργίες, που χρησιμοποιείται για να υποκαταστήσει κάποιον γράφο

communication profile που δείχνει το χρόνο που καταναλώνεται για την επικοινωνία μέσω του μέσου επικοινωνίας.

Με τη βοήθεια των λειτουργιών του profiler εξασφαλίζεται η σωστή επικοινωνία του χρήστη και του simulator μέσω του user interface φυσικά. Για κάθε παράμετρο που μετρά ο profiler υπάρχει ξεχωριστή εφαρμογή για τη μετάφραση της σχετικής πληροφορίας και για την απεικόνισή της στο χρήστη μέσω του user interface.

Τα κυριώτερα πλεονεκτήματα του simulator στο SEPDS είναι [7]:

- δεν είναι απαραίτητο να προστεθεί επιπλέον κώδικας στο πρωτότυπο
- είναι εύκολο να ρυθμιστούν παράμετροι του πρωτοτύπου όπως είναι η καθυστέρηση επικοινωνίας, ο χρόνος εκτέλεσης για έναν actor, η πολιτική δέσμευσης πόρων, και σαν αποτέλεσμα ο φόρτος του συστήματος και ο αριθμός των διαθέσιμων επεξεργαστών
- ο simulator μπορεί να λειτουργήσει τόσο σε σειριακό όσο και σε κατανεμημένο σύστημα γιατί η φύση των μοντέλων προδιαγραφών δίνει τη δυνατότητα σε κάθε actor να εκτελείται σε διαφορετική μηχανή, φτάνει με κάποιο τρόπο τα tokens που απαιτούνται από τη συνάρτηση PRE να είναι διαθέσιμα στα links εισόδου
- ο simulator μπορεί να λειτουργήσει με διαφορετικά στοιχειώδη μεγέθη των τμημάτων του συστήματος και για διαφορετικά επίπεδα του πρωτοτύπου. Έτσι ο χρήστης μπορεί να ερευνά τα ενδιαφέροντα κομμάτια του πρωτοτύπου αντικαθιστώντας τα υπόλοιπα με stubs
- η εξομοίωση μπορεί να σταματήσει οποιαδήποτε στιγμή και έπειτα να συνεχιστεί. Η πληροφορία που κρατά ο simulator διατηρείται με τη μορφή δομών και μεταβλητών.

4 Συμπεράσματα

Στην εργασία αυτή παρουσιάστηκε το SEPDS, ένα περιβάλλον που υποστηρίζει τον σχεδιασμό και την πρωτοτυποποίηση κατανεμημένων εφαρμογών. Το SEPDS χρησιμοποιεί δύο data–flow graph μοντέλα για τον καθορισμό των προδιαγραφών λειτουργίας (EDFG) και επικοινωνίας με τον χρήστη (IDFG) της εφαρμογής. Ακόμα, το σύστημα παρέχει εργαλεία που υποστηρίζουν την γρήγορη πρωτοτυποποίηση και μέτρηση της αποδοτικότητας της εφαρμογής (μέσω εξομοίωσης), ώστε να μπορεί ο σχεδιαστής να διορθώνει τα λάθη του νωρίς στην διαδικασία ανάπτυξης.

Ο τρόπος χρήσης του συστήματος μπορεί να δειχθεί στην αντιμετώπιση της γενικευμένης έκδοσης του προβλήματος των αναγνωστών και γραφέων. Στο πρόβλημα αυτό, ένα παράλληλο σύστημα αποτελείται από N διεργασίες-πελάτες κατανεμημένες σε K κλάσεις. Οι διεργασίες της ίδιας κλάσης αντιμετωπίζονται ισοδύναμα. Για κάθε πόρο, κάποιος μπορεί να ορίσει τον αριθμό των διεργασιών που μπορούν παράλληλα να τον προσπελαύνουν. Διεργασίες διαφορετικών κλάσσεων έχουν αποκλειστική πρόσβαση σε ένα πόρο.

Μία διεργασία εξυπηρετητή που καλείται *Ελεγκτής πόρου*, δρομολογεί τις διεργασίες στο σύστημα σύμφωνα με τις παραπάνω απαιτήσεις. Η διεργασία αλληλεπιδρά με τον ελεκτή πόρου μέσου δύο ειδών απαιτήσεων: *Acq* (απαίτηση για δέσμευση του πόρου) και *Rel* (απαίτηση για απελευθέρωση του πόρου).

Ο στόχος είναι να σχεδιαστεί ένας παράλληλος Ελεγκτής πόρου. Μία μέτρηση του όλου συστήματος πρέπει να είναι διαθέσιμη στο τέλος, για να ερευνηθεί το στοιχειώδες μέγεθος των

αντικειμένων, η πολιτική δέσμευσης πόρων, η συμπεριφορά επικοινωνίας και συγχρονισμού. Το συγκεκριμένο μοντέλο παρουσιάζεται με λεπτομέρεια στο [7].

Αρκετά τμήματα του SEPDS (simulator, graph editor (για EDFG και IDFG γράφους), και τμήματα των Top Down modeller, Actor Base manager, profiler) έχουν ήδη υλοποιηθεί. Προς το παρόν τελειοποιείται η ανάπτυξη των εργαλείων πρωτοτυποποίησης (κυρίως των Top Down modeller και Actor Base manager), και κατασκευάζεται το υποσύστημα user interface generator. Στο μέλλον θα ασχοληθούμε με την υλοποίηση των εργαλείων του Template manager, και την δυνατότητα παραγωγής του κώδικα της εφαρμογής.

References

[1] J. Bonar and B. Liffick, *Communicating with high–level plans.* In Intelligent User Interfaces (J. Sullivan and S. Typer eds), ACM Press, 1991, pp 129-157.

[2] D. J. M. J. de Baar *et al.*, *Coupling application design and user interface design.* In proceedings of the CHI92 Conference: Striking a balance, May 3-7, 1992, Monterey, USA, pp 259-266.

[3] A. J. Dix and C. Runciman, *Abstract models of interactive systems.* In Proceedings of the British Computer Society Conference on People and Computers: Designing the Interface (P. Johnson and S. Cook eds), Cambridge University Press, 1985, pp 13-22.

[4] P. Henderson, editor, *Proceedings of the second SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments.* ACM SIGPLAN Notices, vol 22, January 1989.

[5] C. Jard, J. Monin and R. Groz, *Development of VEDA, a prototyping tool for distributed algorithms.* IEEE Trans. Softw. Eng., SE-14(3), March 1988.

[6] K. Kavi, B. Buckles and V. Bhat, *A formal definition of data–flow graph models.* IEEE Trans. Comp., C-35(11), November 1986.

[7] A. Levy, H. Corporaal and J. van Katwijk, *Design of a parallel resource controller using EDFG.* In Parallel Computing 89, Amsterdam, The Netherlands, August 1989.

[8] A. Levy, J. van Katwijk, G. Pavlides and F. Tolsma, *SEPDS: A Support Environment for Prototyping Distributed Systems.* In Proceedings of the First International on System Integration, New Jersey, USA, April 1990.

[9] T. Lewis, F. Handloser, S. Bose and S. Yang, *Prototyping from standard user interface management systems.* IEEE Computer, 22(5), May 1989.

[10] Luqi, *Software evolution through rapid prototyping.* IEEE Computer, 22(5), May 1989.

[11] Luqi and M. Ketabchi, *A computer–aided prototyping system.* IEEE Software, 5(2), March 1988.

[12] S. Papadimitriou, G. Pavlides and A. Kameas, *A new compression algorithm for data–flow graph models of distributed real–time tasks.* Technical Report TR92.04.9, Computer Technology Institute, Patras, Greece.

[13] S. Papadimitriou, A. Kameas and G. Pavlides, *The derivation of CTMC models for distributed real–time tasks from their data–flow representation.* Technical Report TR92.09.14, Computer Technology Institute, Patras, Greece.

[14] S. Papadimitriou, A. Kameas, P. Fitsilis and G. Pavlides, *A new compression technique for tools that use data–flow graphs to modeldistributed real–time applications.* In proceedings of the fifth International Conference on Software Engineering and its Applications, Toulouse, France, December 1992.

[15] T. Sterlich, *The software life–cycle support environment (SLCSE): A computer–based framework for developing software systems.* In proceedings of the second SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments. ACM SIGPLAN Notices, vol 24, February 1989.