Κεφάλαιο 1ο

Εισαγωγή στη γλώσσα UML

1.1 Προσθέτοντας μια νέα μέθοδο

Στις πρώτες εποχές των υπολογιστών, οι προγραμματιστές συνήθιζαν να περιορίζονται στην ανάλυση σε βάθος των προβλημάτων που αντιμετώπιζαν. Συχνά «έγραφαν» προγράμματα από την αρχή, δημιουργώντας κώδικα καθώς προχωρούσαν. Αν και αυτό προσέθετε μια αύρα ρομαντισμού και τόλμης στη διαδικασία, αποδείχθηκε πως είναι ακατάλληλο για τον σημερινό απαιτητικό επιχειρηματικό κόσμο.

Σήμερα, είναι απαραίτητο ένα καλά δομημένο διάγραμμα. Ο πελάτης πρέπει να καταλάβει το τι πρόκειται να κάνει μια ομάδα ανάπτυξης και να είναι σε θέση να προτείνει αλλαγές στην περίπτωση που η ομάδα δεν έχει καλύψει πλήρως τις σημερινές και μελλοντικές του ανάγκες (ή αν ο πελάτης αλλάξει τις προδιαγραφές του στην πορεία).

Επίσης, η ανάπτυξη είναι μια ομαδική προσπάθεια οπότε κάθε μέλος της ομάδας πρέπει να γνωρίζει σε ποιο σημείο του έργου «ταιριάζει» αυτό που πρόκειται να κάνει.

Καθώς οι ανάγκες των τελικών χρηστών γίνονται ολοένα και πιο πολύπλοκες, τα υπολογιστικά συστήματα γίνονται και αυτά, αναπόφευκτα, πιο σύνθετα. Συχνά περιλαμβάνουν διάφορα κομμάτια από λογισμικό και υλικό, δικτυωμένα σε μεγάλες αποστάσεις, συνδεδεμένα σε βάσεις δεδομένων που περιέχουν πληθώρα πληροφοριών. Αν θέλουμε να δημιουργήσουμε επιτυχημένα συστήματα πρέπει να αντιμετωπίσουμε επιτυχώς αυτή την πολυπλοκότητα.

Το ζητούμενο είναι να οργανωθεί η διαδικασία σχεδιασμού έτσι ώστε οι αναλυτές, οι πελάτες, οι προγραμματιστές και γενικά όλοι όσοι εμπλέκονται στην ανάπτυξη του συστήματος να μπορούν να κατανοήσουν και να συμφωνήσουν σε μια κοινή περιγραφή του συστήματος που αναπτύσσεται. Η UML (Unified Modeling Language) παρέχει αυτή τη δυνατότητα.

Όπως δε θα χτίζαμε μια πολύπλοκη κατασκευή όπως ένα κτίριο χωρίς πρώτα να κατασκευάσουμε ένα αναλυτικό διάγραμμα, έτσι δεν κατασκευάζουμε ένα πολύπλοκο σύστημα για να τοποθετηθεί σε εκείνο το κτίριο χωρίς πρώτα να συνθέσουμε ένα αναλυτικό πλάνο. Το πλάνο πρέπει να είναι τέτοιο ώστε να μπορούμε να το παρουσιάσουμε στον πελάτη, όπως ακριβώς ένας αρχιτέκτονας δείχνει τα σχέδια σε εκείνον που πληρώνει για την κατασκευή του κτιρίου. Αυτό το σχεδιαστικό πλάνο πρέπει να απορρέει από μια προσεκτική ανάλυση των αναγκών του πελάτη.

Οι χρονικοί περιορισμοί που τίθενται κατά την ανάπτυξη ενός συστήματος αποτελούν ένα άλλο γνώρισμα της ανάπτυξης συστημάτων. Όταν υπάρχουν χρονοδιαγράμματα και προθεσμίες που πρέπει να τηρηθούν, ένα σαφές πλάνο ανάπτυξης του συστήματος αποτελεί επιτακτική ανάγκη.

Οι εταιρικές συγχωνεύσεις είναι ένας ακόμη τομέας της μοντέρνας ζωής που απαιτεί «στιβαρό» σχεδιασμό: Όταν μια εταιρεία συγχωνεύεται με μια άλλη, η νέα οργάνωση μπορεί να αλλάξει σημαντικές πτυχές ενός εν ενεργεία αναπτυξιακού έργου (το εργαλείο εφαρμογής, η γλώσσα προγραμματισμού, κ.α). Ένα καλά δομημένο διάγραμμα έργου θα διευκολύνει την αλλαγή. Εάν ο σχεδιασμός είναι σωστός, μια αλλαγή στην εφαρμογή μπορεί να ενσωματωθεί ομαλά.

Η ανάγκη για σωστό σχεδιασμό επέφερε με τη σειρά της μια ανάγκη για σχεδιαστικές σημειώσεις που οι αναλυτές, οι προγραμματιστές και οι πελάτες θα αποδεχθούν ως πρότυπα - ακριβώς όπως οι σημειώσεις στα σχηματικά διαγράμματα αποτελούν πρότυπα για τους ηλεκτρονικούς μηχανικούς και οι σημειώσεις στα διαγράμματα Feynman αποτελούν πρότυπα για τους φυσικούς.

1.2 Πως αναπτύχθηκε η UML

Η UML είναι πνευματική δημιουργία των Grady Booch, James Rumbaugh και Ivar Jacobson. Γνωστοί και ως «The Three Amigos», εργάστηκαν σε διαφορετικούς οργανισμούς μέσα στη δεκαετία του ’80 και στις αρχές του ’90, ο καθένας αναπτύσσοντας τη δική του μεθοδολογία για την αντικειμενοστρεφή ανάλυση και σχεδιασμό. Οι μεθοδολογίες τους ξεπέρασαν αυτές των περισσότερων ανταγωνιστών τους. Στα μέσα της δεκαετίας του ’90, άρχισαν να δανείζονται ιδέες ο ένας από τον άλλον, οπότε αποφάσισαν να εξελίξουν τη δουλειά τους από κοινού.

Το 1994 ο Rumbaugh έγινε μέλος της Rational Software Corporation, όπου ο Booch εργαζόταν ήδη. Ο Jacobson προσχώρησε ένα χρόνο μετά.

Πρώιμες εκδόσεις της UML άρχισαν να κυκλοφορούν στη διαδικασία λογισμικού, πράγμα που επέφερε σημαντικές αλλαγές. Επειδή πολλοί οργανισμοί ένιωσαν πως η UML θα υπηρετούσε τους στρατηγικούς τους στόχους, μια κοινοπραξία άρχισε να ανθίζει. Τα μέλη συμπεριελάμβαναν τις DEC, Hewlett-Packard, Intellicorp, Microsoft, Oracle, Texas Instruments, Rational, και άλλες εταιρείες. Το 1997 η κοινοπραξία παρήγαγε την έκδοση 1.0 της UML και την κατέθεσε στο Object Management Group (OMG) ανταποκρινόμενο στην επιθυμία του OMG για μια πρότυπη γλώσσα μοντελοποίησης.

Η κοινοπραξία επεκτάθηκε, ανέπτυξε την έκδοση 1.1 και την κατέθεσε στον OMG που την υιοθέτησε στο τέλος του 1997. Ο OMG ανέλαβε την συντήρηση της UML και ανέπτυξε δυο ακόμα εκδόσεις το 1998. Η UML αποτελεί ένα de-facto πρότυπο στη βιομηχανία λογισμικού και συνεχίζει να εξελίσσεται. Οι εκδόσεις 1.3, 1.4 και 1.5 έχουν ήδη υλοποιηθεί και ο OMG πρόσφατα ανέπτυξε και την έκδοση 2.0. Οι προηγούμενες εκδόσεις 1.x έχουν αποτελέσει τη βάση των περισσότερων μοντέλων και βιβλίων μοντέλων.

1.3 Κατανοώντας την UML

Η UML δεν είναι τίποτα άλλο από μια γλώσσα. Δεν είναι ένας τρόπος για να σχεδιάσεις ένα σύστημα αλλά ένας τρόπος για να μοντελοποιήσεις ένα σύστημα. Για να χρησιμοποιήσεις την UML χρειάζεται να εφαρμοστεί μια **μέθοδος**. Υπάρχουν πολλές μέθοδοι που έχουν δημιουργηθεί γι' αυτό το λόγο. Στα πλαίσια του συγκεκριμένου βιβλίου θα ασχοληθούμε με τη μέθοδο GRAPPLE η οποία παρουσιάζεται αναλυτικά στο Κεφάλαιο 14.

Η UML είναι μια σημειογραφική και σημειολογική γλώσσα που μπορεί να εφαρμοστεί σε κάθε μορφής ανάπτυξη λογισμικού και όχι μόνο. Χρησιμοποιεί διάφορους τύπους σχεδιαγραμμάτων που μπορούν να χρησιμοποιηθούν για τη μοντελοποίηση αντικειμενοστρεφών συστημάτων λογισμικού.

Η UML αποτελείται από έναν αριθμό γραφικών στοιχείων που συνδυάζονται για να σχηματίσουν διαγράμματα. Επειδή η UML είναι γλώσσα, έχει κανόνες για να συνδυάζει αυτά τα στοιχεία. Στα επόμενα κεφάλαια δεν θα εστιάσουμε στα στοιχεία και στους κανόνες, αλλά θα μελετήσουμε κατευθείαν τα διαγράμματα γιατί αυτά είναι που χρησιμοποιούμε για να κάνουμε την ανάλυση των συστημάτων.

Ο σκοπός αυτών των διαγραμμάτων είναι να παρουσιάσουν διάφορες όψεις του συστήματος. Αυτό το σύνολο των πολλαπλών όψεων ονομάζεται **μοντέλο**. Ένα μοντέλο συστήματος UML είναι κάτι σαν ένα μοντέλο ενός κτιρίου μαζί με την άποψη ενός καλλιτέχνη για το κτίριο. Είναι σημαντικό να σημειώσουμε

πως ένα μοντέλο UML περιγράφει τι πρέπει να κάνει ένα σύστημα, όχι όμως και το πώς θα υλοποιηθεί.

1.4 Αναγνωρίζοντας τα επί μέρους τμήματα της UML

1.4.1 Δομικά διαγράμματα

1.4.1.1 Διαγράμματα αντικειμένων και κλάσεων

Τα **διαγράμματα κλάσεων (class diagrams)** χρησιμοποιούνται για να δείξουν τα διαφορετικά τμήματα του συστήματος, τις σχέσεις μεταξύ των τμημάτων και σε ποια υποσυστήματα ανήκουν. Τα διαγράμματα κλάσεων περιλαμβάνουν λειτουργίες, χαρακτηριστικά καθώς και πολλούς τύπους ρόλων και συσχετίσεων.

Ένα **διάγραμμα αντικειμένων (object diagram)** είναι σε πολλά σημεία όμοιο με ένα διάγραμμα κλάσεων με την διαφορά ότι στην θέση των κλάσεων έχουμε αντικείμενα που είναι στιγμιότυπα των κλάσεων. Αυτά τα διαγράμματα έχουν να κάνουν συνήθως με σχεδιασμό με την χρήση παραδειγμάτων. Με άλλα λόγια, τα αντικείμενα έχουν να κάνουν με πιο συγκεκριμένα θέματα σε αντίθεση με τις κλάσεις που είναι πιο γενικές.

Ως βάσεις για κάθε αντικειμενοστρεφές σύστημα, οι κλάσεις και τα αντικείμενα παρέχουν εύκολη διαχείριση των πληροφοριών τους. Οι κλάσεις χρησιμοποιούνται για την μοντελοποίηση στα αρχικά στάδια της φάσης επεξεργασίας, αλλά και για τη δημιουργία περιπλοκών τμημάτων του συστήματος σε μεταγενέστερα στάδια της ίδιας φάσης ανάπτυξης.

1.4.1.2 Διαγράμματα ανάπτυξης και συστατικών

Ένα **διάγραμμα ανάπτυξης (deployment diagram)** δείχνει που θα καταλήξουν τα συστατικά του συστήματος αφού εγκατασταθούν στο σύστημα και πως θα αλληλεπιδράσουν με αυτό, ενώ ένα **διάγραμμα συστατικών (component diagram)** δείχνει πώς τα συστατικά ενός συστήματος αλληλεπιδρούν μεταξύ τους. Ένα τέτοιο διάγραμμα παρουσιάζει τις συσχετίσεις μεταξύ βασικών αρχείων και κλάσεων καθώς και των συστατικών στα οποία ανήκουν. Στην έκδοση 2.0 της UML (βλέπε Κεφάλαιο 12) τα διαγράμματα συστατικών έχουν ενταχθεί στην κατηγορία των βασικών διαγραμμάτων, γεγονός που δείχνει πόσο σημαντικά θεωρούνται στην μοντελοποίηση σήμερα.

1.4.2 Διαγράμματα συμπεριφοράς

1.4.2.1 Διαγράμματα περιπτώσεων χρήσης

Τα **διαγράμματα περιπτώσεων χρήσης (use case diagrams)** περιέχουν ρόλους και τις σχέσεις μεταξύ αυτών των ρόλων και αποτελούν την αρχή της φά-

σης της ανάλυσης για το σχεδιασμό ενός συστήματος. Βασίζονται στην αρχική επινόηση του Ivar Jacobson, σύμφωνα με την οποία οι σχέσεις είναι η βάση των διαγραμμάτων περιπτώσεων χρήσης. Τα διαγράμματα περιπτώσεων χρήσης ενώνονται με τις σχέσεις και καταλήγουν στους ρόλους, με σκοπό να δείξουν τη συνολική δομή και διαθεσιμότητα του συστήματος στους μη ειδικούς αναγνώστες του μοντέλου καθώς και στους χρήστες.

1.4.2.2 Διαγράμματα αλληλεπίδρασης

Τα **διαγράμματα αλληλεπίδρασης (interaction diagrams)** εστιάζουν στις λεπτομέρειες που αφορούν στην αλληλεπίδραση μέσω μηνυμάτων των συστατικών στοιχείων του συστήματος. Σε αυτή την κατηγορία υπάγονται τα διαγράμματα ακολουθίας και τα διαγράμματα συνεργασίας.

Διαγράμματα ακολουθίας

Τα **διαγράμματα ακολουθίας (sequence diagrams)** χρησιμοποιούνται για να δείξουν την επίδραση μεταξύ των ρόλων και των αντικειμένων του συστήματος. Μηνύματα στέλνονται από τους ρόλους στα αντικείμενα, από αντικείμενα σε αντικείμενα και από αντικείμενα πάλι πίσω στους ρόλους έτσι ώστε να φαίνεται η ροή του ελέγχου μέσα σε ένα σύστημα. Τα διαγράμματα ακολουθίας επαληθεύουν τις περιπτώσεις χρήσης με το να δείχνουν με ποιο τρόπο λειτουργεί η κάθε περίπτωση χρήσης στο σύστημα.

Διαγράμματα συνεργασίας

Τα **διαγράμματα συνεργασίας (collaboration diagrams)** φέρνουν τα διαγράμματα κλάσης στο επόμενο επίπεδο. Περιγράφουν τις επιδράσεις και τις συσχετίσεις μεταξύ των αντικειμένων που δημιουργηθήκαν σε προηγούμενες φάσεις τις μοντελοποίησης του συστήματος. Αυτά τα διαγράμματα μπορούν να χρησιμοποιηθούν και για να μοντελοποιήσουν και μηνύματα μεταξύ διαφορετικών αντικειμένων.

1.4.2.3 Διαγράμματα δραστηριότητας

Τα **διαγράμματα δραστηριότητας (activity diagrams)** χρησιμοποιούνται για να αναλύσουμε τη συμπεριφορά μέσα σε περίπλοκες σχέσεις και να δείξουμε την αλληλεπίδραση με άλλες σχέσεις. Έχουν πολλές ομοιότητες με τα διαγράμματα αλληλεπίδρασης, κυρίως λόγω του ότι αναπαριστούν την ροή των πληροφοριών.

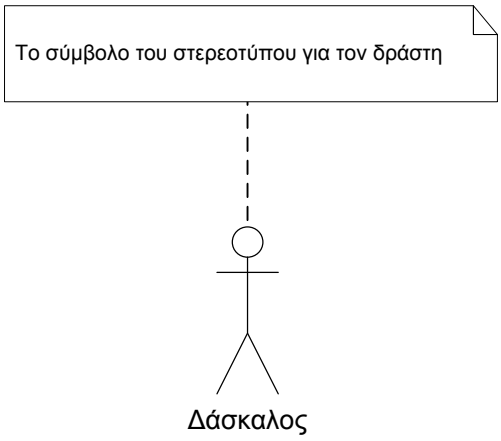
1.4.2.4 Διαγράμματα κατάστασης

Ένα **διάγραμμα κατάστασης (statechart diagram)** χρησιμοποιείται για να μοντελοποιήσει τη συμπεριφορά των υποσυστημάτων, τις σχέσεις των κλάσεων και της διασύνδεσης του συστήματος με τον χρήστη. Τα διαγράμματα καταστάσεων είναι μια εξαίρετη οπτική απεικόνιση της ροής μιας εφαρμογής.

1.5 Τα στερεότυπα στη UML

Τα **στερεότυπα (stereotypes)** αποτελούν ένα ιδιαίτερο χαρακτηριστικό της UML και *ορίζουν μια συγκεκριμένη χρήση ή σκοπό που μπορεί να εφαρμοστεί σχεδόν σε όλα τα διαγράμματα της UML*. Τα στερεότυπα μπορούν να χρησιμοποιηθούν για να αλλάξουν το νόημα ενός στοιχείου σε ένα διάγραμμα και περιγράφουν το ρόλο του συγκεκριμένου στοιχείου στο μοντέλο του συστήματος.

Σε ορισμένες περιπτώσεις, ένα στερεότυπο διαθέτει και μια σχετική εικόνα, όπως στο παράδειγμα του σχήματος 1-1 όπου το ανθρωπάκι είναι η εικόνα που σχετίζεται με το στερεότυπο δράστης της UML (περισσότερα για τους δράστες θα μάθουμε στο Κεφάλαιο 5). Στο συγκεκριμένο σχήμα, ο *Δάσκαλος* αντιπροσωπεύεται με τον ρόλο ενός δράστη επειδή χρησιμοποιείται η εικόνα που σχετίζεται με το συγκεκριμένο στερεότυπο (του δράστη).



Σχήμα 1-1: *Η εικόνα για το στερεότυπο του δράστη*

Σχήμα 1-2: *Ορισμός στερεοτύπου χωρίς χρήση εικόνας*

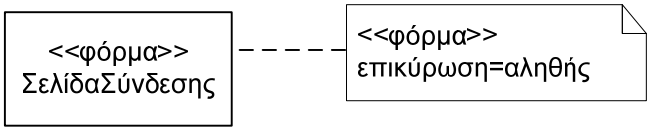
Για τα στερεότυπα δεν χρησιμοποιούμε πάντα εικόνες για να τα αναπαραστήσουμε. Σε αρκετές περιπτώσεις οι εικόνες των στερεοτύπων πιάνουν αρκετό χώρο και κάνουν τα διαγράμματα δυσνόητα. Σε αυτές όπως περιπτώσεις για να αναφερθούμε στο στερεότυπο χρησιμοποιούμε τον ακόλουθο συμβολισμό *<<όνομα στερεοτύπου>>* όπως φαίνεται στο σχήμα 1-2. Εδώ, ο *Δάσκαλος* εξακολουθεί να είναι δράστης, αλλά το στερεότυπο ορίζεται χωρίς τη χρήση εικόνας.

Δεν υπάρχει περιορισμός στο πλήθος των στερεοτύπων που μπορούμε να συσχετίσουμε με ένα συγκεκριμένο στοιχείο του διαγράμματός μας. Σε ορισμένες περιπτώσεις, μπορεί να χρειάζεται να ορίσετε περισσότερα από ένα στερεότυπα, όπως φαίνεται στο σχήμα 1-3.

Σχήμα 1-3: *Ορισμός πολλαπλών στερεοτύπων*

Τα στερεότυπα μπορούν να περιέχουν επιπλέον πληροφορία που σχετίζεται με ένα συγκεκριμένο στοιχείο. Αυτή η επιπλέον πληροφορία προσδιορίζεται μέσω ενός ειδικού χαρακτηριστικού της UML που ονομάζεται **προσαρτημένες τιμές (tagged values)**.

Οι **προσαρτημένες τιμές** σχετίζονται πάντα με ένα στερεότυπο. Για παράδειγμα, θεωρήστε ότι έχετε στο σύστημά σας ένα στοιχείο που αναπαριστά τη σελίδα σύνδεσης (login page) σε ένα δικτυακό τόπο, η οποία αναπαρίσταται μέσω του στερεοτύπου <<φόρμα>>. Το στερεότυπο <<φόρμα>> πρέπει να γνωρίζει αν θα πρέπει να επικυρώσει ή όχι τα περιεχόμενα της φόρμας. Η απόφαση για την επικύρωση ορίζεται σαν προσαρτημένη τιμή του στερεοτύπου <<φόρμα>> επειδή σχετίζεται με το στερεότυπο που εφαρμόζεται σε ένα στοιχείο, και όχι με το ίδιο το στοιχείο.



Σχήμα 1-4: *Χρήση προσαρτημένων τιμών*

Μία **προσαρτημένη τιμή** σχεδιάζεται σε ένα διάγραμμα με το ίδιο τρόπο που σχεδιάζονται και οι σημειώσεις (περισσότερα για τις σημειώσεις θα δούμε στην ενότητα 3.6), αλλά στο σύμβολο της σημείωσης περιέχεται το όνομα του στερεοτύπου και η προσαρτημένη τιμή που σχετίζεται με το στερεότυπο. Το σύμβολο της προσαρτημένης τιμής συνδέεται με το στοιχείο που περιέχει το

στερεότυπο χρησιμοποιώντας μια διακεκομμένη γραμμή, όπως φαίνεται στο σχήμα 1-4.

Η UML διαθέτει ένα σύνολο από προκαθορισμένα στερεότυπα τα οποία μπορούν να χρησιμοποιηθούν στα διαγράμματά της. Αρκετά από αυτά θα τα αναλύσουμε στα επόμενα κεφάλαια. Παράλληλα, δίνει τη δυνατότητα στους σχεδιαστές να ορίσουν και δικά τους στερεότυπα προκειμένου τα μοντέλα του συστήματος που αναπτύσσουν να περιγράφουν το πραγματικό σύστημα με μεγαλύτερη πιστότητα. Τέλος, τα στερεότυπα μπορούν να χρησιμοποιηθούν και για την επέκταση της UML προκειμένου να χρησιμοποιηθεί σε συγκεκριμένες κατηγορίες εφαρμογών μέσω του ορισμού συγκεκριμένων προφίλ της γλώσσας. Η αναφορά στον ορισμό εξειδικευμένων προφίλ της γλώσσας, αν και καλύπτεται εν μέρει στο Κεφάλαιο 12, δεν αποτελεί στόχο του συγκεκριμένου βιβλίου.

1.6 Ο σκοπός των διαγραμμάτων

Τα διαγράμματα της UML καθιστούν εφικτή την εξέταση ενός συστήματος από διάφορες οπτικές γωνίες. Είναι σημαντικό να σημειώσουμε ότι δεν είναι απαραίτητο να εμφανίζονται όλα τα διαγράμματα σε κάθε μοντέλο UML. Τα περισσότερα μοντέλα UML στην πραγματικότητα περιέχουν ένα υποσύνολο των διαγραμμάτων που προαναφέραμε.

Η σημαντικότητα του να παρουσιαστούν διάφορες όψεις του συστήματος έγκειται στο εξής: τυπικά, ένα σύστημα έχει έναν αριθμό από διαφορετικά άτομα που έχουν διαφορετικά ενδιαφέροντα για διαφορετικά μέρη του συστήματος. Αν για παράδειγμα σχεδιάζουμε τη μηχανή ενός αυτοκινήτου έχουμε τη μία όψη του συστήματος, εάν γράφουμε το εγχειρίδιο χρήσης της συγκεκριμένης μηχανής έχουμε μια άλλη όψη του συστήματος. Και τέλος, εάν σχεδιάζουμε το αυτοκίνητο στο σύνολό του βλέπουμε το σύστημα από μια τελείως διαφορετική σκοπιά σε σχέση με κάποιον που απλά θα χρησιμοποιήσει το αυτοκίνητο.

Ο σχεδιασμός ενός συστήματος αφορά όλες τις πιθανές όψεις και κάθε διάγραμμα της UML προσπαθεί να δώσει έναν τρόπο για να περιγραφεί μια συγκεκριμένη όψη του. Ο τελικός σκοπός είναι να επικοινωνήσουν καθαρά μεταξύ τους όλοι οι ενδιαφερόμενοι του συστήματος.