

Περιεχόμενα

1.	Εισαγωγή	2
2.	Κλάσεις	3
3.	Σχέσεις.....	3
3.1	Εξάρτηση	4
3.2	Γενίκευση.....	4
3.3	Σύνδεση.....	4
3.4	Υλοποίηση.....	5
3.5	Είδη εξαρτήσεων.....	5
4.	Διαγράμματα	6
4.1	Διαγράμματα περιπτώσεων χρήσης	6
4.2	Διάγραμμα κλάσεων	7
4.3	Διάγραμμα αντικειμένων.....	8
4.4	Διάγραμμα καταστάσεων	9
4.5	Διάγραμμα δραστηριοτήτων.....	9
4.6	Διάγραμμα ακολουθίας.....	10
4.7	Διάγραμμα συνεργασίας	11
4.8	Διάγραμμα εξαρτημάτων	12
4.9	Διάγραμμα ανάπτυξης	13
4.10	Χρήση διαγραμμάτων.....	13
5.	Βιβλιογραφία	14
6.	Ασκήσεις	15

Ανάλυση και σχεδίαση με UML

Διομήδης Σπινέλλης

Τμήμα Διοικητικής Επιστήμης και Τεχνολογίας

Οικονομικό Πανεπιστήμιο Αθηνών

dds@aueb.gr

1. Εισαγωγή

Η ενοποιημένη γλώσσα σχεδιασμού (*unified modeling language*) (UML) είναι μια γραφική γλώσσα για την οπτική παράσταση, τη διαμόρφωση προδιαγραφών και την τεκμηρίωση συστημάτων που βασίζονται σε λογισμικό. Η UML στοχεύει στο σχεδιασμό αντικειμενοστρεφών συστημάτων. Το σχέδιο είναι μια απλοποιημένη παράσταση της πραγματικότητας.

Σχεδιάζουμε για να μπορέσουμε να καταλάβουμε το σύστημα που αναπτύσσουμε. Έτσι δημιουργώντας ένα σχέδιο επιτυγχάνουμε τέσσερις στόχους:

1. παριστάνουμε οπτικά το σύστημα που έχουμε ή θέλουμε να κατασκευάσουμε,
2. προσδιορίζουμε τη δομή και τη συμπεριφορά του συστήματος,
3. δημιουργούμε ένα πρότυπο για να βασίσουμε την κατασκευή του συστήματος,
4. τεκμηριώνουμε τις αποφάσεις που λάβαμε.

Σε όλους τους τεχνολογικούς τομείς ο σχεδιασμός βασίζεται σε τέσσερις βασικές αρχές:

1. η επιλογή του είδους του σχεδίου έχει επίπτωση στον τρόπο και την μορφή επίλυσης του προβλήματος,
2. όλα τα σχέδια εκφράζονται σε διαφορετικές βαθμίδες ακρίβειας,
3. τα καλύτερα σχέδια σχετίζονται με την πραγματικότητα,
4. ένα είδος σχεδίων δεν είναι ποτέ αρκετό.

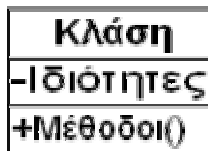
Η UML περιλαμβάνει τρία βασικά στοιχεία:

1. Οντότητες
2. Σχέσεις
3. Διαγράμματα

Η UML είναι μια πλήρης και πλούσια γλώσσα με εξαιρετικά ευρύ πεδίο εφαρμογής. Στο μάθημα αυτό θα εξετάσουμε εξαιρετικά συνοπτικά τον τρόπο παράστασης ορισμένων αντικειμενοστρεφών δομών σε UML.

2. Κλάσεις

- Μια κλάση σχεδιάζεται ως ένα ορθογώνιο με τρία τμήματα όπως στο παρακάτω σχήμα:

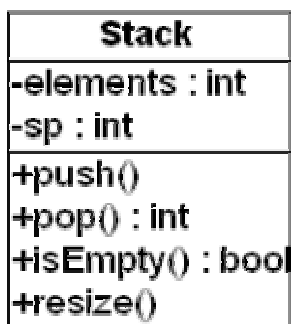


- Τα ονόματα των ιδιοτήτων και μεθόδων της κλάσης μπορούν να έχουν ένα πρόθεμα ανάλογα με την ορατότητά τους:

+
public

protected
-
private

- Οι μεταβλητές και οι συναρτήσεις που αναφέρονται στην κλάση και όχι σε αντικείμενά της (static στη C++) σχεδιάζονται υπογραμμισμένες.
- Οι αφηρημένες κλάσεις έχουν το όνομά τους γραμμένο με πλάγια στοιχεία.
- Το σχήμα που ακολουθεί αποτελεί σχέδιο για μια απλή κλάση στοίβας ακεραίων:



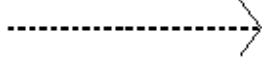
3. Σχέσεις

Στη UML ορίζονται τέσσερις βασικές σχέσεις:

- εξάρτηση (*dependency*)
- γενίκευση (*generalisation*)
- σύνδεση (*association*)
- υλοποίηση (*realisation*)

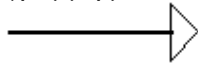
3.1 Εξάρτηση

Η εξάρτηση δηλώνει πως μια αλλαγή σε μιαν οντότητα θα επηρεάσει μιαν άλλη αλλά όχι απαραίτητα και το αντίστροφο. Παριστάνεται με μια διακεκομμένη γραμμή με ανοιχτό βέλος που δείχνει προς την οντότητα που υπάρχει εξάρτηση:



3.2 Γενίκευση

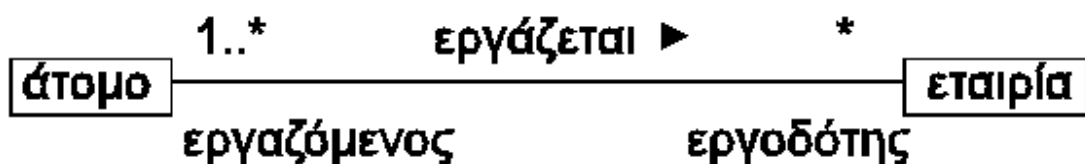
Η γενίκευση δηλώνει μια σχέση ανάμεσα σε κάτι γενικό (τη βασική κλάση ή αλλιώς γονέα) και κάτι ειδικό (μιαν υποκλάση ή αλλιώς παιδί της). Παριστάνεται με μια συνεχή γραμμή με κλειστό βέλος που δείχνει προς τη βασική κλάση:



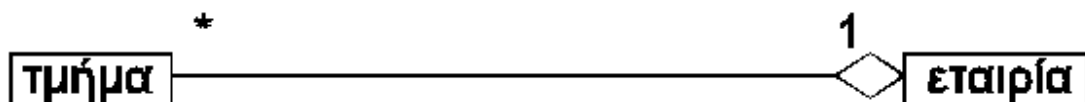
3.3 Σύνδεση

Η σύνδεση αναφέρεται σε αντικείμενα τα οποία συνδέονται με κάποιο τρόπο με άλλα. Όταν δύο κλάσεις είναι συνδεδεμένες μπορεί κανείς να μεταβεί από αντικείμενα της μιας σε αντικείμενα της άλλης. Η σύνδεση παριστάνεται με μια ευθεία γραμμή ανάμεσα στα δύο αντικείμενα.

- Αν η σύνδεση δεν είναι αμφίδρομη τότε η κατεύθυνσή της μπορεί να οριστεί με ένα ανοιχτό βέλος.
- Το όνομα της σύνδεσης μπορεί να γραφεί πάνω από τη γραμμή, ενώ η κατεύθυνση του ονόματος ορίζεται από ένα βέλος πλάι στο όνομα.
- Ο ρόλος των οντοτήτων που συνδέονται προσδιορίζεται από ένα όνομα στην κάθε άκρη της γραμμής.
- Ο αριθμός που δηλώνει πόσα αντικείμενα αντιστοιχούν σε κάθε αντικείμενο στην άλλη άκρη της σχέσης (πολλαπλότητα (*multiplicity*)) δηλώνεται από έναν αριθμό (π.χ. 3), ή μια περιοχή αριθμών (π.χ. 1..* για ένα έως πολλά) πάνω από την αντίστοιχη άκρη της γραμμής.

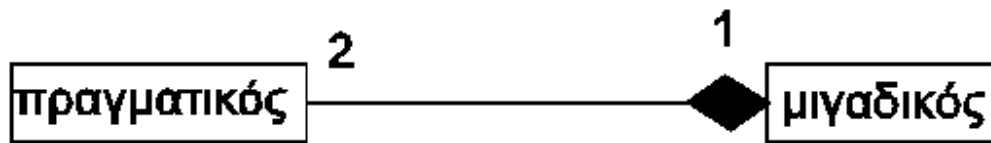


Αν σε μια σχέση τα αντικείμενα απαρτίζουν τμήματα ενός όλου, τότε αυτή απεικονίζεται ως συγκρότημα (*aggregation*) με την παράσταση ενός διαμαντιού στην άκρη του "όλου".



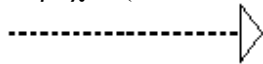
Αν σχέση τα αντικείμενα που απαρτίζουν τμήματα ενός όλου έχουν την ίδια διάρκεια ζωής με το όλο, τότε αυτή απεικονίζεται ως σύνθεση (*composition*) με την παράσταση

ενός γεμάτου διαμαντιού στην άκρη του "όλου".



3.4 Υλοποίηση

Η υλοποίηση δηλώνει πως ο εξυπηρετούμενος (αυτός που βρίσκεται στην ουρά του βέλους) υποστηρίζει τη διεπαφή (τουλάχιστον όλες τις πράξεις) που ορίζονται από τον παροχέα (αυτόν που βρίσκεται στην κεφαλή του βέλους):



3.5 Είδη εξαρτήσεων

Με τη χρήση μιας εξάρτησης εκφράζουμε σημασιολογικές (*semantic*) σχέσεις ανάμεσα σε στοιχεία του μοντέλου. Σε τέτοιες περιπτώσεις μια αλλαγή σε ένα στοιχείο της εξάρτησης μπορεί να έχει επίπτωση στο άλλο. Διακρίνουμε τα παρακάτω είδη εξάρτησης:

πρόσβαση (*access*)

Έδεια σε κάποια στοιχεία από ένα τμήμα να έχουν πρόσβαση σε στοιχεία από άλλο τμήμα (*access*).

σύνδεση (*binding*)

Παροχή τιμών σε ένα πρότυπο για να δημιουργήσει ένα νέο στοιχείο (*bind*).

κλήση (*call*)

Μια μέθοδος καλεί μια άλλη (*call*).

απόρροια (*derivation*)

Ένα στοιχείο που μπορεί να υπολογιστεί από κάποιο άλλο (*derive*).

friend

Ένα στοιχείο μπορεί να έχει πρόσβαση σε στοιχεία άλλης κλάσης παρά τους όποιους περιορισμούς (*friend*).

εισαγωγή (*import*)

Έδεια σε ένα τμήμα να εισάγει και να χρησιμοποιήσει τα στοιχεία ενός άλλου τμήματος (*import*).

δημιουργία (*instantiation*)

Μια μέθοδος μιας κλάσης δημιουργεί αντικείμενα μιας άλλης κλάσης (*instantiate*).

παράμετρος (*parameter*)

Σχέση ανάμεσα σε μια λειτουργία και τις παραμέτρους της (*parameter*).

δημιουργία (*realization*)

Σχέση ανάμεσα σε μια προδιαγραφή και την υλοποίησή της (*realize*).

εκλέπτυνση (*refinement*)

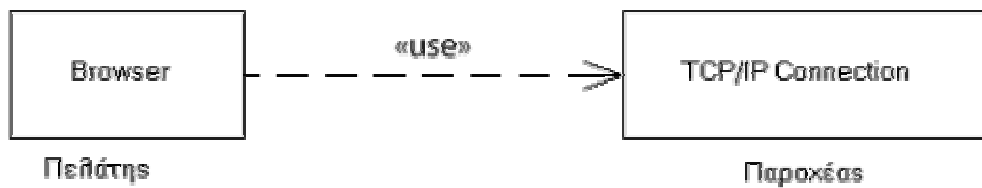
Δήλωση για την ύπαρξη απεικόνισης ανάμεσα σε δύο σημασιολογικά επίπεδα (*refine*).

αποστολή (*send*)

Σχέση ανάμεσα στον αποστολέα και τον παραλήπτη ενός μηνύματος (*send*).

ίχνος (*trace*)

Σχέση ανάμεσα σε δύο στοιχεία δύο διαφορετικών μοντέλων που δεν αποτελεί όμως απεικόνιση (trace).
 χρήση (*usage*)
 Ένα στοιχείο απαιτεί την ύπαρξη ενός άλλου στοιχείου για τη λειτουργία του (π.χ. call, instantiate, parameter, send) (use).
 Στο διάγραμμα της UML γράφουμε μέσα σε εισαγωγικά το είδος της εξάρτησης (αυτό που εμφανίζεται στο τέλος κάθε ορισμού στον παραπάνω πίνακα) πάνω από την αντίστοιχη γραμμή με το βέλος:



4. Διαγράμματα

Η UML ορίζει τα παρακάτω διαγράμματα:

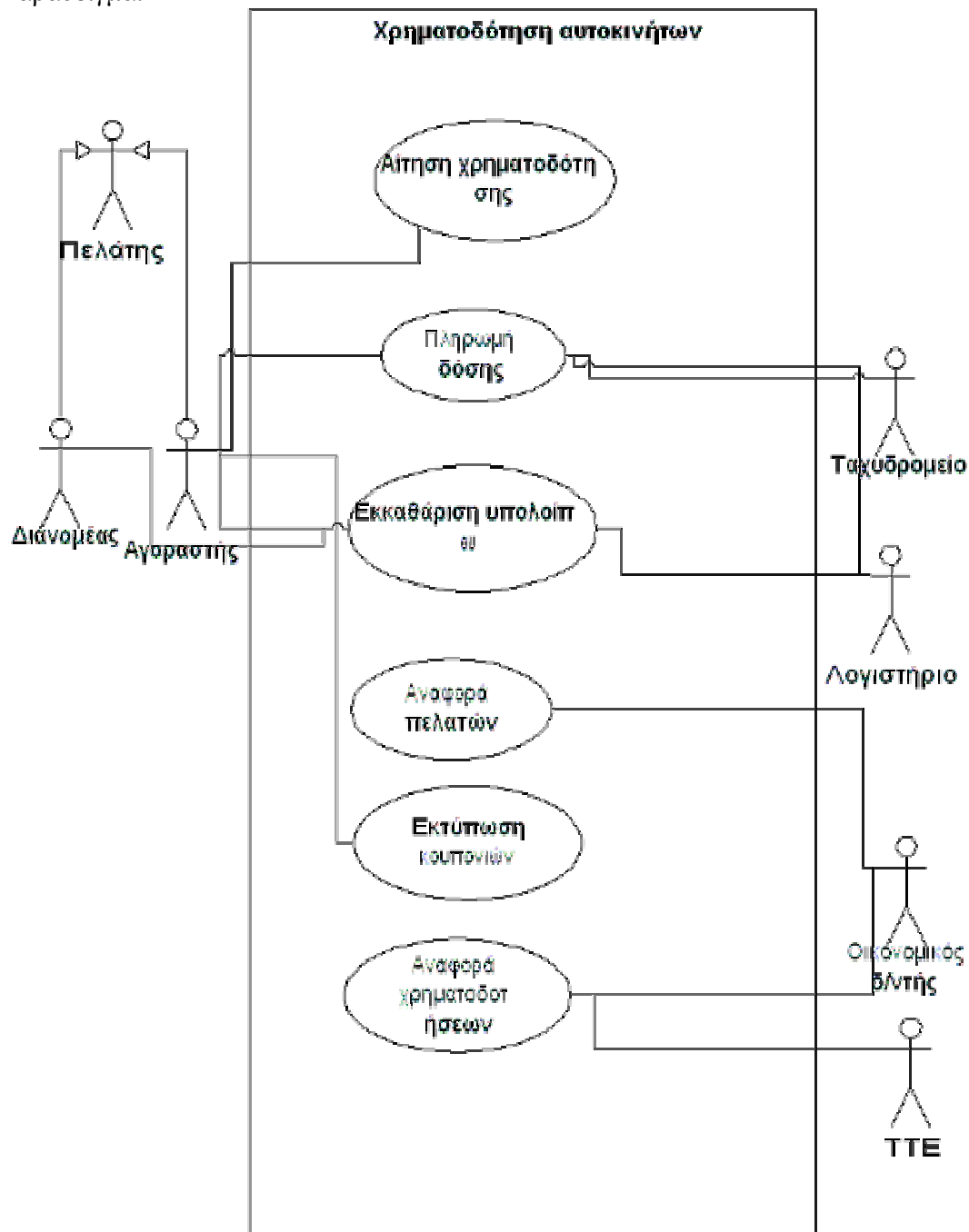
- Διάγραμμα περιπτώσεων χρήσης (*use case diagram*)
- Διαγράμματα δομής
 - ο Διάγραμμα κλάσεων (*class diagram*)
 - ο Διάγραμμα αντικειμένων (*object diagram*)
- Διαγράμματα συμπεριφοράς
 - ο Διάγραμμα καταστάσεων (*statechart diagram*)
 - ο Διάγραμμα δραστηριοτήτων (*activity diagram*)
 - ο Διαγράμματα αλληλεπίδρασης
 - Διάγραμμα ακολουθίας (*sequence diagram*)
 - Διάγραμμα συνεργασίας (*collaboration diagram*)
- Διαγράμματα δομής υλοποίησης
 - ο Διάγραμμα εξαρτημάτων (*component diagram*)
 - ο Διάγραμμα ανάπτυξης (*deployment diagram*)

4.1 Διαγράμματα περιπτώσεων χρήσης

Το διάγραμμα περιπτώσεων χρήσης χρησιμοποιείται για να μοντελοποιήσει το πλαίσιο λειτουργίας του συστήματος καθώς και τις προδιαγραφές του. Περιλαμβάνει:

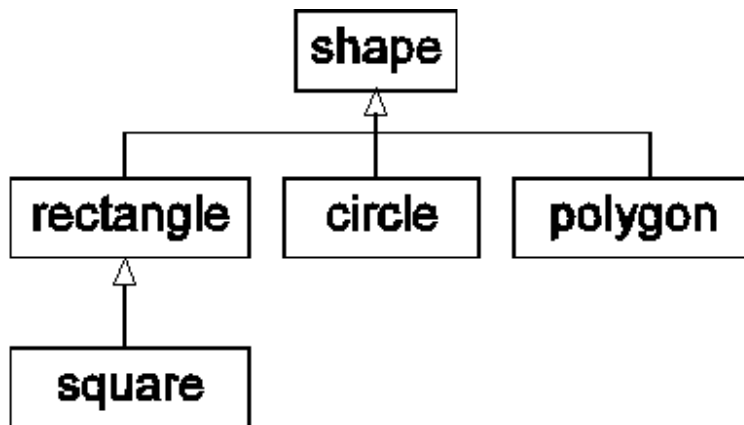
- Περιπτώσεις χρήσης
- Δρώντες (*actors*) (αυτοί που είναι έξω από το σύστημα).
- Σχέσεις εξάρτησης, γενίκευσης, σύνδεσης
- Τα όρια του συστήματος

Παράδειγμα:



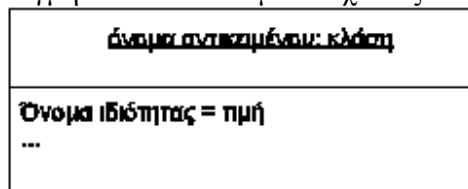
4.2 Διάγραμμα κλάσεων

Το διάγραμμα των κλάσεων ενός συστήματος είναι ένα *διάγραμμα δομής* που περιέχει τις κλάσεις μαζί με του αντίστοιχους δεσμούς εξάρτησης, γενίκευσης και σύνδεσης. Έτσι ένα διάγραμμα κλάσεων μπορεί να απεικονίσει τη χρήση της κληρονομικότητας στο σχεδιασμό με τη χρήση δεσμών γενίκευσης όπως στο παρακάτω σχήμα:

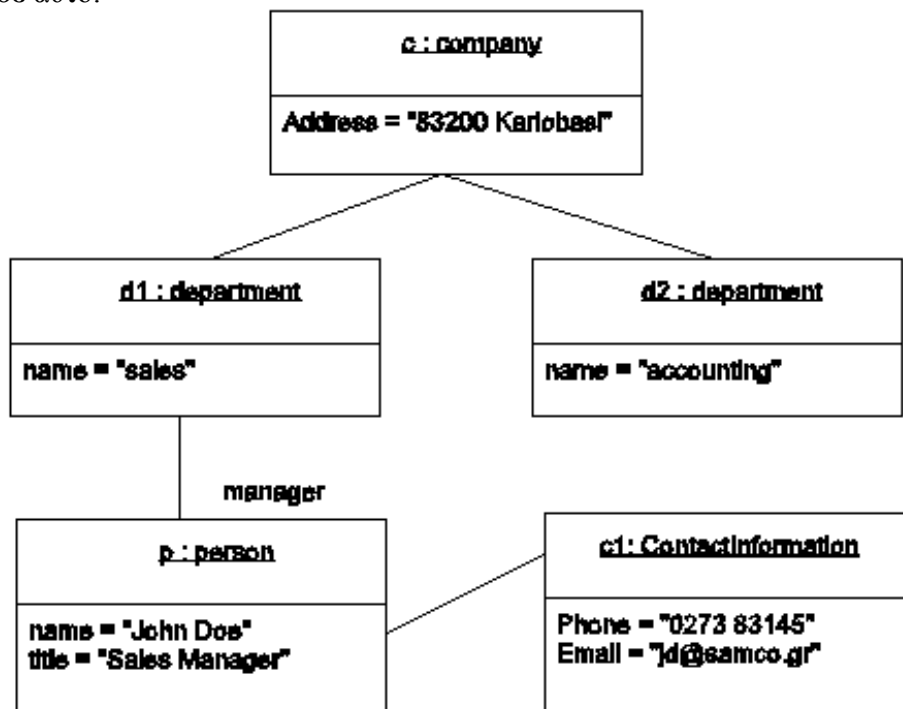


4.3 Διάγραμμα αντικειμένων

Τα διαγράμματα αντικειμένων είναι *διαγράμματα δομής* που χρησιμοποιούνται για το σχεδιασμό της στατικής κατάστασης του συστήματος κατά μια συγκεκριμένη χρονική στιγμή. Κάθε αντικείμενο σχεδιάζεται ως ένα ορθογώνιο με την παρακάτω μορφή:



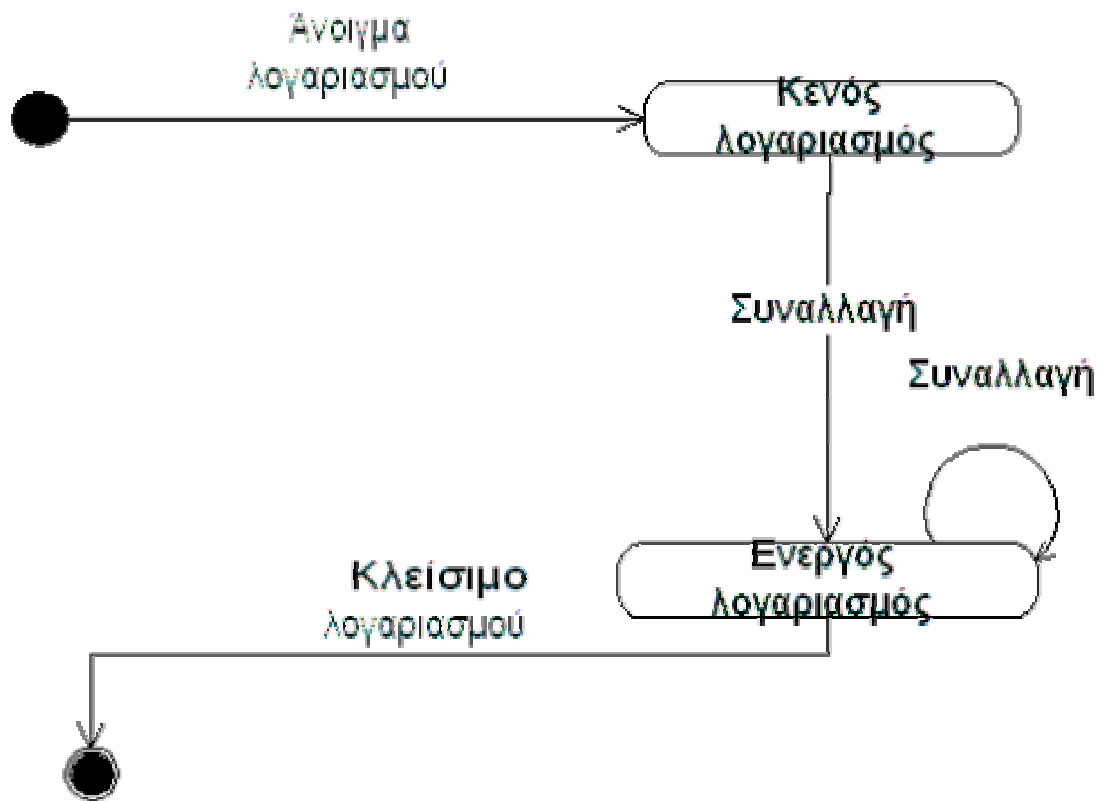
Το σύνολο των αντικειμένων σχεδιάζεται με βάση τους συνδέσμους που ορίζονται πάνω σε αυτό.



4.4 Διάγραμμα καταστάσεων

Το διάγραμμα καταστάσεων είναι ένα *διάγραμμα συμπεριφοράς* που εμφανίζει μια μηχανή καταστάσεων με έμφαση στις μεταπτώσεις μεταξύ καταστάσεων από διάφορα γεγονότα.

Παράδειγμα:

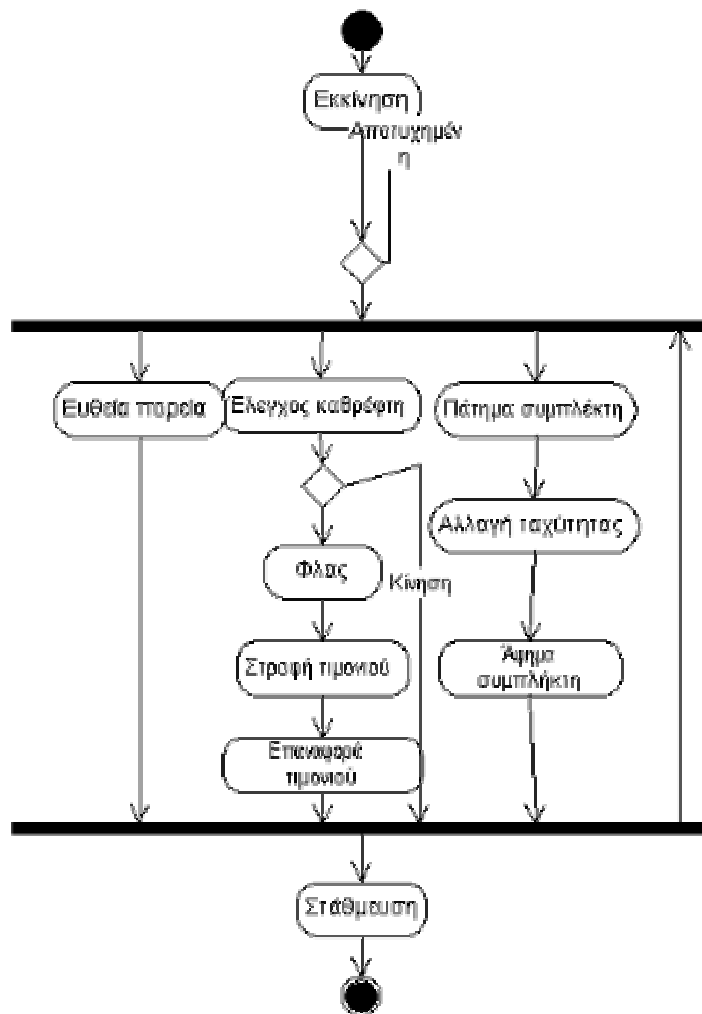


4.5 Διάγραμμα δραστηριοτήτων

Όπως και το διάγραμμα καταστάσεων έτσι και το διάγραμμα δραστηριοτήτων είναι ένα *διάγραμμα συμπεριφοράς* που εμφανίζει μεταπτώσεις. Η διαφορά του από το διάγραμμα καταστάσεων είναι πως οι μεταπτώσεις είναι ανάμεσα σε διαφορετικές δραστηριότητες εκφράζοντας π.χ. τη ροή εργασιών (*workflow*). Εκτός από τα στοιχεία του διαγράμματος καταστάσεων περιλαμβάνει συχνά ακόμα:

- Διακλάδωση υπό συνθήκη (*branch*)
- Διάσπαση (*fork*)
- Ένωση (*join*)

Παράδειγμα:

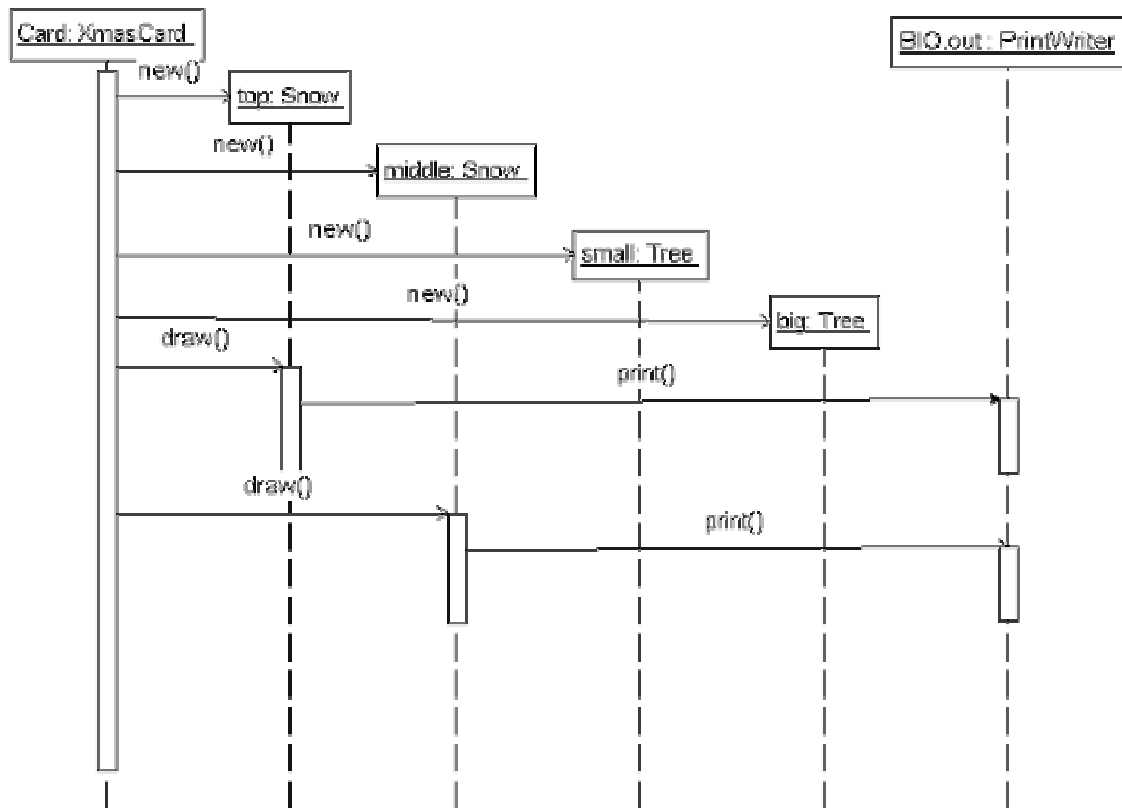


4.6 Διάγραμμα ακολουθίας

Το διάγραμμα ακολουθίας είναι ένα *διάγραμμα αλληλεπίδρασης* (συμπεριφοράς) που παρουσιάζει τον τρόπο που διαφορετικά αντικείμενα συνεργάζονται μεταξύ τους σε μια χρονική ακολουθία. Περιλαμβάνει:

- Αντικείμενα
- Σχέσεις μεταξύ αντικειμένων
- Μηνύματα
- Τη διάρκεια ζωής κάθε αντικειμένου
- Την περιοχή ελέγχου για κάθε αντικείμενο

Παράδειγμα:

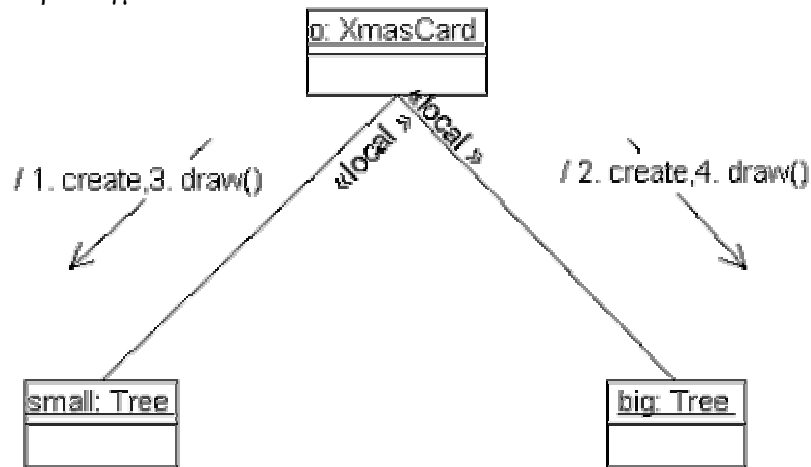


4.7 Διάγραμμα συνεργασίας

Το διάγραμμα συνεργασίας είναι ένα *διάγραμμα αλληλεπίδρασης* (συμπεριφοράς) που παρουσιάζει τον τρόπο που διαφορετικά αντικείμενα σχετίζονται και ανταλλάσσουν μηνύματα μεταξύ τους. Περιλαμβάνει:

- Αντικείμενα
- Σχέσεις μεταξύ αντικειμένων
- Αριθμημένα μηνύματα

Παράδειγμα:



4.8 Διάγραμμα εξαρτημάτων

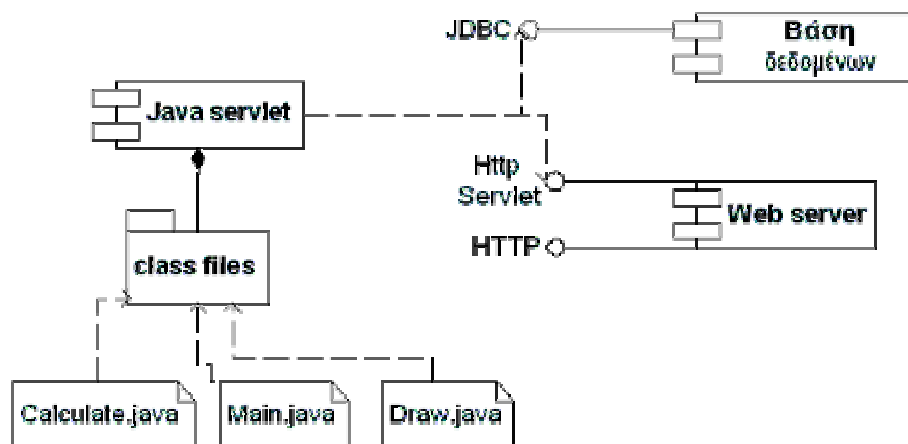
Το διάγραμμα εξαρτημάτων είναι ένα διάγραμμα δομής υλοποίησης που χρησιμοποιείται για να μοντελοποιήσει:

- Πηγαίο κώδικα
- Εκτελέσιμες εκδόσεις
- Βάσεις δεδομένων
- Δυναμικά προσαρμοζόμενα συστήματα

Περιλαμβάνει:

- Εξαρτήματα
- Διεπαφές (*interfaces*)
- Σχέσεις εξάρτησης, γενίκευσης, σύνδεσης και υλοποίησης

Παράδειγμα:

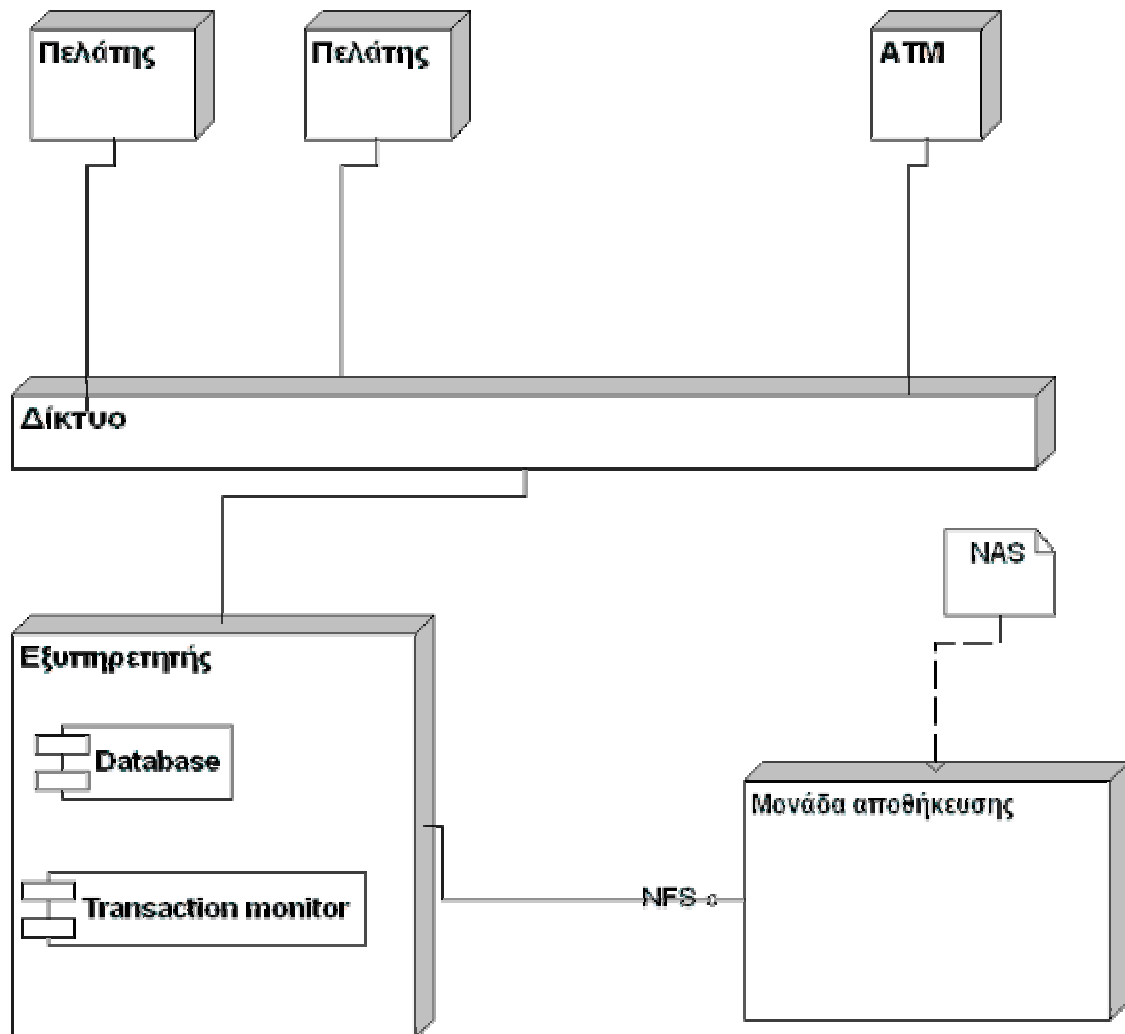


4.9 Διάγραμμα ανάπτυξης

Το διάγραμμα ανάπτυξης είναι ένα *διάγραμμα δομής υλοποίησης* που παρουσιάζει τον τρόπο διαμόρφωσης των υπολογιστικών κόμβων του συστήματος κατά τη λειτουργία του. Περιλαμβάνει:

- Υπολογιστικούς κόμβους
- Σχέσεις εξάρτησης και σύνδεσης
- Εξαρτήματα (μέσα σε κόμβους)
- Πακέτα (που ομαδοποιούν κόμβους)

Παράδειγμα:



4.10 Χρήση διαγραμμάτων

Τα διαγράμματα της UML επιτελούν διαφορετικούς ρόλους ανάλογα με το μοντέλο που μας ενδιαφέρει (Daniels 2002):

Διάγραμμα	Ιδεατό μοντέλο	Μοντέλο προδιαγραφών	Μοντέλο υλοποίησης
Διάγραμμα περιπτώσεων χρήσης	-	Αλληλεπιδράσεις με το λογισμικό	-
Διάγραμμα κλάσεων	Μοντέλα πληροφορίας	Δομές αντικειμένων	Δομές αντικειμένων
Διάγραμμα ακολουθίας ή συνεργασίας	-	Απαιτούμενες αλληλεπιδράσεις αντικειμένων	Υλοποιημένες αλληλεπιδράσεις αντικειμένων
Διάγραμμα δραστηριότητας	Επιχειρηματικές διεργασίες	-	-
Διάγραμμα καταστάσεων	Περιορισμοί αλληλουχίας γεγονότων	Περιορισμοί αλληλουχίας μηνυμάτων	Καθορισμός μηνυμάτων ή αποκρίσεων

5. Βιβλιογραφία

- Martin Fowler και Kendall Scott. *Εισαγωγή στη UML: Συνοπτικός οδηγός της πρότυπης γλώσσας μοντελοποίησης*. Δεύτερη αμερικάνικη έκδοση. Εκδόσεις Κλειδάριθμος, 2001.
- Πανεπιστήμιο Αθηνών. Τμήμα Πληροφορικής. Σημειώσεις του μαθήματος Αντικειμενοστραφής Ανάπτυξη Λογισμικού. <http://cgi.di.uoa.gr/~ys01/> (<http://cgi.di.uoa.gr/~ys01/>)
- Narasimha Bolloju. Improving the quality of business object models using collaboration patterns. *Communications of the ACM*, 47(7):81–86, July 2004.
- Grady Booch, James Rumbaugh, and Ivar Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1999.
- Roger Box and Michael Whitelaw. Experiences when migrating from structured analysis to object-oriented modelling. In *Proceedings of the Australasian conference on Computing education*, pages 12–18. ACM Press, 2000. (doi:10.1145/359369.359372 (<http://dx.doi.org/10.1145/359369.359372>))
- John Daniels. Modeling with a sense of purpose. *IEEE Software*, 19(1):8–10, January/February 2002.
- Roger S. Pressman. *Software Engineering: A Practitioner's Approach*, pages 685–635. McGraw-Hill, fifth edition, 2000. European Adaptation. Adapted by Darrel Ince.
- James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1999.
- Graeme Shanks, Elizabeth Tansley, and Ron Weber. Representing composites in conceptual modeling. *Communications of the ACM*, 47(7):77–80, July 2004.
- Ian Sommerville. *Software Engineering*, pages 42–69. Addison-Wesley, sixth edition, 2001.

- Roel Wieringa. A survey of structured and object-oriented software specification methods and techniques. *ACM Computing Surveys*, 30(4):459–527, December 1998.

6. Ασκήσεις

1. Σχεδιάστε σε UML ένα τουλάχιστον αντιπροσωπευτικό διάγραμμα κάθε τύπου που να αφορά το πληροφοριακό σύστημα της βιβλιοθήκης του πανεπιστημίου.
2. Απεικονίστε σε διάγραμμα καταστάσεων τον τρόπο λειτουργίας ενός κινητού τηλεφώνου.