Αριστοτέλειο
Πανεπιστήμιο
Θεσσαλονίκης

ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗΣ ΑΝΑΛΥΣΗ

Πρότυπα Σχεδίασης

Ιωάννης Σταμέλος
Βάιος Κολοφωτιάς
Πληροφορική

Θεσσαλονίκη, Σεπτέμβριος 2013

Άδειες Χρήσης

Χρηματοδότηση

Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα. Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.

ανοικτάμαθήματα
opencourses

Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Περιεχόμενα

1. Περιεχόμενο Μαθήματος

Εβδομάδα	**Περιεχόμενο**
1η	Εισαγωγή στην Αντικειμενοστρεφή Ανάλυση/UML
2η	Rational Unified Process
3η	Περιπτώσεις Χρήσης
4η	Διαγράμματα Κλάσεων
5η	Διαγράμματα Συνεργασίας
6η	Διαγράμματα Ακολουθίας
7η	**Πρότυπα Σχεδίασης**
8η	Διεργασία ICONIX
9η	Επιχειρηματική Μοντελοποίηση
10η	Υλοποίηση Σχεδίασης με Java
11η	Μετρικές Αντικειμενοστραφούς Σχεδίασης
12η	Επισκόπηση

1.1 Περιεχόμενα ενότητας

Στην ενότητα αυτή, θα ασχοληθούμε με τα πρότυπα σχεδίασης και τις κατηγορίες στις οποίες τα οργανώνουμε. Θα δούμε αναλυτικά τα adapter, Bridge, Composite, Visitor, Abstract Factory και μια μέθοδο υπόδειγμα.

2. Σχεδίαση

Σχεδίαση οποιουδήποτε έργου είναι:

- η αποσύνθεση ενός συστήματος σε τμήματα (μονάδες)
- ο καθορισμός των σχέσεων μεταξύ των τμημάτων
- η ανάθεση αρμοδιοτήτων σε κάθε τμήμα
- η επικύρωση ότι όλα τα τμήματα μαζί επιτυγχάνουν τους σκοπούς του συστήματος

2.1 Κατάλογος Προτύπων Σχεδίασης GoF

23 πρότυπα σχεδίασης

- Όνομα και Κατηγορία
- Σκοπό (μία φράση)
- Συνώνυμα
- Κίνητρο (παράδειγμα προβλήματος)
- Εφαρμοσιμότητα - Applicability(χρησιμοποιείστε το πρότυπα όταν:…)
- Δομή (διάγραμμα κλάσεων)
- Συμμετέχοντες
- Συνεργασία
- Συνέπειες (επιτευχθέντα πλεονεκτήματα)
- Υλοποίηση (language specific)
- Sample Code
- Γνωστές Χρήσεις (παραδείγματα σε πραγματικά συστήματα)
- Συσχετιζόμενα Πρότυπα

2.2 Κατηγορίες Προτύπων Σχεδίασης

- **Creational**: Ασχολούνται με τη διεργασία της δημιουργίας αντικειμένων
- **Structural**: Διαπραγματεύονται τη σύνθεση κλάσεων/αντικειμένων
- **Behavioral**: Χαρακτηρίζουν τους τρόπους με τους οποίους οι κλάσεις αλληλεπιδρούν και κατανέμουν τις αρμοδιότητες

2.3 Γενικοί Κανόνες Προτύπων Σχεδίασης

- Σχεδίαση με διασυνδέσεις
- Προτιμήστε την σύνθεση από την κληρονομικότητα
- Βρείτε τι αλλάζει και ενθυλακώστε το

3. Πρότυπα

3.1 Adapter (Προσαρμογέας)

- **Κατηγορία**: *Structural*
- **Σκοπός**: *Η μετατροπή της διασύνδεσης μιας κλάσης σε μία άλλη που αναμένει το πρόγραμμα πελάτης. Ο προσαρμογέας επιτρέπει τη συνεργασία κλάσεων, η οποία σε διαφορετική περίπτωση θα ήταν αδύνατη λόγω ασύμβατων διασυνδέσεων.*
- **Συνώνυμα**: *Wrapper*
- Συχνά ο κώδικας μιας κλάσης προσφέρεται για επαναχρησιμοποίηση, αλλά αυτή δεν είναι δυνατή λόγω του ότι τα προγράμματα που επιθυμούν να χρησιμοποιήσουν τις λειτουργίες της, αναμένουν διαφορετική διασύνδεση.
- Συνήθως τα προγράμματα πελάτες δεν είναι δυνατόν να τροποποιηθούν (καθώς βρίσκονται ήδη εγκατεστημένα).
- Η κλάση Σχεδίασης είναι επιθυμητό να χρησιμοποιηθεί χωρίς τροποποίηση (καθώς οποιαδήποτε επέμβαση στον κώδικα μιας μεθόδου είναι δυνατόν να προκαλέσει σφάλματα στις υπόλοιπες μεθόδους).
- Εδώ βρίσκει εφαρμογή το πρότυπο "Προσαρμογέας".

3.1.1 Παράδειγμα:

- Εφαρμογή που χειρίζεται διάφορα σχήματα (σημεία, γραμμές, τετράγωνα) με ενιαίο τρόπο, δηλαδή καλεί λειτουργίες επί των σχημάτων, αδιαφορώντας για το συγκεκριμένο είδος σχήματος.
- Τέτοιες λειτουργίες είναι η εμφάνιση του σχήματος, ο καθορισμός του χρώματος, η διαγραφή από την οθόνη, ο καθορισμός θέσης, το γέμισμα με χρώμα κλπ.
- Με άλλα λόγια, όλα τα σχήματα θα πρέπει να ενσωματωθούν σε μία αφηρημένη έννοια σχήματος που θα παρέχει όλες αυτές τις λειτουργίες.

Προφανής αντιμετώπιση:

- **Πολυμορφισμός**: Μία αφηρημένη κλάση ορίζει όλες τις λειτουργίες της διασύνδεσης, ενώ η υλοποίηση των λειτουργιών είναι διαφορετική για κάθε μία από τις παράγωγες κλάσεις.
- Το πρόγραμμα πελάτης είναι σε θέση να χειρίζεται αντικείμενα της αφηρημένης κλάσης διατηρώντας ένα δείκτη προς αυτή, ενώ κατά την εκτέλεση του προγράμματος μεταβιβάζονται ως τιμές στον δείκτη οι διευθύνσεις συγκεκριμένων αντικειμένων των παράγωγων κλάσεων.

- Ζητείται η προσθήκη δυνατότητας σχεδίασης κύκλων: Πρέπει να προστεθεί μία νέα κλάση η οποία θα κληρονομεί από την αφηρημένη κλάση Shape υλοποιώντας την πολυμορφική συμπεριφορά των μεθόδων που δηλώνονται στη διασύνδεση.
- Στο σημείο αυτό, θα πρέπει να γραφεί ο κώδικας για τις μεθόδους display, undisplay και fill.
- Επειδή η συγγραφή των μεθόδων αυτών είναι αρκετά περίπλοκη, αρχικά πραγματοποιείται αναζήτηση εναλλακτικής λύσης, υπό τη μορφή κάποιας κλάσης που ήδη υλοποιεί κύκλους και είναι διαθέσιμη.
- Έστω ότι εντοπίζεται μία τέτοια κλάση με το όνομα OtherCircle, η οποία διαθέτει τις ίδιες λειτουργίες με διαφορετικά ωστόσο ονόματα μεθόδων. Η κλάση OtherCircle δεν μπορεί να ενταχθεί απευθείας στην ιεραρχία των κλάσεων αφενός λόγω της ασυμβατότητας των μεθόδων (ονομάτων και ενδεχομένως παραμέτρων), καθώς και διότι στον ορισμό της δεν κληρονομεί από την κλάση Shape.
- Η λύση στο σημείο αυτό είναι η δημιουργία μιας νέας κλάσης Circle η οποία όντως κληρονομεί από την Shape (και κατά συνέπεια είναι συμβατή με τις λειτουργίες της διασύνδεσης), και η οποία *περιέχει* ένα αντικείμενο της κλάσης OtherCircle.
- Κατά αυτόν τον τρόπο, η κλάση Circle μπορεί να μεταβιβάζει τις αιτήσεις που φθάνουν σε αυτή, στο αντικείμενο OtherCircle που μπορεί να τις ικανοποιήσει. Η υπάρχουσα κλάση OtherCircle επομένως δεν τροποποιείται, αλλά *προσαρμόζεται*.

3.2 Bridge (Γέφυρα)

- **Κατηγορία:** *Structural*
- **Σκοπός:** *Η αποσύνδεση μιας αφαίρεσης από την υλοποίησή της, ώστε να μπορούν να μεταβάλλονται ανεξάρτητα.*
- **Συνώνυμα:** *Handle/Body*
- Όταν μία αφαίρεση μπορεί να έχει περισσότερες από μία υλοποιήσεις, ο συνήθης τρόπος οργάνωσης είναι με τη χρήση κληρονομικότητας.
- Με τον όρο αφαίρεση νοείται μία αφηρημένη κλάση που ορίζει μία διασύνδεση (ένα σύνολο υπογραφών), ενώ υλοποιήσεις είναι οι συγκεκριμένες παράγωγες κλάσεις οι οποίες υλοποιούν τις μεθόδους της αφηρημένης κλάσεις.
- Πρόβλημα: Αφαίρεση και υλοποιήσεις συνδέονται μόνιμα, καθιστώντας δύσκολη την επέκταση, τροποποίηση και επαναχρησιμοποίηση αφαιρέσεων και υλοποιήσεων ανεξάρτητα.
- Το πρότυπο "Γέφυρα" είναι σχετικά δύσκολο στην κατανόησή του. Χρησιμοποιείται ωστόσο σε πληθώρα περιπτώσεων όπου εντοπίζονται:

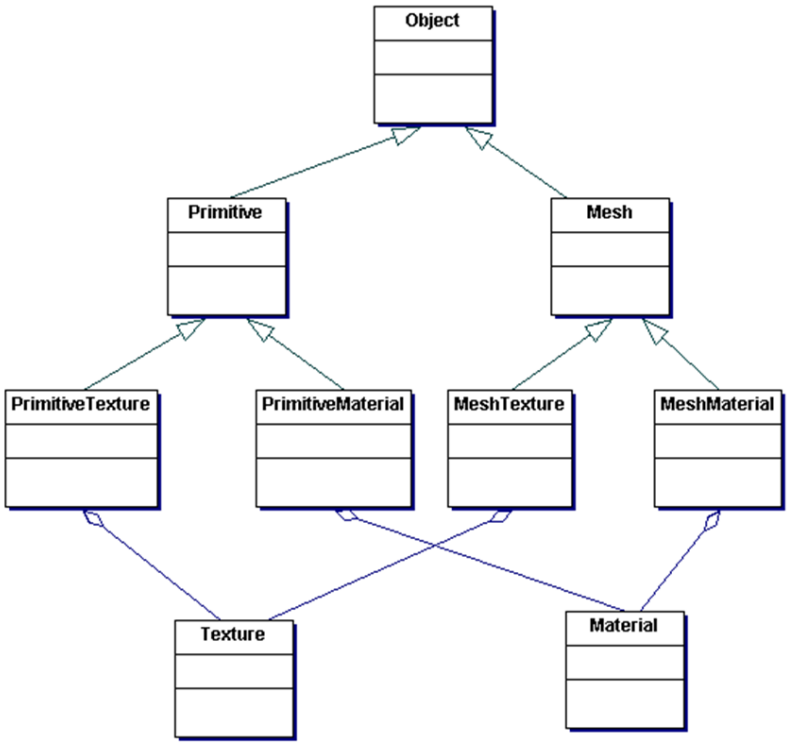
Μεταβολές στην αφαίρεση μιας έννοιας
Μεταβολές στον τρόπο υλοποίησης της έννοιας αυτής

- Η Γέφυρα είναι αντίθετη με τη συνήθη τάση χειρισμού αντίστοιχων καταστάσεων μόνο με κληρονομικότητα. Ικανοποιεί όμως δύο από τους βασικούς κανόνες της αντικειμενοστραφούς κοινότητας: "Εντοπίστε αυτό που μεταβάλλεται και ενσωματώστε το", και "προτιμήστε σύνθεση αντικειμένων από κληρονομικότητα κλάσεων".
- Θεωρούμε μία εφαρμογή η οποία απεικονίζει τρισδιάστατες σκηνές
- Κάθε αντικείμενο στην σκηνή μπορεί να χρωματιστεί με δύο τρόπους:
  είτε με βάση κάποια φωτογραφία,
  είτε με βάση ένα συγκεκριμένο χρώμα.
- Ο χρωματισμός των αντικειμένων πραγματοποιείται από δύο ανεξάρτητα προγράμματα, που αντιστοιχούν στις κλάσεις Texture και Material.
- Η εφαρμογή θα πρέπει να είναι σε θέση να λειτουργήσει με αντικείμενα οποιασδήποτε κατηγορίας χρωματισμού, αλλά δεν οφείλει να γνωρίζει εκ των προτέρων τον τρόπο χρωματισμού κάθε αντικειμένου.

- Τα αντικείμενα θα σχετίζονται με συγκεκριμένο τρόπο χρωματισμού κατά την υλοποίηση των αντικειμένων τους, και κατά συνέπεια είναι λογικό η εφαρμογή να διατηρεί έναν δείκτη προς μία αφηρημένη κλάση *Style* από την οποία θα κληρονομούν οι δύο συγκεκριμένες κατηγορίες με βάση τον τρόπο χρωματισμού. Η κάθε κατηγορία διατηρεί δείκτη προς την αντίστοιχη κλάση χρωματισμού αντικειμένων.

- Η τροποποίηση των απαιτήσεων είναι αναπόφευκτη: Ζητείται ο εμπλουτισμός της εφαρμογής, ώστε να μπορεί να χειρίζεται και την απεικόνιση πιο σύνθετων μοντέλων πχ από 3d studio max (Meshes), τα οποίοι είναι επίσης δυνατόν να χρωματίζονται με οποιονδήποτε από τους δύο τρόπους.

- Η προφανής αντιμετώπιση του προβλήματος από έναν αναλυτή αντικειμενοστραφών συστημάτων είναι η χρήση κληρονομικότητας. Ορίζοντας μία επιπλέον αφηρημένη κλάση, τύπου Object, είναι δυνατόν να κληρονομούν οι δύο ειδικές κατηγορίες μοντέλων, ώστε η εφαρμογή να συνεχίσει να μπορεί να χειρίζεται οποιοδήποτε μοντέλο.



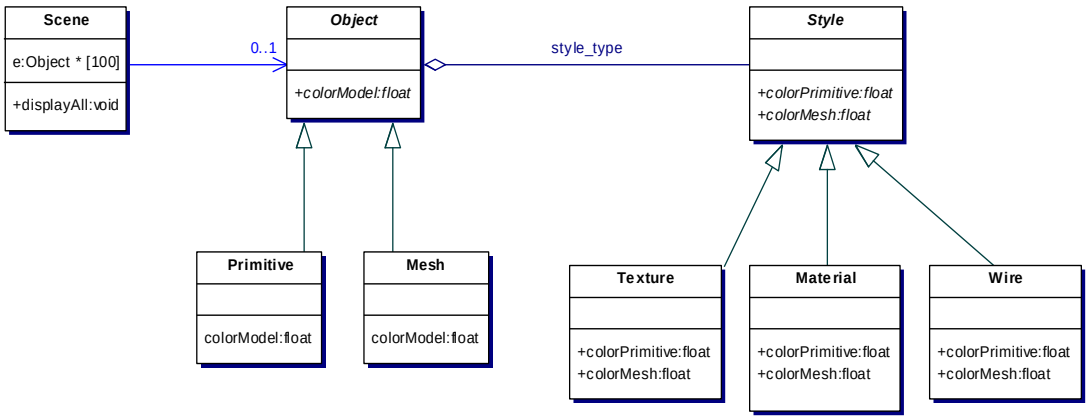
Εικόνα 1 : Διάγραμμα Γέφυρας

- Πολυπλοκότητα του σχεδίου:
- Αν υπάρχει μεταβολή στην υλοποίηση, όπως για παράδειγμα αν προστεθεί ένας επιπλέον τρόπος χρωματισμού (π.χ. wire_render), ο αριθμός των συγκεκριμένων κλάσεων που αναπαριστούν κατηγορίες αντικειμένων/τρόπων χρωματισμού, θα αυξηθεί σε έξι.

- Αν μεταβληθεί η αφαίρεση, όπως για παράδειγμα αν προστεθεί μία επιπλέον κατηγορία μοντέλου, ο αριθμός των κλάσεων επίσης θα αυξηθεί σημαντικά.
- Η εκρηκτική αυτή αύξηση του αριθμού των κλάσεων στο σύστημα, προκαλείται διότι η αφαίρεση (οι κατηγορίες των μοντέλων) και οι υλοποιήσεις (οι διαφορετικοί τρόποι χρωματισμού) έχουν υψηλή σύζευξη.
- Ο στόχος του προτύπου σχεδίασης "Γέφυρα" είναι ο διαχωρισμός των μεταβολών στην αφαίρεση από τις μεταβολές στην υλοποίηση, ώστε ο αριθμός των κλάσεων να αυξάνει γραμμικά.

Στρατηγική:

- Εντοπισμός των εννοιών που μεταβάλλονται και ενσωμάτωσή τους
- Επιλογή σύνθεσης στη θέση της κληρονομικότητας.
- Στη συγκεκριμένη εφαρμογή μεταβάλλονται οι έννοιες του Μοντέλου (*Object*) και του Τρόπου Χρωματισμού(*Style*).
- Η ενσωμάτωση των εννοιών που μεταβάλλονται επιτυγχάνεται με τη χρήση αφηρημένων κλάσεων για κάθε έννοια. Οι διάφορες έννοιες πλέον αναπαρίστανται ως παράγωγες κλάσεις αυτών των αφηρημένων κλάσεων.
- Το ερώτημα που τίθεται είναι με ποιο τρόπο οι δύο αυτές ομάδες κλάσεων θα συσχετιστούν μεταξύ τους.
- Η κληρονομικότητα ωστόσο πρέπει να αποφευχθεί διότι η εισαγωγή ενός επιπλέον επιπέδου δημιουργεί τα προβλήματα που συζητήθηκαν προηγουμένως.
- Ακολουθώντας τον κανόνα που επιβάλλει να προτιμηθεί η σύνθεση, επιλέγεται η επιλογή μιας σχέσης περιεκτικότητας του τρόπου χρωματισμού (Style) στην έννοια του Μοντέλου (Object).



Εικόνα 2

Πλεονεκτήματα:

- Οι υλοποιήσεις δεν είναι μόνιμα συσχετισμένες με μία διασύνδεση. Η υλοποίηση μιας αφαίρεσης μπορεί να διαμορφώνεται ή και να τροποποιείται κατά το χρόνο εκτέλεσης. Στο παράδειγμα, ένα αντικείμενο τύπου Object μπορεί κατά την εκτέλεση του προγράμματος να συσχετιστεί με τον τρόπο χρωματισμού με φωτογραφία και στη συνέχεια να τροποποιηθεί ο τρόπος χρωματισμού του με ένα χρώμα.
- Οποιαδήποτε αλλαγή στις υλοποιήσεις, δεν απαιτεί μεταγλώττιση των κλάσεων στην ιεραρχία της αφαίρεσης ή των προγραμμάτων που τις χρησιμοποιούν.
- Οποιαδήποτε από τις δύο ιεραρχίες κλάσεων (Αφαίρεση και Υλοποίηση) μπορεί να επεκταθεί (προσθέτοντας περισσότερα επίπεδα) με τρόπο ανεξάρτητο.

3.3 Σύνθετο (Composite)

Τύποι Αντικειμένων:

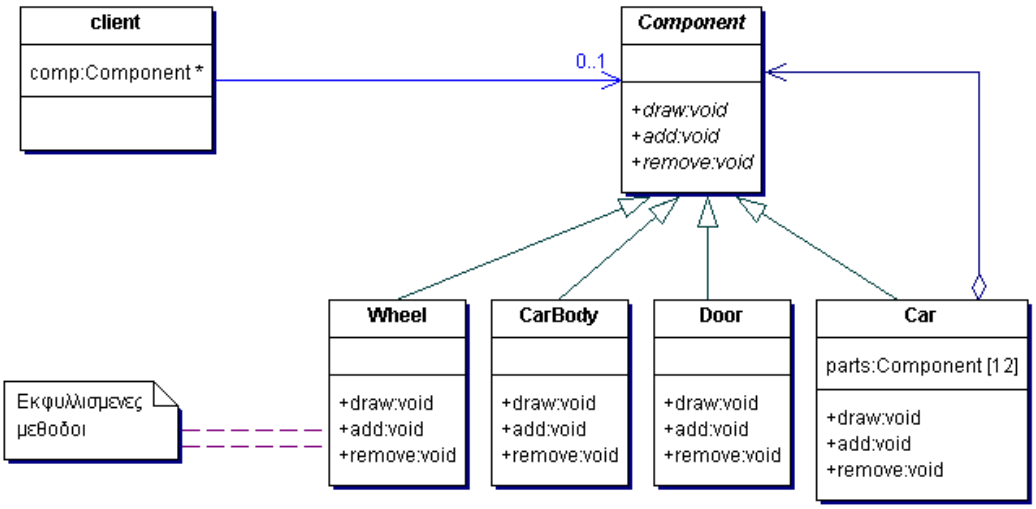
«**πρωταρχικά**»: δηλαδή δεν αναλύονται επιπλέον

«**σύνθετα**»: αποτελούνται από σύνθεση πρωταρχικών

Το χρησιμοποιούμε όταν θέλουμε να έχουμε ενιαία αντιμετώπιση «σύνθετων» & «πρωταρχικών» αντικειμένων

- Έστω ότι θέλουμε να δημιουργήσουμε την τρισδιάστατη γραφική αναπαράσταση ενός αυτοκινήτου.
- Το κάθε μέρος του αυτοκινήτου, μπορεί να αναπαρασταθεί στην οθόνη μέσω της κλήσης μιας δικής του συνάρτησης.

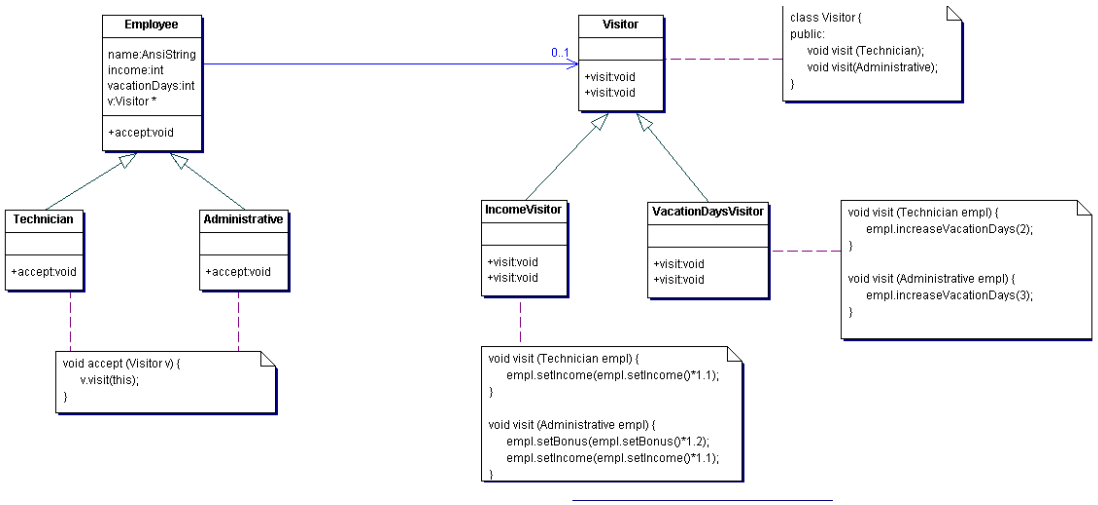
- Επιθυμητό επίσης είναι η έννοια αυτοκίνητο να λειτουργεί ως σύνθεση από τα μέρη που το αποτελούν για ευκολία και μεγαλύτερη ευελιξία



Εικόνα 3 : Διάγραμμα Σύνθετο

3.4 Επισκέπτης (Visitor)

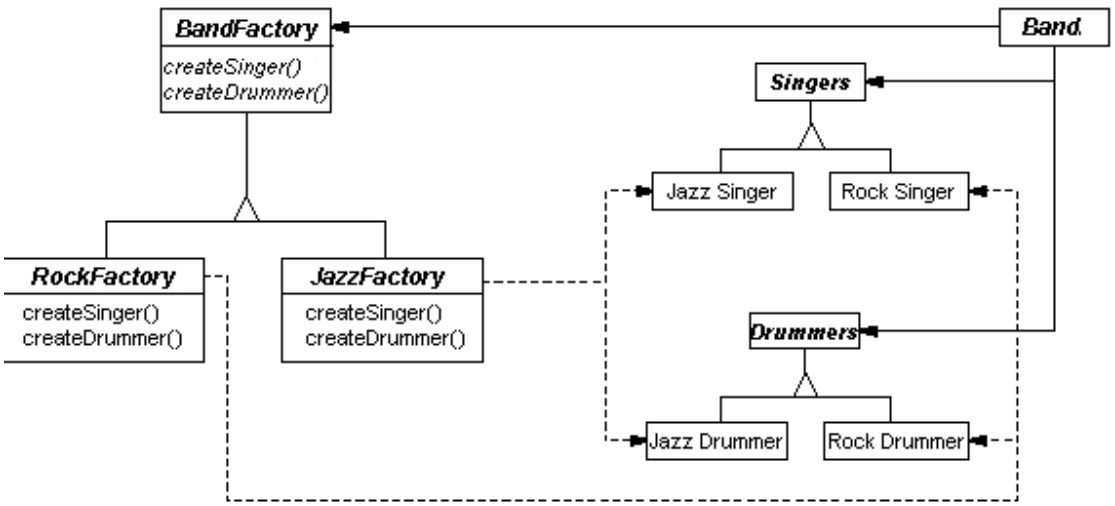
- Το χρησιμοποιούμε όταν θέλουμε να προσθέσουμε λειτουργίες σε αντικείμενα μιας ιεραρχίας, χωρίς να χρειαστεί να τροποποιηθούν οι συγκεκριμένες κλάσεις
- Επιτρέπει να ορίζουμε νέα λειτουργικότητα σε μία κλάση προσθέτοντας στο σύστημα απλά μία νέα κλάση που κληρονομεί από μια αφηρημένη κλάση
- Για να υλοποιηθεί χρησιμοποιεί διπλή αποστολή (dual dispatch)
- Έστω ότι θέλουμε να δημιουργήσουμε ένα σύστημα το οποίο θα χειρίζεται τις αυξήσεις εισοδήματος και ημερών αδείας εργαζομένων μιας εταιρίας.
- Κάθε κατηγορία εργαζομένου πρέπει να διαχειριστεί διαφορετικά
- Κάθε χρόνο ο μισθός όλων των υπαλλήλων αυξάνει κατά 10%. Επιπλέον, το μπόνους των διοικητικών υπαλλήλων αυξάνει κατά 20%
- Τέλος, κάθε χρόνο οι μέρες αδείας των τεχνικών υπαλλήλων αυξάνει κατά 1 ημέρα, ενώ των διοικητικών κατά 2ημέρες



Εικόνα 4

3.5 Αφηρημένο Εργοστάσιο (Abstract Factory)

- Όταν θέλουμε να αποφύγουμε την εξάρτηση από συγκεκριμένες κλάσεις όταν απαιτείται η δημιουργία αντικειμένων
- Για την ομαδοποίηση μεθόδων που δημιουργούν συσχετιζόμενα αντικείμενα σε μία αφηρημένη κλάση.
- Δημιουργία στιγμιότυπων συγκεκριμένων αντικειμένων με εξάρτηση μόνο από αφηρημένες κλάσεις
- Έστω ότι θέλουμε να δημιουργήσουμε ένα πρόγραμμα το οποίο θα διαχειρίζεται την δομή συγκροτημάτων.
- Κάθε συγκρότημα για λόγους απλότητας θα πρέπει να απαρτίζεται από δύο μέλη, έναν κιθαρίστα και έναν τραγουδιστή
- Τέλος επιθυμία των σχεδιαστών του συστήματος είναι τα αντικείμενα να δημιουργούνται μέσω μιας αφηρημένης κλάσης και τα συγκροτήματα όλων των τύπων να υπακούν σε συγκεκριμένο interface

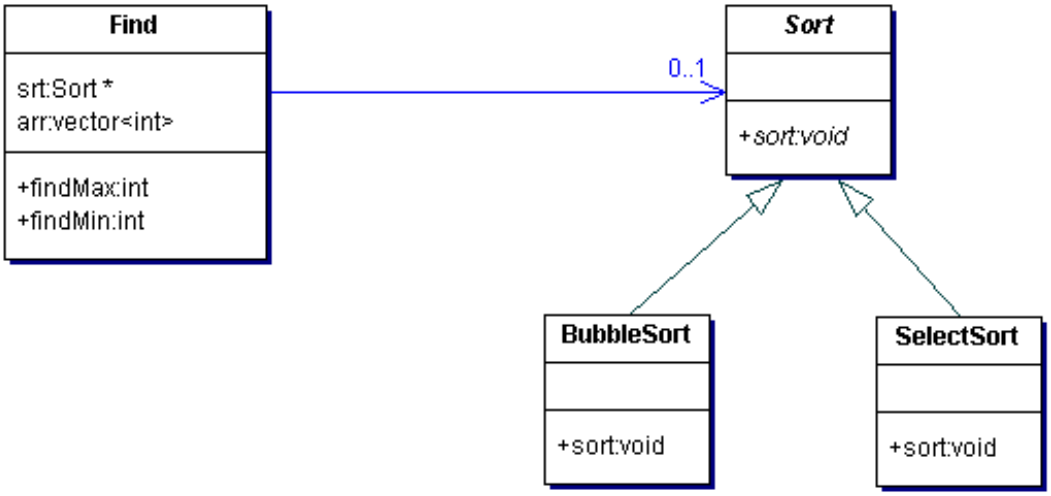

BandFactory
createSinger()
createDrummer()
Band
Singers
Jazz Singer
Rock Singer
RockFactory
createSinger()
createDrummer()
JazzFactory
createSinger()
createDrummer()
Drummers
Jazz Drummer
Rock Drummer

Εικόνα 5

Στρατηγική (Strategy)

- Ορίζει μια οικογένεια αλγορίθμων, τους ενσωματώνει και επιτρέπει την εναλλαγή μεταξύ αυτών.
- Το χρησιμοποιούμε όταν θέλουμε να υλοποιήσουμε έναν κοινό γενικό αλγόριθμο (π.χ. ταξινόμηση) αλλά και πολλές διαφοροποιήσεις του (π.χ. φυσαλίδας, εισαγωγής κ.τ.λ.)
- Έστω ότι θέλουμε να δημιουργήσουμε ένα πρόγραμμα το οποίο να βρίσκει την μέγιστη και την ελάχιστη τιμή σε έναν πίνακα
- Επιπλέον, επιθυμούμε για την υλοποίηση του προγράμματος να εκτελείται αρχικά μια ταξινόμηση του πίνακα.

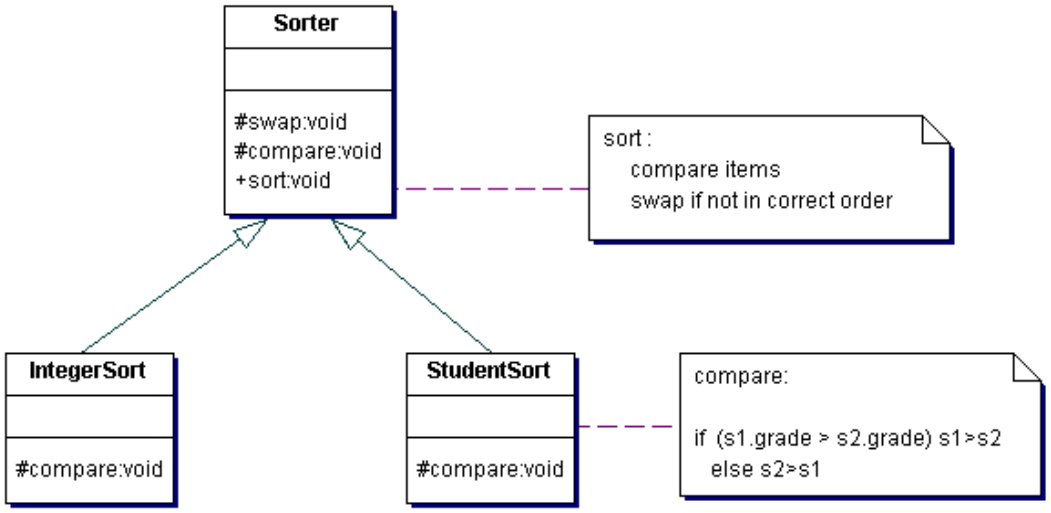
Τέλος, επιθυμητό είναι ο χρήστης του προγράμματος να επιλέγει τον αλγόριθμο ταξινόμησης κατά τη διάρκεια της εκτέλεσης



Εικόνα 6 : Abstract Factory

4. Μέθοδος Υπόδειγμα (Template Method)

- Χρησιμοποιείται σε περιπτώσεις παρόμοιες με αυτές του Strategy.
- Χρησιμοποιείται όταν ο γενικός αλγόριθμος που θέλουμε να εκτελέσουμε λειτουργεί σε βήματα και τα βήματα αυτά διαφοροποιούνται ανάλογα με την περίπτωση
- Δεν μεταβιβάζει την αρμοδιότητα σε άλλη κλάση, αλλά εκμεταλλεύεται το μηχανισμό της κληρονομικότητας
- Έστω ότι κατασκευάζουμε ένα λογισμικό το οποίο να κατατάσσει μια λίστα με μαθητές σε αύξουσα σειρά ανάλογα με τον βαθμό τους.
- Επιπλέον, θεωρούμε ότι έχουμε μια έτοιμη κλάση Sorter η οποία εκτελεί την ταξινόμηση φυσαλίδας κάνοντας χρήση τριών συναρτήσεων:
  - sort
  - swap
  - compare



Εικόνα 7 : Κλάση sorter