Ατομική Διπλωματική Εργασία

# ΣΧΕΔΙΑΣΤΙΚΑ ΠΡΟΤΥΠΑ ΓΙΑ ΤΙΣ ΕΡΕΥΝΗΤΙΚΕΣ ΠΕΡΙΟΧΕΣ ΤΗΣ ΑΛΛΗΛΕΠΙΔΡΑΣΗΣ ΑΝΘΡΩΠΟΥ-ΥΠΟΛΟΓΙΣΤΗ, ΤΑ ΣΥΣΤΗΜΑΤΑ ΠΡΑΓΜΑΤΙΚΟΥ ΧΡΟΝΟΥ ΚΑΙ ΤΟΥ ΣΧΕΔΙΑΣΜΟΥ ΙΣΤΟΣΕΛΙΔΩΝ

**Λοΐζος Τσιάτταλος**

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

**Σεπτέμβριος 2012**

# ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

## ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ


**ΣΧΕΔΙΑΣΤΙΚΑ ΠΡΟΤΥΠΑ ΓΙΑ ΤΙΣ ΕΡΕΥΝΗΤΙΚΕΣ ΠΕΡΙΟΧΕΣ ΤΗΣ ΑΛΛΗΛΕΠΙΔΡΑΣΗΣ ΑΝΘΡΩΠΟΥ-ΥΠΟΛΟΓΙΣΤΗ, ΤΑ ΣΥΣΤΗΜΑΤΑ ΠΡΑΓΜΑΤΙΚΟΥ ΧΡΟΝΟΥ ΚΑΙ ΤΟΥ ΣΧΕΔΙΑΣΜΟΥ ΙΣΤΟΣΕΛΙΔΩΝ**


**Λοΐζος Τσιάτταλος**

Επιβλέπων Καθηγητής
Γιώργος Παπαδόπουλος


Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Σεπτέμβριος 2012

## Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου κ. Γιώργο Παπαδόπουλο, για την χρήσιμη καθοδήγηση που μου παρείχε κατά την διάρκεια εκπόνησης της διπλωματικής μου εργασίας και για την άψογη συνεργασία που είχαμε όλο αυτό το διάστημα. Ευχαριστώ επίσης την οικογένεια και τους φίλους μου για την στήριξη και τις πολύτιμες συμβουλές τους καθ΄όλη τη διάρκεια των σπουδών μου και ιδιαίτερα τον τελευταίο χρόνο.

## Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας είναι η μελέτη των βασικότερων σχεδιαστικών προτύπων στις ερευνητικές περιοχές της Αλληλεπίδρασης ανθρώπου-υπολογιστή(Human-Computer Interaction), των συστημάτων πραγματικού χρόνου(Real-Time Systems), και του σχεδιασμού ιστοσελίδων(Web Desing). Η μελέτη αυτών των προτύπων έχει τη δική της σημασία για κάθε μία από τις πιο πάνω ερευνητικές περιοχές. Για την αλληλεπίδραση ανθρώπου-υπολογιστή τα πρότυπα αυτά λύνουν το επαναλαμβανόμενο πρόβλημα της ευχρηστίας στις διεπιφάνειες χρήσης κατά τη σχεδίαση διαδραστικών συστημάτων. Για τα συστήματα πραγματικού χρόνου τα σχεδιαστικά πρότυπα είναι απαραίτητα για την αντιμετώπιση ζητημάτων όπως συγχρονισμό(concurrency), κατανομή των πόρων(resource sharing), και διανομή(distribution). ΄Οσον αφορά το σχεδιασμό ιστοσελίδων, η μελέτη των αντίστοιχων προτύπων παρέχει χρήσιμη καθοδήγηση κατά την κατασκευή μιας ιστοσελίδας με τρόπο ώστε ο χρήστης να μπορεί να πλοηγείται εύκολα και ευχάριστα σε αυτήν και να βρίσκει γρήγορα την πληροφορία που θέλει.

Η $1^{η}$ ενότητα της διπλωματικής εργασίας περιλαμβάνει μια εισαγωγή στη UML, τη γλώσσα μοντελοποίησης των σχεδιαστικών προτύπων που χρησιμοποιούνται κυρίως στη βιομηχανία λογισμικού. Οι ενότητες 2-4 περιγράφουν τα σπουδαιότερα σχεδιαστικά πρότυπα αντικειμενοστραφούς σχεδιασμού για την παραγωγή λογισμικού. Από την ενότητα 5, μπαίνουμε στο κυρίως θέμα της διπλωματικής μας εργασίας, αφού οι ενότητες 5-7 παρουσιάζουν αναλυτικά τα σχεδιαστικά πρότυπα των ερευνητικών περιοχών που αναφέραμε πιο πάνω.

Τέλος, στην τελευταία ενότητα κάναμε μια προσπάθεια να μοντελοποιήσουμε όσο αυτό ήταν δυνατό μερικά από τα περιγραφικά πρότυπα των περιοχών της αλληλεπίδρασης ανθρώπου-υπολογιστή και του σχεδιασμού ιστοσελίδων, βασισμένοι σε μία πρώτη προσέγγιση που έγινε από κάποιους ερευνητές.

# Περιεχόμενα


Κεφάλαιο **1** **Εισαγωγή**…….. **1**

1.1 Εισαγωγή στη UML…….....1

1.2 Τι είναι τα πρότυπα σχεδίασης…….....4

1.3 Περιγραφή ενός προτύπου σχεδίασης…..…….5

Κεφάλαιο **2** **Κατασκευαστικά Πρότυπα**…… …**7**

2.1 Εισαγωγή……..7

2.2 Πρότυπο Factory method……..……...7

2.3 Πρότυπο Singleton……...9

2.4 Πρότυπο Prototype……10

2.5 Πρότυπο Builder……11

2.6 Πρότυπο Abstract Factory…….12

Κεφάλαιο **3** **Δομικά Πρότυπα**…….....**14**

3.1 Εισαγωγή……...14

3.2 Πρότυπο Adapter……...14

3.3 Πρότυπο Bridge……,16

3.4 Πρότυπο Composite……...17

3.5 Πρότυπο Decorator……18

3.6 Πρότυπο Proxy ……..20

Κεφάλαιο **4** **Συμπεριφορικά Πρότυπα**.....…… **21**

4.1 Εισαγωγή……21

4.2 Πρότυπο Observer……..22

4.3 Πρότυπο State……23

4.4 Πρότυπο Strategy……...24

4.5 Πρότυπο Template method……25

4.6 Πρότυπο Visitor…….26

Κεφάλαιο **5** **Σχεδιαστικά πρότυπα για αλληλεπίδραση ανθρώπου-υπολογιστή……28**

5.1 Εισαγωγή……28

5.2 Η ανάγκη για κοινή γλώσσα στην αλληλεπίδραση ανθρώπου-υπολογιστή……29
5.3 Κατηγοριοποίηση αλληλεπιδραστικών προτύπων ……30
5.4 Ταξινόμηση αλληλεπιδραστικών προτύπων……..33
5.5 Αξιολόγηση αλληλεπιδραστικών προτύπων……..43

Κεφάλαιο **6** **Σχεδιαστικά πρότυπα για συστήματα πραγματικού χρόνου……..……50**
6.1 Εισαγωγή……50
6.2 Η ανάγκη για κοινή γλώσσα στα συστήματα πραγματικού χρόνου……..50
6.3 Πρότυπα επαναχρησιμοποίησης……51
6.4 Πρότυπα κατανεμημένων συστημάτων…….55
6.5 Πρότυπα ασφάλειας και αξιοπιστίας…….57
6.6 Απλά πρότυπα……59

Κεφάλαιο **7** **Σχεδιαστικά πρότυπα για σχεδιασμό ιστοσελίδων……..62**
7.1 Εισαγωγή……62
7.2 Η ανάγκη για κοινή γλώσσα στο σχεδιασμό ιστοσελίδων……63
7.3 Πρότυπα Web Site Level……...64
7.4 Πρότυπα Web Page Level……..68
7.5 Πρότυπα Ornamentation Level……..73

Κεφάλαιο **8** **Μοντελοποίηση περιγραφικών προτύπων με τη χρήση διαγραμμάτων UML…….82**
8.1 Εισαγωγή……82

8.2 Μοντελοποίηση προτύπων αλληλεπίδρασης ανθρώπου υπολογιστή……84
8.2.1 Μοντελοποίηση προτύπου Narrative…….84
8.2.2 Μοντελοποίηση προτύπου Form…..……84
8.2.3 Μοντελοποίηση προτύπου Control Panel……..85
8.2.4 Μοντελοποίηση προτύπου Composed Command…….85
8.2.5 Μοντελοποίηση προτύπου Navigable Spaces……86
8.2.6 Μοντελοποίηση προτύπου Small Groups of Related Things…….....87
8.2.7 Μοντελοποίηση προτύπου Hierarchical Set……..87
8.2.8 Μοντελοποίηση προτύπου Optional Detail……88
8.2.9 Μοντελοποίηση προτύπου Interaction History……..89
8.2.10 Μοντελοποίηση προτύπου Grid Layout……89

8.3 Μοντελοποίηση προτύπων σχεδιασμού ιστοσελίδων……90
8.3.1 Μοντελοποίηση προτύπου Welcome……90
8.3.2 Μοντελοποίηση προτύπου Indication……90
8.3.3 Μοντελοποίηση προτύπου Ready…….....91
8.3.4 Μοντελοποίηση προτύπου Busy……....92
8.3.5 Μοντελοποίηση προτύπου Second Chance……92
8.3.6 Μοντελοποίηση προτύπου Print……..93

8.3.7 Μοντελοποίηση προτύπου About This……..94
8.3.8 Μοντελοποίηση προτύπου Secret……...94
8.3.9 Μοντελοποίηση προτύπου Subscription ……95
8.3.10 Μοντελοποίηση προτύπου Search……..95

Κεφάλαιο **9** **Συμπεράσματα και Μελλοντική Εργασία ……...97**

9.1 Συμπεράσματα……97

9.2 Μελλοντική Εργασία……..97

Βιβλιογραφία……98

# Κεφάλαιο 1

## Εισαγωγή


1.1 Εισαγωγή στη UML 1
1.2 Τι είναι τα πρότυπα σχεδίασης 4
1.3 Περιγραφή ενός προτύπου σχεδίασης 5


### 1.1 Εισαγωγή στη UML

Τα πρότυπα σχεδίασης πρεσβεύουν τον πιο ορθό τρόπο ανάπτυξης του λειτουργικού σώματος σε ένα σύστημα, των οποίων η επίδραση πολλές φορές είναι ευεργετική. Η μορφή τους παρουσιάζεται με τη χρήση της Ενοποιημένης Γλώσσας Μοντελοποίησης, UML. Η UML αποτελεί την κυρίαρχη γλώσσα μοντελοποίησης πληροφοριακών συστημάτων της αντικειμενοστραφούς τεχνολογίας και τυγχάνει ευρείας αποδοχής στη βιομηχανία κατασκευής λογισμικού.

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. Μπορεί να προσαρμοστεί ώστε να ταιριάζει σε διάφορες καταστάσεις ανάπτυξης και κύκλους ζωής λογισμικού..

Βασισμένη στις αρχές της αντικειμενοστρέφειας, η UML μπορεί να χρησιμοποιηθεί σε όλες τις φάσεις ανάπτυξης λογισμικού, από την προδιαγραφή των απαιτήσεων μέχρι τη σχεδίαση και τη συγγραφή του κώδικα. Χρησιμοποιώντας το μηχανισμό της αφαίρεσης εστιάζεται στις απαραίτητες πληροφορίες που μπορεί να κατανοηθούν, απεικονισθούν και τεκμηριωθούν πριν αλλά και κατά τη διάρκεια της ανάπτυξης του συστήματος. Είναι εξαιρετικά χρήσιμη για την περιγραφή διαφόρων εναλλακτικών σχεδίων και προσεγγίσεων του ιδίου προβλήματος, παρέχοντας μια κοινή γλώσσα επικοινωνίας σε όλες τις ομάδες ανάπτυξης.

Παρόλο που η UML δεν είναι μια μεθοδολογία ανάπτυξης λογισμικού αποτελεί απαραίτητο εργαλείο για την επιτυχή ανάπτυξη λογισμικού. Διατηρεί μία ουδέτερη στάση ως προς τις διάφορες μεθοδολογίες και είναι συμβατή με τις περισσότερες αν όχι όλες.

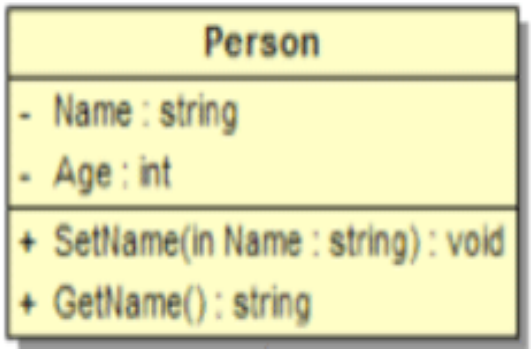
Η δύναμη της UML κρύβεται στους διάφορους τύπους διαγραμμάτων που παρέχει .Τα διαγράμματα της UML περιλαμβάνουν τη δυναμική όψη του συστήματος, τη στατική όψη, τους περιορισμούς και την τυποποίηση τους.

Η δυναμική όψη παρουσιάζεται με το διάγραμμα περιπτώσεων χρήσης(use case diagram),το οποίο χρησιμοποιείται για τη μοντελοποίηση της συμπεριφοράς ενός συστήματος και υποσυστημάτων του από περιπτώσεις και εναλλακτικά σενάρια, όπως αυτή γίνεται αντιληπτή από τον τελικό χρήστη. Τα διαγράμματα ροής εργασίας(workflow diagrams) καθορίζουν τη συνολική επιχειρηματική διεργασία που θα υποστηρίξει το σύστημα. Τα διαγράμματα δραστηριοτήτων(activity diagrams) εμφανίζουν όλες τις δραστηριότητες που μπορούν να προκύψουν στο σύστημα καθώς αλλάζουν οι τιμές ενός αντικειμένου. Τα διαγράμματα καταστάσεων(state diagrams) υποδεικνύουν τις πιθανές καταστάσεις στις οποίες ένα αντικείμενο μπορεί να βρεθεί μέσω ανταλλαγής μηνυμάτων τα οποία αναπαριστούν ένα γεγονός.

Η στατική όψη απεικονίζεται με διαγράμματα κλάσεων(class diagrams), τις διάφορες σχέσεις(γενίκευση, συσχέτιση, εξάρτηση) και την επεκτασιμότητα (περιορισμοί, τιμές ετικετών, και στερεότυπα). Τα διαγράμματα ακολουθίας(sequence diagrams) δείχνουν τον τρόπο με τον οποίο τα μηνύματα ρέουν από ένα αντικείμενο σε ένα άλλο, τυποποιώντας τις άτυπες περιγραφές των γεγονότων στις απαιτήσεις. Τα διαγράμματα συνεργασί-ας(collaboration diagrams) χρησιμοποιούν πληροφορίες για τα αντικείμενα και τις ακολουθίες για να δείξουν τη ροή των γεγονότων μεταξύ των αντικειμένων. Τα διαγράμματα πακέτων(package diagrams) δείχνουν τον τρόπο που οι κλάσεις διαιρούνται λογικά σε ομάδες ενώ τα διαγράμματα συστατικών(component diagrams) αντανακλούν τις τελικές μονάδες του συστήματος. Οι κλάσεις αποτελούν τη βάση κατασκευής οποιουδήποτε αντικειμενοστρεφούς συστήματος. Το μοντέλο του πεδίου του προβλήματος(domain model) αποτελείται από κλάσεις που αποτελούν έννοιες του πεδίου προβλήματος, χωρίς αυτό να σημαίνει ότι θα χρησιμοποιηθούν στο λογισμικό που θα κατασκευαστεί. Επίσης, για κάθε κλάση του μοντέλου μπορούν να αναγραφούν κάποιες υψηλού επιπέδου υποχρεώσεις (λειτουργίες) αλλά και να απεικονιστούν οι συσχετίσεις μεταξύ των κλάσεων. Στο στάδιο της επεξεργασίας το μοντέλο του πεδίου προβλήματος θα αποτελέσει τη βάση για την επιλογή των κλάσεων αλλά και των συσχετίσεων, ίσως με περισσότερες αλλαγές και βελτιώσεις. Οι λειτουργίες των κλάσεων αποτελούν τις μεθόδους(methods) των αντικειμένων αυτών των

κλάσεων ,που καλούνται από άλλα αντικείμενα συνεργαζόμενων κλάσεων. Στα διαγράμματα κλάσεων υπάρχει η δυνατότητα απεικόνισης υψηλού επιπέδου λεπτομερειών , όπως οι ιδιότητες και λειτουργίες μιας κλάσης ,σχέσεις κληρονομικότητας μεταξύ κλάσεων , εξαρτήσεις μεταξύ πακέτων κ.ο.κ..

Μια κλάση σχεδιάζεται ως ένα ορθογώνιο διαμελισμένο σε τρία τμήματα όπως στο παρακάτω σχήμα:

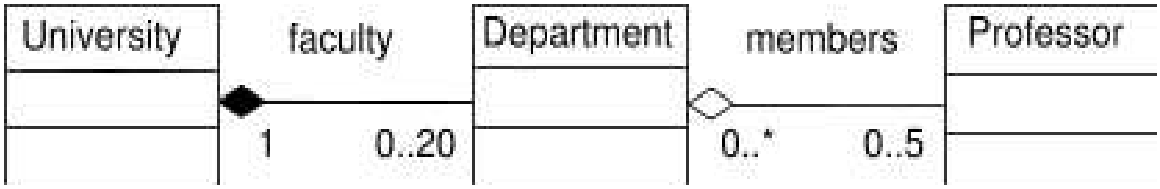



Στο άνω τμήμα αναγράφεται το όνομα της κλάσης ,στο μεσαίο τμήμα οι ιδιότητες ,δηλαδή καταστάσεις των αντικειμένων και στο κάτω τμήμα οι μέθοδοι της, δηλαδή οι λειτουργίες της κλάσης.

Οι συσχετίσεις(associations) βάσει των οποίων τα αντικείμενα διαφόρων κλάσεων μπορούν να αλληλεπιδρούν και να ανταλλάσσουν μηνύματα. Χωρίς συσχετίσεις υπάρχουν μόνο ανεξάρτητα αντικείμενα που δεν συνεργάζονται. Μια συσχέτιση συμβολίζεται σε ένα διάγραμμα κλάσεων ως μια συνεχής γραμμή που συνδέει τις κλάσεις που συμμετέχουν. Το όνομα της συσχέτισης τοποθετείται πλησίον της γραμμής, ενώ το όνομα ρολών των ακρών και η πολλαπλότητα στις άκρες της γραμμής.

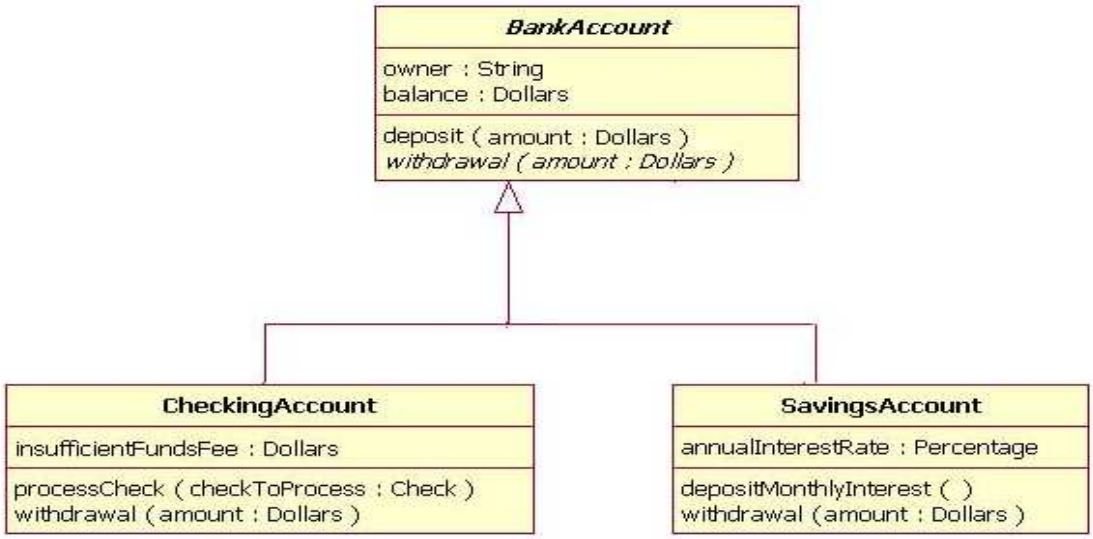
Η συσσωμάτωση(aggregation) είναι μία συσχέτιση ειδικής μορφής που αναπαριστά μία σχέση συνόλου-τμήματος ή όλου-τμήματος, δηλαδή ότι μια κλάση είναι τμήμα μιας άλλης κλάσης. Η δήλωση αυτής της σχέσης είναι μια γραμμή που καταλήγει σ'ένα λευκό-κενό ρόμβο, ο οποίος υποδεικνύει την ευρύτερη οντότητα της οποίας η συνδεδεμένη κλάση αποτελεί τμήμα. Η σύνθεση(composition) είναι μια ησυχότερης μορφής συσχέτιση, στην οποία το σύνολο έχει την αποκλειστική διαχείρισης των τμημάτων όπως η δημιουργία και η διαγραφή τους. Συμβολίζεται με ένα μαύρο κύκλο στην άκρη του συνόλου.

Παράδειγμα συσσωμάτωσης(aggregation) –σύνθεσης (composition):




Τέλος, η κληρονομικότητα(inheritance) είναι μια συσχέτιση μεταξύ μιας γενικής και μιας ειδικότερης περιγραφής που την επεκτείνει. Η ειδικότερη περιγραφή είναι απολύτως συνεπής με την γενική, δηλαδή έχει τις ίδιες ιδιότητες, λειτουργίες και σχέσεις αλλά μπορεί να περιλαμβάνει και επιπρόσθετη πληροφορία. Η κληρονομικότητα συμβολίζεται με ένα μικρό τρίγωνο βέλος στο άκρο που αντιστοιχεί στη γονική περιγραφή δηλαδή τη βασική κλάση γονέας. Χρησιμοποιείται κάθετη διάταξη, έτσι ώστε η βασική κλάση να βρίσκεται υψηλότερα από τις παράγωγες κλάσεις. Η κληρονομικότητα επιτρέπει πολυμορφική συμπεριφορά και λειτουργίες στο σύστημα.

Παράδειγμα κληρονομικότητας:




Στην παραπάνω εικόνα η οντότητα CheckingAccount και SavingsAccount ανήκουν στη γενικευμένη οντότητα BankAccount κληρονομώντας όλες τις ιδιότητες και τα χαρακτηριστικά της κλάσης BankAccount.

### 1.2 Τι είναι τα πρότυπα σχεδίασης

Τέλη της δεκαετίας του 1970, ο αρχιτέκτονας Κρίστοφερ Αλεξάντερ επιχείρησε να βρει και να καταγράψει αποδεδειγμένα ποιοτικούς σχεδιασμούς στον τομέα των κατασκευών. Έτσι μελέτησε μαζί με τους συνεργάτες του πολλές διαφορετικές κατασκευές που εξυπηρετούσαν τον ίδιο σκοπό και προσπάθησε να ανακαλύψει κοινά στοιχεία, τα οποία κατηγοριοποίησε σε

σχεδιαστικά πρότυπα. Το 1987 η ιδέα της εύρεσης σχεδιαστικών προτύπων εφαρμόστηκε για πρώτη φορά στη μηχανική λογισμικού και μέχρι τα μέσα της δεκαετίας του ’90 η εν λόγω έννοια καθιερώθηκε και εξαπλώθηκε προσανατολισμένη πλέον προς τον κόσμο της αντικειμενοστρέφειας.

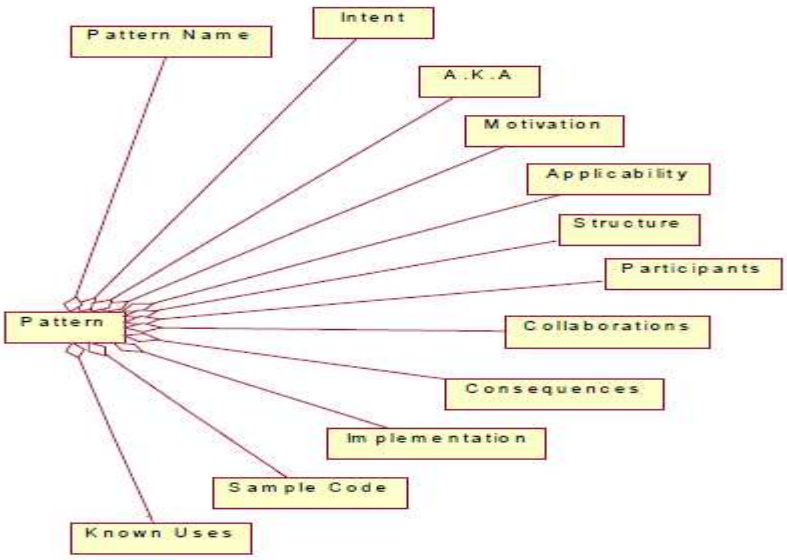
Ένα πρότυπο σχεδίασης ορίζεται ως μία αποδεδειγμένα καλή λύση που έχει εφαρμοστεί με επιτυχία στην επίλυση ενός επαναλαμβανόμενου προβλήματος σχεδίασης συστημάτων λογισμικού. Είναι μία περιγραφή για το πώς μπορεί να λυθεί ένα πρόβλημα που μπορεί να χρησιμοποιηθεί σε πολλές διαφορετικές καταστάσεις.

Ένα πρότυπο σχεδίασης εξετάζει αφαιρετικά και αναγνωρίζει τις κρίσιμες πλευρές μιας κοινής δομής σχεδιασμού που είναι χρήσιμη για τη δημιουργία επαναχρησιμοποιούμενου σχεδίου. Το πρότυπο σχεδίου αναγνωρίζει τις συμμετέχουσες κλάσεις και στιγμιότυπα ,τους ρόλους, τις συνεργασίες ,καθώς και την κατανομή ευθυνών.

Τα πρότυπα σχεδίασης ορίζονται τόσο σε επίπεδο μακροσκοπικής σχεδίασης όσο και σε επίπεδο υλοποίησης, ενώ με τη χρήση τους ένας προγραμματιστής αντικαθιστά πρακτικώς μεγάλα τμήματα του κώδικα με μαύρα κουτιά. Πρόκειται για αφαιρέσεις υψηλού επιπέδου που αποτελούν πλήρη υποσυστήματα, καταλλήλως ρυθμισμένα για την επίλυση συγκεκριμένων προβλημάτων σχεδίασης λογισμικού και έτοιμα για χρήση. Τα αντικειμενοστρεφή σχεδιαστικά πρότυπα παρουσιάζουν τις σχέσεις και τις αλληλεπιδράσεις μεταξύ των κλάσεων ή των αντικειμένων χωρίς διευκρίνιση των τελικών κλάσεων ή των αντικειμένων εφαρμογής που περιλαμβάνονται.

Το όνομα κάθε προτύπου διευκολύνει την επικοινωνία μεταξύ προγραμματιστών και την εύκολη αναφορά σε κοινά είδη προβλημάτων. Το πρόβλημα σε κάθε πρότυπο προσδιορίζει ένα γενικότερο πλαίσιο όπου υπό κανονική αντιμετώπιση θα προέκυπταν ανεπιθύμητες συνέπειες αναφορικά με τη λειτουργία, τον έλεγχο και τη συντήρηση του λογισμικού.

### 1.3 Περιγραφή ενός προτύπου σχεδίασης



Ένα πρότυπο σχεδίασης αποτελείται από τα πιο κάτω τμήματα:

**Pattern Name and Classification:** Ένα περιγραφικό και μοναδικό όνομα που βοηθά στον εντοπισμό και την παραπομπή στο πρότυπο.

**Intent:** Η περιγραφή του στόχου του προτύπου και του λόγου για τον οποίο χρησιμοποιείται.

**Also Known As:** ΄Αλλα ονόματα για το πρότυπο.

**Motivation:** Ένα σενάριο που αποτελείται από ένα πρόβλημα και ένα πλαίσιο στο οποίο αυτό το πρότυπο μπορεί να χρησιμοποιηθεί.

**Applicability:** Οι περιπτώσεις στις οποίες αυτό το πρότυπο μπορεί να χρησιμοποιηθεί.

**Structure:** Μια γραφική αναπαράσταση του προτύπου. Διαγράμματα κλάσεων και διαγράμματα αλληλεπίδρασης μπορεί να χρησιμοποιηθούν για το σκοπό αυτό.

**Participants:** Μια λίστα με τις κλάσεις και τα αντικείμενα που χρησιμοποιούνται στο πρότυπο και τους ρόλους τους στο σχεδιασμό.

**Collaboration:** Η περιγραφή του τρόπου με τον οποίο τα αντικείμενα και οι κλάσεις αλληλεπιδρούν μεταξύ τους.

**Consequences:** Μια περιγραφή των αποτελεσμάτων και των επιδράσεων που έχει η χρήση του προτύπου.

**Implementation:** Μια περιγραφή της υλοποίησης του προτύπου.

**Sample Code:** Ένα παράδειγμα για το πώς το σχέδιο μπορεί να χρησιμοποιηθεί σε μια γλώσσα προγραμματισμού.

**Known Uses:** Παραδείγματα πραγματικών χρήσεων του προτύπου.

Κάθε πρότυπο επιτρέπει να αλλάξει μια όψη από την αρχιτεκτονική δομή χωρίς να επηρεάζει τις υπόλοιπες. Έχουν οριστεί διάφορες κατηγορίες προτύπων, για διαφορετικά προβλήματα και κάθε κατηγορία περιλαμβάνει πολλαπλά στοιχεία. Συνήθως τα σχεδιαστικά πρότυπα κατηγοριοποιούνται σε κατασκευαστικά(creational) ,δομικά(structural) και συμπεριφορικά(behavioral).

Στα επόμενα 3 κεφάλαια θα μελετηθούν εκτενώς οι πιο πάνω κατηγορίες προτύπων με λεπτομερή περιγραφή των σπουδαιοτέρων κατασκευαστικών προτύπων.

# Κεφάλαιο 2

## Κατασκευαστικά πρότυπα


2.1 Εισαγωγή 7
2.2 Πρότυπο Factory method 7
2.3 Πρότυπο Singleton 9
2.4 Πρότυπο Prototype 10
2.5 Πρότυπο Builder 11
2.6 Πρότυπο Abstract Factory 12


### 2.1 Εισαγωγή

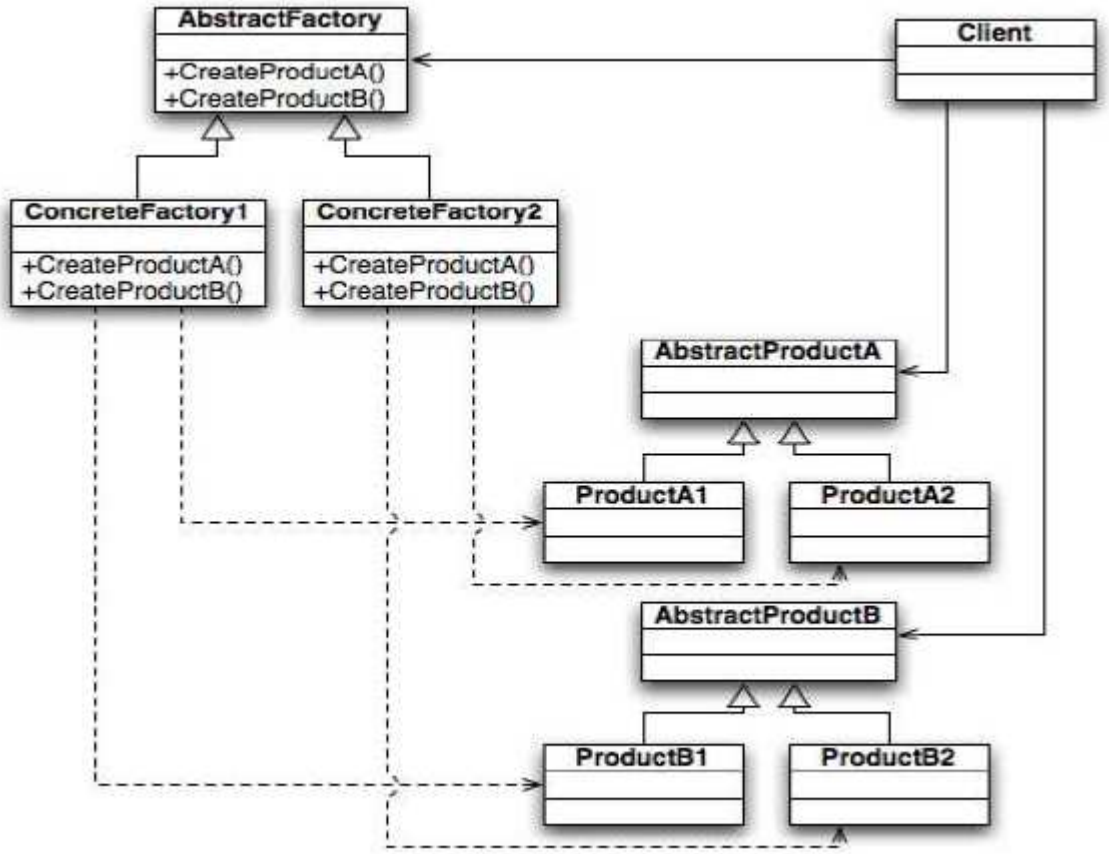
Τα κατασκευαστικά πρότυπα(creational patterns) αφορούν τυποποιημένους τρόπους δυναμικής κατασκευής αντικειμένων κατά το χρόνο εκτέλεσης. Απώτερος στόχος τους είναι η ανεξαρτητοποίηση του κώδικα που χρησιμοποιεί κάποια αντικείμενα από τις κλάσεις που ορίζουν τα αντικείμενα αυτά και τον τρόπο που κατασκευάζονται στη μνήμη, σύμφωνα με την αρχή ανοιχτότητας-κλειστότητας για ορθή αντικειμενοστρεφή σχεδίαση. Τέτοια πρότυπα είναι το Abstract Factory, το Builder, το Factory Method,το Lazy initialization, το Multiton, το Object Pool, το Prototype, το Singleton και το RAII(Resource Acquisition Is Initialization**).**

### 2.2 Πρότυπο Factory method

Το πρότυπο αυτό συγκεντρώνει σε μία κατάλληλη κλάση, τη Factory, όλη τη λειτουργικότητα κατασκευής στιγμιοτύπων μίας σειράς παραγόμενων κλάσεων που κληρονομούν κάποια κοινή υπερκλάση ή υλοποιούν την ίδια διασύνδεση. Οι μέθοδοι της κλάσης Factory, όταν καλούνται, κατασκευάζουν ένα νέο αντικείμενο κατάλληλου τύπου και

επιστρέφουν έναν αφηρημένο δείκτη προς αυτό, δηλαδή έναν δείκτη τύπος του οποίου είναι η γενική διασύνδεση, οπότε το εξωτερικό πρόγραμμα μπορεί να τις καλεί για να λαμβάνει το ζητούμενο κατά περίπτωση αντικείμενο χωρίς το ίδιο να χρειάζεται να γνωρίζει κάθε παραγόμενο τύπο δεδομένων που υλοποιεί τη διασύνδεση. Με αυτόν τον τρόπο το πρόγραμμα είναι κλειστό ως προς πιθανές επεκτάσεις και μόνο η κλάση Factory είναι ανοιχτή ως προς αυτές, καθώς μόνον ο δικός του κώδικας χρειάζεται να τροποποιηθεί σε περίπτωση π.χ. προσθήκης μίας νέας παραγόμενης κλάσης. Η κλάση Factory συνήθως παρέχει μία μέθοδο για κάθε δυνατό παραγόμενο τύπο, αλλά μία παραλλαγή με ονόματα για παραμετροποιημένη κλάση Factory η οποία περιέχει μία μοναδική μέθοδο η οποία επιλέγει το αντικείμενο που θα δημιουργήσει και θα επιστρέψει, αναλόγως με την τιμή ενός ορίσματος που δέχεται. Το όρισμα αυτό μπορεί π.χ. να διαβάζεται από κάποιο αρχείο ρυθμίσεων ή να μεταβιβάζεται ως όρισμα γραμμής εντολών στο εξωτερικό πρόγραμμα (στον πελάτη), έτσι ώστε το τελευταίο να είναι κλειστό ως προς το σύνολο των παραγόμενων κλάσεων και να μη χρειάζεται επαναμεταγλώττιση σε περίπτωση που τροποποιηθεί το σύνολο αυτό. Ένας εναλλακτικός τύπος Factory είναι όταν ορίζεται ως αφηρημένη κλάση και η παρεχόμενη μέθοδος κατασκευής αντικειμένων υποσκελίζεται από υποκλάσεις του, έτσι ώστε η καθεμία από τις τελευταίες να κατασκευάζει αντικείμενο διαφορετικού τύπου. Σε κάθε περίπτωση στόχος είναι να μπορεί το πρόγραμμα να δημιουργεί στιγμιότυπα κλάσεων χωρίς να προσδιορίζει ρητά τον ακριβή τύπο τους και αφήνοντας το Factory να τον αποφασίσει εσωτερικά· το πρόγραμμα αρκεί να έχει γνώση του γενικού αφηρημένου τύπου. Υπάρχει ωστόσο και μία εναλλακτική, αρκετά διαφορετική περίπτωση όπου, αντί το Factory να δηλώνεται ως κάποια/ες ξεχωριστή/ες κλάση/εις, αποτελείται απλώς από μία ή περισσότερες στατικές μεθόδους της κλάσης της οποίας πρέπει να κατασκευάζει αντικείμενα (έστω της κλάσης Α). Τότε ο κατασκευαστής (constructor) της κλάσης δηλώνεται ως ιδιωτική μέθοδος ώστε κάθε απόπειρα του εξωτερικού προγράμματος να δημιουργήσει στιγμιότυπα της Α να γίνεται αναγκαστικά μέσω των μεθόδων Factory οι οποίες επιστρέφουν ένα νέο αντικείμενο τύπου Α. Σε αυτήν την περίπτωση όμως η Α δεν μπορεί να κληρονομηθεί, αφού για τον σκοπό αυτό πρέπει να υπάρχει τουλάχιστον ένας δημόσιος κατασκευαστής της.

Γενική Δομή Factory method:



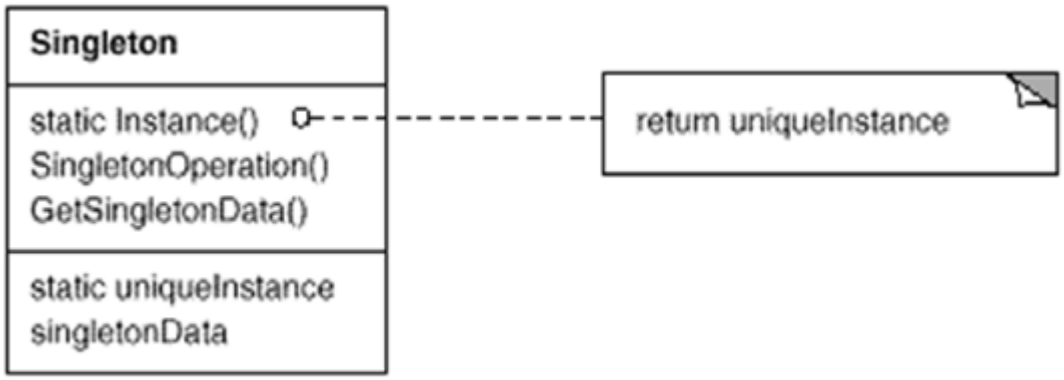

### 2.3 Πρότυπο Singleton

Το πρότυπο αυτό εξασφαλίζει ότι μια κλάση θα έχει μόνο ένα στιγμιότυπο και παρέχει ένα καθολικό σημείο πρόσβασης σ'αυτό. Χρησιμοποιείται όταν σε κάποιο σύστημα λογισμικού υπάρχει η απαίτηση από μία κλάση να δημιουργείται ένα και μόνο ένα αντικείμενο. Επίσης παρέχεται ένας συγκεκριμένος τρόπος για να μπορούμε να το ανακτούμε από οπουδήποτε. Το αντικείμενο αυτό δημιουργείται συνήθως κατά την έναρξη της εφαρμογής και διαγράφεται μετά το πέρας της.

Ο ρόλος αυτός του μοναδικού αντικειμένου είναι η διαχείριση των υπόλοιπων αντικειμένων της εφαρμογής και για το λόγο αυτό, αποτελεί λογικό σφάλμα να δημιουργηθούν περισσότερα του ενός τέτοια αντικείμενα-διαχειριστές. Σε μία τέτοια περίπτωση, η εφαρμογή θα έχει περισσότερα του ενός σημεία εκκίνησης, και ο υποθετικός χρήστης, ανάλογα με το σημείο εκκίνησης, μπορεί να καταλήξει να χρησιμοποιεί ένα υποσύνολο των αντικειμένων του συστήματος.

Επιπλέον, αν υπάρχουν περισσότεροι του ενός διαχειριστές, ενώ ο επιθυμητός στόχος είναι η ακολουθιακή εκτέλεση δραστηριοτήτων, πολλές δραστηριότητες θα εκτελούνται παράλληλα.
Το πρότυπο Singleton, εξασφαλίζει τη δημιουργία ενός και μόνο αντικειμένου, περιλαμβάνοντας μία ειδική μέθοδο κατασκευής στιγμιοτύπων. Όταν καλείται αυτή η μέθοδος, ελέγχει αν κάποιο αντικείμενο έχει δημιουργηθεί. Αν ναι, η μέθοδος επιστρέφει απλώς ένα δείκτη προς το υπάρχον αντικείμενο. Αν όχι, η μέθοδος δημιουργεί ένα νέο αντικείμενο και επιστέφει δείκτη προς αυτό. Για να εξασφαλιστεί ότι αυτός είναι και ο μοναδιαίος τρόπος δημιουργίας αντικειμένων από αυτή την κλάση, ο κατασκευαστής της κλάσης δηλώνεται ως προστατευμένος(PROTECTED) ή ιδιωτικός(PRIVATE). Με αυτό τον τρόπο δεν είναι δυνατό να δημιουργηθεί ένα αντικείμενο παρακάμπτοντας την παραπάνω ειδική μέθοδο.
Η απλότητα του προτύπου επιτρέπει οποιαδήποτε κλάση να μετατραπεί σε μοναδιαία, κάνοντας τον κατασκευαστή ιδιωτικό ή προστατευμένο και προσθέτοντας μια στατική μέθοδο και μια στατική ιδιότητα. Ένα πλεονέκτημα του προτύπου είναι ότι σε περίπτωση δημιουργίας παραγώγων κλάσεων από μια μοναδιαία κλάση, κάθε απόγονος μπορεί να είναι επίσης μοναδιαία κλάση αν προστεθούν σ'αυτές η στατική ιδιότητα και μέθοδος.

Γενική Δομή Singleton:



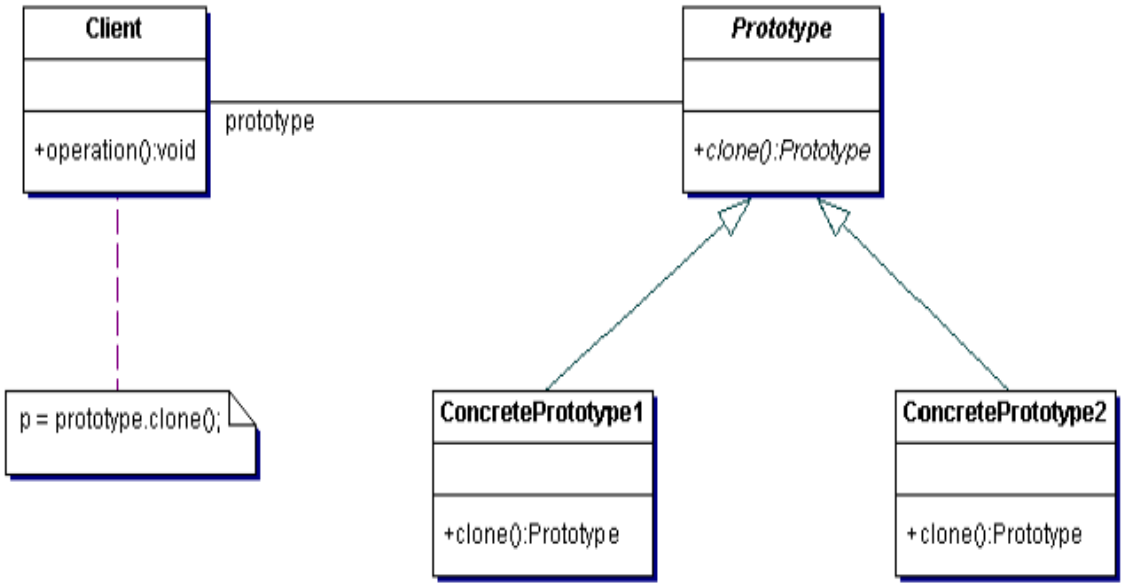

**2.4 Πρότυπο Prototype**

Το πρότυπο αυτό προσδιορίζει τους τύπους των αντικειμένων, δημιουργεί στιγμιότυπα χρησιμοποιώντας πρωτότυπα με σκοπό να δημιουργεί νέα αντικείμενα αντιγράφοντας αυτά τα πρότυπα.
Το Prototype είναι ένα σχεδιαστικό πρότυπο με το οποίο ένα νέο αντικείμενο κατασκευάζεται με "κλωνοποίηση" κάποιου υπάρχοντος. Η κλωνοποίηση αυτή γίνεται μέσω μίας μεθόδου clone η οποία παρέχεται από μία αφηρημένη κλάση ή διασύνδεση A και υλοποιείται σε κάθε παραγόμενη κλάση B η οποία κληρονομεί την A. Έτσι η κλήση της clone σε ένα στιγμιότυπο της B επιστρέφει ένα αντίγραφο του εν λόγω στιγμιότυπου, το

οποίο αναλόγως με την υλοποίηση μπορεί να είναι είτε ρηχό, δηλαδή να περιέχει δείκτες προς τις εσωτερικές δομές δεδομένων του αρχικού στιγμιότυπου, είτε βαθύ, δηλαδή να περιέχει πλήρη, νεοδημιουργηθέντα αντίγραφα αυτών των δομών δεδομένων.

Το Prototype χρειάζεται σε περιπτώσεις όπου πρέπει να κατασκευαστεί μία ρέπλικα ενός αντικειμένου αλλά με κάποιον άλλον τρόπο, π.χ. στην Java με χρήση του τελεστή new και ενός κατασκευαστή αντιγράφου (copy constructor), προσβάλλεται η αρχή ανοιχτότητας-κλειστότητας. Η μέθοδος clone μπορεί να επικαλύπτει την κλήση του εκάστοτε κατασκευαστή αντιγράφου με ένα κοινό επίπεδο αφαίρεσης έτσι ώστε να μη χρειάζεται το εξωτερικό πρόγραμμα να γνωρίζει όλους τους παραγόμενους τύπους δεδομένων που υλοποιούν τη διασύνδεση A, καθώς η clone επιστρέφει αναφορά του αφηρημένου τύπου A. Με τη χρήση αυτού του πρωτοτύπου πετυχαίνουμε μείωση του αριθμού των κλάσεων. Είναι πιο ωφέλιμο να εγκατασταθεί ένας αριθμός προτύπων και να δημιουργηθούν κλώνοι από το να δημιουργούνται κλάσεις χειροκίνητα κάθε φορά για συγκεκριμένη κατάσταση.

Γενική Δομή Prototype:

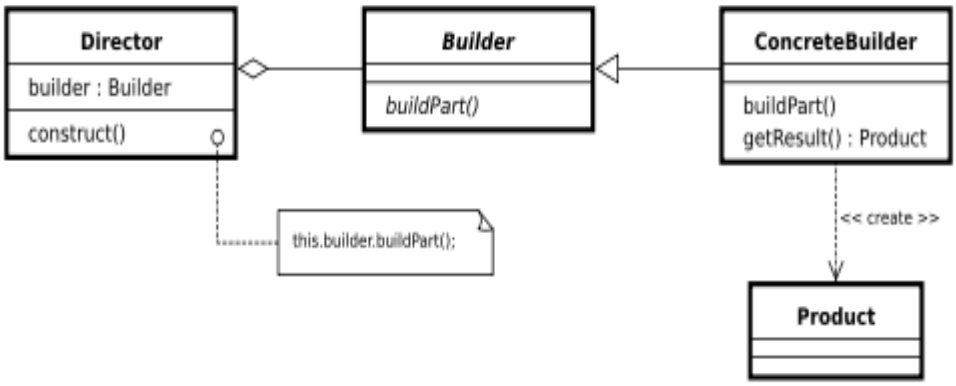



## 2.5 Πρότυπο Builder

Το πρότυπο αυτό χρησιμοποιείται για τη δημιουργία αντικειμένων. Στοχεύει στο να αφαιρέσει τα στάδια κατασκευής των αντικειμένων έτσι ώστε από διαφορετικές υλοποιήσεις από αυτά τα βήματα να μπορούν να κατασκευαστούν διαφορετικές αναπαραστάσεις των αντικειμένων.

Χρησιμοποιώντας αυτό το πρότυπο, η διαδικασία της κατασκευής ενός τέτοιου αντικειμένου μπορεί να είναι πιο αποτελεσματική. Το πρότυπο προτείνει τη μετακίνηση από τη λογική κατασκευής της κλάσης του αντικειμένου σε μια ξεχωριστή κλάση που αναφέρεται ως Builder class. Μπορούν να υπάρξουν περισσότερες από μία τέτοιες κλάσεις, η κάθε μία με διαφορετική υλοποίηση για τη σειρά των βημάτων για την κατασκευή του αντικειμένου.
Αυτός ο τύπος διαχωρισμού μειώνει το μέγεθος του αντικειμένου. Επιπλέον, ο σχεδιασμός αποδεικνύεται ότι είναι πιο σπονδυλωτός για κάθε υλοποίηση που περιλαμβάνεται σε ένα διαφορετικό αντικείμενο. Είναι πιο εύκολο να προσθέσουμε μία νέα υλοποίηση ενώ η διαδικασία κατασκευής αντικειμένων γίνεται ανεξάρτητα από τα συστατικά που αποτελούν το αντικείμενο. Αυτό παρέχει περισσότερο έλεγχο πάνω στη διαδικασία κατασκευής αντικειμένων.
΄Οσον αφορά την υλοποίηση, καθένα από τα διάφορα στάδια της διαδικασίας κατασκευής μπορεί να δηλωθεί ως μέθοδος μιας κοινής διεπαφής που υλοποιείται από διαφορετικούς κατασκευαστές.

Γενική Δομή Builder:

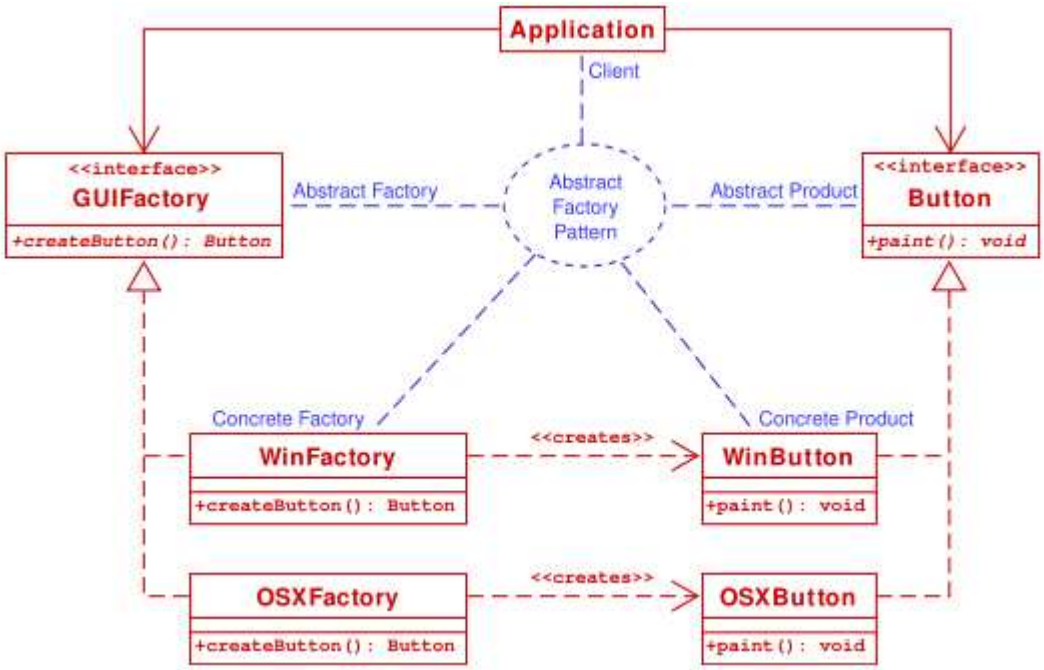



## 2.6 Πρότυπο Abstract Factory

Το Abstract Factory είναι ένα πρότυπο που αποτελεί παραλλαγή του Factory και χρησιμοποιείται όταν έχουμε παράλληλες κληρονομικές ιεραρχίες κλάσεων οι οποίες μοιράζονται κάποια κοινή ιδιότητα. Έστω π.χ. ότι γράφουμε μία βιβλιοθήκη για κατασκευή γραφικών διασυνδέσεων χρήστη (GUI) η οποία περιέχει διάφορα οπτικά στοιχεία (widgets) τα οποία κληρονομούν από μία κοινή διασύνδεση ονόματι Widget (π.χ. κουμπιά, μπάρες κύλισης κλπ). Υποθέτοντας ότι η βιβλιοθήκη υλοποιείται σε δύο εκδοχές, μία για λειτουργι-

κό σύστημα Windows και μία για Unix, μπορούμε να δηλώσουμε ένα αφηρημένο Factory το οποίο παρέχει μεθόδους για την κατασκευή κάθε πιθανού οπτικού στοιχείου ώστε αυτές οι μέθοδοι να υποσκελίζονται διαφορετικά στις δύο παραγόμενες κλάσεις που υλοποιούν το αφηρημένο Factory: μία για Windows και μία για Unix.

Γενική Δομή Abstract Factory:



# Κεφάλαιο 3

## Δομικά πρότυπα


3.1 Εισαγωγή 14
3.2 Πρότυπο Adapter 14
3.3 Πρότυπο Bridge 16
3.4 Πρότυπο Composite 17
3.5 Πρότυπο Decorator 18
3.6 Πρότυπο Proxy 20


### 3.1 Εισαγωγή

Τα δομικά πρότυπα(structural patterns) αφορούν τυποποιημένους τρόπους δυναμικής κατασκευής σύνθετων αντικειμένων τα οποία χρησιμοποιούν υπάρχουσες ιεραρχίες κλάσεων. Κρατάνε τις συζεύξεις χαμηλές, τις κλάσεις αμετάβλητες, και προσφέρουν εναλλακτικές λύσεις στην κληρονομικότητα. Τέτοια πρότυπα είναι το Adapter, το Bridge, το Composite, το Decorator, το Facade, το Front Controller, το Flyweight, το Proxy και το Module.

### 3.2 Πρότυπο Adapter

Το πρότυπο αυτό έχει ως στόχο τη μετατροπή της διασύνδεσης μιας κλάσης σε μια άλλη που αναμένεται από ένα εξωτερικό πρόγραμμα. Το πρότυπο επιτρέπει τη συνεργασία κλάσεων, η οποία σε διαφορετική περίπτωση θα ήταν αδύνατη λόγω ασύμβατων διασυνδέσεων. Χρησιμοποιούμε αυτό το πρότυπο όταν θέλουμε ανεξάρτητες κλάσεις να λειτουργούν μαζί

σε ένα ενιαίο πρόγραμμα. Γράφουμε μια κλάση που έχει την επιθυμητή διεπαφή και την κάνουμε έπειτα να επικοινωνήσει με την κλάση που έχει μια διαφορετική διεπαφή.

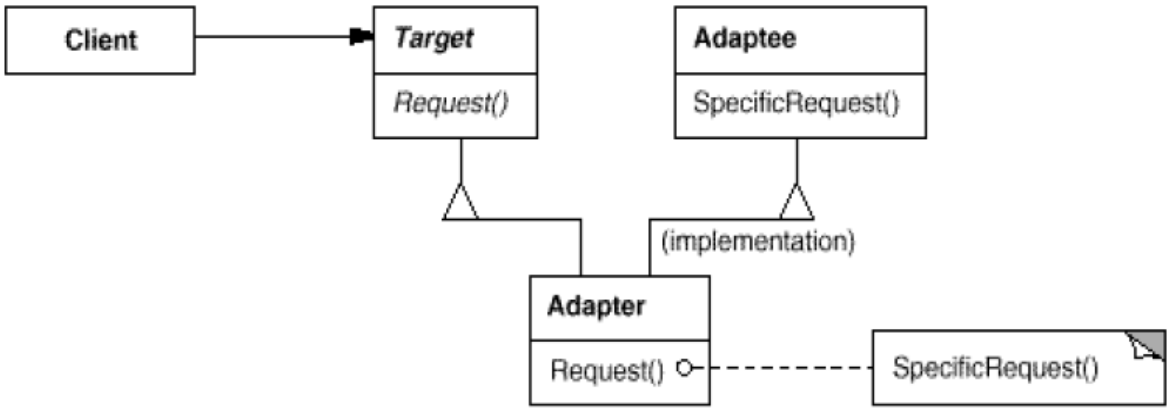
Το πρόβλημα που επιλύει το εν λόγω σχεδιαστικό πρότυπο εν γένει προκύπτει όταν μία υπάρχουσα ιεραρχία κλάσεων, οι οποίες υλοποιούν ένα συγκεκριμένο στοιχείο αφαίρεσης, πρέπει να επεκταθεί με την προσθήκη μίας νέας κλάσης η διασύνδεση της οποίας δεν είναι συμβατή με το υπάρχον στοιχείο αφαίρεσης.

Το πρότυπο αυτό επιτρέπει την προσαρμογή της διασύνδεσης που εξάγει μία κλάση Α σε μία πρότυπη διασύνδεση, έστω Interface, που αναμένεται από ένα εξωτερικό πρόγραμμα ώστε να μην παραβιάζεται η αρχή ανοιχτότητας-κλειστότητας. Αυτό γίνεται μέσω μίας ενδιάμεσης κλάσης, του Adapter, η οποία υλοποιεί το Interface και ταυτοχρόνως είτε κληρονομεί την Α (υλοποίηση με κληρονομικότητα), είτε περιέχει μία ιδιωτική αναφορά σε αντικείμενο τύπου Α (υλοποίηση με συνάθροιση). Σε κάθε περίπτωση ο Adapter έχει πρόσβαση στις μεθόδους της Α και οι δικές του μέθοδοι, οι υπογραφές των οποίων συμφωνούν με αυτές που αναμένονται από κάποιον πελάτη καθώς προδιαγράφονται από τη διασύνδεση / αφηρημένη κλάση Interface, τις καλούν εσωτερικά και επιστρέφουν τα αποτελέσματα, αποκρύπτοντας παράλληλα τη διαδικασία αυτή από το εκάστοτε πρόγραμμα πελάτη.

Η υλοποίηση του προτύπου Adapter μέσω κληρονομικότητας επιτρέπει την προσαρμογή στη ζητούμενη διασύνδεση και των προστατευμένων μεθόδων της κλάσης Α, αντί μόνο των δημοσίων της, ενώ είναι και ελαφρώς πιο αποδοτική από την υλοποίηση με συνάθροιση, αφού κάθε κλήση στον Adapter «μεταφράζεται» σε κλήση σε μια άλλη μέθοδο του ιδίου αντί για κλήση σε μέθοδο άλλου αντικειμένου. Από την άλλη η υλοποίηση με συνάθροιση δεν αντιμετωπίζει προβλήματα ακόμα και αν η Α δεν μπορεί να κληρονομηθεί, ενώ επιτρέπει την προσαρμογή όχι μόνο της Α αλλά και κάθε υποκλάσης της λόγω του πολυμορφισμού.

Συνήθως το πρότυπο αυτό χρησιμοποιείται όταν θέλουμε να χρησιμοποιήσουμε μια υπάρχουσα κλάση, αλλά η διασύνδεσή της δεν συμβαδίζει με τις ανάγκες μας. Ένας προσαρμογέας αντικειμένου(object adapter) βασίζεται στη σύνθεση αντικειμένων και στη διαβίβαση μηνυμάτων(delegation). Ένας προσαρμογέας κλάσης(class adapter) χρησιμοποιεί πολλαπλή κληρονομικότητα για να προσαρμόσει μια διασύνδεση σε μια άλλη.

Γενική Δομή Adapter:



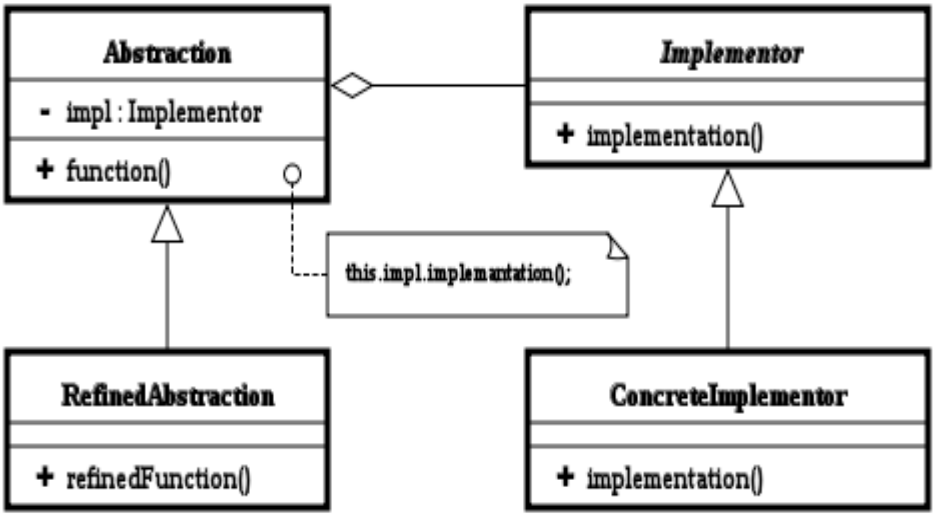

### 3.3 Πρότυπο Bridge

Σε περιπτώσεις που έχουμε ένα στοιχείο αφαίρεσης Α (π.χ. αφηρημένη κλάση ή διασύνδεση) και κάποιες κλάσεις που το υλοποιούν, είναι εύκολη η επέκταση της εν λόγω ιεραρχίας με ακόμα μία παραγόμενη κλάση χωρίς ιδιαίτερο κόπο· αρκεί το εξωτερικό πρόγραμμα να χρησιμοποιεί αναφορές προς τον αφηρημένο τύπο Α και όχι προς τους επιμέρους εξειδικευμένους παραγόμενους τύπους (αρχή ανοιχτότητας-κλειστότητας και αξιοποίηση του πολυμορφισμού). Όμως δεν είναι εύκολη η επέκταση της ίδιας της διασύνδεσης Α με νέες μεθόδους, ιδιότητες κλπ., έστω με ένα καινούργιο στοιχείο αφαίρεσης Β το οποίο κληρονομεί το Α, καθώς οι πιθανές κλάσεις που θα υλοποιούν το Β δεν είναι εύκολο να εκμεταλλευτούν τον υπάρχοντα κώδικα αντίστοιχων κλάσεων οι οποίες υλοποιούν το παλαιό στοιχείο αφαίρεσης Α. Θα μπορούσε βέβαια αυτό να γίνει με πολλαπλή κληρονομικότητα αλλά η εν λόγω σχεδίαση δεν είναι ευέλικτη και δεν υποστηρίζεται πάντοτε.

Τη λύση δίνει το πρότυπο Bridge, το οποίο χρησιμοποιείται όταν το υπό ανάπτυξη σύστημα περιέχει στοιχεία αφαίρεσης από τα οποία μπορούν να προκύψουν διαφορετικές υλοποιήσεις αλλά και νέα στοιχεία αφαίρεσης, που επίσης μπορούν να υλοποιηθούν σε διάφορες παραγόμενες κλάσεις. Η εφαρμογή του προτύπου έγκειται στη σχεδίαση δύο ιεραρχιών κληρονομικότητας, μίας ιεραρχίας στοιχείων αφαίρεσης (υψηλού επιπέδου) και μίας ιεραρχίας υλοποιήσεων (χαμηλού επιπέδου). Οι κλάσεις χαμηλού επιπέδου χρησιμοποιούνται για την οικοδόμηση των κλάσεων υψηλού επιπέδου μέσω συνάθροισης, καθώς το βασικό στοιχείο αφαίρεσης περιέχει μία αναφορά σε γενικού τύπου στιγμιότυπο του βασικού στοιχείου της ιεραρχίας χαμηλού επιπέδου. Έτσι οποιαδήποτε κλάση της ιεραρχίας υψηλού επιπέδου μπορεί να κάνει χρήση αντικειμένων οποιασδήποτε κλάσης της ιεραρχίας χαμηλού επιπέδου και αλλαγές μπορούν να γίνουν κατά τον χρόνο εκτέλεσης. Ας σημειωθεί βέβαια ότι οι μέθοδοι των κλάσεων υψηλού επιπέδου μπορούν να κάνουν χρήση

μόνο των μεθόδων του βασικού στοιχείου της ιεραρχίας χαμηλού επιπέδου οι οποίες υποσκελίζονται στις επιμέρους υλοποιήσεις.

Γενική Δομή Bridge:




### 3.4 Πρότυπο Composite

Το πρότυπο αυτό επιτρέπει τη σύνθεση αντικειμένων σε δενδροειδείς δομές για την αναπαράσταση ιεραρχιών τμήματος όλου. Επιτρέπει στα προγράμματα πελάτες να διαχειρίζονται με ενιαίο τρόπο τόσο τα ανεξάρτητα αντικείμενα όσο και συνθέσεις αντικειμένων.

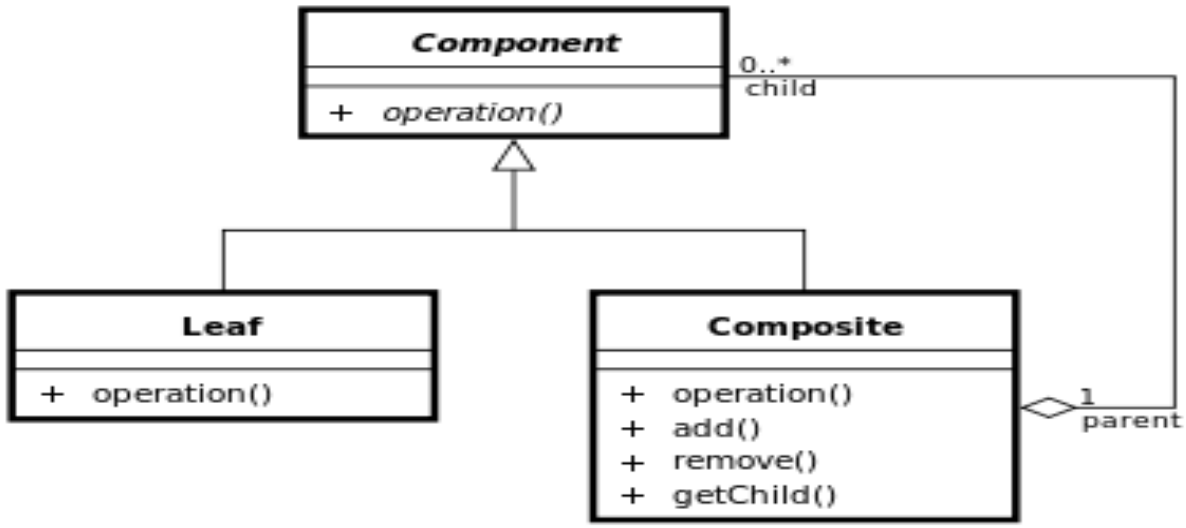
Το πρότυπο δίνει λύσεις με κομψό τρόπο σε προβλήματα χρήσης σχέσεων περιεκτικότητας μεταξύ κλάσης που αντιπροσωπεύει το όλον(περικλείουσα κλάση) και των κλάσεων που αντιπροσωπεύουν τα τμήματα.

Το σημείο κλειδί σ'αυτό είναι η ύπαρξη μιας αφηρημένης κλάσης που αναπαριστά τόσο τις πρωταρχικές κλάσεις όσο και τις περικλείουσες κλάσεις. Έτσι, είναι πλέον δυνατή η δημιουργία οποιουδήποτε πρωταρχικού ή σύνθετου αντικειμένου επιτρέποντας ομοιόμορφο χειρισμό των αντικειμένων από ένα πρόγραμμα πελάτη.

Ο χρήστης είναι σε θέση να δημιουργήσει οποιαδήποτε σύνθετη οντότητα και να την προσθέσει στην εφαρμογή. Η σχεδίαση ενός σύνθετου αντικειμένου ουσιαστικά συνιστάται στη σχεδίαση των επιμέρους τμημάτων του. Για το λόγο αυτό, είναι επιθυμητή η ενιαία αντιμετώπιση όλων των αντικειμένων. Το πρότυπο επιτρέπει τον αναδρομικό ορισμό

περιεκτικότητας, ώστε οι πελάτες να μην αντιλαμβάνονται τη διαφορά μεταξύ πρωταρχικών και σύνθετων αντικειμένων.

Γενική Δομή Composite:




### 3.5 Πρότυπο Decorator

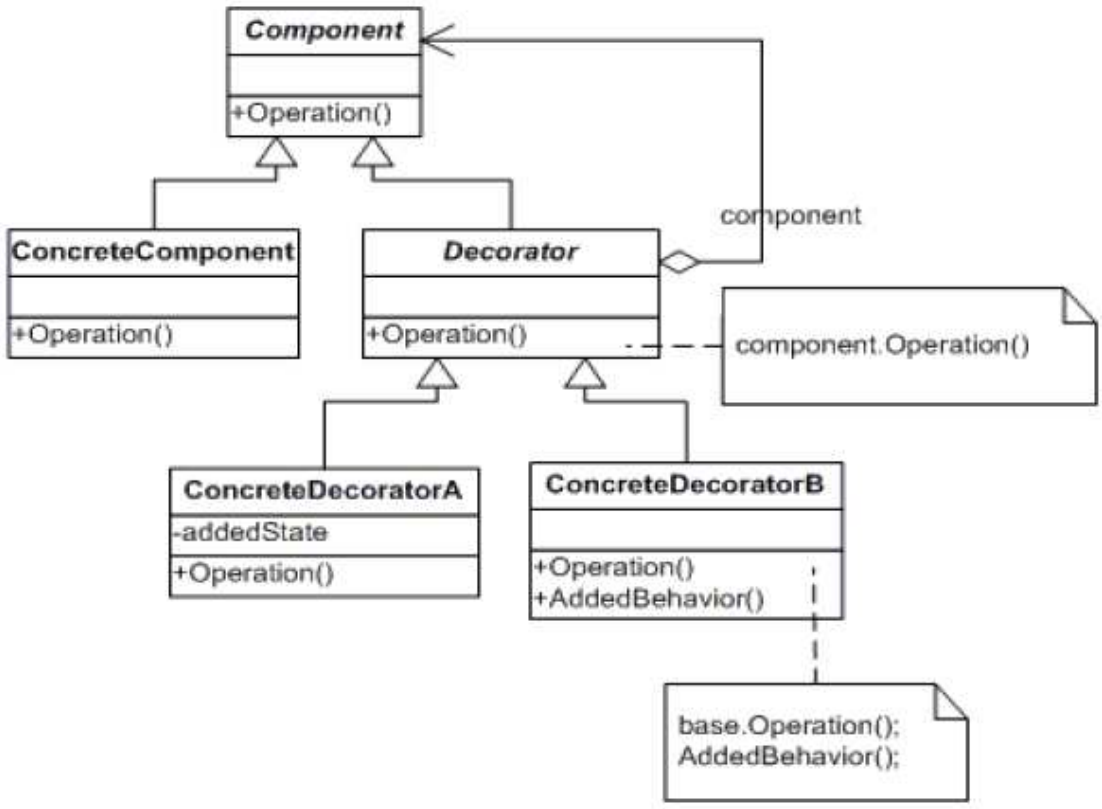
Το πρότυπο αυτό επιτρέπει την εύκολη και δυναμική επέκταση της λειτουργικότητας κάποιων υπαρχόντων κλάσεων Α, Β κλπ, οι οποίες υλοποιούν την ίδια διασύνδεση ή κληρονομούν την ίδια αφηρημένη κλάση (έστω Interface), σε χρόνο εκτέλεσης. Επισυνάπτει επιπρόσθετες ευθύνες σε αντικείμενα δυναμικά και παρέχει εναλλακτική ευελιξία σε υποκλάσεις για περαιτέρω λειτουργικότητες, χωρίς να επηρεάζονται τα άλλα αντικείμενα. Αυτό γίνεται μέσω του Decorator, μίας νέας κλάσης η οποία επίσης υλοποιεί την Interface αλλά περιέχει ως ιδιωτικό πεδίο και μία αναφορά σε ένα στιγμιότυπο του γενικού τύπου Interface (έστω το instance), η οποία τυπικά μεταβιβάζεται ως όρισμα στον κατασκευαστή της Decorator. Έτσι οι μέθοδοι της τελευταίας υλοποιούν εσωτερικά την καινούργια λειτουργικότητα αλλά για τις κοινές εργασίες καλούν τις αντίστοιχες μεθόδους του instance. Κατά τον χρόνο εκτέλεσης θα μπορούσε το αντικείμενο Decorator να κατασκευάζεται με όρισμα οποιοδήποτε στιγμιότυπο τύπου Interface (ακόμα και του ίδιου του Decorator, αν και αυτό δε θα είχε ιδιαίτερο νόημα) ώστε κατά περίπτωση το αντικείμενο να παρέχει τη λειτουργικότητα οποιασδήποτε κλάσης τύπου Interface, είτε της Α είτε κάποιας άλλης, επεκταμένης με ένα συγκεκριμένο σύνολο δυνατοτήτων. Με αυτόν τον τρόπο γίνεται εφικτός

ένας δυναμικός συνδυασμός λειτουργιών από στοιχειώδεις δομικούς λίθους κατά τον χρόνο εκτέλεσης.

Η εναλλακτική λύση, χωρίς χρήση κάποιου σχεδιαστικού προτύπου, θα ήταν η απλή κληρονομικότητα, με τον ορισμό κλάσεων οι οποίες επεκτείνουν τις Α, Β κλπ. και προσθέτουν τη νέα λειτουργικότητα. Ωστόσο η λύση αυτή δεν είναι εφικτή σε περίπτωση που οι Α, Β κλπ. δεν μπορούν να επεκταθούν με κληρονομικότητα (π.χ. αν δηλωθούν ως τελικές κλάσεις στην Java), ενώ σε άλλες περιπτώσεις δεν είναι καθόλου πρακτική, π.χ. αν έχουμε πολλαπλά διαφορετικά σύνολα νέων δυνατοτήτων τα οποία πρέπει να συνδυαστούν με τις Α, Β κλπ. Το πρόβλημα έγκειται στο ότι με την κληρονομικότητα όλοι οι πιθανοί συνδυασμοί δυνατοτήτων πρέπει να προβλεφθούν και να ληφθούν υπ' όψιν κατά τη συγγραφή του προγράμματος. Αντιθέτως με την κλάση Decorator, η οποία δρα ως περίβλημα (wrapper) άλλων αντικειμένων τύπου Interface προς τα οποία περιέχει αναφορές / δείκτες, η σύνθεση νέων αντικειμένων ουσιαστικά γίνεται δυναμικά ενώ το πρόγραμμα εκτελείται.

Το πρότυπο αυτό είναι ωφέλιμο σε περιπτώσεις προέκτασης υποκλάσεων που είναι μη πρακτικές. Έτσι αποφεύγεται πιθανός πολλαπλασιασμός υποκλάσεων.

Γενική Δομή Decorator:

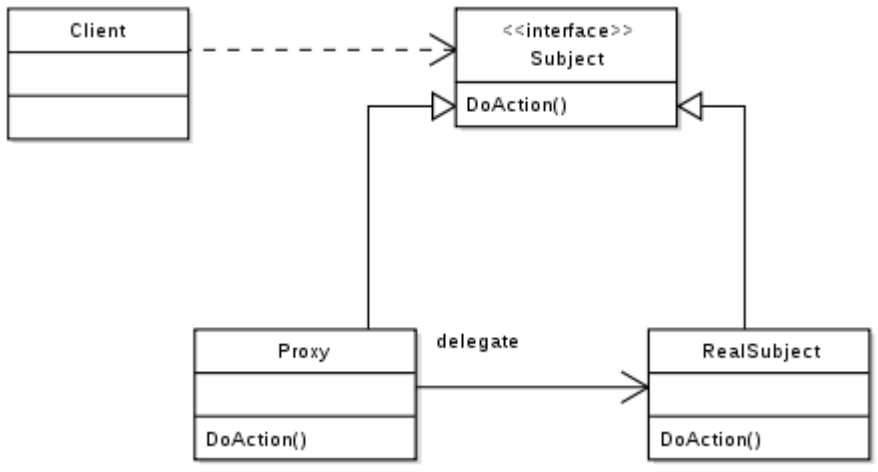


### 3.6 Πρότυπο Proxy

Όταν η πρόσβαση ενός εξωτερικού προγράμματος σε αντικείμενα κάποιας συγκεκριμένης κλάσης (έστω της *A*) πρέπει να είναι ελεγχόμενη και να πληροί ορισμένες προϋποθέσεις (π.χ., για λόγους εξοικονόμησης μνήμης, να μη φορτώνεται πραγματικά η Α μέχρι να κληθεί μία μέθοδος της), το πρότυπο Proxy παρέχει έναν τυποποιημένο τρόπο ώστε αυτό να γίνεται διατηρώντας την κλειστότητα του εξωτερικού προγράμματος ως προς την εν λόγω κλάση και τον τρόπο λειτουργίας της· ακόμα και αν η υλοποίηση της αλλάξει ο πελάτης είναι αδιάφορος απέναντι σε αυτές τις αλλαγές (δε χρειάζεται τροποποίηση) χάρη σε ένα επίπεδο αφαίρεσης που του παρέχεται από μία ενδιάμεση κλάση, τον Proxy, η οποία παίζει το ρόλο του μεσολαβητή πρόσβασης. Ο Proxy είναι που εκτελεί όλους τους απαραίτητους ελέγχους και προσπελαύνει πραγματικά τα αντικείμενα, ενώ ταυτόχρονα υλοποιεί την ίδια διασύνδεση με την Α κι έτσι ο πελάτης, ο οποίος κατέχει μία αναφορά προς στιγμιότυπο του Proxy αντί της Α, δεν αντιλαμβάνεται καν τη μεσολάβησή του· ο Proxy ουσιαστικώς εικονικοποιεί την πρόσβαση στη ζητούμενη κλάση. Συνήθως κατασκευάζεται, αντί για την Α, από ένα Factory και εντελώς διαφανώς για το εξωτερικό πρόγραμμα, ενώ δεν είναι σπάνιο να περιέχει μία αναφορά στο πραγματικό στιγμιότυπο της Α ως ιδιωτικό πεδίο.

Το σχεδιαστικό αυτό πρότυπο αποτελεί ειδική εφαρμογή της γενικότερης έννοιας του proxy στην πληροφορική. Με αυτήν τη γενικότερη έννοια ο proxy μπορεί να μεσολαβεί μεταξύ ενός πελάτη και ενός οποιουδήποτε ακριβού, σπάνιου ή περίπλοκου πόρου. Μία ιδιαίτερη περίπτωση εφαρμογής της έννοιας του proxy αποτελεί ο απομακρυσμένος proxy, όπως τα στελέχη (stubs) πελάτη και διακομιστή στο υπόδειγμα απομακρυσμένης κλήσης διαδικασιών στα συστήματα ενδιάμεσου λογισμικού.

Γενική Δομή Proxy:



# Κεφάλαιο 4

## Συμπεριφορικά πρότυπα


4.1 Εισαγωγή 21
4.2 Πρότυπο Observer 22
4.3 Πρότυπο State 23
4.4 Πρότυπο Strategy 24
4.5 Πρότυπο Template method 25
4.6 Πρότυπο Visitor 26


### 4.1 Εισαγωγή

Τα συμπεριφορικά πρότυπα αφορούν τον καταμερισμό των αρμοδιοτήτων σε διάφορες κλάσεις και τον ορισμό του τρόπου επικοινωνίας μεταξύ των αντικειμένων τους κατά τον χρόνο εκτέλεσης. Σε αντίθεση με τα δομικά πρότυπα, τα συμπεριφορικά βρίσκουν εφαρμογή στον αρχικό σχεδιασμό μίας ιεραρχίας κλάσεων και όχι στην εκ των υστέρων επέκταση κάποιας υπάρχουσας ιεραρχίας. Ο γενικός κανόνας είναι ότι έχουμε να κάνουμε με μία ομάδα κλάσεων οι οποίες υλοποιούν με διαφορετικούς τρόπους μία κοινή διασύνδεση Α· ως συνήθως το εξωτερικό πρόγραμμα είναι προτιμότερο να χειρίζεται μόνον αναφορές του αφηρημένου τύπου Α και να βασίζεται στον πολυμορφισμό ώστε να μην παραβιάζεται η αρχή ανοιχτότητας-κλειστότητας. Η επικοινωνία μεταξύ αντικειμένων γίνεται με παρόμοιο τρόπο, καθώς όχι σπάνια μία κλήση μεθόδου (ίσως κλήση κατασκευαστή) δέχεται ως όρισμα μία αναφορά αφηρημένου τύπου κι έτσι το αντίστοιχο αντικείμενο μπορεί να προσπελαύνει με τυποποιημένο τρόπο δημόσιες μεθόδους στιγμιότυπων κάθε κλάσης της επίμαχης ιεραρχίας. Τέτοια πρότυπα είναι το Command, το Interpreter, το Iterator, το Mediator, το Observer, το State, το Strategy, το Template method και το Visitor.

### 4.2 Πρότυπο Observer

Το πρότυπο αυτό ορίζει μια σχέση εξάρτησης ένα-προς-πολλά(1:N) μεταξύ των αντικειμένων έτσι ώστε όταν μεταβάλλεται η κατάσταση ενός αντικειμένου, όλα τα εξαρτώμενα αντικείμενα να ενημερώνονται και να τροποποιούνται αυτόματα. Επιδιώκει να μειώσει τη σύζευξη μεταξύ των αντικειμένων, παρέχοντας αυξημένη δυνατότητα επαναχρησιμοποίησης και τροποποίησης του συστήματος. Το συγκεκριμένο πρότυπο, επιτρέπει την αυτόματη ειδοποίηση και ενημέρωση ενός συνόλου αντικειμένων τα οποία αναμένουν ένα γεγονός, που εκδηλώνεται ως αλλαγή στην κατάσταση ενός αντικειμένου.

Ο στόχος είναι η απόζευξη των παρατηρητών από το παρακολουθούμενο αντικείμενο(υποκείμενο), έτσι ώστε κάθε φορά που προστίθεται ένας νέος παρατηρητής(με διαφορετική διασύνδεση ενδεχομένως), να μην απαιτούνται αλλαγές στο παρατηρούμενο αντικείμενο.

Η χρήση του προτύπου είναι ωφέλιμη όταν η λίστα των παρατηρητών μπορεί να αλλάζει κατά τη διάρκεια εκτέλεσης του προγράμματος, ή όταν η αλλαγή της κατάστασης ενός αντικειμένου απαιτεί απαιτεί την ειδοποίηση άλλων αντικειμένων(παρατηρητών), αλλά το αντικείμενο δεν θα πρέπει να κάμει καμιά υπόθεση για το ποια είναι αυτά τα αντικείμενα.

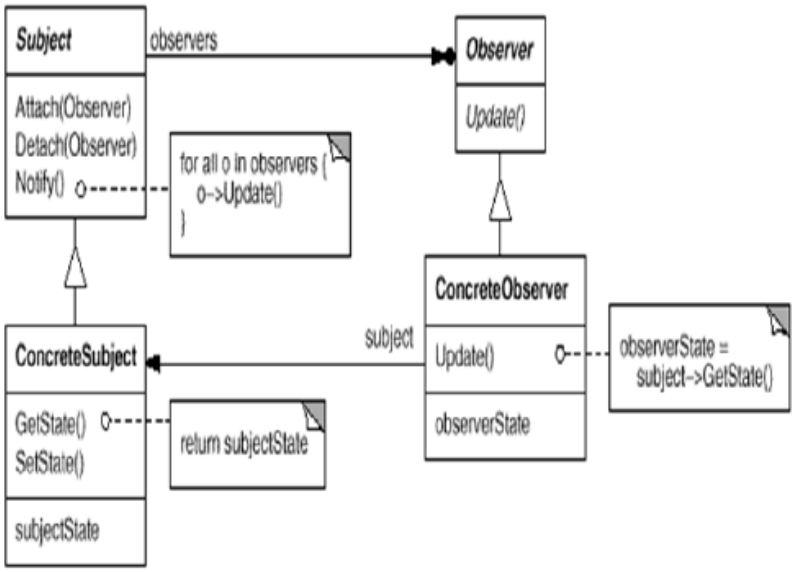
Η χρήση αυτού του προτύπου επιβάλλεται μόνο σε κατάλληλες περιπτώσεις, αλλιώς προσθέτει περιττή πολυπλοκότητα στο σύστημα.

Η εφαρμογή αυτού του προτύπου προϋποθέτει τον εντοπισμό των εξής δύο τμημάτων: ενός υποκειμένου και του παρατηρητή. Μεταξύ των δύο υφίσταται μία σχέση ένα-προς-πολλά(1:N). Το υποκείμενο θεωρείται ότι διατηρεί το μοντέλο των δεδομένων και η λειτουργικότητα που αφορά στην παρατήρηση των δεδομένων κατανέμεται σε διακριτά αντικείμενα-παρατηρητές. Οι παρατηρητές καταχωρούνται στο υποκείμενο κατά τη δημιουργία τους. Όταν το υποκείμενο αλλάζει, ανακοινώνει προς όλους τους καταχωρημένους παρατηρητές το γεγονός της αλλαγής, και κάθε παρατηρητής ρωτά το υποκείμενο για το σύνολο της κατάστασης του υποκειμένου που το ενδιαφέρει.

Το πρότυπο εφαρμόζεται για πολλά χρόνια στην ευρέως γνωστή αρχιτεκτονική Μοντέλου-΄Οψης-Ελεγκτή(Model-View-Controller).

Είναι επίσης γνωστό ως πρότυπο Δημοσίευση-Εγγραφή(Publish-Subscribe).

Γενική Δομή Observer:

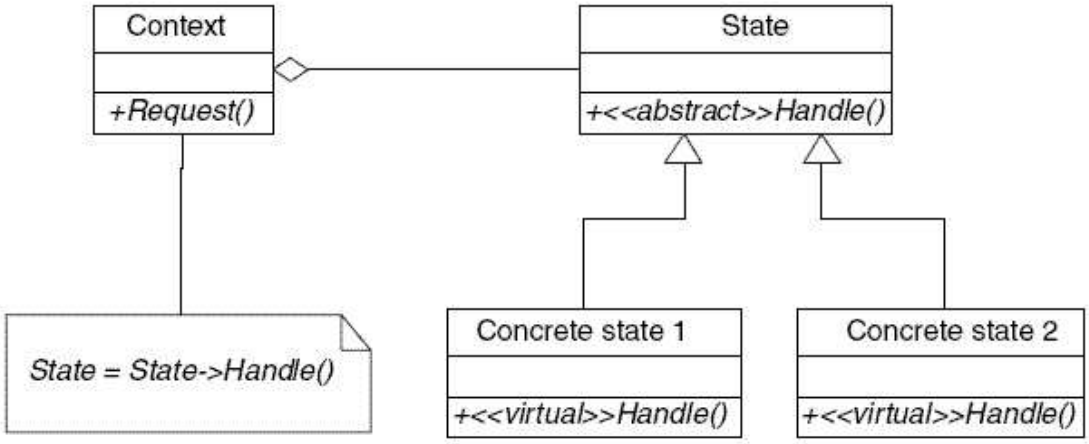



### 4.3 Πρότυπο State

Το πρότυπο αυτό επιτρέπει σε ένα αντικείμενο να τροποποιεί τη συμπεριφορά του όταν αλλάζει η εσωτερική κατάσταση. Το αντικείμενο μοιάζει να αλλάζει κλάση.

Το πρότυπο δίνει τη δυνατότητα σε ένα αντικείμενο να συμπεριφέρεται σαν να αλλάζει η κλάση του κάτι που στις περισσότερες γλώσσες προγραμματισμού είναι αδύνατο. Στο πρότυπο αυτό η κλάση-πελάτης περιέχει μια αφηρημένη κλάση, η οποία όμως δεν αντιπροσωπεύει μια στρατηγική αλλά μια κατάσταση. Οι παράγωγες κλάσεις υλοποιούν τις διάφορες συγκεκριμένες καταστάσεις και κατά συνέπεια η κλάση πελάτης μπορεί να εναλλάσσει την κατάστασή της αλλάζοντας την τιμή του δείκτη αναφοράς προς την περιεχόμενη κατάσταση. Σε μια παραλλαγή του προτύπου, η κάθε κατάσταση μπορεί να διατηρεί αναφορά προς την κλάση πελάτη, έτσι ώστε μετά την εκτέλεση κάποιας ενέργειας, να ενημερώνει την κατάσταση του πελάτη και να την προωθεί ενδεχομένως σε κάποια άλλη.

Γενική Δομή State:



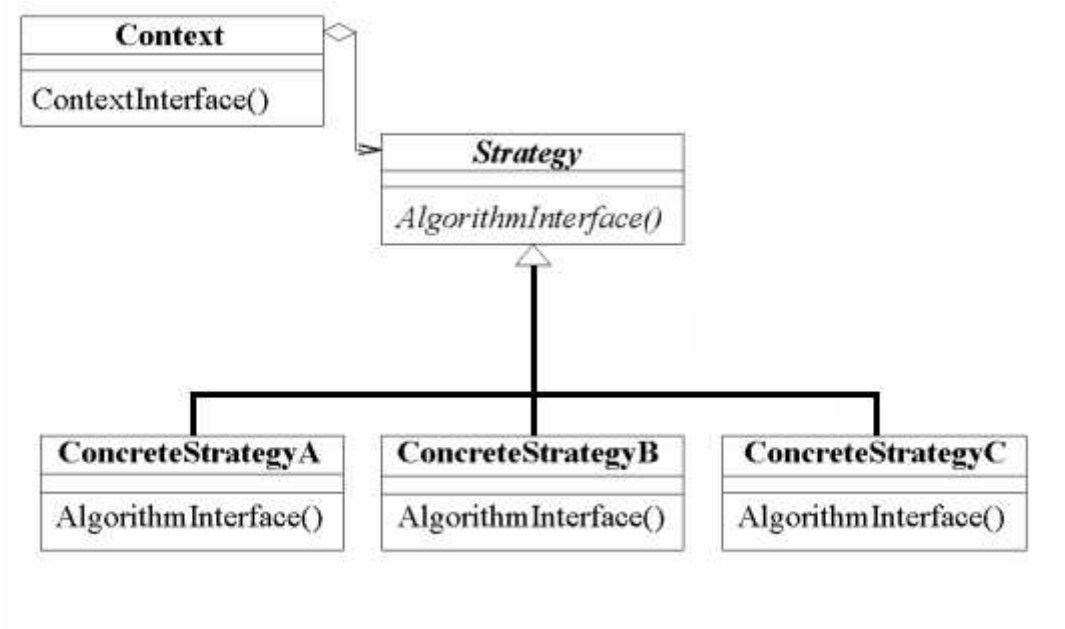

### 4.4 Πρότυπο Strategy

Το πρότυπο αυτό ορίζει μια οικογένεια αλγορίθμων, τους ενσωματώνει και επιτρέπει την εναλλαγή μεταξύ αυτών. Το πρότυπο δίνει τη δυνατότητα μεταβολής των αλγορίθμων ανεξάρτητα από τους πελάτες που τους χρησιμοποιούν.

Το πρότυπο σχετίζεται άμεσα με την αναγκαιότητα συχνής τροποποίησης του λογισμικού και είναι ίσως το πλέων ευρέως χρησιμοποιημένο πρότυπο, ανεξαρτήτως του αν η εφαρμογή του γίνεται αντιληπτή από τον σχεδιαστή. Το πρότυπο έχει την αναγκαιότητα να αναγνωρίζει ότι υπάρχει στην εφαρμογή μια γενική φιλοσοφία-στρατηγική που είναι κοινή σε πολλές περιπτώσεις, η συγκεκριμένη υλοποίηση κάθε περίπτωσης είναι διαφορετική. Ουσιαστικά υπάρχει ένας κοινός γενικός αλγόριθμος με διαφορετική λεπτομερής υλοποίηση σε κάθε περίπτωση. Αυτό επιτυγχάνεται με τη χρήση της κληρονομικότητας αλλά και τη σύνθεση αντικειμένων.

Το πρότυπο εφαρμόζει την αρχή της Αντιστροφής των Εξαρτήσεων αναγκάζοντας και το γενικό αλγόριθμο αλλά και τις λεπτομερείς υλοποιήσεις να εξαρτώνται από τις αφαιρέσεις. Για το λόγο αυτό το πρότυπο αυξάνει τον αριθμό των αντικειμένων στο σύστημα.

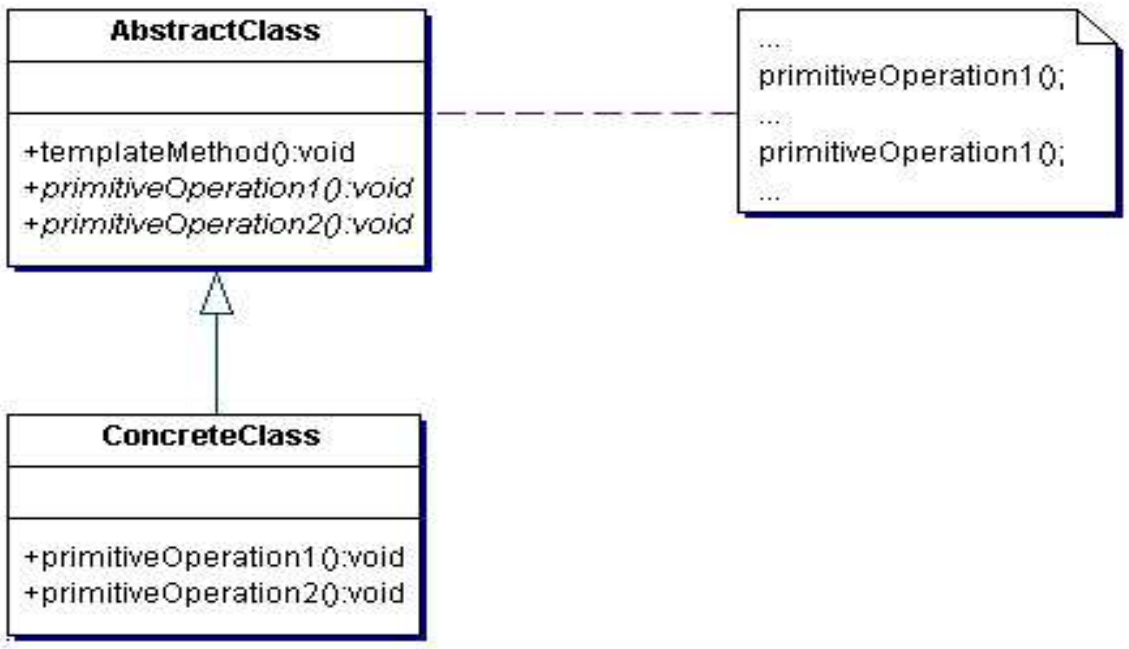
Γενική Δομή Strategy:




### 4.5 Πρότυπο Template method

Το πρότυπο αυτό ορίζει το περίγραμμα ενός αλγορίθμου σε μια λειτουργία, αφήνοντας ορισμένα βήματα στις παράγωγες κλάσεις. Επιτρέπει στις παράγωγες κλάσεις να επαναπροσδιορίσουν ορισμένα βήματα του αλγορίθμου χωρίς να αλλάξουν τη δομή του.
Είναι ένα πολυχρησιμοποιημένο πρότυπο. Εφαρμόζεται συχνά από τους σχεδιαστές αντικειμενοστρεφούς λογισμικού, ακόμα και αν δεν γίνεται αντιληπτό ως ξεχωριστή τεχνική. Κλασικότερο παράδειγμα εφαρμογής του προτύπου είναι στους αλγόριθμους ταξινόμησης.
Στόχος του προτύπου είναι ο διαχωρισμός ενός γενικού αλγόριθμου από συγκεκριμένες υλοποιήσεις, καθώς και η πλήρης εκμετάλλευση του μηχανισμού της κληρονομικότητας.
Το πρότυπο αυτό είναι εκλέπτυνση του Strategy για την περίπτωση που κάθε αλγόριθμος έχει πολλαπλές παραλλαγές οι οποίες διαφέρουν σε ορισμένα βήματα αλλά κάποια άλλα σημεία τους είναι κοινά. Τότε για κάθε αλγόριθμο μπορούμε να ορίσουμε ένα στοιχείο αφαίρεσης Β το οποίο υλοποιεί τη διασύνδεση Α και με τη σειρά του κληρονομείται από πολλές παραλλαγές του αλγορίθμου. Το κάθε Β περιέχει την υλοποίηση των αμετάβλητων βημάτων και, στα σημεία που οι παραλλαγές διαφέρουν, καλεί αφηρημένες προστατευμένες μεθόδους. Οι μέθοδοι αυτές υλοποιούνται διαφορετικά σε κάθε παραλλαγή που κληρονομεί το Β.
Το πρότυπο χρησιμοποιείται για τον ορισμό των αμετάβλητων τμημάτων και τη μετάθεση της υλοποίησης των μεταβλητών τμημάτων του αλγόριθμου σε παράγωγες κλάσεις.

Γενική Δομή Template method:



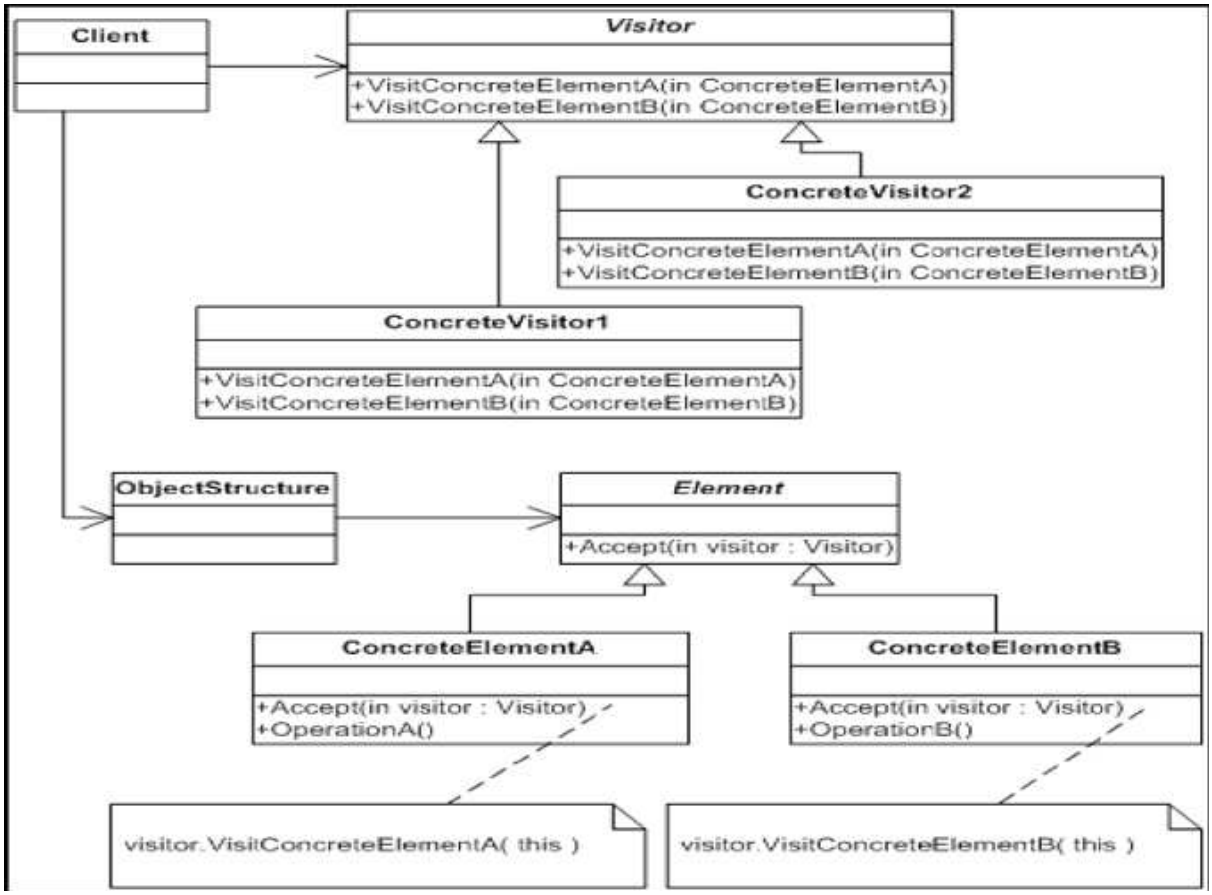

### 4.6 Πρότυπο Visitor

Το πρότυπο αυτό έχει στόχο την αναπαράσταση μιας λειτουργίας που πρόκειται να πραγματοποιηθεί στα στοιχεία μιας δομής αντικειμένων. Επιτρέπει την προσθήκη μιας νέας λειτουργίας στις υπάρχουσες κλάσεις του συστήματος χωρίς όμως την τροποποίηση του κώδικα στις κλάσεις στις οποίες επιδρά. Έτσι νέες λειτουργικότητες μπορούν να προστεθούν εύκολα στην αρχική ιεραρχία, χωρίς την τροποποίηση των ιδίων των κλάσεων, μόνο με τη δημιουργία μιας νέας υποκλάσης στο συγκεκριμένο πρότυπο.

Το πρότυπο έχει αρκετά μεγάλη πολυπλοκότητα στη λειτουργία του καθώς υλοποιεί τη λεγόμενη «διπλή αποστολή»(double ή dual dispatch). Τα μηνύματα στον αντικειμενοστρεφή προγραμματισμό κατά κανόνα επιδεικνύουν συμπεριφορά που αντιστοιχεί στην «απλή αποστολή», δηλαδή ότι η λειτουργία που εκτελείται εξαρτάται από το όνομα της αίτησης και τον τύπο του αποδέκτη. Στη διπλή αποστολή που προβλέπει το πρότυπο η λειτουργία που εκτελείται εξαρτάται από το όνομα της αίτησης και τον τύπο δύο αποδεκτών(από τον τύπο του Visitor και τον τύπο του στοιχείου που επισκέπτεται).

Η χρήση του προτύπου είναι ωφέλιμη κατά την προσθήκη λειτουργιών στα αντικείμενα μιας ιεραρχίας χωρίς να προκαλέσει μόλυνση στις κλάσεις τους. Το πρότυπο επιτρέπει να διατηρούνται οι σχετιζόμενες λειτουργίες αυτές σε μια νέα κλάση.

Γενική Δομή Visitor:


Client
Visitor
+VisitConcreteElementA(in ConcreteElementA)
+VisitConcreteElementB(in ConcreteElementB)
ConcreteVisitor2
+VisitConcreteElementA(in ConcreteElementA)
+VisitConcreteElementB(in ConcreteElementB)
ConcreteVisitor1
+VisitConcreteElementA(in ConcreteElementA)
+VisitConcreteElementB(in ConcreteElementB)
ObjectStructure
Element
+Accept(in visitor : Visitor)
ConcreteElementA
+Accept(in visitor : Visitor)
+OperationA()
ConcreteElementB
+Accept(in visitor : Visitor)
+OperationB()
visitor.VisitConcreteElementA( this )
visitor.VisitConcreteElementB( this )

# Κεφάλαιο 5

## Σχεδιαστικά πρότυπα για αλληλεπίδραση ανθρώπου-υπολογιστή


5.1 Εισαγωγή 28
5.2 Η ανάγκη για κοινή γλώσσα στην αλληλεπίδραση ανθρώπου-υπολογιστή 29
5.3 Κατηγοριοποίηση αλληλεπιδραστικών προτύπων 30
5.4 Ταξινόμηση αλληλεπιδραστικών προτύπων 33
5.5 Αξιολόγηση αλληλεπιδραστικών προτύπων 43


### 5.1 Εισαγωγή

Σε αυτή την ενότητα, θα ασχοληθούμε με τα πρότυπα για την Αλληλεπίδραση Ανθρώπου Υπολογιστή (Human Computer Interaction). Τα πρότυπα αυτά αποτελούν ένα εργαλείο για την τεκμηρίωση της γνώσης του σχεδιασμού διεπαφής και την αλλαγή του τρόπου που σχεδιάζουμε διεπαφές σε διάφορες εφαρμογές. Παρέχονται από σχεδιαστές που χρησιμοποιούν μια κοινή γλώσσα και πιο επαγγελματική τεκμηρίωση από ότι προσφέρουν οι παραδοσιακές κατευθύνσεις.

Ένα σχεδιαστικό πρότυπο HCI συλλαμβάνει την ουσία μιας επιτυχημένης λύσης σε ένα επαναλαμβανόμενο πρόβλημα ευχρηστίας σε διαδραστικά συστήματα.

Σήμερα, πολλές εταιρείες χρησιμοποιούν πρότυπα ως συμπλήρωμα στην ανάπτυξη του λογισμικού τους. Τα πρότυπα αυτά έχουν σημειώσει τη μεγαλύτερη πρόοδο στο σχεδιασμό του συστήματος, ή για να είμαστε πιο ακριβείς στον αντικειμενοστρεφή σχεδιασμό. Συνεπώς, αυτά τα πρότυπα είναι πιο ευρέως χρησιμοποιούμενα και τα HCI πρότυπα δεν έχουν χρησιμοποιηθεί ποτέ σε βιομηχανική ανάπτυξη. Σε λίγα χρόνια θα υπάρξει πιθανών, ένας αριθμός από εταιρείες που θα χρησιμοποιούν τα πρότυπα HCI κατά την ανάπτυξη των εφαρμογών. Το πρόβλημα σήμερα είναι ότι δεν υπάρχει ευρέως αποδεκτή γλώσσα για HCI πρότυπα την οποία οι προγραμματιστές μπορούν να εμπιστεύονται και να χρησιμοποιούν. Αν μια εταιρεία επιθυμεί να αναπτύξει το δικό της σύνολο HCI προτύπων θα χρειαστεί πολύ

χρόνο και προσπάθεια, επομένως η ανάγκη για μια γενική γλώσσα σχεδιαστικών προτύπων θα ήταν ευπρόσδεκτη.

Αν σε μια συλλογή προτύπων κάποια προβλήματα εξακολουθούν να υπάρχουν και πρέπει να συζητηθούν και να επιλυθούν πριν από την ένταξή τους στον κύκλο ζωής ενός έργου λογισμικού, τότε αυτό μπορεί να είναι δυνατό να συμβεί. Πρώτον και κύριον, οι συμφωνίες πρέπει να προβλέπουν ότι η χρήση των HCI προτύπων θα βελτιώσει κάποια πτυχή του σχεδιασμού διεπαφής. Με τη χρήση των HCI προτύπων ο αριθμός των ωρών που δαπανώνται για το σχεδιασμό θα μπορούσε να είναι πολύ λιγότερος από ό,τι σήμερα, αλλά η εξοικονόμηση χρόνου δε έχει καμία σημασία μέχρι τα HCI πρότυπα χρησιμοποιηθούν σε πραγματικά έργα. Μια άλλη πιθανή συνέπεια είναι ότι η διαδικασία σχεδιασμού της διεπιφάνειας χρήστη μπορεί να γίνει πολύ ευκολότερη από ό,τι σήμερα, αρκεί να καταβληθεί σοβαρή προσπάθεια.

Εάν τα πρότυπα πρέπει να χρησιμοποιούνται στη διαδικασία του σχεδιασμού διεπαφής, η ενσωμάτωση των προτύπων στον κύκλο ζωής του λογισμικού πρέπει να συζητηθεί. Ένας αριθμός ερωτήσεων πρέπει να απαντηθεί: Τα πρότυπα θα χρησιμοποιούνται μόνο όταν έχουμε προβλήματα ή θα χρησιμοποιηθούν κατά τη διαδικασία σχεδιασμού; Ένα άλλο σημαντικό ερώτημα είναι αν μόνο οι HCI σχεδιαστές θα χρησιμοποιούν τα πρότυπα ή αν οι προγραμματιστές θα έχουν την ευκαιρία να συζητήσουν τις διεπιφάνειες με επιχειρήματα από τα πρότυπα; Ακόμη και οι τελικοί χρήστες θα μπορούσαν με τη βοήθεια των HCI προτύπων να επικρίνουν το γραφικό σχεδιασμό του συστήματος;

### 5.2 Η ανάγκη για κοινή γλώσσα στην αλληλεπίδραση ανθρώπου-υπολογιστή

Σίγουρα, δεν είναι καθόλου εύκολη διαδικασία να σχεδιαστεί ένα σύστημα με το οποίο οι άνθρωποι να αλληλεπιδρούν μαζί του. Είναι δύσκολο να δημιουργηθεί ένα καλό σύστημα, χωρίς τη συμμετοχή των τελικών χρηστών. Όταν οι σχεδιαστές πρόκειται να έχουν μια συνομιλία με τους τελικούς χρήστες θα ήταν εύκολο αν όλοι στη συζήτηση κατανοήσει ο ένας τον άλλον χωρίς κανένα πρόβλημα. Αυτό δυστυχώς δεν συμβαίνει σήμερα, οι χρήστες συχνά δυσκολεύονται να εκφράζουν αυτό που θέλουν και οι σχεδιαστές έχουν πρόβλημα να καταλάβουν τα προβλήματα που εκφράζουν οι χρήστες. Τα προβλήματα γίνονται δυσκολότερα όταν οι πελάτες, επίσης, εμπλέκονται στη διαδικασία σχεδιασμού ενός συστήματος.

Αν οι σχεδιαστές, οι χρήστες και οι πελάτες μιλούσαν όλοι την ίδια γλώσσα, θα έκαναν την όλη διαδικασία λίγο πιο απλή, αλλά αυτό δεν θα έλυνε όλα τα προβλήματα που μπορεί να προκύψουν κατά τη διάρκεια της ανάπτυξης ενός συστήματος. Μια λύση στο πρόβλημα της επικοινωνίας θα μπορούσε να ήταν η χρησιμοποίηση των προτύπων ως μιας κοινή γλώσσα

εισόδου. Όλα τα πρότυπα αλληλεπίδρασης έχουν ένα όνομα και αυτό το όνομα είναι το πιο σημαντικό ολόκληρου του προτύπου. Το όνομα του προτύπου θα πρέπει να εξηγήσει τι κάνει το πρότυπο και να μπορεί εύκολα να θυμάται και να συνδεθεί με το σχετικό πρόβλημα. Οι ονομασίες αυτές μπορούν να λειτουργήσουν ως οι πυλώνες σε μία κοινή γλώσσα και η σημασιολογία του προτύπου ως βοήθεια για να μιλήσουμε για πιο ουσιαστικές λεπτομέρειες κατά τη φάση του σχεδιασμού. Αυτό το νέο λεξιλόγιο θα μπορούσε πιθανότατα να είναι πιο χρήσιμο σε μια συνομιλία μεταξύ σχεδιαστών όπου σήμερα δεν υπάρχει μια κοινή γλώσσα. Το πρόβλημα, όπως προαναφέρθηκε νωρίτερα, είναι ότι δεν υπάρχει μια ευρέως αποδεκτή γλώσσας σχεδιαστικών προτύπων για το σχεδιασμό HCI. Όσο δεν υπάρχει μια τέτοια κοινή γλώσσα προτύπων, κάθε κοινά ομιλούμενη γλώσσα μεταξύ των χρηστών, σχεδιαστών , και πελατών θα ήταν δύσκολο να βρεθεί και να χρησιμοποιηθεί. Για το πρόβλημα αυτό θα χρειαστούν μερικά χρόνια για την επίλυση του, αλλά θα μπορούσε να γίνει. Μια απόδειξη αυτού ότι τα πρότυπα GoF, που τα ονόματα των προτύπων χρησιμοποιούνται σε διάφορους τομείς.

Στον τομέα της αλληλεπίδρασης εμπλέκονται άνθρωποι με διαφορετικό υπόβαθρο και αυτός είναι ο λόγος για τον οποίο τα πρότυπα πρέπει να είναι γραμμένα με τέτοιο τρόπο ώστε ένας χρήστης με λίγη ή καθόλου γνώση HCI να μπορεί να καταλάβει την ουσία του προτύπου. Οι χρήστες δεν χρειάζεται να καταλάβουν πώς το σύστημα είναι χτισμένο μέσα, το μόνο που χρειάζεται να γνωρίζουν είναι πως το διαδραστικό σύστημα έχει σχεδιαστεί.

### 5.3 Κατηγοριοποίηση αλληλεπιδραστικών προτύπων

Καθώς τα πρότυπα θα χρησιμοποιηθούν περισσότερο στο μέλλον, θα πρέπει να είναι εύκολα να βρεθούν για να χρησιμοποιηθούν. Μια εφαρμογή για να χειριστεί τα πρότυπα είναι απαραίτητο να ενσωματώσει τα πρότυπα στον κύκλο ζωής της ανάπτυξης λογισμικού. Για να δημιουργηθεί μια εφαρμογή που προσφέρει περισσότερα από μια ανοργάνωτη απλή απαρίθμηση των υφιστάμενων προτύπων, η κατηγοριοποίηση που χωρίζει πρότυπα από διαφορετικά πεδία από κάθε άλλο είναι απαραίτητη. Η δημιουργία αυτής της κατηγοριοποίησης είναι αναγκαία για διάφορους λόγους. Οι λόγοι αυτοί βασίζονται στην παραδοχή ότι στο μέλλον θα υπάρχουν αρκετές εκατοντάδες HCI πρότυπα και είναι οι εξής:

Ο λόγος του δημιουργού προτύπων(pattern developer's reason): Κάποιος που αναπτύσσει κάποιο πρότυπο θέλει να ξέρει κατά τη φάση της τεκμηρίωσης, αν αυτό το πρότυπο έχει τεκμηριωθεί από πριν. Κατά συνέπεια, μια κατηγοριοποίηση όπου μπορεί να κατατάξει το δικό του πρότυπο και να ψάξει αν στην ίδια κατηγορία υπάρχει ήδη ένα παρόμοιο πρότυπο είναι απαραίτητη.

Ο λόγος του έμπειρου HCI(HCI expert's reason): Κάποιος έμπειρος στην ανάπτυξη διεπιφανειών θα ήθελε να επεκτείνει κάποιο μέρος της διεπιφάνειας. Αυτό θα μπορούσε να γίνει με αναζήτηση σε διάφορα πρότυπα για να πάρει συμβουλές για το πώς να το κάνει αυτό με καλό και τεκμηριωμένο τρόπο.

Ο λόγος του δημιουργού διεπιφανειών(interface developer's reason): Κάποιος που αναπτύσσει μια διεπιφάνεια, κατά τη διάρκεια ανάπτυξης ενός ξεχωριστού τμήματος της διεπιφάνειας, θα μπορούσε να βοηθηθεί από την ανάγνωση ενός HCI προτύπου το οποίο έχει καταγράψει τα προβλήματα που αφορούν το συγκεκριμένο τμήμα. Κατά συνέπεια, μια κατηγοριοποίηση όπου τα πρότυπα κατατάσσονται βάση των τμημάτων της διεπιφάνειας, θα διευκόλυνε τον δημιουργό της διεπιφάνειας.

Ο λόγος του χρήστη διεπιφανειών(interface user's reason): Κάποιος που χρησιμοποιεί μια διεπιφάνεια, θα μπορούσε να χρησιμοποιήσει πρότυπα για να καταλάβει τους λόγους που οι σχεδιαστές και οι προγραμματιστές της διεπιφάνειας επέλεξαν το συγκεκριμένο σχεδιασμό. Ένας άλλος λόγος, είναι για να συζητήσει θέματα σχεδιασμού με τους προγραμματιστές της διεπιφάνειας.

Οι λόγοι αυτοί υποδηλώνουν ότι μία τέτοια κατηγοριοποίηση δεν θα είναι αρκετή. Τουλάχιστον δύο διαφορετικοί τύποι των κατηγοριών με πολλαπλά χαρακτηριστικά πρέπει να δημιουργηθούν.

Η πρώτη κατηγορία ονομάζεται κατηγοριοποίηση των δημιουργών των προτύπων(Pattern Developers Categorization)και η άλλη κατηγοριοποίηση των χρηστών των προτύπων(Pattern Users Categorization).

Κατηγοριοποίηση των δημιουργών των προτύπων(Pattern Developers Categorization):

Όπως αναφέρθηκε πιο πάνω οι δημιουργοί και οι έμπειροι HCI χρειάζονται μια κατηγοριοποίηση των σχεδιαστικών HCI προτύπων. Ένας λόγος για αυτό είναι ότι οι δημιουργοί θα επωφεληθούν γνωρίζοντας εάν το πρότυπο που δημιουργούν τεκμηριώνεται ή όχι. Ένας άλλος λόγος είναι ότι η κατηγοριοποίηση θα μπορούσε να βοηθήσει τους δημιουργούς να δουν ότι το πρότυπο βελτιώνει πραγματικά την ευχρηστία του συστήματος. Για τον ίδιο λόγο οι έμπειροι πρέπει να βρουν τα πρότυπα που βελτιώνουν την ευχρηστία της διεπιφάνειας σε κάποια πτυχή.

Η κατηγορία αυτή χωρίζεται σε δύο μέρη, όπου τα δύο μέρη αποτελούνται από διάφορα χαρακτηριστικά. Το πρώτο μέρος ονομάζεται ευχρηστία(usability) και το δεύτερο ονομάζεται αποτελεσματικότητα(effectivity). Η ευχρηστία επικεντρώνεται στην ευχρηστία και αποτελείται από πέντε ιδιότητες και η αποτελεσματικότητα επικεντρώνεται στους μετρήσιμους ανθρώπινους παράγοντες και αποτελείται από επίσης πέντε χαρακτηριστικά.

Τα χαρακτηριστικά της ευχρηστίας είναι τα εξής:

Ορατότητα(Visibility): Το πρότυπο χρησιμοποιεί τις αρχές που δίνουν στο χρήστη την αίσθηση ότι τα αντικείμενα στη διεπιφάνεια φαίνονται καθαρά.

Περιορισμοί(Constraints): Το πρότυπο χρησιμοποιεί τις αρχές που χρησιμοποιούν οριοθετήσεις ή περιορισμούς που δίνουν στο χρήστη μια καλύτερη αίσθηση της διεπιφάνειας.

Διαθεσιμότητα(Affordance): Το πρότυπο χρησιμοποιεί τις αρχές που εκμεταλλεύονται την "δύναμη" που κάθε τεχνούργημα έχει να δώσει στο χρήστη μια καλύτερη αίσθηση της διεπιφάνειας.

Αντιστοιχίσεις(Mappings): Το πρότυπο χρησιμοποιεί τις αρχές που κάνουν το χρήστη να πιάσει τη σύνδεση μεταξύ ενός πράγματος στη διεπιφάνεια και ενός τεχνουργήματος στον "πραγματικό κόσμο". Αυτό μπορεί να δώσει στο χρήστη μια καλύτερη αίσθηση της διεπιφάνειας.

Ανάδραση(Feedback): Το πρότυπο χρησιμοποιεί τις αρχές που δίνουν στο χρήστη ένα σημάδι ότι κάτι έχει συμβεί στο σύστημα.

Τα χαρακτηριστικά της αποτελεσματικότητας είναι τα εξής:

Χρόνος για μάθηση(Time to learn): Πόσο χρόνο χρειάζονται τα τυπικά μέλη της κοινότητας των χρηστών να μάθουν πώς να χρησιμοποιούν τις εντολές που σχετίζονται με μια σειρά από διεργασίες.

Ταχύτητα απόδοσης(Speed of performance): Πόσος χρόνος χρειάζεται για να εκτελείται η κάθε διεργασία.

Ποσοστό των σφαλμάτων από τους χρήστες(Rate of Errors by Users): Πόσα και τι είδους λάθη κάνουν οι χρήστες κατά την εκτέλεση των διεργασιών.

Διατήρηση με την πάροδο του χρόνου(Retention over Time): Πόσο καλά οι χρήστες διατηρούν τις γνώσεις τους μετά από μια ώρα, μια μέρα, ή μια εβδομάδα; Η διατήρηση μπορεί να συνδέεται στενά με το χρόνο για να μάθουν και η συχνότητα της χρήσης διαδραματίζει σημαντικό ρόλο.

Υποκειμενική ικανοποίηση(Subjective Satisfaction): Πόσο αρέσει στους χρήστες να χρησιμοποιούν διάφορες πτυχές του συστήματος; Η απάντηση μπορεί να διαπιστωθεί από συνεντεύξεις ή γραπτές έρευνες που περιλαμβάνουν την ικανοποίηση ανά κλίμακα και χώρο για ελεύθερα σχόλια και διάφορες παρατηρήσεις.

Κατηγοριοποίηση των χρηστών των προτύπων(Pattern Users Categorization):

Υπάρχουν πολλές ανάγκες για την κατηγοριοποίηση που επικεντρώνεται σε αυτό που επικεντρώνεται ο χρήστης των προτύπων. Ο χρήστης δεν θα μπορούσε να θεωρηθεί ως

έμπειρος HCI, γεγονός που καθιστά δύσκολο να χρησιμοποιηθούν κατηγορίες που αφορούν τη διαθεσιμότητα και σημασιολογικούς περιορισμούς. Ο χρήστης χρειάζεται μια κατηγοριοποίηση που τον βοηθά να βρεί τα πρότυπα που βασίζονται στο μέρος της διεπιφάνειας που πρόκειται να αναπτυχθούν.
Η κατηγορία αυτή χωρίζεται σε τρία μέρη, όπου όλα τα μέρη αποτελούνται από διάφορα χαρακτηριστικά. Το πρώτο μέρος ονομάζεται εμπειρία, το δεύτερο στυλ αλληλεπίδρασης και το τρίτο επίπεδο διεπιφάνειας. Η εμπειρία επικεντρώνεται στο είδος των τελικών χρηστών, που θα έχουν τα συστήματα και αποτελείται από τρεις παραμέτρους. Το στυλ αλληλεπίδρασης εστιάζεται στο στυλ αλληλεπίδρασης των συστημάτων και αποτελείται από πέντε παραμέτρους. Το επίπεδο διεπιφάνειας επικεντρώνεται στο επίπεδο διεπιφάνειας των συστημάτων και αποτελείται από τρεις παραμέτρους.

Τα χαρακτηριστικά της εμπειρίας είναι τα εξής:
Αρχάριοι χρήστες ή χρήστες που χρησιμοποιούν το σύστημα για πρώτη φορά(Novice or first-time Users)
Καλά περιοδικοί χρήστες(Knowledgeable Intermittent Users)
΄Εμπειροι χρήστες(Expert Frequent Users)

Τα χαρακτηριστικά του στυλ αλληλεπίδρασης είναι τα εξής:
΄Αμεσος χειρισμός(Direct Manipulation)
Επιλογή Μενού(Menu Selection)
Συμπλήρωση Φόρμας(Form Fillin)
Γλώσσα εντολών(Command Language)
Φυσικής Γλώσσα(Natural Language)

Τα χαρακτηριστικά του επιπέδου διεπιφάνειας είναι τα εξής:
Εφαρμογή(Application)
Περιεχόμενα(Containers)
Μικρές εφαρμογές(Widgets)

### 5.4 Ταξινόμηση αλληλεπιδραστικών προτύπων

Όλα τα αλληλεπιδραστικά πρότυπα ταξινομούνται βάση μιας κοινής γλώσσας όπου κάθε πρότυπο που περιγράφεται θα πρέπει να είναι κατανοητό τουλάχιστον από ένα άτομο που έχει δει την περίπτωση από πριν. Πολλές από τις περιπτώσεις είναι εύκολο να αναγνωριστούν διαβάζοντας το πρότυπο. Εάν υπάρχει κάποιο πρόβλημα στην αναγνώριση

παρέχεται μια αναφορά στο πρότυπο στην αρχική του μορφή. Η αρχική μορφή που οι συντάκτες αυτών των προτύπων χρησιμοποιούν είναι πολύ σαφέστερη από τη συμπαγή μορφή, ωστόσο καμιά πληροφορία δεν πρόκειται να χαθεί.

Για κάθε πρότυπο, το πρόβλημα και η απάντηση στις ερωτήσεις when(πότε), how(πώς) και why(γιατί) η λύση μπορεί να εφαρμοστεί σε αυτό το πρόβλημα θα πρέπει να δηλώνεται. Ο στόχος είναι να λαμβάνονται όσο γίνεται περισσότερες πληροφορίες από τα πρωτότυπα, αλλά μερικές φορές οι προτάσεις πρέπει να αναδιατυπωθούν. Οι επαναδιατυπώσεις δεν γίνονται για να βελτιωθούν τα πρότυπα με οποιονδήποτε τρόπο, αλλά να τα κάνουν να ταιριάζουν σε συμπαγή μορφή.

Το τμήμα του αρχικού προβλήματος θα πρέπει να χρησιμοποιηθεί χωρίς καμία εξαίρεση σα τμήμα του προβλήματος στη συμπαγή του μορφή. Το πρόβλημα είναι συνήθως πολύ μικρό και δεν χρειάζεται να επαναδιατυπωθεί.

Το τμήμα που αρχίζει με τη λέξη ‘when’(πότε), σημαίνει ότι κάποια πρόταση πρέπει να επαναδιατυπωθεί ώστε να είναι κατάλληλη.

Το τμήμα how(πώς), δίνει έμφαση στη λύση του προβλήματος. Η περιγραφή της λύσης είναι συνήθως πολύ λεπτομερής και ο στόχος είναι να κάνει τον αναγνώστη να κατανοήσει την έννοια του προτύπου. Συνεπώς, μόνο ένα μικρό μέρος της λύσης του τμήματος του πρωτότυπου θα πρέπει να ληφθεί, η οποία εξακολουθεί να αποδίδει την βασική ιδέα που βρίσκεται πίσω από το πρότυπο. Το τμήμα αυτό αρχίζει με τη λέξη ‘then’, ακολουθούμενη από κόμμα, και είναι μία λέξη που ταιριάζει καλύτερα σε αυτό το πλαίσιο.

Το τμήμα why(γιατί), αρχίζει με τη φράση ‘A reason for this is that’ και μπορεί να συνεχίζει με τη φράση ‘Another reason is that’ εξηγά γιατί η λύση μπορεί να εφαρμοστεί στο συγκεκριμένο πρότυπο.

Ένα παράδειγμα σύνταξης ενός αλληλεπιδραστικού προτύπου φαίνεται πιο κάτω:

Problem:[Το πρόβλημα το οποίο θα λύσει το πρότυπο]

When, How and Why: When[το πλαίσιο στο οποίο το πρότυπο θα χρησιμοποιηθεί].Then[Η λύση που δίνεται στο συγκεκριμένο πρόβλημα]. A reason for this is that[Μία εξήγηση γιατί δουλεύει η λύση στο συγκεκριμένο πρόβλημα]. Another reason is that [Μία δεύτερη εξήγηση γιατί δουλεύει η λύση στο συγκεκριμένο πρόβλημα].

Πιο κάτω βλέπουμε σε συμπαγή μορφή τα κυριότερα αλληλεπιδραστικά πρότυπα:

**Narrative**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Σε ποια μορφή θα πρέπει οι πληροφορίες να εμφανίζονται στο χρήστη; Πότε, πώς και γιατί: Όταν υπάρχει ανάγκη να μεταφέρονται πληροφορίες για τον χρήστη, οι πληροφορίες μπορεί να είναι αλληλένδετες, αλλά διαφορετικών ειδών, και μπορεί να υπάρξει κάποια εμπλοκή υποκειμενικότητας . Στη συνέχεια, μεταφέρονται οι πληροφορίες μέσω της φυσικής γλώσσας. Ένας λόγος για αυτό είναι ότι πολλοί άνθρωποι θεωρούν ότι είναι πιο ευχάριστο να εργάζονται με φυσική γλώσσα σε σχέση παρά με συμβολικές αναπαραστάσεις. Ένας άλλος λόγος είναι ότι πολλοί άνθρωποι βρίσκουν τη φυσική γλώσσα πιο ευχάριστη από τα ακατέργαστα δεδομένα και τις συμβολικές αναπαραστάσεις.

**Form**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς θα πρέπει να το σύστημα να αναφέρει τι είδους πληροφορίες πρέπει να παρέχονται, και αν θα επεκτείνονται ή όχι;

Πότε, πώς και γιατί: Όταν ο χρήστης πρέπει να παρέχει πληροφορίες διαμορφωμένες, συνήθως σύντομες (μη επεξηγηματικές) απαντήσεις σε ερωτήσεις. Στη συνέχεια, παρέχονται τα κατάλληλα "κενά" που πρέπει να συμπληρωθούν, τα οποία σαφέστατα και ορθά αναφέρουν τι είδους πληροφορίες πρέπει να παρέχονται. Οπτικά, τα κενά φαίνονται επεξεργάσιμα, με ανεπαίσθητες αλλαγές στο χρώμα φόντου, έτσι ώστε ο χρήστης να μπορεί να δει με μια ματιά τι πρέπει να συμπληρωθεί. Οι ετικέτες είναι σαφείς και χρησιμοποιούν ορολογία με την οποία ο χρήστης είναι εξοικειωμένος. Τοποθετούνται όλα σε μια σειρά που έχει νόημα σημασιολογικά, και όχι απλώς να ομαδοποιηθούν. Ένας λόγος για αυτό είναι ότι βοηθά το χρήστη να δει τι χρειάζεται και τι προαιρετικό. Ένας άλλος λόγος είναι ότι είναι απλό ώστε να μην υπάρχει μεγάλη ανάγκη ο χρήστης να καταφύγει στις οδηγίες, τις οποίες διαβάζει σπάνια.

**Control Panel**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς μπορεί το σύστημα να παρουσιάσει καλύτερα τις ενέργειες που ο χρήστης πιθανών να ακολουθήσει;

Πότε, πώς και γιατί: Όταν το σύστημα πρέπει να παρέχει έναν τρόπο για τον χρήστη, είτε να αλλάξει κατάσταση, ή την εντολή για να κάνει κάτι. Στη συνέχεια, για κάθε μεταβλητή λειτουργία ή κατάσταση που αποτελεί μέρος του νοητικού μοντέλου του χρήστη, επιλέγει ένα καλά σχεδιασμένο έλεγχο που εκτελεί τη λειτουργία ή εμφανίζει την τιμή της μεταβλητής. Τα βάζουμε όλα μαζί έτσι ώστε οι πιο συχνά χρησιμοποιούμενοι έλεγχοι να

είναι οι σημαντικότεροι. Ένας λόγος για αυτό είναι ότι ο χρήστης θέλει ένα μέρος όπου ξέρει ότι μπορεί να βρει τους απαραίτητους ελέγχους, χωρίς να χρειάζεται να ψάξει γι'αυτούς.

**Composed Command**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς μπορεί το σύστημα να παρουσιάσει καλύτερα τις ενέργειες που ο χρήστης πιθανών να ακολουθήσει;

Πότε, πώς και γιατί: Όταν οι πιθανές ενέργειες που θα ακολουθηθούν μπορούν να εκφραστούν μέσα από τις εντολές, οι οποίες μπορεί να αποτελούνται από μικρότερα μέρη, σε μια γλώσσα με ακριβείς και εύκολους στη μάθηση κανόνες, τους οποίους οι χρήστες είναι πρόθυμοι και ικανοί να μάθουν. Στη συνέχεια, παρέχει έναν τρόπο για το χρήστη να πληκτρολογήσει την εντολή άμεσα, μιλώντας ή πληκτρολογώντας την. Ένας λόγος για αυτό είναι ότι οι έμπειροι συχνά βρίσκουν τις γλωσσικές εντολές πιο αποτελεσματικές από οπτικές αναπαραστάσεις. Ένας άλλος λόγος είναι ότι μερικές φορές οι διαθέσιμες ενέργειες δεν μπορούν, ή δεν πρέπει, να εκφραστούν γραφικά.

**Navigable Spaces**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς μπορεί να παρουσιαστεί το περιεχόμενο, έτσι ώστε ο χρήστης να μπορεί να εξερευνήσει με το δικό τους ρυθμό, με τρόπο που να είναι κατανοητό και ελκυστικό για τον χρήστη;

Πότε, πώς και γιατί: Όταν το σύστημα περιέχει πολύ περιεχόμενο, τότε είναι δύσκολο να παρουσιαστεί σε μια ενιαία προβολή. Αυτό το περιεχόμενο μπορεί να οργανωθεί σε διαφορετικούς εννοιολογικά χώρους ή επιφάνειες εργασίας, οι οποίες συνδέονται μεταξύ τους εννοιολογικά, έτσι ώστε να είναι φυσικό και να έχει νόημα για να μεταφερθείς από τον έναν χώρο στον άλλο. Στη συνέχεια, δημιουργείται η ψευδαίσθηση ότι οι επιφάνειες εργασίας είναι οι κενές και ο χρήστης μπορεί να μπει και να βγει από αυτές. Ένας λόγος για αυτό είναι ότι ο χρήστης θέλει να μάθει πού μπορεί (ή πρέπει) να πάει μετά, και πώς έχει σχέση με το χώρο που βρίσκεται τώρα. Ένας άλλος λόγος είναι ότι είναι ευχάριστο ο χρήστης να εξερευνήσει ένα νέο χώρο, όπου δεν γνωρίζει ποιοι είναι οι "γύρω από τη γωνία».

**Step-by-Step Instructions**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: : Πώς μπορεί να ξεδιπλώσει το σύστημα τις πιθανές ενέργειες για το χρήστη με τρόπο που δεν θα τον συγχύσει, αλλά θα τον οδηγήσει σε επιτυχή ολοκλήρωση του έργου;

Πότε, πώς και γιατί: Όταν ένας χρήστης πρέπει να εκτελέσει ένα πολύπλοκο έργο, με περιορισμένο χρόνο, γνώση, προσοχή, ή χώρο. Εναλλακτικά, η φύση της εργασίας είναι βήμα-βήμα, και δεν έχει νόημα να δείξουμε όλες τις πιθανές ενέργειες σε ένα βήμα. Στη συνέχεια, καθοδηγούμε το χρήστη με ένα βήμα κάθε φορά δίνοντας του πολύ σαφείς οδηγίες σε κάθε βήμα. Ένας λόγος για αυτό είναι ότι ο χρήστης δεν θέλει πάντα, ή δεν χρειάζεται, να κατανοήσει λεπτομερώς το τι κάνει. Ένας άλλος λόγος είναι ότι ένας χρήστης ο οποίος φοβάται αν κάνει κάτι λάθος μπορεί να προτιμά οι ενέργειες που πρέπει να εκτελεστούν να αναφέρονται ρητά γι 'αυτούς.

**Small Groups of Related Things**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς θα πρέπει τα αντικείμενα ή οι ενέργειες να οργανωθούν;

Πότε, πώς και γιατί: Όταν υπάρχουν πολλά αντικείμενα ή ενέργειες, μερικά από τα οποία είναι πιο στενά συνδεδεμένα μεταξύ τους από ό,τι άλλα πράγματα. Στη συνέχεια, ομαδοποιούνται τα στενά συνδεδεμένα πράγματα σε μια ιεραρχία ομάδων, αν χρειαστεί. Ένας λόγος για αυτό είναι ότι οι μεγάλες, αδιαφοροποίητες μάζα στοιχείων μπορεί να είναι εκφοβιστικό και δύσκολο να γίνουν κατανοητές, ειδικά για κάποιον που τις βλέπει για πρώτη φορά. Ένας άλλος λόγος είναι ότι οι άνθρωποι υποθέτουν σημασιολογική συνοχή όπου υπάρχει οπτική συνοχή.

**Hierarchical Set**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς θα πρέπει οι πληροφορίες να οργανωθούν;

Πότε, πώς και γιατί: Όταν υπάρχουν πολλά πράγματα και είναι αλληλένδετα σε μια ιεραρχία. (ή μπορεί να γίνει για να εμφανιστεί με αυτόν τον τρόπο). Στη συνέχεια, τα στοιχεία εμφανίζονται σε μια δενδρική δομή. Ένας λόγος για αυτό είναι ότι ο χρήστης θα πρέπει να δει τη δομή των δεδομένων. Ένας άλλος λόγος είναι ότι οι ιεραρχίες γίνονται πιο εύκολα κατανοητές.

**Optional Detail On Demand**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πότε θα πρέπει αυτά τα περιττά αντικείμενα συνήθως να παρουσιάζονται στο χρήστη, και πώς;

Πότε, πώς και γιατί: Όταν ένα μεγάλο ποσοστό των διαθέσιμων πληροφοριών ή ενεργειών (που ονομάζονται "στοιχεία" για το υπόλοιπο αυτού του προτύπου) μπορούν να θεωρηθούν λεπτομέρειες, και είναι αχρείαστες τις περισσότερες φορές. Λεπτομέρειες και περαιτέρω

επιλογές που δεν θα χρειάζονται τις περισσότερες φορές - ας πούμε 20% ή λιγότερο αναμενόμενες χρήσεις - μπορεί να κρύβονται σε ένα ξεχωριστό χώρο ή επιφάνεια εργασίας (ένα άλλο παράθυρο, ένα άλλο κομμάτι χαρτί, ένα κενό πάνελ). Ένας λόγος για αυτό είναι ότι ο χρήστης μπορεί να συγχυστεί εφόσον παρουσιάζονται πολλά στοιχεία ταυτόχρονα. Ένας άλλος λόγος είναι ότι μερικές φορές δεν υπάρχει αρκετός χώρος για να εμφανιστούν όλα τα πιθανά στοιχεία.

**Short Description**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς θα πρέπει να παρουσιάσει το σύστημα επιπρόσθετο περιεχόμενο. Με τη μορφή διευκρινιστικών στοιχείων ή εξηγήσεων των πιθανών ενεργειών, στο χρήστη όπου χρειάζεται;

Πότε, πώς και γιατί: Όταν το σύστημα περιέχει ένα οπτικό δείκτη, ή «εικονικές δακτύλου" (το ποντίκι ή το στυλό, για παράδειγμα), που είναι το σημείο εστίασης για την αλληλεπίδραση του χρήστη με τη διεπιφάνεια. Στη συνέχεια, δείχνει μια μικρή (μια φράση ή μικρότερη) περιγραφή ενός πράγματος, σε σύντομη και / ή χρονική εγγύηση στο ίδιο πράγμα. Ένας λόγος για αυτό είναι ότι οι χρήστες συνήθως δεν θέλουν να εγκαταλείψουν το σύστημα και να καταφύγουν κάπου αλλού για βοήθεια(π.χ. εγχειρίδιο). Αυτό χαλάει συνήθως τη συγκέντρωση και κοστίζει πάρα πολύ χρόνο. Ένας άλλος λόγος είναι ότι δεν υπάρχει περιθώριο για να θέσει ο χρήστης στατικό περιγραφικό κείμενο στο σύστημα.

**Go Back to a Safe Place**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς μπορεί το σύστημα να κάνει την πλοήγηση εύκολη, άνετη, και ψυχολογικά ασφαλή για το χρήστη;

Πότε, πώς και γιατί: Όταν το σύστημα επιτρέπει στον χρήστη να κινηθεί διαμέσου των κενών χώρων. Στη συνέχεια, παρέχει έναν τρόπο για να επιστρέψει σε ένα σημείο ελέγχου της επιλογής του χρήστη. Ένας λόγος για αυτό είναι ότι ένας χρήστης μπορεί να ξεχάσει πού ήταν, εάν σταματήσει να χρησιμοποιεί το σύστημα ενώ είναι στη μέση κάτι και δεν πάει πίσω σε αυτό για κάποιο διάστημα. Ένας άλλος λόγος είναι ότι αν ο χρήστης μπαίνει σε ένα χώρο που δεν θέλει να είναι, α θέλουν να βγει από αυτό με έναν ασφαλή και προβλέψιμο τρόπο.

**Actions for Multiple Objects**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς μπορεί το σύστημα να κάνει τις επαναλαμβανόμενες εργασίες ευκολότερες για το χρήστη;

Πότε, πώς και γιατί: Όταν το σύστημα περιέχει πολλά πραγματικά ή εικονικά αντικείμενα, όπως αρχεία, ή CD κλπ. Υπάρχουν ενέργειες που πρέπει να εκτελεστούν σε αυτά τα αντικείμενα, και οι χρήστες είναι πιθανό να θέλουν να εκτελέσουν ενέργειες σε δύο ή περισσότερα αντικείμενα τον ίδιο χρόνο. Στη συνέχεια, επιτρέπει στη διεργασία να εκτελείται "παράλληλα" σε ένα σύνολο από το χρήστη επιλεγμένων αντικειμένων. Ένας λόγος για αυτό είναι ότι οι άνθρωποι δεν είναι καλοί σε επαναλαμβανόμενες εργασίες ενώ οι μηχανές είναι. Ένας άλλος λόγος είναι ότι οι χρήστες δεν θέλουν να σπαταλούν το χρόνο τους εκτελούν την ίδια διεργασία για κάθε αντικείμενο ξεχωριστά.

**Toolbox**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς μπορεί το σύστημα να παρουσιάσει τις ενέργειες που ο χρήστης μπορεί να κάνει;

Πότε, πώς και γιατί: Όταν το σύστημα υποστηρίζει τη δημιουργία άλλων αντικειμένων, όπως ο WYSIWYG επεξεργαστής. Στη συνέχεια, έχουμε τα εργαλεία μαζί, τα βάζουμε φυσικά κοντά στην επιφάνεια εργασίας του χρήστη και βεβαιωνόμαστε ότι είναι διαφορετικά από άλλες ενέργειες που ο χρήστης μπορεί να κάνει. Αν τα εργαλεία μπορούν να παρουσιαστούν σαν εικονίδια και όχι σαν λέξεις, χρησιμοποιούμε εικονίδια. Ο λόγος για αυτό είναι ότι τα εργαλεία που χρησιμοποιούνται για να δημιουργήσουν τα αντικείμενα είναι διαφορετικά σε είδος από τις συνήθεις ενέργειες που εκτελούνται από τα υπάρχοντα αντικείμενα. Ένας άλλος λόγος είναι ότι ένας έμπειρος χρήστης χρειάζεται τα απαραίτητα εργαλεία έτοιμα στο χέρι.

**User's Annotations**

Συγγραφέας: Jennifer Tidwell

Πρόβλημα: Πώς μπορεί το σύστημα να συμβάλει στη διατήρηση της κατανόησης του χρήστη από τη μία λειτουργία στην άλλη;

Πότε, πώς και γιατί: Όταν το σύστημα είναι πολύπλοκο και δύσκολο για να το μάθει κανείς, αλλά θα πρέπει να χρησιμοποιηθεί ξανά από τον ίδιο χρήστη ή από άλλους. Στη συνέχεια, εισηγείται τρόπους για τους χρήστες να προσθέτουν τα δικά τους σχόλια και άλλα σχόλια στο σύστημα. Επιτρέπει στους χρήστες να τοποθετούν τα σχόλια φυσικά κοντά εκεί όπου χρειάζονται, και αν είναι δυνατόν, επιτρέπει απλά σχέδια εκτός κειμένου. Ένας λόγος για αυτό είναι ότι οι χρήστες γνωρίζουν καλύτερα τι είναι αξέχαστο γι'αυτούς σε σχέση με το τι

κάνει το σύστημα. Ένας άλλος λόγος είναι ότι η σωστή χωρική τοποθέτηση του κειμένου βοήθειας μπορεί να αυξήσει την αποτελεσματικότητά του συστήματος.

**Interaction History**

Συγγραφέας: Jennifer Tidwell
Πρόβλημα: Μπορεί το σύστημα να παρακολουθεί τι κάνει ο χρήστης με αυτό; Αν ναι, πώς;
Πότε, πώς και γιατί: Όταν ο χρήστης πραγματοποιεί μια σειρά ενεργειών με το σύστημα, ή περιηγείται μέσα από αυτό. Στη συνέχεια, καταγράφεται η ακολουθία των αλληλεπιδράσεων ως "ιστορικό". Ένας λόγος για αυτό είναι ότι οι άνθρωποι ξεχνούν τις κουραστικές λεπτομέρειες. Οι χρήστες δεν μπορούν να θυμηθούν ακριβώς τι έχουν κάνει πρόσφατα με το σύστημα, και οι υπολογιστές είναι καλύτερα σε αυτό από ό,τι είναι οι άνθρωποι. Ένας άλλος λόγος είναι ότι ο χρήστης μπορεί να θέλει να γνωρίζει τι ακριβώς έχει κάνει, , έτσι ώστε να μπορείτε να αναιρέσει την εργασία του ή να πάει πίσω.

**Important Message**

Συγγραφέας: Jennifer Tidwell
Πρόβλημα: Πώς θα πρέπει το σύστημα να μεταφέρει ένα μήνυμα έτσι ώστε να πιάσει την προσοχή του χρήστη αμέσως;
Πότε, πώς και γιατί: Όταν ο χρήστης πρέπει να ενημερώνεται αμέσως κατά τη διάρκεια της χρήσης του συστήματος. Στη συνέχεια, διακόπτεται ό,τι κάνει ο χρήστης με το μήνυμα, χρησιμοποιώντας εικόνα και ήχο, αν είναι δυνατόν. Το μήνυμα μεταδίδεται σε σαφή, σύντομη γλώσσα που μπορεί να γίνει κατανοητή αμέσως, και παρέχονται πληροφορίες σχετικά με το πώς να διορθώσει την κατάσταση, εκτός αν η πολιτισμική έννοια του μη-γλωσσικού μηνύματος είναι τόσο ισχυρή ώστε να μην μπορεί να εκληφθεί. Ένας λόγος για αυτό είναι ότι ο χρήστης πληρώνει πιθανώς την προσοχή του σε κάτι άλλο κατά τη στιγμή που συμβαίνει το γεγονός. Ένας άλλος λόγος είναι ότι ο χρήστης μπορεί να βρίσκεται κάτω από την πίεση του χρόνου ή του άγχους, πιθανώς ως αποτέλεσμα του ίδιου του μηνύματος, έτσι δεν θα είναι σε καλή θέση για να σταματήσει και να σκεφτεί πώς να αντιδράσει.

**Grid Layout**

Συγγραφέας: Martijn van Welie
Πρόβλημα: Ο χρήστης πρέπει να καταλάβει γρήγορα τις πληροφορίες και να αναλάβει δράση ανάλογα με τις εν λόγω πληροφορίες.

Πότε, πώς και γιατί: Όταν υπάρχουν διάφορα αντικείμενα πληροφοριών που παρουσιάζονται και είναι διατεταγμένα χωρικά σε μια περιορισμένη περιοχή. Συνήθως στο σχεδιασμό των οθονών, εντύπων και ιστοσελίδων. Στη συνέχεια, τακτοποιούνται όλα τα αντικείμενα σε ένα πλέγμα με τον ελάχιστο αριθμό γραμμών και στηλών, καθιστώντας τα κελιά όσο το δυνατόν μεγαλύτερα. Ένας λόγος για αυτό είναι ότι ο χρήστης πρέπει να δει πολλά αντικείμενα, αλλά θέλει να τα δει με σαφή και οργανωμένο τρόπο.

**Progress**

Συγγραφέας: Martijn van Welie
Πρόβλημα: Ο χρήστης επιθυμεί να γνωρίζει αν ή όχι η λειτουργία εξακολουθεί να εκτελείται, καθώς και πόσο καιρό θα χρειαστεί να περιμένει.
Πότε, πώς και γιατί: Όταν μια εργασία του συστήματος χρειάζεται αρκετό χρόνο για να εκτελεστεί(συνήθως περισσότερο από μερικά δευτερόλεπτα) και πρέπει να ολοκληρωθεί πριν ξεκινήσει η επόμενη. Στη συνέχεια, δείχνει ότι η εργασία εξακολουθεί να εκτελείται και δίνεται μια ένδειξη της εξέλιξης της εργασίας. Ένας λόγος για αυτό είναι ότι οι χρήστες μπορεί να μην είναι εξοικειωμένοι με την πολυπλοκότητα του έργου. Ένας άλλος λόγος είναι ότι σε ορισμένες περιπτώσεις ο χρήστης μπορεί να θέλει να τερματίσει την λειτουργία, γιατί θα πάρει πάρα πολύ χρόνο.

**Preferences**

Συγγραφέας: Martijn van Welie
Πρόβλημα: Κάθε χρήστης είναι διαφορετικός και προτιμά να κάνει τα πράγματα λίγο διαφορετικά.
Πότε, πώς και γιατί: Όταν η εφαρμογή είναι πολύ περίπλοκη και πολλές από τις λειτουργίες της μπορούν να ρυθμιστούν με τις προτιμήσεις του χρήστη. Θα πρέπει να γίνουν αρκετά γνωστές οι προτιμήσεις του χρήστη, προκειμένου να αναληφθούν προεπιλογές που θα ταιριάζουν σε όλους τους χρήστες. Οι πιθανοί χρήστες μπορεί να είναι αρχάριοι ή έμπειροι. Στη συνέχεια, επιτρέπεται στο χρήστη να προσαρμόσει τις προτιμήσεις του. Ένας λόγος για αυτό είναι ότι ο χρήστης μπορεί να βελτιώσει την ικανοποίηση από τη χρήση του συστήματος. Ένας άλλος λόγος είναι ότι αυτό δίνει στο χρήστη τη δυνατότητα να αλλάξει μόνο ένα υποσύνολο όλων των επιλογών.

**Like in the real world**

Συγγραφέας: Martijn van Welie

Πρόβλημα: Ο χρήστης πρέπει να ξέρει πώς να ελέγξει ένα αντικείμενο στη διεπιφάνεια που μοιάζει με ένα αντικείμενο που ο χρήστης γνωρίζει από τον πραγματικό κόσμο.

Πότε, πώς και γιατί: Όταν έχουμε να κάνουμε με μια εφαρμογή που χρησιμοποιεί μεταφορές του πραγματικού κόσμου και άμεσου χειρισμού των διεπιφανειών. Τα αντικείμενα μέσα στις διεπιφάνειες μοιάζουν με αντικείμενα του πραγματικού κόσμου και η αλληλεπίδραση εισηγείται τα αντικείμενα να μοιάσουν στον πραγματικό κόσμο της αλληλεπίδρασης. Στη συνέχεια, ταιριάζουμε τη συσκευή εισόδου και το widget που χρησιμοποιείται. Χρησιμοποιούμε ένα συνδυασμό του widget και της συσκευής εισόδου που ταιριάζουν σε όλες τις πιθανές μετακινήσεις. Ένας λόγος για αυτό είναι ότι η αλληλεπίδραση γίνεται με συσκευές εισόδου, αλλά κάθε συσκευή διαθέτει ένα περιορισμένο σύνολο των δυνατοτήτων χειραγώγησης όπως η κίνηση, περιστροφή και οι βαθμοί ελευθερίας. Ένας άλλος λόγος είναι ο τρόπος που χειρίζονται τα αντικείμενα στον πραγματικό κόσμο. Αυτό δημιουργεί προσδοκίες για το πώς η αλληλεπίδραση θα πραγματοποιηθεί.

**Favourites**

Συγγραφείς: Martijn van Welie, Hallvard Traetteberg

Πρόβλημα: Ο χρήστης πρέπει να βρει ένα αντικείμενο που χρησιμοποιείται τακτικά σε ένα μεγάλο σύνολο των αντικειμένων.

Πότε, πώς και γιατί: Όταν ο χρήστης ψάχνει για ένα στοιχείο που περιέχεται σε ένα μεγάλο σύνολο των αντικειμένων. Το αντικείμενο έχει σημασία και ο χρήστης το αναζητά τακτικά. Τα αντικείμενα είναι συνήθως τα αρχεία, τα χρώματα, οι ιστοσελίδες ή τα αρχεία βάσεων δεδομένων και αποτελούν μέρος μεγάλων συλλογών, αντίστοιχα, ένα σύστημα αρχείων, το διαδίκτυο ή μια βάση δεδομένων. Στη συνέχεια, επιτρέπεται στο χρήστη να χρησιμοποιεί τα αγαπημένα αντικείμενα της επιλογής του. Ένα αγαπημένο είναι μια ετικέτα που οδηγεί απευθείας στο συγκεκριμένο σημείο. Ένας λόγος για αυτό είναι ότι ο χρήστης γνωρίζει την ύπαρξη του αντικειμένου και το χρησιμοποιεί τακτικά, ως εκ τούτου, τα αντικείμενα πρέπει να είναι στο χέρι. Ένας άλλος λόγος είναι ότι ο χρήστης ενδιαφέρεται για το αντικείμενο, αλλά μπορεί να χρησιμοποιήσει το αντικείμενο μόνο για ένα μικρό χρονικό διάστημα.

**Warning**

Συγγραφείς: Martijn van Welie, Hallvard Traetteberg, David KaneΑπό το σημείο αυτό

Πρόβλημα: Ο χρήστης μπορεί να προκαλέσει ακούσια ένα πρόβλημα, το οποίο πρέπει να επιλυθεί.

Πότε, πώς και γιατί: Σε καταστάσεις που προκύπτουν όταν ο χρήστης εκτελεί μια ενέργεια που μπορεί να οδηγήσει ακούσια σε ένα πρόβλημα. Στη συνέχεια, προειδοποιεί το χρήστη πριν συνεχίσει την εργασία του και του δίνει την δυνατότητα να εγκαταλείψει όλες τις εργασίες του. Ένας λόγος για αυτό είναι ότι σημαντική δουλειά μπορεί να χαθεί εάν η ενέργεια έχει ολοκληρωθεί πλήρως. Ένας άλλος λόγος είναι ότι ορισμένες ενέργειες είναι δύσκολο ή αδύνατο να ανακτηθούν.

**Hinting**

Συγγραφέας: Martijn van Welie
Πρόβλημα: Ο χρήστης πρέπει να ξέρει πώς να επιλέξει τις λειτουργίες.
Πότε, πώς και γιατί: Όταν η λειτουργία σε μια εφαρμογή είναι προσβάσιμη με περισσότερους από έναν τρόπους, π.χ. μέσω του μενού, χρησιμοποιώντας τις συντομεύσεις πληκτρολογίου, είτε μέσω εργαλείων. Στη συνέχεια, δίνει στο χρήστη συμβουλές για άλλους τρόπους για να αποκτήσει πρόσβαση στην ίδια λειτουργία. Ένας λόγος για αυτό είναι ότι ο χρήστης χρειάζεται κάποιο τρόπο για να ανακαλύψει και να μάθει αυτούς τους εναλλακτικούς και πιθανών πιο αποτελεσματικούς τρόπους.

## 5.5 Αξιολόγηση αλληλεπιδραστικών προτύπων

Όπως είδαμε πιο πάνω τα αλληλεπιδραστικά πρότυπα κατηγοριοποιούνται με βάση την ευχρηστία(usability), την αποτελεσματικότητα(effectivity), την εμπειρία(experience), το στυλ αλληλεπίδρασης(interaction style) και το επίπεδο διεπιφάνειας(interface level).
Στους πιο κάτω πίνακες θα αξιολογήσουμε τα αλληλεπιδραστικά πρότυπα με βάση τις πιο πάνω παραμέτρους για να δούμε πια από τα χαρακτηριστικά κάθε παραμέτρου ισχύουν για κάθε πρότυπο και ποια όχι.

**Πίνακας 1:Αξιολόγηση βάση ευχρηστίας(usability).**

| Pattern | Visibility | Constraints | Affordance | Mappings | Feedback |
|---|---|---|---|---|---|
| Narrative | OXI | OXI | OXI | NAI | OXI |
| Form | OXI | OXI | OXI | NAI | OXI |
| Control Panel | OXI | OXI | NAI | NAI | NAI |
| Composed Command | NAI | NAI | NAI | NAI | NAI |

| | | | | | |
|---|---|---|---|---|---|
| Navigable Spaces | NAI | OXI | NAI | NAI | NAI |
| Step-by-Step Instructions | NAI | OXI | NAI | OXI | NAI |
| Small Groups of Related Things | OXI | OXI | NAI | NAI | NAI |
| Hierarchical Set | OXI | NAI | NAI | OXI | NAI |
| Optional Detail On Demand | OXI | NAI | NAI | NAI | NAI |
| Short Description | NAI | NAI | NAI | NAI | NAI |
| Go Back to a Safe Place | NAI | NAI | NAI | NAI | NAI |
| Actions for Multiple Objects | NAI | NAI | NAI | NAI | NAI |
| Toolbox | OXI | OXI | NAI | NAI | NAI |
| User's Annotations | NAI | NAI | NAI | OXI | NAI |
| Interaction History | NAI | NAI | NAI | NAI | NAI |
| Important Message | OXI | NAI | NAI | NAI | NAI |
| Grid Layout | OXI | OXI | NAI | NAI | NAI |
| Progress | NAI | NAI | NAI | NAI | OXI |
| Preferences | NAI | OXI | NAI | NAI | OXI |
| Like in the real world | NAI | NAI | NAI | OXI | NAI |
| Favourites | NAI | NAI | NAI | OXI | NAI |
| Warning | NAI | NAI | NAI | NAI | OXI |
| Hinting | NAI | NAI | NAI | NAI | NAI |

**Πίνακας 2:Αξιολόγηση βάση αποτελεσματικότητας(effectivity).**

| **Pattern** | **Time to Learn** | **Speed of Performance** | **Rate of Errors by Users** | **Retention over Time** | **Subjective Satisfaction** |
|---|---|---|---|---|---|
| Narrative | NAI | NAI | NAI | NAI | OXI |
| Form | OXI | OXI | NAI | NAI | NAI |
| Control Panel | NAI | OXI | NAI | OXI | NAI |
| Composed Command | NAI | OXI | NAI | NAI | OXI |
| Navigable Spaces | OXI | NAI | NAI | OXI | OXI |
| Step-by-Step Instructions | OXI | NAI | OXI | NAI | NAI |
| Small Groups of Related Things | OXI | NAI | NAI | OXI | NAI |
| Hierarchical Set | OXI | NAI | NAI | OXI | OXI |
| Optional Detail On Demand | NAI | NAI | NAI | NAI | NAI |
| Short Description | OXI | OXI | OXI | NAI | NAI |
| Go Back to a Safe Place | NAI | NAI | NAI | NAI | OXI |
| Actions for Multiple Objects | NAI | OXI | NAI | NAI | OXI |
| Toolbox | OXI | OXI | NAI | OXI | NAI |
| User's Annotations | NAI | NAI | NAI | OXI | NAI |
| Interaction History | NAI | OXI | OXI | NAI | OXI |
| Important Message | NAI | NAI | OXI | NAI | OXI |

| | | | | | |
|---|---|---|---|---|---|
| Grid Layout | OXI | OXI | NAI | NAI | NAI |
| Progress | NAI | NAI | NAI | NAI | OXI |
| Preferences | NAI | NAI | NAI | OXI | OXI |
| Like in the real world | NAI | NAI | NAI | OXI | OXI |
| Favourites | NAI | OXI | OXI | NAI | OXI |
| Warning | NAI | NAI | OXI | OXI | NAI |
| Hinting | OXI | NAI | NAI | NAI | OXI |

**Πίνακας 3:Αξιολόγηση βάση εμπειρίας(experience)**

| **Pattern** | **Novice or First-time Users** | **Knowledgeable Intermittent Users** | **Expert Frequent Users** |
|---|---|---|---|
| Narrative | OXI | OXI | NAI |
| Form | OXI | OXI | OXI |
| Control Panel | OXI | OXI | OXI |
| Composed Command | NAI | NAI | OXI |
| Navigable Spaces | OXI | OXI | OXI |
| Step-by-Step Instructions | OXI | OXI | NAI |
| Small Groups of Related Things | OXI | OXI | OXI |
| Hierarchical Set | OXI | OXI | OXI |
| Optional Detail On Demand | OXI | OXI | OXI |
| Short Description | OXI | OXI | NAI |
| Go Back to a Safe Place | OXI | OXI | NAI |
| Actions for Multiple Objects | OXI | OXI | OXI |
| Toolbox | OXI | OXI | OXI |
| User's Annotations | NAI | OXI | OXI |
| Interaction History | OXI | OXI | OXI |
| Important Message | OXI | OXI | OXI |

| | | | |
|---|---|---|---|
| Grid Layout | OXI | OXI | OXI |
| Progress | OXI | OXI | OXI |
| Preferences | OXI | OXI | OXI |
| Like in the real world | OXI | OXI | OXI |
| Favourites | OXI | OXI | OXI |
| Warning | OXI | OXI | NAI |
| Hinting | OXI | OXI | NAI |

**Πίνακας 4:Αξιολόγηση βάση στηλ αλληλεπίδρασης(interaction style).**

| **Pattern** | **Direct Manipulation** | **Menu Selection** | **Form Fillin** | **Command Language** | **Natural Language** |
|---|---|---|---|---|---|
| Narrative | OXI | OXI | OXI | NAI | OXI |
| Form | NAI | NAI | OXI | NAI | NAI |
| Control Panel | OXI | NAI | NAI | NAI | NAI |
| Composed Command | NAI | NAI | NAI | OXI | NAI |
| Navigable Spaces | OXI | NAI | NAI | NAI | NAI |
| Step-by-Step Instructions | OXI | NAI | OXI | NAI | NAI |
| Small Groups of Related Things | OXI | OXI | OXI | NAI | NAI |
| Hierarchical Set | OXI | OXI | NAI | NAI | NAI |
| Optional Detail On Demand | OXI | NAI | NAI | NAI | NAI |
| Short Description | OXI | NAI | NAI | NAI | NAI |
| Go Back to a Safe Place | OXI | NAI | NAI | NAI | NAI |

| | | | | | |
|---|---|---|---|---|---|
| Actions for Multiple Objects | OXI | NAI | NAI | NAI | NAI |
| Toolbox | OXI | NAI | NAI | NAI | NAI |
| User's Annotations | OXI | OXI | OXI | NAI | NAI |
| Interaction History | OXI | NAI | NAI | OXI | OXI |
| Important Message | OXI | OXI | OXI | OXI | OXI |
| Grid Layout | OXI | NAI | OXI | NAI | NAI |
| Progress | OXI | NAI | NAI | OXI | OXI |
| Preferences | OXI | NAI | NAI | NAI | NAI |
| Like in the real world | OXI | NAI | NAI | NAI | NAI |
| Favourites | OXI | NAI | NAI | NAI | NAI |
| Warning | OXI | OXI | OXI | OXI | OXI |
| Hinting | OXI | NAI | NAI | NAI | NAI |

**Πίνακας 5:Αξιολόγηση βάση επιπέδου διεπιφάνειας(interface level)**

| **Pattern** | **Application** | **Containers** | **Widget** |
|---|---|---|---|
| Narrative | NAI | NAI | OXI |
| Form | NAI | OXI | NAI |
| Control Panel | NAI | OXI | NAI |
| Composed Command | OXI | NAI | NAI |
| Navigable Spaces | OXI | NAI | NAI |
| Step-by-Step Instructions | OXI | NAI | NAI |
| Small Groups of Related Things | NAI | OXI | NAI |
| Hierarchical Set | NAI | OXI | NAI |
| Optional Detail On Demand | NAI | OXI | NAI |

| | | | |
|---|---|---|---|
| Short Description | NAI | NAI | OXI |
| Go Back to a Safe Place | OXI | NAI | NAI |
| Actions for Multiple Objects | NAI | OXI | NAI |
| Toolbox | NAI | OXI | NAI |
| User's Annotations | OXI | NAI | NAI |
| Interaction History | OXI | NAI | NAI |
| Important Message | OXI | NAI | NAI |
| Grid Layout | NAI | OXI | NAI |
| Progress | NAI | NAI | OXI |
| Preferences | OXI | NAI | NAI |
| Like in the real world | NAI | NAI | OXI |
| Favourites | OXI | NAI | NAI |
| Warning | OXI | NAI | NAI |
| Hinting | NAI | NAI | OXI |

# Κεφάλαιο 6

## Σχεδιαστικά πρότυπα για συστήματα πραγματικού χρόνου


6.1 Εισαγωγή 50
6.2 Η ανάγκη για κοινή γλώσσα στα συστήματα πραγματικού χρόνου 50
6.3 Πρότυπα επαναχρησιμοποίησης 51
6.4 Πρότυπα κατανεμημένων συστημάτων 55
6.5 Πρότυπα ασφάλειας και αξιοπιστίας 57
6.6 Απλά πρότυπα 59


### 6.1 Εισαγωγή

Τα συστήματα πραγματικού χρόνου αποτελούνται συνήθως από ένα συνδυασμό των συστατικών του υλικού και του λογισμικού που πρέπει να συγχρονιστούν για να συντονίζουν τις διάφορες διεργασίες και δραστηριότητες. Από τα πρώτα στάδια του σχεδιασμού, πολλές αποφάσεις πρέπει να ληφθούν για να εξασφαλιστεί ότι η υλοποίηση μιας τέτοιας πολύπλοκης μηχανής πληροί όλες τις λειτουργικές και μη λειτουργικές ως προς το χρόνο απαιτήσεις.
Ένα από τα κύρια προβλήματα αφορά την πλέον κατάλληλη αρχιτεκτονική του συστήματος, όπου όλες οι απαιτήσεις θα πληρούνται. Έτσι, λόγω της ανομοιογένειας και της πολυπλοκότητας των συστημάτων, για την ανάλυση των διαφόρων πτυχών, διαφορετικά μοντέλα πρέπει να κατασκευαστούν, το ταίριασμα των οποίων δεν θα είναι καθόλου εύκολο.

### 6.2 Η ανάγκη για κοινή γλώσσα στα συστήματα πραγματικού χρόνου

Ένα ενιαίο μοντέλο, που καλύπτει όλα τα ενδιαφέροντα στοιχεία, είναι πράγματι αναγκαίο για την επιτάχυνση της διαδικασίας σχεδιασμού. Για το λόγο αυτό, η Ενοποιημένη Γλώσσα

Μοντελοποίησης (UML), για την οποία αναφερθήκαμε στην εισαγωγή μας, αν και σχεδιάστηκε για αντικειμενοστρεφείς προδιαγραφές λογισμικού, επεκτάθηκε για να συμπεριλάβει και τα συστήματα πραγματικού χρόνου(real-time systems).

Κατά την ανάπτυξη νέων συστημάτων, κάποια προβλήματα συναντούνται ξανά και ξανά, επομένως οι έμπειροι σχεδιαστές καλούνται να εφαρμόσουν τις λύσεις, για τις οποίες εργάστηκαν στο παρελθόν. Με την αύξηση στο ρυθμό ανάπτυξης των συστημάτων πραγματικού χρόνου, τα σχεδιαστικά πρότυπα ήταν απαραίτητα για την αντιμετώπιση ζητημάτων όπως συγχρονισμό(concurrency), κατανομή των πόρων(resource sharing), και διανομή(distribution). Καθώς η UML έχει καθιερωθεί ως η επίσημη γλώσσα για την μοντελοποίηση, τα πρότυπα αυτά θα περιγραφούν στη UML. Τα πρότυπα σχεδίασης αναφέρονται σε προβλήματα που ανέκυψαν κατά τη διαδικασία σχεδιασμού, αλλά τα προβλήματα αυτά εμφανίζονται και στον προσδιορισμό των στοιχείων που απαντώνται συχνά σε σύνθετα συστήματα. Αν και τα συστατικά των συστημάτων που αναλύθηκαν παρουσιάζουν κάποια κοινά στοιχεία για όλα τα συστήματα πραγματικού χρόνου (π.χ. χαρακτηριστικά των διεργασιών όπως η περιοδικότητα ή απεριοδικότητα, επεξεργαστές, και χρονοδρομολογητές), έχουν κατασκευαστεί από την αρχή κάθε φορά και παρόμοια προβλήματα συναντώνται ξανά και ξανά.

Παρακάτω θα δούμε τα βασικότερα πρότυπα που έχουν σχεδιαστεί έτσι ώστε να αντιμετωπιστούν αυτά τα προβλήματα.

### 6.3 Πρότυπα επαναχρησιμοποίησης(reuse patterns)

#### Πρότυπο Microkernel

Το πρότυπο αυτό διαχωρίζει την αρχιτεκτονική του λογισμικού σε ένα σύνολο στρωμάτων(layers). Τα στρώματα είναι τοποθετημένα ως ένα σύνολο client-server ενώσεων στις οποίες τα ανώτερα στρώματα μπορούν να καλούν τις υπηρεσίες των κατώτερων στρωμάτων, αλλά όχι το αντίστροφο.

Η οργάνωση αυτή έχει πολλά πλεονεκτήματα όπως:

Φορητότητα(portability):Ενισχύεται η επαναχρησιμοποίηση σε διαφορετικές πλατφόρμες, αφού οι λεπτομέρειες του υλικού απομονώνονται στα χαμηλότερα στρώματα.

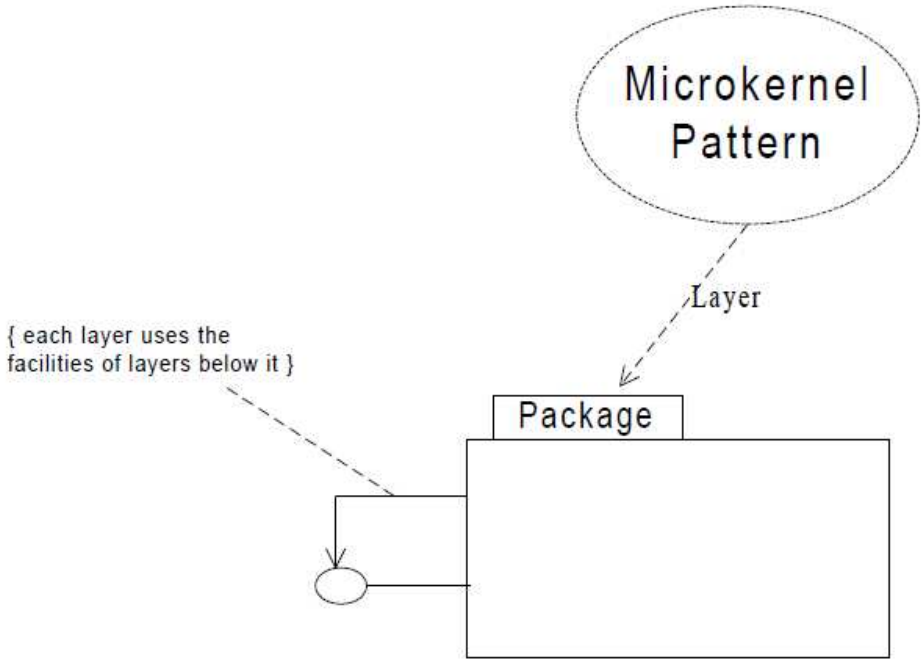
Επαναχρησιμοποίηση σε διαφορετικές εφαρμογές ενισχύεται, επειδή τα ανώτερα στρώματα απομονώνουν την εφαρμογή μακριά από τα χαμηλότερα στρώματα.

Επεκτασιμότητα στις εφαρμογές είναι δυνατή αν εξαιρέσουμε μερικά από τα στρώματα.

Μια κλειστή-διαστρωματωμένη αρχιτεκτονική σημαίνει ότι κάθε στρώμα μπορεί να καλέσει μόνο τις υπηρεσίες του στρώματος του που βρίσκεται αμέσως κάτω από αυτό. Σε μια

ανοιχτή διαστρωματωμένη αρχιτεκτονική, ένα στρώμα μπορεί να καλεί τις υπηρεσίες σε κάθε στρώμα που βρίσκεται κάτω από αυτό.

Γενική δομή Microkernel:

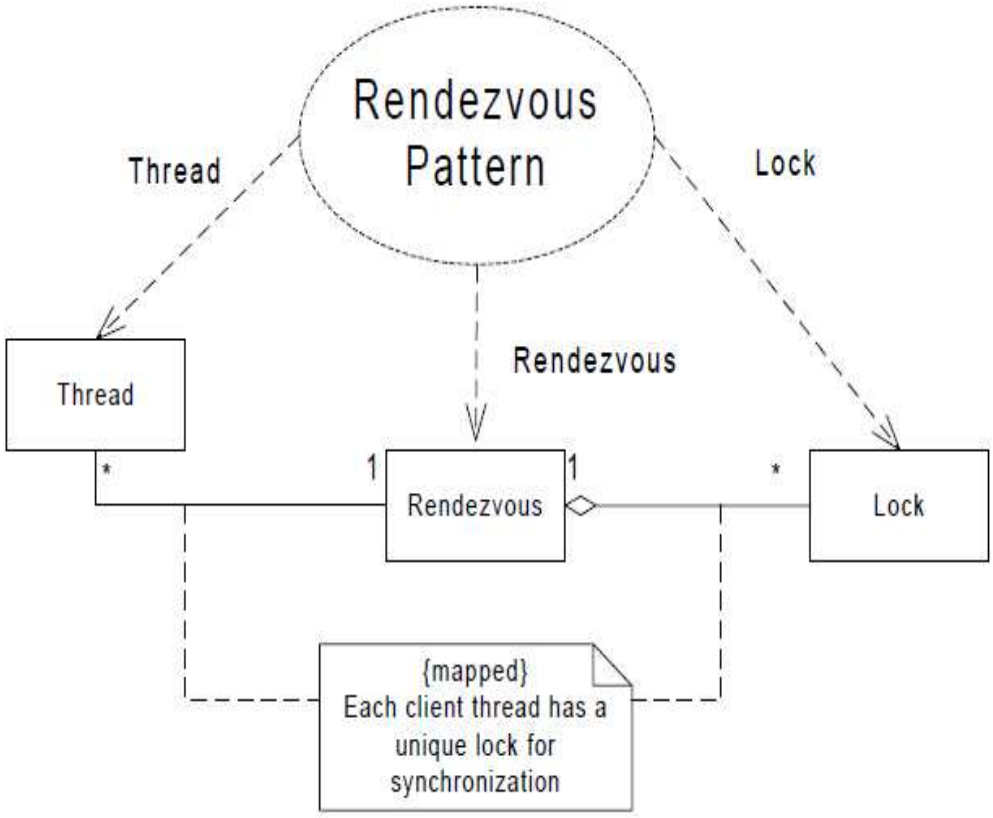



**Πρότυπο Rendezvous**

Το πρότυπο αυτό αναφέρεται στο συγχρονισμό των ταυτόχρονων εργασιών. Στην πραγματικότητα, η χρήση ενός αμοιβαίου αποκλεισμού ονομάζεται μονομερές ραντεβού(unilateral rendezvous). Η πιο κοινή χρήση του όρου αναφέρεται στον συγχρονισμό περισσότερων από μία εργασιών.

Το πρότυπο αυτό αποτελείται από ένα αντικείμενο συντονισμού(rendezvous object), τους πελάτες(clients) που πρέπει να συγχρονίζονται και τους σημαφόρους κλειδώματος(mutex blocking semaphores). Οι σημαφόροι αυτοί χρησιμοποιούνται για να αναγκάσουν τα νήματα(threads) να περιμένουν μέχρι να πληρούνται όλες οι προσυνθήκες(preconditions).

Γενική δομή Rendezvous:




**Πρότυπο Interface**

Υπάρχει ένας αριθμός περιπτώσεων στις οποίες γίνεται ρητή διάκριση του interface και η υλοποίηση είναι χρήσιμη.

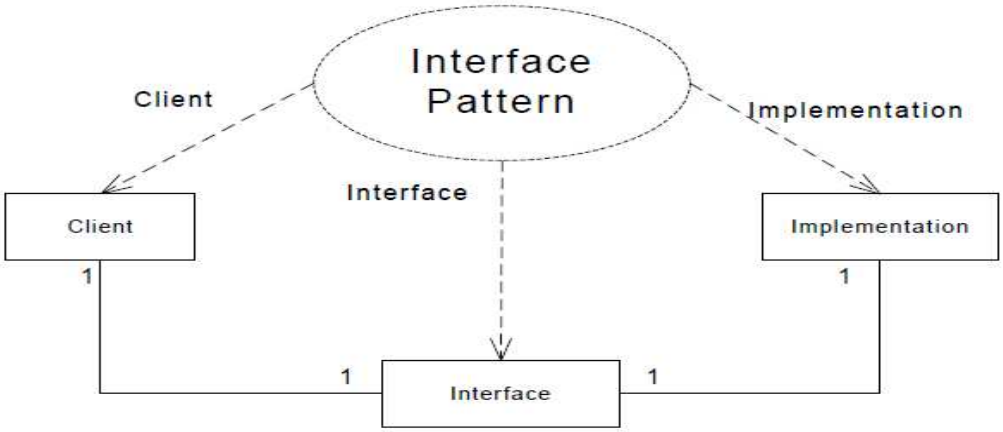
Πρώτον, μια κοινή εφαρμογή μπορεί να είναι κατάλληλη για διάφορες χρήσεις. Αν η κλάση θα μπορούσε να παρέχει διαφορετικά interfaces, μια ενιαία εφαρμογή θα μπορούσε να καλύπτει αρκετές ανάγκες. Για παράδειγμα, πολλές κοινές δομές της επιστήμης των υπολογιστών, όπως τα δέντρα, οι ουρές και οι στοίβες μπορούν πραγματικά να χρησιμοποιούν μια ενιαία βασική εφαρμογή, όπως μια συνδεδεμένη λίστα. Η πολυπλοκότητα της εφαρμογής μπορεί να χρησιμοποιηθεί με διαφορετικούς τρόπους για να εφαρμόσει την επιθυμητή συμπεριφορά, ακόμα και αν οι τελικοί πελάτες είναι ανίδεοι για το τι συμβαίνει.

Δεύτερον, με το διαχωρισμό ενός interface, γίνεται ευκολότερο να αλλάξουμε μια υλοποίηση ή να προσθέσουμε νέα και διαφορετικά interfaces. Έτσι δημιουργείται η ανάγκη να προσθέσουμε ένα νέο container, δημιουργώντας ένα interface που παρέχει τις σωστές υπηρεσίες προς τον πελάτη, αλλά εφαρμόζει τις εν λόγω υπηρεσίες από την άποψη των στοιχειωδών πράξεων των υπαρχόντων containers.

Τέλος, υπάρχουν μερικές φορές που θέλουμε διαφορετικά επίπεδα πρόσβασης στα εσωτερικά ενός αντικειμένου. Διαφορετικά interfaces μπορούν να κατασκευαστούν για να παρέχονται διαφορετικά επίπεδα πρόσβασης για διαφορετικά αντικείμενα και περιβάλλοντα πελάτη. Για παράδειγμα, μπορεί να θέλουμε ένα σύνολο λειτουργιών που διατίθενται στους

χρήστες να είναι μόνο ένα υποσύνολο του συνόλου των λειτουργιών, ενώ σε ένα διαφορετικό σύνολο πρέπει να παρέχεται σε "service mode" και ένα πολύ διαφορετικό σύνολο να παρέχεται σε "remote debugging mode".
Το πρότυπο Interface λύνει όλα αυτά τα προβλήματα. Πρόκειται για ένα πολύ απλό πρότυπο, τόσο κοινό ώστε η UML παρέχει πραγματικά το στερεότυπο «τύπο» στην καθορισμένη γλώσσα.

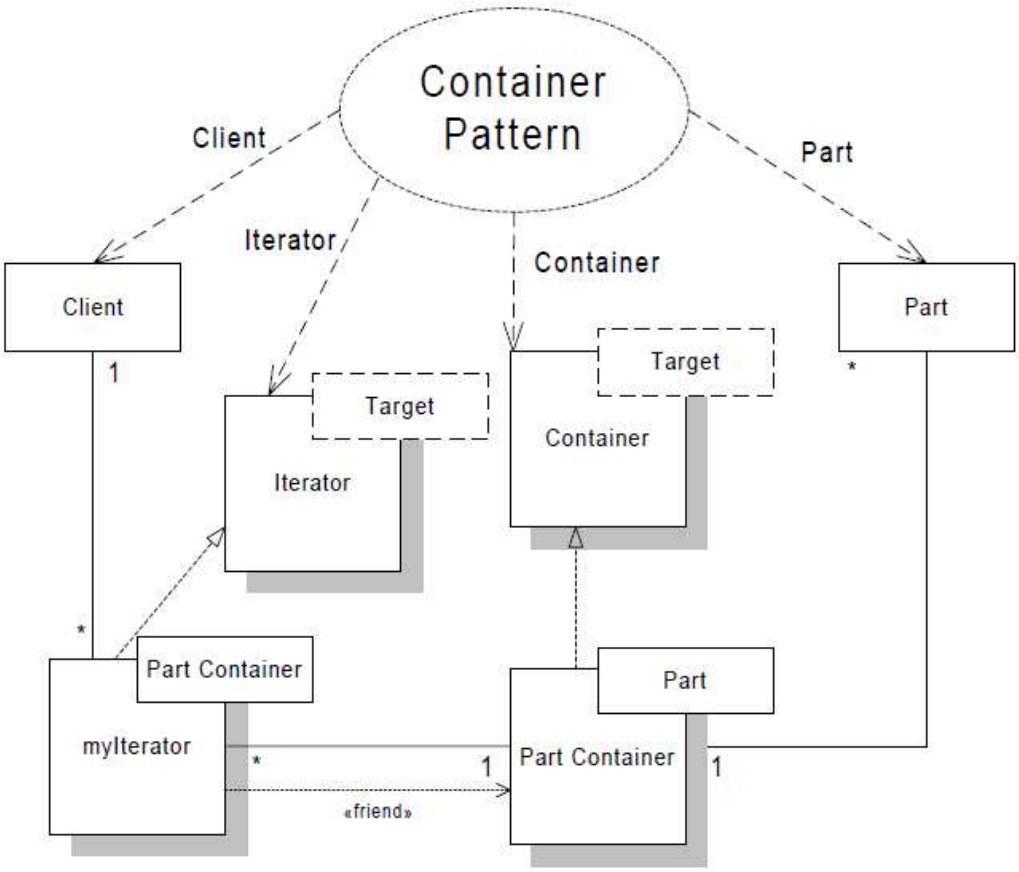
Γενική δομή Interface:




**Πρότυπο Container**

Όταν ένα αντικείμενο έχει ένα 1-προς-πολλά συσχέτιση, τίθεται το ερώτημα ως προς τον ακριβή μηχανισμό όπου η κλάση "1" θα χρησιμοποιήσει για να έχει πρόσβαση στα "πολλά" αντικείμενα. Μια λύση είναι να χτίσουμε χαρακτηριστικά στην κλάση "1" να διαχειριστεί το σύνολο των αντικειμένων που περιέχονται. Αυτό πετυχαίνεται με την προσθήκη των λειτουργιών όπως add(), remove(), first(), next(), last(), and find(). Η κοινή λύση είναι να εισαγάγουμε ένα container object μεταξύ του "1" και το "πολλά".
Προσθέτοντας ένα container object για τη διαχείριση των συσχετιζόμενων αντικειμένων δεν μπορεί να λυθεί το συνολικό πρόβλημα, επειδή συχνά το container πρέπει να έχει πρόσβαση από πολλούς clients. Για να απαλλαγούμε από αυτό, κάποια iterators χρησιμοποιούνται σε συνδυασμό με τα containers. Ένα iterator παρακολουθεί όταν ο client είναι στο container. Διαφορετικοί clients χρησιμοποιούν διαφορετικά iterators ενώ ένας μόνο client μπορεί να χρησιμοποιήσει διαφορετικά iterators. Η πρότυπη βιβλιοθήκη προτύπων (STL), ένα μέρος της ANSI C + +, παρέχει πολλά διαφορετικά containers με μια ποικιλία των iterators, όπως first, last, κ.λ.π.

Γενική δομή Container:




### 6.4 Πρότυπα κατανεμημένων συστημάτων

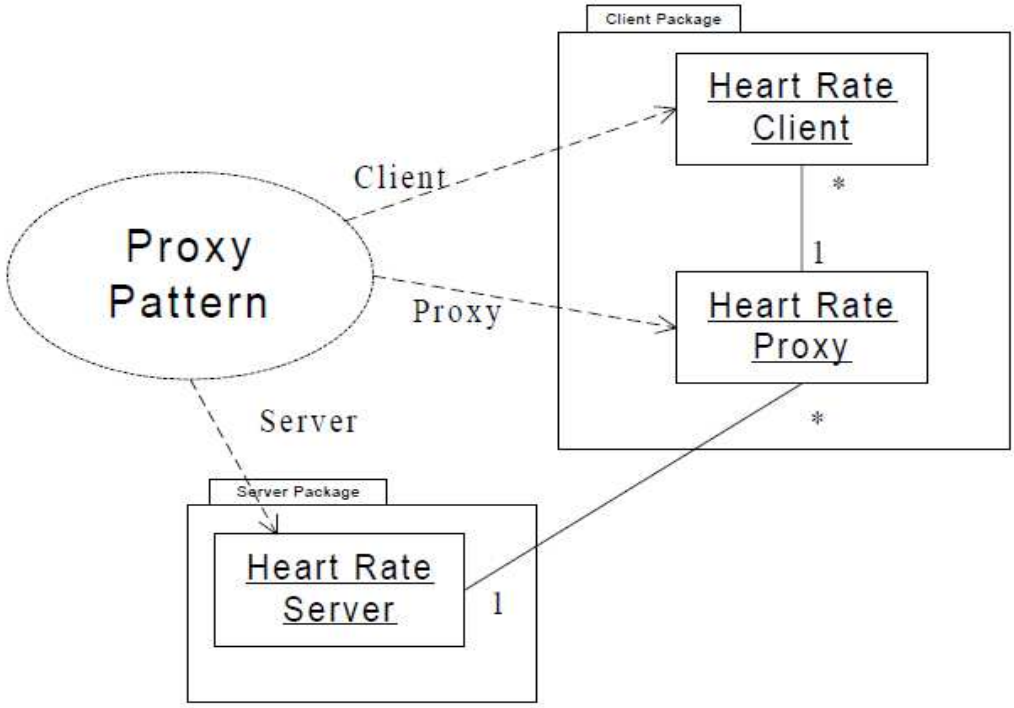
#### Πρότυπο Proxy

Σε πολλά ενσωματωμένα συστήματα δεδομένων έναw μόνο αισθητήρα χρησιμοποιείται από πολλούς πελάτες(clients) οι οποίοι μένουν σε διαφορετικό χώρο διευθύνσεων (χώρος εργασίας ή επεξεργαστή). Η αφελής προσέγγιση στο πρόβλημα αυτό είναι κάθε πελάτης να έχει τη δυνατότητα να ζητεί τα δεδομένα από το διακομιστή δεδομένων(data server). Αυτό είναι προβληματικό, διότι εάν τα χαρακτηριστικά του απομακρυσμένου διακομιστή αλλάζουν, κάθε πελάτης πρέπει να ενημερώνεται.

Το πρότυπο Proxy λύνει αυτό το πρόβλημα χρησιμοποιώντας ένα τοπικό σταθμό για τον απομακρυσμένο διακομιστή δεδομένων, που ονομάζεται proxy. Ο proxy ενσωματώνει τις απαραίτητες πληροφορίες για να επικοινωνήσει με το server. Εν τω μεταξύ οι τοπικοί πελάτες(local clients)μπορούν να καλούν άμεσα το proxy για να λάβουν τα δεδομένα, αλλά εξακολουθούν να είναι αποσυνδεδεμένοι από τον απομακρυσμένο διακομιστή δεδομένων(remote data server). Ο πελάτης μπορεί να συνδεθεί με τον proxy, είτε καλώντας τον, όταν χρειάζεται τα δεδομένα, ή με την εφαρμογή callbacks. Αν μια στρατηγική callback

χρησιμοποιείται, τότε ο proxy χρησιμοποιεί μία από τις πολλές στρατηγικές για να καθορίσει πότε ο πελάτης(client) πρέπει να ενημερώνεται. Οι πιο γνωστές στρατηγικές είναι: περιοδική(periodic), επεισοδιακή(episodic) και επι-περιοδική(epi-periodic). Η στρατηγική της περιοδικής ενημέρωσης ενημερώνει τους πελάτες συχνά, έστω και αν τα δεδομένα δεν έχουν αλλάξει. Η στρατηγική της επεισοδιακή ενημέρωσης ενημερώνει τον πελάτη μόνο όταν τα δεδομένα αλλάζουν. Η στρατηγική της επι-περιοδικής ενημέρωσης ειδοποιεί τους πελάτες χρησιμοποιώντας ένα συνδυασμό των άλλων δύο στρατηγικών.

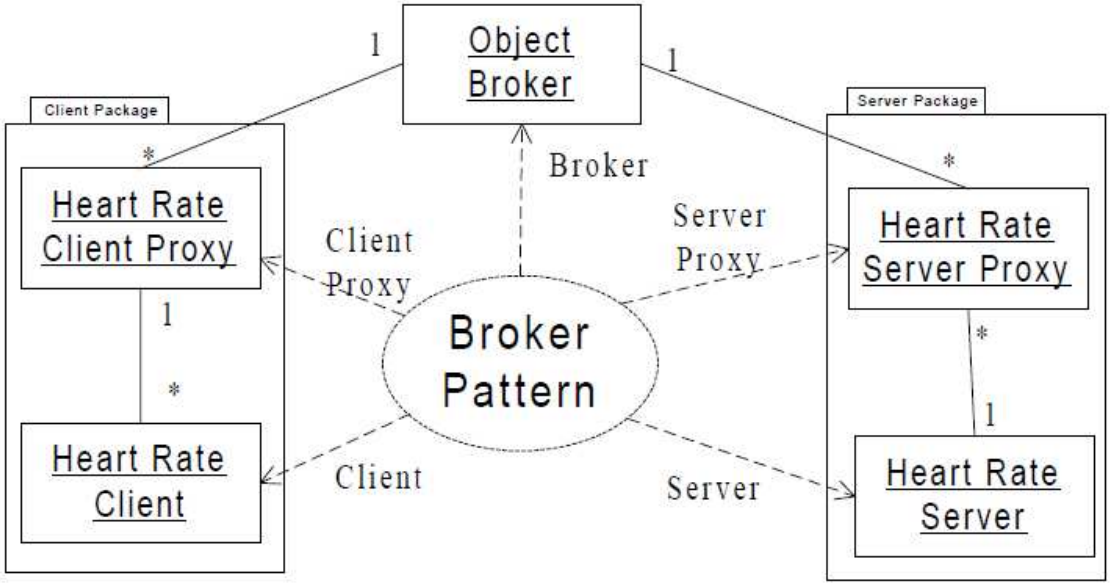
Γενική δομή Proxy:




**Πρότυπο Broker**

Το πρότυπο αυτό είναι βασισμένο στη λογική του proxy, αλλά πηγαίνει ένα ακόμη βήμα προς την κατεύθυνση αποσύνδεσης των πελατών(clients) από τους διακομιστές(servers). Ένα broker object είναι ένα αντικείμενο που γνωρίζει τη θέση των άλλων αντικειμένων. Μπορεί να έχει τη γνώση εκ των προτέρων (κατά τη στιγμή μεταγλώττισης) ή μπορεί να συγκεντρώσει τις πληροφορίες δυναμικά, καθώς το αντικείμενο εγγράφεται., ή με ένα συνδυασμό και των δύο. Το κύριο πλεονέκτημα του προτύπου Broker είναι ότι είναι δυνατόν να κατασκευάσει ένα Proxy Pattern, όταν η θέση του διακομιστή δεν είναι γνωστή όταν το σύστημα μεταγλωττίζεται. Αυτό το καθιστά ιδιαίτερα χρήσιμο για συστήματα που χρησιμοποιούν συμμετρική ή ημισυμμετρική πολυεπεξεργασία.

Γενική δομή Broker:

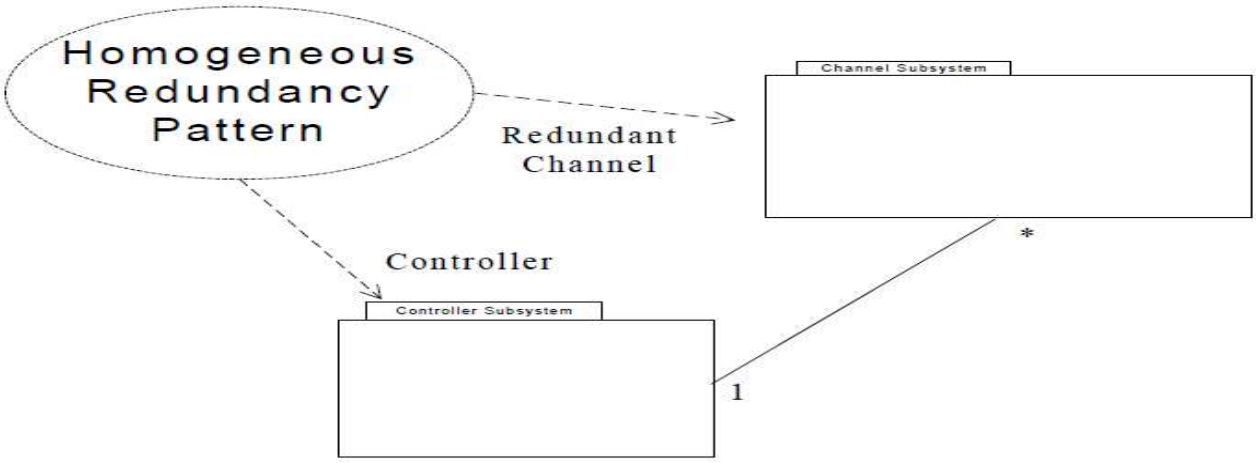



### 6.5 Πρότυπα ασφάλειας και αξιοπιστίας(safety and reliability patterns)

#### Πρότυπο Homogeneous Redundancy

Μια κοινή προσέγγιση για τη βελτίωση της αξιοπιστίας και της ασφάλειας είναι μέσω πολλαπλών πανομοιότυπων καναλιών. Στο πλαίσιο αυτό, ένα κανάλι είναι μια συνεργασία των αντικειμένων που εκτελεί μια σειρά που αφορούν την ασφάλεια υπολογισμών ή ενεργοποιήσεων. Η χρησιμοποίηση των ίδιων διαύλων επιτρέπει την προστασία από τυχαία λάθη, όπως αποτυχίες συστατικών, αλλά όχι κατά των συστηματικών σφαλμάτων, όπως σφάλματα λογισμικού. Το πρότυπο αυτό πορεί να χρησιμοποιηθεί παράλληλα με ένα σύστημα ψηφοφορίας. Μπορεί επίσης να χρησιμοποιηθεί ως μια μετάβαση αντιγράφων ασφαλείας σε περίπτωση αποτυχίας στο τρέχων κανάλι.

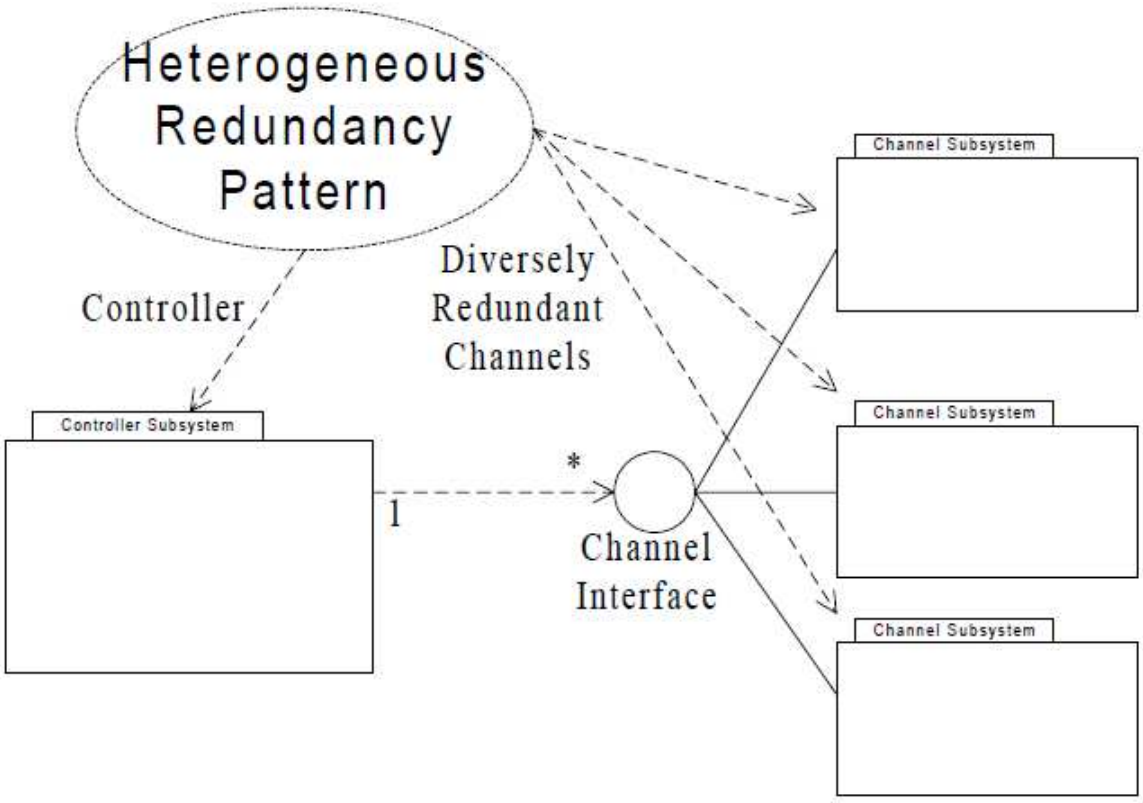
Γενική δομή Homogeneous Redundancy:




**Πρότυπο Heterogeneous Redundancy**

Ένα άλλο κοινό πρότυπο που είναι παρόμοιο με το προηγούμενο με την εξαίρεση ότι τα κανάλια είναι υλοποιημένα με διαφορετικό τρόπο: το υλικό είναι διαφορετικό και το λογισμικό είναι συνήθως κατασκευασμένο με διαφορετικό σχεδιασμό και εκτελέσιμο κώδικα. Το πρότυπο αυτό καλύπτει συστηματικά καθώς και τυχαία σφάλματα.

Γενική δομή Heterogeneous Redundancy:

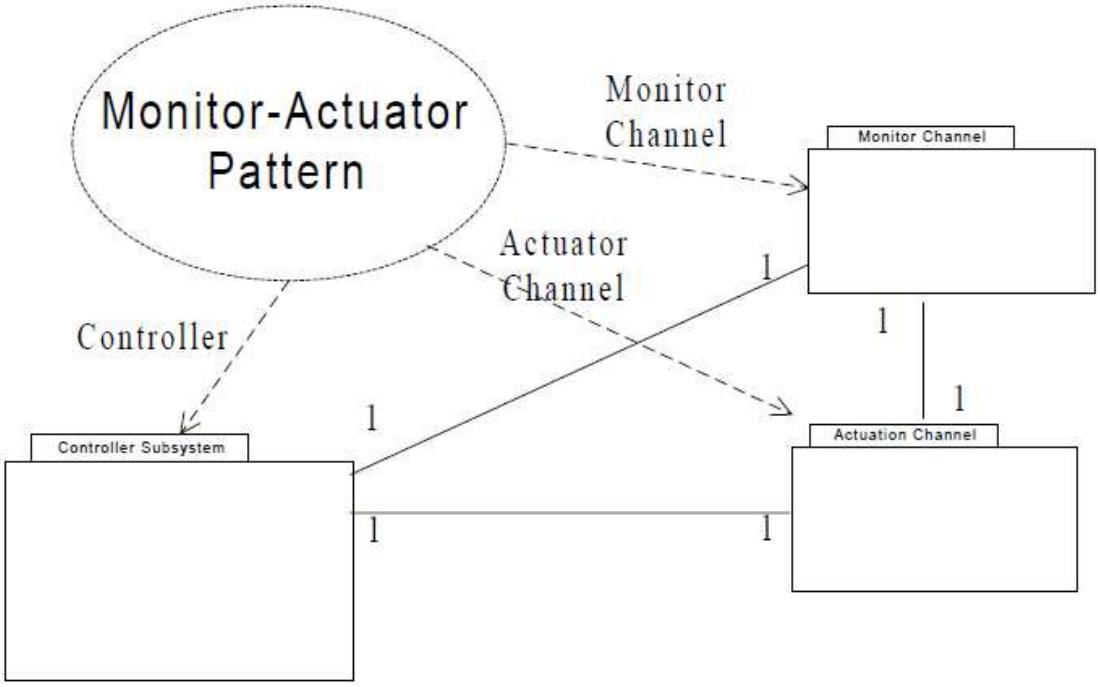


**Πρότυπο Monitor-Actuator**

Σε αυτό το πρότυπο το κανάλι του παρακολουθητή δεν ελέγχει τους υπολογισμούς του πρωτεύοντος καναλιού άμεσα, αλλά οι έλεγχοι είναι σχετικοί με τα αποτελέσματα της ενεργοποίησης του πρωτεύοντος καναλιού. Είναι σημαντικό ότι αυτή η παρακολούθηση παρέχεται από ένα ξεχωριστό σύνολο των αισθητήρων από αυτό που χρησιμοποιείται από το κανάλι ενεργοποίησης για την παροχή προστασίας από κοινά λειτουργικά λάθη (λάθη που συμβαίνουν σε περισσότερα από ένα κανάλια).

Το πρότυπο αυτό βασίζεται σε εξωτερικούς αισθητήρες για να επιτύχει την αναγνώριση σφαλμάτων. Έτσι, ο προσδιορισμός των σφαλμάτων όπως σε κάποια άλλα πρότυπα, δεν θα είναι τόσο εύκολος αλλά θα βασίζεται σε δεδομένα του πραγματικού κόσμου.

Γενική δομή Monitor-Actuator:




**6.6 Απλά πρότυπα(Simple Patterns)**

**Πρότυπο Transcation**

Τα συστήματα πραγματικού χρόνου χρησιμοποιούν πρωτόκολλα επικοινωνίας για αποστολή και λήψη πληροφοριών ζωτικής σημασίας, τόσο εσωτερικά μεταξύ των επεξεργαστών,

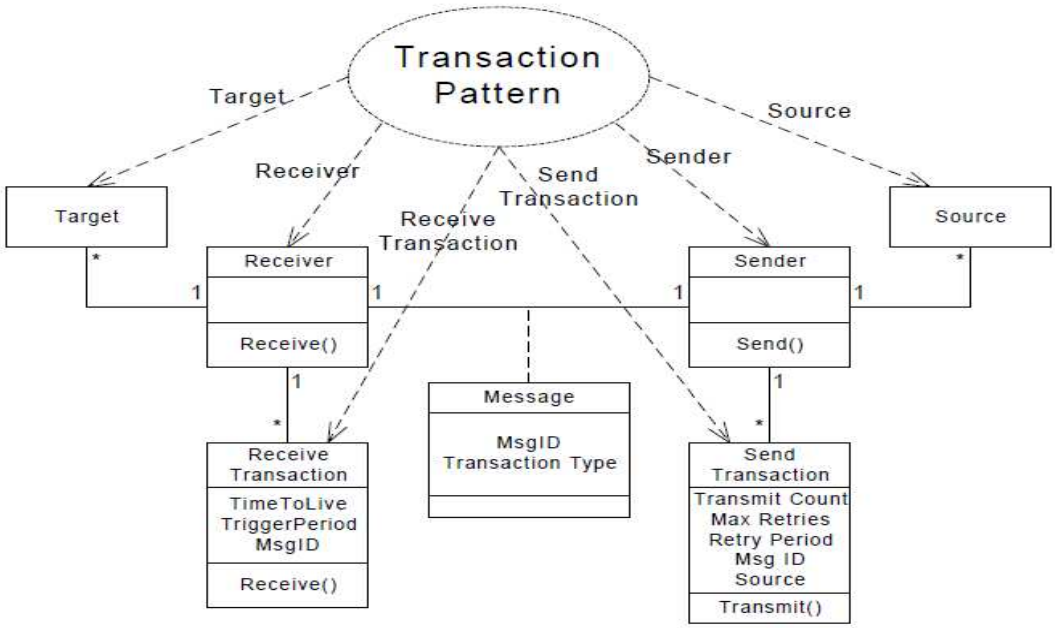
καθώς και με εξωτερικούς παράγοντες στο περιβάλλον. Στο ίδιο σύστημα, διαφορετικά μηνύματα μπορεί να έχουν διαφορετικά επίπεδα κρισιμότητας, και έτσι μπορεί να έχουν διαφορετικές απαιτήσεις για την αξιοπιστία της μεταφοράς του μηνύματος.

Το πρότυπο αυτό χρησιμοποιείται όταν αξιόπιστες επικοινωνίες απαιτούνται πάνω από αναξιόπιστα μέσα ή όταν απαιτείται επιπλέον αξιοπιστία. Για παράδειγμα, ένα σύστημα μπορεί να χρειαστεί τρία διαφορετικά επίπεδα αξιόπιστης επικοινωνίας:

1. At Most Once (AMO) –ένα μήνυμα που μεταδίδεται μόνο μία φορά. Αν το μήνυμα χάνεται ή είναι κατεστραμμένο, χάνεται. Αυτό χρησιμοποιείται όταν η αξιοπιστία της μεταφοράς του μηνύματος είναι ψηλή σε σχέση με την πιθανότητα απώλειας του μηνύματος.
2. At Least Once (ALO) –ένα μήνυμα που εκπέμπεται κατ 'επανάληψη μέχρι τη ρητή βεβαίωση παραλαβής από τον αποστολέα ή όταν έχει υπερβεί τη μέγιστη μέτρηση επανάληψης. Αυτό χρησιμοποιείται όταν η αξιοπιστία της μεταφοράς μηνύματος είναι σχετικά χαμηλή σε σχέση με την πιθανότητα απώλειας μηνύματος, αλλά η παραλαβή του ίδιου μηνύματος πολλές φορές είναι εντάξει.
3. Exactly Once (EO) – ένα μήνυμα που αντιμετωπίζεται ως ALO, εκτός αν ένα μήνυμα πρέπει να ληφθεί περισσότερες από μία φορές λόγω επαναλήψεων, μόνο το πρώτο μήνυμα θα πρέπει να ενεργοποιηθεί. Αυτό χρησιμοποιείται όταν η αξιοπιστία της μεταφοράς του μηνύματος είναι σχετικά χαμηλή, αλλά είναι σημαντικό ότι ένα μήνυμα ενήργησε μόνο μία φορά.

Το πρότυπο αυτό είναι ιδιαίτερα κατάλληλο για συστήματα πραγματικού χρόνου που χρησιμοποιούν ένα γενικό πρωτόκολλο επικοινωνίας με πλούσια γραμματική. Επιτρέπει στους σχεδιαστές της εφαρμογής ευελιξία στην επιλογή της μεθόδου επικοινωνίας, έτσι ώστε να μπορούν να βελτιστοποιήσουν την ταχύτητα της αξιοπιστίας.

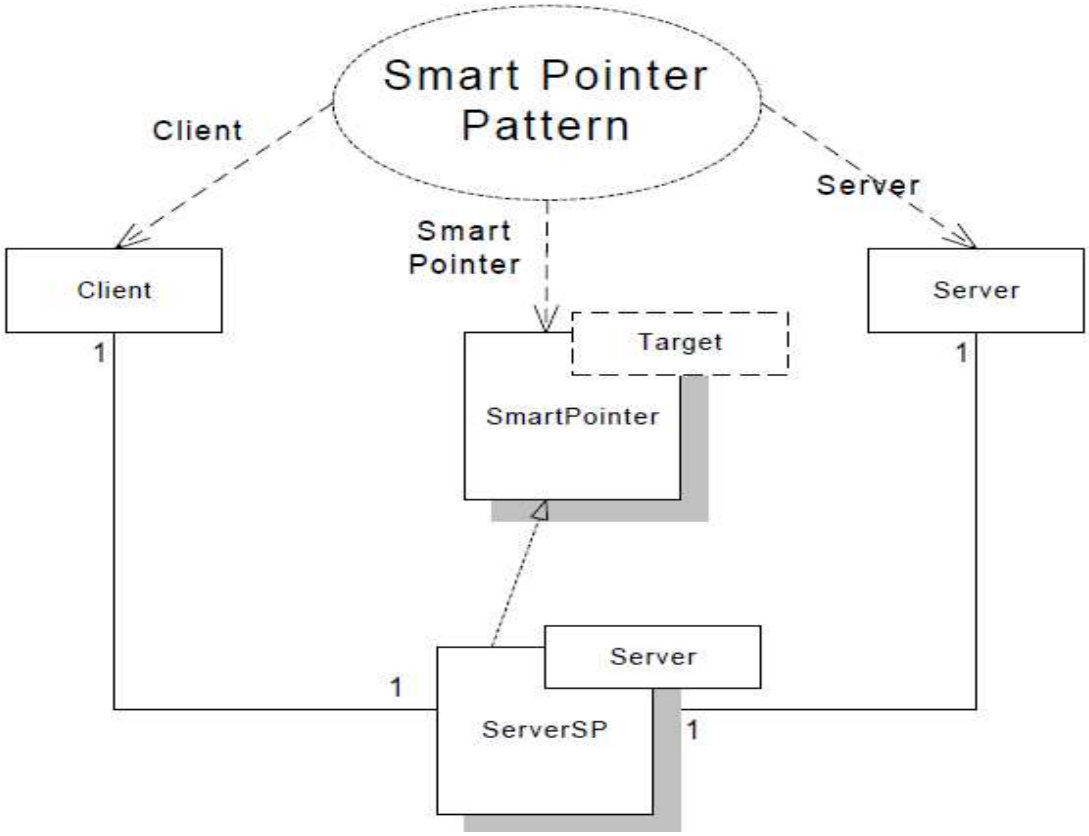
Γενική δομή Transaction:



**Πρότυπο Smart Pointer**

Το πρότυπο αυτό χρησιμοποιείται για την εξάλειψη ή τουλάχιστον τον περιορισμό ων προβλημάτων που απορρέουν από το εγχειρίδιο χρήσης των πρώτων δεικτών(raw pointers):
1. Όταν οι raw pointers δεν έχουν κατασκευαστή(constructor) να τους αφήσει σε μια έγκυρη αρχική κατάσταση, οι smart pointers μπορούν να χρησιμοποιήσουν τους κατασκευαστές τους για να αρχικοποιήσουν την τιμή τους σε NULL ή να δείχνουν σε ένα έγκυρο αντικείμενο.
2. Όταν οι raw pointers δεν έχουν destructor και έτσι δεν μπορούν να αποδεσμεύσουν μνήμη αν ξαφνικά βγουν εκτός εμβέλειας(scope), οι smart pointers μπορούν να καθορίσουν κατά πόσον είναι ή όχι σκόπιμο να αποδεσμεύσουν μνήμη όταν βγαίνουν εκτός εμβέλειας και καλούν τον τελεστή delete.
3. Όταν ο raw pointer σε ένα αντικείμενο που έχει διαγραφεί εξακολουθεί να κατέχει τη διεύθυνση της μνήμης του αντικειμένου που χρησιμοποιείται, οι smart pointers μπορεί να ανιχνεύσουν αυτόματα την προϋπόθεση αυτή και να αρνηθούν την πρόσβαση.
Εσωτερικά, ο smart pointer είναι (στατικός) αριθμός αναφοράς, ο οποίος καταγράφει τον αριθμό των τρέχουσων πελατών(clients) στο διακομιστή(server). Αυτή η μέτρηση αυξάνεται όταν οι smart pointers για το ίδιο αντικείμενο αυξάνονται και μειώνονται όταν οι smart pointers διαγράφονται. Όταν ο τελευταίος smart pointer διαγραφεί η μνήμη του αντικειμένου ελευθερώνεται πριν την διαγραφή του smart pointer.

Γενική δομή Smart Pointer:



# Κεφάλαιο 7

## Σχεδιαστικά πρότυπα για σχεδιασμό ιστοσελίδων


7.1 Εισαγωγή 62
7.2 Η ανάγκη για κοινή γλώσσα στο σχεδιασμό ιστοσελίδων 63
7.3 Πρότυπα Web Site Level 64
7.4 Πρότυπα Web Page Level 68
7.5 Πρότυπα Ornamentation Level 73

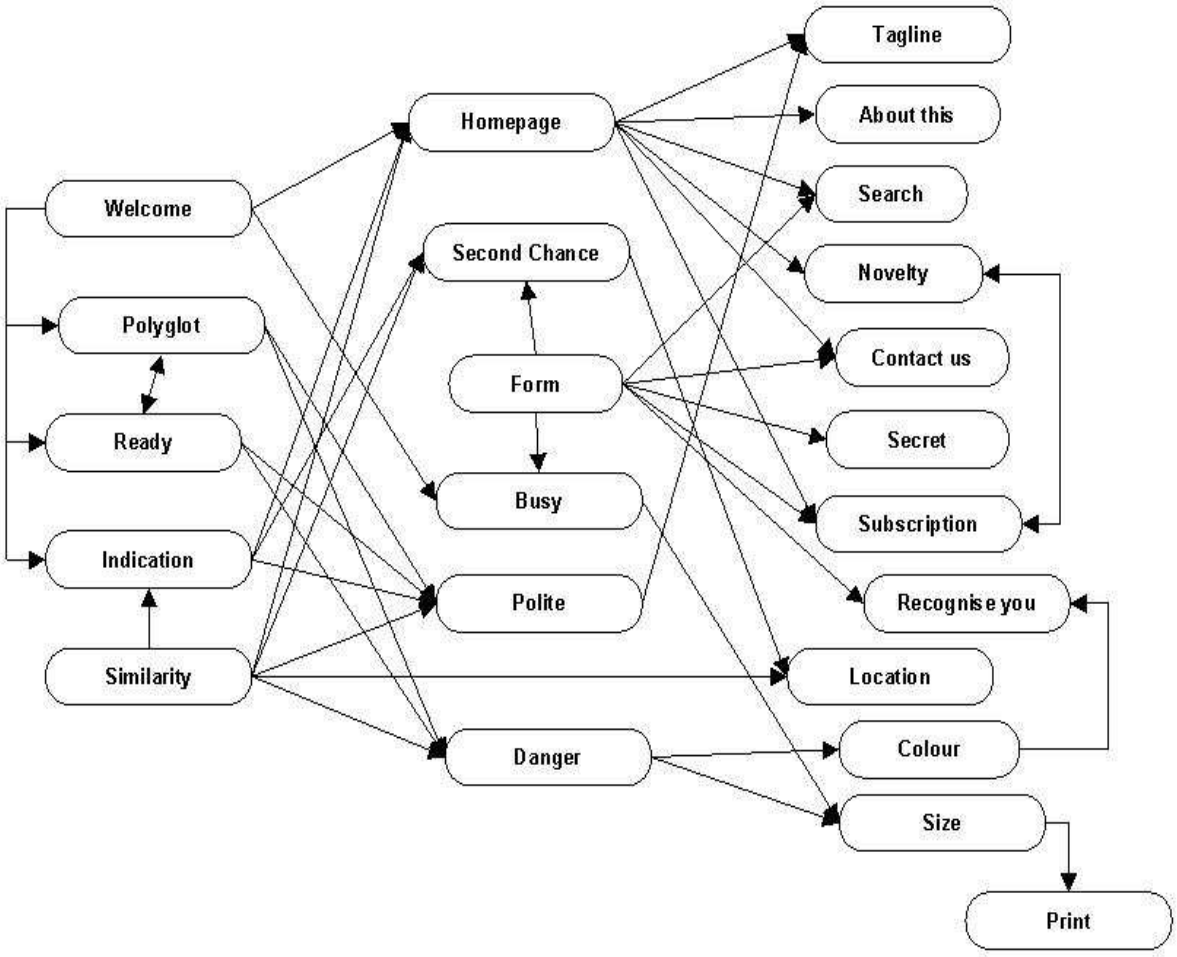

### 7.1 Εισαγωγή

Ο Παγκόσμιος Ιστός(World Wide Web) έχει γίνει το κυρίαρχο εργαλείο στο Internet, συνδυάζοντας το υπερκείμενο και τα πολυμέσα για να παρέχει ένα δίκτυο διεπιστημονικών πόρων. Είναι σημαντικό να βεβαιωθούμε ότι κάθε τμήμα της ιστοσελίδας είναι χρήσιμο. Ένας χρήστης που θα επισκεφτεί μια ιστοσελίδα αναμένεται να είναι σε θέση να επιτελέσει ένα συγκεκριμένο έργο, ή να διαβάσει ένα συγκεκριμένο κομμάτι πληροφοριών. Όταν σχεδιάζουμε μια ιστοσελίδα θέλουμε να βεβαιωθούμε ότι ο χρήστης μπορεί να βρει την πληροφορία γρήγορα και εύκολα. Εάν δεν μπορεί να βρει τις πληροφορίες γρήγορα, τότε μπορεί να φύγει από τον ιστότοπό μας, και να προχωρήσει σε άλλη τοποθεσία όπου μπορεί να βρει αυτό που χρειάζεται πιο εύκολα.

Πολλά σύνολα κατευθυντήριων γραμμών για σχεδιασμό ιστοσελίδων, έχουν δημοσιευθεί. Υπάρχουν πολλά που μπορούν να χρησιμοποιηθούν για τη βελτίωση του σχεδιασμού των ιστοσελίδων μας. Ωστόσο, οι περισσότερες από αυτές τις συστάσεις(πρότυπα) για τους σχεδιαστές ιστοσελίδων, δεν βασίζονται στην έρευνα, αλλά στη διαίσθηση. Τα πρότυπα αυτά βασίζονται στην εμπειρία του σχεδιαστή. Ο στόχος μιας γλώσσας προτύπων είναι να συλλάβει πρότυπα σε περιβάλλοντα τους, και να παράσχει ένα μηχανισμό για την κατανόηση των μη τοπικών επιπτώσεων των αποφάσεων του σχεδιασμού.

### 7.2 Η ανάγκη για κοινή γλώσσα στο σχεδιασμό ιστοσελίδων

Μια γλώσσα προτύπων στο σχεδιασμό ιστοσελίδων πρέπει να έχει τη δομή της ιεραρχίας του δικτύου. Έτσι, ένα απαραίτητο συστατικό για τον ορισμό ενός προτύπου είναι οι σχέσεις του με τα άλλα πρότυπα. Στο διάγραμμα της προτεινόμενης γλώσσας προτύπων(βλέπε σχήμα 1), τα βέλη μεταξύ των προτύπων δείχνουν αυτές τις σχέσεις. Η γλώσσα αυτών των προτύπων διαχωρίζεται σε 3 επίπεδα, τα οποία είναι εμπνευσμένα από τον αρχιτέκτονα των σχεδιαστικών προτύπων Κρίστοφερ Αλεξάντερ. Η δική του γλώσσα προτύπων επεκτείνεται στο σχεδιασμό ιστοσελίδων με την εισαγωγή προτύπων που σχετίζονται με **Web sites**, **Web pages** και **ornamentation details(**λεπτομέρειες διακόσμησης)..




**Σχήμα 1,Προτεινόμενη γλώσσα σχεδιαστικών προτύπων για σχεδιασμό ιστοσελίδων.**

Όλα τα πιο πάνω πρότυπα παρουσιάζονται με την πιο κάτω μορφή:

**Motivation:**[Κίνητρο για τη δημιουργία του προτύπου]
**Problem:**[Ορισμός του προβλήματος]
**Forces:**[Δεδομένα του προβλήματος]
**Solution:**[Λύση που προτείνει το πρότυπο]
**Consequences:**[Συνέπειες από τη χρήση αυτού του προτύπου]
**Examples & implementation details:** [Παραδείγματα και λεπτομέρειες υλοποίησης]

Πιο κάτω θα μελετηθούν αναλυτικά τα πρότυπα που σχετίζονται με **Web sites**, **Web pages** και **ornamentation details**.

### 7.3 Πρότυπα Web Site Level

Τα σχεδιαστικά πρότυπα που σχετίζονται με το σχεδιασμό ιστοσελίδας(Web Site), συνδέονται με κοινά χαρακτηριστικά που μπορούν να βρεθούν σε πολλές ιστοσελίδες και επεκτείνονται από άλλο διαφορετικό πλαίσιο. Ο χρήστης απαιτεί να ξέρει που βρίσκεται, και που μπορεί να πάει. Ο χρήστης επιθυμεί να επισκεφθεί την ιστοσελίδα με ένα κατάλληλο τρόπο.

#### Welcome

**Motivation:** Όταν ένας χρήστης φτάσει σε μια ιστοσελίδα, όπως όταν φτάνει σε μια πόλη, κωμόπολη ή οποιοδήποτε σημαντικό κτίριο, πρέπει να ξέρει που είναι, τι μπορεί να κάνει εκεί, και τι χρειάζεται για να επισκεφτεί αυτό το μέρος.
**Problem:** Πώς ο χρήστης ξέρει πού είναι;
**Forces:**
-Οι χρήστες πρέπει να γνωρίζουν πού βρίσκονται.
-Οι χρήστες θέλουν να ξέρουν πού μπορούν να πάνε μετά.
-Μια σύνθετη ιστοσελίδα μπορεί να αποπροσανατολίσει πολύ τους χρήστες.
-Οι χρήστες που είναι εξοικειωμένοι με τη δομή και το περιεχόμενο μιας ιστοσελίδας θα πρέπει να μεταβούν κατευθείαν στο χώρο όπου θέλουν να πάνε.
**Solution:** Δώστε ένα χώρο υποδοχής, όπου οι συνθήκες πρόσβασης των χρηστών μπορούν να αξιολογηθούν. Από το σημείο αυτό του καλωσορίσματος, ο χρήστης θα είναι σε θέση να εισέλθει στην αρχική σελίδα και σε άλλες ενδείξεις. Οι πληροφορίες των χρηστών, όπως η γλώσσα ή το μέγεθος της οθόνης θα πρέπει να συγκεντρωθούν για την παροχή των υπηρεσιών της ιστοσελίδας στο χρήστη. Ο χρήστης πρέπει να ενημερώνεται σχετικά με τις καλύτερες προϋποθέσεις για την επίσκεψη στο Web site ή οι όροι αυτοί πρέπει να

προσφέρονται άμεσα. Ο χρήστης μπορεί να βρει πληροφορίες σχετικά με το περιεχόμενο και τον ιδιοκτήτη του δικτυακού τόπου σε αυτή τη σελίδα. **Welcome** και **Homepage** είναι το ίδιο πρότυπο σε πολλές περιπτώσεις.

**Consequences:** Παροχή βελτιώσεων για την πλοήγηση, τη λειτουργικότητα και την ανατροφοδότηση.

**Examples & implementation details:** Αυτές οι ιστοσελίδες, και πολλές άλλες στο διαδίκτυο, έχουν μια αρχική σελίδα και ως κύρια χαρακτηριστικά: το χαμηλό χρόνο φορτώματος τους, προσφέρουν τη δυνατότητα να προσαρμόσουμε τις ιδιότητες ή τη γλώσσα του προγράμματος περιήγησης, και παρέχουν πληροφορίες σχετικά με το ποιος, τι, πότε και που ο χρήστης μπορεί να βρει στην ιστοσελίδα.

**Παραδείγματα**: http://www.crescere.co.uk/, http://www.graffiti.ie/

**Indication**

**Motivation:** Μία ιστοσελίδα, είναι ένας χώρος πλοήγησης όπου οι χρήστες θέλουν να επιτύχουν τους στόχους, όπως η αναζήτηση πληροφοριών ή την αγορά ενός προϊόντος. Με παρόμοιο τρόπο, όπως συμβαίνει σε σούπερ μάρκετ, στα μουσεία ή σημαντικά κτίρια, οι χρήστες χρειάζεται να γνωρίζουν πού μπορούν να πάνε και τι μπορούν να κάνουν τη στιγμή που θα φτάσουν εκεί.

**Problem:** Πώς ο χρήστης ξέρει πού μπορεί να πάει και τι θα βρει εκεί;

**Forces:**

-Ο χρήστης πρέπει να γνωρίζει ποιες θέσεις είναι διαθέσιμες.

-Ο χρήστης κάνοντας λάθος ενέργειες μπορεί να αποπροσανατολιστεί.

-Βάζοντας μαζί μια συλλογή από αντικείμενα μπορεί να πάρει χρόνο και διανοητική προσπάθεια.

-Οι σύνδεσμοι στην ιστοσελίδα θα πρέπει να οργανωθούν αρκετά καλά έτσι ώστε ο χρήστης να μπορεί να βρει αυτό που χρειάζεται.

**Solution:** Οι ιστοσελίδες πρέπει να παρέχουν τους απαραίτητους μηχανισμούς που επιτρέπουν σε κάθε χρήστη να μετακινείται από το ένα μέρος στο άλλο. Ο χρήστης μπορεί να αποπροσανατολίζεται και πρέπει να λαμβάνει πληροφορίες σχετικά με την τοποθεσία του και να έχει τη δυνατότητα να επιστρέφει σε ένα ασφαλές μέρος. Σημαντικοί σύνδεσμοι θα πρέπει να τοποθετηθούν ψηλά στην σελίδα και περιγραφικές ετικέτες συνδέσμου θα πρέπει να χρησιμοποιηθούν. Αν χρησιμοποιηθούν τα πλαίσια θα πρέπει να μπει τίτλος σε κάθε πλαίσιο για να διευκολύνει την αναγνώριση των πλαισίων και την πλοήγηση.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα και την πλοήγηση.

**Examples & implementation details:** Αυτές οι σελίδες, και άλλες σαν και αυτές παρέχουν πληροφορίες πλοήγησης με μενού, κουμπιά ή απλές συνδέσεις**.** Η κατάσταση ή η εμφάνιση των στοιχείων αυτών των πληροφοριών πλοήγησης μπορεί να ποικίλει (αριστερά, επάνω, δεξιά, κλπ.).

**Παραδείγματα**: http://www.fourthcube.com, http://www.amazon.com

**Polyglot**

**Motivation:** Η δύναμη του διαδικτύου είναι στην καθολικότητά του. Η πρόσβαση από όλους, ανεξαρτήτως αναπηρίας είναι ένα σημαντικό στοιχείο. Πολλοί παράγοντες πρέπει να εξετάζονται: υλικό, λογισμικό, αισθητική, κ.λπ.

**Problem:** Πώς μπορεί ο χρήστης να κάνει μια χρήσιμη χρήση της ιστοσελίδας και των πληροφοριών πρόσβασης στο δικό του ρυθμό;

**Forces:**

-Ο χρήστης θέλει εύκολη πρόσβαση σε πληροφορίες.

-Ο χρήστης κάνει κυρίως κάτι άλλο, και αυτό δεν πρέπει να συγκρούεται με αυτό.

-Ο χρήστης έχει μικρό ή κανένα κίνητρο για να ξοδεύει το χρόνο του μαθαίνοντας τεχνικές λεπτομέρειες.

- Ο χρήστης θέλει πλήρη πρόσβαση σε όλα με τη μία, ακόμη και αν η ιστοσελίδα είναι περίπλοκη.

**Solution:** Η γλώσσα του χρήστη είναι σχεδιασμός για όλους. Τα παιδιά, οι ηλικιωμένοι ή άτομα με ειδικές ανάγκες μπορούν να επισκεφθούν μια ιστοσελίδα και οι καθολικές τεχνικές σχεδίασης μπορούν να εφαρμοστούν στο σχεδιασμό της ιστοσελίδας και των υπηρεσιών της. Το μέγεθος της οθόνης, η ανάλυση της οθόνης του χρήστη, το μέγεθος της γραμματοσειράς, η ταχύτητα σύνδεσης και ο χρόνος λήψης πρέπει να λαμβάνονται υπόψη όταν σχεδιάζεται μια ιστοσελίδα. Οι πληροφορίες πρέπει να παρέχονται με κάποιο κατάλληλο τρόπο, εξετάζοντας διάφορα είδη των ανθρώπων και των τεχνικών χαρακτηριστικών και με τη χρήση ευγενικής γλώσσας.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα, τη γλώσσα και τη συνοχή.

**Examples & implementation details:** Αυτές οι σελίδες έχουν πολλές συνδέσεις ή κουμπιά που σχετίζονται με διαφορετικούς τρόπους απεικόνισης. Βασικά ο χρήστης μπορεί να επιλέξει μεταξύ πολλών γλωσσών ή επιπέδων. Δεδομένου ότι πολλοί άνθρωποι προτιμούν να διαβάζουν τυπωμένο κείμενο πολλές ιστοσελίδες παρέχουν έντυπη έκδοση των αντικειμένων ή εγγράφων. **Παραδείγματα**: http://europa.eu.int, http://wap.uclm.es

**Similarity**

**Motivation:** Όταν ένας χρήστης πλοήγηση σε όλη την ιστοσελίδα, πρέπει να γνωρίζει αν είναι ακόμα εκεί ή όχι. Η ιστοσελίδα μπορεί να είναι πολύ πολύπλοκη και πολλοί σύνδεσμοι μπορεί να είναι εξωτερικοί στην αρχική.

**Problem:** Πώς ο χρήστης γνωρίζει ότι επισκέπτεται την ίδια ιστοσελίδα.

**Forces:**

-Οι χρήστες πρέπει να γνωρίζουν πού βρίσκονται.

-Μια πολύπλοκη ιστοσελίδα μπορεί να αποπροσανατολίσει πολύ τους χρήστες.

-Οι χρήστες κάνοντας λάθος ενέργειες μπορεί να αποπροσανατολιστούν.

**Solution:** Η ιστοσελίδα θα πρέπει να σχεδιαστεί με βάση τα ίδια κριτήρια: τα χρώματα, τις γραμματοσειρές, πλοήγηση, τοποθεσία και διάταξη. Χρησιμοποιήστε ένα μόνο φύλλο για όλες τις σελίδες στον ιστότοπό σας. Ένα από τα κύρια πλεονεκτήματα των φύλλων είναι να εξασφαλίσει οπτική συνέχεια καθώς ο χρήστης περιηγείται στην ιστοσελίδα σας, αλλά τα έγγραφα θα πρέπει να οργανωθεί έτσι ώστε να μπορούν να διαβαστούν χωρίς φύλλα Ο χρήστης πάντα θα πρέπει να ενημερώνεται με τη χρήση κατάλληλου τρόπου, που είναι και που μπορεί να πάει. Μέσω των μηχανισμών undo/redo αποφεύγεται ο αποπροσανατολισμός.

**Consequences:** Παροχή βελτιώσεων για την πλοήγηση, τη συνοχή και την ανατροφοδότηση.

**Examples & implementation details:** Η εφαρμογή αυτού του προτύπου μπορεί να γίνει με τη χρήση φύλλων**.** Τα φύλλα είναι ένας τρόπος να διαχωριστεί το style από το ύφος του περιεχομένου σε ιστοσελίδες. Η γλώσσα HTML αναμιγνύει το style με το περιεχόμενο. Τα φύλλα επιτρέπουν να τροποποιήσετε τις προεπιλεγμένες ιδιότητες πολλών HTML ετικετών.

**Παραδείγματα**: http://www.ibm.com, http://www.plop.dk/VikingPLoP

**Ready**

**Motivation:** Μπορείτε να χρησιμοποιήσετε τα πάντα για να σχεδιάσετε την ιστοσελίδα σας, αλλά ο χρήστης που θέλει να επισκεφθεί την ιστοσελίδα σας θα πρέπει να εγκαταστήσει το απαραίτητο plug-ins, και θα πρέπει να θυμάστε ότι πρέπει να μιλούν τη γλώσσα σας.

**Problem:** Πώς ο χρήστης γνωρίζει ότι μπορεί να επισκεφτεί την ιστοσελίδα χωρίς προβλήματα;

**Forces:**

-Ο χρήστης θέλει εύκολη πρόσβαση στις πληροφορίες.

- Μια πολύπλοκη ιστοσελίδα μπορεί να αποπροσανατολίσει πολύ τους χρήστες.

-Ο χρήστης θέλει να έχει τον έλεγχο των ενεργειών.

-Ο χρήστης δεν θέλει να διακόπτεται από τις πτυχές ασφάλειας που σχετίζονται με το σχεδιασμό.

**Solution:** Παρέχετε τα εργαλεία ή τις απαραίτητες πληροφορίες για να επισκεφθούν την ιστοσελίδα σας με κατάλληλο τρόπο**.** Η ιστοσελίδα πρέπει να ανιχνεύει αν ο χρήστης έχει όλα τα αναγκαία και παρέχει συνδέσμους για να κατεβάσει μέρη όπου θα πάρει απαιτείται plug-ins. Ο χρήστης δεν χρειάζεται να ξέρετε τις τεχνικές πτυχές. Βεβαιωθείτε ότι οι σελίδες μπορούν να χρησιμοποιηθούν όταν τα αρχεία εντολών, εφαρμογές ή άλλα προγραμματιστικά αντικείμενα απενεργοποιούνται ή δεν υποστηρίζονται. Εάν αυτό δεν είναι δυνατόν, να παρέχετε πληροφορίες σχετικά με μια εναλλακτική, προσβάσιμη σελίδα.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα, τον έλεγχο και την πλοήγηση.

**Examples & implementation details:** Υπάρχουν πολλές ιστοσελίδες όπου απαιτείται plug-in, ή ελάχιστη ανάλυση οθόνης. Τα πλαίσια δεν υποστηρίζονται σε άλλες καταστάσεις. Ο χρήστης πρέπει να γνωρίζει ότι χρειάζεται.

**Παραδείγματα**: http://www.mercksharpdohme.com/, http://www.hp.com

### 7.4 Πρότυπα Web Page Level

Τα σχεδιαστικά πρότυπα που σχετίζονται με web design είναι συνήθη στοιχεία και θεωρούνται χαρακτηριστικά όταν σχεδιάζουμε ιστοσελίδες. Σε αυτή την ιεραρχική δομή μια οικοσελίδα(Homepage) είναι απαραίτητη. Σε ορισμένες περιπτώσεις, που ο χρήστης χρειάζεται την παροχή πληροφοριών τότε πρέπει να συμπληρώσει μία φόρμα και πάντα ο χρήστης θέλει να έχει τον έλεγχο και να επισκέπτονται ιστοσελίδες με το δικό του ρυθμό.

#### Homepage

**Motivation:** Μία ιστοσελίδα μπορεί να επιτευχθεί με τυχαίο τρόπο, αλλά πάντα πρέπει να έχουν ένα σημείο αναφοράς. Όταν ένας χρήστης σε μια ιστοσελίδα, όπως όταν φτάνει σε μια πόλη, κωμόπολη ή οποιοδήποτε σημαντικό κτίριο, πρέπει να ξέρει που είναι, τι μπορεί να κάνει εκεί, και τι χρειάζεται για να επισκεφτεί αυτήν την ιστοσελίδα. Η αρχική σελίδα αποτελεί βασικό συστατικό μιας ιστοσελίδας. Ερωτήματα όπως ποιος, τι, πότε, και πού θα πρέπει να έχουν απάντηση.

**Problem:** Πώς ο χρήστης γνωρίζει πού βρίσκεται**;**

**Forces:**

-Οι χρήστες πρέπει να γνωρίζουν πού βρίσκονται

-Οι χρήστες θέλουν να ξέρουν  πού μπορούν να πάνε μετά.

-Μια σύνθετη ιστοσελίδα μπορεί να αποπροσανατολίσει πολύ τους χρήστες.

-Οι χρήστες που είναι εξοικειωμένοι με τη δομή και το περιεχόμενο μιας ιστοσελίδας θα πρέπει να μεταβούν κατευθείαν στο χώρο όπου θέλουν να πάνε.

**Solution:** Δώστε μια αρχική σελίδα όπου ο χρήστης αισθάνεται σαν στο σπίτι του**.** Η κεντρική σελίδα(homepage) είναι ένα μέρος όπου ο χρήστης μπορεί να πάει πίσω αν είναι αποπροσανατολισμένος**.** Η διάταξή της βάζει σημαντικές πληροφορίες στο πάνω μέρος, περιλαμβάνει τα λογότυπα, προσεγγίσεις αναζήτησης και τα στοιχεία επικοινωνίας.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα, τον έλεγχο και την πλοήγηση.

**Examples & implementation details:** Κάθε ιστοχώρος έχει μια ιστοσελίδα. Πρόκειται για μια ειδική σελίδα που παρουσιάζει ιδιαίτερα χαρακτηριστικά. Ο πιο κρίσιμος ρόλος της αρχικής σελίδας είναι να επικοινωνήσει αυτό που η εταιρεία είναι, η αξία που προσφέρει το site πέρα από τον ανταγωνισμό και το φυσικό κόσμο, και τα προϊόντα ή οι υπηρεσίες που προσφέρονται. Η πρόκληση είναι να σχεδιάσουμε μια ιστοσελίδα που επιτρέπει την πρόσβαση σε όλα τα σημαντικά χαρακτηριστικά. Οι αναφορές στην ιστοσελίδα πρέπει να περιλαμβάνονται σε κάθε σελίδα του δικτυακού τόπου.

**Παραδείγματα**: http://www.apple.com, http://www.ieee.org

**Polite**

**Motivation:** Για πολλά χρόνια, τεχνικές επικοινωνίας τόνισαν την ανάγκη να χρησιμοποιούν τη γλώσσα που έχει νόημα για τους αναγνώστες**.** Γι αυτό θα ήταν χρήσιμο για τους ανθρώπους να φαίνεται διαισθητικά προφανές**.** Ωστόσο, η δυσκολία στο να το πετύχετε αυτό μπορεί να είναι λιγότερο προφανής: οι άνθρωποι διαφέρουν σημαντικά στον τρόπο που επιλέγουν να περιγράψουν ιδιαίτερες έννοιες.

**Problem:** Πώς ο χρήστης μπορεί να έχει πρόσβαση στο περιεχόμενο ενός web page με απλό και σωστό τρόπο.

**Forces:**

-Οι χρήστες μπορεί να επιβραδύνει το ρυθμό του όταν πρέπει να αναλογιστούμε τη διαφορά μεταξύ συνδέσμων παρόμοιων ετικετών.

- Υπάρχουν φορές που οι όροι δεν έχουν νόημα για όλους τους χρήστες της ιστοσελίδας.

**Solution:** Χρησιμοποιήστε τη σαφέστερη και απλούστερη γλώσσα που είναι κατάλληλη για το περιεχόμενο της ιστοσελίδας. Δημιουργήστε έγγραφα, σύμφωνα με δημοσιευμένες, επίσημες γραμματικές. Συσχετίστε ρητά τις ετικέτες με τους ελέγχους. Εκφράστε μόνο μια ιδέα, σε κάθε πρόταση. Μεγάλες, περίπλοκες προτάσεις, συχνά σημαίνουν ότι δεν είναι

σαφείς για το τι θέλετε να πείτε. Ρωτώντας τους χρήστες, είναι ένας ιδιαίτερα αποτελεσματικός τρόπος για να επιλέξετε τα ονόματα των επιλογών, χρησιμοποιήστε κάρτες ταξινόμησης, τη συμμετοχική σχεδίαση, ή άλλες μεθόδους που αναμειγνύουν τους χρήστες όποτε είναι δυνατόν.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα, την ανατροφοδότηση, τη γλώσσα και τη συνοχή.

**Examples & implementation details:** Οι γλώσσες συχνά έχουν εναλλακτικές εκφράσεις για το ίδιο πράγμα, και μια συγκεκριμένη λέξη μπορεί να φέρει διαφορετική σημασία ή να λειτουργεί ως διαφορετικό μέρος του λόγου. Επειδή οι γλώσσες είναι φυσικά προσαρμοσμένες στις καταστάσεις της χρήσης τους και επίσης αντανακλούν τις κοινωνικές ταυτότητες των ομιλητών τους, η γλωσσική ποικιλία είναι φυσικό και αναπόφευκτο φαινόμενο. Μερικές λέξεις μπορούν να δημιουργήσουν προβλήματα για σας, ειδικά, όταν τις χρησιμοποιείτε χωρίς να σκέφτεστε το πραγματικό νόημα τους. Ο τρόπος που σκεφτόμαστε και οι λέξεις που χρησιμοποιούμε καθορίζουν τις αντιδράσεις μας στη ζωή. Οι υπερσυνδέσμοι(hyperlinks) θα πρέπει να αναφέρουν με σαφήνεια πού οδηγούν όταν πατηθούν. Αποφύγετε τα λογοπαίγνια και βεβαιωθείτε ότι ο σύνδεσμος έχει επαρκές μήκος για τον αναγνώστη να κάνει κλικ. Στόχος είναι να βοηθήσετε τους αναγνώστες σας.

**Παραδείγματα**: http://www.rae.es, http://ox.ac.uk

**Busy**

**Motivation:** Οι ιστοσελίδες είναι μέρη όπου οι χρήστες μπορούν να κατεβάσουν πληροφορίες, εικόνες, αρχεία ή εφαρμογές, αλλά αυτή η λήψη μπορεί να πάρει πολύ χρόνο, δημιουργεί σημαντικές καθυστερήσεις ή μπορεί να επιτευχθεί με διάφορους τρόπους.

**Problem:** Πώς ο χρήστης ξέρει πότε οι λειτουργίες του έχουν ολοκληρωθεί ή η τελική μορφή τους;

**Forces:**

-Ο χρήστης θέλει να ξέρει πόσο καιρό θα πρέπει να περιμένει για τον τερματισμό της διαδικασίας.

-Ο χρήστης θέλει να ξέρει πόσο γρήγορα γίνεται η διαδικασία, ειδικά αν η ταχύτητα ποικίλλει.

-Μερικές φορές αδύνατο να μάθει σε πόσο χρόνο η διαδικασία πρόκειται να λάβει τέλος.

σημείο αυτό αρχίζει η διατύπωση της πρώτης παραγράφου του δεύτερου υποκεφαλαίου. Από το σημείο αυτό αρχίζει η διατύπωση της πρώτης παραγράφου του δεύτερου υποκεφαλαίου.

**Solution:** Ζητάτε πληροφορίες για τις ενέργειες του χρήστη. Εικόνες, αρχεία και κάθε

στοιχείο που ο χρήστης μπορεί να κατεβάσει πρέπει να έχει πληροφορίες για το μέγεθος, έτσι ώστε οι χρήστες να γνωρίζουν πόσο καιρό πρέπει να περιμένουν για τη διαδικασία λήψης. Εικόνες και κείμενο θα πρέπει να κατεβαίνουν on-demand.

**Consequences:** Παροχή βελτιώσεων σχετικά με τη λειτουργικότητα, την ανατροφοδότηση και την πρόληψη σφαλμάτων.

**Examples & implementation details:** Πολλές ιστοσελίδες χρειάζονται ένα plug-ing ια να πάρετε μια σωστή απεικόνιση, μια γραμμή προόδου(progress bar) χρησιμοποιείται για την παροχή των εν λόγω πληροφοριών. Μερικές φορές μία εργασία που εκτελείται στο πλαίσιο ενός προγράμματος μπορεί να πάρει λίγο χρόνο για να ολοκληρωθεί. Ένα φιλικό προς το χρήστη πρόγραμμα παρέχει κάποια ένδειξη για το χρήστη για το πόσο καιρό η εργασία θα μπορούσε να διαρκέσει και πόση δουλειά έχει ήδη γίνει. Εάν δεν γνωρίζετε ή δεν θέλετε να δείχνετε πόσο πλήρης είναι η εργασία, μπορείτε να χρησιμοποιήσετε ένα δρομέα ή μια κινούμενη εικόνα για να δείξετε ότι κάποια εργασία εκτελείται.

**Second Chance**

**Motivation:** Όταν ο χρήστης πλοηγείται σε μια ιστοσελίδα, θέλει να αισθανθεί τον έλεγχο των δραστηριοτήτων του. Χρειάζεται να γνωρίζει ότι κάθε ενέργεια μπορεί να ακυρωθεί και ότι μπορεί να επιστρέψει στην προηγούμενη κατάσταση.

**Problem:** Πώς μπορεί ο χρήστης να είναι σίγουρος για τις ενέργειες του;

**Forces:**

-Οι χρήστες κάνοντας λάθος ενέργειες μπορεί να αποπροσανατολιστούν.

-Ο χρήστης χρειάζεται ασφάλεια και πρόληψη σφαλμάτων.

-Ο χρήστης θέλει να εξερευνήσει και όχι να μάθει.

-Ο χρήστης είναι βιαστικός.

**Solution:** Παροχή μηχανισμών για undo / redo, backing και clearing. Οι μηχανισμοί αυτοί σε ένα περιβάλλον διαδικτύου αποτελούνται από συνδέσμους στην προηγούμενη σελίδα, την προηγούμενη θέση, ή την οικοσελίδα. Σε μία φόρμα είναι απαραίτητο να παρέχονται δύο κουμπιά: submit και reset.

**Consequences:** Παροχή βελτιώσεων σχετικά με τη λειτουργικότητα, τον έλεγχο και την πρόληψη σφαλμάτων.

**Examples & implementation details:** Οι σελίδες που εφαρμόζουν αυτό το πρότυπο έχουν συνδέσμους ή κουμπιά που παρέχουν αναίρεση(undo). Έτσι συνδέσμοι με προηγούμενες ενότητες ή ιστοσελίδα είναι συνηθισμένα σε κάθε σελίδα του δικτυακού τόπου. Το κουμπί επαναφοράς(reset) είναι ένα άλλο παράδειγμα χρησιμοποίησης αυτού του προτύπου. Οι

browsers παρέχουν αυτή τη δυνατότητα με τη χρήση του πίσω(back) κουμπιού. **Παραδείγματα**: http://www.iomega.com, http://www.acrobat.com

**Form**

**Motivation:** Ο χρήστης πρέπει να παρέχει πληροφορίες, συνήθως σύντομες απαντήσεις σε ερωτήσεις.

**Problem:** Πώς μπορεί ο χρήστης να παρέχει πληροφορίες στον ιδιοκτήτη της ιστοσελίδας;

**Forces:**

-Ο χρήστης πρέπει να γνωρίζει τι είδους πληροφορίες θα παρέχει.

-Οι χρήστες γενικά δεν απολαμβάνουν την παροχή πληροφοριών με αυτόν τον τρόπο.

-Θα πρέπει να είναι σαφές τι απαιτείται, και τι είναι προαιρετικό.

-Ο χρήστης είναι βιαστικός.

**Solution:** Παροχή κατάλληλων κενών που πρέπει να συμπληρωθούν, τα οποίο σαφέστατα και σωστά υποδεικνύουν ποιες πληροφορίες θα πρέπει να παρέχονται. Search, Contact Us και Subscription είναι παραδείγματα των φορμών. Συνήθως, μία φόρμα γεμίζει μια πλήρη σελίδα. Ο χρήστης θέλει να ξέρει αν η φόρμα του έχει καταχωρηθεί σωστά.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα.

**Examples & implementation details:** Μία HTML φόρμα είναι ένα τμήμα ενός εγγράφου που περιέχει κανονικό περιεχόμενο, σήμανση, ειδικά στοιχεία που ονομάζονται στοιχεία ελέγχου(checkboxes, radio buttons, menus) και ετικέτες των ελέγχων αυτών. Οι χρήστες συνήθως συμπληρώνουν μία φόρμα με την τροποποίηση αυτών των ελέγχων (entering text, selecting menu items) πριν από την υποβολή της αίτησης σε ένα μέσο για επεξεργασία(web server, mail server).

**Παραδείγματα**: http://www.iomega.com, http://www.iberia.es

**Danger**

**Motivation:** Υπάρχει μια πληθώρα plug-ins για ήχο,animation και όλα τα είδη των πραγμάτων. Αλλά δεν μπορούμε να υποθέσουμε ότι κάποιος πρόκειται να τους έχει, ή μπορεί να τα χρησιμοποιήσει με τον συγκεκριμένο υπολογιστή.

**Problem:** Πώς μπορεί ο χρήστης να επισκεφθείτε μια ιστοσελίδα, χωρίς να συγχυστεί, να ενοχληθεί ή να αποπροσανατολιστεί;

**Forces:**

-Οι χρήστες γενικά δεν έχουν το plug-in που χρειάζονται.

-Θα πρέπει να είναι σαφές τι απαιτείται για να επισκεφθεί την ιστοσελίδα.

-Οι χρήστες που μπορεί να επισκεφτούν την ιστοσελίδα είναι άγνωστοι.

-Όλοι είναι ανίκανοι με τον ένα ή τον άλλο τρόπο.

**Solution:**Να είστε προσεκτικοί με τη χρήση συστατικών που μπορεί να σας αποπροσανατολίσουν. Για παράδειγμα, μπορείτε να χρησιμοποιήσετε το ευανάγνωστο μέγεθος της γραμματοσειράς, καλά σχεδιασμένα headings, περιορισμένο αριθμό πλαισίων, animated gifs, flash, applets, music, rollovers, αλλά μην υπερβείτε το μέγιστο μέγεθος σελίδας. Χρησιμοποιήστε φύλλα για να ελέγξετε τη διάταξη και την παρουσίαση.

**Consequences:** Παροχή βελτιώσεων για την οπτική ευκρίνεια, τον έλεγχο, τη λειτουργικότητα και την πλοήγηση.

**Examples & implementation details:** Η βασική σχεδίαση του διαδικτύου στηρίζεται στην κατοχή της σελίδας ως ατομική μονάδα των πληροφοριών, καθώς και η έννοια της σελίδας διαπερνά όλες τις πτυχές του διαδικτύου. Τα πλαίσια μπορεί να δημιουργήσουν πολλά προβλήματα: για παράδειγμα, η πλοήγηση δεν λειτουργεί με πλαίσια δεδομένου ότι η μονάδα της πλοήγησης είναι διαφορετική από τη μονάδα της όψης. Τα πλαίσια ιστοσελίδων είναι είτε δύσκολα ή αδύνατα για τις μηχανές αναζήτησης στο ευρετήριο. Εάν χρησιμοποιείτε τα πλαίσια, οι άνθρωποι θα έχουν μία δύσκολη περιήγηση στην ιστοσελίδα σας. Τα πλαίσια προκαλούν προβλήματα στην εκτύπωση σε παλαιότερα προγράμματα περιήγησης, τα οποία τείνουν να εκτυπώσουν το πλαίσιο στο τέλος, που δεν είναι αυτό που θέλετε να εκτυπώσετε.

**Παραδείγματα**: http://www.garbage.com, http://www.biblioteca.uclm.es

### 7.5 Πρότυπα Ornamentation Level

Τα πρότυπα αυτά σχετίζονται με τη διακόσμηση μιας ιστοσελίδας. Τα χαρακτηριστικά αυτά παρέχουν βελτιώσεις για τη γενική ευχρηστία κάθε δικτυακού τόπου. Σχετίζονται με τη χρήση του χρώματος, του μεγέθους, της ασφάλειας και την παροχή αναφορών στις τοποθεσίες και πληροφοριών.

### Tag Line

**Motivation:** Όταν σχεδιάζετε μια ιστοσελίδα θα πρέπει να παρέχετε πληροφορίες σχετικά με το σκοπό της**.**

**Problem:** Πώς μπορεί ο χρήστης να γνωρίζει το σκοπό της ιστοσελίδας;

**Forces:**

-Οι χρήστες είναι βιαστικοί.

-Οι χρήστες δεν διαβάζουν ιστοσελίδες, ρίχνουν μια ματιά στις σελίδες.

**Solution:** Συμπεριλάβετε ένα μήνυμα(tag line) που συνοψίζει ρητά ό,τι η ιστοσελίδα κάνει. Το μήνυμα θα πρέπει να είναι σύντομο, απλό και στο σημείο. Συμπεριλάβετε μια σύντομη περιγραφή της ιστοσελίδας στο παράθυρο.

**Consequences:** Παροχή βελτιώσεων για την οπτική ευκρίνεια, τη λειτουργικότητα και την ανατροφοδότηση.

**Examples & implementation details:** Αυτές οι σελίδες έχει εικόνες ή taglines που εφαρμόζουν αυτό το πρότυπο. Ένα tagline είναι μια σύντομη φράση που επικοινωνεί το «ποιος» και «γιατί» της ιστοσελίδας σας.

**Παραδείγματα**: http://www.coolhomepages.com, http://www.bbva.es

**Print**

**Motivation:** Οι περισσότεροι άνθρωποι διαβάζουν το κείμενο online με διαφορετικό τρόπο από τον τρόπο που διαβάζουν τυπωμένα κείμενα, και όχι με την ανάγνωση λέξη προς λέξη. Κατά την ανάγνωση κειμένου σε σελίδες μέσα σε μια ιστοσελίδα, οι περισσότεροι άνθρωποι κινούνται γρήγορα μεταξύ των σελίδων.

**Problem:** Πώς μπορεί ο χρήστης να πάρει μια κατάλληλη εκτύπωση των πληροφοριών;

**Forces:**

-Οι αναγνώστες εστιάζονται σε μικρά κομμάτια που μπορούν να τα βρουν γρήγορα.

-Οι περισσότεροι χρήστες αποθηκεύουν τα έγγραφα στο δίσκο ή για να τα τυπώσουν.

**Solution:** Δώστε μια έκδοση του κειμένου των ιστοσελίδων απευθείας εκτυπώσιμη, ή προσφέρετε μορφοποιημένη έκδοση του εγγράφου που θα εκτυπωθεί σε download μορφή.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα και τον έλεγχο.

**Examples & implementation details:** Αυτό το πρότυπο χρησιμοποιείται όταν το περιεχόμενο μεγάλων εγγράφων έχει σπάσει σε μικρότερα κομμάτια. Στη συνέχεια, παρέχοντας μια μεγάλη σελίδα για την εκτύπωση, ένα αρχείο για να κατεβάσετε (. Pdf,. Ps,. Doc) ή την ικανότητα για την εκτύπωση όλων των εγγράφων σε ένα βήμα που είναι χρήσιμο για τον χρήστη. Από το σημείο αυτό αρχίζει η διατύπωση της πρώτης παραγράφου του

**Παραδείγματα**: http://www.borland.es, http://www.sport.es.

**Subscription**

**Motivation:** Οι χρήστες θέλουν να μην επισκέπτονται μια ιστοσελίδα καθημερινά, προτιμούν να ενημερώνονται όταν φτάσουν τα νέα προϊόντα ή ειδήσεις.

**Problem:** Πώς μπορεί ο χρήστης να ενημερώνεται με σημαντικές πληροφορίες για αυτόν; Πώς μπορεί ο χρήστης να έχει πρόσβαση σε περιοδική πληροφόρηση;

**Forces:**

-Οι χρήστες είναι βιαστικοί.

-Οι χρήστες θέλουν να ενημερώνονται.

**Solution:** Δώστε τη δυνατότητα στο χρήστη να κάνετε κράτηση on-line, παρέχοντας ένα email. Έτσι, ο ιδιοκτήτης της ιστοσελίδας μπορεί να στείλει πληροφορίες σε εγγεγραμμένους χρήστες για καινοτομίες. Ο χρήστης θα πρέπει να είναι σίγουρος ότι η διεύθυνση e-mail σας είναι μυστική.

**Consequences:** Παροχή βελτιώσεων για ανατροφοδότηση**.**

**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με τη χρήση ενός απλού εντύπου όπου ο χρήστης συνήθως πρέπει να παρέχει μόνο ένα email. Σε άλλες περιπτώσεις είναι απαραίτητο να παρέχετε περισσότερες πληροφορίες σχετικά με τις προτιμήσεις του χρήστη και το προφίλ σας, έτσι ώστε να είναι δυνατόν να παρέχετε εξατομικευμένες πληροφορίες. Η επιλογή της διαγραφής(unsubscribe) πρέπει να παρέχεται επίσης.

**Παραδείγματα**: http://www.prenhall.com, http://www.sun.es

**Contact Us**

**Motivation:** Όλες οι επιχειρηματικές ιστοσελίδες πρέπει να παρέχουν σαφή τρόπο για να επικοινωνήσετε με τον ιδιοκτήτη του δικτυακού τόπου.

**Problem:** Πώς μπορεί ο χρήστης να πάρει περισσότερες πληροφορίες σχετικά με τα προϊόντα ή τα έγγραφα**;**

**Forces:**

-Οι άνθρωποι θέλουν να ξέρουν με ποιον έχουν επιχειρηματική δραστηριότητα.

-Να πάρουν πληροφορίες της εταιρείας θα μπορούσε να είναι ο μοναδικός λόγος για τον οποίο οι χρήστες έρχονται στην ιστοσελίδα.

-Πολλοί χρήστες θέλουν να ξέρουν πώς είναι πίσω από την υπηρεσία.

**Solution:** Δώστε μια φόρμα, μια τοποθεσία ή ένα σύνδεσμο στην ιστοσελίδα, όπου ο χρήστης μπορεί να πάρει πρόσθετες πληροφορίες σχετικά με τον ιδιοκτήτη της ιστοσελίδας και τα προϊόντα του.

**Consequences:** Παροχή βελτιώσεων για ανατροφοδότηση**.**

**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με τη συμπερίληψη μια σελίδας ή ενός τμήματος όπου ο χρήστης μπορεί να βρει τα στοιχεία επικοινωνίας. Σε πολλές περιπτώσεις η πληροφορία αυτή περιλαμβάνεται στο κάτω μέρος όλων των σελίδων του δικτυακού τόπου. Σε άλλες περιπτώσεις, ένα έντυπο παρέχεται στο χρήστη. Το έντυπο αυτό περιέχει χαρακτηριστικά, όπως ένα textarea ή textfields, ώστε να μπορεί ο χρήστης να παρέχει το e-mail και τις ερωτήσεις του. Τα δεδομένα που παρέχονται από το χρήστη είναι μυστικά.

**Παραδείγματα**: http://www.intel.com, http://www.lucent.com

**Novelty**

**Motivation:** Οι χρήστες θέλουν να γνωρίζουν εάν υπάρχουν νέες δυνατότητες στην ιστοσελίδα. Οι χρήστες δέχονται εισηγήσεις και θέλουν να μάθουν τις προσφορές και τις διαφημήσεις.

**Problem:** Πώς μπορεί ο χρήστης να γνωρίζει τις καινοτομίες και τις τελευταίες ειδήσεις ή τις προτάσεις ενός δικτυακού τόπου**;**

**Forces:**

-Οι χρήστες δεν διαβάζουν ιστοσελίδες, ρίχνουν μια ματιά στις σελίδες.

-Οι χρήστες είναι βιαστικοί.

**Solution:** Παροχή καινοτομιών της ιστοσελίδας με τρόπο σαφή και διαισθητικό, όπου οι χρήστες θα έχουν γρήγορη πρόσβαση σε νέες υπηρεσίες που προσφέρονται από την ιστοσελίδα.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα και την πλοήγηση.

**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με την τοποθέτηση τελευταίων ειδήσεων ή προτάσεων σε μια εξαιρετική θέση στην Ιστοσελίδα.

**Παραδείγματα**: http://www.microsoft.com, http://www.terra.es

**Search**

**Motivation:** Η αναζήτηση είναι ένα από τα πιο σημαντικά στοιχεία της ιστοσελίδας, και είναι σημαντικό ότι οι χρήστες είναι σε θέση να βρουν κάτι εύκολα και να το χρησιμοποιήσουν αβίαστα.

**Problem:** Πώς μπορεί ο χρήστης να γνωρίζει αν μια ιστοσελίδα μπορεί να του παρέχει συγκεκριμένες πληροφορίες που θέλει;

**Forces:**

-Οι χρήστες θέλουν να ξέρουν αν η πληροφορία που αναζητούν είναι στην ιστοσελίδα.

-Οι χρήστες δεν διαβάζουν ιστοσελίδες, ρίχνουν μια ματιά στις σελίδες.

**Solution:** Δώστε μια μηχανή αναζήτησης ή σελίδα επισκόπησης. Δώστε στους χρήστες ένα πλαίσιο εισαγωγής στην αρχική σελίδα για να εισάγουν τα ερωτήματα αναζήτησης, όχι κατ'ανάγκην παρέχοντάς τους ένα σύνδεσμο σε μια σελίδα αναζήτησης.

**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα και τον έλεγχο.

**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με την παροχή μιας φόρμας αναζήτησης. Οι φόρμες αναζήτησης είναι η διεπιφάνεια χρήσης της μηχανής αναζήτησης. Μπορεί να αποτελείται από ένα το πολύ απλό έντυπο με μόλις ένα πεδίο κειμένου και ένα κουμπί. Προηγμένες δυνατότητες αναζήτησης μπορεί να αξίζει να προστεθούν. Μια σελίδα σύνθετης αναζήτησης με επιλογές για φράσεις, πολλαπλά πεδία, ειδικές συλλογές ή ζώνες, επιτρέπει την εκτέλεση πιο ακριβών αναζητήσεων.

**Παραδείγματα**: http://www.paginasamarillas.es, http://www.microsoft.com.

**Recognize**

**Motivation:** Όταν ένας χρήστης επιστρέφει σε μια ιστοσελίδα χρειάζεται να ξέρει τι μέρη έχει επισκεφτεί, τι έγγραφα έχει κατεβάσει και αν υπάρχουν τροποποιήσεις από την τελευταία επίσκεψη.

**Problem:** Πώς γνωρίζει ο χρήστης που βρίσκεται;

**Forces:**

-Ο χρήστης δεν θέλει να χάνει το χρόνο του.

-Ο χρήστης θέλει να παίρνει προσωπικές πληροφορίες.

**Solution:** Διατηρήστε τις πληροφορίες σχετικά με τις ενέργειες των χρηστών, ιστοχώρους που επισκέφτηκε, συνδέσεις, κλπ, για παράδειγμα με τη χρήση cookies. Καθώς το HTTP είναι non-persistent πρωτόκολλο, είναι αδύνατο να γίνει διάκριση μεταξύ των επισκέψεων σε μια ιστοσελίδα, εκτός αν ο διακομιστής μπορεί να σηματοδοτήσει κάπως ένα επισκέπτη.

**Consequences:** Παροχή βελτιώσεων για ανατροφοδότηση και έλεγχο σφαλμάτων.

**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με τη χρήση cookies. Τα web cookies είναι απλά κομμάτια του λογισμικού που τοποθετούνται στον υπολογιστή σας όταν κάνετε περιήγηση σε ιστοσελίδες, έτσι ώστε η ιστοσελίδα θα αναγνωρίσει τον υπολογιστή του χρήστη, όταν αυτός έρχεται πίσω για να την επισκεφτεί ξανά. Τα cookies έχουν κάποιες ευεργετικές συνέπειες. Για παράδειγμα, όταν συνδέεστε ή κάνετε αγορές σε ορισμένες ιστοσελίδες, μπορείτε να παρατηρήσετε ότι όταν επιστρέψετε πάλι, δεν χρειάζεται να κάνετε εγγραφή(sign up); Αυτό συμβαίνει γιατί αποθηκεύεται ο κωδικός πρόσβασης σας και ταυτοποιείται στον υπολογιστή σας σε ένα cookie. Έτσι, ο φόρτος εργασίας του χρήστη μειώνεται.

**Παραδείγματα**: http://www.kinkos.com, http://www.americanairlines.com

**Colour**

**Motivation:** Πολλοί σχεδιαστές ιστοσελίδων παραβλέπουν τη σημασία των χρωμάτων κατά το σχεδιασμό μιας ιστοσελίδας**.** Το χρώμα πρέπει να είναι μία από τις πρώτες ανησυχίες σας όταν έρχεται η ώρα να ξεκινήσετε το σχεδιασμό της ιστοσελίδας σας**.**

**Problem:** Πώς ο χρήστης μπορεί να έχει πρόσβαση στο περιεχόμενο ενός web page με τον κατάλληλο τρόπο;

**Forces:**

-Οι web browsers μπορούν να αναγνωρίσουν μόνο 256 χρώματα.

-Οι άνθρωποι διαβάζουν φωτεινούς χαρακτήρες σε σκούρο φόντο για μεγάλα χρονικά διαστήματα κάτι που προκαλεί λιγότερο οπτική κόπωση.

**Solution:** Παροχή πληροφοριών με τη χρήση κατάλληλων χρωμάτων στις γραμματοσειρές, υπόβαθρα και εικόνες. Για παράδειγμα, τα προεπιλεγμένα χρώματα για τους συνδέσμους της ιστοσελίδας είναι μπλε για μη-επισκέψιμους συνδέσμους και μωβ για συνδέσμους που έχετε επισκεφθεί. Βεβαιωθείτε ότι οι χρωματικοί συνδυασμοί προσκηνίου και παρασκηνίου παρέχουν αρκετή αντίθεση όταν προβάλλονται σε κάποιον με αδυναμίες διάκρισης των χρωμάτων ή όταν προβάλλονται σε λευκή οθόνη. Θα πρέπει να χρησιμοποιείτε κίτρινο και κόκκινο χρώμα με φειδώ στην ιστοσελίδα σας, σε χώρους όπου θέλετε ο επισκέπτης να επικεντρωθεί. Μην κάνετε μεγάλα τμήματα της ιστοσελίδας σας με φωτεινά χρώματα.

**Consequences:** Παροχή βελτιώσεων στην ανατροφοδότηση, τη λειτουργικότητα, τη συνέπεια και την οπτική ευκρίνεια.

**Examples & implementation details:** Μια αρχική ιδέα για το σχεδιασμό μιας ιστοσελίδας είναι η παλέτα των 216 χρωμάτων. Ευτυχώς, οι περισσότεροι HTML editors, έχουν αυτή την παλέτα. Αυτό το σχέδιο υλοποιείται με την επιλογή χρωματικών συνδυασμών, όπου η αναγνωσιμότητα και η οπτική ευκρίνεια βελτιώνεται και υπάρχει η διαβεβαίωση ότι όλες οι

πληροφορίες που μεταφέρονται με χρώματα είναι διαθέσιμες και χωρίς χρώμα, για παράδειγμα από το περιεχόμενο ή τη σήμανση. Τα φόντα δεν θα πρέπει να επηρεάσουν την αναγνωσιμότητα. Η πρώτη αναφορά είναι ένα παράδειγμα της κακής χρήσης του χρώματος.
**Παραδείγματα**: http://www.biblioteca.uclm.es, http://www.trashclub.com.

**Location**

**Motivation:** Όταν ένας χρήστης φτάσει σε μια ιστοσελίδα, όπως όταν φτάνει σε μια πόλη, κωμόπολη ή οποιοδήποτε σημαντικό κτίριο, πρέπει να ξέρει που είναι.
**Problem:** Πώς ο χρήστης γνωρίζει πού βρίσκεται**;**
**Forces:**
-Οι χρήστες πρέπει να γνωρίζουν πού βρίσκονται.
-Μια σύνθετη ιστοσελίδα μπορεί να αποπροσανατολίσει πολύ τους χρήστες.
**Solution:** Παροχή πληροφοριών ανατροφοδότησης σχετικά με τη θέση του χρήστη στην ιστοσελίδα.
**Consequences:** Παροχή βελτιώσεων στην πλοήγηση, τη συνέπεια και την ανατροφοδότηση.
**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με την τοποθέτηση αναφορών σε όλες τις ιστοσελίδες του δικτυακού τόπου: τη χρήση τίτλων και breadcrumbs. Συνήθως μια πολύπλοκη ιστοσελίδα περιλαμβάνει ένα sitemap.
**Παραδείγματα**: http://java.sun.com, http://www.acrobat.com

**Size**

**Motivation:** Ο σχεδιασμός του World Wide Web είναι μια πράξη εξισορρόπησης μεταξύ των γραφικών "wow" και του πραγματικού χρόνου "now". Όσο πιο πολλά γραφικά έχει μια ιστοσελίδα τόσο περισσότερο χρόνο θα πάρει για να κατεβεί.
**Problem:** Πώς ο χρήστης μπορεί να έχει πρόσβαση στο περιεχόμενο ενός web page με τον κατάλληλο τρόπο;
**Forces:**
-Με 28.8K σύνδεση, ο υπολογιστής σας μπορεί να λάβει, κατά μέσο όρο, 2K ανά δευτερόλεπτο. Κανείς δεν θέλει να περιμένει 30 δευτερόλεπτα μόνο και μόνο για να δει το λογότυπο του site σας.
-Η ανάπτυξη σταθερού μεγέθους ιστοσελίδες είναι μια διαβλητή πρακτική.

**Solution:** Παροχή πληροφοριών με τη χρήση κατάλληλων μεγεθών σε εικόνες, γραμματοσειρές, και σελίδες. Animations, εικόνες, αρχεία μεγάλου μήκους θα πρέπει να παρέχονται, εάν ο χρήστης θέλει πραγματικά αυτό (on-demand). Μήκος σελίδας, ανάγκες σελιδοποίησης και μέγεθος γραμματοσειράς αποτελούν σημαντικές πτυχές.
**Consequences:** Παροχή βελτιώσεων για τον έλεγχο, τη συνέπεια και την οπτική ευκρίνεια.
**Examples & implementation details:** Αυτό το πρότυπο εφαρμόζεται όταν χρησιμοποιούνται μικρογραφίες**,** όταν το πλάτος και το ύψος χρησιμοποιούνται σαν χαρακτηριστικά του <IMG> tag, χρησιμοποιώντας βασικές γραμματοσειρές, ή όταν μεγάλος όγκος των πληροφοριών είναι μοιρασμένος σε κομμάτια και γραμμένος σε πολλές σελίδες. Η οργάνωση των εγγράφων, ώστε να μπορούν να διαβαστούν χωρίς φύλλα είναι χρήσιμη.
**Παραδείγματα**: http://www.number-10.gov.uk, http://www.tagheuer.com

**About this**

**Motivation:** Όλες οι επιχειρηματικές ιστοσελίδες πρέπει να παρέχουν σαφή τρόπο για την εύρεση πληροφοριών για την εταιρεία, ανεξάρτητα πόσο μεγάλη ή μικρή είναι η εταιρεία.
**Problem:** Πώς ο χρήστης γνωρίζει ποιος είναι ο σκοπός της ιστοσελίδας;
**Forces:**
-Οι άνθρωποι θέλουν να ξέρουν με ποιον έχουν επιχειρηματική δραστηριότητα.
-Να πάρουν πληροφορίες της εταιρείας θα μπορούσε να είναι ο μοναδικός λόγος για τον οποίο οι χρήστες έρχονται στην ιστοσελίδα.
-Πολλοί χρήστες θέλουν να γνωρίζουν ποιος είναι πίσω από την υπηρεσία.
**Solution:** Δώστε μια θέση ή ένα σύνδεσμο, όπου ο χρήστης μπορεί να πάρει πληροφορίες σχετικά με το περιεχόμενο του δικτυακού τόπου.
**Consequences:** Παροχή βελτιώσεων στη λειτουργικότητα και την ανατροφοδότηση.
**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με την προσθήκη ενός τμήματος όπου μπορείτε να βρείτε τις πληροφορίες σχετικά με τον ιδιοκτήτη της ιστοσελίδας. Κανονικά ένας σύνδεσμο για αυτό το τμήμα βρίσκεται στην οικοσελίδα(Homepage).
**Παραδείγματα**: http://www.sunspot.net, http://www.ireland.com

**Secret**

**Motivation:** Εάν ο χρήστης παρέχει προσωπικές πληροφορίες, θα πρέπει να έχει το δικαίωμα να αναμένει εχεμύθεια. Η ταχεία πρόοδος στην τεχνολογία της επικοινωνίας έχει τονίσει την ανάγκη για την ασφάλεια στο Διαδίκτυο.

**Problem:** Πώς μπορεί ο χρήστης να είναι σίγουρος ότι οι πληροφορίες που παρέχει ο ίδιος προστατεύονται;

**Forces:**

-Οι χρήστες θέλουν ασφάλεια.

-Οι χρήστες δεν χρειάζεται να γνωρίζουν τεχνικές πτυχές.

**Solution:** Για την παροχή ασφάλειας απαιτούνται μηχανισμοί (πρόσβαση και προστασία της ιδιωτικής ζωής) και να ενημερώσει το χρήστη για τις συνθήκες ασφαλείας και τους όρους χρήσης. Οι χρήστες θα πρέπει να εγγραφούν και έτσι η πρόσβαση σε ιδιωτικά τμήματα για την ιστοσελίδα επιτρέπεται, αλλά μόνο ένα όνομα χρήστη(username) και ένας κωδικός πρόσβασης(password) δεν είναι αρκετά μερικές φορές.

**Consequences:** Παροχή βελτιώσεων στην ανατροφοδότηση και τον έλεγχο.

**Examples & implementation details:** Αυτό το πρότυπο υλοποιείται με την μορφή σύνδεσης(login), όπου θα ζητήσει από το χρήστη για το όνομα χρήστη(username) και τον κωδικό πρόσβασής(password) του με τη χρήση php ή asp. Αλλά εάν η φόρμα σας βρίσκεται σε έναν ασφαλή διακομιστή, η πληροφορία μεταδίδεται σε μορφή απλού κειμένου, και η κρυπτογράφηση δεν θα συμβεί μέχρι το php script τρέξει. Συμπεριλάβετε συνδέσμους, όπου οι χρήστες μπορούν να διαβάσουν τους όρους χρήσης περί Προστασίας Προσωπικών Δεδομένων(Privacy Policy Statement ).

**Παραδείγματα**: http://www.bankofamerica.com, http://www.cdnow.com

# Κεφάλαιο 8

## Μοντελοποίηση περιγραφικών προτύπων με τη χρήση διαγραμμάτων UML


8.1 Εισαγωγή 82
8.2 Μοντελοποίηση προτύπων αλληλεπίδρασης ανθρώπου υπολογιστή 84
8.2.1 Μοντελοποίηση προτύπου Narrative 84
8.2.2 Μοντελοποίηση προτύπου Form 84
8.2.3 Μοντελοποίηση προτύπου Control Panel 85
8.2.4 Μοντελοποίηση προτύπου Composed Command 85
8.2.5 Μοντελοποίηση προτύπου Navigable Spaces 86
8.2.6 Μοντελοποίηση προτύπου Small Group of Related Things 87
8.2.7 Μοντελοποίηση προτύπου Hierarchical Set 87
8.2.8 Μοντελοποίηση προτύπου Optional Detail 88
8.2.9 Μοντελοποίηση προτύπου Interaction History 89
8.2.10 Μοντελοποίηση προτύπου Grid Layout 89
8.3 Μοντελοποίηση προτύπων σχεδιασμού ιστοσελίδων 90
8.3.1 Μοντελοποίηση προτύπου Welcome 90
8.3.2 Μοντελοποίηση προτύπου Indication 90
8.3.3 Μοντελοποίηση προτύπου Ready 91
8.3.4 Μοντελοποίηση προτύπου Busy 92
8.3.5 Μοντελοποίηση προτύπου Second Chance 92
8.3.6 Μοντελοποίηση προτύπου Print 93
8.3.7 Μοντελοποίηση προτύπου About This 94
8.3.8 Μοντελοποίηση προτύπου Secret 94
8.3.9 Μοντελοποίηση προτύπου Subscription 95
8.3.10 Μοντελοποίηση προτύπου Search 95


### 8.1 Εισαγωγή

Στο κεφάλαιο 5 μελετήσαμε τα σπουδαιότερα αλληλεπιδραστικά πρότυπα που χρησιμοποιούνται κατά την ανάπτυξη διαδραστικών συστημάτων. Ο σχεδιασμός αποτελεσματικών διαδραστικών συστημάτων αναγνωρίζεται ως ένα δύσκολο έργο. Όχι μόνο η αρχική σχεδίαση του συστήματος δεν μπορεί να θεωρηθεί ένα ασήμαντο πρόβλημα, αλλά θα πρέπει να τροποποιηθεί εκ νέου ανάλογα με το πρίσμα των μεταβαλλόμενων τεχνολογικών ή εμπορικών απαιτήσεων. Αυτό σημαίνει ότι ένα τέτοιο σύστημα είναι συχνά

component-based, δηλαδή αποτελείται από ένα σύνολο μικρότερων, απλούστερων μηχανισμών που επιλύουν ορισμένα ζητήματα αξιόπιστα και αποτελεσματικά. Κατά τη διαδικασία της οικοδόμησης ενός μεγάλου συστήματος, τα συστατικά συναρμολογούνται μαζί για να δημιουργήσουν ένα πιο σύνθετο σύστημα. Από τη σκοπιά του HCI, το να επιτευχθεί ο αρχικός σχεδιασμός και η σωστή αλληλεπίδραση είναι δύσκολο, μερικές φορές περισσότερο θέμα της βιοτεχνίας και επανάληψης από την καθορισμένη μεθοδολογία, και οι δυσκολίες αυτές επιδεινώνονται κατά τη διάρκεια των κύκλων της ταχείας παραγωγής λογισμικού. Ο χρήστης μπορεί να ξεχάσει την πρόοδο της τεχνολογίας καθώς πολύ συχνά νέα χαρακτηριστικά εμφανίζονται σε αρχικά καλά σχεδιασμένα συστήματα που είναι περιττά, ανεπιθύμητα, ή απλά απρόσιτα. Ακόμα και όταν είναι αρχικά καλά σχεδιασμένα, τα συστήματα μπορούν να εξελιχθούν μακριά από τις ανάγκες των χρηστών.

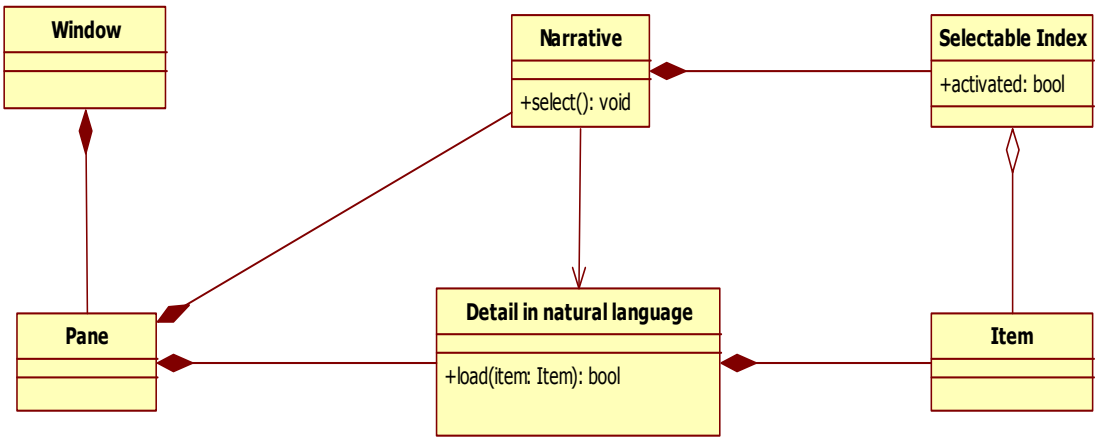
Ένα κεντρικό ερώτημα είναι αν κάτι που είναι αναγνωρίσιμα δύσκολο να ποσοτικοποιηθεί, όπως τα HCI πρότυπα, μπορούν να αναπαρασταθούν στη UML. Είναι διδακτικό να προβληματιστούμε σχετικά με τις επιτυχίες στη μοντελοποίηση της Ποιότητας Υπηρεσίας(QoS). Σχεδιάζουμε να αντιμετωπίσουμε τα HCI πρότυπα με τον ίδιο τρόπο όπως και οι ερευνητές έχουν ασχοληθεί με το σχεδιασμό και τις προδιαγραφές των μη-λειτουργικών πτυχών των συστημάτων, όπως η Ποιότητα Υπηρεσίας(QoS). Η QoS είναι, όπως μια καλή HCI ,μια αναδυόμενη ιδιοκτησία με βάση το πλήθος των αλληλεπιδράσεων μεταξύ των διαφόρων συνιστωσών, όχι μια ειδική ενότητα που μπορεί να προστεθεί σε μεταγενέστερο στάδιο.

Υπάρχουν μερικά ενδιαφέροντα ερωτήματα που πρέπει να απαντηθούν για το αν αυτός ή οποιοσδήποτε άλλος φορμαλισμός μπορεί πραγματικά να συλλάβει αποτελεσματικά όλα τα σχετικά τμήματα του προτύπου (π.χ. εμφάνιση και αισθητική). Ωστόσο, δεδομένου ότι έχουμε καταφέρει να συλλάβει την αισθητική και άλλων άυλων περιουσιακών στοιχείων σε πράγματα όπως η εσωτερική διακόσμηση καταλόγων, αυτό σημαίνει ότι μπορεί να είμαστε σε θέση να αναπαραστήσουμε πολλά πράγματα αποτελεσματικά.

Μια πρώτη προσέγγιση που έγινε, δείχνει ότι αν έχουμε δεδομένα που έχουν τη μορφή επιλογής περιεχομένου(selectable index), μπορούμε να αναπαραστήσουμε μερικά από τα αλληλεπιδραστικά πρότυπα που αναφέραμε πιο πάνω με τη χρήση διαγραμμάτων UML. Τα selectable index είναι γραμμικές δομές όπως λίστα, πίνακας ή δέντρο. Κάθε selectable index αναπαρίσταται από ένα Item, το οποίο απεικονίζεται σε ένα άλλο παράθυρο. Κάθε ένα παράθυρο απεικονίζει μόνο ένα Item. Η προσέγγιση αυτή μας επιτρέπει να παρέχουμε την αυτόματη αναγνώριση του προτύπου, όταν τα πρότυπα θα μπορούσαν να εφαρμοσθούν, βάσει των σχετικών δεδομένων. Βασισμένοι στην πιο πάνω προσέγγιση θα προσπαθήσουμε να αναπαραστήσουμε τα περιγραφικά πρότυπα των περιοχών της αλληλεπίδρασης ανθρώπου υπολογιστή και του σχεδιασμού ιστοσελίδων με τη χρήση διαγραμμάτων UML.

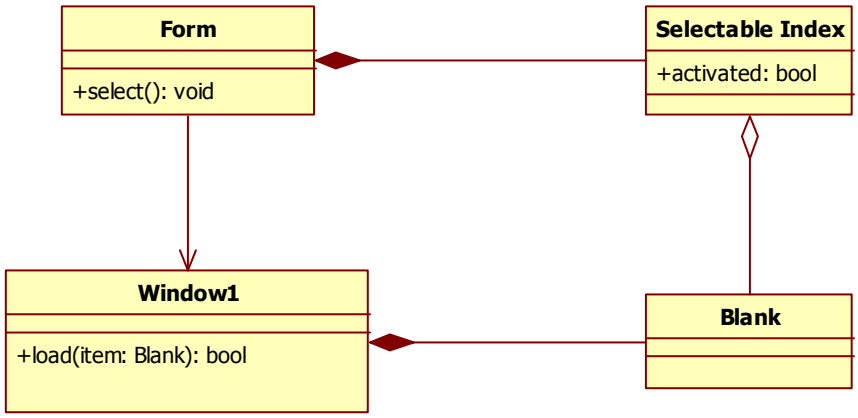
## 8.2 Μοντελοποίηση προτύπων αλληλεπίδρασης ανθρώπου υπολογιστή

### 8.2.1 Μοντελοποίηση προτύπου Narrative



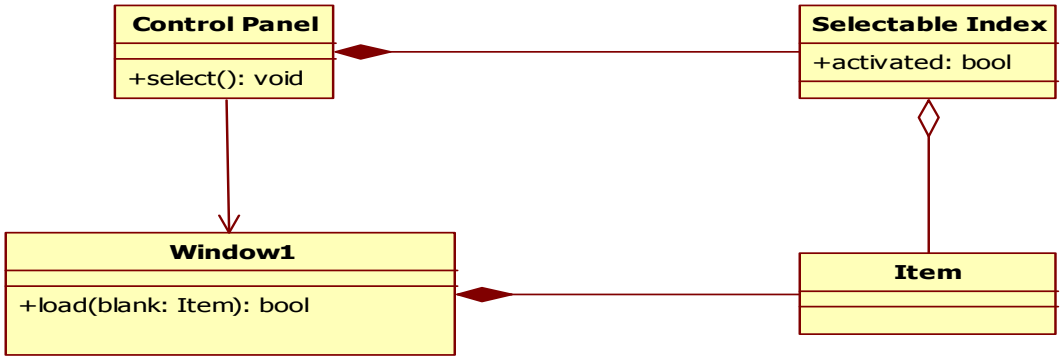

Στο πιο πάνω διάγραμμα έχουμε την κλάση Window η οποία αποτελείται από 2 ξεχωριστά παράθυρα(panes). Το ένα παράθυρο είναι για το Narrative, το οποίο αντιπροσωπεύει τη μορφή στην οποία εμφανίζεται η πληροφορία στο χρήστη και το άλλο για το Detail in natural language, το οποίο απεικονίζει την πληροφορία σε γλώσσα κατανοητή για το χρήστη. Αν ο χρήστης χρησιμοποιήσει τη μέθοδο select() της κλάσης Narrative, για να επιλέξει κάποιο δεδομένο(Selectable Index), π.χ κάποιο email, τότε η κατάσταση του θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Narrative. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο Item θα φορτωθεί στην κλάση Detail in natural language μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

### 8.2.2 Μοντελοποίηση προτύπου Form



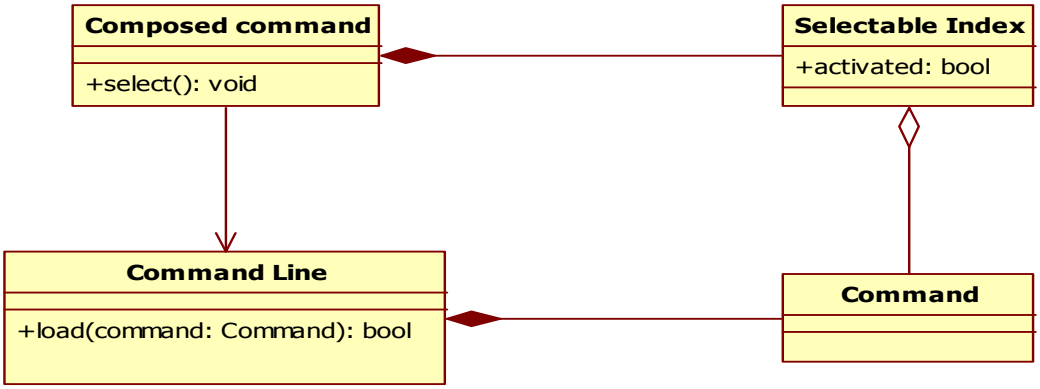
Στο πιο πάνω διάγραμμα έχουμε την κλάση Form, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select() για να επιλέξει κάποιο κενό(blank) το οποίο θα πρέπει να συμπληρώσει. Τότε, η κατάσταση του blank θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Form. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο Blank θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

### 8.2.3 Μοντελοποίηση προτύπου Control Panel



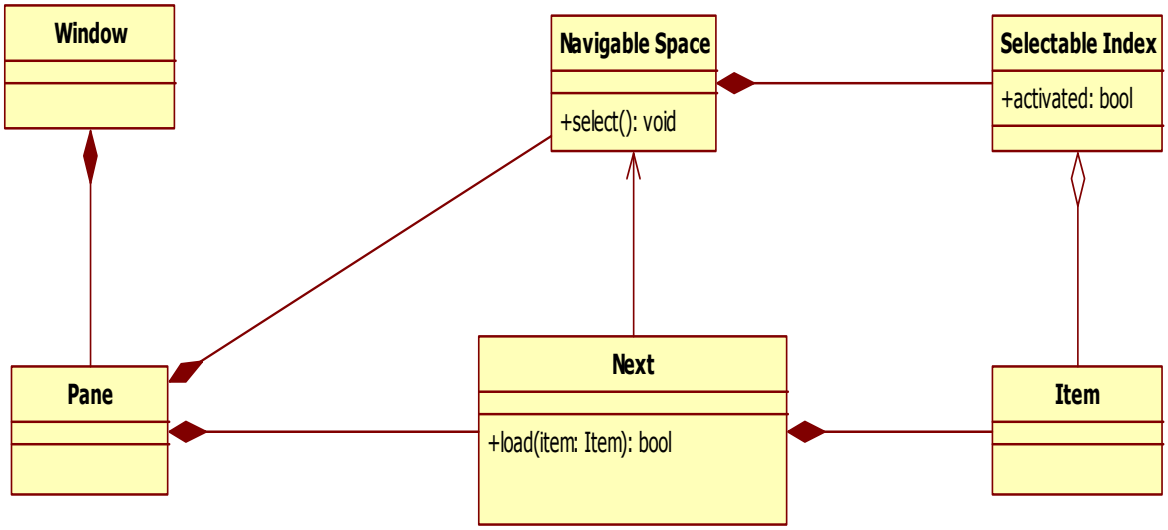

Στο πιο πάνω διάγραμμα έχουμε την κλάση Control Panel, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select() για να επιλέξει κάποιο καλά σχεδιασμένο έλεγχο(Selectable Index) που εκτελεί τη λειτουργία ή εμφανίζει την τιμή της μεταβλητής. Τότε, η κατάσταση του θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Control Panel. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο Item θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

### 8.2.4 Μοντελοποίηση προτύπου Composed Command




Στο πιο πάνω διάγραμμα έχουμε την κλάση Composed command, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select() για να επιλέξει κάποια εντολή. Τότε, η κατάσταση της εντολής θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Composed command. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, η αντίστοιχη εντολή(Command) θα φορτωθεί στην κλάση Command Line μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση της εντολής επιστρέφεται true, διαφορετικά false.

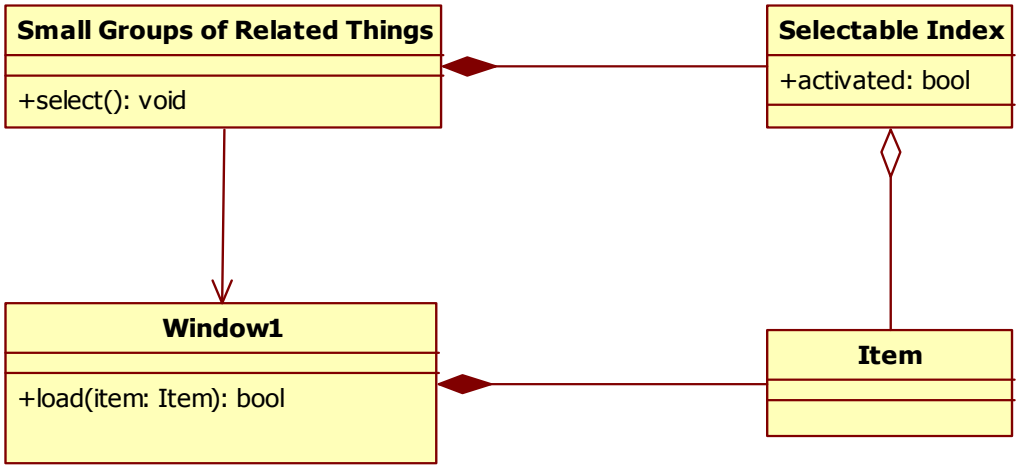
### 8.2.5 Μοντελοποίηση προτύπου Navigable Spaces




Στο πιο πάνω διάγραμμα έχουμε την κλάση Navigable Spaces, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select() για να επιλέξει κάποιο περιεχόμενο. Τότε, η κατάσταση του στοιχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Navigable Spaces.Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Κατά την επιλογή του στοιχείου(Item) από το Selectable Index, το αντίστοιχο στοιχείο φορτώνεται ως το επόμενο

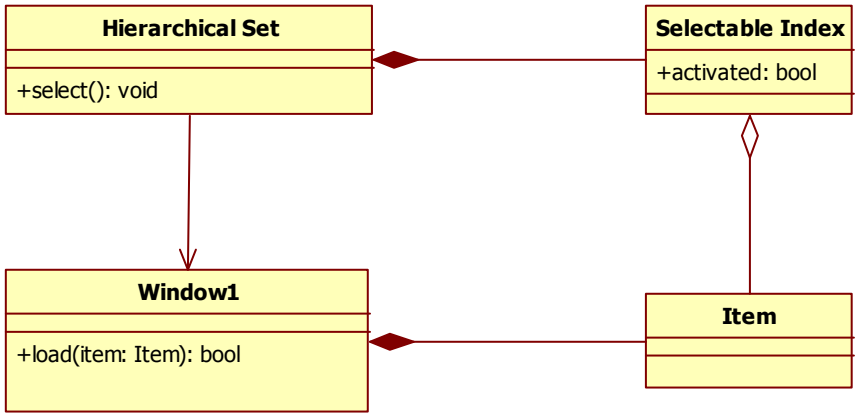
παράθυρο(Next). Ως εκ τούτου, δημιουργείται η ανάγκη για να θέσουμε κάποιο περιορισμό(constraint) στο διάγραμμα ότι πρέπει να υπάρχει 0 ή 1 τρέχουσα οθόνη κάθε φορά, η οποία θα εμφανίζεται μόνο μία φορά. Σε περίπτωση επιτυχίας στην εμφάνιση της οθόνης επιστρέφεται true, διαφορετικά false.

### 8.2.6 Μοντελοποίηση προτύπου Small Groups of Related Things



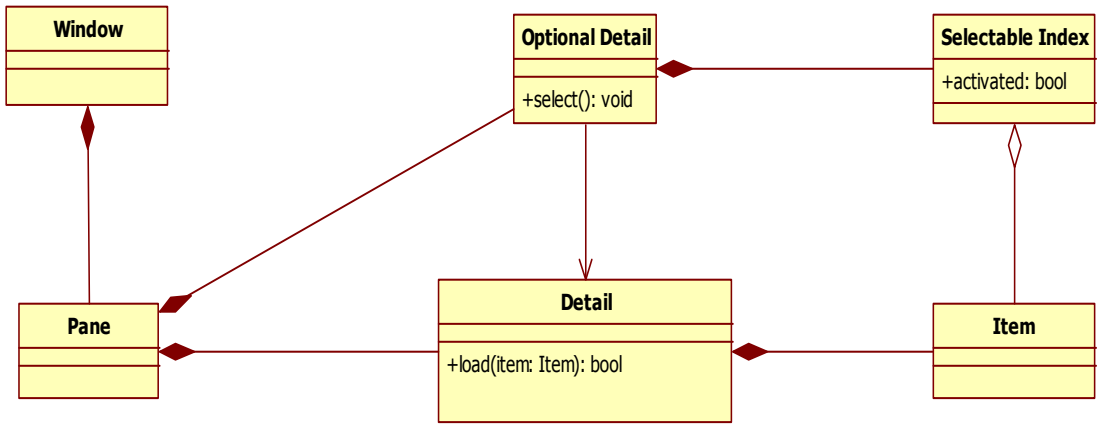

Στο πιο πάνω διάγραμμα έχουμε την κλάση Small Group of Related Things, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select() για να επιλέξει κάποιο αντικείμενο ή ενέργεια. Τότε, η κατάσταση του αντικειμένου ή της ενέργειας θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Small Group of Related Things. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο Item(αντικείμενο ή ενέργεια) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

### 8.2.7 Μοντελοποίηση προτύπου Hierarchical Set




Στο πιο πάνω διάγραμμα έχουμε την κλάση Hierarchical Set, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να επιλέξει κάποιο περιεχόμενο. Τότε, η κατάσταση του στοιχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Hierarchical Set. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο Item θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

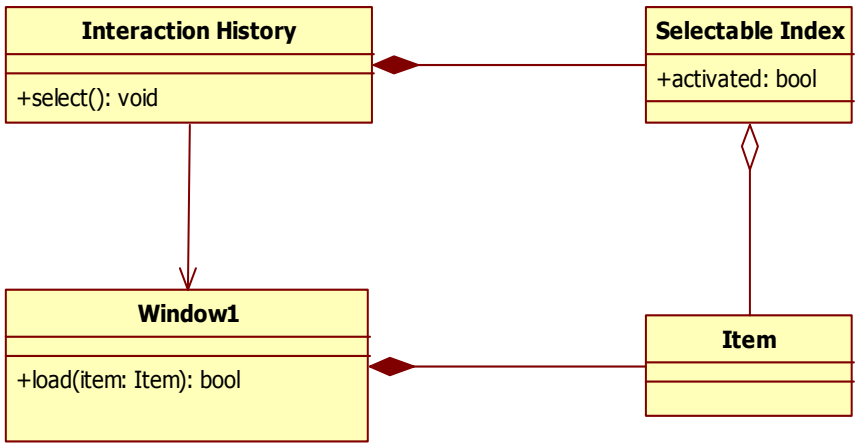
### 8.2.8 Μοντελοποίηση προτύπου Optional Detail




Στο πιο πάνω διάγραμμα έχουμε την κλάση Window η οποία αποτελείται από 2 ξεχωριστά παράθυρα(panes). Το ένα παράθυρο είναι για το Optional, το οποίο αντιπροσωπεύει τις πληροφορίες που είναι χρήσιμες στο χρήστη και το άλλο για το Detail, το οποίο απεικονίζει τις πληροφορίες που είναι αχρείαστες για το χρήστη. Αν ο χρήστης χρησιμοποιήσει τη μέθοδο select() της κλάσης Optional Detail, για να επιλέξει κάποια πληροφορία (Selectable Index), π.χ κάποιο email, τότε η κατάσταση του θα αλλάξει στο GUI και η αλλαγή θα είναι

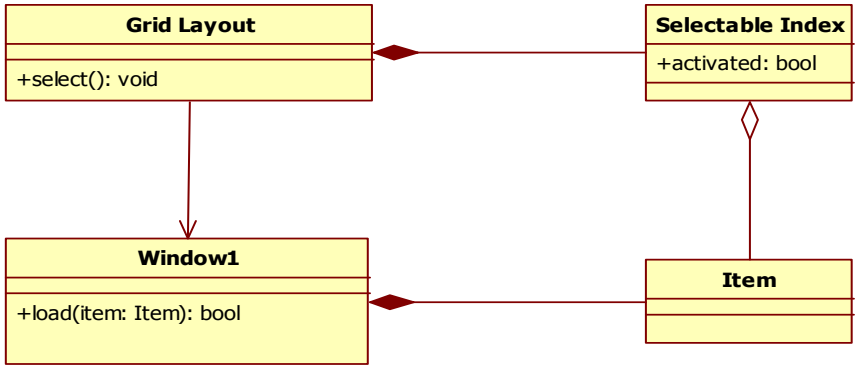
εμφανής στην κλάση Optional Detail. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο Item θα φορτωθεί στην κλάση Detail μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

### 8.2.9 Μοντελοποίηση προτύπου Interaction History




Στο πιο πάνω διάγραμμα έχουμε την κλάση Interaction History, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να επιλέξει κάποια ενέργεια. Τότε, η κατάσταση της ενέργειας θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Interaction History. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, η αντίστοιχη ενέργεια(Item) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση της ενέργειας επιστρέφεται true, διαφορετικά false.

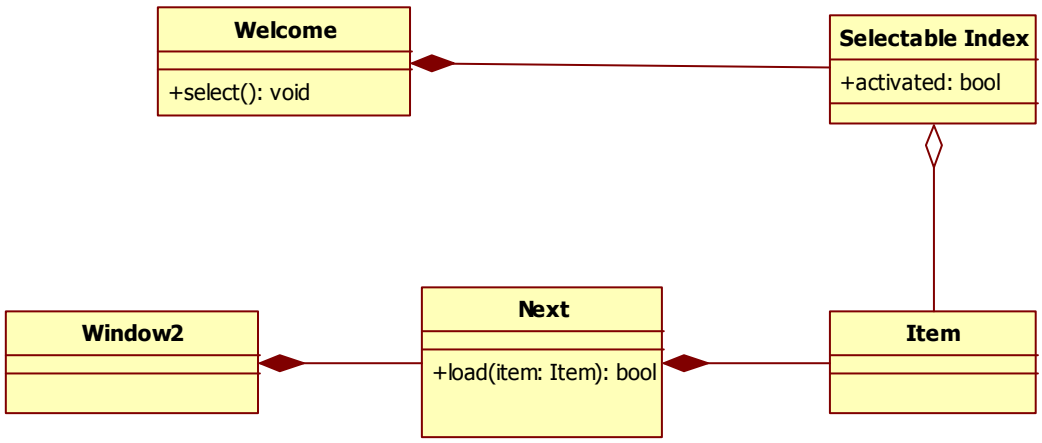
### 8.2.10 Μοντελοποίηση προτύπου Grid Layout



Στο πιο πάνω διάγραμμα έχουμε την κλάση Grid Layout, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να επιλέξει κάποιο αντικείμενο το οποίο περιέχει κάποια πληροφορία. Τότε, η κατάσταση του αντικειμένου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Grid Layout. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο αντικείμενο (Item) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του αντικειμένου επιστρέφεται true, διαφορετικά false.

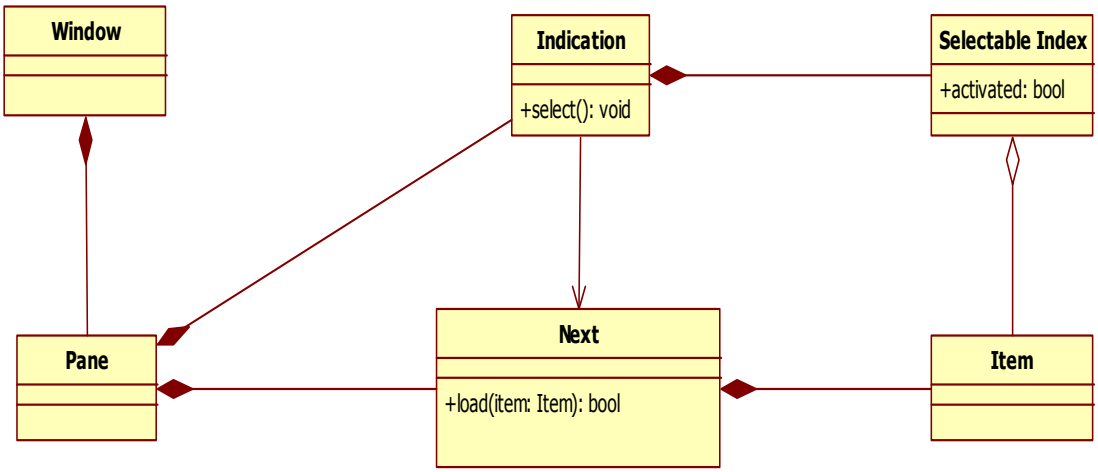
## 8.3 Μοντελοποίηση προτύπων σχεδιασμού ιστοσελίδων

### 8.3.1 Μοντελοποίηση προτύπου Welcome



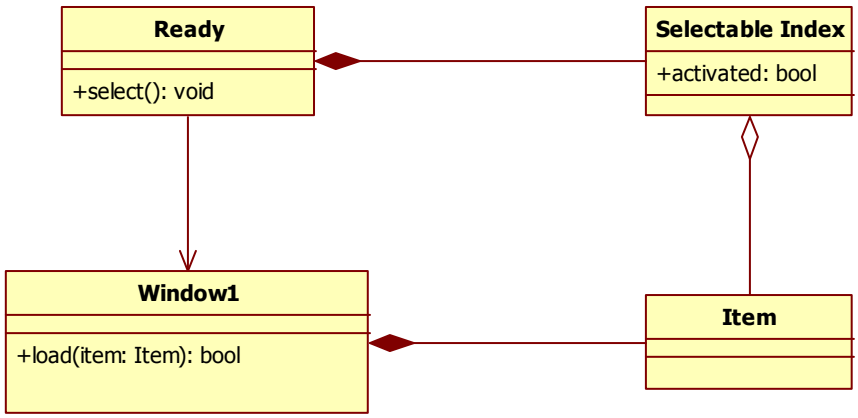

Στο πιο πάνω διάγραμμα έχουμε την κλάση Welcome, όπου ο χρήστης χρησιμοποιήσει τη μέθοδο select()για να επιλέξει κάποιο σύνδεσμο(link) για να μεταβεί σε κάποιο άλλο μέρος της ιστοσελίδας. Τότε, η κατάσταση του στοιχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Welcome. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Κατά την επιλογή του συνδέσμου(Item) από το Selectable Index, το αντίστοιχο στοιχείο φορτώνεται ως το επόμενο παράθυρο(Next). Σε περίπτωση επιτυχίας στο άνοιγμα του συνδέσμου σε νέο παράθυρο επιστρέφεται true, διαφορετικά false.

### 8.3.2 Μοντελοποίηση προτύπου Indication



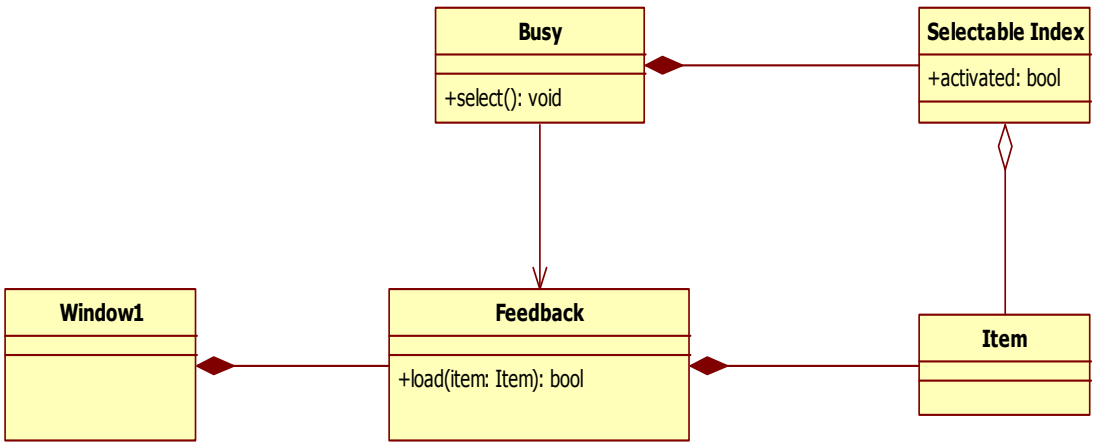

Στο πιο πάνω διάγραμμα έχουμε την κλάση Indication, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select() για να επιλέξει κάποιο σύνδεσμο. Τότε, η κατάσταση του συνδέσμου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Indication.Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Κατά την επιλογή του συνδέσμου(Item) από το Selectable Index, το αντίστοιχο στοιχείο φορτώνεται ως το επόμενο παράθυρο(Next). Ως εκ τούτου, δημιουργείται η ανάγκη για να θέσουμε κάποιο περιορισμό(constraint) στο διάγραμμα ότι πρέπει να υπάρχει 0 ή 1 τρέχουσα οθόνη κάθε φορά, η οποία θα εμφανίζεται μόνο μία φορά. Σε περίπτωση επιτυχίας στην εμφάνιση της οθόνης επιστρέφεται true, διαφορετικά false.

### 8.3.3 Μοντελοποίηση προτύπου Ready



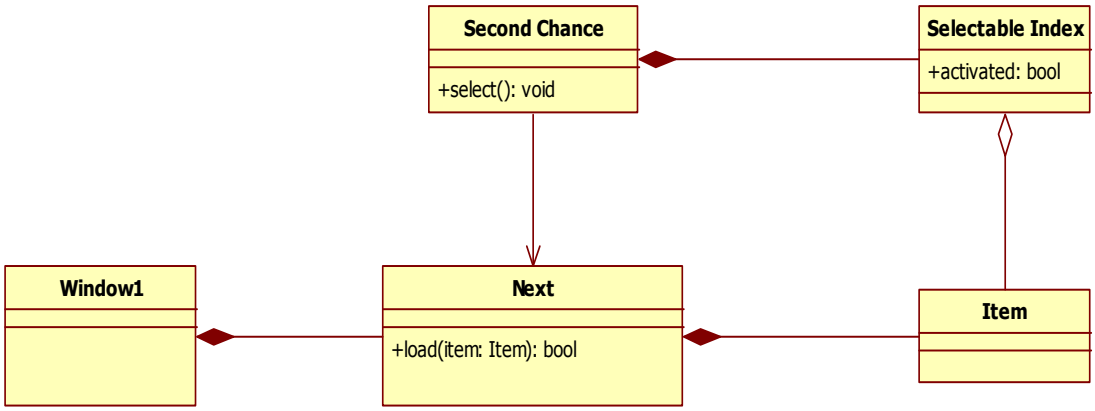
Στο πιο πάνω διάγραμμα έχουμε την κλάση Ready, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να επιλέξει το κατάλληλο plug-in το οποίο είναι απαραίτητο για την είσοδο του στην ιστοσελίδα. Τότε, η κατάσταση του plug-in θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Ready. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο plug-in (Item) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του plug-in επιστρέφεται true, διαφορετικά false.

### 8.3.4 Μοντελοποίηση προτύπου Busy



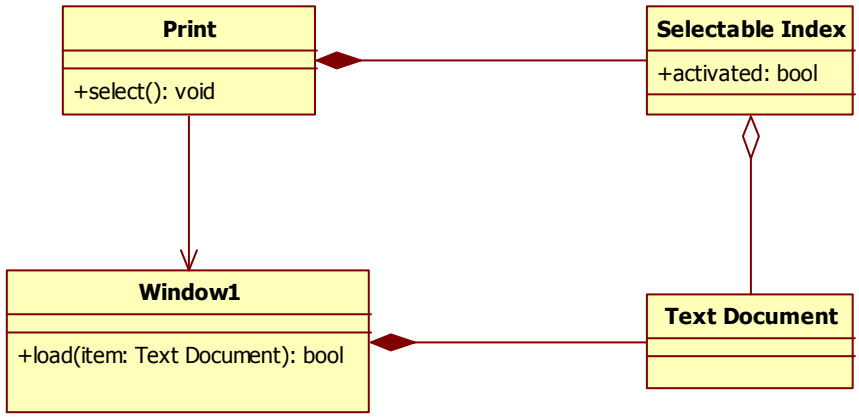

Στο πιο πάνω διάγραμμα έχουμε την κλάση Busy, όπου ο χρήστης χρησιμοποιήσει τη μέθοδο select()για να κατεβάσει κάποιο αρχείο, εικόνα ή εφαρμογή. Τότε, η κατάσταση του στοιχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Welcome. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Κατά την επιλογή του στοιχείου(Item) από το Selectable Index, το αντίστοιχο στοιχείο φορτώνεται στην κλάση Feedback η οποία επιστρέφει το χρόνο που χρειάζεται για να ολοκληρωθεί η εν λόγω διεργασία. Σε περίπτωση επιτυχίας στην επιστροφή του χρόνου επιστρέφεται true, διαφορετικά false.

### 8.3.5 Μοντελοποίηση προτύπου Second Chance




Στο πιο πάνω διάγραμμα έχουμε την κλάση Second Chance, όπου ο χρήστης χρησιμοποιήσει τη μέθοδο select()για να επιλέξει κάποιο μηχανισμό που θα του επιτρέψει να έχει τον έλεγχο των ενεργειών του. Τότε, η κατάσταση του μηχανισμού θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Second Chance. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Κατά την επιλογή του μηχανισμού (Item) από το Selectable Index, το αντίστοιχο στοιχείο φορτώνεται ως το επόμενο παράθυρο(Next). Σε περίπτωση επιτυχίας στο άνοιγμα του μηχανισμού σε νέο παράθυρο επιστρέφεται true, διαφορετικά false.

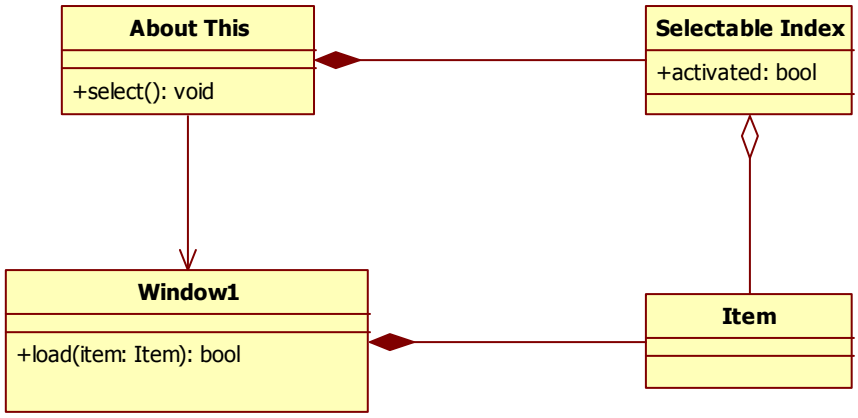
### 8.3.6 Μοντελοποίηση προτύπου Print




Στο πιο πάνω διάγραμμα έχουμε την κλάση Print, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να επιλέξει το αρχείο(Text document) που θέλει να τυπώσει. Τότε, η κατάσταση του αρχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Print. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το

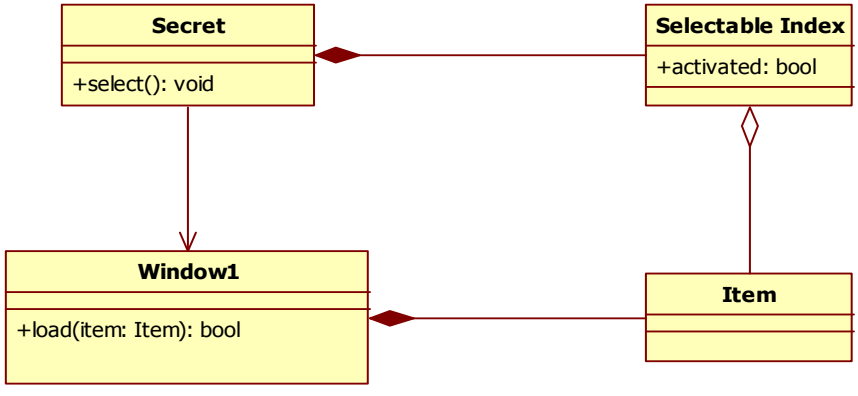
αντίστοιχο αρχείο (Text document) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του αρχείου επιστρέφεται true, διαφορετικά false.

### 8.3.7 Μοντελοποίηση προτύπου About This



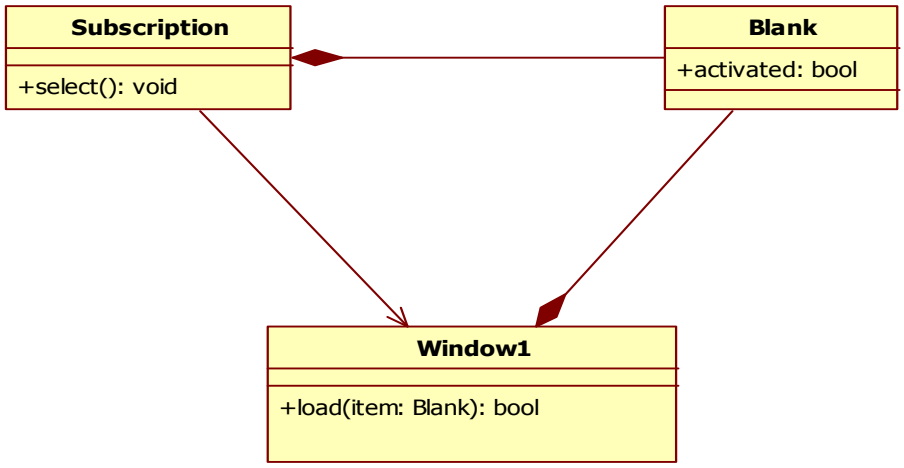

Στο πιο πάνω διάγραμμα έχουμε την κλάση About This, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να επιλέξει τον σύνδεσμο που παραπέμπει σε πληροφορίες της ιστοσελίδας που έχει επισκεφτεί. Τότε, η κατάσταση του συνδέσμου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση About This. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, ο αντίστοιχος σύνδεσμος (Item) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στο άνοιγμα του συνδέσμου επιστρέφεται true, διαφορετικά false.

### 8.3.8 Μοντελοποίηση προτύπου Secret



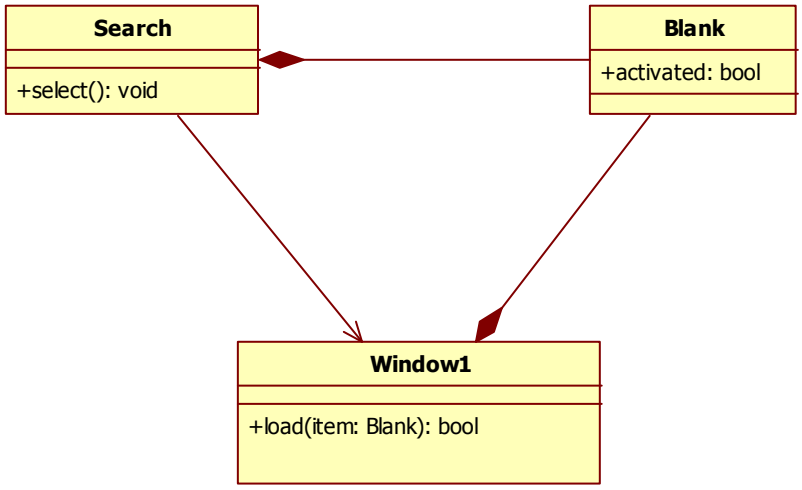
Στο πιο πάνω διάγραμμα έχουμε την κλάση Secret, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να συμπληρώσει το username και το password του. Τότε, η κατάσταση του στοιχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Secret. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο στοιχείο(Item) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

### 8.3.9 Μοντελοποίηση προτύπου Subscription




Στο πιο πάνω διάγραμμα έχουμε την κλάση Subscription, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να συμπληρώσει το κενό(Blank) που αντιστοιχεί στο e-mail του. Τότε, η κατάσταση του στοιχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Subscription. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο κενό (Blank) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

### 8.3.10 Μοντελοποίηση προτύπου Search




Στο πιο πάνω διάγραμμα έχουμε την κλάση Search, όπου ο χρήστης χρησιμοποιεί τη μέθοδο select()για να συμπληρώσει το κενό(Blank) που αντιστοιχεί στη μηχανή αναζήτησης. Τότε, η κατάσταση του στοιχείου θα αλλάξει στο GUI και η αλλαγή θα είναι εμφανής στην κλάση Subscription. Αυτό θα έχει ως αποτέλεσμα την αλλαγή της μεταβλητής activated=true. Ως αποτέλεσμα, το αντίστοιχο κενό (Blank) θα φορτωθεί στην κλάση Window1 μέσω της κλήσης της μεθόδου load().Σε περίπτωση επιτυχίας στην εμφάνιση του στοιχείου επιστρέφεται true, διαφορετικά false.

# Κεφάλαιο 9

## Συμπεράσματα και Μελλοντική Εργασία

9.1 Συμπεράσματα 97
9.2 Μελλοντική Εργασία 97

### 9.1 Συμπεράσματα

Σε αυτή την διπλωματική εργασία κάναμε μια σφαιρική ανασκόπηση των κυριοτέρων σχεδιαστικών προτύπων που χρησιμοποιούνται σε διάφορες ερευνητικές περιοχές στην επιστήμη της Πληροφορικής όπως Τεχνολογία Λογισμικού, Αλληλεπίδραση ανθρώπου-υπολογιστή, Συστήματα Πραγματικού Χρόνου και Σχεδιασμού Ιστοσελίδων. Η εργασία αυτή θεωρώ πως μπορεί να αποτελέσει ένα χρήσιμο εγχειρίδιο για τον οποιονδήποτε που θέλει να εμβαθύνει τις γνώσεις του στους σχετικούς τομείς της Πληροφορικής.
Μέσα από την ανάλυση όλων αυτών των προτύπων διαπιστώσαμε πως κάποια από αυτά είναι περιγραφικά και κάποια άλλα αναπαρίστανται με τη χρήση διαγραμμάτων UML. Αυτό ήταν η αφορμή για την τελευταία ενότητα της διπλωματικής μας εργασίας, όπου είδαμε πόσο εύκολο ή δύσκολο ήταν να μοντελοποιήσουμε τα περιγραφικά πρότυπα στη UML. Μερικά πρότυπα είχαν ομοιότητες με κάποια άλλα κάτι που έκανε πιο εύκολη την προσπάθεια φορμαλισμού, ωστόσο κάποια άλλα πρότυπα ήταν τόσο σύντομα περιγραφικά κάτι που έκανε την αναπαράσταση πιο δύσκολη.

### 9.2 Μελλοντική Εργασία

Μία μελλοντική εργασία θα μπορούσε να ήταν η μελέτη προτύπων σε άλλους τομείς της πληροφορικής. Θα μπορούσε επίσης να δημιουργηθεί κάποια εφαρμογή σε κάποια γλώσσα προγραμματισμού που να φαίνεται η χρήση αυτών των προτύπων(Τεχνολογία Λογισμικού), η ανάπτυξη κάποιου διαδραστικού συστήματος ακολουθώντας τα σχετικά πρότυπα(Αλληλε-πίδραση ανθρώπου-υπολογιστή) ή ο σχεδιασμός μιας ιστοσελίδας βάση της προτεινόμενης γλώσσας σχεδιαστικών προτύπων(Σχεδιασμός Ιστοσελίδων).

# Βιβλιογραφία


[1] Design Patterns: Elements of Reusable Object-Oriented Software, Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides.

[2] Software Architecture Design Patterns in Java,Partha Kuchana.

[3] Beck, K.Coplien, J. O.; Crocker, R,; Dominick, L. Meszaros G. Paulisch, F. Vlissides, J. Industrial Experience with Design Patterns.

[4] Borchers, J. O. A Pattern Approach to Interaction Design. John Wiley & Sons, 2001.

[5] M. J Mahemoff and L. J. Johnston. Pattern Language for Usability: An Investigation of Alternative Approaches. In J. Tanaka, editor, APCHI '98 Proceedings, pages 25-31. IEEE Computer Society, Los Alamitos, CA. 1998

[6] Patterns Home Page. http://hillside.net/patterns/.

[7] Preece, J. Human-Computer Interaction. Addison-Wesley Pub Co; 1st edition. 1994.

[8] Tidwell, J. Interaction Design Patterns. PLoP'98 Conference on Pattern Languages of Programming, Illinois, extended version including Common Ground: A Pattern Language for Human-Computer Interface Design at http://www.mit.edu/ ˜ jtidwell/interaction patterns.html. 1999.

[9] M. van Welie; Traetteberg, H. Interaction Patterns in User Interfaces. In: 7th. Pattern Languages of Programs Conference, Allerton Park Monticello, Illinois, USA, 13-16 August 2000.

[10] M. van Welie; G. C. van der Veer; Eliëns, A. Patterns as Tools for User Interface Design. In: International Workshop on Tools for Working with Guidelines, Biarritz, France, 7-8 October 2000.

[11] Douglass, Bruce Powel Real-Time UML: Efficient Objects for Embedded Systems, Reading, MA: Addison-Wesley-Longman, 1998

[12] A Practitioner's Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems by Klein, Ralya, Pollak, Obenza, and Harbour, Kluwer Academic Publishers, 1993.

[13] Safety Critical Systems by Bruce Powel Douglass, Embedded Systems Conference – Spring, 1998.

[14] Constantine Stephanidis, Anthony Savidis. “Universal Access in the Information Society: Methods, Tools and Interaction Technologies.” Springer-Verlang. 2001.

[15] Mary Evans. “Web Design: An Empiricist’s Guide”. University of Whasington, Seatle, Washington. 1998.

[16] Fernando Lyardet, Gustavo Rossi. Web Usability Patterns.2000

[17] Jakob Nielsen, Marie Tahir. “Homepage Usability: 50 Websites deconstructed ”. New Riders. 2002