
Αρχιτεκτονική Λογισμικού

περιεχόμενα παρουσίασης

- Τι είναι η αρχιτεκτονική λογισμικού
- Αρχιτεκτονική και απαιτήσεις
- Σενάρια ποιότητας
- Βήματα αρχιτεκτονικής σχεδίασης
- Αρχιτεκτονικά πρότυπα
- Διαστρωματωμένη αρχιτεκτονική
- Μοντέλο – όψη – ελεγκτής
- Τεκμηρίωση αρχιτεκτονικής

τι είναι η αρχιτεκτονική λογισμικού

- Η αρχιτεκτονική λογισμικού ενός συστήματος είναι η δομή (ή οι δομές) του συστήματος, οι σχέσεις μεταξύ των δομικών στοιχείων της δομής (ή των δομών), καθώς και οι ιδιότητες αυτών των στοιχείων.
- Εάν η σχεδίαση είναι το σύνολο των αποφάσεων που αφορούν τον τρόπο επίλυσης του προβλήματος, τότε η αρχιτεκτονική σχεδίαση είναι το σύνολο των αποφάσεων σχεδίασης που είναι δύσκολο να αλλάξουν.

η σημασία της αρχιτεκτονικής

- Η αρχιτεκτονική βοηθά στην κατανόηση του συστήματος.
- Η κατανόηση της αρχιτεκτονικής από όλους τους ενδιαφερομένους βοηθά και την επικοινωνία των ενδιαφερομένων σε σχέση με το σύστημα.
- Ο προσδιορισμός της αρχιτεκτονικής βοηθά στην οργάνωση του τρόπου ανάπτυξης του λογισμικού.
- Η αρχιτεκτονική παρέχει μία σταθερή βάση πάνω στην οποία θα βασιστεί η συντήρηση του λογισμικού.
- Η αρχιτεκτονική βοηθά στην οργάνωση του έργου.

αρχιτεκτονική και απαιτήσεις

- Η δημιουργία της αρχιτεκτονικής του λογισμικού δεν επηρεάζεται μόνο από τις λειτουργικές απαιτήσεις αλλά και από τις μη λειτουργικές απαιτήσεις και τους περιορισμούς σχεδίασης.
- Ανάλογα με τη φύση του λογισμικού οι απαιτήσεις που αφορούν κάποια χαρακτηριστικά έχουν μεγαλύτερη βαρύτητα από κάποιες άλλες.
- Οι απαιτήσεις με τη μεγαλύτερη βαρύτητα και επηρεάζουν και την αρχιτεκτονική του λογισμικού, καλούνται και **αρχιτεκτονικοί παράγοντες** (architectural factors) ή **αρχιτεκτονικοί οδηγοί** (architectural drivers).
- Υπάρχει αλληλεπίδραση μεταξύ των μη λειτουργικών απαιτήσεων και ποιοτικών χαρακτηριστικών του λογισμικού

αλληλεπίδραση ποιοτικών χαρακτηριστικών

	Απόδοση	Ευελιξία	Ακεραιότητα	Διαλειτουργικότητα	Συντηρησιμότητα	∴
Απόδοση		-		-	-	
Ευελιξία	-		-		+	
Ακεραιότητα	-			-	+	
Διαλειτουργικότητα	-	+	-			
Συντηρησιμότητα	-	+				
....						

αρχιτεκτονική σχεδίαση

- Για τη σχεδίαση της αρχιτεκτονικής του λογισμικού ο αρχιτέκτονας επανεξετάζει και αποσαφηνίζει τις μη λειτουργικές απαιτήσεις, όταν αυτές δεν είναι μετρήσιμες.
- Μία μέθοδος για τον καθορισμό μετρήσιμων μεγεθών για την αξιολόγηση των ποιοτικών χαρακτηριστικών είναι η δημιουργία **σεναρίων ποιότητας** (quality scenarios).
- Με τον ίδιο τρόπο που οι περιπτώσεις χρήσης μας παρέχουν τα σενάρια της λειτουργικότητας του λογισμικού, τα σενάρια ποιότητας καθορίζουν μετρήσιμα κριτήρια για την επίτευξη των κριτηρίων που θέτουν οι μη λειτουργικές απαιτήσεις.

σενάρια ποιότητας

Κάθε σενάριο ποιότητας αποτελείται από τα παρακάτω στοιχεία:

- **Πηγή του ερεθίσματος.** Μπορεί να είναι κάποιος χρήστης ή κάποιο εξωτερικό σύστημα που γεννά το ερέθισμα.
- **Ερέθισμα.** Η συνθήκη την οποία θα πρέπει να διαχειριστεί το σύστημα.
- **Περιβάλλον.** Το γενικό περιβάλλον κάτω από το οποίο γεννάται το ερέθισμα. Π.χ. το σύστημα είναι υπερφορτωμένο.
- **Προϊόν (artifact).** Το προϊόν του συστήματος μπορεί να είναι ένα τμήμα ή το σύνολο του συστήματος.
- **Απόκριση.** Αυτό που πρέπει να γίνει για τη διαχείριση του ερεθίσματος.
- **Μέτρηση Απόκρισης.** Η μέτρηση της απόκρισης μπορεί να αφορά χρόνους απόκρισης, κόστη κ.λπ.

παράδειγμα: πρότυπο σεναρίων διαθεσιμότητας

Πηγή ερεθίσματος	Εσωτερική στο σύστημα ή εξωτερική του συστήματος.
Ερέθισμα	Παράλειψη, κατάρρευση, χρονισμός, απόκριση.
Περιβάλλον	Κανονική λειτουργία, μειωμένων δυνατοτήτων.
Προϊόν	Επεξεργαστές, κανάλια επικοινωνίας, συσκευές αποθήκευσης δεδομένων, διεργασίες.
Απόκριση	Εγγραφή στο ημερολόγιο συμβάντων, ενημέρωση διαχειριστών και χρηστών, απενεργοποίηση των πηγών των γεγονότων που προκαλούν το σφάλμα, μη διαθεσιμότητα του συστήματος για κάποιο χρονικό διάστημα, συνέχιση της λειτουργίας σε κανονική λειτουργία ή λειτουργία μειωμένων δυνατοτήτων.
Μέτρηση απόκρισης	Χρονικό διάστημα στο οποίο θα πρέπει το σύστημα να είναι διαθέσιμο, ποσοστό διαθεσιμότητας, χρονικό διάστημα λειτουργίας σε κατάσταση μειωμένων δυνατοτήτων, χρόνος επιδιόρθωσης.

βήματα αρχιτεκτονικής σχεδίασης

- Επιλογή του δομικού στοιχείου προς αποσύνθεση. Όταν ξεκινάμε τη σχεδίαση, η μονάδα προς αποσύνθεση είναι το σύστημα ως ολότητα. Η προϋπόθεση της αποσύνθεσης είναι η ύπαρξη των λειτουργικών απαιτήσεων, των περιορισμών και των σεναρίων ποιότητας. Σε πρώτη φάση η αποσύνθεση θα έχει ως αποτέλεσμα τα βασικά υποσυστήματα του λογισμικού.
- Εκλέπτυνση του δομικού στοιχείου σύμφωνα με τα παρακάτω βήματα:
 - Επιλογή των αρχιτεκτονικών παραγόντων από τις λειτουργικές απαιτήσεις και τα σενάρια ποιότητας. Αυτό το βήμα καθορίζει τι είναι σημαντικό για την αποσύνθεση που θα επακολουθήσει.
 - Καθορισμός των αρχιτεκτονικών αποφάσεων σχεδίασης οι οποίες θα χρησιμοποιηθούν για την κάλυψη των αρχιτεκτονικών παραγόντων. Οι αρχιτεκτονικές αυτές αποφάσεις καθοδηγούνται από τους αρχιτεκτονικούς παράγοντες.

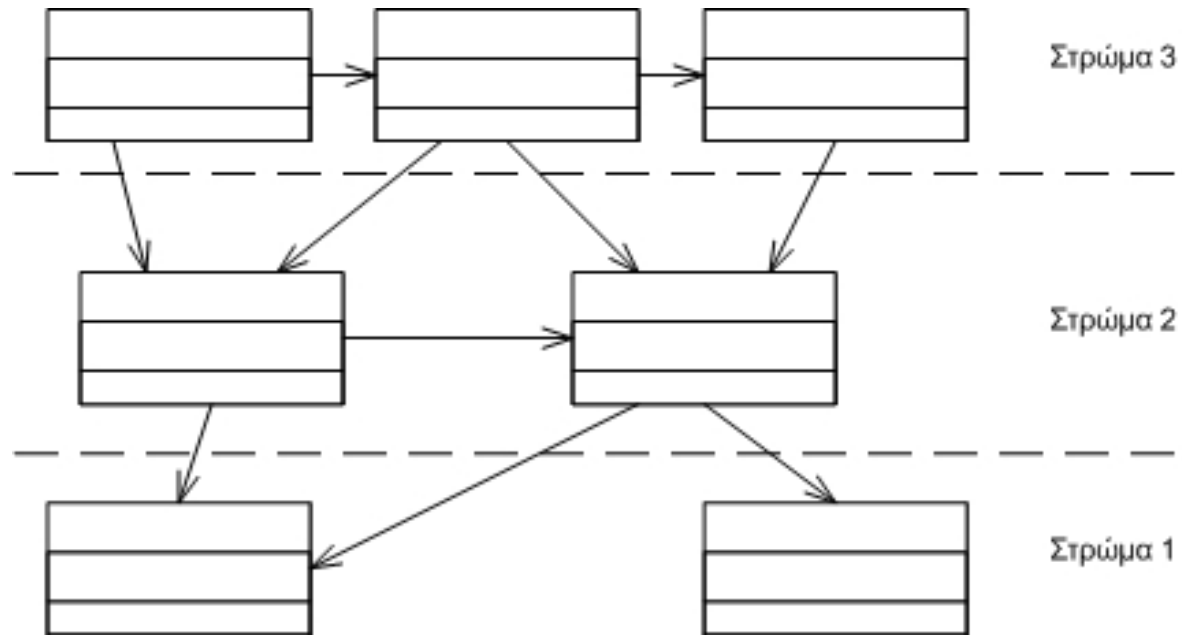
βήματα αρχιτεκτονικής σχεδίασης

- Ορισμός των δομικών μονάδων απογόνων και ο καθορισμός της λειτουργικότητας σε κάθε δομική μονάδα απόγονο. Τεκμηρίωση των δομικών μονάδων.
- Καθορισμός των διεπαφών των δομικών μονάδων απογόνων. Η αποσύνθεση παρέχει μονάδες και περιορισμούς για τον τρόπο επικοινωνίας μεταξύ των μονάδων. Τεκμηρίωση των διεπαφών κάθε μονάδας.
- Επαλήθευση και εκλέπτυνση των λειτουργικών απαιτήσεων και των σεναρίων ποιότητας και μετατροπή τους σε περιορισμούς των μονάδων-απογόνων. Αυτό το βήμα επαληθεύει ότι τίποτα το σημαντικό δεν έχει ξεχαστεί και προετοιμάζει τις μονάδες απογόνους για περαιτέρω αποσύνθεση ή υλοποίηση.
- Επανάληψη των παραπάνω βημάτων για κάθε δομική μονάδα που χρειάζεται αποσύνθεση

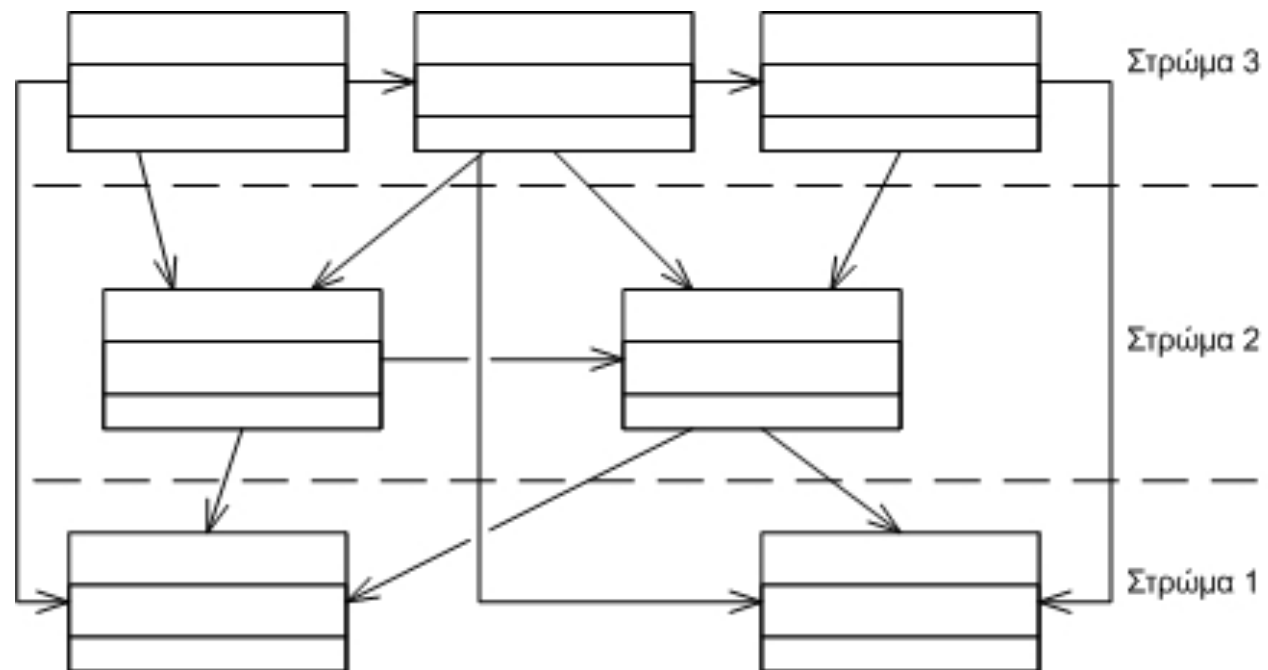
αρχιτεκτονικά πρότυπα

- Η διαδικασία προσδιορισμού μίας αρχιτεκτονική «από το μηδέν» δεν είναι ο μόνος τρόπος.
- Μπορούμε να αξιοποιήσουμε την εμπειρία μας για τις καλές πρακτικές οργάνωσης της αρχιτεκτονική στο πλαίσιο των αρχιτεκτονικών παραγόντων που μελετούμε.
- Η αξιοποίηση αυτής της εμπειρίας απεικονίζεται στα **αρχιτεκτονικά πρότυπα** (architectural patterns) ή **αρχιτεκτονικά στυλ** (architectural styles).
- Παραδείγματα αρχιτεκτονικών προτύπων: Διαστρωματωμένη αρχιτεκτονική (Layered Architecture) και Μοντέλο – Όψη – Ελεγκτής (Model – View – Controller)

διαστρωματωμένη αρχιτεκτονική



χαλαρή διαστρωμάτωση



πλεονεκτήματα διαστρωμάτωσης

- Αυξάνεται η συνεκτικότητα του λογισμικού. Κάθε στρώμα παρέχει ένα σύνολο συνεκτικών υπηρεσιών.
- Μπορούμε να κατανοήσουμε ένα επίπεδο, χωρίς να μας απασχολούν τα υπόλοιπα.
- Πολλές αλλαγές στο λογισμικό έχουν τοπικό χαρακτήρα.
- Μπορούμε να υποκαταστήσουμε τις υπηρεσίες ενός στρώματος με κάποια άλλη υλοποίηση.

πλεονεκτήματα διαστρωμάτωσης

- Παρέχονται περισσότερες ευκαιρίες ελέγχου των εξαρτήσεων.
- Διαφορετικές ευθύνες των στρωμάτων καθοδηγούν και τη διανομή της υλοποίησης στην ομάδα ανάπτυξης.
- Δίνονται περισσότερες ευκαιρίες διανομής των μονάδων λογισμικού σε διαφορετικούς κόμβους επεξεργασίας, υιοθετώντας κατανεμημένες αρχιτεκτονικές.
- Ένα (χαμηλού επιπέδου) στρώμα μπορεί να επαναχρησιμοποιηθεί σε πολλά διαφορετικά προϊόντα λογισμικού.

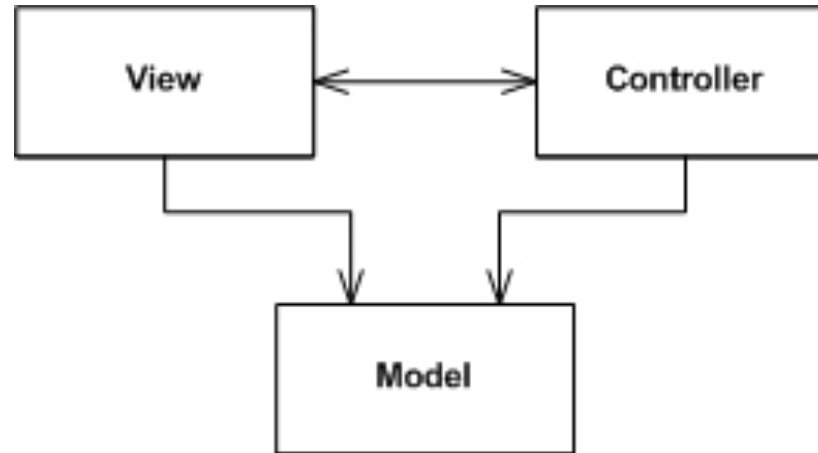
προβλήματα διαστρωμάτωσης

- Η ενθυλάκωση μεταξύ στρωμάτων δεν είναι πάντα πλήρης. Αλλαγές σε χαμηλότερο επίπεδο μπορεί να επεκτείνονται έως και τα ανώτερα στρώματα.
- Η διαστρωμάτωση πιθανόν να μειώνει την απόδοση του λογισμικού.
- Ο αριθμός των στρωμάτων επηρεάζει κατά πολύ τη συνολική αρχιτεκτονική, αλλά σε πολύπλοκα συστήματα ο ακριβής προσδιορισμός των στρωμάτων είναι αρκετά δύσκολος.

τρία στρώματα πληροφοριακών συστημάτων

Στρώμα	Ευθύνες
Παρουσίαση	Υπηρεσίες που σχετίζονται με την παρουσίαση της πληροφορίας με τις διεπαφές χρήστη
Πεδίου	Η λογική του πεδίου όπου είναι ο κορμός της λειτουργικότητας του λογισμικού
Πηγές Δεδομένων	Επικοινωνία με βάσεις δεδομένων ή άλλες πηγές δεδομένων

μοντέλο – όψη – ελεγκτής (MVC)



αρχές προτύπου MVC

- Ο διαχωρισμός της αρμοδιότητας της παρουσίασης της πληροφορίας από την ίδια την πληροφορία.
- Η όψη εξαρτάται από το μοντέλο αλλά όχι το αντίστροφο. Είναι η σημαντικότερη αρχή του προτύπου και μία από τις σημαντικότερες αρχές στη σχεδίαση διεπαφών χρήστη.
- Ο διαχωρισμός της μετάφρασης των ενεργειών και των εντολών του τελικού χρήστη από την όψη σε διαφορετικό αντικείμενο που είναι ο ελεγκτής. Η αρχή αυτή είναι λιγότερο σημαντική.
- Σε ορισμένες περιπτώσεις τα αντικείμενα της διεπαφής χρήστη μπορεί να συμπεριλαμβάνουν και τη λογική των ελεγκτών.

πλεονεκτήματα προτύπου MVC

- Η ίδια πληροφορία μπορεί να παρουσιαστεί με διαφορετικές μορφές.
- Διευκολύνονται οι αλλαγές στη διεπαφή χρήστη.
- Μπορεί να υπάρξουν εναλλάξιμες διεπαφές χρήσης.
- Τροποποίηση στη λογική του πεδίου ιδιαίτερα, όταν αυτή υλοποιεί επιχειρησιακούς κανόνες, δεν επηρεάζει τις όψεις και τους ελεγκτές. Ακόμα και όταν τους επηρεάζει, οι απαιτούμενες αλλαγές έχουν τοπικό χαρακτήρα και εντοπίζονται σχετικά εύκολα.
- Διευκολύνεται ο έλεγχος. Το μοντέλο ελέγχεται ανεξάρτητα από τους ελεγκτές και τις όψεις. Ο έλεγχος μίας συμπεριφοράς για κάποιο γεγονός μπορεί να γίνει μόνο με τη χρήση των ελεγκτών και του μοντέλου, χωρίς να χρησιμοποιηθεί η όψη.

συστατικά και πλατφόρμες

- Ένας από τους παράγοντες επιτυχίας ενός έργου ανάπτυξης είναι η χρήση έτοιμων και ώριμων τεχνολογικών υποδομών. Η χρήση έτοιμων υποδομών μάς δίνει τη δυνατότητα να εστιάσουμε την προσπάθειά μας στην επίλυση του προβλήματος του πελάτη και των χρηστών, χωρίς να αναλωνόμαστε στη δημιουργία κώδικα για τεχνικές υπηρεσίες, οι οποίες αφενός είναι δύσκολες στην υλοποίησή τους αφετέρου επιρρεπείς σε σφάλματα.
- Θα εξετάσουμε δύο διαφορετικούς τρόπους με τους οποίους μπορούμε να επιλύσουμε ορισμένα από αυτά τα προβλήματα.
- Ο πρώτος τρόπος είναι η χρήση των συστατικών (components) με τα οποία γίνεται επαναχρησιμοποίηση του λογισμικού
- και ο δεύτερος είναι η χρήση των πλατφορμών (frameworks) με τις οποίες γίνεται επαναχρησιμοποίηση του λογισμικού αλλά και της σχεδίασης.

συστατικά

- Θα θέλαμε να χρησιμοποιούμε έτοιμες μονάδες λογισμικού, οι οποίες έχουν κατασκευαστεί από τρίτους και να τις «συνδέουμε» με κάποιο τρόπο. Η σύνθεση των επιμέρους μονάδων θα παρείχε μία συνολικότερη υπηρεσία από αυτή των μονάδων.
- Αυτή είναι η επιδίωξη και των συστατικών (components). Ο στόχος των συστατικών είναι η δημιουργία και η χρήση ανεξάρτητων μονάδων λογισμικού, πιθανόν από διαφορετικούς κατασκευαστές, που όμως διασυνδέονται μεταξύ τους και παρέχουν κάποια συγκεκριμένη υπηρεσία.
- Τα πλεονεκτήματα χρήσης των συστατικών είναι εμφανή. Είναι έτοιμα τμήματα του λογισμικού τα οποία έχουν ήδη ελεγχθεί και διατίθενται προς χρήση.

συστατικά

- Ένα απλό πακέτο δεν είναι συστατικό. Ένα πακέτο παρέχει μία λογική οργάνωση των κλάσεων χωρίς κάποια αναφορά στην φυσική του αντικατάσταση. Ένα πακέτο έχει κρυφά και μη κρυφά στοιχεία. Ένα συστατικό έχει μόνο δημόσιες διεπαφές και μάλιστα πολλές φορές απαιτεί την υλοποίηση κάποιων διεπαφών από τον κώδικα του πελάτη.
- Μία βιβλιοθήκη κλάσεων δεν είναι συστατικό. Οι κλάσεις σε μία βιβλιοθήκη μπορεί να είναι σχετικές μεταξύ τους, δεν εξυπηρετούν όμως ένα προδιαγεγραμμένο σκοπό. Κάθε συστατικό το συνοδεύει μία προδιαγραφή για το σκοπό του.

ιδιότητες συστατικών

- Ένα συστατικό συμμορφώνεται με κάποιο στάνταρ, όπως το JavaBeans το Microsoft COM+ και το Enterprise JavaBeans.
- Ένα συστατικό έχει κάποια προδιαγραφή. Η προδιαγραφή ενός συστατικού είναι το σύνολο των διεπαφών του.
- Ένα συστατικό έχει κάποιες διεπαφές. Κάποιο συστατικό παρέχει τις υπηρεσίες του μέσω κάποιων διεπαφών, ενώ ταυτόχρονα μπορεί να απαιτεί την υλοποίηση κάποιων άλλων διεπαφών από τον κώδικα του πελάτη.
- Ένα συστατικό έχει μία υλοποίηση. Ένα συστατικό υλοποιεί τις διεπαφές του. Η υλοποίηση είναι κρυφή και ενθυλακωμένη πίσω από τις διεπαφές.

Διεπαφή και υλοποίηση συστατικού

- Ένα συστατικό συμμορφώνεται με κάποιο στάνταρ, όπως το JavaBeans το Microsoft COM+ και το Enterprise JavaBeans.
- Ένα συστατικό έχει κάποια προδιαγραφή. Η προδιαγραφή ενός συστατικού είναι το σύνολο των διεπαφών του.
- Ένα συστατικό έχει κάποιες διεπαφές. Κάποιο συστατικό παρέχει τις υπηρεσίες του μέσω κάποιων διεπαφών, ενώ ταυτόχρονα μπορεί να απαιτεί την υλοποίηση κάποιων άλλων διεπαφών από τον κώδικα του πελάτη.
- Ένα συστατικό έχει μία υλοποίηση. Ένα συστατικό υλοποιεί τις διεπαφές του. Η υλοποίηση είναι κρυφή και ενθυλακωμένη πίσω από τις διεπαφές.

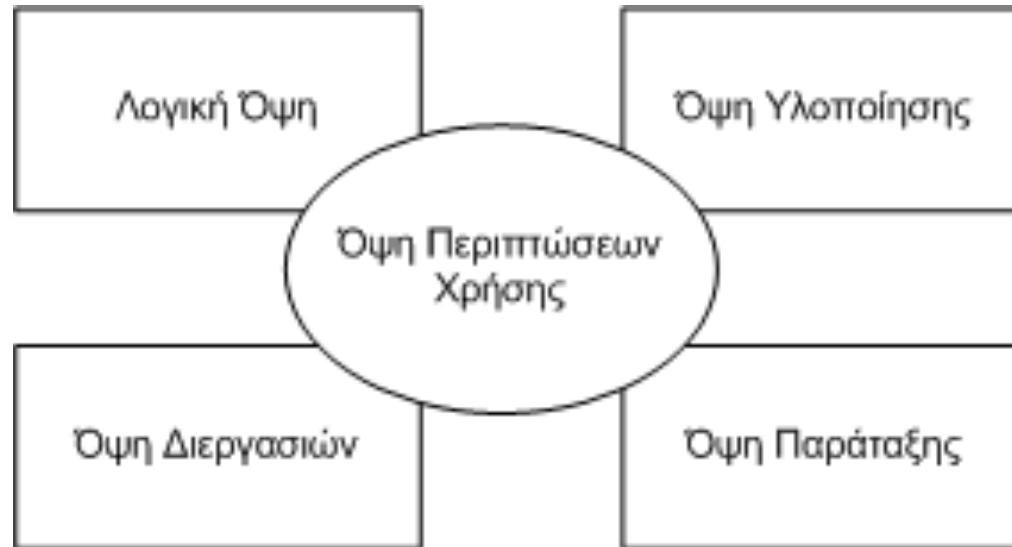
πλατφόρμες

- Οι Πλατφόρμες (frameworks) είναι ένα σύνολο από κλάσεις που εκφράζουν μία αφηρημένη σχεδίαση για ένα σύνολο συσχετιζόμενων προβλημάτων και παρέχουν την επαναχρησιμότητα μονάδων λογισμικού σε ένα υψηλότερο επίπεδο από αυτό των κλάσεων.
- Οι πλατφόρμες αναλαμβάνουν κάποιες από τις ευθύνες του λογισμικού και παρέχουν κλάσεις και σχεδίαση στους μηχανικούς λογισμικού που τα χρησιμοποιούν.
- Το σημαντικό στοιχείο των πλατφορμών είναι ότι δεν περιορίζονται στην παροχή κάποιων βοηθητικών κλάσεων στην ομάδα ανάπτυξης, αλλά προσφέρουν και μία σχεδιαστική λύση για κάποιο συγκεκριμένο πρόβλημα. Δεν έχουμε δηλαδή μόνο την επαναχρησιμοποίηση σε επίπεδο κλάσεων αλλά επαναχρησιμότητα και σε επίπεδο σχεδίασης.

τεκμηρίωση αρχιτεκτονικής

- Η καλή τεκμηρίωση της αρχιτεκτονικής του λογισμικού δεν μπορεί να αποδοθεί με έναν και μόνο τρόπο.
- Η κατανόηση της αρχιτεκτονικής βελτιώνεται, όταν αυτή τεκμηριώνεται με διαφορετικές όψεις.
- Ένας πρώτος διαχωρισμός των όψεων της αρχιτεκτονικής του λογισμικού είναι η λογική αρχιτεκτονική (logical architecture) και η φυσική αρχιτεκτονική (physical architecture). Η λογική αρχιτεκτονική αφορά στην περιγραφή των μονάδων λογισμικού και στις σχέσεις μεταξύ τους. Η φυσική αρχιτεκτονική αφορά στη διανομή των τμημάτων του λογισμικού.

Το μοντέλο των 4+1 όψεων



Όψεις

- Λογική όψη (logical view). Περιλαμβάνει τις σημαντικότερες κλάσεις της σχεδίασης και την οργάνωσή του λογισμικού σε πακέτα και υποσυστήματα.
- Όψη διεργασιών (process view). Περιλαμβάνει την περιγραφή των διεργασιών, την επικοινωνία τους και τη διανομή των κλάσεων σε διεργασίες. Η όψη αυτή είναι απαραίτητη σε λογισμικό με υψηλό βαθμό παράλληλης επεξεργασίας.
- Όψη υλοποίησης (implementation view). Περιλαμβάνει μία επισκόπηση της οργάνωσης του μοντέλου υλοποίησης (implementation model), τη δομή και οργάνωση του κώδικα και την απεικόνιση στοιχείων του λογικού μοντέλου, όπως κλάσεις, πακέτα και συστατικά, στα στοιχεία του κώδικα που τις υλοποιούν.

Όψεις

- Όψη παράταξης (deployment view). Περιγράφει τις τυπικές καταστάσεις του φυσικού περιβάλλοντος στο οποίο θα εκτελείται το λογισμικό και τους κόμβους επεξεργασίας που εκτελούνται οι διεργασίες του λογισμικού.
- Όψη περιπτώσεων χρήσης (use case view). Περιλαμβάνει τις σημαντικότερες περιπτώσεις χρήσης οι οποίες επηρεάζουν την αρχιτεκτονική.

επεκτάσεις του μοντέλου των 4 + 1 όψεων

- Η όψη δεδομένων (data view) που αφορά τη σχεδίαση της βάσης δεδομένων και τους μηχανισμούς απεικόνισης των αντικειμένων σε πίνακες της βάσης δεδομένων. Η όψη δεδομένων είναι απαραίτητη, όταν το λογισμικό είναι μέρος ενός πληροφοριακού συστήματος.
- Η όψη της ασφάλειας (security view) που αφορά τους μηχανισμούς ασφάλειας που ακολουθούνται.
- Εισάγοντας διαφορετικές όψεις στην τεκμηρίωση της αρχιτεκτονικής, το μοντέλο των 4+1 όψεων γενικεύεται σε μοντέλο N+1 όψεων, οι οποίες ορίζονται ανάλογα με τη βαρύτητα κάθε όψης στη διαμόρφωση της αρχιτεκτονικής.

έγγραφο αρχιτεκτονικής λογισμικού

- Όλες οι διαφορετικές όψεις της αρχιτεκτονικής συγκεντρώνονται σε ένα έγγραφο το οποίο τεκμηριώνει την αρχιτεκτονική.
- Το έγγραφο αυτό καλείται Έγγραφο Αρχιτεκτονικής Λογισμικού (Software Architecture Document) το οποίο περιγράφει τις σημαντικές αρχιτεκτονικές αποφάσεις.