ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάλυση και Κατηγοριοποίηση Αρχιτεκτονικών Τεχνοτροπιών Υπηρεσιοκεντρικών Συστημάτων Λογισμικού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σπύρος-Γεράσιμος Φ. Βάκκας

Επιβλέπων: Κώστας Κοντογιάννης

Καθηγητής Ε.Μ.Π

Αθήνα, Μάιος 2018

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάλυση και Κατηγοριοποίηση Αρχιτεκτονικών Τεχνοτροπιών Υπηρεσιοκεντρικών Συστημάτων Λογισμικού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σπύρος-Γεράσιμος Φ. Βάκκας

Επιβλέπων: Κώστας Κοντογιάννης

Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

.....	.....	.....
Κ. Κοντογιάννης	Α-Γ. Σταφυλοπάτης	Γ. Στάμου
Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Αν. Καθηγητής Ε.Μ.Π

Αθήνα, Μάιος 2018

.....
Σπύρος-Γεράσιμος Φιλολάου Βάκκας

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π

Περίληψη

Ο σκοπός αυτής της διπλωματικής εργασίας είναι να μεταδώσει μία γενικότερη αντίληψη και κατανόηση των αρχιτεκτονικών τεχνοτροπιών για υπηρεσιοκεντρικές εφαρμογές. Μέσω γενικής κατηγοριοποίησης των μοτίβων εκτελείται μία εισαγωγή στο να μπορεί κάποιος να κάνει τη σωστή επιλογή μοτίβου αναλόγως τις ανάγκες μιας εφαρμογής. Η εργασία στοχεύει να διαδώσει την γνώση τεκμηριωμένων και εφαρμοσμένων μοτίβων πάνω στον χώρο των υπολογιστικών συστημάτων λογισμικού που έχουν καθιερωθεί έως τώρα για την αντιμετώπιση πολλαπλών προβλημάτων και είναι αποδεκτά από όλο τον σχετικό χώρο αυτού του τομέα.

Η κύρια ανάλυση της εργασίας επικεντρώνεται στα μοτίβα που σχετίζονται σε αρχιτεκτονικές επιχειρησιακών εφαρμογών. Γίνεται ανάλυση και περιγραφή 51 μοτίβων πάνω σε αυτές τις εφαρμογές. Η προσέγγιση σε κάθε μοτίβο αναπτύσσεται ολοκληρωμένα μέσω διάφορων πηγών με στόχο την σφαιρική κατανόηση του καθενός ξεχωριστά. Η προσπάθεια μεταφοράς αυτών των πληροφοριών προς τον αναγνώστη, έχει σκοπό να του δώσει τα εφόδια ώστε να μπορεί να διαμορφώσει κατάλληλα μια επιχειρησιακή εφαρμογή έχοντας συλλέξει τις κατάλληλες γνώσεις. Δηλαδή μία εφαρμογή που υποστηρίζεται από τα κατάλληλα μοτίβα αναλόγως των απαιτήσεων κάθε φορά.

Λέξεις κλειδιά:

Οι χαρακτηριστικές λέξεις αυτής της εργασίας είναι τα μοτίβα, οι εφαρμογές, η επιχείρηση, οι πελάτες, η αρχιτεκτονική και ο σχεδιασμός. Εννοιολογικά, το θέμα τις εργασίες επικεντρώνεται σε αυτές τις λέξεις.

Summary

The purpose of this diploma thesis is to convey a broader understanding and understanding of architectural styles for service-oriented applications. Through a general classification of patterns success an introduction which is made to enable someone to make the right choice of pattern according to the needs of an application. The project aims to disseminate the knowledge of documented and applied patterns in the field of software computing systems that have been established so far to address multiple problems and are accepted by all the relevant area of this field.

The main analysis of the work focuses on the patterns related to business application architectures. There are analyzed and described 51 patterns on these applications. The approach to each pattern is developed in a comprehensive way through various sources, aiming at the global understanding of each individual. The attempt to transfer this information to the reader is intended to give him the necessary skills to be able to properly design a business application having gained the appropriate knowledge. That is an application supported by appropriate patterns depending on the requirements each time.

KEYWORDS:

Some important keywords in this work are : patters, application, clients, data and enterprise.

Ευχαριστίες

Σε αυτό το σημείο θα ήθελα να εκφράσω τις ευχαριστίες μου σε όσους με έχουν στηρίξει σε όλα αυτά τα χρόνια των σπουδών μου και εξαιρέτως στον κύριο Κοντογιάννη για την εμπιστοσύνη που μου έδειξε ώστε να μου ανατεθεί αυτή η εργασία και για την πολύτιμη στήριξη που μου προσέφερε ώστε να ολοκληρωθεί άρτια. Επίσης θα ήθελα να ευχαριστήσω τα αδέρφια μου και τους φίλους μου για την παρότρυνση και την καθολική ψυχολογική μου στήριξη και ιδιαιτέρως ευχαριστώ, για το δώρο της ζωής που μου χάρισαν, τη μητέρα μου και τον θανόντα πατέρα μου που δε πρόλαβα να του χαρίσω αυτή τη στιγμή.

Περιεχόμενα

1. Εισαγωγή

Η ζωή του ανθρώπου έχει κατακλυστεί από τεχνολογικά επιτεύγματα καθώς η τρέχουσα εποχή είναι ομόφωνα η εποχή της τεχνολογίας. Η καθημερινότητα απαιτεί υψηλή απόδοση σε αξιοπιστία και ταχύτητα των υπολογιστικών συστημάτων. Ένα μεγάλο μέρος της ρουτίνας ενός ανθρώπου είναι διαδικασίες οι οποίες πραγματώνονται με την υποστήριξη από κάποιο υπολογιστικό σύστημα τόσο κατά την υλοποίηση όσο και κατά τη λειτουργία τους. Τα συστήματα αυτά αποτελούνται από το κομμάτι των μηχανικών εξαρτημάτων και από το κομμάτι του λογισμικού. Για την βέλτιστη λειτουργία του λογισμικού των υπολογιστικών συστημάτων συναντάται ένα πεδίο στον τομέα του εφαρμοσμένου προγραμματισμού όπου είναι τα μοτίβα.

Ο κόσμος των αρχιτεκτονικών τεχνοτροπιών (μοτίβων) απαρτίζει μεγάλο και σημαντικό χώρο στον προγραμματισμό υπηρεσιοκεντρικών συστημάτων λογισμικού. Η εξέλιξη τους τα τελευταία 20 χρόνια είναι ραγδαία. Από απλός τρόπος αντιμετώπισης συγκεκριμένων προβλημάτων εξελίχτηκε σε ένα ολόκληρο ξεχωριστό τομέα εργασίας για πολλούς προγραμματιστές, όπου μέσω μοτίβων υποστηρίζει τα προγράμματα των εφαρμογών να αντιμετωπίζουν επαναλαμβανόμενα προβλήματα και να επιτελούν σημαίνοντα ρόλο για την απόδοση τους.

1.1 Σκοπός Διπλωματικής Εργασίας

Ο σκοπός αυτής της διπλωματικής εργασίας εκφράζεται ολοκληρωμένα στην περίληψη της εργασίας. Με ένα σύντομο τρόπο ο στόχος της είναι να μεταλαμπαδεύσει γνώσεις, τεχνικές και ιδέες ώστε να αντιμετωπίζονται με τον καλύτερο δυνατό τρόπο όσα προβλήματα συναντώνται σε επιχειρησιακές εφαρμογές μέσω μοτίβων.

1.2 Περιγραφή Προβλήματος

Το πρόβλημα γύρω από το οποίο κυμαίνεται αυτή η εργασία αφορά τις πολλαπλές ανάγκες που απαιτούν οι επιχειρησιακές εφαρμογές. Υπάρχουν πολλές ανάγκες και κάθε μία που λύνεται αναφέρεται και στην ανάλυση των μοτίβων. Μία από αυτές είναι η αρμονική ενσωμάτωση της επιχειρησιακής λογικής μέσα στο προγραμματιστικό μοντέλο. Άλλες ανάγκες είναι η διαχείριση πηγών δεδομένων, συμπεριφορών, διάφορων κατασκευασμένων μοντέλων και όλα αυτά στη σφαίρα της αντικειμενοστρέφειας. Σημαντική ανάγκη είναι η χαρτογράφηση δεδομένων και μεταδεδομένων. Επιπλέον, η άρτια λειτουργία διαδικτυακών απεικονίσεων, η διαχείριση και ασφάλεια των καταστάσεων διαλόγου (sessions), η διανομή των

δεδομένων και ο κατάλληλος συγχρονισμός στις ενέργειες πάνω στα δεδομένα έχουν σαν αποτέλεσμα την εξασφάλιση της σωστής και αποδοτικής λειτουργίας όλης της εφαρμογής. Όλες αυτές οι ανάγκες αποτελούν ένα σύνθετο πρόβλημα που αντιμετωπίζουν τα μοτίβα για την καλύτερη εξυπηρέτηση μίας επιχείρησης και των πελατών της.

1.3 Λίστα Ονομάτων Μοτίβων

Αλφαβητική λίστα μοτίβων που θα αναφερθούν στην εργασία:

- Acceptor-Connector design
- Active Object design
- Active Record
- Application Controller
- Association Table Mapping
- Asynchronous Completion Token
- Blackboard
- Broker
- Caching
- Class Table Inheritance
- Client Session State
- Client-Dispatcher-Server
- Coarse-Grained Lock
- Command Processor
- Component Configurator
- Concrete Table Inheritance
- Coordinator
- Counted Pointer idiom
- Data Mapper
- Data Transfer
- Database Session State
- Dependent Mapping
- Domain Model
- Double-Checked Locking Optimization
- Eager Acquisition
- Embedded Value
- Evictor
- Extension Interface
- Foreign Key Mapping
- Forwarder-Receiver
- Front Controller
- Gateway
- Half-Sync/Half-Async
- Identity Field

- Identity Map
- Implicit Lock
- Inheritance Mappers
- Interceptor
- Layer Supertype
- Layers
- Lazy Acquisition
- Lazy Load
- Leader/Followers
- Leasing
- Lookup
- Mapper
- Master-Slave
- Metadata Mapping
- Microkernel
- Model View Controller
- Model-View-Controller
- Money
- Monitor Object
- Optimistic Offline Lock
- Page Controller
- Partial Acquisition
- Pessimistic Offline Lock
- Pipes and Filters
- Plugin
- Pooling
- Presentation-Abstraction-Control
- Proactor architectural
- Proxy
- Query Object
- Reactor
- Record Set
- Reflection
- Registry
- Remote Façade
- Repository
- Resource Lifecycle Manager
- Row Data Gateway
- Scoped Locking
- Separated Interface
- Serialized LOB
- Server Session State
- Service Layer
- Service Stub
- Single Table Inheritance

- Special Case
- Strategized Locking
- Table Data Gateway
- Table Module
- Template View
- Thread-Safe Interface
- Thread-Specific Storage
- Transaction Script
- Transform View
- Two Step View
- Unit of Work
- Value Object
- View Handler
- Whole-Part
- Wrapper Façade

1.4 Δομή Διπλωματικής Εργασίας

Η δομή της διπλωματικής εργασίας διαμορφώνεται ως εξής:

Στα κεφάλαια 2 και 3 γίνεται αναφορά και περιγραφή μοτίβων. Συγκεκριμένα στο κεφάλαιο 2 αναφέρονται 4 τόμοι βιβλίων που εξυπηρετούν να διαχωρίζουμε θεματικά όλο το εύρος των μοτίβων. Δίνεται μια περιληπτική περιγραφή στα μοτίβα κάθε τόμου και αναφέρονται ονομαστικά. Στο κεφάλαιο 3, που είναι και το κυρίως θέμα της εργασίας, αναπτύσσονται τα μοτίβα που έχουν να κάνουν με την αρχιτεκτονική επιχειρησιακών εφαρμογών και αναλύονται όλα τα μοτίβα με συγκεκριμένο τρόπο. Για κάθε μοτίβο δίδεται ο σκοπός του, η δομή του, το πως λειτουργεί, τα πλεονεκτήματα και τα μειονεκτήματα του, οι εφαρμογές του και μία σχετική βιβλιογραφία γι' αυτό. Στο κεφάλαιο 4 γίνεται μία πολύ συνοπτική περίληψη του τι αναπτύχθηκε εντός της εργασίας και παραθέτονται αμέσως μετά δύο υποκεφάλαια για τα επιτεύγματα των μοτίβων μέχρι τώρα και για τις μελλοντικές προσδοκίες αυτών. Τέλος στο κεφάλαιο 5 αναφέρεται η βιβλιογραφία όλης της διπλωματικής εργασίας.

2. Αρχιτεκτονική λογισμικού με βάση τα μοτίβα

Σε αυτό το κεφάλαιο θα γίνει αναφορά σε 4 θεματικές κατηγορίες μοτίβων βασισμένη σε 4 τόμους βιβλίων. Σε κάθε είδος θα γίνει περιληπτική περιγραφή της λειτουργίας αυτών των μοτίβων και θα αναφέρουμε ονομαστικά κάποια εξ' αυτών πάνω στο είδος. Τα είδη είναι τα εξής:

- A system of patterns (ένα σύστημα μοτίβων)
- Patterns for concurrent and networked objects (μοτίβα για συμπίπτων και δικτυωμένα αντικείμενα)
- Patterns for resource management (μοτίβα για διαχείριση πόρων)
- Patterns for enterprise application architecture (μοτίβα για αρχιτεκτονική επιχειρησιακών εφαρμογών)

2.1 A System of Patterns

Ο πρώτος τόμος ασχολείται με τα απλά μοτίβα διαχωρίζοντας τα σε τρία κομμάτια: το αρχιτεκτονικό κομμάτι, το κομμάτι της σχεδίασης και τους ιδιωματισμούς. Τα αρχιτεκτονικά μοτίβα θεωρούνται ως το υψηλότερο επίπεδο μοτίβων σε σχέση με τις άλλες δύο κατηγορίες. Ουσιαστικά, είναι υψηλού επιπέδου στρατηγικές που αφορούν κατασκευαστικά στοιχεία μεγάλης κλίμακας, καθολικές ιδιότητες και μηχανισμούς ενός συστήματος. Αυτά τα μοτίβα είναι υπεύθυνα να παρέχουν την κύρια δομή της συνολικής αρχιτεκτονικής ενός συστήματος. Μετέπειτα, τα σχεδιαστικά μοτίβα σχετίζονται με μεσαίου επιπέδου κώδικα και παρέχουν προγράμματα βελτίωσης και κατασκευής μικρότερων υποσυστημάτων. Έχουν ως ευθύνη να αντιμετωπίσουν προβλήματα που μπορεί να προκύψουν από μία ήδη καθορισμένη συνολική δομή ενός συστήματος, όπως η πολυπλοκότητα και η κατανομή εργασιών. Συνήθως επηρεάζονται από τη γλώσσα προγραμματισμού. Τέλος, τα μοτίβα που ασχολούνται με τους ιδιωματισμούς είναι τεχνικές προγραμματισμού ειδικά για συγκεκριμένες γλώσσες ,οι οποίες συμπληρώνουν λεπτομέρειες χαμηλού επιπέδου. Σε γενικές γραμμές περιγράφουν πως υλοποιούνται συγκεκριμένες πτυχές των στοιχείων ή τις σχέσεις μεταξύ τους μέσω συγκεκριμένων χαρακτηριστικών.

1. Αρχιτεκτονικά μοτίβα:
    - From Mud to Structure (Από λάσπη σε δομή):
        - Layers pattern
        - Pipes and Filters pattern
        - Blackboard pattern
    - Distributed Systems (Κατανεμημένα συστήματα):
        - Broker pattern
    - Interactive Systems (Διαδραστικά συστήματα):
        - Model-View-Controller pattern
        - Presentation-Abstraction-Control pattern
    - Adaptable Systems (Προσαρμόσιμα συστήματα):
        - Reflection pattern
        - Microkernel pattern
2. Σχεδιαστικά μοτίβα:

- Structural Decomposition (Διαρθρωτική αποσύνθεση):
    - Whole-Part pattern
- Organization of Work (Οργάνωση της εργασίας):
    - Master-Slave pattern
- Access Control (Έλεγχος πρόσβασης):
    - Proxy pattern
- Management (Διαχείριση):
    - Command Processor pattern
    - View Handler pattern
- Communication (Επικοινωνία):
    - Forwarder-Receiver pattern
    - Client-Dispatcher-Server pattern

3. Ιδιωματικά μοτίβα:
    - Counted Pointer idiom pattern

Βιβλιογραφία: Pattern-Oriented Software Architecture: A System of Patterns, Volume 1 by Wiley.

2.2 Patterns for concurrent and networked

Ο δεύτερος τόμος επικεντρώνεται σε μοτίβα που αντιμετωπίζουν τα ζητήματα της ταυτότητας και της δικτύωσης. Γίνεται η περιγραφή των μοτίβων όπου ασχολούνται με κύρια στοιχεία της δομής αναπτυγμένα την ίδια στιγμή και με δικτυωμένα συστήματα. Κάποιες από τις ενέργειες είναι η πρόσβαση και διαμόρφωση υπηρεσιών, ο χειρισμός συμβάντων, ο συγχρονισμός και η ταυτόχρονη λειτουργία.

- Service Access and Configuration Patterns (Μοτίβα για υπηρεσίες πρόσβασης και διαμόρφωσης):
    - Wrapper Façade design pattern
    - Component Configurator design pattern
    - Interceptor architectural pattern
    - Extension Interface design patter
- Event Handling Pattern (Μοτίβα για χειρισμό συμβάντων):
    - Reactor architectural pattern
    - Proactor architectural pattern
    - Asynchronous Completion Token design pattern
    - Acceptor-Connector design pattern
- Synchronization Patterns (Μοτίβα συγχρονισμού):
    - Scoped Locking pattern
    - Strategized Locking design pattern
    - Thread-Safe Interface design pattern

    - Double-Checked Locking Optimization design pattern
- Concurrency Patterns (Μοτίβα ταυτοχρονισμού):
    - Active Object design pattern
    - Monitor Object design pattern
    - Half-Sync/Half-Async architectural pattern
    - Leader/Followers architectural pattern
    - Thread-Specific Storage design pattern

Βιβλιογραφία: Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects, Volume 2 by Wiley.

2.3 Patterns for resource management

Ο τρίτος τόμος ασχολείται με τεχνικές για αποτελεσματική υλοποίηση διαχείρισης πόρων σε ένα σύστημα μέσω κυρίως σχεδιαστικών μοτίβων. Ο συγκεκριμένος τόμος κάνει μία πολύ λεπτομερή εισαγωγή στη διαχείριση των πόρων και δύο περιπτωσιολογικές μελέτες όπου τα μοτίβα εφαρμόζονται στα πεδία των ad hoc δικτύων (είναι ένας αποκεντρωμένος τύπος ασύρματου δικτύου ή αλλιώς αυτοοργανωμένο δίκτυο) και των κινητών ραδιοφωνικών δικτύων. Τέλος, αναφέρονται τα μοτίβα που περιγράφονται μέσα στον τόμο:

- Resource Acquisition (Απόκτηση πόρων):
    - Lookup pattern
    - Lazy Acquisition pattern
    - Eager Acquisition pattern
    - Partial Acquisition pattern
- Resource Lifecycle (Κύκλος ζωής πόρων):
    - Caching pattern
    - Pooling pattern
    - Coordinator pattern
    - Resource Lifecycle Manager
- Resource Release (Απελευθέρωση πόρων):
    - Leasing pattern
    - Evictor pattern

Βιβλιογραφία: Pattern-Oriented Software Architecture: Patterns for Resource Management, Volume 3 by Wiley.

2.4 Patterns for enterprise application architecture

Στον τέταρτο τόμο αναλύονται τα μοτίβα που υποστηρίζουν την καλύτερη λειτουργία της αρχιτεκτονικής των επιχειρησιακών εφαρμογών, όπου αφορά και το κυρίως θέμα της διπλωματικής. Στο επόμενο κεφάλαιο παραθέτεται η ανάλυση όλων των μοτίβων που θα αναφερθούν εδώ.

Τα μοτίβα των εφαρμογών των επιχειρήσεων που θα αναπτύξουμε είναι τα εξής:

- Domain Logic Patterns (Μοτίβα λογικής domain)
    - Transaction Script pattern
    - Domain Model pattern
    - Table Module pattern
    - Service Layer pattern
- Data Source Architectural Patterns (Αρχιτεκτονικά μοτίβα για πηγές δεδομένων)
    - Table Data Gateway pattern
    - Row Data Gateway pattern
    - Active Record pattern
    - Data Mapper pattern
- Object-Relational Behavioral Patterns (Μοτίβα αντικειμενοσχεσιακής συμπεριφοράς)
    - Unit of Work pattern
    - Identity Map pattern
    - Lazy Load pattern
- Object-Relational Structural Patterns (Αντικειμενοσχεσιακά κατασκευαστικά μοτίβα)
    - Identity Field pattern
    - Foreign Key Mapping pattern
    - Association Table Mapping pattern
    - Dependent Mapping pattern
    - Embedded Value pattern
    - Serialized LOB pattern
    - Single Table Inheritance pattern
    - Class Table Inheritance pattern
    - Concrete Table Inheritance pattern
    - Inheritance Mappers pattern
- Object-Relational Metadata Mapping Patterns (Μοτίβα αντικειμενοσχεσιακών μεταδεδομένων χαρτογράφησης)
    - Metadata Mapping pattern
    - Query Object pattern
    - Repository pattern

- Web Presentation Patterns (Μοτίβα διαδικτυακών παρουσιάσεων)
    - Model View Controller pattern
    - Page Controller pattern
    - Front Controller pattern
    - Template View pattern
    - Transform View pattern
    - Two Step View pattern
    - Application Controller pattern
- Distribution Patterns (Μοτίβα διανομής)
    - Remote Façade pattern
    - Data Transfer pattern
- Offline Concurrency Patterns (Μοτίβα συγχρονικότητας εκτός σύνδεσης)
    - Optimistic Offline Lock pattern
    - Pessimistic Offline Lock pattern
    - Coarse-Grained Lock pattern
    - Implicit Lock pattern
- Session State Patterns (Μοτίβα κατάστασης διαλόγου)
    - Client Session State pattern
    - Server Session State pattern
    - Database Session State pattern
- Base Patterns (Βασικά μοτίβα)
    - Gateway pattern
    - Mapper pattern
    - Layer Supertype pattern
    - Separated Interface pattern
    - Registry pattern
    - Value Object pattern
    - Money pattern
    - Special Case pattern
    - Plugin pattern
    - Service Stub pattern
    - Record Set pattern

Βιβλιογραφία: Patterns of Enterprise Application Architecture by Martin Fowler.

3. Μοτίβα για αρχιτεκτονική επιχειρησιακών εφαρμογών

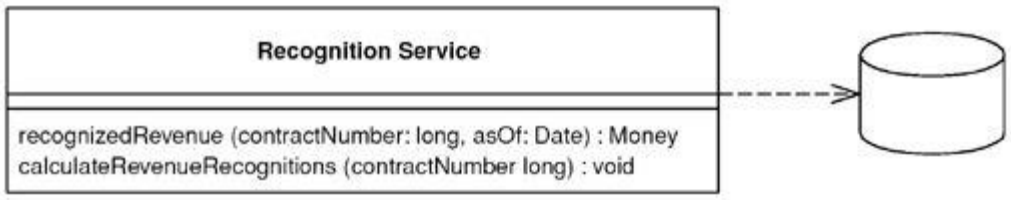
Σε αυτό το κεφάλαιο θα γίνει η ανάλυση των μοτίβων που αναφέραμε στο παραπάνω κεφάλαιο 2.4. Αυτά τα μοτίβα έχουν σκοπό τη διευκόλυνση των επιχειρησιακών εφαρμογών για να εξυπηρετούν ανάγκες βασισμένες στις

προδιαγραφές της λογικής που θέλει να ακολουθήσει μία επιχείρηση. Είναι ένα σύνθετο προγραμματιστικό κομμάτι, διότι τα μοτίβα δεν ασχολούνται με μία απλή εφαρμογή. Το κομμάτι αυτό ασχολείται με τα εξής: πελάτες, συναλλαγές, επιχειρήσεις όπου θέλουν να ακολουθήσουν μία συγκεκριμένη στρατηγική, με σημαντικά προσωπικά δεδομένα πελατών, με ασφάλεια δεδομένων και άλλα πεδία, τα οποία αναλύονται παρακάτω.

3.1 Domain Logic Patterns

3.1.1 Transaction Script

Το transaction script οργανώνει την επιχειρησιακή λογική με διαδικασίες, όπου κάθε διαδικασία χειρίζεται ένα μόνο αίτημα από την παρουσίαση.



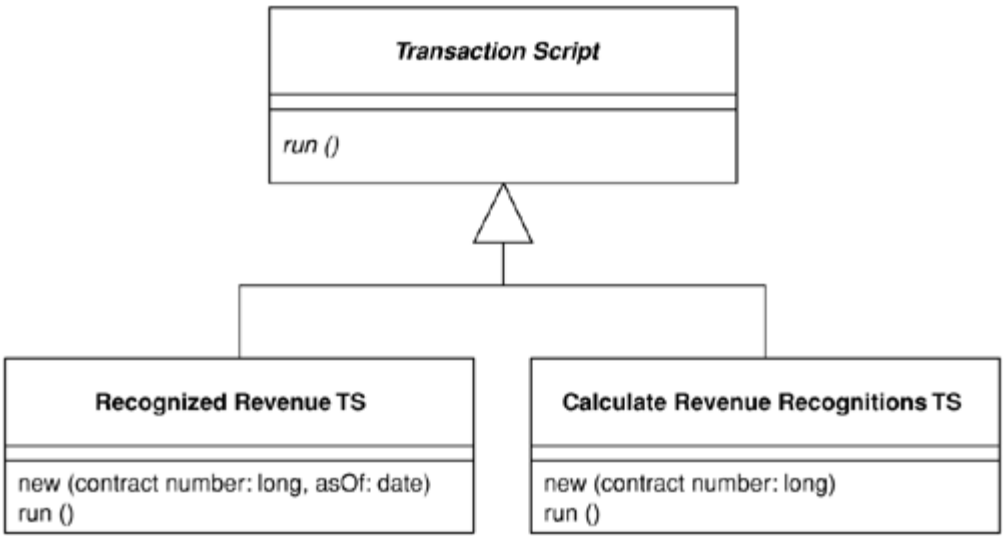
Σκοπός:

Οι επιχειρησιακές εφαρμογές χρειάζονται να κάνουν κάποιες ενέργειες για το χειρισμό μιας ακολουθίας συναλλαγών. Ο σκοπός του transaction script είναι να οργανώνει σε μία ενιαία διαδικασία τις απλές ενέργειες όπως απλή εμφάνιση πληροφοριών στη βάση δεδομένων ή επικυρώσεις ή υπολογισμούς κλπ.

Δομή:

Η δομή του transaction script ουσιαστικά διαμορφώνεται από την περίπτωση κάθε φορά και δομεί μια συλλογή σεναρίων που εκτελούνται σε ένα διακομιστή. Κάθε σενάριο χειρίζεται μία ενιαία διαδικασία που αντιστοιχεί σε ένα συγκεκριμένο αίτημα που υποβάλλεται από χρήστη ή άλλο σύστημα.

Στο παρακάτω σχήμα φαίνεται η χρήση εντολών για ένα σενάριο συναλλαγής:



Πως λειτουργεί:

Για τη λειτουργία του συνήθως δημιουργείται μία ενιαία στατική κλάση που αντιστοιχεί στην εκάστοτε επιχειρηματική συναλλαγή και όλη η ροή εργασίας (επιχειρηματικοί κανόνες, επικυρώσεις, υπολογισμοί κλπ) που απαιτούνται για την ολοκλήρωση της περιλαμβάνονται μέσα σε αυτή.

Παράδειγμα:

Έστω μια εφαρμογή που παρέχει λειτουργίες όπου γίνονται παραγγελίες, δημιουργούνται τιμολόγια και υπολογίζονται μπόνους για πελάτες. Αυτές οι λειτουργίες μπορούν να αντιμετωπιστούν ως δύο συναλλαγές, παραγωγή τιμολογίου και υπολογισμός μπόνους. Μπορεί να δημιουργηθεί μία κλάση όπου περιέχει δύο μεθόδους με τις δύο παραπάνω λειτουργίες.

```
class InvoiceGenerator

{

    public void GenerateInvoices()

    {

        load orders to deliver

        iterate through orders

        generate invoice

        mark them ready to deliver

    }

}
```

Για παραπάνω λειτουργίες μπορούν να χρησιμοποιηθούν και παραπάνω κλάσεις.

Πλεονεκτήματα και μειονεκτήματα:

Το βασικότερο πλεονέκτημα αυτού του μοτίβου είναι ότι είναι πολύ εύκολο στην εφαρμογή του για τους προγραμματιστές, λόγω απλότητας. Οπότε σαν αποτέλεσμα έχει την γρήγορη δημιουργία ενός τέτοιου μοτίβου όπως και την εύκολη κατανόηση του. Επίσης είναι πολύ αποτελεσματικό και βολικό για εφαρμογές με απλή επιχειρησιακή λογική όπου εκτελεί απλές ενέργειες.

Το βασικό μειονέκτημα του transaction script είναι ότι δεν υποστηρίζει καλά εφαρμογές με υψηλή πολυπλοκότητα. Όσο πιο περίπλοκη είναι η επιχειρησιακή λογική τόσο πιο περίπλοκη γίνεται και η εφαρμογή του μοτίβου ώστε να διατηρηθεί το ίδιο καλά η σχεσιακή κατάσταση. Υπάρχει ακόμα και η περίπτωση ο κώδικας να αναπαραχθεί καθώς μπορεί να υπάρξει επαναληψιμότητα μεταξύ συναλλαγών.

Εφαρμογές:

Όπως είπαμε παραπάνω χρησιμοποιείται κυρίως σε εφαρμογές που δεν έχουν υψηλή επιχειρησιακή λογική. Μία τέτοια είναι το Estonian ID όπου είναι card based door lock control δηλαδή για τον έλεγχο κλειδώματος θυρών, το οποίο χρησιμοποιήθηκε κυρίως σε υπηρεσίες και εφαρμογές των Windows όπου επιτρέπουν τη διαχείριση χρηστών, παρέχουν άδειες για το άνοιγμα θυρών και αναφέρει ποιός και πότε έκανε το άνοιγμα. Ακόμη μία εφαρμογή που το χρησιμοποιεί είναι το Jira importer όπου χρησιμοποιεί μικρές υπηρεσίες που εισάγουν αρχεία καταγραφής εργασίας και καθήκοντα από την Jira στην κύρια βάση δεδομένων της εταιρείας, όπου συναντώνται εργασίες προγραμματισμού, πωλήσεων και χρέωσης.

Βιβλιογραφία:

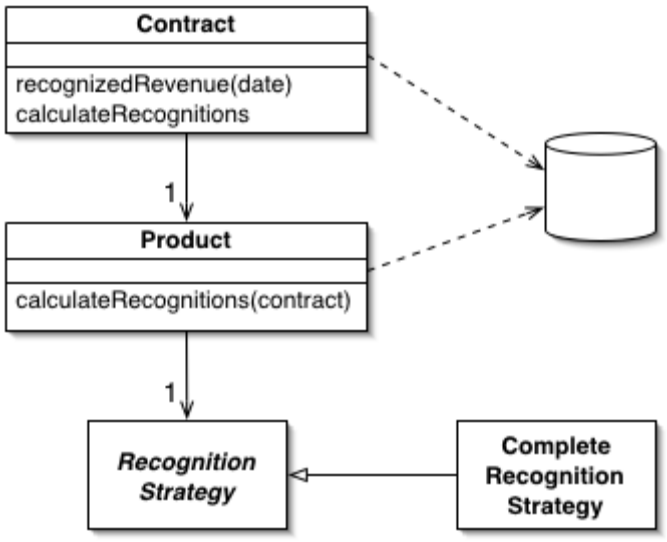
Article **The Software as a Service Delivery Model Explained** posted by **Omri Erel** in **SAAS,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler,**

Book **Architecting Applications for the Enterprise** by **Dino Esposito and Andrea Saltarello.**

3.1.2 Domain Model

Το domain model ενσωματώνει και τη συμπεριφορά και τα δεδομένα.



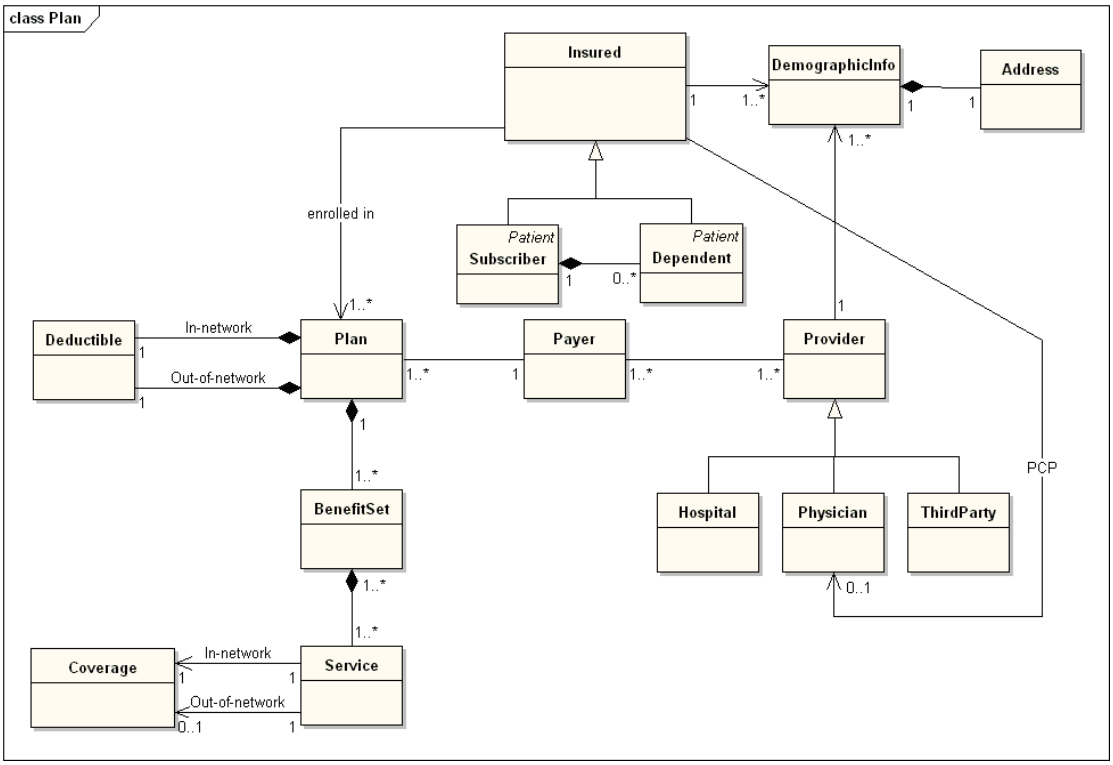
Σκοπός:

Στη χείριστη περίπτωση η επιχειρησιακή λογική μπορεί να είναι πολύ περίπλοκη. Οι κανόνες και η λογική μπορεί να περιγράφουν πολλές διαφορετικές περιπτώσεις, συμπεριφορές και σχέσεις μεταξύ των αντικειμένων όπου καθορίζουν και τη πολυπλοκότητα που σχεδιάστηκαν τα αντικείμενα. Ο σκοπός του domain model είναι να δημιουργεί έναν ιστό αλληλοσυνδεόμενων αντικειμένων, όπου κάθε αντικείμενο αντιπροσωπεύει κάποια σημαντική οντότητα, είτε τόσο μεγάλη όσο μία εταιρεία είτε μικρή όσο μία απλή συναλλαγή ή παραγγελία.

Δομή:

Η δομή του έχει ως στόχο να περιγράψει αντικείμενα και σχέσεις ενός πραγματικού κόσμου και να μπορεί να επικρατήσει η επιθυμητή επιχειρησιακή λογική της εφαρμογής μέσα σε αυτά. Ουσιαστικά αυτό το μοτίβο παρέχει έναν αντικειμενοστραφή τρόπο αντιμετώπισης περίπλοκων λογικών. Γι’ αυτό το domain model είναι χτισμένο από μικρά, χαλαρά αντικείμενα με κομψές διεπαφές, που παραμένουν εντελώς απαλλαγμένες από εξωτερικές εξαρτήσεις. Το Data mapper(3.2.4) μπορεί να βοηθήσει στην χαρτογράφηση των υπαρχόντων δεδομένων σε αυτό το μοτίβο (θα γίνει περιγραφή παρακάτω για το μοτίβο data mapper(3.2.4)).

Σχήμα: Δομή ενός παραδείγματος με domain model.



Πως λειτουργεί:

Χρησιμοποιώντας το domain model σε μία εφαρμογή σημαίνει ότι θα γίνει εισαγωγή ενός ολόκληρου στρώματος αντικειμένων που μορφοποιούν την επιχειρησιακή περιοχή όπου εργάζεται η ίδια η εφαρμογή. Κάποια αντικείμενα θα αντιπροσωπεύουν κάποια δεδομένα της επιχείρησης και άλλα κάποιες οντότητες, κανόνες, διαδικασίες και σχέσεις που υπάρχουν μεταξύ τους. Σχεδιάζονται κυρίως με τέτοιο τρόπο ώστε οι διαδικασίες και οι οντότητες που συνεργάζονται να βρίσκονται κοντά μεταξύ τους για πιο άμεση και πιο αποτελεσματική λειτουργία.

Το domain model μοιάζει με βάση δεδομένων αλλά προφανώς έχει πολλά περισσότερα χαρακτηριστικά, διαδικασίες, δεδομένα και κατά βάση δημιουργεί ένα πολύπλοκο δίκτυο ενώσεων και χρησιμοποιεί κληρονομικότητα. Λόγω αυτού προκύπτουν δύο είδη, το απλό και το σύνθετο domain model. Το απλό είναι αυτό που μοιάζει κυρίως με το σχεδιασμό μιας βάσης δεδομένων ενώ το σύνθετο έχει πολύπλοκο σχεδιασμό λόγω κληρονομικότητας, στρατηγικών και σύνθετων δικτύων μικρών διασυνδεδεμένων αντικειμένων. Προφανώς η χρήση του καθενός επιλέγεται αναλόγως των απαιτήσεων της επιχειρησιακής λογικής, δηλαδή σε κάτι σύνθετο και περίπλοκο θα επιλεγεί το σύνθετο μοντέλο.

Επιπλέον, στη λειτουργία του μοτίβου αυτού αξίζει να προστεθεί ότι, λόγω πολλών ενημερώσεων και αλλαγών σε συμπεριφορές των επιχειρήσεων, θα πρέπει να είναι εύκολο να γίνονται τροποποιήσεις και ενημερώσεις. Αυτό επιτυγχάνεται με το να έχει όσο λιγότερες εξαρτήσεις γίνεται από άλλα τμήματα του συστήματος.

Μία απλή λειτουργία θα περιλαμβάνει την άντληση όλων των αντικείμενων από ένα γραφικό αντικείμενο που εμπλέκονται σε αυτό. Αυτό δε σημαίνει πως θα είναι όλα τα αντικείμενα και όλες οι κλάσεις. Έτσι, στην εξέταση ενός συνόλου συμβάσεων μπορούν να τραβηχτούν μόνο τα αντικείμενα που εμπλέκονται στις συμβάσεις εντός του συνόλου και να εκτελεστούν οι εκάστοτε εντολές. Βελτίωση της λειτουργίας επίσης, μπορεί να γίνει με τη συνεργασία μιας αντικειμενοστραφούς βάσης δεδομένων όπου διευκολύνει το χειρισμό της μνήμης σε οποιαδήποτε άντληση δεδομένων ή ακόμα και με μία κατάλληλη χαρτογράφηση των αντικειμένων.

Πλεονεκτήματα και μειονεκτήματα:

Ένα σπουδαίο πλεονέκτημα αυτού του μοτίβου είναι ότι είναι σε θέση να διαχειριστεί τις πιο σύνθετες και απαιτητικές επιχειρησιακές λογικές. Αυτό δίνει το προβάδισμα στην αντιμετώπιση όλων των πολύπλοκων περιπτώσεων σε σχέση με άλλα μοτίβα. Επίσης η μικρή εξάρτηση του από άλλα τμήματα του συστήματος το κάνει εύχρηστο στις τροποποιήσεις και τις ενημερώσεις στις εκάστοτε αλλαγές και νέες απαιτήσεις της επιχειρησιακής εφαρμογής.

Βασικότερο πλεονέκτημα του, είναι ότι χρειάζεται προγραμματιστική εμπειρία. Για έναν αρχάριο προγραμματιστή ίσως είναι μία δύσκολη κατάσταση που θα πρέπει να αντιμετωπίσει. Όπως και ότι μπορεί να μετατραπεί σε μία χρονοβόρα διαδικασία για να βρεθεί η λύση σε ένα πιθανό σύνθετο πρόβλημα. Ένα ακόμα πιθανό πρόβλημα μπορεί να είναι η περιττή επανάληψη ή η υπερφόρτωση μιας κλάσης. Δηλαδή εάν μια συμπεριφορά χρειάζεται για μία μόνο συγκεκριμένη πιθανή περίπτωση είναι προτιμότερο να συμπεριλαμβάνεται στο συγκεκριμένο σενάριο συναλλαγής και όχι σε μία πιο γενική κλάση όπου μπορεί να υπερφορτωθεί από τέτοιες πιθανές συμπεριφορές.

Εφαρμογές:

Όπως είπαμε και παραπάνω οι κατάλληλες εφαρμογές που μπορεί να χρησιμοποιηθεί ένα domain model είναι αυτές που απαιτούν πολύπλοκη επιχειρησιακή λογική. Κάποιες τυπικές περιπτώσεις χρήσης του domain model είναι οι εξής:

- Όταν ο πελάτης καλεί μία κλάση απεικόνισης στέλνοντας δεδομένα ως έγγραφο XML και έτσι η κλάση ξεκινά μία νέα συναλλαγή.
- Για επικυρώσεις σε εισερχόμενα δεδομένα, όπου αυτές περιλαμβάνουν τις βασικές και επιχειρησιακές επικυρώσεις.
- Τη μορφοποίηση και τροποποίηση των δεδομένων για να είναι περισσότερο φιλικά.

- Για οποιαδήποτε κατηγοριοποίηση των χαρακτηριστικών ή κάποιων ιδιοτήτων ή κάποιων σχέσεων αντικειμένων και οντοτήτων.

Βιβλιογραφία:

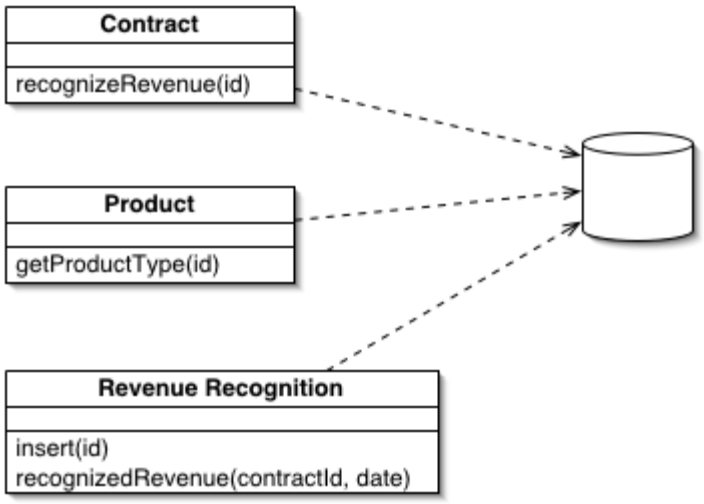
Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**,

Book **Domain-Driven Design: Definitions and Pattern Summaries** by **Eric Evans,**

Book **Description Logics in Multimedia Reasoning** by **Leslie F. Sikos.**

3.1.3 Table Module

Μία μεμονωμένη οντότητα που χειρίζεται την επιχειρηματική λογική για όλες τις γραμμές ενός πίνακα βάσης δεδομένων ή μίας απεικόνισης (view).



Σκοπός:

Ο σκοπός του table module είναι να οργανώσει όλη την επιχειρησιακή λογική με μία μόνο κλάση. Μία μόνο κλάση ενσωματώνει όλη τη λογική του domain για όλες τις εγγραφές που είναι αποθηκευμένες σε έναν πίνακα ή σε μία απεικόνιση. Οπότε ο κύριος σκοπός του μοτίβου είναι να έχει ένα αντικείμενο για να χειριστεί όλες τις ενέργειες της επιχειρησιακής εφαρμογής.

Δομή:

Η δομή του παρουσιάζει μια προσέγγιση βασισμένη στη βάση δεδομένων. Όλη η επιχειρησιακή λογική είναι οργανωμένη γύρω από πίνακες της βάσης δεδομένων. Η εφαρμογή αυτού του μοτίβου είναι σχετικά απλή, παρέχει μία διεπαφή για την εκτέλεση εργασιών σε όλα τα δεδομένα. Με απλά λόγια είναι μία κλάση που υποστηρίζει τις ενέργειες πινάκων.

Πως λειτουργεί:

Στον τρόπο λειτουργίας του υπάρχει μια βασική διαφορά από ένα απλό αντικείμενο και είναι ότι δεν έχει την έννοια της ταυτότητας με τα αντικείμενα που δουλεύει. Αυτό σημαίνει ότι όταν θα υπάρχει ανάγκη να γίνει εργασία με μία συγκεκριμένη σειρά, μία μέθοδος θα αναμένει κάποιο είδος αναφοράς ταυτότητας που θα περάσει ως μία από τις παραμέτρους. Η ισχύς του table module είναι ότι επιτρέπει να συσκευαστούν μαζί τα δεδομένα και η συμπεριφορά και την ίδια στιγμή να γίνεται χρήση των δυνάμεων μίας σχεσιακής βάσης δεδομένων. Το μοτίβο αυτό δίνει μια ρητή μέθοδο γύρω από τη διεπαφή όπου δρα στα δεδομένα που είναι σε ένα πίνακα (συνήθως από κάποια κλήση SQL). Η ομαδοποίηση της συμπεριφοράς σε πίνακα λειτουργεί ως όφελος για την ενθυλάκωση της μέσα σε πίνακα αντικειμένων, με τη συμπεριφορά να είναι κοντά στα δεδομένα όπου χρησιμοποιούνται. Όταν χρειαστούν πολλαπλές συμπεριφορές στα δεδομένα τότε μπορεί να χρησιμοποιηθούν πολλά table module για τις ίδιες εγγραφές.

Πλεονεκτήματα και μειονεκτήματα:

Βασικό πλεονέκτημα του είναι ότι μπορεί να χειρίζεται εφαρμογές με πινακοποιημένα δεδομένα και αυτό έχει ως αποτέλεσμα να μπορεί να κάνει ενέργειες σε μία εφαρμογή με πολύ μεγάλο όγκο δεδομένων όπου άλλα μοτίβα δε θα μπορούσαν. Επίσης πολύ σημαντικό είναι ότι είναι φιλικό προς τον προγραμματιστή όσων αφορά την απλότητα του και την ευκολία στην κατανόηση για την δημιουργία ενός τέτοιου μοτίβου.

Το βασικότερο και ίσως μεγάλο μειονέκτημα του είναι ότι απευθύνεται κυρίως σε εφαρμογές που χρησιμοποιούν πίνακες. Σε άλλες περιπτώσεις δηλαδή γίνονται προτιμότερα τα άλλα μοτίβα. Είτε σε περίπτωση που κάποια εφαρμογή έχει υψηλής πολυπλοκότητας επιχειρησιακή λογική είτε ακόμα και σε αρκετά χαμηλής εάν δεν είναι πινακοποιημένα τα αντικείμενα δεν είναι εύχρηστο μοτίβο.

Εφαρμογές:

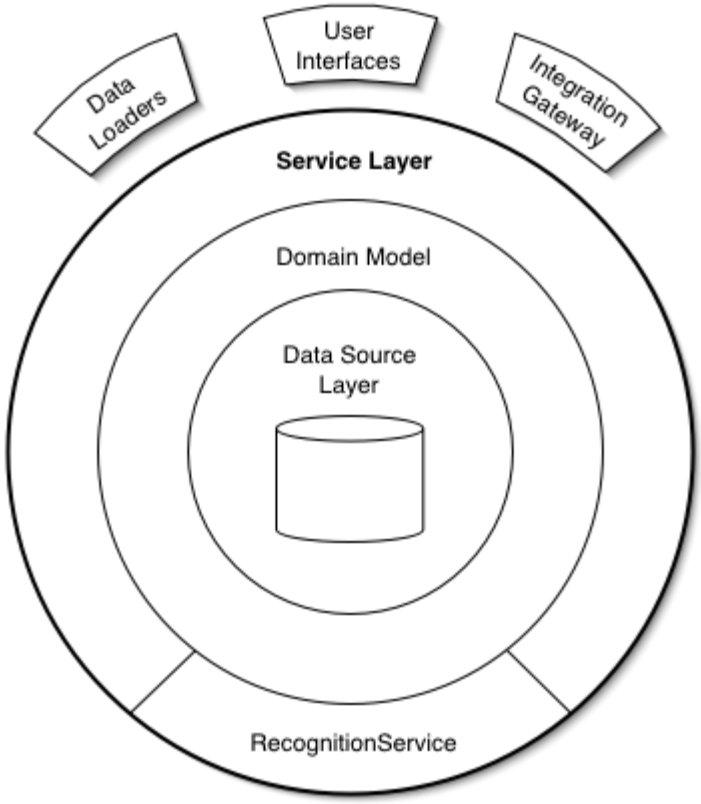
Το table module είναι ιδανικό για την εφαρμογή του σε επιχειρησιακές λογικές μεσαίου επιπέδου, δηλαδή μεταξύ transaction script (3.1.1) και domain model(3.1.2). Επίσης είναι ιδανικό μοτίβο για μία εφαρμογή που κάνει βαριά χρήση πινακοποιημένων δεδομένων τα οποία είναι αποτέλεσμα μιας κλήσης SQL. Το πιο σύνηθες που συμβαίνει είναι αυτό το μοτίβο να συνεργάζεται αρμονικά με το Record Set(3.10.11).

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**

3.1.4 Service Layer

Ορίζει ως όριο μιας εφαρμογής ένα στρώμα υπηρεσιών που θεσπίζει ένα σύνολο διαθέσιμων λειτουργιών και συντονίζει την απόκριση της εφαρμογής σε κάθε λειτουργία.



Σκοπός:

Το service layer είναι ένα μοτίβο σχεδιασμού με σκοπό την οργάνωση των υπηρεσιών μέσα σε ένα απόθεμα υπηρεσιών, εντοιχισμένο σε ένα σύνολο λογικών επιπέδων. Δηλαδή είναι υπηρεσίες που κατηγοριοποιούνται σε μία συγκεκριμένη λειτουργικότητα οργανωμένα σε επίπεδα κοινής χρήσης. Αυτό στοχεύει στη μείωση των εννοιολογικών δαπανών που σχετίζονται με τη διαχείριση του αποθέματος υπηρεσιών, καθώς οι υπηρεσίες που ανήκουν στο ίδιο στρώμα απευθύνονται σε μικρότερο σύνολο δραστηριοτήτων.

Η ομαδοποίηση των υπηρεσιών σε λειτουργικά στρώματα μειώνει τον αντίκτυπο της αλλαγής. Δηλαδή οι εκάστοτε αλλαγές επηρεάζουν μόνο το στρώμα στο οποίο δημιουργούνται, έχοντας μικρή επιρροή σε άλλα επίπεδα. Αυτό απλοποιεί και βοηθάει τη συντήρηση της υπηρεσίας.

Δομή:

Για τη δομή αυτού του μοτίβου απαιτείται η δημιουργία μιας λίστας. Η λίστα περιέχει υπηρεσίες με συναφή λειτουργικότητα, είναι ουσιαστικά ένα σχέδιο

απογραφής υπηρεσιών. Τα στρώματα των υπηρεσιών ομαδοποιούνται με βάση τη λειτουργία ώστε να διευκολύνει και τον σκοπό του μοτίβου, δηλαδή της επαναχρησιμοποίησης των υπηρεσιών σε άλλες εφαρμογές. Η δομή συνήθως για μια κοινή διαστρωμάτωση χρησιμοποιεί τα καθήκοντα, τις οντότητες και τις δυνατότητες. Η προσέγγιση απόδοσης από τη κορυφή προς τα κάτω της λίστας καθιστά πιο εύκολη τη χρήση του.

Πως λειτουργεί:

Το service layer μπορεί να εφαρμοστεί με διάφορους τρόπους χωρίς να παραβιάζει τα καθοριστικά χαρακτηριστικά του. Η ουσία του κρύβεται στην κατανομή των ευθυνών πίσω από τα στρώματα υπηρεσιών του μοτίβου. Ένας τρόπος υλοποίησης του είναι να δημιουργείται ως ένα σύνολο απεικονίσεων πάνω από ένα μοτίβο Domain model(3.1.2). Όπου σε αυτή τη περίπτωση την επιχειρησιακή λογική την υλοποιεί το domain model(3.1.2) ενώ το service layer μέσω των απεικονίσεων δημιουργεί ένα συγκεκριμένο σύνολο λειτουργιών μέσω του οποίου ο πελάτης αλληλεπιδρά με την εφαρμογή. Σε έναν άλλο βασικό τρόπο υλοποίησης το service layer μοτίβο εφαρμόζεται ως ένα σύνολο κλάσεων με περισσότερες λειτουργίες όπου υλοποιεί άμεσα τη λογική της εφαρμογής αλλά τις μεταβιβάζει σε ενθυλακωμένες κλάσεις του domain object για κύρια λογική. Οι λειτουργίες που παρέχονται στον πελάτη του μοτίβου υλοποιούνται σε δέσμες ενεργειών οργανωμένες σε μία κλάση που ορίζει ένα θέμα σχετικής λογικής. Κάθε τέτοια κλάση αποτελεί για την εφαρμογή υπηρεσία, λόγω αυτού συνηθίζεται το όνομα τους να τελειώνει με “Service” για αναγνωριστικούς λόγους για έναν προγραμματιστή.

Με μία πιο γενική οπτική το μοτίβο αυτό λειτουργεί μέσω κλάσεων για να διευκολύνει τις κινήσεις ενός πελάτη. Εξασφαλίζει την εκτέλεση λειτουργιών όπου μπορεί να κάνει χρήση κάθε πελάτης μέσω της διεπαφής χρήστη. Ουσιαστικά είναι το μοντέλο περιπτώσεων χρήσης και ο σχεδιασμός διεπαφής χρήστη για την εφαρμογή. Μέσα σε μία κλάση καθορίζει στην κάθε περίπτωση τι θα ακολουθήσει μετά ώστε να αντιμετωπίσει κάτι που έχει νόημα για τον πελάτη, δηλαδή λειτουργεί σαν συντονιστής των αιτήσεων και των αποκρίσεων.

Πλεονεκτήματα και μειονεκτήματα:

Το πλεονέκτημα του μοτίβου αυτού είναι ότι ορίζει ένα κοινό σύνολο λειτουργιών της εφαρμογής που είναι διαθέσιμες σε πολλών ειδών πελάτες και συντονίζει την απόκριση της εφαρμογής για κάθε πιθανή ενέργεια. Ένα ακόμα μεγάλο προνόμιο είναι ότι είναι εύκολο να προστεθούν νέες μέθοδοι παροχής υπηρεσιών, επειδή οι λειτουργίες είναι επαναχρησιμοποιήσιμες. Επίσης υπάρχει η δυνατότητα για

ασφαλή και εύκολο τρόπο αλλαγής μεθόδων διότι οι αλλαγές δεν επηρεάζουν άλλη μέθοδο.

Ένα μειονέκτημα του είναι ότι η διαδικασία χειρισμού του αιτήματος και της απόκρισης στη κατάσταση ενός αντικειμένου είναι σύνθετη και πολλές φορές επιφυλάσσει επιπλοκές στην ακεραιότητα των ενεργειών. Όπως ακόμα υπάρχει περίπτωση να οδηγηθεί κανείς στο να αντιμετωπίσει μεγάλο αριθμό αιτήσεων και αποκρίσεων. Τέλος, χρειάζεται να υπάρχει μεγάλη προσοχή στην οποιαδήποτε εξάρτηση μπορεί να δημιουργηθεί διότι μετά για οποιαδήποτε αλλαγή χρειάζεται επανεξέταση μεγάλου όγκου λειτουργιών.

Εφαρμογές:

Η καταλληλότερη εφαρμογή αυτού του μοτίβου είναι για επιχειρησιακές λογικές που απευθύνονται σε πολλών ειδών πελάτες. Επιπλέον είναι κατάλληλο για όσες εφαρμογές χρειάζονται να ενσωματώσουν το domain logic κάτω από το API (Application Programming Interface).

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**,

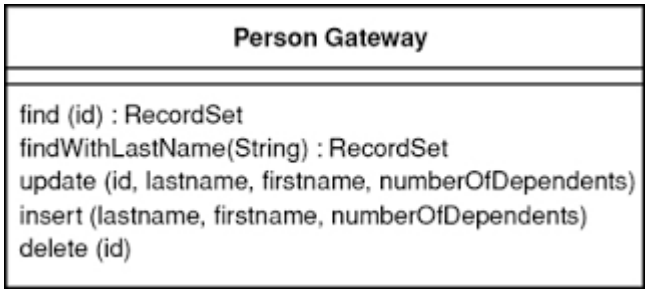
Book **SOA Design Patterns** by **Thomas Erl,**

Book **Service-oriented Architecture Compass: Business Value, Planning, and Enterprice Roadmap** by **Norbert Bieberstein, Sanjay Bose, Marc Fiammante, Keith Jones, Rawn Shah.**

3.2 Data Source Architectural Patterns

3.2.1 Table Data Gateway

Ένα αντικείμενο που δρα σαν πύλη σε ένα πίνακα βάσης δεδομένων. Μία ύπαρξη που χειρίζεται όλες τις σειρές του πίνακα.



Σκοπός:

Ο σκοπός αυτού του μοτίβου είναι να χωριστεί η ευθύνη της ανάκτησης αντικειμένων από μία βάση δεδομένων από τις πραγματικές χρήσεις αυτών των αντικειμένων. Οι χρήστες της πύλης απομονώνονται από τις αλλαγές στον τρόπο αποθήκευσης των αντικειμένων στη βάση δεδομένων. Ουσιαστικά, ο στόχος του είναι να ενσωματώσει την πλήρη αλληλεπίδραση με έναν πίνακα βάσης δεδομένων, διατηρώντας συμπαγή τη λογική για συγκεκριμένες υλοποιήσεις των ενεργειών των αντικειμένων.

Δομή:

Στην πραγματικότητα, η δομή αυτού του μοτίβου είναι το αντικειμενοστραφές ισοδύναμο ενός σχεσιακού πίνακα. Κατά κύριο λόγο είναι ένα σχεδιαστικό μοτίβο στο οποίο ένα αντικείμενο λειτουργεί ως πύλη σε έναν πίνακα βάσης δεδομένων. Δημιουργώντας ένα αντικείμενο όπου μέσω των κλάσεων του, που κωδικοποιούνται, χειρίζεται όλα τα ερωτήματα SQL και παρουσιάζει μια συγκεκριμένη διεπαφή.

Πως λειτουργεί:

Το table data gateway έχει μία απλή διεπαφή, που συνήθως αποτελείται από πολλές ευρηματικές μεθόδους έτσι ώστε να πάρει στοιχεία από τη βάση δεδομένων και μεθόδους ενημέρωσης, εισαγωγής και διαγραφής. Συνήθως για την επιστροφή πληροφορίας από ένα ερώτημα επιστρέφει κάποια απλή δομή δεδομένων (σαν χάρτη) ή ένα αντικείμενο μεταφοράς δεδομένων. Είναι πολύ βολικό να χρησιμοποιείται το Record set(3.10.11) για τα αποτελέσματα που έρχονται από τα ερωτήματα SQL.

Πλεονεκτήματα και μειονεκτήματα:

Το πλεονέκτημα στη χρήση αυτού του μοτίβου είναι ότι η ίδια διεπαφή μπορεί να δουλέψει και για τη χρήση SQL ερωτημάτων με αποτελεσματικό χειρισμό της βάσης δεδομένων και για τη χρήση αποθηκευμένων διαδικασιών. Επίσης κάθε μέθοδος χαρτογραφεί τις παραμέτρους εισόδου σε μια κλήση SQL και το εκτελεί χωρίς σύνδεση στη βάση δεδομένων.

Το βασικό του μειονέκτημα είναι ότι ένας προγραμματιστής χρειάζεται να είναι καλά εξοικειωμένος με την SQL. Οι περισσότεροι προγραμματιστές δεν είναι τόσο αλλά και όσοι είναι δεν είναι βέβαιο ότι αξιοποιούνται όλες οι δυνατότητες της γλώσσας. Επίσης, λόγω της έντονης μετακίνησης δεδομένων είναι πιθανό κάποιες φορές κάποια δεδομένα να είναι άγνωστης προέλευσης.

Εφαρμογές:

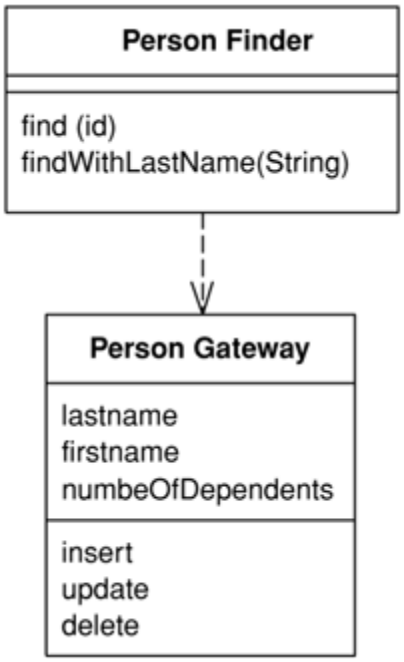
Η κατάλληλη εφαρμογή αυτού του μοτίβου είναι για περιπτώσεις που χρειάζεται ή είναι βολικότερο να χρησιμοποιήσουμε κάποια πύλη να προσεγγίζει όλα τα δεδομένα και μετά το καθένα ξεχωριστά. Είναι από τα πιο απλά μοτίβα διεπαφής βάσης δεδομένων και είναι αρκετά χρήσιμο σαν ένας καλός χάρτης όπου συχνά μπορεί να χρειαστεί μέσω αυτού να υπάρξει επικοινωνία στην βάση δεδομένων.

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.2.2 Row Data Gateway

Ένα αντικείμενο που λειτουργεί ως πύλη σε μια μόνο εγγραφή σε μία δεδομένη πηγή. Υπάρχει μία περίπτωση ανά σειρά.



Σκοπός:

Το row data gateway είναι ένα σχεδιαστικό μοτίβο που έχει σκοπό να λειτουργεί σαν πύλη για μία σειρά της βάσης δεδομένων και να δίνει αντικείμενα που μοιάζουν ακριβώς με την εγγραφή στη δομή των εγγραφών. Οπότε στοχεύει να λειτουργεί ως μία καλή διασύνδεση για κάθε σειρά δεδομένων.

Δομή:

Η δομή του μοτίβου αυτού είναι δομή ενός αντικειμένου που μιμείται μια ενιαία εγγραφή ή μια σειρά βάσης δεδομένων. Κάθε στήλη στη βάση δεδομένων αποτελεί ένα πεδίο γι’ αυτό το αντικείμενο. Ουσιαστικά μπορεί να είναι μία κλάση όπου οι ενέργειες της μοιάζουν με τα ερωτήματα SQL για μια βάση δεδομένων, μόνο που

αφορά μία γραμμή αυτής. Συνήθως οι ενέργειες αυτές μπορούν να θεωρηθούν είτε χαρακτηριστικά δεδομένων που αντιστοιχούν ακριβώς στις στήλες στον πίνακα είτε μέθοδοι που χειρίζονται ερωτήματα στη βάση δεδομένων.

Πως λειτουργεί:

Η λειτουργία αυτού του μοτίβου κυρίως είναι να διατηρεί τα δεδομένα με μία σειρά, έτσι ώστε ο πελάτης μέσω μιας καλής διασύνδεσης να έχει πρόσβαση στη πύλη, δηλαδή σε κάθε σειρά δεδομένων της βάσης. Όσων αφορά τις λειτουργίες εύρεσης , είναι προτιμότερο να έχουν διαφορετική κλάση από αυτής της πηγής για να μπορεί να υπάρχει πολυμορφισμός ή να χρησιμοποιηθεί στατική μέθοδος εύρεσης. Γενικότερα οι λειτουργίες της αφορούν τη λογική πρόσβασης στη βάση δεδομένων και όχι λογική για το domain.

Πλεονεκτήματα και μειονεκτήματα:

Το row data gateway αποδίδει θετικά συνήθως μαζί με το Metadata mapping(3.5.1) όπου είναι μοτίβο χαρτογράφησης μεταδεδομένων που περιγράφεται παρακάτω. Μπορεί να φτιαχτεί αυτόματα όλος ο κώδικας για τη πρόσβαση στη βάση δεδομένων κατά τη διάρκεια της διαδικασίας δημιουργίας.

Τα βασικά μειονεκτήματα είναι ότι αυξάνεται η πολυπλοκότητα εφόσον χρειάζεται τα αντικείμενα που αποθηκεύονται να έχουν δική τους επιχειρησιακή λογική και υπάρχει δυσκολία στον τρόπο ενημερώσεων. Μπορεί να δημιουργήσει προβλήματα μία πιθανή ενημέρωση ενός row data gateway εάν υπάρχουν δύο τέτοια μοτίβα που λειτουργούν στον ίδιο πίνακα.

Εφαρμογές:

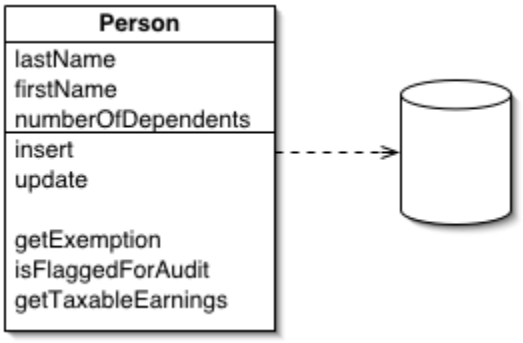
Συνήθως αυτό το μοτίβο είναι κατάλληλο να εφαρμοστεί όταν χρησιμοποιείται το transaction script(3.1.1) και ακατάλληλο με το domain model(3.1.2). Επιπλέον όπως είπαμε προηγουμένως είναι αρκετά αποδοτικό σε προγράμματα βασισμένα στο metadata mapping(3.5.1).

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler.**

3.2.3 Active Record

Ένα αντικείμενο που αναδιπλώνει μια σειρά σε έναν πίνακα βάσης δεδομένων ή ενός view, ενθυλακώνει την πρόσβαση στη βάση δεδομένων και προσθέτει λογική domain σε αυτά τα δεδομένα.



Σκοπός:

Το μοτίβο είναι ένα αντικείμενο που είναι υπεύθυνο για τα επιχειρησιακά δεδομένα και τη συμπεριφορά. Σκοπός αυτού του αντικειμένου είναι να διευκολύνει τη δημιουργία και τη χρήση επιχειρησιακών αντικειμένων των οποίων τα δεδομένα απαιτούν μόνιμη αποθήκευση σε βάση δεδομένων. Ουσιαστικά η αποθήκευση αυτών των δεδομένων των αντικειμένων παραπέμπουν σε σχεσιακή βάση δεδομένων όπου στη διεπαφή τους περιλαμβάνονται λειτουργίες που εκφράζουν αυτές τις σχέσεις.

Δομή:

Προσεγγιστικά το active record είναι ένας τρόπος πρόσβασης σε βάση δεδομένων. Μία υπόσταση ενός αντικειμένου είναι συνδεδεμένη με μία σειρά ενός πίνακα μίας βάσης δεδομένων ο οποίος περιλαμβάνεται σε μία κλάση. Οπότε η δομή του διαμορφώνεται μέσω της κλάσης με μεθόδους ή ιδιότητες πρόσβασης για κάθε στήλη στον πίνακα και πρέπει να ταιριάζει ακριβώς με της βάσης δεδομένων.

Πως λειτουργεί:

Η διεπαφή ενός αντικειμένου που χρησιμοποιεί αυτό το μοτίβο περιλαμβάνει τις λειτουργίες της εισαγωγής, ενημέρωσης και διαγραφής, όπως ακόμα και ιδιότητες που αντιστοιχούν άμεσα στις στήλες του πίνακα της βάσης δεδομένων. Πρακτικά σε κάθε δημιουργία ενός αντικειμένου που αποθηκεύεται, προστίθεται μία νέα σειρά στον πίνακα της βάσης και κάθε υπάρχων αντικείμενο παίρνει πληροφορίες από τη βάση δεδομένων. Όποτε ενημερώνεται κάποιο αντικείμενο, ενημερώνεται και η σειρά στον πίνακα. Αυτές οι μέθοδοι και οι λειτουργίες περιλαμβάνονται στην κλάση που αναφέραμε στη δομή του μοτίβου.

Πλεονεκτήματα και μειονεκτήματα:

Τα πλεονεκτήματα του active record είναι ότι περιέχει δεδομένα πηγών και λογική σε ένα αντικείμενο. Αυτό βοηθάει στο να μην αυξηθεί η πολυπλοκότητα αφού περιέχει και τα δύο. Επίσης η λογική του δεν είναι περίπλοκη και αυτό είναι πολύ

βολικό, δηλαδή μπορεί και δημιουργεί, διαβάζει, ενημερώνει και διαγράφει χωρίς να έχει κάτι πιο σύνθετο.

Το βασικό μειονέκτημα αυτού του μοτίβου είναι ότι εάν η επιχειρησιακή λογική είναι περίπλοκη, τότε δεν είναι το κατάλληλο να εφαρμόσουμε διότι θα προκύψουν πολλές δυσκολίες για την αντιμετώπιση σύνθετων καταστάσεων.

Εφαρμογές:

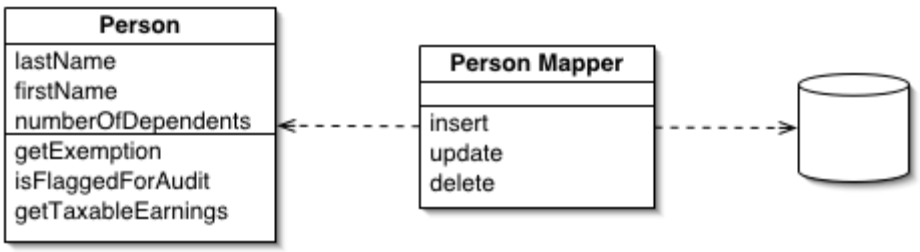
Προφανώς μέσω των πλεονεκτημάτων βγαίνει συμπέρασμα ότι το active record εφαρμόζεται σε επιχειρησιακές λογικές που δεν είναι περίπλοκες, δηλαδή σε κάτι με απλή λογική. Επιπλέον, συνήθως χρησιμοποιείται σε αντικειμενοστραφή χαρτογράφηση και από εργαλεία επίμονων αντικειμένων (όπου έτσι συνήθως αναφέρονται τα αντικείμενα που έχουν εμμονική συμπεριφορά όσων αφορά τη λειτουργία τους μέσα στο πρόγραμμα).

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler.**

3.2.4 Data Mapper

Ένα στρώμα χαρτογράφησης που μετακινεί δεδομένα μεταξύ αντικειμένων και σε μια βάση δεδομένων, διατηρώντας παράλληλα την ανεξαρτησία τους το ένα από το άλλο και από τον ίδιο το χάρτη.



Σκοπός:

Ο σκοπός του μοτίβου είναι να διατηρηθούν ανεξάρτητα μεταξύ τους η αναπαράσταση μνήμης δεδομένων, η μόνιμη αποθήκευση δεδομένων και ο ίδιος ο χάρτης δεδομένων.

Δομή:

Το data mapper είναι ένα Data Access Layer, δηλαδή ένα στρώμα πρόσβασης δεδομένων, που εκτελεί αμφίδρομη μεταφορά δεδομένων μεταξύ των μόνιμα

αποθηκευμένων δεδομένων (όπως μίας σχεσιακής βάσης δεδομένων) και μίας αναπαράστασης μνήμης δεδομένων. Το στρώμα αυτό αποτελείται από έναν ή περισσότερους χαρτογράφους που εκτελούν τη μεταφορά δεδομένων. Αυτή η ανεξαρτησία δομικά σημαίνει ότι οι κλάσεις, που περιέχουν την αναπαράσταση μνήμης δεδομένων, μπορούν να χρησιμοποιήσουν στρατηγικές ή μεθόδους συμπεριφοράς ή ακόμα και κληρονομικότητα χωρίς να επηρεάζει τα μόνιμα αποθηκευμένα δεδομένα και ας υπάρχει η δυνατότητα επικοινωνίας μεταξύ τους.

Πως λειτουργεί:

Ένα αντικείμενο διαμορφωμένο με αυτό το μοτίβο θα περιλαμβάνει στη διεπαφή του λειτουργίες όπως η δημιουργία, ενημέρωση, ανάγνωση και διαγραφή, οι οποίες λειτουργούν σε αντικείμενα που αντιπροσωπεύουν τύπους οντοτήτων του domain σε μόνιμα αποθηκευμένα δεδομένα. Ουσιαστικά σε μία απλή περίπτωση, όταν ένας πελάτης ζητήσει να γίνει μία ενέργεια, ο χάρτης θα αντιστοιχίσει την εικονική μνήμη με την μόνιμη και θα εκτελέσει την κατάλληλη επιχειρησιακή λογική πάνω στην εικονική μνήμη και στο τέλος εάν χρειαστεί θα ενημερώσει και τις δύο. Συνήθως δεν είναι τόσο απλή η περίπτωση που έχει να χειριστεί το μοτίβο αλλά ήταν μία απλή περίπτωση για την βασική κατανόηση του.

Πλεονεκτήματα και μειονεκτήματα:

Το κύριο πλεονέκτημα του είναι ότι μπορεί να υποστηρίξει περίπλοκες λογικές. Λόγω της ανεξαρτησίας του βοηθάει στο να μην επιβαρύνεται συνεχώς η μόνιμη μνήμη ή σχεσιακή βάση δεδομένων γιατί μπορεί να λειτουργήσει ανεξάρτητα με τα αντικείμενα στην εικονική μνήμη.

Ένα αρνητικό σε αυτό το μοτίβο είναι ότι μπορεί να γίνει πολύ περίπλοκη μία κατάσταση ώστε να γίνει χρονοβόρα η αντιμετώπιση του.

Εφαρμογές:

Το μοτίβο αυτό ενδείκνυται σε περιπτώσεις με σύνθετη επιχειρησιακή λογική και σε περιπτώσεις όπου είναι χρήσιμο και βολικό να υπάρχει ανεξαρτησία μεταξύ βάσης δεδομένων, εικονικής μνήμης δεδομένων και του χάρτη. Με συνέπεια να μπορεί η λογική να υλοποιηθεί ανεξάρτητα από τη βάση δεδομένων.

Βιβλιογραφία:

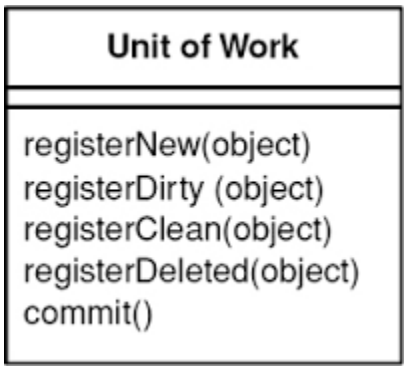
Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler.**

3.3 Object-Relational Behavioral Patterns

3.3.1 Unit of Work

Διατηρεί μια λίστα αντικειμένων που επηρεάζονται από μια συναλλαγή επιχείρησης και συντονίζει το γράψιμο των αλλαγών και την επίλυση των προβλημάτων ταυτόχρονης συμπεριφοράς.



Σκοπός:

Ο σκοπός αυτού του μοτίβου όπως φαίνεται και από τον ορισμό του είναι να διατηρεί ενημερώσεις των αντικειμένων εντός της μνήμης και να τις αποστέλλει εντός μνήμης ως μία συναλλαγή στη βάση δεδομένων. Ουσιαστικά, παρακολουθώντας όσα γίνονται κατά τη διάρκεια μίας συναλλαγής όπου επηρεάζουν τη βάση δεδομένων, έχει στόχο να αποφορτίσει τη βάση δεδομένων από συνεχείς πιθανές μικρές κλήσεις ή παρενέργειες και αναλαμβάνει να κάνει τις συνολικές αλλαγές στη βάση δεδομένων όταν τελειώσει το έργο της συναλλαγής.

Δομή:

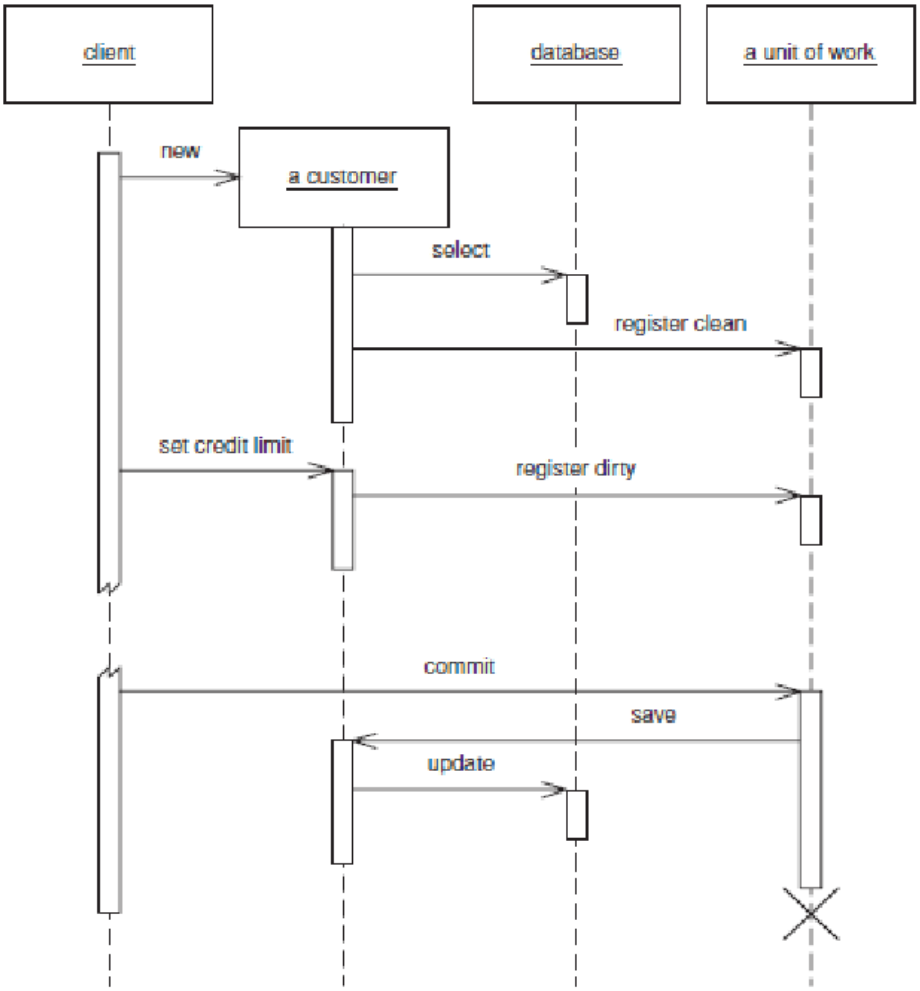
Η δομή του είναι ένα σύνολο πράξεων όπου μπορούν να εκπροσωπούν διαφορετικών τύπων ενέργειες όπως κλήσεις Web Service (υπηρεσίες που αφορούν τον ιστότοπο), λειτουργίες βάσης δεδομένων ή ακόμα και εργασίες μνήμης. Συνήθως δομείται ως μία κλάση όπου απορροφά όλες τις ενέργειες και τις αλλαγές σε πιθανών κάποιες λίστες όπου κρατάει τα αντικείμενα που δημιουργήθηκαν, άλλαξαν και διαγράφηκαν και όταν ολοκληρωθεί το έργο της ενέργειας θα ενημερώσει την βάση για όλες τις αλλαγές.

Πως λειτουργεί:

Οι κύριες λειτουργίες του είναι να διατηρεί τους καταλόγους των αντικειμένων που έχουν εισαχτεί, ενημερωθεί ή διαγραφεί κατά τη διάρκεια μίας συναλλαγής και μόλις ολοκληρωθεί θα αποστέλλει μία ενιαία εργασία όπου θα ενημερώσει τη βάση

δεδομένων με μία κίνηση. Ουσιαστικά η δομή του μοτίβου εκφράζει και τη λειτουργία του.

Σχήμα: Λήψη του αντικειμένου του δέκτη για την εγγραφή του.



Ένα βασικό χαρακτηριστικό του unit of work είναι ότι όταν έρχεται η ώρα να πράξει, τότε αποφασίζει μόνο του τι να κάνει. Κάνει τις ενέργειες μίας συναλλαγής και μετά στέλνει τις αλλαγές στη βάση δεδομένων. Επίσης είναι χρήσιμο το μοτίβο να είναι ενημερωμένο ποιά αντικείμενα πρέπει να παρακολουθεί, αλλά αυτό δε σημαίνει ότι οποιοδήποτε άλλο θα το διαγράψει, απλά αυτό ίσως προκαλέσει μία σχετική καθυστέρηση στο πρόγραμμα. Τέλος, στην λειτουργία του υπάρχει και η σημαία που μπορεί να βάλει σε κάποια αντικείμενα που έχουν αλλάξει ως “dirty” ώστε να ξέρει ποια αντικείμενα χρειάζεται να παρακολουθεί και δέχονται αλλαγές.

Πλεονεκτήματα και μειονεκτήματα:

Το βασικότερο πλεονέκτημα του είναι ότι μπορεί και μαζεύει ένα σύνολο ενεργειών που αφορούν κλήσεις στη βάση δεδομένων και την μετατρέπει σε μία ενιαία κλήση ή ενέργεια προς τη βάση δεδομένων. Επίσης είναι ένα μοτίβο όπου συνίσταται σαν μία σταθερή πλατφόρμα για περίπλοκες συναλλαγές.

Αρνητική λειτουργία του μοτίβου αυτού θα συναντήσει κανείς στην περίπτωση χρήσης ενός domain model(3.1.2) για δύο λόγους. Επειδή το domain model(3.1.2) είναι ένα γενικό δίκτυο αντικειμένων οπότε είναι δύσκολο να διασχιστεί και διότι για το μοτίβο unit of work είναι βολικό να μπορεί να χρησιμοποιεί μεταβλητές όπου δεν ενδείκνυται με το domain model(3.1.2).

Εφαρμογές:

Το μοτίβο αυτό εφαρμόζεται καλά με το transaction script(3.1.1) λόγω της δυνατότητας χρήσης μεταβλητών. Επίσης έχει κατάλληλη εφαρμογή, για περιπτώσεις όπου όλη η εργασία γίνεται μέσα σε μία συναλλαγή του συστήματος που χρειάζεται παρακολούθηση, εκεί είναι ιδανική επιλογή το unit of work.

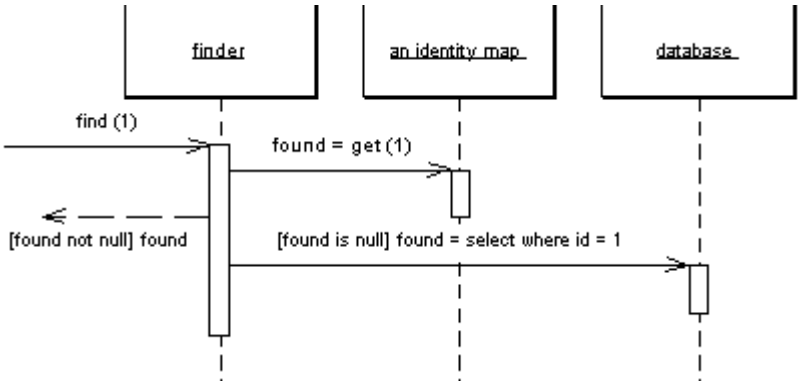
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.3.2 Identity Map

Εξασφαλίζει ότι κάθε αντικείμενο φορτώνεται μόνο μία φορά διατηρώντας κάθε φορτωμένο αντικείμενο σε ένα χάρτη. Αναζητά τα αντικείμενα που χρησιμοποιούν το χάρτη όταν γίνεται αναφορά σε αυτά.



Σκοπός:

Εάν φορτωθούν σε δύο διαφορετικά αντικείμενα μία εγγραφή ίδιας βάσης δεδομένων αυτό μπορεί να προκαλέσει πρόβλημα απόδοσης ή ακόμα και σφάλματος στην τελική ενημέρωση των δεδομένων στη βάση δεδομένων. Ο σκοπός του identity map είναι να μην συμβεί αυτό. Οπότε ο κύριος στόχος του είναι να βελτίωση την απόδοση της μνήμης cache αποφεύγοντας την διπλή ανάκτηση ίδιων αντικειμένων.

Δομή:

Η δομή αυτού του μοτίβου είναι ένας χάρτης όπου κρατάει τα αντικείμενα που είναι ήδη φορτωμένα.

Πως λειτουργεί:

Ο χάρτης αυτός λειτουργεί σαν μεσολαβητής των ενεργειών με τη βάση δεδομένων. Εάν ζητηθεί να φορτωθεί κάποιο αντικείμενο, πρώτα θα γίνει αναζήτηση στον χάρτη ώστε εάν είναι ήδη φορτωμένο το αντικείμενο να επιστραφεί η ίδια εμφάνιση του ήδη υπάρχοντος αντικειμένου. Εάν δεν υπάρχει το αντικείμενο στον χάρτη, τότε φορτώνεται από τη βάση δεδομένων και εισάγεται το αντικείμενο στον χάρτη για πιθανή μελλοντική ενέργεια. Για την άρτια λειτουργία τέτοιων χαρτών σε βάσεις δεδομένων είναι σύνηθες να χρησιμοποιούνται κάποια κλειδιά αναγνωριστικά που απευθύνονται άμεσα στον ίδιο τον χάρτη (θα μπορούσε να είναι και το πρωτεύον κλειδί αντικειμένων από μία βάση δεδομένων αλλά για αποδοτικότερη σύγκριση στον χάρτη είναι προτιμότερο να χρησιμοποιείται άλλο ξεχωριστό κλειδί για τον χάρτη). Υπάρχουν 4 τύποι του identity map: σαφής, γενικός, session και κλάση. Για παράδειγμα στον σαφή χάρτη η προσέγγιση των αντικειμένων για κάθε είδος αυτών γίνεται αναζητώντας κάποιο συγκεκριμένο χαρακτηριστικό ενώ σε έναν γενικό χάρτη θα μπορούσαν να υπάρχουν διάφορων ειδών αντικείμενα οπότε η αναζήτηση μπορεί να χρειάζεται περισσότερες παραμέτρους.

Πλεονεκτήματα και μειονεκτήματα:

Το σημαντικότερο πλεονέκτημα αυτού του μοτίβου είναι η βελτίωση της απόδοσης στη μνήμη cache μέσω του χειρισμού της αποφυγής διπλών ανακτήσεων ίδιων αντικειμένων. Επίσης θετικό είναι ότι μειώνει τις άσκοπες κλήσεις στη βάση δεδομένων.

Αρνητικά μπορεί να λειτουργήσει σε περίπτωση όπου τα αντικείμενα που υπάρχουν στη βάση δεδομένων δεν επιδέχονται αλλαγές και παραμένουν αμετάβλητα, όπου και εκεί απλά δεν μπορεί να χρησιμοποιηθεί αξιόλογα σαν χάρτης αλλά ίσως μόνο σαν δείκτης που μάλλον μοιάζει περιττό.

Εφαρμογές:

Η χρήση αυτού του μοτίβου ενδείκνυται για εφαρμογές όπου μπορεί να έχουν συχνές τροποποιήσεις και ενημερώσεις ίδιων αντικειμένων. Γεγονός όπου μοιάζει απαραίτητο για πολλές επιχειρησιακές εφαρμογές με μεγάλο πελατολόγιο.

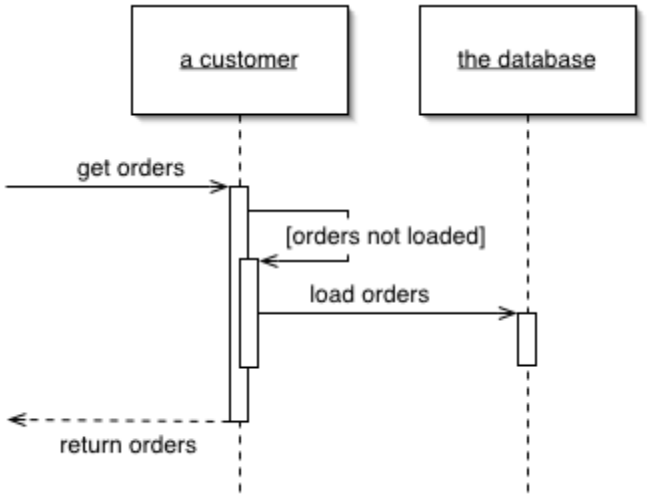
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.3.3 Lazy Load

Ένα αντικείμενο που δεν περιέχει όλα τα δεδομένα που χρειάζεται αλλά ξέρει πώς να τα πάρει.



Σκοπός:

Στόχος του lazy load είναι να φορτώσει μόνο όσα είναι απαραίτητα από τα δεδομένα ενός αντικειμένου. Εκτιμώντας ότι πολλές φορές όταν φορτώνουν όλα τα δεδομένα ενός αντικειμένου κάποια από αυτά δε χρησιμοποιούνται, το μοτίβο αυτό επιδιώκει να μη φορτώσει δεδομένα που δε χρειάζονται μέχρι τη στιγμή που όντως μπορεί να χρειαστούν και άρα να κάνει αποδοτικότερο το πρόγραμμα.

Δομή:

Επειδή υπάρχουν 4 διαφορετικές υλοποιήσεις αυτού του μοτίβου, η δομή τους θα γίνει καλύτερα κατανοητή μαζί με τον τρόπο λειτουργίας του καθενός που θα περιγραφεί παρακάτω.

Πως λειτουργεί:

Υπάρχουν 4 διαφορετικές υλοποιήσεις:

- Lazy initialization (τεμπέλικη αρχικοποίηση)
- Virtual proxy (εικονικός διακομιστής)
- Ghost (φάντασμα)
- Value holder (κάτοχος αξίας)

Με την υλοποίηση του lazy initialization κάθε πρόσβαση στο πεδίο ελέγχει πρώτα να δει εάν είναι null. Αν είναι, ελέγχει την αξία του πεδίου που θα επιστρέψει όπου γίνεται μέσω μίας μεθόδου λήψης όπου πρέπει να εξασφαλιστεί ότι έχουμε πρόσβαση σε όλο το πεδίο. Αν δεν είναι, προφανώς κρατάει ότι έχει ήδη φορτώσει. Δομικά αυτό συμβαίνει μέσα σε μία κλάση σηματοδοτήσεων για το εάν το πεδίο έχει φορτωθεί ή όχι. Επειδή η χρήση αυτού μπορεί να αναπτύξει μία εξάρτηση του αντικειμένου με τη βάση δεδομένων, συνίσταται να γίνεται χρήση με τα μοτίβα active record(3.2.3) και row data gateway(3.2.2).

Το virtual proxy είναι ένα αντικείμενο που μοιάζει με το αντικείμενο που ζητήθηκε να φορτωθεί αλλά στην πραγματικότητα δεν περιέχει τίποτα. Ο τρόπος να φορτώσει το σωστό αντικείμενο είναι να ονομαστεί μία από τις μεθόδους της. Παρότι μπορεί να μοιάζει με το ζητούμενο αντικείμενο υπάρχει η περίπτωση να υπάρξει πρόβλημα ταυτότητας. Ένας τρόπος να αντιμετωπιστεί αυτό είναι να χρησιμοποιηθούν δομικά λίστες. Αυτή η υλοποίηση λειτουργεί καλά με το data mapper(3.2.4).

Η υλοποίηση του φαντάσματος είναι το ζητούμενο αντικείμενο σε μερική κατάσταση. Δηλαδή στην πρώτη φόρτωση του αντικειμένου επιστρέφεται μόνο το αναγνωριστικό της βάσης δεδομένων και σε οποιαδήποτε ενέργεια πρόσβασης σε αυτό το πεδίο επιστρέφει την πλήρη κατάσταση. Δομείται απλά με μία κλάση.

Τέλος, το value holder από τη πρώτη πρόσβαση κρατάει την αξία του ζητούμενου αντικειμένου. Οπότε αυτό κρατά τις ιδιότητες και τη συμπεριφορά του αντικειμένου. Είναι χρήσιμο αυτός ο κάτοχος να βρίσκεται στην ιδιοκτησία της κλάσης που δομείται το μοτίβο για να προλάβουμε προβλήματα ταυτότητας.

Πλεονεκτήματα και μειονεκτήματα:

Υπάρχουν δύο θετικά με αυτό το μοτίβο. Το πρώτο είναι ότι φορτώνει μνήμη μόνο όση χρειάζεται και όταν την χρειάζεται και το δεύτερο είναι ότι μειώνει τον αρχικό χρόνο φόρτωσης κάνοντας την εκτέλεση αργότερα. Επομένως δίνει μία γρηγορότερη απόδοση στο πρόγραμμα.

Το μειονέκτημα του είναι ότι μπορεί να υπάρξει καθυστέρηση όταν είναι απαραίτητο από την αρχή- να χρησιμοποιηθεί όλο το αντικείμενο.

Εφαρμογές:

Το lazy load είναι αποδοτικό να εφαρμόζεται σε εφαρμογές που δεν έχουν μόνο μία αλληλεπίδραση με τον χρήστη και υπάρχει μεγάλη πιθανότητα να συμβεί και άλλη κλήση στη βάση δεδομένων. Γενικά είναι προτιμότερο να φορτώνονται όλα σε μία κλήση, αλλά όταν γίνεται σε μία επιπλέον κλήση και τα δεδομένα που ανακτώνται δεν χρησιμοποιούνται μαζί με το κύριο αντικείμενο, τότε αυτό το μοτίβο δίνει αποδοτικότερο χρόνο στην εκτέλεση του προγράμματος.

Βιβλιογραφία:

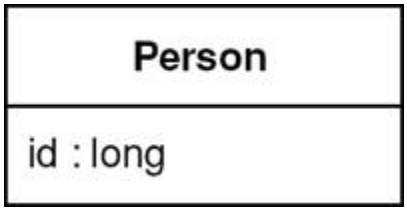
Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4 Object-Relational Structural Patterns

3.4.1 Identity Field

Αποθηκεύει ένα πεδίο ID της βάσης δεδομένων σε ένα αντικείμενο για να διατηρήσει την ταυτότητα μεταξύ ενός αντικειμένου μνήμης και μια σειράς βάσης δεδομένων.



Σκοπός:

Ο σκοπός του identity field είναι να βελτιώσει την επικοινωνία μεταξύ της σχεσιακής βάσης δεδομένων και των αντικειμένων που βρίσκονται στη μνήμη.

Δομή:

Η δομή αυτού του μοτίβου είναι μία κλάση που αποτελεί ένα αναπόσπαστο πεδίο σε μία βάση δεδομένων που αντιστοιχίζει ή χαρτογραφεί ένα ολοκληρωμένο πεδίο ενός αντικειμένου μέσα στη μνήμη.

Πως λειτουργεί:

Η βασική του λειτουργία για να πετύχει το στόχο του είναι να αποθηκεύει ένα κλειδί, που συνήθως είναι το πρωτεύον του πίνακα σχεσιακής βάσης δεδομένων, στα πεδία του αντικειμένου. Για να υλοποιηθεί σωστά η χαρτογράφηση των αντικειμένων χρειάζεται το κλειδί αυτό να είναι μοναδικό και αμετάβλητο για το αντικείμενο που εκπροσωπεί στην βάση δεδομένων, γι' αυτό και συνηθίζεται να χρησιμοποιείται το πρωτεύον κλειδί όπως αναφέρθηκε παραπάνω. Εάν δε χρησιμοποιηθεί το πρωτεύον κλειδί τότε υπάρχει η επιλογή ενός απλού κλειδιού ή ενός σύνθετου. Το απλό κλειδί μπορεί να είναι εύχρηστο αλλά πολλές φορές μπορεί να προκαλέσει προβλήματα και να υπάρξει κάποια διπλοτυπία. Το σύνθετο κλειδί συνήθως χρησιμοποιεί κάποιο μοναδικό συνδυασμό χαρακτηριστικών του αντικειμένου ώστε να επιτύχει την μοναδικότητα στο κλειδί. Το κλειδί μπορεί να είναι και string και integer μεταβλητή. Η χρήση του GUID μπορεί να εξασφαλίσει την παραγωγή μοναδικού κλειδιού, όπου είναι ένα μηχάνημα που κατασκευάζει έναν μοναδικό αριθμό μέσα στο χώρο και στο χρόνο.

Πλεονεκτήματα και μειονεκτήματα:

Το βασικότερο πλεονέκτημα είναι ότι βελτιώνει την επικοινωνία μεταξύ αντικειμένων στην μνήμη και στη βάση δεδομένων. Επίσης μέσω της λειτουργίας του μοτίβου αυτού επιτυγχάνεται ότι δε θα υπάρξουν διπλοτυπίες ή κάποια σφάλματα ανάμεσα σε διπλές ανακτήσεις ίδιων αντικειμένων εφόσον βοηθάει στη χαρτογράφηση τους.

Το αρνητικό σε αυτό το μοτίβο είναι ότι ενώ έχει απλότητα στην υλοποίηση του μπορεί να υπάρξουν σύνθετες περιπτώσεις που θα χρειαστεί να αντιμετωπίσει. Αυτό μπορεί να είναι η διασφάλιση της μοναδικότητας του κλειδιού ενός αντικειμένου στη μνήμη όπου αντιστοιχεί σε μία σχεσιακή βάση δεδομένων στην οποία μπορεί να υπάρξουν σύνθετες περιπτώσεις ανάμεσα σε κάποιες βάσεις δεδομένων και του χάρτη που αντιστοιχεί συμπεριφορές και ιδιότητες.

Εφαρμογές:

Συνήθως το identity field εφαρμόζεται σε περιπτώσεις όπου υπάρχει χαρτογράφηση μεταξύ των αντικειμένων της μνήμης και των σειρών μίας σχεσιακής βάσης δεδομένων. Όπου αυτό συμβαίνει όταν χρησιμοποιείται το domain model(3.1.2) και το row data gateway(3.2.2). Επίσης το μοτίβο αυτό εφαρμόζεται και σε εφαρμογές που είναι βασισμένες στα μοτίβα active record(3.2.3) και data mapper(3.2.4).

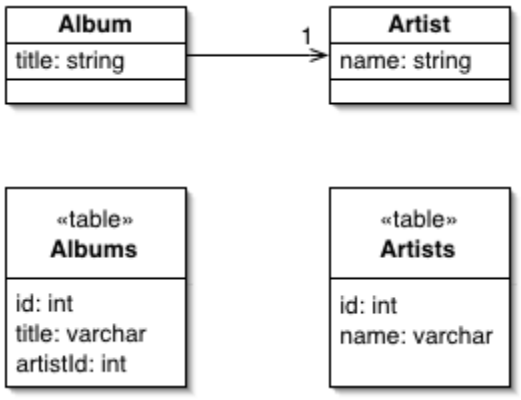
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.2 Foreign Key Mapping

Χαρτογράφει μια συσχέτιση μεταξύ αντικειμένων με αναφορά ξένων κλειδιών μεταξύ των πινάκων.



Σκοπός:

Ο σκοπός του foreign key mapping είναι να εξυπηρετήσει την επικοινωνία μεταξύ ενός αντικειμένου με πολλά. Δηλαδή εάν έχουμε δύο διαφορετικούς πίνακες όπου εκφράζουν δύο διαφορετικά είδη αντικείμενων σε μία βάση δεδομένων και μεταξύ των αντικειμένων υπάρχει συσχέτιση, αυτό το μοτίβο έχει σκοπό να διευκολύνει την εφαρμογή της σχέσης αυτής.

Δομή:

Η δομή του εξυπηρετείται μέσω κλάσεων που εκπροσωπούν αντικείμενα στη μνήμη όπου μαζί μπορεί να εκφράζουν μία ολοκληρωμένη οντότητα (ολοκληρωμένη εννοώντας ότι δίνουν περισσότερες πληροφορίες για την οντότητα). Υπάρχουν κλάσεις που μέσω αναφορών η μία στην άλλη δίνουν χαρακτηριστικά δεδομένα για μία κοινή οντότητα. Ουσιαστικά φτιάχνει ένα χάρτη συσχετίσεων.

Πως λειτουργεί:

Η λειτουργία του βασίζεται στην αναφορά των κλειδιών. Για παράδειγμα, μία κλάση που εκπροσωπεί ένα αντικείμενο έχει ένα μοναδικό κλειδί (συνήθως το πρωτεύον όπως είδαμε με το identity field(3.4.1)) και ουσιαστικά είναι ένας πίνακας στη βάση δεδομένων. Αυτό το κλειδί μπορεί να γίνει ένα ξένο κλειδί για ένα άλλο αντικείμενο (άλλη κλάση και άλλος πίνακας στη βάση δεδομένων) όπου του προσάπτει μία

πληροφορία συσχέτισης. Αυτό το ξένο κλειδί λειτουργεί σαν συσχέτιση μεταξύ των αντικειμένων, δηλαδή ο δεύτερος πίνακας στη βάση δεδομένων θα έχει και αυτός ξεχωριστά πρωτεύοντα κλειδιά για κάθε αντικείμενο (κάθε σειρά της βάσης δεδομένων). Όταν γίνονται αλλαγές σε κάποιο αντικείμενο, για να γίνει ενημέρωση στο άλλο αντικείμενο που συσχετίζεται υπάρχουν 3 τρόποι: 1) να γίνει διαγραφή όλων των αντικειμένων και εκ' νέου εισαγωγή των νέων αντικειμένων, 2) προσθήκη πίσω δείκτη, δηλαδή προσθήκη ενός απευθείας συνδέσμου από αντικείμενο σε αντικείμενο και 3) η παρατήρηση διαφορών μεταξύ των αντικειμένων που έχουν μείνει στην κλάση και των αντικειμένων που έχουν μείνει στη βάση δεδομένων.

Πλεονεκτήματα και μειονεκτήματα:

Ένα προφανές πλεονέκτημα είναι η αποδοτικότερη συσχέτιση και ενημέρωση των αντικειμένων στη μνήμη μειώνοντας την επικοινωνία με τη βάση δεδομένων. Επίσης η χαρτογράφηση που προσφέρει μπορεί να είναι χρήσιμη σχεδόν για κάθε εφαρμογή που χρησιμοποιεί σχεσιακή βάση δεδομένων.

Μειονέκτημα αυτού του μοτίβου μπορεί να χαρακτηριστεί ότι κάποιες φορές μπορεί να γίνει περίπλοκη η σύνθεση των αντικειμένων με πολλές συσχετίσεις διότι θα χρειαστεί πολλά ξένα κλειδιά, γι' αυτό χρειάζεται την κατάλληλη προσοχή για την οργάνωση του χάρτη. Επίσης σαν μειονέκτημα θα μπορούσε να ειπωθεί ότι οι συσχετίσεις αντικειμένων αφορούν από ένα σε πολλά και όχι από πολλά σε πολλά αντικείμενα.

Εφαρμογές:

Το foreign key mapping είναι κατάλληλο να υλοποιηθεί για εφαρμογές που έχουν πολλές συσχετίσεις μεταξύ των αντικειμένων της βάσης δεδομένων (από ένα σε πολλά). Γενικά μπορεί να χρησιμοποιηθεί σχεδόν σε όλες τις περιπτώσεις που υπάρχουν σχέσεις μεταξύ των κλάσεων.

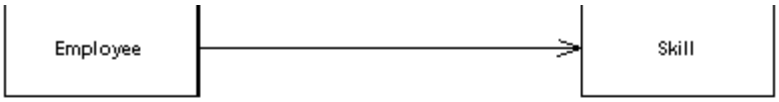
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.3 Association Table Mapping

Αποθηκεύει μια συσχέτιση ως πίνακα με ξένα κλειδιά στους πίνακες που συνδέονται από αυτή τη συσχέτιση.



Σκοπός:

Ο σκοπός αυτού του μοτίβου είναι να αναλάβει τις συσχετίσεις από πολλά προς πολλά αντικείμενα.

Δομή:

Η δομή του association table mapping είναι ουσιαστικά ένας πίνακας που εκφράζει τις συσχετίσεις μεταξύ των αντικειμένων σε άλλους πίνακες. Δηλαδή μέσω ενός πίνακα συνδέονται οι κλάσεις που συσχετίζονται μεταξύ τους.

Πως λειτουργεί:

Η λειτουργία του βασίζεται στα πρωτεύοντα κλειδιά των αντικειμένων από τους πίνακες της βάσης δεδομένων. Το μοτίβο αυτό φτιάχνει ένα πίνακα όπου αντιστοιχεί τα πρωτεύονται κλειδιά μεταξύ δύο αντικειμένων που συσχετίζονται καταχωρώντας τα στην ίδια γραμμή χωρίς να εκφράζει κάθε γραμμή κάποιο ξεχωριστό αντικείμενο. Αυτή η χαρτογράφηση δίνει μόνο την πληροφορία της συσχέτισης και αυτός ο πίνακας ονομάζεται πίνακας συνδέσμων. Οπότε για να φορτωθούν δεδομένα από αυτό τον πίνακα συνδέσμων θα γίνουν δύο κινήσεις, μία για να βρεθούν οι σειρές που αφορούν το υποκείμενο αντικείμενο και δεύτερη για να φορτώσει τις αντιστοιχίες του από τον πίνακα συνδέσμων. Αυτό είναι πολύ αποδοτικό εάν είναι ήδη φορτωμένες οι αντιστοιχίες στη μνήμη αλλιώς για να αντιμετωπιστεί αυτό θα πρέπει να υπάρξουν περισσότεροι σύνδεσμοι.

Πλεονεκτήματα και μειονεκτήματα:

Το θετικό αυτού του μοτίβου είναι ότι μπορεί να χαρτογραφήσει συσχετίσεις πολλών αντικειμένων με άλλα πολλά αντικείμενα. Αυτό μπορεί να βοηθήσει στην απόδοση της εκτέλεσης ενεργειών στην εφαρμογή γιατί όταν υπάρχουν πολλά αντικείμενα σε μία βάση που συσχετίζονται μεταξύ τους θα ήταν μεγάλη επιβάρυνση η αναζήτηση των συσχετίσεων σε κάθε ξεχωριστή περίπτωση ενώ με αυτό το τρόπο δίνεται μία οργάνωση συσχετίσεων που εξυπηρετεί το πρόγραμμα.

Ένα βασικό μειονέκτημα του είναι, όπως είπαμε και στη λειτουργία του παραπάνω, ότι χρειάζεται να γίνουν οι κατάλληλες συνδέσεις για να μη προκύπτουν πολλές φορές καταστάσεις όπου αντικείμενα που χρειάζονται και συσχετίζονται με άλλα αντικείμενα να μην είναι φορτωμένα στη μνήμη.

Εφαρμογές:

Η κύρια του εφαρμογή είναι όταν υπάρχουν συσχετίσεις από πολλά σε πολλά αντικείμενα γιατί είναι το μοναδικό μοτίβο που εξυπηρετεί αυτή την ανάγκη.

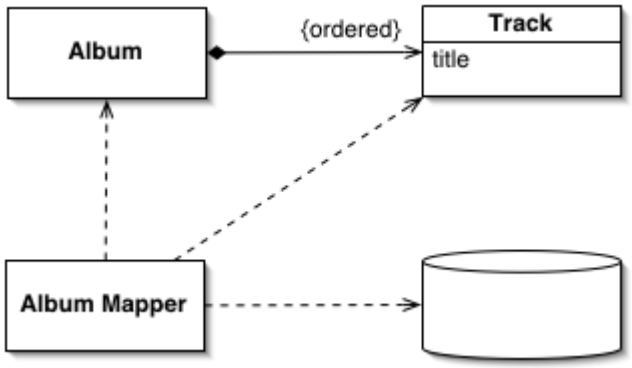
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.4 Dependent Mapping

Έχει μια κλάση που εκτελεί τη χαρτογράφηση της βάσης δεδομένων για μια κλάση παιδιού.



Σκοπός:

Το dependent mapping έχει ως στόχο να ορίσει κάποια αντικείμενα ως εξαρτώμενα αντικείμενα, αντικείμενα όπου η ύπαρξη τους εξαρτάται ολοκληρωτικά από άλλα, και να τα αποτυπώσει στην χαρτογράφηση όλων ως εξαρτώμενα από αντικείμενα ιδιοκτήτες. Ουσιαστικά ο σκοπός του είναι να αποφύγει τη σύγχυση ύπαρξης ως ανεξάρτητη οντότητα κάποιων αντικειμένων όπου ο μόνος λόγος ύπαρξης τους είναι η αναφορά τους από άλλα αντικείμενα.

Δομή:

Η δομή του είναι μία σειρά κλάσεων όπου φτιάχνουν μία εξαρτώμενη χαρτογράφηση. Δηλαδή, μία κλάση (η εξαρτώμενη) βασίζεται σε μία άλλη κλάση (τον ιδιοκτήτη) και κάθε εξαρτημένη κλάση πρέπει να έχει μόνο έναν ιδιοκτήτη. Οι εξαρτώμενες κλάσεις μπορούν να έχουν άλλη εξαρτώμενη κλάση από κάτω τους. Δομικά όλο αυτό καταλήγει σε μία αλληλουχία κλάσεων με εξαρτήσεις και όλο αυτό μπορεί να καταλήξει και σε μία κλάση ιδιοκτήτη, όπου για την εγγραφή και την αποθήκευση των εξαρτώμενων είναι υπεύθυνη η κλάση ιδιοκτήτης. Βασικό χαρακτηριστικό των εξαρτώμενων είναι ότι δεν έχουν πεδίο ταυτότητας και άρα δε μπορούν να χαρτογραφηθούν μέσω αυτής.

Πως λειτουργεί:

Η κύρια λειτουργία του μοτίβου είναι ότι σε κάθε περίπτωση που φορτώνεται μία κλάση ιδιοκτήτης θα φορτωθούν και οι εξαρτώμενες του. Επειδή όπως είπαμε πριν δεν έχουν ταυτότητα τα εξαρτώμενα αντικείμενα όλες τις μεθόδους εύρεσης τους έχει ο ιδιοκτήτης και εάν υπάρχει μεγαλύτερη αλυσίδα στην ιεραρχία των εξαρτώμενων τότε τα ευρήματα τα έχει ο “μεγαλύτερος” ιδιοκτήτης. Εάν υπάρξει κάποια ενημέρωση σε κάποιο εξαρτώμενο αντικείμενο τότε στο αντικείμενο ιδιοκτήτη γίνεται διαγραφή του παλιού αντικειμένου και επαναφόρτιση του νέου εξαρτώμενου. Φυσικά αναφορά σε μία εξαρτώμενη κλάση μπορεί να γίνεται μόνο από την κλάση ιδιοκτήτης.

Πλεονεκτήματα και μειονεκτήματα:

Το κέρδος του dependent mapping στο πρόγραμμα είναι ότι επιτυγχάνεται η οργάνωση σε μία βάση δεδομένων όπου κάποια αντικείμενα είναι φυσική συνέπεια κάποιων άλλων και δεν είναι ανεξάρτητες οντότητες. Οπότε εξυπηρετεί και την γενική χαρτογράφηση των αντικειμένων προσθέτοντας την πληροφορία της εξάρτησης.

Το κακό που μπορεί να προκαλέσει αυτό το μοτίβο είναι ότι μπορεί να επιβαρυνθεί πολύ μία κλάση ιδιοκτήτης με τον χειρισμό όλων των εξαρτώμενων κλάσεων που μπορεί να αποτελούν δική του συνέπεια.

Εφαρμογές:

Το μοτίβο αυτό εφαρμόζεται ιδανικά σε περιπτώσεις που ένα αντικείμενο έχει μια συλλογή εξαρτώμενων αντικειμένων που ο λόγος ύπαρξης τους είναι αυτό το αντικείμενο και μόνο.

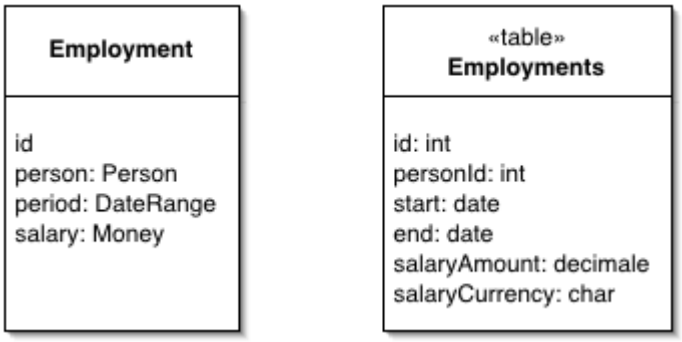
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.5 Embedded Value

Εμφανίζει ένα αντικείμενο σε διάφορα πεδία του πίνακα ενός άλλου αντικειμένου.



Σκοπός:

Ο σκοπός του embedded value είναι να εξοικονομήσει χώρο στη βάση δεδομένων από αντικείμενα που δε χρειάζεται να έχουν πίνακα στη βάση δεδομένων.

Δομή:

Δομικά είναι μία κλάση όπου έχει κάποια πεδία που χαρακτηρίζουν ένα αντικείμενο και έχουν μία ενσωματωμένη αξία. Ουσιαστικά είναι ένα μοντέλο σχεσιακής δομής που χαρτογραφεί μία κατάσταση αντικειμένου συνεργάτη στον πίνακα ενός άλλου αντικειμένου, το οποίο έχει άμεση αναφορά σε αυτό.

Πως λειτουργεί:

Το embedded value είναι μία ιδιαίτερη περίπτωση του dependent mapping. Λειτουργεί πάλι, κυρίως μέσω εξάρτησης. Όταν φορτώνεται ή αποθηκεύεται ένα κύριο αντικείμενο τότε γίνεται το ίδιο και με τα εξαρτώμενα αντικείμενα. Το αντικείμενο ιδιοκτήτης πάλι αναλαμβάνει όλες τις μεθόδους. Η διαφορά εδώ από το προηγούμενο μοτίβο είναι ότι η τιμή είναι ένα εξαρτώμενο αντικείμενο.

Πλεονεκτήματα και μειονεκτήματα:

Βασικό πλεονέκτημα είναι ότι όπως και πριν, κάνει πάλι τους πίνακες της βάσης δεδομένων να έχουν νόημα και να μη φορτώνονται με δεδομένα χωρίς νόημα.

Αρνητικό θα μπορούσε ομοίως να χαρακτηριστεί όπως και στο προηγούμενο μοτίβο η πιθανότητα επιβάρυνσης μίας κλάσης ιδιοκτήτης.

Εφαρμογές:

Η εφαρμογή του μπορεί να γίνει σε εφαρμογές που έχουν αντικείμενα με συγκεκριμένο εύρος απλών τιμών όπως τα χρήματα και οι ημερομηνίες.

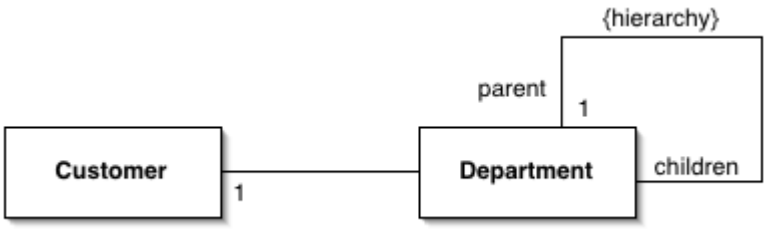
Βιβλιογραφία:

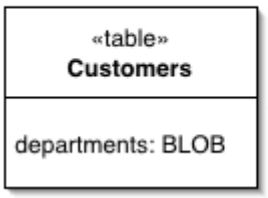
Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.6 Serialized LOB

Αποθηκεύει ένα γράφημα αντικειμένων με τη σειριοποίηση τους σε ένα μεγάλο αντικείμενο (LOB), το οποίο αποθηκεύεται σε ένα πεδίο βάσης δεδομένων.





Σκοπός:

Ο σκοπός του μοτίβου είναι να διατηρήσει ένα γραφικό αντικείμενο ως ενιαίο μεγάλο αντικείμενο, αντί να το σπάσει σε ομοιογενή κομμάτια για αποθήκευση σε ξεχωριστούς πίνακες συσχετισμού. Είναι μία προσπάθεια μέσω των ιεραρχιών που επικρατούν να κατασκευαστεί ένα μοντέλο όπου περιέχει τις σχέσεις των αντικειμένων σε ένα συνολικό σχεσιακό σχήμα.

Δομή:

Η δομή του είναι ένα σειριοποιημένο γράφημα, οπότε δομικά δεν είναι κάτι συγκεκριμένο και αυτό προκαλεί και αρκετά προβλήματα.

Πως λειτουργεί:

Για την υλοποίηση του υπάρχουν δύο τρόποι σειριοποίησης: ο δυαδικός τρόπος (BLOB) και ο τρόπος της συμβολοσειράς (CLOB). Ο πρώτος τρόπος θεωρείται εύκολος προγραμματιστικά να υλοποιηθεί και βολικός όσων αφορά τον χώρο καθώς χρησιμοποιεί πολύ λίγο χώρο στη βάση δεδομένων. Όλα αυτά όμως εφόσον υποστηρίζεται από τη βάση δεδομένων ο δυαδικός τύπος. Ο δεύτερος τρόπος είναι πιο εύκολος στην ανάγνωση λόγω της συμβολοσειράς αλλά ενδεχομένως χρησιμοποιεί περισσότερο χώρο απ' ότι ο προηγούμενος τρόπος. Το CLOB σε συνεργασία με την XML είναι αποδοτικότερος.

Πλεονεκτήματα και μειονεκτήματα:

Πλεονέκτημα αυτού του μοτίβου είναι ότι εφόσον υλοποιηθεί σωστά και με την κατάλληλη σειριοποίηση μπορεί να βοηθήσει πολύ στη σχεσιακή οργάνωση της βάσης δεδομένων.

Αρνητικά του serialized LOB είναι ότι υπάρχει περίπτωση να δημιουργήσει προβλήματα απόδοσης και συνέπειας στην εφαρμογή μέσω ενός μεγάλου γράφου. Επίσης ένα κύριο μειονέκτημα είναι ότι δε γίνεται να ζητηθεί η δομή μέσω ερωτημάτων SQL.

Εφαρμογές:

Δεν είναι πολυεφαρμοσμένο το συγκεκριμένο μοτίβο. Οι εφαρμογές που θα μπορούσε να υποστηρίξει όμως, είναι αυτές που χρησιμοποιούν XML διότι διευκολύνουν πολύ την υλοποίηση του καθώς και αυτές που θα μπορούσαν να υποστηρίξουν ξεχωριστή βάση δεδομένων αναφορών.

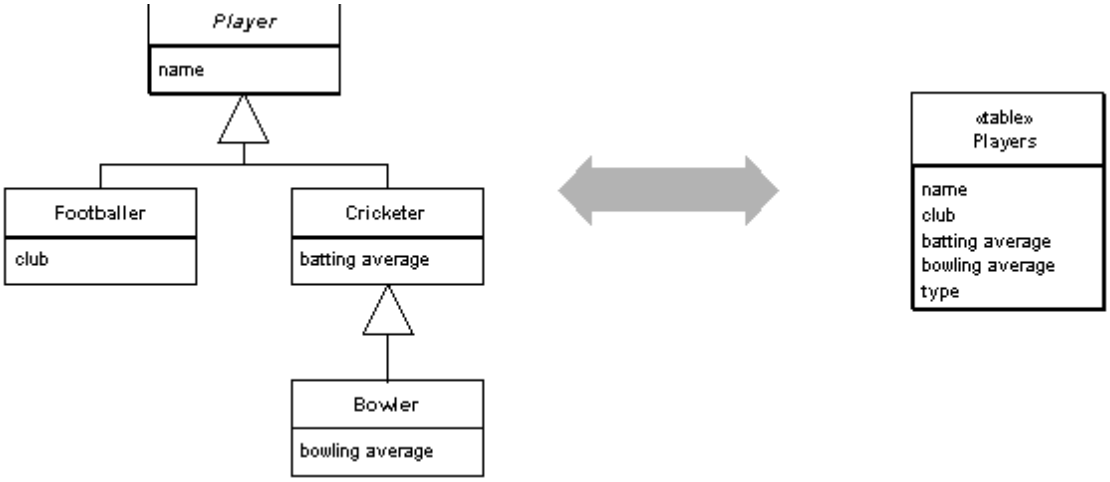
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.7 Single Table Inheritance

Αντιπροσωπεύει μια ιεραρχία κληρονομιών των κλάσεων ως ένα μόνο πίνακα που έχει στήλες για όλα τα πεδία των διαφόρων κλάσεων.



Σκοπός:

Σκοπός του single table inheritance είναι να λύσει την αδυναμία κληρονομικότητας σε μία σχεσιακή βάση δεδομένων.

Δομή:

Η δομή του είναι μία κλάση όπου είναι ένας πίνακας που περιέχει μία κύρια κλάση, η οποία περιέχει κλάσεις παιδιά που την κληρονομούν και χαρακτηρίζουν την κύρια κλάση με μία πιο εξειδικευμένη ιδιότητα ή συμπεριφορά. Σε αυτό το πίνακα περιέχει όλα τα δεδομένα για όλες τις κλάσεις στην ιεραρχία κληρονομιών. Η κλάση που περιέχει αυτό το πίνακα ονομάζεται κλάση κληρονομιάς.

Πως λειτουργεί:

Στην κλάση κληρονομιάς αποθηκεύονται σχετικά δεδομένα από κάθε κλάση. Η λειτουργία της χαρτογράφησης κατά τη φόρτωση ενός αντικειμένου στη μνήμη πρέπει να γνωρίζει σε ποιά κλάση κληρονομιάς ανήκει μέσω ενός πεδίου στον πίνακα όπου γνωρίζει αυτή τη πληροφορία και συνήθως (όπως φαίνεται και στο παραπάνω σχήμα) το πεδίο αυτό ονομάζεται “type”. Εφόσον γίνεται η αντιστοίχιση μπορεί να εκτελεστούν οι εκάστοτε λειτουργίες της κλάσης για το υποκείμενο αντικείμενο.

Πλεονεκτήματα και μειονεκτήματα:

Βασικό πλεονέκτημα του μοτίβου είναι ότι υπάρχει μόνο ένας πίνακας για τη βάση δεδομένων. Επίσης, λειτουργεί θετικά για την ανάκτηση δεδομένων διότι δεν υπάρχουν συνδέσεις και ακόμα δεν υπάρχει εξάρτηση από τη βάση δεδομένων σε οποιοδήποτε επαναπροσδιορισμό στην ιεραρχία.

Το βασικό μειονέκτημα του single table inheritance είναι ότι υπάρχει περίπτωση να επιβαρύνει την απόδοση του προγράμματος διότι ο πίνακας μπορεί να καταλήξει υπερβολικά μεγάλος με πολλούς δείκτες και συχνά κλειδωμένος. Επιπλέον μπορεί να υπάρξει σύγχυση με το πεδίο προσδιορισμού της κύριας κλάσης, είτε με τη μοναδικότητα του ονόματος είτε με την ευκολία κατανόησης του προσδιορισμού του ονόματος.

Εφαρμογές:

Το μοτίβο συνηθίζει να εφαρμόζεται για τη χαρτογράφηση των πεδίων σε μια κληρονομική ιεραρχία μιας σχεσιακή βάση δεδομένων. Εφαρμογές που εμπεριέχουν έντονα την κληρονομικότητα και την ιεραρχία υποστηρίζονται ιδανικά μέσω του single table inheritance για να μην γίνει μεγάλη επιβάρυνση στην βάση δεδομένων.

Βιβλιογραφία:

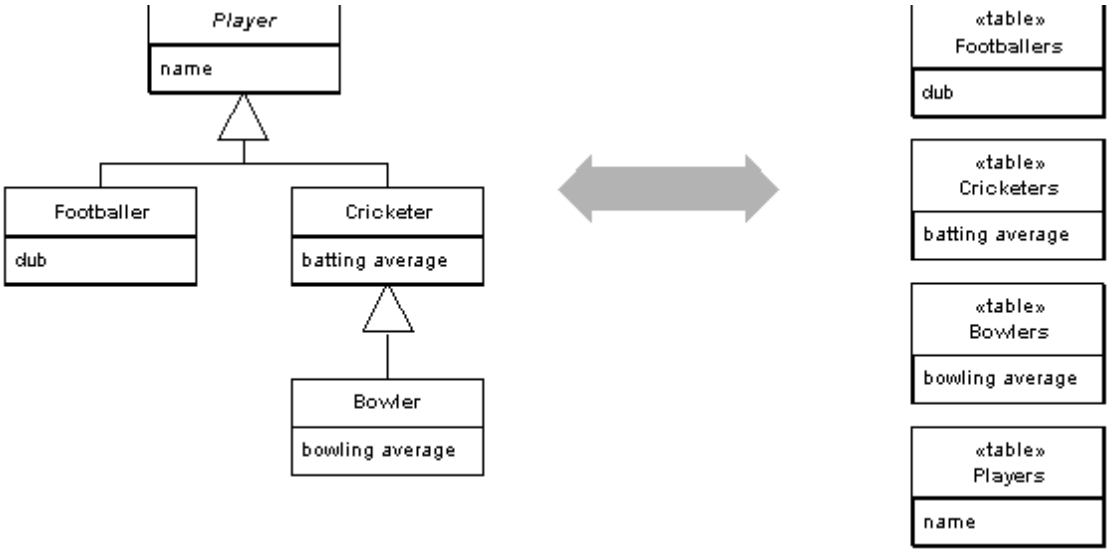
Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**,

Link
https://docs.jboss.org/hibernate/orm/3.5/reference/en/html/inheritance.html
about ***Hibernate ORM documentation (5.0), Chapter 9. Inheritance mapping*, Retrieved November 3, 2015.**

3.4.8 Class Table Inheritance

Αντιπροσωπεύει μια ιεραρχία κληρονομιών των κλάσεων με ένα πίνακα για κάθε κλάση.



Σκοπός:

Σκοπός του μοτίβου είναι μέσω ενός πίνακα σε κάθε κλάση στην ιεραρχία να αντιστοιχίσει απευθείας τα πεδία της κλάσης με τα πεδία του πίνακα. Άρα μέσω πινάκων υπάρχει η φυσική χαρτογράφηση της βάσης δεδομένων.

Δομή:

Δομικά, προφανώς είναι ένα σύνολο πινάκων που χαρτογραφούν την ιεραρχία των κλάσεων που εκπροσωπούν τα αντικείμενα της σχεσιακής βάσης δεδομένων.

Πως λειτουργεί:

Η λειτουργία του μοιάζει απλή αφού χαρτογραφεί τα αντικείμενα μέσω ενός πίνακα σε κάθε κλάση. Η σύνδεση του όμως με τη βάση δεδομένων, για κάθε πεδίο που μπορεί να χρειαστεί να εκτελέσει κάποια ενέργεια, μοιάζει να είναι μεγάλη επιβάρυνση για τη βάση δεδομένων εάν πρέπει να συνδεθεί για πολλά αντίστοιχα πεδία. Ο τρόπος είναι να χρησιμοποιεί το πρωτεύον κλειδί των αντικειμένων για την αντιστοίχιση τους με τη βάση δεδομένων αλλά αυτό σίγουρα θα μειώσει κατά πολύ την απόδοση εάν γίνουν ενέργειες φόρτωσης ή αποθήκευσης σε πολλά μαζί γιατί αμέσως σημαίνει και πολλές συνδέσεις στη βάση δεδομένων. Εάν γίνει κάποια ένωση μεταξύ κάποιων υποκείμενων πινάκων, ώστε να γίνουν οι ενέργειες στη βάση δεδομένων με μία κλήση, ίσως αποτελέσει κάποια λύση στην επιβάρυνση της απόδοσης.

Πλεονεκτήματα και μειονεκτήματα:

Το βασικό χαρακτηριστικό που χαρακτηρίζεται ως θετικό του μοτίβου είναι ότι η σχέση μεταξύ του μοντέλου του αντικειμένου, μέσω της χαρτογράφησης των πινάκων, με τη βάση δεδομένων κάθε φορά είναι πολύ αντιπροσωπευτική της

πραγματικής κατάστασης των αντικειμένων. Επίσης, οι πίνακες με αυτό τον τρόπο είναι κατανοητοί λόγω της καλής συσχέτισης των στηλών τους με της αντίστοιχης σειράς που εκπροσωπούν.

Το μεγαλύτερο μειονέκτημα του class table inheritance είναι ότι για να φορτωθεί ένα ολοκληρωμένο αντικείμενο πρέπει να γίνουν πολλά ερωτήματα στη βάση και σύνδεση στη μνήμη. Επιπλέον, εδώ ο επαναπροσδιορισμός της ιεραρχίας προκαλεί αλλαγές και στη βάση δεδομένων.

Εφαρμογές:

Μπορεί να αποτελέσει καλή εφαρμογή για τη χαρτογράφηση της ιεραρχικής κληρονομιάς για τις κλάσεις που βρίσκονται στην κορυφή της αλλά όχι για όλες τις κλάσεις και ιδιαίτερα όταν είναι πολλές οι χαμηλότερες στην ιεραρχία.

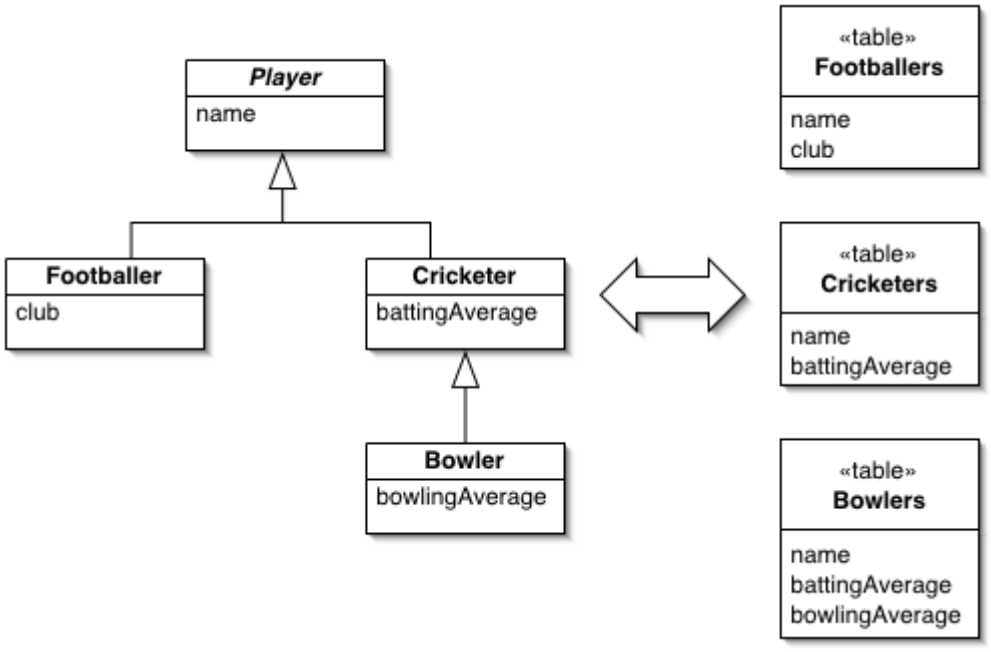
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.9 Concrete Table Inheritance

Αντιπροσωπεύει μια ιεραρχία κληρονομιών των κλάσεων με ένα πίνακα ανά συμπαγή κλάση στην ιεραρχία.



Σκοπός:

Το concrete table inheritance επιδιώκει και αυτό να αποτελέσει μία στρατηγική αντιμετώπισης της κληρονομιάς στην ιεραρχία των κλάσεων όπου δε μπορεί να το κάνει η βάση δεδομένων.

Δομή:

Η δομή του είναι η δημιουργία πινάκων για κλάσεις που είναι συμπαγής, δηλαδή που μπορούν να εκπροσωπήσουν μία ολοκληρωμένη οντότητα.

Πως λειτουργεί:

Κάθε κλάση στο συμπαγές πλέγμα χρησιμοποιεί ένα πίνακα από τη βάση δεδομένων. Ο πίνακας που αντιπροσωπεύει μία συμπαγής κλάση περιέχει πεδία της συγκεκριμένης κλάσης αλλά και όλων των προγόνων του. Έτσι τα πεδία κάθε κλάσης είτε κληρονομούνται στα παιδία είτε έχουν κληρονομηθεί από τους προγόνους του ,εκτός των πεδίων που χαρακτηρίζουν την ίδια κλάση. Στην λειτουργία του ένα βασικό σημείο είναι η χρήση των κλειδιών που γίνεται για την αντιστοίχιση των πεδίων άλλων κλάσεων διότι πρέπει να είναι μοναδικά. Εάν χρησιμοποιηθούν τα πρωτεύοντα κλειδιά μπορεί να δημιουργηθούν προβλήματα σε περίπτωση που τα χρησιμοποιούν παραπάνω από μία κλάση. Οπότε πρέπει να φτιάχνεται μοναδικό κλειδί κάθε φορά. Εν κατακλείδι λειτουργεί σαν χαρτογράφος κληρονομιών για διευκόλυνση της βάσης δεδομένων.

Πλεονεκτήματα και μειονεκτήματα:

Ένα κύριο θετικό του μοτίβου είναι ότι εάν ζητηθεί χρήση των αντικειμένων από άλλες εφαρμογές μπορεί να γίνει διότι κάθε πίνακας είναι αυτοτελής, χωρίς άσχετα πεδία και μπορεί να εκπροσωπήσει μία ολοκληρωμένη οντότητα. Επιπλέον, για την ανάγνωση δεδομένων στα αντικείμενα δεν χρειάζεται να γίνει κάποια σύνδεση στη βάση δεδομένων και έτσι δεν επιβαρύνεται η απόδοση του προγράμματος.

Δύο είναι τα κύρια μειονεκτήματα του concrete table inheritance. Το πρώτο είναι ότι η χρήση των πρωτευόντων κλειδιών είναι δύσκολη, σε σημείο να προκαλέσει σοβαρά προβλήματα στις συνδέσεις. Και το δεύτερο, ότι δεν μπορεί να υποστηρίξει τις αφηρημένες κλάσεις που μπορεί να έχουν συσχετίσεις σε μία βάση δεδομένων.

Εφαρμογές:

Το μοτίβο αυτό είναι ιδανικό για εφαρμογές όπου χρησιμοποιούν συγκεκριμένες κλάσεις με κληρονομικότητα και ιεραρχία, χωρίς αφηρημένες κλάσεις καθώς και όταν υπάρχει ανάγκη χρήσης των αντικειμένων σε παραπάνω από μία εφαρμογές του συστήματος.

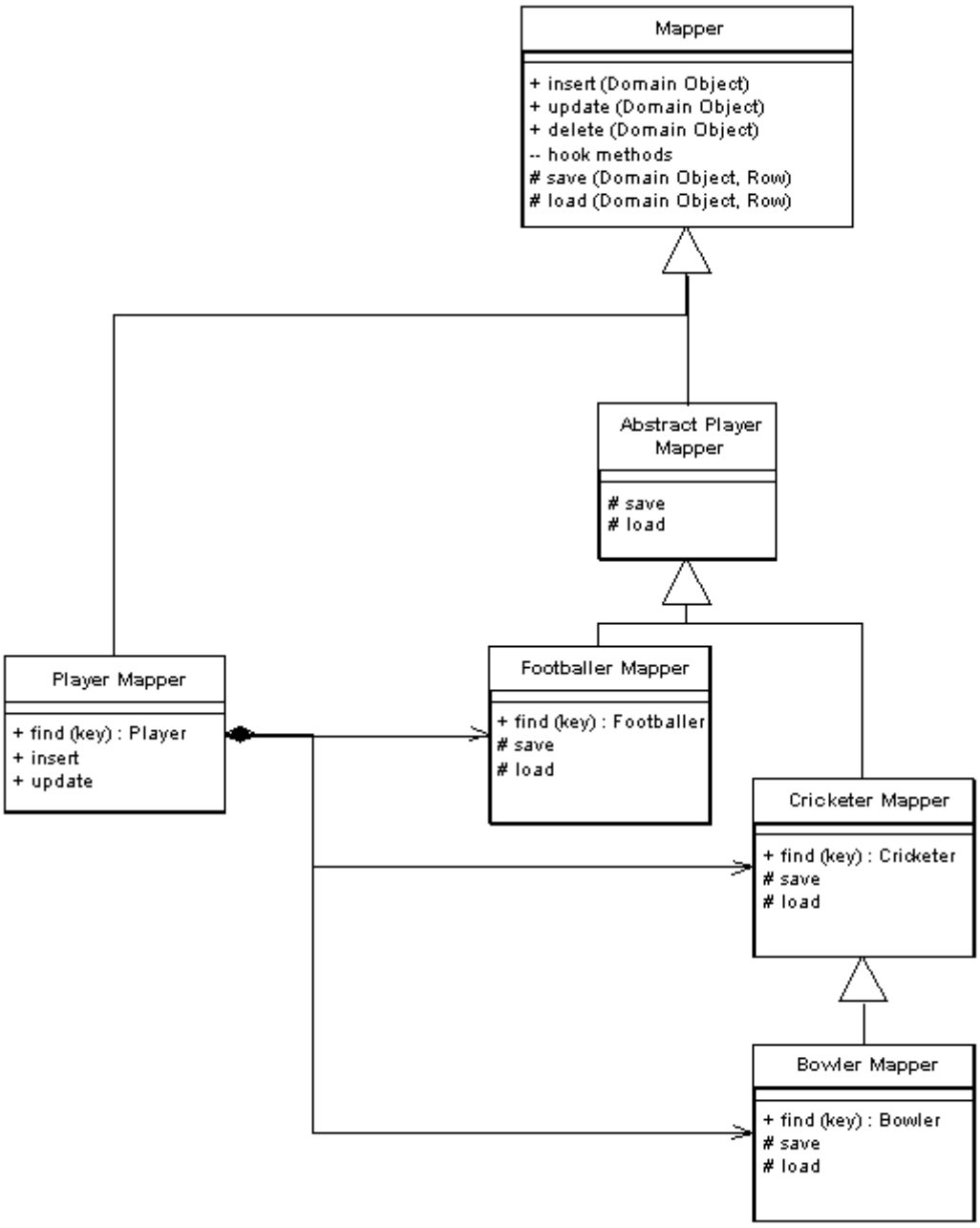
Βιβλιογραφία:

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.4.10 Inheritance Mappers

Μια δομή για την οργάνωση των χαρτογράφων βάσης δεδομένων που χειρίζονται ιεραρχίες κληρονομιάς.



Σκοπός:

Σκοπός αυτού του μοτίβου χαρτογράφησης είναι να χαρτογραφήσει με μία γενική δομή, τους χαρτογράφους που χειρίζονται τις κληρονομιές ιεραρχίας ώστε να ελαχιστοποιηθεί το ποσό του κώδικα που απαιτείται για την φόρτωση ή αποθήκευση δεδομένων στη βάση δεδομένων.

Δομή:

Η δομή του είναι ένας χάρτης όπου ιεραρχεί τους προηγούμενους χαρτογράφους που περιλαμβάνει και αφηρημένες και μη αφηρημένες κλάσεις.

Πως λειτουργεί:

Η λειτουργία του είναι να ιεραρχήσει τους χαρτογράφους ώστε να υπάρχει ένας χάρτης για κάθε κλάση αντικειμένου domain όπου χειρίζεται την αποθήκευση και τη φόρτωση δεδομένων σε αυτή. Στις μεθόδους της υπερκλάσης υπάρχουν οι λειτουργίες της εύρεσης, της εισαγωγής, της ενημέρωσης και της διαγραφής αντικειμένων. Τις λειτουργίες αυτές μπορεί να τις μοιράσει σε υποκλάσεις. Τις λειτουργίες της εισαγωγής και της ενημέρωσης τις κληρονομεί σε μη αφηρημένη υποκλάση ενώ σε μία αφηρημένη υποκλάση μπορεί να δώσει τη δυνατότητα φόρτωσης ή αποθήκευσης κάποιου αντικειμένου, αλλά εφόσον κληρονομηθούν και στις υποκλάσεις αυτής, με αποτέλεσμα μία μέθοδος εύρεσης σε μία υποκλάση (συμπαγής, δηλαδή που έχει κληρονομήσει πεδία από μία υπερκλάση της και αυτή το ίδιο μέχρι την κύρια κλάση της ιεραρχίας) να μπορεί να επιστρέψει ένα αντικείμενο με νόημα και όχι κάτι αφηρημένο.

Πλεονεκτήματα και μειονεκτήματα:

Το κύριο πλεονέκτημα του είναι ότι με την οργάνωση των χαρτογράφων μπορεί να βοηθήσει την απόδοση του προγράμματος όπως και τις συνδέσεις με τη βάση δεδομένων. Επίσης ένα θετικό είναι ότι υποστηρίζει και μία σχετική αφηρημένη χαρτογράφηση κλάσεων.

Σαν αρνητικό αυτού του μοτίβου θα μπορούσε να χαρακτηριστεί ότι κάποιες φορές θα μπορούσε να μη χρειαστεί, εάν κάποιος χαρτογράφος (από τους προηγούμενους που περιγράφηκαν) μπορεί να λειτουργήσει αποδοτικά από μόνος του, που σε μία τέτοια περίπτωση δεν είναι βέβαιο ότι θα λειτουργήσει επιβαρυντικά.

Εφαρμογές:

Η εφαρμογή του inheritance mappers μπορεί να λάβει μέρος σχεδόν σε όλες τις περιπτώσεις που έχουν ιεραρχική κληρονομιά, γιατί συμβάλλει θετικά στην γενική δομή της χαρτογράφησης σχεδόν όλων των περιπτώσεων.

Βιβλιογραφία:

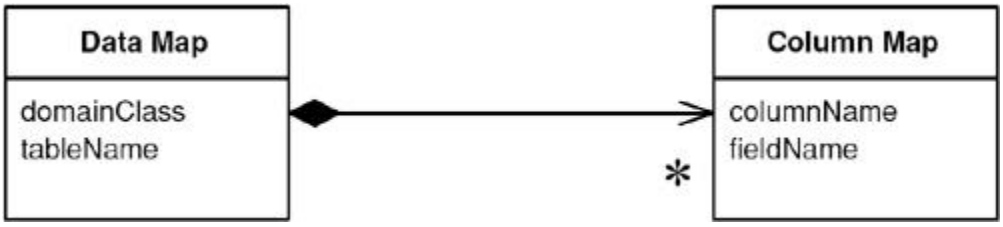
Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Jognson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.5 Object-Relational Metadata Mapping Patterns

3.5.1 Metadata Mapping

Διατηρεί λεπτομέρειες σχετικά με τη σχεσιακή αντιστοίχιση χαρτογραφήσεων σε μεταδεδομένα.



Σκοπός:

Ο σκοπός του μοτίβου είναι να εκφράσει λεπτομέρειες της εφαρμογής, οι οποίες σχετίζονται με συγκεκριμένο domain και domain model(3.1.2), ως μεταδεδομένα μίας γενικού σκοπού βιβλιοθήκης. Τα μεταδεδομένα (δεδομένα των δεδομένων) σχετίζονται με τις έμμονες λειτουργίες, δηλαδή όσες λειτουργίες αφορούν μεταφορά αντικειμένων από και προς μία βάση δεδομένων (λειτουργίες ανάγνωσης, εισαγωγής και ενημέρωσης).

Δομή:

Η δομή του metadata mapping είναι μια απλή πινακοειδή μορφή όπου ορίζει τις αντιστοιχίες των αντικειμένων της μνήμης με μία σχεσιακή βάση δεδομένων.

Πως λειτουργεί:

Η κύρια λειτουργία της χαρτογράφησης αυτής που ορίζει τις αντιστοιχίες είναι ότι μπορεί να επεξεργαστεί με τον γενικό κώδικα να πραγματοποιηθούν οι λεπτομέρειες των λειτουργιών (ανάγνωσης, εισαγωγής και ενημέρωσης δεδομένων). Αυτό μπορεί να πραγματοποιηθεί με τη γραφή ενός προγράμματος του οποίου η είσοδος είναι τα μεταδεδομένα και του οποίου η έξοδος είναι ο πηγαίος κώδικας των κλάσεων που κάνουν τη χαρτογράφηση. Με αυτό το τρόπο, οι κλάσεις χαρτών πρέπει να είναι ενσωματωμένες κατά τη διαδικασία κατασκευής

της εφαρμογής για να γίνουν οι κατάλληλες αντιστοιχίσεις με όλα τα σενάρια. Ένας άλλος τρόπος υλοποίησης είναι το ανακλαστικό πρόγραμμα. Σε αυτή τη περίπτωση το πρόγραμμα μπορεί να μετατρέπει το ζητούμενο αντικείμενο σε κατάλληλο όρισμα για τη μέθοδο μέσω ενσωματωμένης δικής του μεθόδου πάνω σε μία μέθοδο όπου χειρίζεται ένα αντικείμενο της βάσης δεδομένων. Έτσι ουσιαστικά, μπορεί να κάνει τη χαρτογράφηση αφού έχει τη δυνατότητα να αντλήσει πεδία και μεθόδους των μεταδεδομένων.

Πλεονεκτήματα και μειονεκτήματα:

Το βασικό πλεονέκτημα αυτού του μοτίβου είναι ότι ο κώδικας που περιγράφει τις αντιστοιχίες μεταξύ πεδίων αντικειμένων στη μνήμη και πεδίων στη βάση δεδομένων δε γίνεται επαναλαμβανόμενος και κουραστικός.

Ένα μειονέκτημα του metadata mapping είναι ότι μπορεί λόγω μετονομασιών μεταδεδομένων να οδηγηθεί το πρόγραμμα σε εξαιρέσεις και σφάλματα όπου αυτό επηρεάζει αρνητικά.

Εφαρμογές:

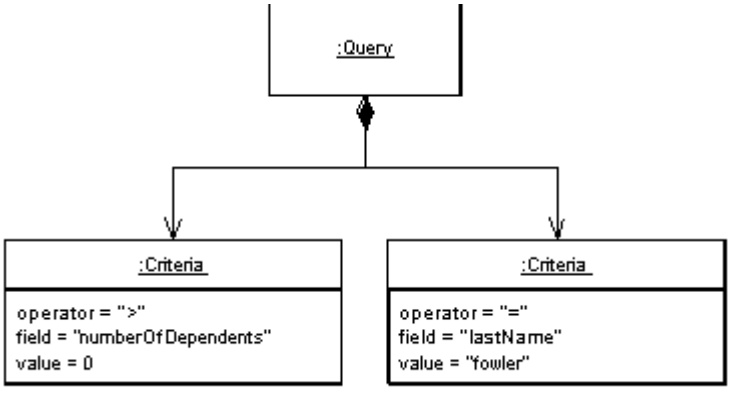
Η εφαρμογή του είναι κατάλληλη για όσες εφαρμογές χρειάζονται χαρτογράφηση βάσης δεδομένων και έχει γίνει πλέον τόσο συχνή και απαραίτητη όπου τα εμπορικά προϊόντα που παράγουν ένα εξελιγμένο μεταδεδομένο χρησιμοποιούν metadata mapping για αντικειμενοσχεσιακά εργαλεία χαρτογράφησης.

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.5.2 Query Object

Ένα αντικείμενο που αντιπροσωπεύει ένα ερώτημα βάσης δεδομένων.



Σκοπός:

Ο σκοπός του μοτίβου είναι να δημιουργήσει ένα αντικείμενο όπου μπορεί να μετατρέψει ένα ερώτημα πελάτη χωρίς να είναι αυστηρά διατυπωμένο ως ερώτημα SQL. Δηλαδή αυτό το αντικείμενο έχει στόχο να μετατρέψει ένα ερώτημα διατυπωμένο σε γενικό πλαίσιο ή ένα ερώτημα που δε βασίζεται στο σχήμα δομής της βάσης δεδομένων σε ένα ερώτημα SQL για τη βάση δεδομένων.

Δομή:

Η δομή του query object είναι μία κλάση όπου αντιπροσωπεύει ένα ερώτημα SQL και μπορεί να δεχτεί σαν αντικείμενο αναζήτησης κλάσης και πεδία των αντικειμένων που είναι στη μνήμη.

Πως λειτουργεί:

Η κύρια λειτουργία του είναι να αποτελέσει διερμηνέας για ερωτήματα SQL της βάσης δεδομένων από γενικού τύπου ερωτήματα. Το μοτίβο αυτό μπορεί να χρησιμοποιήσει ονόματα αντικειμένων και πεδίων όπου είναι στη μνήμη και όχι απαραίτητα πίνακες και στήλες. Έτσι παίρνει ένα ερώτημα όπου απευθύνεται σε αυτά και με ένα χειρισμό αναζήτησης του υποκείμενου αντικειμένου δημιουργεί το αντίστοιχο κατάλληλο ερώτημα για τη βάση δεδομένων. Αυτό επιτυγχάνεται με κατάλληλες παραμετροποιημένες μεθόδους εντός της κλάσης. Μπορεί να μην καλύπτει όλο το εύρος δυνατοτήτων χρήσης της SQL αλλά καλύπτει ένα μεγάλο και ικανοποιητικό κομμάτι της.

Πλεονεκτήματα και μειονεκτήματα:

Το θετικό αντίκτυπο αυτού είναι ότι μπορεί να υποστηρίξει πολλαπλές βάσεις δεδομένα και πολλαπλά σχήματα καθώς και την αποφυγή πολλαπλών ερωτημάτων. Επίσης μπορεί να αντιμετωπίσει ερωτήματα που απευθύνονται σε κλάσεις αντικειμένων και πεδία με αποτέλεσμα να προσφέρει ευελιξία.

Το μειονέκτημα του είναι ότι η δημιουργία αυτού του μοτίβου δεν είναι απλή. Είναι αρκετά περίπλοκη και απαιτεί πολύ καλή αντίληψη ενός προγραμματιστή για να διαχειριστεί όλο το εύρος των ερωτημάτων.

Εφαρμογές:

Για όσες εφαρμογές χρειάζονται την υποστήριξη όσων αναφέρονται στα πλεονεκτήματα του μοτίβου είναι ένα απαραίτητο μοτίβο για το κατάλληλο στήσιμο τους. Στις μόνες περιπτώσεις που θα μπορούσε να μην είναι χρήσιμο είναι σε αυτές που τα αντικείμενα και η βάση δεδομένων έχουν την ίδια δομή. Επίσης, είναι

χρήσιμο μοτίβο όταν χρησιμοποιούνται domain model(3.1.2) και data mapper(3.2.4).

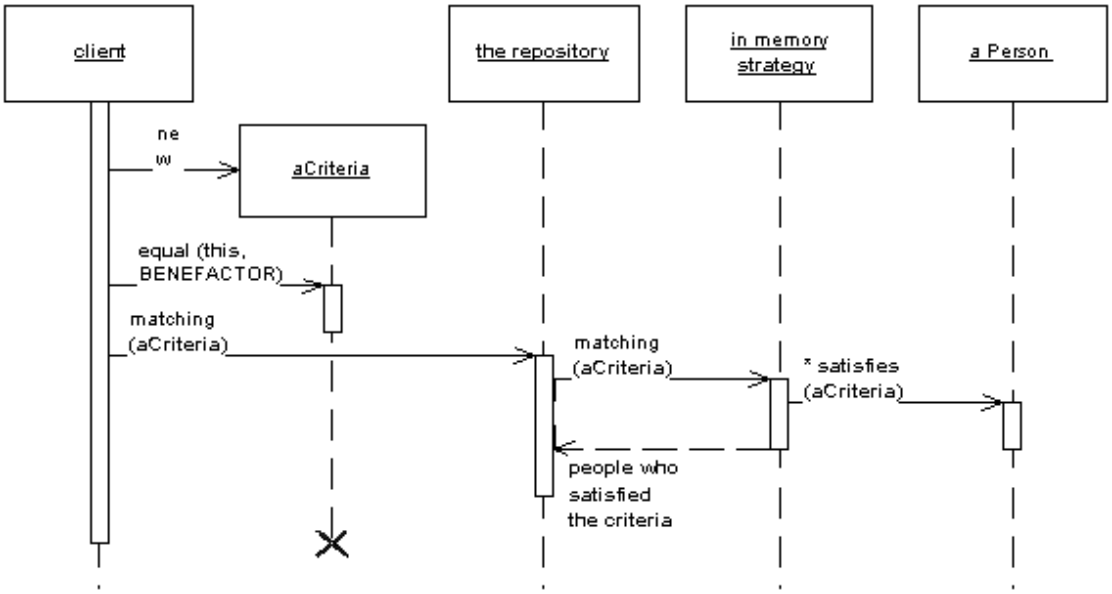
Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

.

3.5.3 Repository (by Edward Hieatt and Rob Mee)

Μεσολαβεί μεταξύ domain και στρωμάτων χαρτογράφησης δεδομένων χρησιμοποιώντας μία συλλογή σαν διεπαφή για την πρόσβαση σε αντικείμενα του domain.



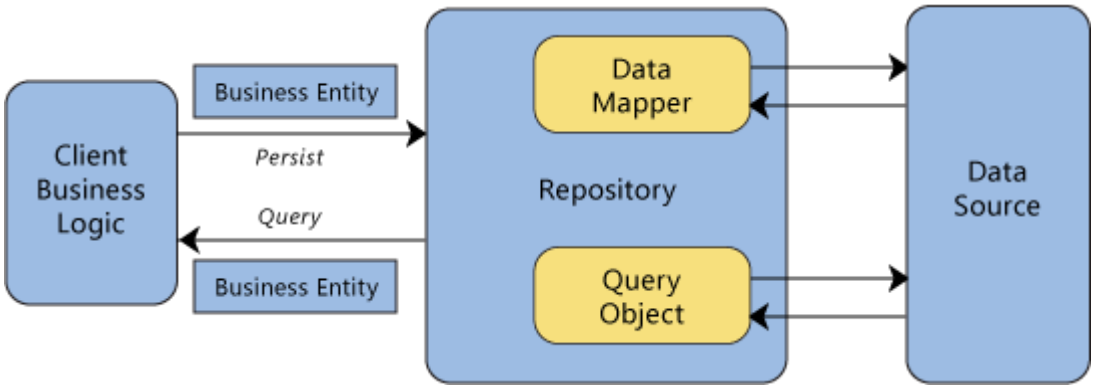
Σκοπός:

Ο γενικότερος σκοπός του repository είναι να αποτελέσει μεσολαβητής μεταξύ των στρωμάτων επιχειρησιακής λογικής και των στρωμάτων πηγών δεδομένων. Παρέχοντας ένα ξεχωριστό χώρο ως συλλογή αντικειμένων για τη διαχείριση των ερωτημάτων, τη χαρτογράφηση και την αλλαγή των δεδομένων επιτυγχάνει το στόχο του όπου είναι η αποφυγή διπλών ερωτημάτων, ουσιαστικότερη αντικειμενοστρέφεια και συγκέντρωση του κώδικα.

Δομή:

Η δομή του είναι μία κλάση όπου λειτουργεί σαν στρώμα ή αποθήκη πάνω από ή για τις λειτουργίες χαρτογράφησης (data mapper(3.2.4)) και διαχείρισης ερωτημάτων (query object(3.5.2)).

Σχήμα: Γενική απεικόνιση του συνδεσμολογικού ρόλου του repository.



Πως λειτουργεί:

Το repository λειτουργεί σαν μια συλλογή αντικειμένων που μεσολαβεί για την καλύτερη αλληλεπίδραση της βασικής στρατηγικής για τον πελάτη της επιχειρηματικής εφαρμογής με τον χώρο των δεδομένων. Οι λειτουργίες είναι η διαχείριση των πολλαπλών ερωτημάτων, η καλύτερη χαρτογράφηση ενός χαρτογράφου που συμπεριλαμβάνεται μέσα σε αυτή τη συλλογή αντικειμένων και ακόμα οι μέθοδοι που αφορούν τα αντικείμενα της συλλογής, όλα είναι μέσα σε αυτό το χώρο που δημιουργεί αυτό το μοτίβο. Θα μπορούσε να χαρακτηριστεί σαν ένας αποθηκευτικός χώρος που αποθηκεύει εκεί τα αντικείμενα, τους χάρτες, τα ερωτήματα και τις μεθόδους των αντικειμένων. Μπορεί να εκτελέσει ακόμα και τις λειτουργίες των αντικειμένων για αποθήκευση, ενημέρωση, αναζήτηση ή φόρτωση αντικειμένων από μία βάση δεδομένων. Ουσιαστικά απομονώνει και συγκεντρώνει μεγάλο μέρος του κώδικα του προγράμματος σε έναν αντικειμενοστραφή χώρο.

Πλεονεκτήματα και μειονεκτήματα:

Το κύριο πλεονέκτημα του είναι ότι εκμεταλλεύεται έναν χώρο για την αποδοτικότερη λειτουργία της εφαρμογής μέσω της οργάνωσης όλων των λειτουργιών που παρέχει. Επίσης, παρέχει ένα χώρο όπου μπορούν να εκτελεστούν δοκιμές σε ενέργειες χωρίς να προκαλούν σύγχυση. Τέλος, η αρχιτεκτονική του είναι ευέλικτη και μπορεί να προσαρμοστεί σε εξελίξεις της εφαρμογής.

Σαν αρνητικό του repository μπορεί να τεθεί το ερώτημα ότι εάν είναι πάντα απαραίτητο, όπου ουσιαστικά η απάντηση εξαρτάται από τις απαιτήσεις της εφαρμογής και το βάρος που μπορεί να κουβαλάει όλο το πρόγραμμα.

Εφαρμογές:

Οι εφαρμογές που υποστηρίζονται κατάλληλα από αυτό το μοτίβο είναι κυρίως αυτές που χρησιμοποιούν μεγάλο αριθμό κλάσεων στο domain και ερωτήματα με βαρύτητα ή συνεχής ροής καθώς και σύστημα πολλαπλών χώρων δεδομένων.

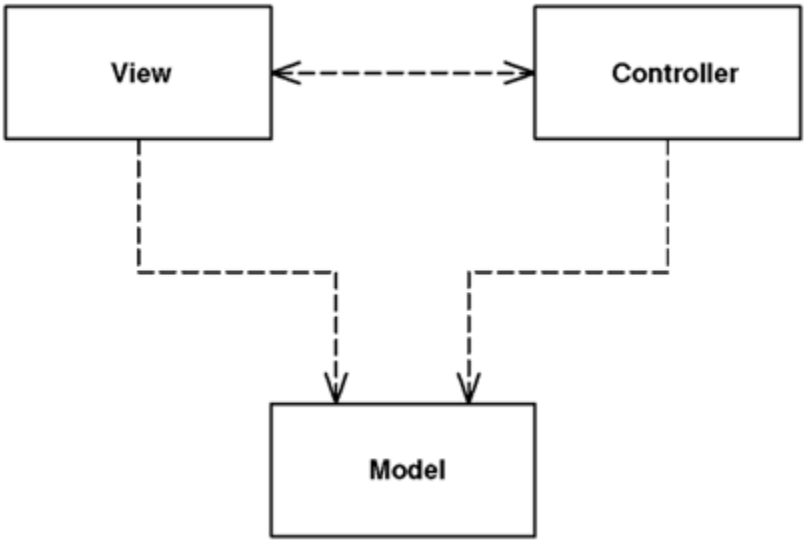
Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.6 Web Presentation Patterns

3.6.1 Model View Controller

Διαχωρίζει την αλληλεπίδραση διεπαφής χρήστη σε τρεις ξεχωριστούς ρόλους.



Σκοπός:

Ο σκοπός του model view controller είναι να διευκολύνει τη δημιουργία εφαρμογών που απαιτούν μεγάλη αλληλεπίδραση μεταξύ της διεπαφής και του χρήστη.

Δομή:

Η δομή του χωρίζεται σε 3 διασυνδεδεμένες κλάσεις ή μέρη όπου χειρίζονται κάτι ξεχωριστό και ενημερώνει η μία την άλλη όπου χρειαστεί να συμβεί:

- Model – Δομικά είναι ένα αντικείμενο όπου επικοινωνεί με τον αποθηκευτικό χώρο δεδομένων, συνήθως μία βάση δεδομένων. Εκτελεί εντολές για ανάκτηση ή αποθήκευση δεδομένων όπως ακόμα και εντολές για την ενημέρωση αντικειμένων στη βάση δεδομένων.

- View – Αυτό το μέρος είναι ουσιαστικά η γραφική αναπαράσταση των πληροφοριών που περιέχει το model. Μια φιλική παρουσίαση των δεδομένων στον χρήστη.
- Controller – Είναι κατά κάποιο τρόπο ο μεσολαβητής του model και του view. Ανάλογα τις ενέργειες του χρήστη ενημερώνει και στέλνει εντολές στο model και επίσης στέλνει εντολές στο view ώστε να γίνει η κατάλληλη αντιστοίχιση γραφικής απεικόνισης για την εκάστοτε κατάσταση.

Πως λειτουργεί:

Στη δομή του μοτίβου αυτού έγινε μία μερική παρουσίαση της λειτουργίας του. Θα γίνει πάλι περιγραφή της όλης λειτουργίας του. Βασικό χαρακτηριστικό του είναι η διαχώριση των τριών διαφορετικών ρόλων. Στο model εμπεριέχεται όλη η επιχειρησιακή λογική του domain. Επίσης, είναι το κομμάτι όπου διαχειρίζεται την επικοινωνία με τη βάση δεδομένων και συμμετέχει στην επικοινωνία με το controller ώστε να μπορεί να γίνει η σωστή αλληλεπίδραση με τον χρήστη. Στη σωστή αλληλεπίδραση φυσικά, παίζει ρόλο και η σωστή ενημέρωση του controller και στο model και στο view, για να ανακτώνται ή να αποθηκεύονται ή να ενημερώνονται τα σωστά αντικείμενα και για να απεικονίζεται στον χρήστη η σωστή γραφική αναπαράσταση αντίστοιχα. Τέλος, το view είναι υπεύθυνο για την γραφική απεικόνιση των δεδομένων του model, παρουσιάζει μία φιλική διεπαφή προς τον χρήστη ώστε να μπορεί να γίνει αποτελεσματικά η αλληλεπίδραση με το χρήστη.

Πλεονεκτήματα και μειονεκτήματα:

Βασικά πλεονεκτήματα του model view controller είναι ότι δίνει την ευκολία στους προγραμματιστές να κάνουν ταυτόχρονη ανάπτυξη σε αυτά τα 3 μέρη χωρίς να χρειάζεται να πραγματοποιηθούν με κάποια σειρά και αυτό προφανώς επιτρέπει την ευκολία σε οποιαδήποτε τροποποίηση λόγω αυτού του διαχωρισμού των τριών ρόλων. Επίσης, υπάρχει η δυνατότητα ομαδοποίησης ενεργειών σε έναν controller όπως και πολλαπλά views(προβολές) για ένα model(μοντέλο).

Δύο μειονεκτήματα του μοτίβου είναι ότι:

Απαιτείται από τους προγραμματιστές να προσδίδουν πολλαπλές αναπαραστάσεις ταυτόχρονα διότι η αφαίρεση ενός χαρακτηριστικού και από τα τρία αντικείμενα είναι χρονοβόρο και μπορεί να προκαλέσει και σύγχυση.

Μπορεί να γίνει αρκετά περίπλοκη η σωστή ακολουθία των πλαισίων ώστε να παραμένει η διεπαφή φιλική προς τον χρήστη.

Εφαρμογές:

Η εφαρμογή του είναι κατάλληλη για τον σχεδιασμό των περισσότερων ιστοσελίδων, ειδικά αυτών που θέλουν τον διαχωρισμό του model με το view. Επομένως, μόνο σε πολύ απλές εφαρμογές δεν θα ήταν χρήσιμο αυτό το μοτίβο που πετυχαίνει αυτό το σημαντικό διαχωρισμό.

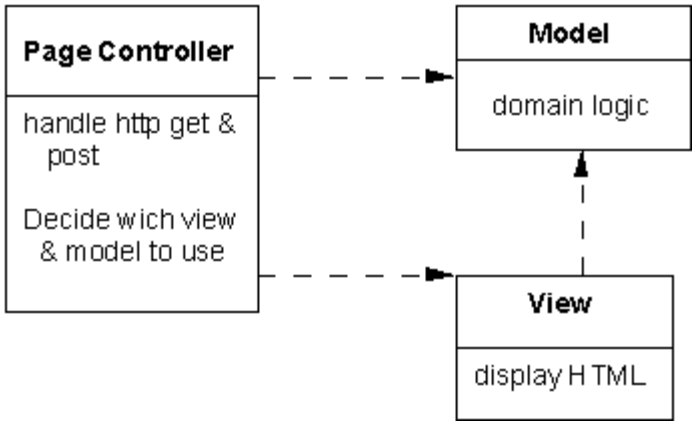
Βιβλιογραφία:

Interface Paradigm in the Smalltalk-80 System) by **Glenn E. Krasner and Stephen T. Pope,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.6.2 Page Controller

Ένα αντικείμενο που χειρίζεται ένα αίτημα για μία συγκεκριμένη σελίδα ή ενέργεια σε μια τοποθεσία Web.



Σκοπός:

Ο στόχος του page-controller είναι κάνει καλύτερη του λειτουργία του controller για ένα αίτημα μιας συγκεκριμένης σελίδας.

Δομή:

Η δομή του είναι μία κλάση ή ένα αντικείμενο ή μία δήλωση αρχείου όπου χειρίζεται το αίτημα για μία λογική ιστοσελίδα ή ενέργεια.

Πως λειτουργεί:

Όπως φαίνεται και στο παραπάνω σχήμα μπορεί να χειριστεί αίτημα για μία συγκεκριμένη σελίδα και να αποφασίζει ποια παρουσίαση (view) και ποιό μοντέλο (model) είναι κατάλληλο να χρησιμοποιηθεί. Λειτουργεί σαν μία απλή αντιστοίχιση

μεταξύ μιας αίτησης σελίδας (URL) και αρχείων. Ένα μόνο αρχείο διαχειρίζεται το αίτημα για μία συγκεκριμένη σελίδα. Αυτός ο μηχανισμός αντικατοπτρίζει τον τρόπο χειρισμού των στατικών ιστοσελίδων.

Πλεονεκτήματα και μειονεκτήματα:

Το βασικό πλεονέκτημα του page-controller είναι ότι βοηθάει την ενίσχυση των λειτουργιών του controller και πετυχαίνει να γίνει η αποφυγή σύγχυσης λειτουργιών και ανάκτησης διπλών αντικειμένων.

Αρνητικό του μοτίβου θα μπορούσε να χαρακτηριστεί ότι χρειάζεται ξεχωριστή δημιουργία controller για κάθε ιστοσελίδα και αυτό μπορεί να οδηγήσει σε σημαντική αλληλοεπικάλυψη κώδικα.

Εφαρμογές:

Η χρήση ενός page-controller για μία εφαρμογή Web είναι μία κοινή και βασική ανάγκη. Έχει διαπιστωθεί επίσης ότι τα περισσότερα πλαίσια των εφαρμογών web ενσωματώνουν το μοτίβο αυτό με τη μορφή σελίδας διακομιστή. Γενικά είναι αρκετά χρήσιμο και επειδή είναι κυρίως καλό στη διαχείριση απλών περιπτώσεων, δηλαδή στην ανάκτηση μιας σελίδας που δεν περιέχεται μαζί κάποια λογική. Τέλος, συνεργάζεται καλά, σε ένα είδος μοιράσματος των περιπτώσεων, με το επόμενο μοτίβο front-controller.

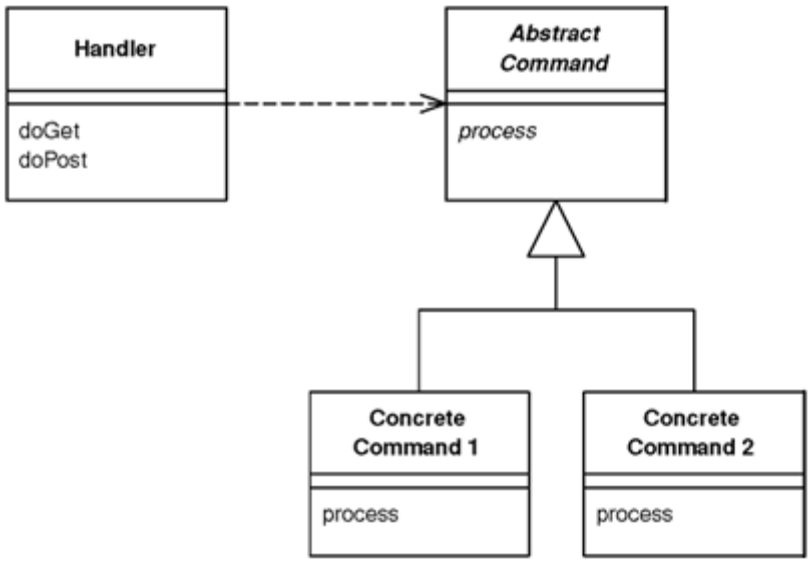
Βιβλιογραφία:

Link https://msdn.microsoft.com/en-us/library/ff649595.aspx ,

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.6.3 Front Controller

Ένας controller που χειρίζεται όλα τα αιτήματα για μια τοποθεσία Web.

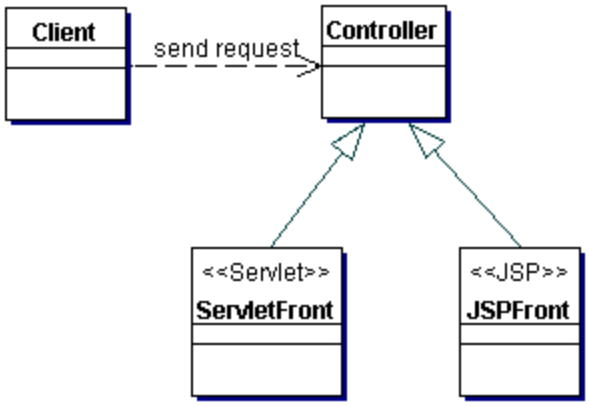


Σκοπός:

Ο σκοπός του front controller είναι επικεντρωμένος στο να διαχειρίζεται όλα τα αιτήματα και να μη περιορίζει τον αριθμό των χρηστών του συστήματος για έναν ιστότοπο. Ουσιαστικά έχει σαν στόχο να διαχειριστεί πιθανά προβλήματα όπως την απαίτηση κάθε view να παρέχει τις δικές του υπηρεσίες στο σύστημα και όπως η αναπαράσταση της πλοήγησης να παραμένει μέσα στα view.

Δομή:

Η δομή του μοτίβου είναι μία κλάση όπου διαχειρίζεται όλα αυτά τα προβλήματα σε ένα σύστημα όπως φαίνεται και στο παρακάτω σχήμα.

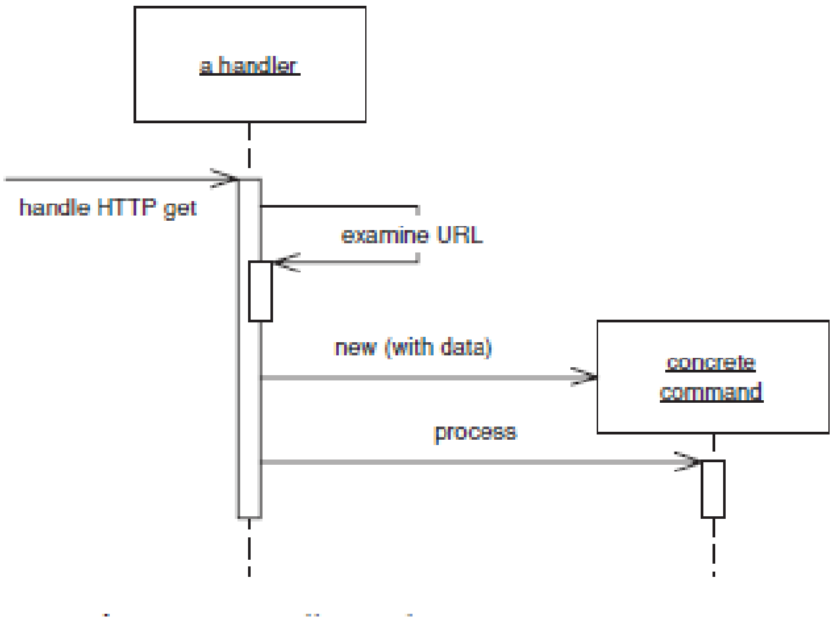


Πως λειτουργεί:

Η λειτουργία του front controller είναι κυρίως να υλοποιεί μία ροή εργασιών για μία εφαρμογή Web. Δίνει τη δυνατότητα να ελέγχεται η πλοήγηση σε ένα σύνολο συσχετιζόμενων σελίδων, παρά κάθε σελίδα να είναι υπεύθυνη για την πλοήγηση.

Ουσιαστικά ξεχωριστά σενάρια όπως π.χ. login.php, register.php και order.php που θα ικανοποιούν κάποιο αίτημα, τα συσσωρεύει σε ένα σενάριο όπως π.χ. index.php όπου χειρίζεται όλα τα αιτήματα και τις εργασίες μαζί και για κάθε συγκεκριμένο αίτημα θα προκύπτει κάποιο αντικείμενο και μέθοδος κλήσης για να χειριστεί τις απαιτούμενες ενέργειες.

Σχήμα : Πώς λειτουργεί ο front controller



Πλεονεκτήματα και μειονεκτήματα:

Το σημαντικότερο πλεονέκτημα του είναι ότι συμβάλλει στην αποφυγή διπλού κώδικα για κάποιες παρόμοιες περιπτώσεις των views. Επιπλέον, υπάρχει η δυνατότητα πρόσθεσης εντολών χωρίς να χρειαστεί κάποια αλλαγή και τέλος η θέση του controller είναι ιδανική για να επιβάλει κάποια πολιτική σε επίπεδο εφαρμογής, όπως η ασφάλεια και η παρακολούθηση χρήσης.

Τα δύο βασικά μειονεκτήματα είναι τα εξής: η αυξημένη πολυπλοκότητα διότι είναι περίπλοκο να υλοποιηθεί σε σχέση με το προηγούμενο και επομένως το επόμενο μειονέκτημα είναι η χαμηλή απόδοση του συστήματος και λόγω υψηλής πολυπλοκότητας αλλά και λόγω των μαζεμένων αιτημάτων, όπως δηλαδή κάποιο από αυτά να είναι η σύνδεση με βάση δεδομένων με αποτέλεσμα να γίνεται αργή η απόδοση.

Εφαρμογές:

Η εφαρμογή του είναι κατάλληλη όταν ο διακομιστής web είναι περίπλοκος, διότι με τη βοήθεια του front controller στη διαχείριση των αιτημάτων γίνεται πιο απλή η διαμόρφωση του διακομιστή. Άρα είναι ιδανικό αυτό το μοτίβο για μία σύνθετη τοποθεσία web.

Βιβλιογραφία:

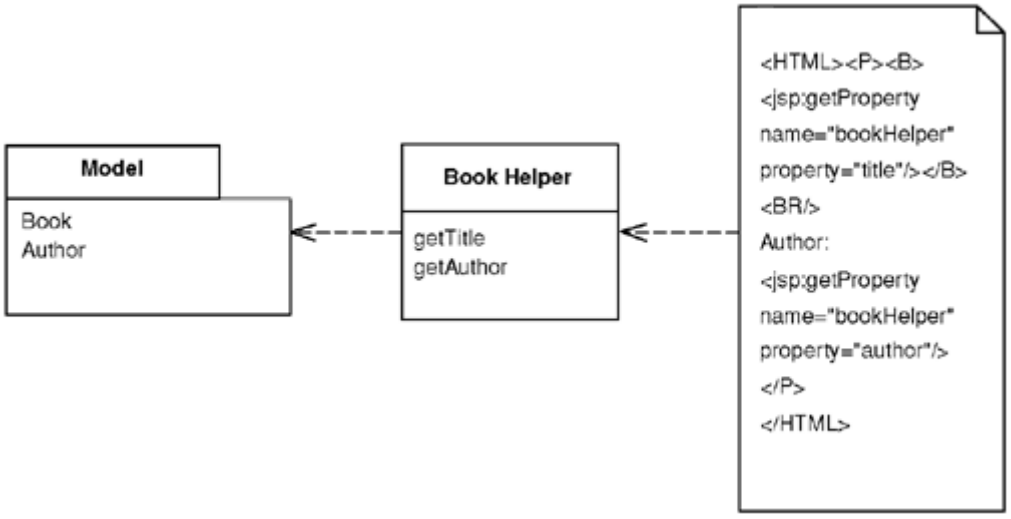
Book **Core J2EE Patterns: Best Practices and Design Strategies** by **Deepak Alur, Dan Malks, John Crupi,**

Book **Design Patterns: Elements of Reusable Object-Oriented Software** by **Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Grady Booch,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.6.4 Template View

Εμφανίζει πληροφορίες σε HTML ενσωματώνοντας δείκτες σε μια σελίδα HTML.



Σκοπός:

Το μοτίβο αυτό έχει σκοπό να κάνει τη διαχείριση αιτημάτων μέσω κάποιων στατικών σελίδων HTML με ενσωματωμένους δείκτες.

Δομή:

Δομικά το template view είναι μία σελίδα HTML όπου είναι βασισμένη στη δομή της κύριας σελίδας ώστε να μη χάνεται η συνοχή των προβολών.

Πως λειτουργεί:

Η λειτουργία του μοτίβου βασίζεται στους ενσωματωμένους δείκτες. Για να μπορέσει να εξυπηρετήσει ένα αίτημα η σελίδα HTML αντλεί μέσω των δεικτών τα κατάλληλα δεδομένα, διότι οι δείκτες επικοινωνούν απευθείας με πραγματικά προγράμματα ώστε να αντλήσουν αποτελέσματα για να υλοποιήσουν τα αιτήματα που ζητούνται. Ουσιαστικά χωρίς να εξαρτάται από κάποια κλάση, κρατάει τη δομή της σελίδας, μέσω της σελίδας HTML και εξυπηρετεί τα αιτήματα μέσω δεικτών που αντλούν τις απαντήσεις με την κατάλληλη επικοινωνία με τα προγράμματα. Αυτό επιτρέπει την ενσωμάτωση λογικής στη σελίδα χωρίς να χρειάζεται να ενσωματωθεί η επιχειρησιακή οπτική.

Πλεονεκτήματα και μειονεκτήματα:

Το ιδιαίτερο πλεονέκτημα αυτού του μοτίβου είναι αυτό που αναφέραμε στο τέλος της λειτουργίας του. Δίνει την δυνατότητα να διαχωρίζει την λογική ενός view από την επιχειρησιακή λογική που μπορεί να έχει το μοντέλο της εφαρμογής. Επίσης, η μοντελοποίηση μέσω της HTML δίνει την ελευθερία να ασχοληθούν στην δημιουργία μοτίβων και άνθρωποι που έχουν κυρίως γνώσεις πάνω σε αυτή τη γλώσσα.

Κάποιο πρόβλημα που μπορεί να δημιουργηθεί μέσω του template view είναι ότι η λογική του view μπορεί να κυριαρχήσει σε αυτό αντί να είναι ενθυλακωμένη στο controller ή στο model.

Εφαρμογές:

Οι εφαρμογές που μπορεί να χρησιμοποιηθεί αξιόλογα το μοτίβο αυτό είναι όσες θέλουν να επικεντρωθούν στην γλώσσα HTML για την δυνατότητα δημιουργίας δομών με συνοχή. Είναι σαν να απευθύνεται κυρίως σε εφαρμογές που δεν υλοποιούνται ως επί το πλείστον από προγραμματιστές λογισμικού αλλά από προγραμματιστές γραφικού περιβάλλοντος.

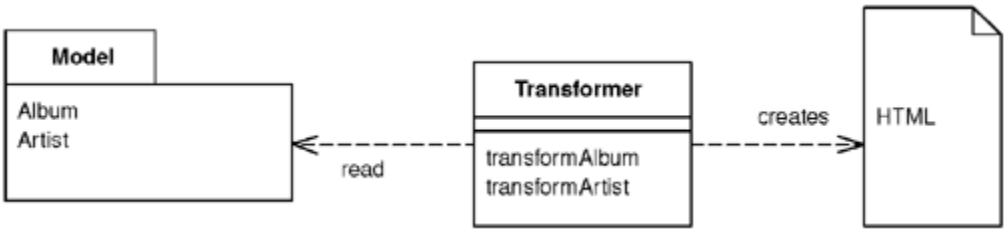
Βιβλιογραφία:

Book **Presentation Patterns: Techniques for Crafting Better Presentations** by **Neal Ford, Matthew McCullough, Nathaniel Schutta,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.6.5 Transform View

Μια προβολή που επεξεργάζεται τα δεδομένα του domain, στοιχείο προς στοιχείο και το μετατρέπει σε HTML.



Σκοπός:

Σκοπός του transform view είναι να αποτελέσει τη μετατροπή δεδομένων domain από το μοντέλο σε μία σελίδα HTML. Ουσιαστικά, στοχεύει να γίνεται άμεση μετατροπή δεδομένων όπου είναι βαθιά στη λογική της εφαρμογής σε σελίδα HTML.

Δομή:

Η δομή του είναι μία κλάση όπου δέχεται σαν είσοδο δεδομένα του domain model(3.1.2) και παράγει σαν έξοδο μία σελίδα HTML.

Πως λειτουργεί:

Η βασική λειτουργία του μοτίβου είναι να εξετάζει που είναι τοποθετημένα τα δεδομένα του domain και τη δομή τους, αναγνωρίζοντας τη μορφή των δεδομένων και γράφει γι’ αυτό τη μετατροπή του σε HTML. Η επιλογή του μετασχηματισμού γίνεται ξεχωριστά για κάθε είσοδο που δέχεται, ελέγχει ποιος μετασχηματισμός είναι κατάλληλος για κάθε είσοδο και τον καλεί. Μπορεί να διαχειριστεί με όποια σειρά θέλει τα δεδομένα χωρίς κάποια επίπτωση στην συνοχή. Επίσης, στην λειτουργία αυτού είναι συμβατή και βοηθητική η χρήση XSLT για τους μετασχηματισμούς, σε σημείο να αποτελεί βασική προϋπόθεση για το αν θα επιλεγεί αυτό το μοτίβο.

Πλεονεκτήματα και μειονεκτήματα:

Ένα βασικό χαρακτηριστικό που λειτουργεί θετικά του transform view είναι ότι η είσοδος στον μετασχηματισμό μπορεί να προέρχεται από οποιαδήποτε γλώσσα ή πλατφόρμα εφόσον ισχύει έγκυρη XML. Επιπλέον, είναι δυνατή η επαναχρησιμοποίηση κώδικα σε περισσότερες από μία κλάσεις.

Μειονέκτημα θα μπορούσε να χαρακτηριστεί ότι η χρήση του XSLT δεν αποτελεί ακόμα άνετη γλώσσα για τους τακτικούς προγραμματιστές με αποτέλεσμα σε κάποιες περιπτώσεις να υπάρχει έλλειψη διαθεσιμότητας εργαλείων πάνω σε αυτή τη γλώσσα και να αποδεικνύεται χρηστικότερο το template view.

Εφαρμογές:

Η εφαρμογή αυτού του μοτίβου γίνεται κατάλληλη για πλατφόρμες εφαρμογών που διαθέτουν εργαλεία XSML γλώσσας για τους κατάλληλους μετασχηματισμούς.

Βιβλιογραφία:

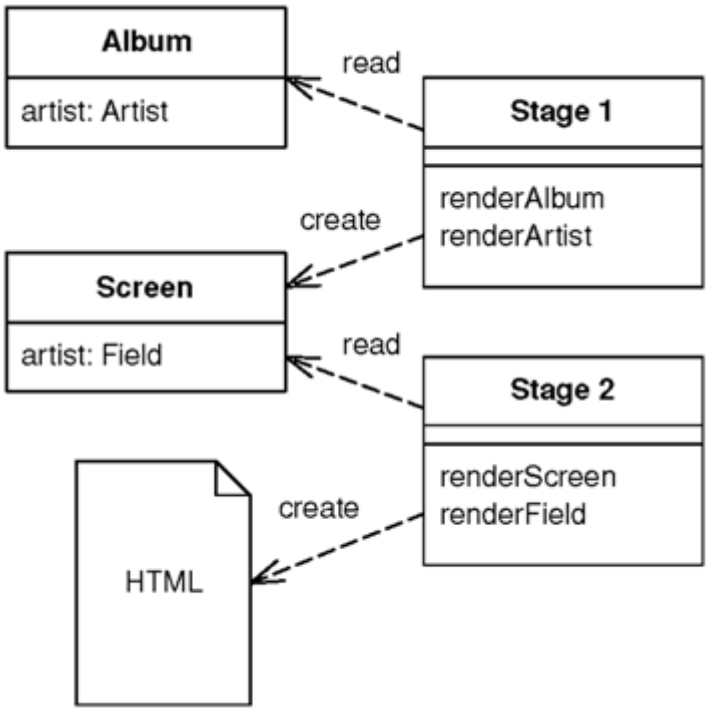
Book **SOA Design Patterns** by **Thomas Erl,**

Link http://wiki3.cosc.canterbury.ac.nz/index.php/Transform_view_pattern ,

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.6.6 Two Step View

Μετατρέπει τα δεδομένα domain σε HTML σε δύο βήματα: πρώτα με τη διαμόρφωση ορισμένου είδους λογικής σελίδας, και στη συνέχεια την απόδοση της λογικής σελίδας σε HTML.



Σκοπός:

Ο σκοπός αυτού του μοτίβου είναι να οργανώσει τη μετατροπή των δεδομένων του domain σε δύο στάδια. Αυτό έχει ως αποτέλεσμα την συνεπή οργάνωση και εμφάνιση του ιστότοπου των εφαρμογών Web που έχουν πολλές σελίδες.

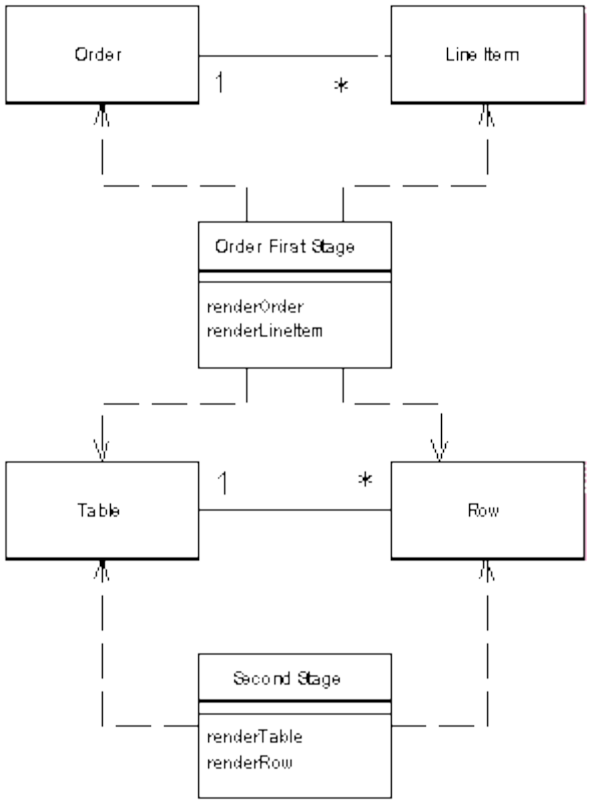
Δομή:

Η δομή του two step view είναι δύο διαφορετικά αρχεία που αποτελούν τα δύο στάδια του μοτίβου. Το πρώτο στάδιο έχει επικοινωνία με τα δεδομένα του domain δηλαδή με το μοντέλο και το δεύτερο στάδιο παράγει την παρουσίαση του πρώτου με HTML.

Πως λειτουργεί:

Η λειτουργία του στο πρώτο στάδιο είναι να μετατρέπει τα δεδομένα του μοντέλου σε μια λογική παρουσίαση χωρίς συγκεκριμένη μορφοποίηση για το καθένα και στο δεύτερο να μετατρέπει εκείνη την λογική παρουσίαση με την πραγματική μορφοποίηση που απαιτείται. Συγκεκριμένα στο πρώτο στάδιο συγκεντρώνει τις πληροφορίες σε μία λογική δομή παρουσίασης χωρίς να περιέχει κώδικα HTML, οπότε η ευθύνη του είναι να αντλεί τις πληροφορίες της σελίδας από το domain ή από τη βάση δεδομένων και να τις παρουσιάζει σε μία λογική σειρά αλλά όχι μορφοποιημένα. Στη συνέχεια, το δεύτερο στάδιο παίρνει αυτή τη λογική δομή παρουσίασης και το μορφοποιεί σε HTML διότι γνωρίζει τις κατάλληλες αντιστοιχίες σε γλώσσα HTML μέσω της προσανατολισμένης δομής της παρουσίασης. Σε αυτό το στάδιο ουσιαστικά μπορεί να δημιουργηθούν και μοναδικές εμφανίσεις ιστοσελίδων και πολλαπλές εμφανίσεις ιστοσελίδων που θέλουν να έχουν την ίδια εμφάνιση για διαφορετικούς χρήστες.

Σχήμα: Δοκιμαστικές κλάσεις για απόδοση σε δύο βήματα.



Πλεονεκτήματα και μειονεκτήματα:

Το σημαντικότερο πλεονέκτημα του μοτίβου αυτού είναι ότι μπορεί να υποστηρίξει μια παγκόσμια αλλαγή στη μορφή κάποιας σελίδας γιατί θα χρειαστεί να αλλαχθεί μόνο το δεύτερο στάδιο. Επιπλέον, θετικό είναι ότι μπορεί να υποστηρίξει και εφαρμογές Web με μοναδική εμφάνιση και πολλαπλές εφαρμογές με κοινή εμφάνιση. Τέλος, δεν υπάρχει επικάλυψη κώδικα HTML διότι το δεύτερο στάδιο χρησιμοποιεί κάποια παρουσίαση σε παγκόσμιο επίπεδο ή κλάση.

Το μοναδικό μειονέκτημα του two step view είναι ότι με τη χρήση του υπάρχει κάποια επιπρόσθετη πολυπλοκότητα στο πρόγραμμα.

Εφαρμογές:

Το μοτίβο αυτό μπορεί να υποστηρίξει σχεδόν όλων των ειδών τις εφαρμογές Web. Είναι κατάλληλο, όπως είπαμε και παραπάνω, και για εφαρμογές όπου θέλουν μοναδική εμφάνιση του ιστότοπου αλλά και για εφαρμογές με κοινή οθόνη. Επιπλέον, είναι ιδανικό και εφαρμογές που πιθανό να χρειάζονται συχνά αλλαγές στην παρουσίαση της διεπαφής και με αυτό το τρόπο μέσω της αλλαγής μόνο του δεύτερου σταδίου μπορεί να γίνεται πολύ εύκολα.

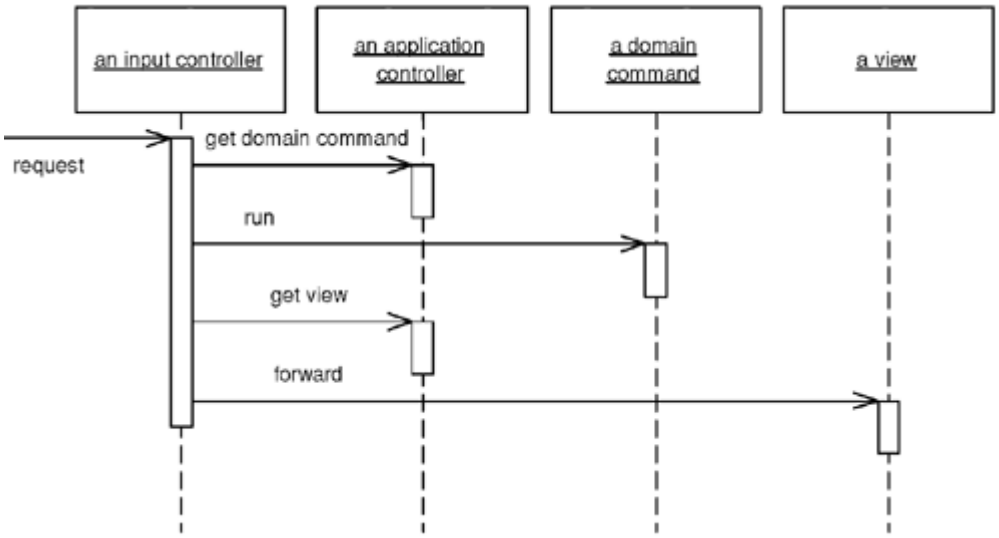
Βιβλιογραφία:

Book **Presentation Patterns: Techniques for Crafting Better Presentations** by **Neal Ford, Matthew McCullough, Nathaniel Schutta,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.6.7 Application Controller

Ένα κεντρικό σημείο για τον χειρισμό της οθόνης πλοήγησης και της ροή μιας εφαρμογής.



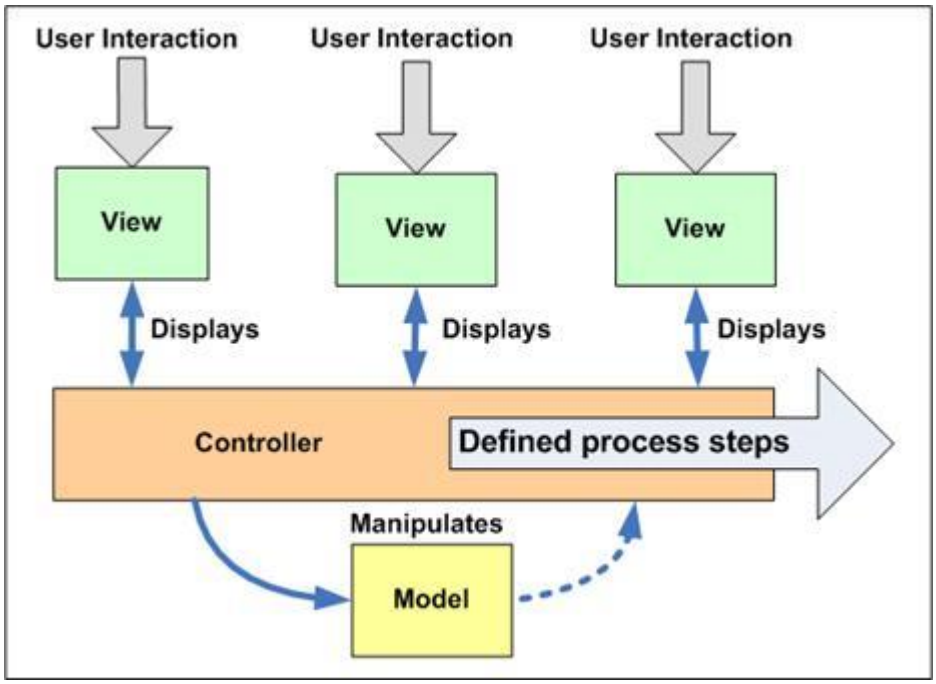
Σκοπός:

Σκοπός του application controller είναι να αντιμετωπίσει το πρόβλημα που προκύπτει όταν χρειάζεται να αλλαχθεί η ροή της σελίδας όπου τότε πρέπει να αλλαχθεί ο πηγαίος κώδικας των views για να γίνει η σωστή πλοήγηση αυτών.

Δομή:

Δομικά είναι μία ενιαία τοποθεσία όπου βρίσκεται και η ροή της εφαρμογής και η οθόνη πλοήγησης, ώστε να μπορούν να γίνουν αλλαγές είτε στη ροή σελίδας είτε στη λογική χωρίς αλλαγές ή με ελάχιστες αλλαγές στον πηγαίο κώδικα των views.

Σχήμα: Μία απεικόνιση της δομής και λειτουργίας του application controller.



Πως λειτουργεί:

Το μοτίβο έχει κύρια λειτουργία να επιλέγει ποιές εντολές θα εκτελεστούν στο στρώμα του domain ώστε να εξυπηρετηθεί η λογική της εφαρμογής και ποιο view θα προωθήσει για εμφάνιση. Για την άρτια λειτουργία αυτού του ελεγκτή (controller) μπορεί να χρησιμοποιηθεί ένα πλαίσιο ελέγχου όπου ασχολείται με τις μεταβλητές που καθορίζουν τη σωστή πλοήγηση κάθε φορά και το επιχειρηματικό πλαίσιο που ασχολείται με τις μεταβλητές που χρησιμοποιούνται για την επίτευξη της επιχειρησιακής λογικής. Μέσω αυτού του πλαισίου υπάρχει επικοινωνία με το κατάλληλο view κάθε φορά, στο οποίο χειρίζεται και η λογική του βήματος που εφαρμόζει, αγνοώντας τη λογική πλοήγησης των σελίδων διότι τελικά καθορίζεται από το πλαίσιο ελέγχου. Ουσιαστικά όλη αυτή η διαδικασία συμβάλλει στο να μπορεί να γίνει κάποια απαραίτητη αλλαγή στη ροή σελίδας χωρίς να χρειαστεί να γίνει μεγάλη αλλαγή στον πηγαίο κώδικα αλλά να μπορεί να χειριστεί την αλλαγή στην πλοήγηση και στη λογική το ίδιο στρώμα εργασιών, δηλαδή το application controller.

Πλεονεκτήματα και μειονεκτήματα:

Η θετικότερη συμβολή αυτού του μοτίβου είναι ότι δίνει τη δυνατότητα ευελιξίας στη ροή σελίδας σε περιπτώσεις που είναι απαραίτητο και αποδοτικότερο χωρίς να χρειαστεί να υπάρξει μεγάλη αλλαγή από προγραμματιστική άποψη. Επίσης, εξασφαλίζει ότι δε θα υπάρξει σύγχυση στην επιχειρησιακή λογική και στη πλοήγηση σε κάποια πιθανή αλλαγή.

Το επιβαρυντικό με το application controller είναι ότι μπορεί να υπάρξουν πολλές μεταβλητές πάνω σε αυτή τη κοινή τοποθεσία για την σωστή λειτουργία της εφαρμογής και ότι θα πρέπει να βρεθεί τρόπος σύνδεσης για τους controllers και για τα views.

Εφαρμογές:

Για τις εφαρμογές όπου η σειρά που πλοηγείται τις οθόνες ο χρήστης, χρειάζεται να είναι συγκεκριμένη και γενικά μια σύνθετη εφαρμογή με απαιτήσεις πλοήγησης, το μοτίβο αυτό είναι απαραίτητο και ιδανικό για να υποστηρίξει απαιτήσεις συγκεκριμένης αλλά δυναμικής πλοήγησης οθονών.

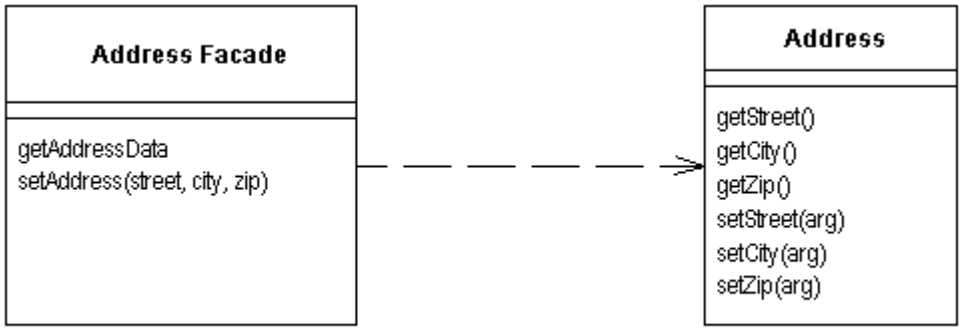
Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.7 Distribution Patterns

3.7.1 Remote Facade

Παρέχει μια χονδροειδής πρόσοψη σε λεπτόκοκκο αντικειμένων για τη βελτίωση της αποδοτικότητας σε ένα δίκτυο.



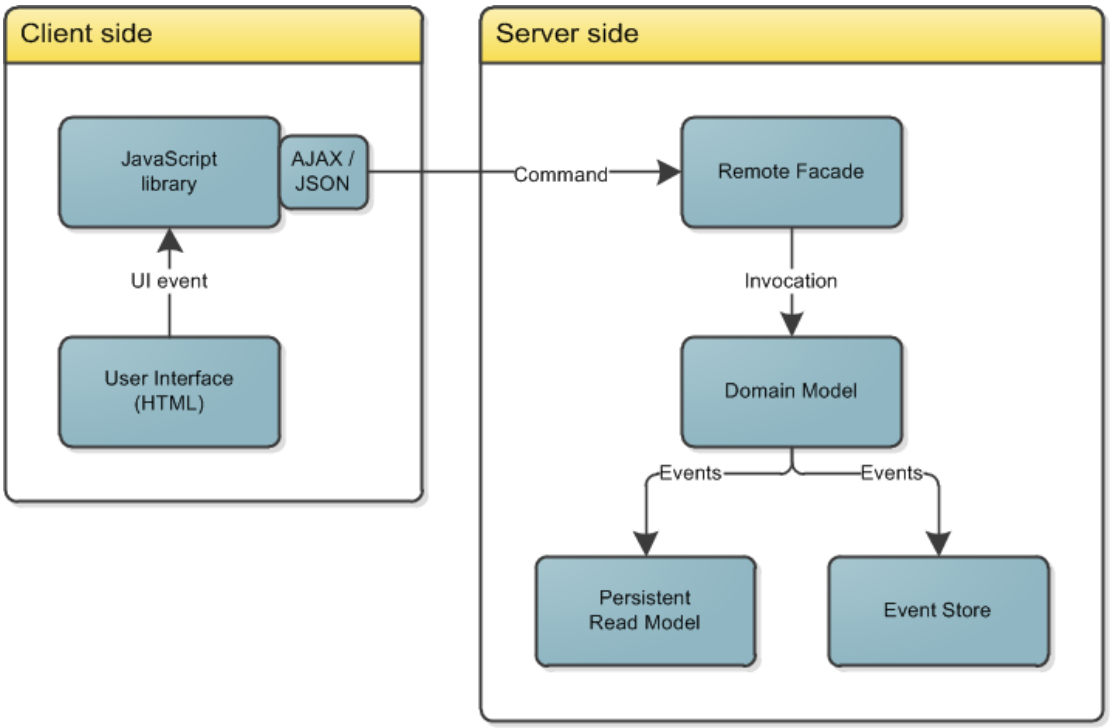
Σκοπός:

Σκοπός του remote facade είναι να διαχειριστεί τις μακρινές κλήσεις σε λεπτοκομμένα αντικείμενα (αντικείμενα που αποτελούνται από πολλά μικρά μέρη, όπως για παράδειγμα στην εικόνα το αντικείμενο Address). Το να είναι λεπτοκομμένα τα αντικείμενα είναι κάτι που εξυπηρετεί πολύ τον αντικειμενοστραφή προγραμματισμό και κάνει πιο κατανοητό ένα αίτημα μεταξύ αντικειμένων. Αυτό προφανώς δημιουργεί κάποιες φορές πολλαπλές κλήσεις για να μπορεί να διατηρηθεί υψηλή η αλληλεπίδραση μεταξύ των αντικειμένων. Ως συνέπεια αυτών των κλήσεων μπορεί να δημιουργήσει πολλαπλές μακρινές κλήσεις και ο στόχος του μοτίβου είναι να χειριστεί αυτό το πρόβλημα.

Δομή:

Η δομή του είναι μία κλάση όπου δημιουργεί μια πρόσοψη που είναι απομακρυσμένη και προσανατολίζει ένα χοντροκομμένο αντικείμενο σε ένα λεπτοκομμένο αντικείμενο του domain. Αυτό δομικά μπορεί να φανεί και στο παραπάνω σχήμα πως μπορεί να διαμορφωθεί μία κλάση για να έχει την κατάλληλη συμπεριφορά.

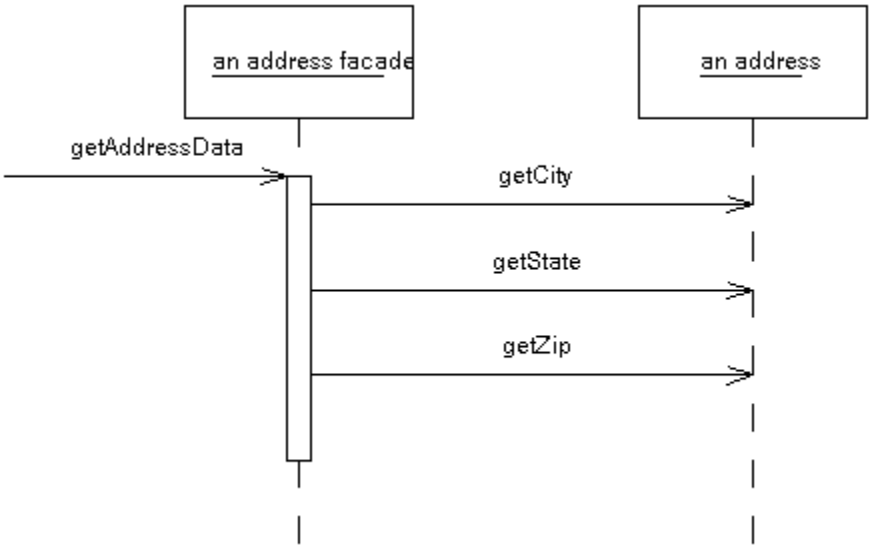
Σχήμα: Άλλη μία εικόνα που βοηθάει στην κατανόηση που βρίσκεται δομικά αυτή η πρόσοψη.



Πως λειτουργεί:

Η κύρια λειτουργία του remote facade είναι μία πρόσοψη να αντικαθιστά όλα τα get και set των μεθόδων του αντικειμένου κανονικής διεύθυνσης με ένα getter και ένα setter αντίστοιχα. Οπότε όταν ένας χρήστης καλεί ένα setter, η πρόσοψη της αντίστοιχης διεύθυνσης διαβάζει τα δεδομένα της μεθόδου και καλεί τα αντίστοιχα κομμάτια του πραγματικού αντικειμένου διεύθυνσης να κάνουν τις ενέργειες. Η λειτουργία του μοτίβου σταματάει εκεί και αυτό σημαίνει ότι η μόνη εργασία που κάνει είναι αυτή και δεν κουβαλάει μαζί του τίποτα από τη λογική του domain.

Σχήμα: Λειτουργία κλήσεων από την πρόσοψη προς το αντικείμενο του domain.



Πλεονεκτήματα και μειονεκτήματα:

Υπάρχει συνεκτικότητα με αυτό το μοτίβο και βοηθάει στο να μη γίνεται επικάλυψη κώδικα. Επιπλέον, μέσω της πρόσοψης αυτής υπάρχει η δυνατότητα να γίνουν δοκιμές πάνω τους και συμβάλει θετικά στον έλεγχο της διατήρησης της λογικής της εφαρμογής.

Το μόνο που μπορεί να χαρακτηριστεί σαν μειονέκτημα είναι ότι αυτός ο επιπρόσθετος κώδικας θα μπορούσε να επιβαρύνει την απόδοση του προγράμματος.

Εφαρμογές:

Το remote facade μπορεί να εφαρμοστεί σε εφαρμογές που χρειάζονται πρόσβαση σε ένα λεπτοκομμένο αντικείμενο του μοντέλου το οποίο είναι απομακρυσμένο. Σε τέτοιες εφαρμογές είναι κατάλληλο αυτό το μοτίβο διότι δίνει τα πλεονεκτήματα και ενός χοντροκομμένου αντικειμένου μέσω της πρόσοψης και του ίδιου του λεπτοκομμένου αντικειμένου.

Βιβλιογραφία:

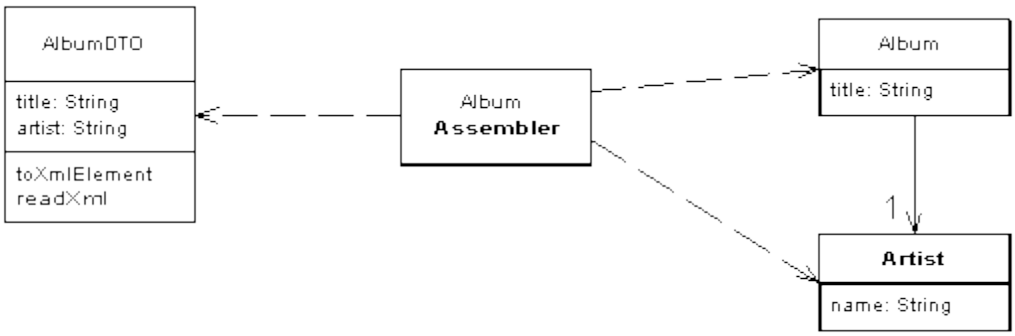
Book **SOA Design Patterns** by **Thomas Erl,**

Book **Business Patterns for Software Developers** by **Allan Kelly,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.7.2 Data Transfer Object

Ένα αντικείμενο που μεταφέρει δεδομένα μεταξύ των διαδικασιών σε σειρά για να μειωθεί ο αριθμός των κλήσεων μεθόδου.



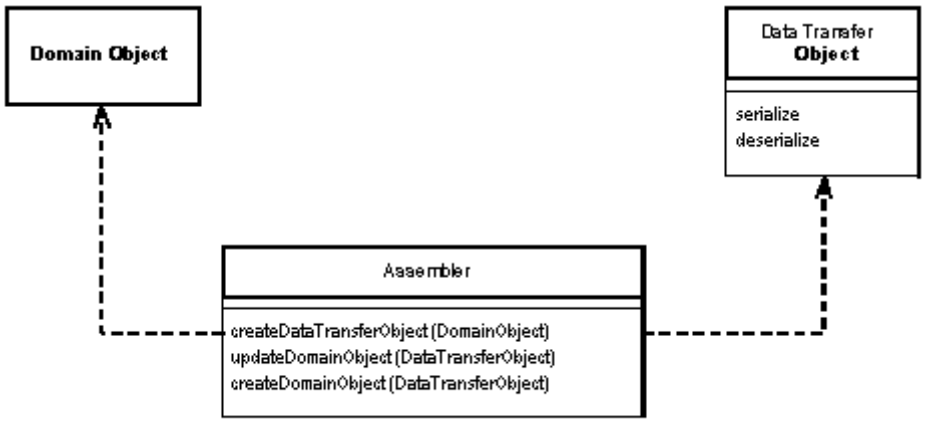
Σκοπός:

Ο σκοπός του data transfer object είναι να αντιμετωπίσει το πρόβλημα των δαπανηρών κλήσεων για απομακρυσμένες διεπαφές. Όπως είδαμε και στο προηγούμενο μοτίβο τα αντικείμενα τα οποία για να ανακτηθούν χρειάζονται μακρινές κλήσεις είναι κάτι που μειώνει σημαντικά την απόδοση. Αυτό το μοτίβο έχει στόχο να χειριστεί αυτό το πρόβλημα.

Δομή:

Δομικά υπάρχει μία κλάση όπου δέχεται δεδομένα ενός αντικειμένου από τα αντικείμενα του domain και τα σειριοποιεί. Επιπλέον, είναι μία κλάση όπου βρίσκεται ενδιάμεσα από τον πελάτη/χρήστη και το αντικείμενο στο domain, θα γίνει περισσότερο κατανοητό στη περιγραφή της λειτουργίας του.

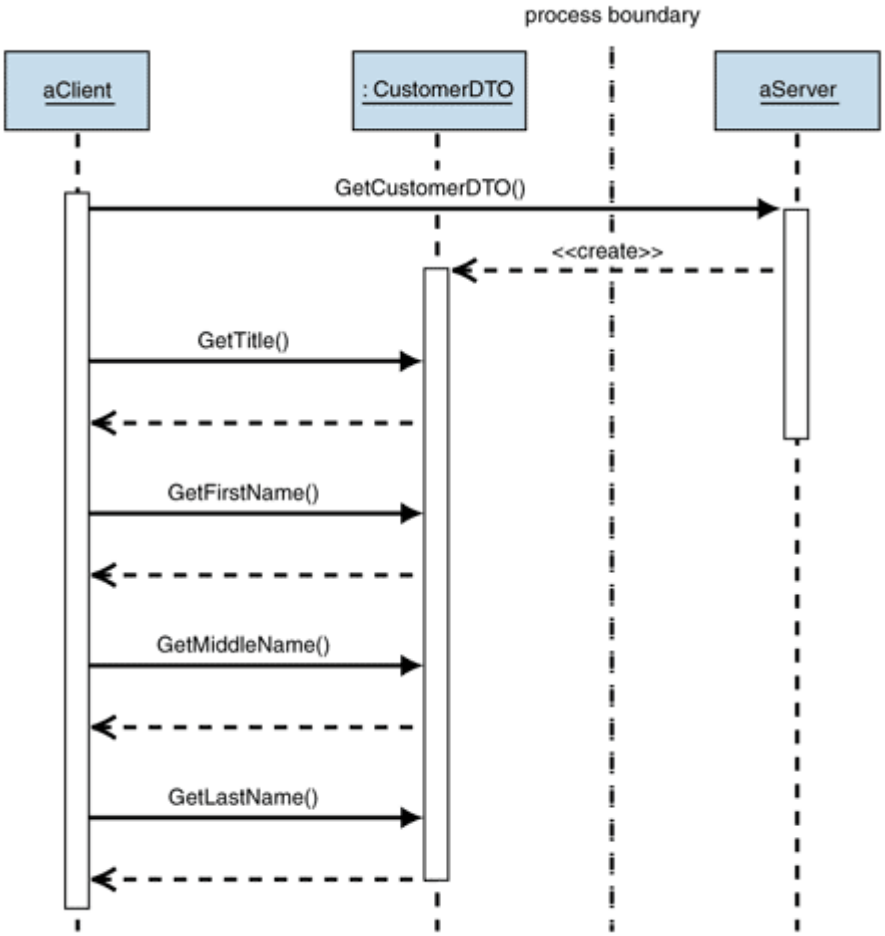
Σχήμα: Φαίνεται πως δέχεται δεδομένα από το αντικείμενο του domain.



Πως λειτουργεί:

Η λειτουργία του data transfer object είναι να αντλεί παραπάνω δεδομένα, από το αντικείμενο του domain, σε σχέση με αυτά που ζήτησε ο πελάτης από μία κλήση του στον server και τα σειριοποιεί. Αυτό γίνεται για να γίνει λιγότερο δαπανηρή η διαδικασία των κλήσεων, διότι αμέσως μετά εάν ζητήσει κάτι παραπάνω ο πελάτης από τα δεδομένα θα τα αντλήσει από το data transfer object μέσω της αποσειριοποίησης χωρίς να χρειαστεί να γίνει και άλλη κλήση στον server. Ουσιαστικά, γίνεται προτιμότερο να ανακτηθούν πολλά περισσότερα δεδομένα με μία κλήση παρά να γίνουν πολλές κλήσεις. Μέσα στη λειτουργία του δεν εκφράζεται κάποια επιχειρησιακή λογική, το μόνο που συμβαίνει είναι να εξυπηρετεί στην ανάκτηση δεδομένων χωρίς να χρειάζεται να γίνεται για όλα τα αιτήματα του πελάτη νέα κλήση.

Σχήμα: Η λειτουργία του μοτίβου σε κάποιες πιθανές κλήσεις.



Πλεονεκτήματα και μειονεκτήματα:

Το βασικότερο και πολυτιμότερο πλεονέκτημα είναι ότι με τη χρήση του μοτίβου αποφεύγονται πολλές δαπανηρές κλήσεις και αυτό συμβάλλει στην καλύτερη απόδοση του προγράμματος. Επιπλέον, είναι εύκολα κατανοητό για την υλοποίηση του από προγραμματιστική άποψη.

Είναι δύσκολο να χαρακτηριστεί κάτι σαν μειονέκτημα σε αυτό το μοτίβο, καθώς συμβάλλει μόνο θετικά σε ένα πρόγραμμα, αλλά σε εκλεπτυσμένες περιπτώσεις θα μπορούσε να κάνει μεγαλύτερη μεταφορά δεδομένων απ’ ότι μπορεί τελικά να χρειαστεί, αλλά σ’ αυτό υπάρχει τρόπος αντιμετώπισης μέσω και άλλων μοτίβων.

Εφαρμογές:

Η εφαρμογή αυτού του μοτίβου είναι ιδανική στο να υποστηρίξει το προηγούμενο μοτίβο remote façade(3.7.1) ώστε να μειώσει τις δαπανηρές κλήσεις που μπορεί να χρειαστεί να κάνει. Προφανώς, για όσες εφαρμογές γίνονται μακρινές κλήσεις για ανακτήσεις δεδομένων ή για πιθανές ενέργειες σε απομακρυσμένα αντικείμενα, είναι ένα μοτίβο που γίνεται απαραίτητο για να τις κάνει πιο λειτουργικές.

Βιβλιογραφία:

Link https://msdn.microsoft.com/en-us/library/ms978717.aspx ,

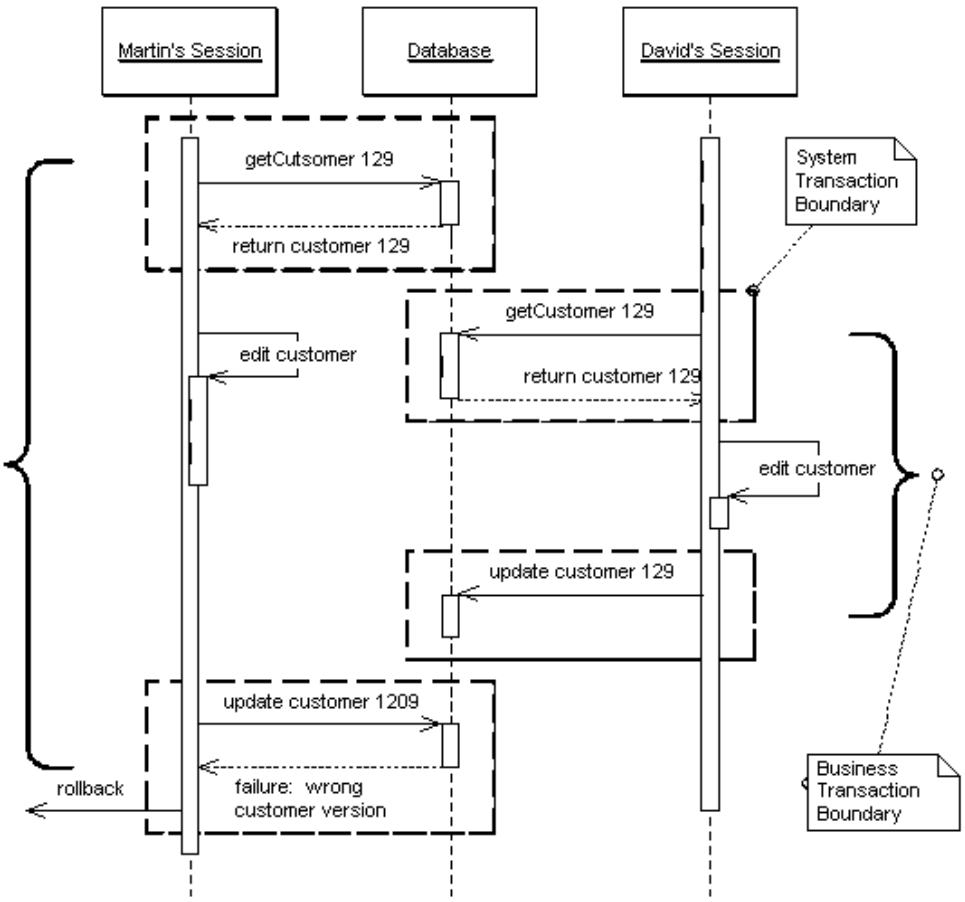
Book **Business Patterns for Software Developers** by **Allan Kelly,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.8 Offline Concurrency Patterns

3.8.1 Optimistic Offline Lock (by David Rice)

Αποτρέπει τις συγκρούσεις μεταξύ επιχειρησιακών συναλλαγών που γίνονται ταυτόχρονα ανιχνεύοντας μία σύγκρουση και ανατρέποντας τη συναλλαγή.



Σκοπός:

Ο σκοπός του optimistic offline lock είναι να διαχειριστεί το πιθανό πρόβλημα συγκρούσεων κάποιων επιχειρησιακών συναλλαγών που συμβαίνουν ταυτόχρονα και επιδιώκουν να αλλάξουν κάποιο πεδίο στο ίδιο αντικείμενο στη βάση δεδομένων.

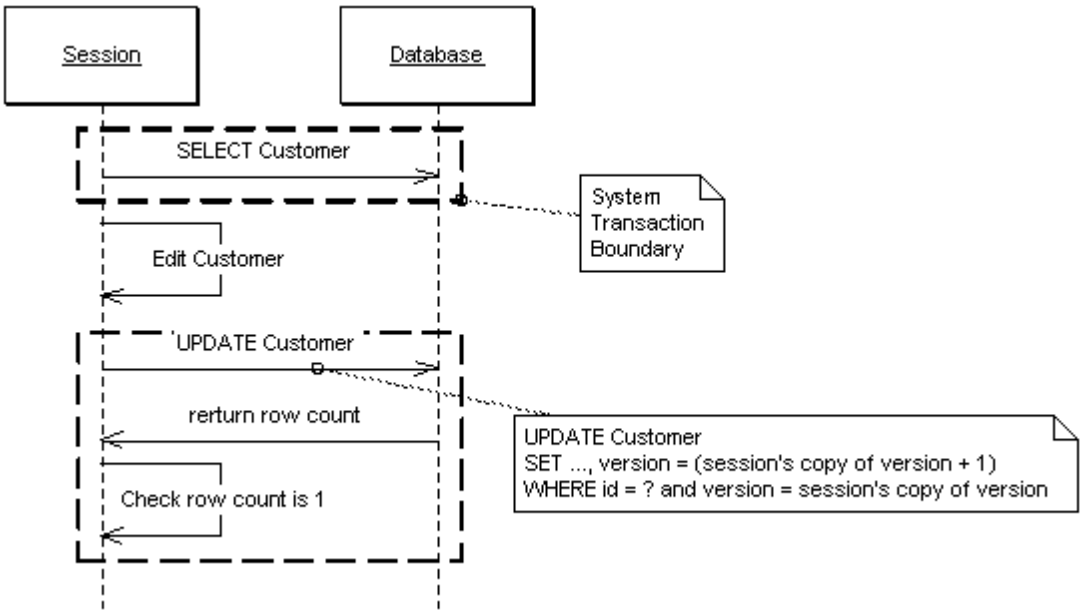
Δομή:

Το μοτίβο αυτό λειτουργεί πάνω στη δομή των sessions σε μία εφαρμογή που κάνει συναλλαγές. Αυτό σημαίνει ότι πάνω σε αυτό το διάλογο που ανοίγει μεταξύ ενός πελάτη και της βάσης δεδομένων, εφαρμόζει τη λειτουργία του πάνω στα ερωτήματα SQL που προκύπτουν μέσα στο διάλογο όπως θα περιγραφεί παρακάτω.

Πως λειτουργεί:

Ο τρόπος που λειτουργεί το optimistic offline lock είναι ότι υποθέτει πως η πιθανότητα να συμβεί κάποια σύγκρουση είναι χαμηλή. Οπότε εάν γίνουν δύο παρόμοια αιτήματα ανάκτησης των ίδιων δεδομένων από δύο διαφορετικά session, θα κάνει και τα δύο γιατί ποντάρει στο ότι δε θα υπάρξει κάποια σύγκρουση. Για να διαχειριστεί όμως τη σύγκρουση που μπορεί να υπάρξει αργότερα, σε περίπτωση που και από τους δύο διαλόγους αυτού του αντικειμένου γίνει η ενέργεια κάποιας ενημέρωσης του κοινού αντικειμένου, τότε πρέπει να ανατρέψει κάποια συναλλαγή. Αυτό το πετυχαίνει συνήθως, χρησιμοποιώντας ένα νέο πεδίο στη γραμμή του αντικειμένου στη βάση δεδομένων που αντιστοιχεί στην έκδοση (version) του αντικειμένου. Η έκδοση είναι αρχικοποιημένη στο 1 και όταν κάποιος πελάτης ζητήσει το αντικείμενο δέχεται και αυτό το δεδομένο μαζί, το ίδιο συμβαίνει και στον επόμενο πελάτη που θα ζητήσει το ίδιο αντικείμενο. Εάν κάποιος πελάτης κάνει κάποια ενημέρωση σε αυτό το αντικείμενο, το μοτίβο αυτό θα ελέγξει εάν η έκδοση παραμένει η ίδια, εάν παραμένει ίδια θα κάνει την αλλαγή και θα αυξήσει κατά ένα την έκδοση ενώ εάν η έκδοση έχει αλλάξει στη βάση δεδομένων σε σχέση με την έκδοση του διαλόγου (session) τότε θα ανατραπεί αυτή η συναλλαγή και δε θα πραγματοποιηθεί γιατί σημαίνει ότι κάποιος άλλος πελάτης έκανε ενημέρωση και δεν ισχύουν ακόμα τα ίδια δεδομένα στο ανακτηθέν αντικείμενο.

Σχήμα: Ενημέρωση σε optimistic offline lock.



Πλεονεκτήματα και μειονεκτήματα:

Μεγάλο πλεονέκτημα σε αυτό το μοτίβο είναι ότι μπορεί να υποστηρίξει συγχρονισμό σε πραγματικό χρόνο, όπως ο έλεγχος πολλαπλών στοιχείων από πολλαπλούς χρήστες ταυτόχρονα, εφόσον υπάρχει μία στρατηγική συγχώνευσης. Επίσης, η υλοποίηση του είναι εύκολη και κατανοητή.

Η περίπτωση όπου μπορεί το optimistic offline lock να συμβάλλει αρνητικά στην εφαρμογή είναι όταν υπάρχουν πολλές ταυτόχρονες συναλλαγές και η πιθανότητα σύγκρουσης είναι μεγάλη.

Εφαρμογές:

Οι εφαρμογές που είναι καταλληλότερες να υποστηρίξει αυτό το μοτίβο, είναι αυτές που η πιθανότητα σύγκρουσης μεταξύ συναλλαγών είναι χαμηλή.

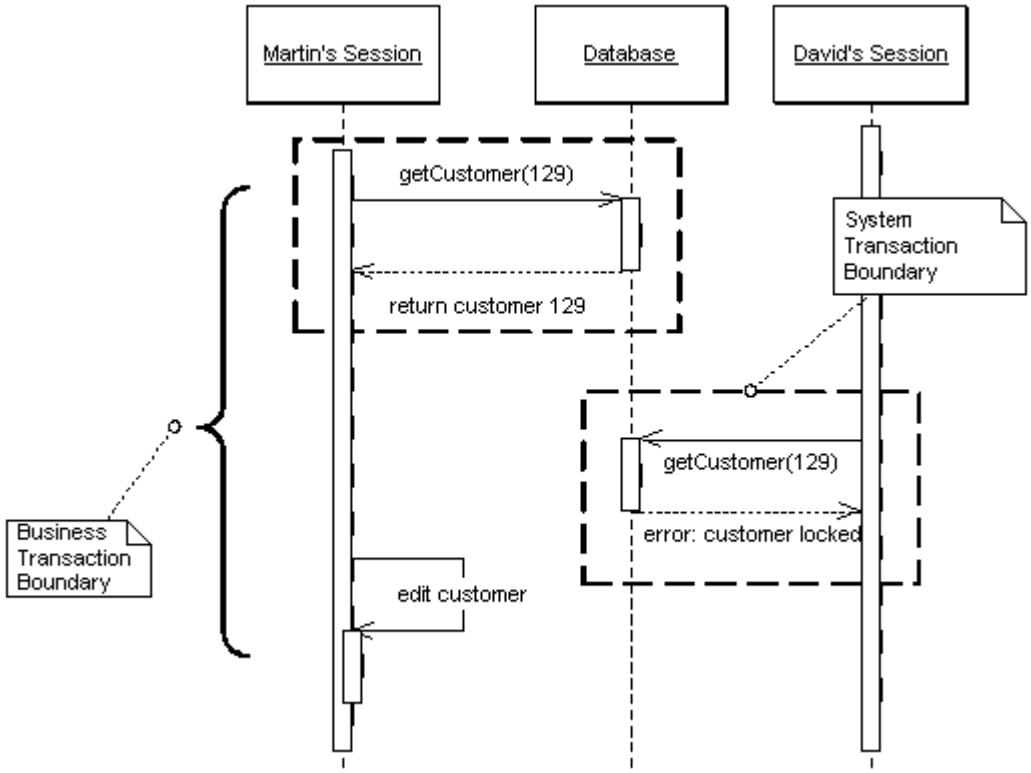
Βιβλιογραφία:

Book **Introduction to Concurrency in Programming Languages** by **Matthew J. Sottile, Timothy G. Mattson, Graig E Rasmussen,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.8.2 Pessimistic Offline Lock (by David Rice)

Αποτρέπει τις συγκρούσεις μεταξύ επιχειρησιακών συναλλαγών που γίνονται ταυτόχρονα επιτρέποντας μόνο μία επιχειρησιακή συναλλαγή κάθε φορά να έχει πρόσβαση στα δεδομένα.



Σκοπός:

Ο σκοπός του pessimistic offline lock είναι ίδιος με το προηγούμενο μοτίβο αλλά το κάνουν με διαφορετικό τρόπο.

Δομή:

Η δομή του μοτίβου αυτού είναι πάλι ίδια με το προηγούμενο μοτίβο, δηλαδή λειτουργεί πάνω στο session μιας συναλλαγής.

Πως λειτουργεί:

Σε αντίθεση με το optimistic offline lock(3.8.1), αυτή τη φορά η λειτουργία του μοτίβου βασίζεται στο ότι η πιθανότητα να γίνει σύγκρουση μεταξύ ταυτόχρονων συναλλαγών είναι μεγάλη. Οπότε αυτή τη φορά όταν ένας πελάτης θέλει να κάνει μία συναλλαγή και στέλνει κάποιο αίτημα για να αντλήσει δεδομένα από τη βάση δεδομένων, θα του επιτραπεί να συνεχίσει μόνο αν δεν υπάρχει κάποιο άλλο αίτημα άλλου πελάτη που έχει ήδη πρόσβαση στα δεδομένα. Αυτό επιτυγχάνεται δημιουργώντας ένα πεδίο στα αντικείμενα στη βάση δεδομένων που συνήθως

ονομάζεται lock και είναι αρχικοποιημένο με το γράμμα N. Εάν κάποιο session κάποιου πελάτη ζητήσει πρόσβαση σε ένα αντικείμενο από τη βάση δεδομένων τότε θα του επιστραφούν τα δεδομένα της σειράς του αντικειμένου που ζητήθηκαν και μαζί το νέο πεδίο εφόσον είναι N, αλλά αυτή τη φορά μόλις επιστρέψει στη βάση δεδομένων τα δεδομένα, το πεδίο lock του αντικειμένου αυτού θα γίνει Y. Έτσι, ο επόμενος πελάτης που θα ζητήσει πρόσβαση στα ίδια στοιχεία δε θα του επιτραπεί γιατί το lock θα είναι Y. Ουσιαστικά το μοτίβο είναι σαν να προφυλάσσεται από την αρχή για τη χειρότερη περίπτωση του, δηλαδή του να γίνει σύγκρουση μεταξύ ταυτόχρονων συναλλαγών.

Πλεονεκτήματα και μειονεκτήματα:

Η αποτροπή πολλαπλών ανατροπών και συγκρούσεων σε εφαρμογές με υψηλή πιθανότητα να συμβεί είναι το μεγαλύτερο όφελος που δίνει αυτό το μοτίβο. Επιπλέον, κατά κάποιο τρόπο μειώνει και το κόστος από ενέργειες που θα έπρεπε να ανατραπούν μετά.

Ένα βασικό μειονέκτημα του μοτίβου είναι ότι είναι λίγο περίπλοκο στην εφαρμογή του σε σχέση με το προηγούμενο και μπορεί να προκαλέσει και προβλήματα στη σύνδεση με τα δεδομένα.

Εφαρμογές:

Σε αντίθεση με το optimistic offline lock, το pessimistic offline lock είναι καταλληλότερο να υποστηρίξει εφαρμογές που η πιθανότητα σύγκρουσης μεταξύ συναλλαγών είναι υψηλή.

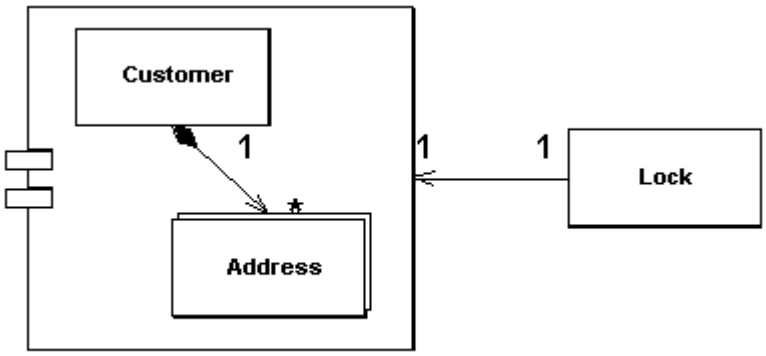
Βιβλιογραφία:

Book **Introduction to Concurrency in Programming Languages** by **Matthew J. Sottile, Timothy G. Mattson, Graig E Rasmussen,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.8.3 Coarse-Grained Lock (by David Rice and Matt Foemmel)

Κλειδώνει ένα σύνολο σχετικών αντικειμένων με μία μόνο κλειδαριά.



Σκοπός:

Σκοπός του coarse-gained lock είναι να διασφαλίσει ότι θα προστατέψει και σύνολα αντικειμένων που σχετίζονται με ένα αντικείμενο που έχει ανακτηθεί.

Δομή:

Δομικά πάλι λειτουργεί μέσα από πεδία των αντικειμένων της βάσης δεδομένων και κάνει ελέγχους μέσα στα sessions των συναλλαγών των πελατών.

Πως λειτουργεί:

Ο τρόπος που λειτουργεί αυτό το μοτίβο είναι πάνω στα προηγούμενα μοτίβα. Παίρνει μία κύρια κλάση και την κάνει ρίζα όπου μόνο από εκεί έχουν πρόσβαση οι πελάτες στο αντικείμενο και ακόμα έχει όρια όπου ορίζουν τι περιλαμβάνει το σύνολο. Μέσα στο σύνολο βρίσκονται κλάσεις ή αντικείμενα που σχετίζονται άμεσα με την κλάση ρίζα. Κάθε αντικείμενο από αυτά έχει ένα επιπλέον πεδίο στη βάση δεδομένων, το lock, όπου όπως προηγουμένως όταν δεν έχει δεσμευτεί από κάποιον πελάτη είναι N ενώ όταν χρησιμοποιείται είναι Y. Η ουσία αυτού του μοτίβου είναι ότι αυτή τη φορά όταν αλλάξει ένα αντικείμενο μέσα σε αυτό το σύνολο, τότε όλα τα αντικείμενα μέσα στο σύνολο έχουν στο πεδίο του lock Y. Αντίστοιχα μπορεί να γίνει με το πεδίο των εκδόσεων στο optimistic offline lock(3.8.1) μοτίβο.

Πλεονεκτήματα και μειονεκτήματα:

Το θετικό του μοτίβου είναι ότι υπάρχουν λιγότερα αντικείμενα που κλειδώνουν και λιγότερα πεδία έκδοσης για να γεμίσει τους πίνακες με τον τρόπο που λειτουργεί και έτσι απλοποιεί τη διαδικασία κλειδώματος.

Το βασικό μειονέκτημα του είναι ότι μπορεί να δημιουργηθούν φανταστικές σχέσεις μεταξύ αντικειμένων και αυτό να δημιουργήσει μία σύγχυση μακροπρόθεσμα.

Εφαρμογές:

Οι εφαρμογές που είναι κατάλληλο να υλοποιηθεί αυτό το μοτίβο είναι οι εφαρμογές που έχουν κάποιες σημαντικές συσχετίσεις με ένα κύριο αντικείμενο, όπως για παράδειγμα κάποιοι χρήστες όπου η εγγραφή τους σχετίζεται με διάφορες διευθύνσεις και διάφορους τραπεζικούς λογαριασμούς ή κάτι σχετικό με αυτό.

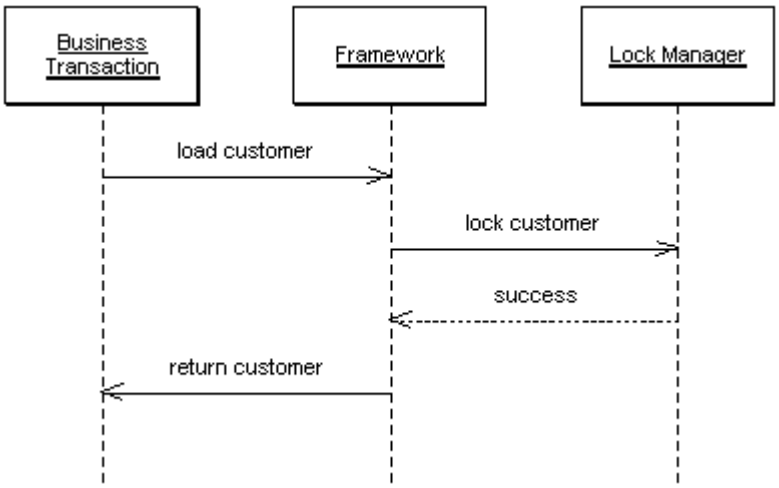
Βιβλιογραφία:

Book **Introduction to Concurrency in Programming Languages** by **Matthew J. Sottile, Timothy G. Mattson, Graig E Rasmussen,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.8.4 Implicit Lock (by David Rice)

Επιτρέπει στη δομή του πλαισίου ή στον κώδικα του στρώματος supertype να αποκτήσουν κλειδώματα εκτός σύνδεσης.



Σκοπός:

Ο σκοπός αυτού του μοτίβου είναι να διασφαλίζει ότι δε θα ξεχαστεί κανένα κλείδωμα κλειστό μετά την ολοκλήρωση μιας επιχειρησιακής συναλλαγής και ακόμα να εξασφαλίσει ότι και στο πλαίσιο της εφαρμογής θα παραμείνει κλειδωμένο το αντικείμενο που έχει κληθεί για ενέργειες από κάποιον πελάτη.

Δομή:

Η δομή του προσαρμόζεται πάνω στο framework(πλαίσιο) της εφαρμογής και μέσω μιας κλάσης, που ενημερώνεται από τις ενέργειες που γίνονται στα αντικείμενα

στην βάση δεδομένων από τις συναλλαγές, ώστε να κρατάει ενήμερη την κατάσταση της κλειδαριάς κάθε αντικειμένου.

Πως λειτουργεί:

Το implicit lock λειτουργεί πάνω στο πλαίσιο της εφαρμογής ενώ χρησιμοποιείται optimistic/pessimistic offline lock(3.8.1/3.8.2) πάνω στους διαλόγους των πελατών με τη βάση δεδομένων. Ενημερώνει μία κλάση που κρατάει σε μία λίστα τα καθήκοντα των αντικειμένων που είναι δεσμευμένα από κάποιον πελάτη είτε μέσω του πεδίου της έκδοσης είτε μέσω του πεδίου lock, αναλόγως το εφαρμοσμένο μοτίβο. Σε αυτή τη κλάση, που αφορά τα αντικείμενα πάνω στο πλαίσιο της εφαρμογής, υπάρχουν πάλι τα αντίστοιχα πεδία όπου εάν για παράδειγμα ζητηθεί η ανάκτηση δεδομένων για μία επιχειρησιακή συναλλαγή, θα κλειδώσει τα αντίστοιχα αντικείμενα και καθήκοντα αυτών πάνω στο επίπεδο του πλαισίου της εφαρμογής και μετά θα επιτραπεί η ανάκτηση των δεδομένων εφόσον είναι διαθέσιμο το αντικείμενο να ανακτηθεί. Στη συνέχεια όταν ολοκληρωθεί η ενέργεια, η κλειδαριά του αντικειμένου και όσων σχετίζονται με αυτό και είναι κλειδωμένα, ελευθερώνονται και στο επίπεδο του πλαισίου της εφαρμογής. Αυτό εξυπηρετεί ακόμα και στο να μη ξεχαστεί κάποια κλειδαριά κλειδωμένη σε κάποιο αντικείμενο όπου έχει ολοκληρωθεί η ενέργεια του διότι αυτή η κλάση ασχολείται αποκλειστικά με το να ελέγχει να γίνεται η σωστή ενημέρωση.

Πλεονεκτήματα και μειονεκτήματα:

Το σημαντικότερο πλεονέκτημα που προσφέρεται με αυτή την υλοποίηση είναι ότι επιτυγχάνεται η ασφάλεια των κλειδωμάτων των αντικειμένων από τον κώδικα του πελάτη όπου θα μπορούσαν σε κάποιες περιπτώσεις να τις απενεργοποιήσει. Οπότε προσφέρει και μεγαλύτερη ασφάλεια και διευκόλυνση για τη διαχείριση της ασφάλειας από τον κώδικα του πελάτη.

Το αρνητικό του implicit lock είναι ότι επειδή είναι μία αφαιρετική διαδικασία μπορεί μέσω κάποιου συνδυασμού αντικειμένων να υπάρξει αδιάκοπη αφαίρεση, αλλά αυτό μας επισημάνει ότι πρέπει να γίνει καλή και συμπαγή υλοποίηση των μοτίβων για να μην υπάρξουν τέτοιες περιπτώσεις.

Εφαρμογές:

Η λειτουργία του μοτίβου είναι χρήσιμη για όλες τις επιχειρησιακές και μη εφαρμογές, εκτός απλών εφαρμογών που δεν διαθέτουν framework (πλαίσιο). Είναι κοινό και σημαντικό θέμα για όλες τις εφαρμογές να υπάρξουν ξεχασμένες κλειδαριές, οπότε θεωρείται απαραίτητη η υλοποίηση του.

Βιβλιογραφία:

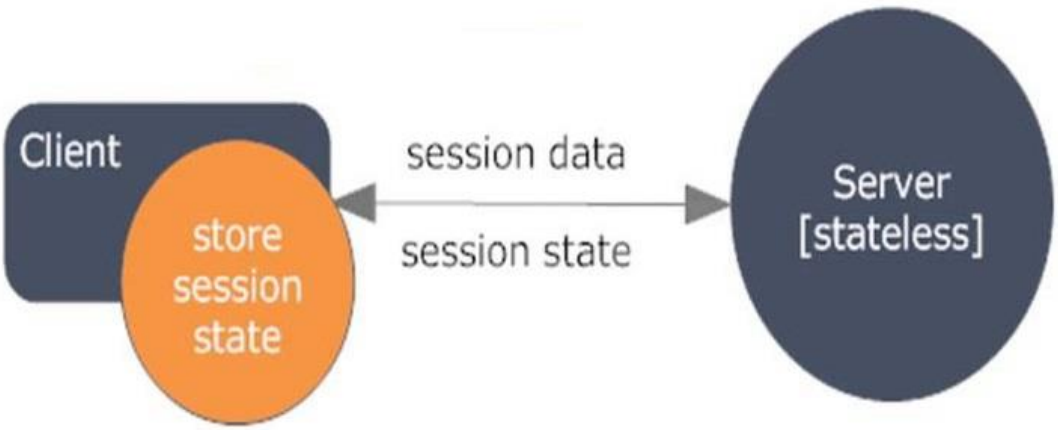
Book **Introduction to Concurrency in Programming Languages** by **Matthew J. Sottile, Timothy G. Mattson, Graig E Rasmussen,**

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.9 Session State Patterns

3.9.1 Client Session State

Αποθηκεύει κατάσταση session (διαλόγου) στον πελάτη.



Σκοπός:

Σκοπός του client session state είναι σε κάθε διάλογο που κάνει ένας πελάτης με έναν διακομιστή να κρατάει κάποιο αρχείο διαλόγου ο ίδιος ο πελάτης, επιτυγχάνοντας καλύτερη επικοινωνία μεταξύ τους και δίνοντας περισσότερες δυνατότητες και στους δύο.

Δομή:

Η δομή του είναι ένα αρχείο που κρατάει ο πελάτης από το session, όπου μπορεί να περιέχει 3 διαφορετικά περιεχόμενα και εξαρτάται από τον τρόπο υλοποίησης του μοτίβου που θα δούμε παρακάτω.

Πως λειτουργεί:

Η κύρια λειτουργία αυτού του αρχείου είναι στη σύνδεση του session με τον διακομιστή, να σταλούν από τον πελάτη κάποια δεδομένα και ο διακομιστής να αποκριθεί με το αντίστοιχο αποθηκευμένο session για αυτό τον πελάτη. Η λειτουργία αυτού του μοτίβου γίνεται με τρείς διαφορετικούς τρόπους:

- Με παραμέτρους URL, όπου κρατιόνται σε ένα αρχείο οι διευθύνσεις URL οι οποίες συνήθως χρησιμοποιούν την κατάσταση ενός session σαν παράμετρο τους. Αυτός ο τρόπος είναι ο πιο εύκολος στην υλοποίηση του αλλά χρειάζεται να μη ξεπερνάει κάποιο όριο η διεύθυνση URL γιατί το αρχείο

χρειάζεται να είναι μικρό και επίσης υπάρχει περίπτωση σύγχυσης σε διευθύνσεις που αλλάζουν αυτόματα URL που τελευταία γίνεται συχνά.

- Με κρυφά πεδία, όπου σε αυτή τη περίπτωση μπορεί να γίνει κάποια σειριοποίηση της κατάστασης του session όταν γίνεται μία απάντηση και διαβάζεται πίσω από κάθε ερώτημα και περνιέται σε ένα κρυφό πεδίο. Το κρυφό πεδίο στέλνεται στο πρόγραμμα περιήγησης αλλά όχι στον ιστότοπο Web και με μια ετικέτα της φόρμας <INPUT type = “hidden”> μπορεί να ανακτηθεί.
- Ο πιο σύνηθες τρόπος πλέον είναι τα cookies. Όπου είναι πάλι σαν ένα κρυφό πεδίο όπως η σειριοποίηση της κατάστασης του session και αυτή τη φορά στέλνεται αυτόματα από την ιστοσελίδα γιατί πλέον βασίζονται και τα ίδια στις πληροφορίες των cookies. Υπάρχει περιορισμός στο μέγεθος που μπορεί να έχει ένα τέτοιο αρχείο.

Πλεονεκτήματα και μειονεκτήματα:

Κύριο πλεονέκτημα του client session state είναι ότι με τη χρήση του επιτυγχάνεται η μέγιστη δυνατή ευελιξία των servers (διακομιστών) γιατί μπορούν να εξυπηρετούν πολλά αιτήματα χωρίς συγκεκριμένες ομαδοποιήσεις και χωρίς απώλειες δεδομένων σε περιπτώσεις σφάλματος. Επιπλέον, συμβάλλει θετικά για την αποθήκευση των αναγνωριστικών των sessions.

Εκτός του προβλήματος στη μετάδοση της κατάστασης του session σε πρωτόκολλα όπου όλα είναι μία συμβολοσειρά, υπάρχουν και άλλα δύο μειονεκτήματα αυτού του μοτίβου και είναι τα εξής: ότι υπάρχει μία συνέπεια των δεδομένων που αποστέλλονται πίσω στον πελάτη όπου θα μπορούσε να αποτελέσει πρόβλημα ασφάλειας από κάποιον κακόβουλο χρήστη, και επιπλέον υπάρχει η περίπτωση κάποια από αυτά τα αρχεία να είναι αρκετά μεγάλα (αναλόγως τον τρόπο υλοποίησης) και να επιβαρύνουν την απόδοση της εφαρμογής.

Εφαρμογές:

Η εφαρμογή του είναι κατάλληλη για πρωτόκολλα όπου με φυσικές λειτουργίες σειριοποιούν κάθε δομή δεδομένων και για εφαρμογές που τα δεδομένα τους είναι μικρά γιατί το μοτίβο αποδίδει πολύ καλύτερα έτσι.

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.9.2 Server Session State

Διατηρεί την κατάσταση του session σε ένα σύστημα διακομιστή σε σειριακή μορφή.

Σκοπός:

Ο σκοπός του server session state είναι να διατηρήσει την κατάσταση του session ενός πελάτη στον διακομιστή, με μόνη προϋπόθεση το αναγνωριστικό του session των πελατών μέσω του οποίου ξεχωρίζουν μεταξύ τους οι διαφορετικοί πελάτες.

Δομή:

Όπως στο προηγούμενο, δομικά είναι ένα αρχείο που κρατάει τη κατάσταση του session σε σειριακή μορφή για τον διακομιστή.

Πως λειτουργεί:

Η γενική λειτουργία του μοτίβου είναι ένα αντικείμενο session να διατηρείται στη μνήμη ενός διακομιστή της εφαρμογής. Μέσω του αναγνωριστικού του session και μιας κατάλληλης χαρτογράφησης που μπορεί να το χρησιμοποιήσει σαν κλειδί του αντικειμένου, υπάρχει η δυνατότητα να φορτωθεί το αντικείμενο session κάποιου πελάτη στο διακομιστή. Αυτό είναι βέβαια μία γενική λειτουργία και για να αποφευχθούν κάποια σύνηθες προβλήματα πρέπει να γίνουν κάποιες βοηθητικές υλοποιήσεις. Κάποιες από αυτές είναι: η μορφοποίηση της κατάστασης του διαλόγου (session) όπου μπορεί να είναι δυαδική σειριοποίηση ή κάποια άλλη μορφή όπως μέσω ενός αρχείου XML και ο χώρος αποθήκευσης του session εκτός από τη μνήμη του διακομιστή της εφαρμογής μπορεί να είναι και μοιρασμένος σε διακομιστές όπου δίνει ένα τρόπο υποστήριξης συσσώρευσης και ανακατεύθυνσης.

Πλεονεκτήματα και μειονεκτήματα:

Το βασικότερο πλεονέκτημα είναι όμοιο με το θετικό όφελος του προηγούμενου μοτίβου. Δίνει τη δυνατότητα στους διακομιστές να έχουν τη μέγιστη δυνατή ευελιξία και αποδοτικότητα.

Μειονέκτημα παραμένει και σε αυτό το μοτίβο το θέμα της ασφάλειας, όπου κάποιοι κακόβουλοι χρήστες ίσως επιθυμήσουν τη παραβίαση των session κάποιων πελατών. Επιπλέον, σαν αρνητικό μπορεί να χαρακτηριστεί ότι χρειάζεται να παρθούν αποφάσεις όπως για τη διάρκεια ζωής ενός αδρανούς session.

Εφαρμογές:

Η εφαρμογή του server session state είναι γενικά απαραίτητη στις εφαρμογές γιατί είναι από τα λίγα που βοηθούν με τέτοιο τρόπο στην απόδοση των διακομιστών.

Μια άλλη εναλλακτική είναι το επόμενο μοτίβο όπου όμως στρέφεται γύρω από τη βάση δεδομένων η λειτουργία του, άρα σε εφαρμογές που δεν επιθυμούν να λειτουργήσουν με γνώμονα τη βάση δεδομένων είναι το κατάλληλο μοτίβο.

Βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.9.3 Database Session State

Αποθηκεύει τα δεδομένα του session ως δεσμευμένα δεδομένα στη βάση δεδομένων.

Σκοπός:

Στόχος του database session state είναι να επικεντρώσει τη διατήρηση της κατάστασης ενός session στην βάση δεδομένων.

Δομή:

Δομικά τα δεδομένα του session αυτή τη φορά αποθηκεύονται στη βάση δεδομένων, οπότε αυτό που προστίθεται είναι είτε ένα παραπάνω πεδίο για το session είτε ένας πίνακας που καταχωρεί τις εκκρεμότητες των δεσμεύσεων.

Πως λειτουργεί:

Η κύρια λειτουργία του μοτίβου είναι όταν γίνεται μία κλήση από πελάτη στον διακομιστή, τότε ο διακομιστής τραβάει τα ζητούμενα δεδομένα απευθείας από τη βάση δεδομένων και μετά κάνει τις εκάστοτε ενέργειες που ζητήθηκαν και τα αποθηκεύει πάλι απευθείας στη βάση δεδομένων. Ένας συνηθισμένος τρόπος να βρεθούν τα κατάλληλα δεδομένα ενός διαλόγου (session) είναι έχοντας για κάθε γραμμή της βάσης δεδομένων ένα πεδίο με το αναγνωριστικό του session. Έτσι, μέσω του αναγνωριστικού τραβιέται όλο το ποσό δεδομένων που αφορά το session. Ένας άλλος τρόπος είναι να δημιουργηθεί ένας πίνακας εκκρεμοτήτων. Όσα session εκκρεμούν θα αποθηκεύονται στον πίνακα εκκρεμοτήτων του αντίστοιχου κανονικού πίνακα (εκτός μιας νέας εγγραφής που γίνεται απευθείας στον κανονικό πίνακα). Αυτό για να υλοποιηθεί είναι λίγο πιο περίπλοκο γιατί χρειάζεται να γίνει και η κατάλληλη χαρτογράφηση και για αυτούς τους πίνακες. Επίσης, κατά τη διάρκεια ενός ενεργού session εάν γίνουν αλλαγές και μετά ζητηθεί επαναφορά υπάρχουν δύο τρόποι αντιμετώπισης. Είτε δεν επιτρέπεται η ακύρωση κάποιου session και στο τέλος του αιτήματος αποθηκεύονται όλες οι ενημερώσεις είτε εάν υπάρχει πίνακας εκκρεμοτήτων να μπορέσει να κάνει σε αυτούς τους πίνακες

οποιαδήποτε αλλαγή χωρίς να επηρεάσει τους κανονικούς πίνακες και να τους ενημερώσει για τις τελικές αλλαγές στο τέλος του session.

Πλεονεκτήματα και μειονεκτήματα:

Με αυτό το μοτίβο επιτυγχάνεται καλύτερα η συσσώρευση και η ανακατεύθυνση των δεδομένων του session με πιο άμεσο και κατανοητό τρόπο. Επίσης, υπάρχουν περιπτώσεις που προγραμματιστικά δίνει ευκολία στην αντιμετώπιση του χειρισμού των καταστάσεων των session.

Το σημαντικότερο μειονέκτημα είναι ότι είναι αρκετά χρονοβόρο το να τραβιούνται και να αποθηκεύονται τα δεδομένα, και ακόμα και να χρησιμοποιηθεί cache (κρυφή μνήμη) μνήμη στον διακομιστή ώστε να μη χρειάζεται να τραβιούνται συνεχώς θα χρειάζεται να γίνεται συχνά η αποθήκευση.

Εφαρμογές:

Οι εφαρμογές που είναι κατάλληλες να εφαρμοστεί αποδοτικότερο το database session state είναι στις περιπτώσεις που δεν υπάρχουν καταστάσεις session και είναι δυνατόν να αποθηκεύονται όλα τα δεδομένα σε κάθε αίτημα, δηλαδή με τη χρήση cache στον server. Επιπλέον ιδανικό είναι όταν χρειάζεται από την εφαρμογή να βασιστεί στην συσσώρευση και ανακατεύθυνση των δεδομένων.

Βιβλιογραφία:

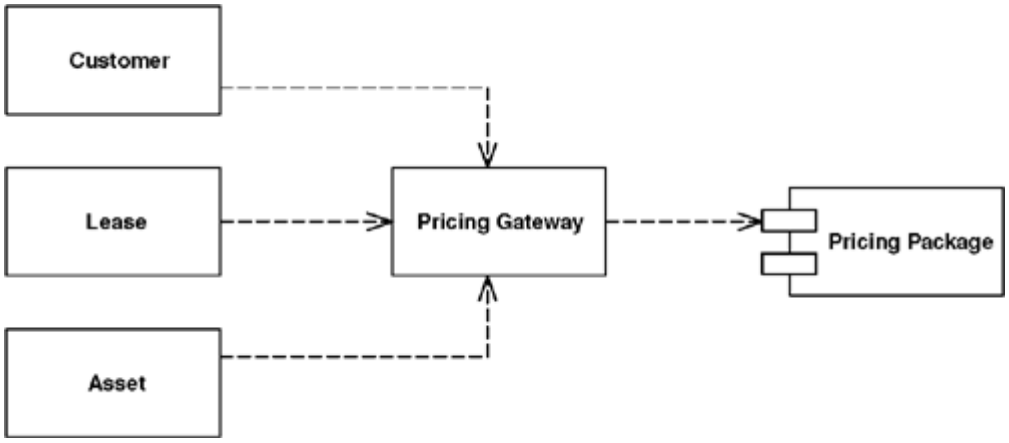
Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**.

3.10 Base Patterns

Τα παρακάτω βασικά μοτίβα θα αναφερθούν περιληπτικά γιατί είναι γενικού περιεχομένου.

3.10.1 Gateway

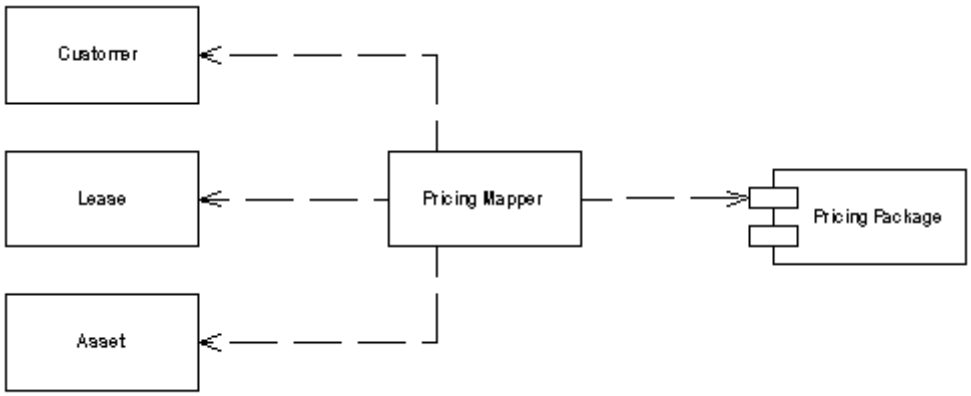
Ένα αντικείμενο που ενσωματώνει πρόσβαση σε ένα εξωτερικό σύστημα ή πόρο.



Το gateway μοτίβο χρησιμοποιείται για την επικοινωνία με εξωτερικές πηγές του συστήματος χωρίς να χρειάζεται η εφαρμογή να έχει κάποια σύνδεση για πρόσβαση σε αυτές. Η εφαρμογή Web στέλνει το αίτημα στο gateway και αυτό είναι υπεύθυνο να το μετατρέψει σε κατάλληλη μορφή ώστε να το στείλει σε μία εξωτερική πηγή. Στη συνέχεια, όταν απαντηθεί το ερώτημα από την εξωτερική πηγή στο gateway, ξαναμετατρέπει το μοτίβο την απάντηση με κατάλληλη μορφοποίηση, όπου μπορεί η εφαρμογή να την καταλάβει. Για την αποθήκευση ή ανάκτηση δεδομένων το μοτίβο μπορεί να χρησιμοποιήσει κάποιο στρώμα που τρέχει παράλληλα από πίσω για να προσφέρει αυτή τη δυνατότητα.

3.10.2 Mapper

Ένα αντικείμενο που δημιουργεί μια επικοινωνία μεταξύ δύο ανεξάρτητων αντικειμένων.

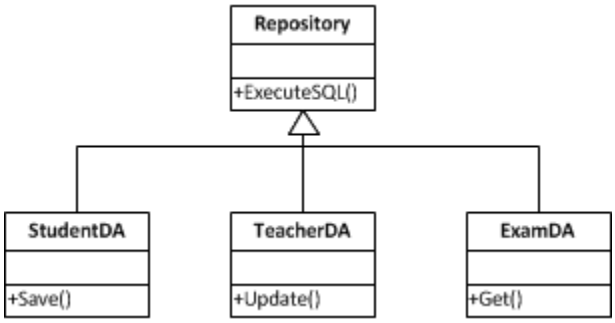


Το mapper είναι η βασική διαδικασία κάποιας χαρτογράφησης δύο αντικειμένων. Κάποιες φορές χρειάζεται να επικοινωνήσουν δύο υποσυστήματα μεταξύ τους, αλλά επίσης χρειάζεται να διατηρηθεί η άγνοια της ύπαρξης του ενός από του άλλου. Αυτό μπορεί να συμβαίνει γιατί δε γίνεται να μετατραπούν αυτά τα αντικείμενα ή γίνεται αλλά δε πρέπει να υπάρξει εξάρτηση μεταξύ τους. Αυτό μπορεί να επιτευχθεί με τη χρήση αυτού του μοτίβου, όπου λειτουργεί σαν

μεσολαβητής μεταξύ αυτών υποσυστημάτων και ελέγχει τις λεπτομέρειες της επικοινωνίας αυτών χωρίς να χρειάζεται να ανησυχούν για κάτι διότι ουσιαστικά δημιουργείται ένα απομονωμένο στρώμα μεταξύ των υποσυστημάτων. Επιπλέον, έχει τη δυνατότητα να αναμιγνύει ή να μετακινεί τα δεδομένα από το ένα στρώμα στο άλλο.

3.10.3 Layer Supertype

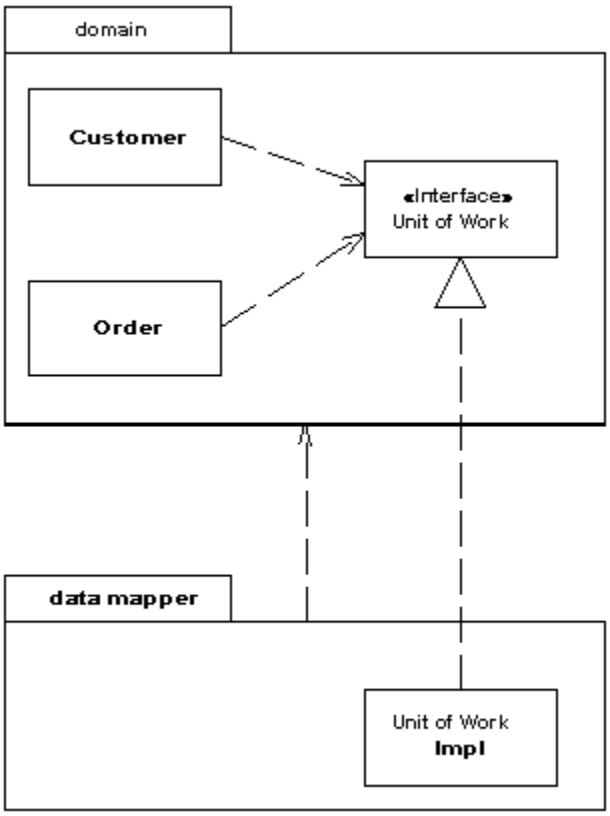
Ένας τύπος που ενεργεί ως supertype για όλους τους τύπους στο στρώμα του.



Το layer supertype μοτίβο έχει πολύ απλή λογική και εξυπηρετεί στο να μην γράφεται διπλός κώδικας για κάποιο τύπο ή μέθοδο που επαναχρησιμοποιείται. Αυτό το επιτυγχάνει, όπως φαίνεται και στο σχήμα, μετακινώντας κοινές συμπεριφορές από διάφορες κλάσεις στο ίδιο στρώμα σε ένα ενιαίο supertype ώστε να βελτιώσει την επαναχρησιμοποίηση των συμπεριφορών.

3.10.4 Separated Interface

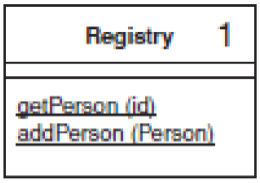
Ορίζει μια διεπαφή σε ξεχωριστό πακέτο από την εφαρμογή της.



Το separated interface μοτίβο ορίζει μία διεπαφή σε ένα πακέτο, αλλά την υλοποίηση του και την εφαρμογή του σε άλλο πακέτο. Με αυτό το τρόπο όταν ένας πελάτης χρειάζεται εξάρτηση από τη διεπαφή μπορεί να αγνοεί εντελώς την εφαρμογή του. Η υλοποίηση της διεπαφής μπορεί να παρέχεται σε compile time ή σε runtime μέσω σύνδεσης.

3.10.5 Registry

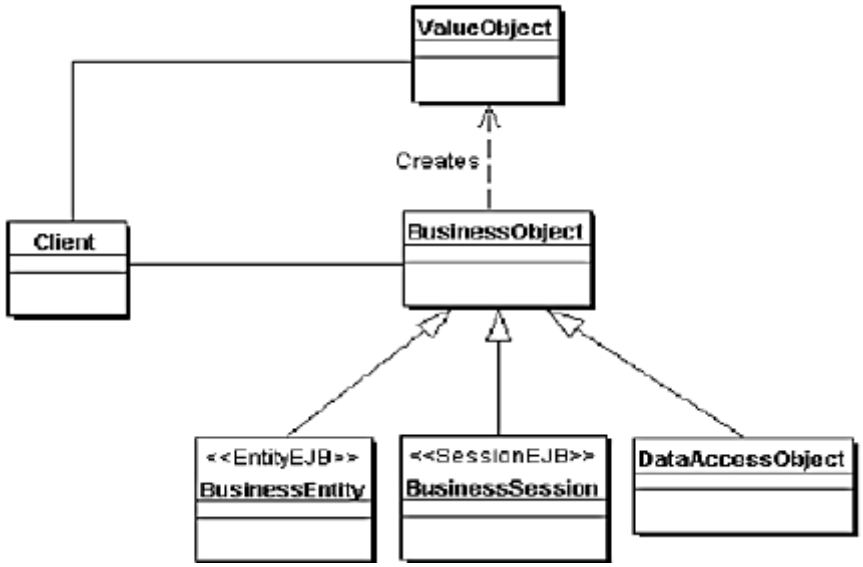
Ένα πολύ γνωστό αντικείμενο που μπορούν να χρησιμοποιήσουν άλλα αντικείμενα για να βρουν κοινά αντικείμενα και υπηρεσίες.



Μία εγγραφή είναι απαραίτητο να είναι καθολικό αντικείμενο και γενικά οι εγγραφές είναι υλοποιημένες σαν μοναδικές υπάρξεις. Το registry μοτίβο είναι ένα αντικείμενο, όπως φαίνεται και στο σχήμα παραπάνω, γνωστό σε όλα τα αντικείμενα και μπορούν να χρησιμοποιηθούν οι μέθοδοι για να βρεθούν κοινά αντικείμενα και για να εκτελεστούν υπηρεσίες, όπως είναι το getPerson ή addPerson στο σχήμα, από άλλα αντικείμενα. Αυτό εξυπηρετεί πάλι στην επαναχρησιμοποίηση κοινών συμπεριφορών.

3.10.6 Value Object

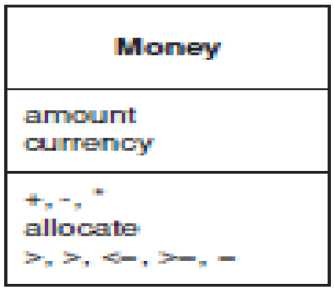
Ένα μικρό απλό αντικείμενο, όπως τα χρήματα ή ένα εύρος ημερομηνιών των οποίων η ισότητα δεν βασίζεται στην ταυτότητα.



Το value object μοτίβο αφορά τα μικρά αντικείμενα όπου αναπαριστούν μία μικρή οντότητα όπου η ισότητα τους εξαρτάται από την αξία τους και όχι από την ταυτότητα τους. Δηλαδή, δύο αξίες αντικειμένων είναι ίσες όταν έχουν την ίδια τιμή, χωρίς να είναι απαραίτητο να είναι τα ίδια αντικείμενα. Οπότε, αυτό το μοτίβο διαχειρίζεται ξεχωριστά αυτά τα αντικείμενα και τα διαχωρίζει από τα άλλα όπως φαίνεται και στο σχήμα, γιατί σε αντίθεση με τα αντικείμενα αναφοράς όπου εξαρτάται η ισότητα τους από την ταυτότητα τους σε αυτά εξαρτάται από την τιμή ή την αξία τους.

3.10.7 Money

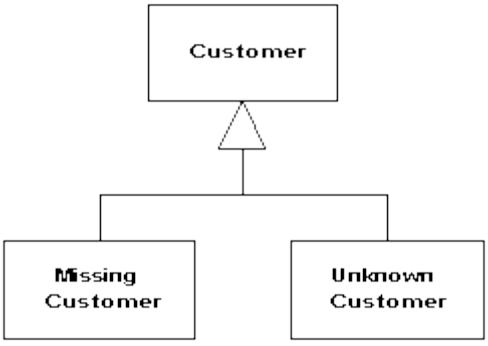
Αντιπροσωπεύει μια νομισματική αξία.



Το μοτίβο money βασίζεται στο μοτίβο value object(3.10.6) αλλά η ανάγκη του να αντιμετωπιστεί η μεταβλητή των χρημάτων με ξεχωριστό τρόπο έχει γίνει μεγάλη. Οι περισσότερες εφαρμογές διαχειρίζονται χρήματα και πολλές φορές αυτό χρειάζεται λεπτή αντιμετώπιση γιατί τα χρήματα έχουν και διαφορετικά νομίσματα με διαφορετικές αξίες. Αυτό το μοτίβο προσπαθεί να λύσει αυτά τα προβλήματα κάνοντας τις κατάλληλες πράξεις, στρογγυλοποιήσεις και μετατροπές νομισμάτων όποτε χρειάζεται, αντιμετωπίζοντας το σαν ξεχωριστό είδος μεταβλητής. Είναι πολύ χρήσιμο μοτίβο πόσο μάλλον και σε αυτή την εποχή που υπάρχει και μεγάλη κίνηση ηλεκτρονικού χρήματος.

3.10.8 Special Case

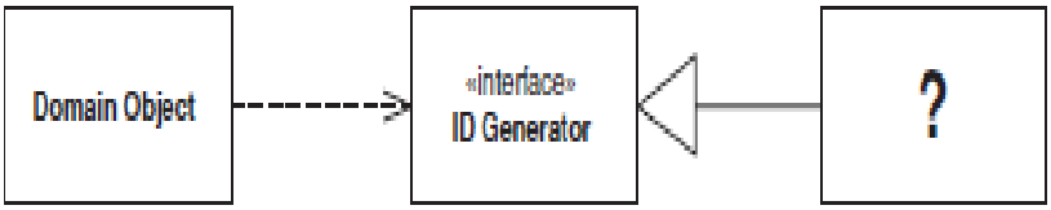
Μια υποκατηγορία που παρέχει ειδική συμπεριφορά για συγκεκριμένες περιπτώσεις.



Το μοτίβο special case είναι πολύ απλό στην εφαρμογή του και η λειτουργία του έχει να κάνει με ειδική συμπεριφορά για κάποιες ξεχωριστές ιδιαίτερες περιπτώσεις. Αυτό το μοτίβο μπορεί να συμβάλλει ουσιαστικά με τέτοιο τρόπο στη διαχείριση των αιτημάτων ώστε να προβλέψει την αντιμετώπιση κάποιων ενδεχόμενων εξαιρέσεων που μπορούν να προκύψουν.

3.10.9 Plugin (by David Rice and Matt Foemmel)

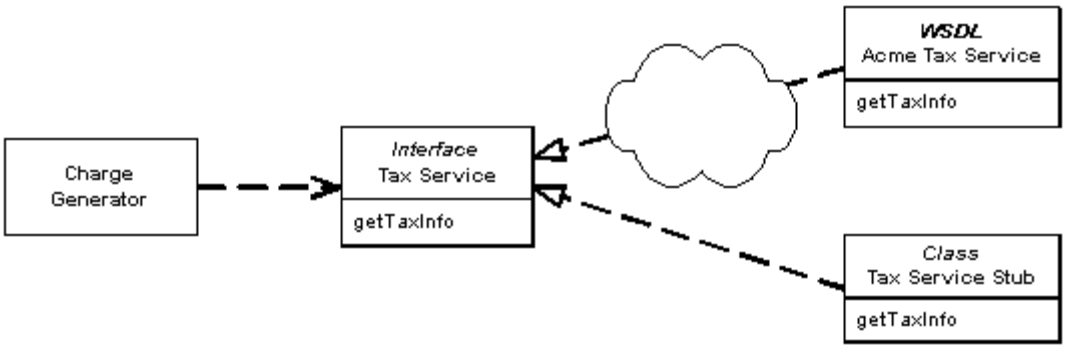
Συνδέει κλάσεις κατά τη διάρκεια της διαμόρφωσης αντί κατά τη διάρκεια της σύνταξης.



Το plugin μοτίβο διαχωρίζει περιπτώσεις συμπεριφοράς όταν ο κώδικας της εφαρμογής τρέχει σε χρόνο εκτέλεσης πολλαπλών περιβαλλόντων, όπου σε κάθε ένα από τα περιβάλλοντα όταν υπάρχει αίτημα εφαρμόζεται διαφορετικά η κάθε συμπεριφορά. Ουσιαστικά, αυτό το μοτίβο για μία διεπαφή κάνει τις κατάλληλες συνδέσεις κλάσεων αναλόγως του περιβάλλοντος που τρέχει η εφαρμογή.

3.10.10 Service Stub (by David Rice)

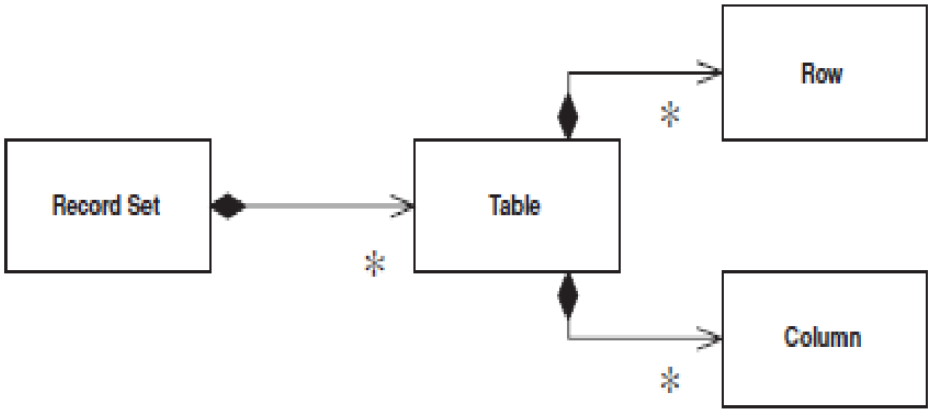
Καταργεί την εξάρτηση από τις προβληματικές υπηρεσίες κατά τη διάρκεια των δοκιμών.



Το service stub μοτίβο χρησιμοποιείται σε περιπτώσεις που κάποια από τις αληθινές υπηρεσίες είναι προβληματικές, έχουν υλοποιηθεί μέσω του μοτίβου υπηρεσίες που τις αντικαθιστούν αλλά με γενικότερο τρόπο, για να υπάρξει μία κατάλληλη απάντηση στο πρόβλημα. Συνήθως η λειτουργία του βασίζεται πάνω στο μοτίβο gateway(3.10.1).

3.10.11 Record Set

Μια αναπαράσταση σε μνήμη των πινακοποιημένων δεδομένων.



Το record set μοτίβο δημιουργεί στη μνήμη την αναπαράσταση ενός πίνακα της βάσης δεδομένων. Αυτό μπορεί να συντελέσει στην πιο άμεση εκτέλεση των ερωτημάτων της εφαρμογής γιατί μπορεί να αντλήσει τα δεδομένα μέσω αυτού του αρχείου αναπαράστασης. Συνήθως αυτή η δυνατότητα υλοποίησης αυτού του μοτίβου δίνεται άμεσα από τις πλατφόρμες λογισμικού για την δημιουργία της εφαρμογή.

Γενική βιβλιογραφία:

Book **Patterns of Enterprise Application Architecture** by **Martin Fowler**,

Book **SOA Design Patterns** by **Thomas Erl,**

Book **Patterns for e-business, A Strategy for Reuse** by **Jonathan Adams, Srinivas Koushik, Guru Vasudeva, George Galambos.**

4. Επίλογος

Συμπερασματικά, όλες οι πληροφορίες που αντλήθηκαν από την ανάλυση των παραπάνω μοτίβων μας δίνει μία συνολική εικόνα πως μπορεί ένας προγραμματιστής να στήσει μία επιχειρησιακή εφαρμογή, χρησιμοποιώντας αρχιτεκτονικές τεχνοτροπίες, όπου θα διευκολύνουν και θα προσφέρουν καλύτερα τη λειτουργικότητα, την απόδοση και τη συντήρηση αυτής της εφαρμογής. Σε κάθε

δυσκολία ή σε κάθε σύνθετη κατάσταση, μπορεί να δομηθεί και να υλοποιηθεί ένα μοτίβο, όπου η λειτουργία του θα ξεπερνά το πρόβλημα αυτής της κατάστασης. Με τη συσσώρευση της γνώσης για αυτές τις τεχνοτροπίες υπάρχει η δυνατότητα της εξέλιξης των ίδιων ή της δημιουργίας νέων μοτίβων για την αντιμετώπιση νέων ή μέχρι τώρα άλυτων προβλημάτων.

Η δύναμη της κατανόησης των όσων περιγράφηκαν, έχει ως αποτέλεσμα την ελπίδα για το μέλλον πάνω σε αυτό το τομέα, που φαίνεται να έχει χώρο για πολλά νέα ευρήματα. Σε τούτη την εργασία ενισχύεται η ελπίδα αυτή, έχοντας πλέον μία κατανόηση και μία λίστα μοτίβων, όπου μπορεί να επιλεγούν τα κατάλληλα μοτίβα ώστε να αντιμετωπιστούν τα αντίστοιχα προβλήματα.

Όσα αναφέρθηκαν δίνουν λύσεις πλέον σε πολλούς τομείς προβλημάτων. Υπάρχουν αρχιτεκτονικές και σχεδιαστικές τεχνοτροπίες όπου μπορούν να αντιμετωπίσουν θέματα βέλτιστης εφαρμογής λογικής στο κύριο μέρος του προγράμματος. Υπάρχουν μοτίβα που δίνουν λύσεις για βέλτιστη διαχείριση πηγών δεδομένων, αντικειμενοσχεσιακών συμπεριφορών και αντικειμενοσχεσιακών κατασκευών. Υπάρχει γνώση για το που μπορεί να αναζητήσει κανείς τρόπους καλύτερης λειτουργικότητας των διαδικτυακών απεικονίσεων ή των μεταδεδομένων και της διανομής οποιονδήποτε δεδομένων μέσα στην εφαρμογή. Για θέματα χαρτογράφησης, εσωτερικής διακριτικής συγχρονικότητας, καταστάσεων των sessions και ακόμα βασικών απλών θεμάτων όπως χρημάτων, εγγραφών, ειδικών περιπτώσεων, συνδέσεων και πολλά άλλα, υπάρχει πλέον η επιλογή για να γίνει η καλύτερη δυνατή υλοποίηση τους πάνω σε ένα συνολικό σύστημα που εξυπηρετεί ανθρώπινες και επιχειρησιακές ανάγκες.

Όλες αυτές οι πληροφορίες πλέον μπορούν να αποτελέσουν λύσεις για το παρόν και βάσεις για το μέλλον. Παρακάτω, θα γίνει το κλείσιμο της διπλωματικής εργασίας με δύο τελευταία υποκεφάλαια. Στο ένα θα περιγραφεί η γενική χρησιμότητα και η μέχρι τώρα πορεία των μοτίβων και στο άλλο θα περιγραφεί η πορεία για μελλοντικές αρχιτεκτονικές τεχνοτροπίες. Και στα δύο κομμάτια έχουν λόγο και αρχάριοι και έμπειροι προγραμματιστές, όπως και αυτή η εργασία έχει στόχο να πραγματοποιήσει την καλύτερη εισαγωγή για αρχάριους και να αποτελέσει το καλύτερο βοήθημα εξέλιξης για έμπειρους προγραμματιστές.

4.1 Η πορεία των μοτίβων μέχρι τώρα

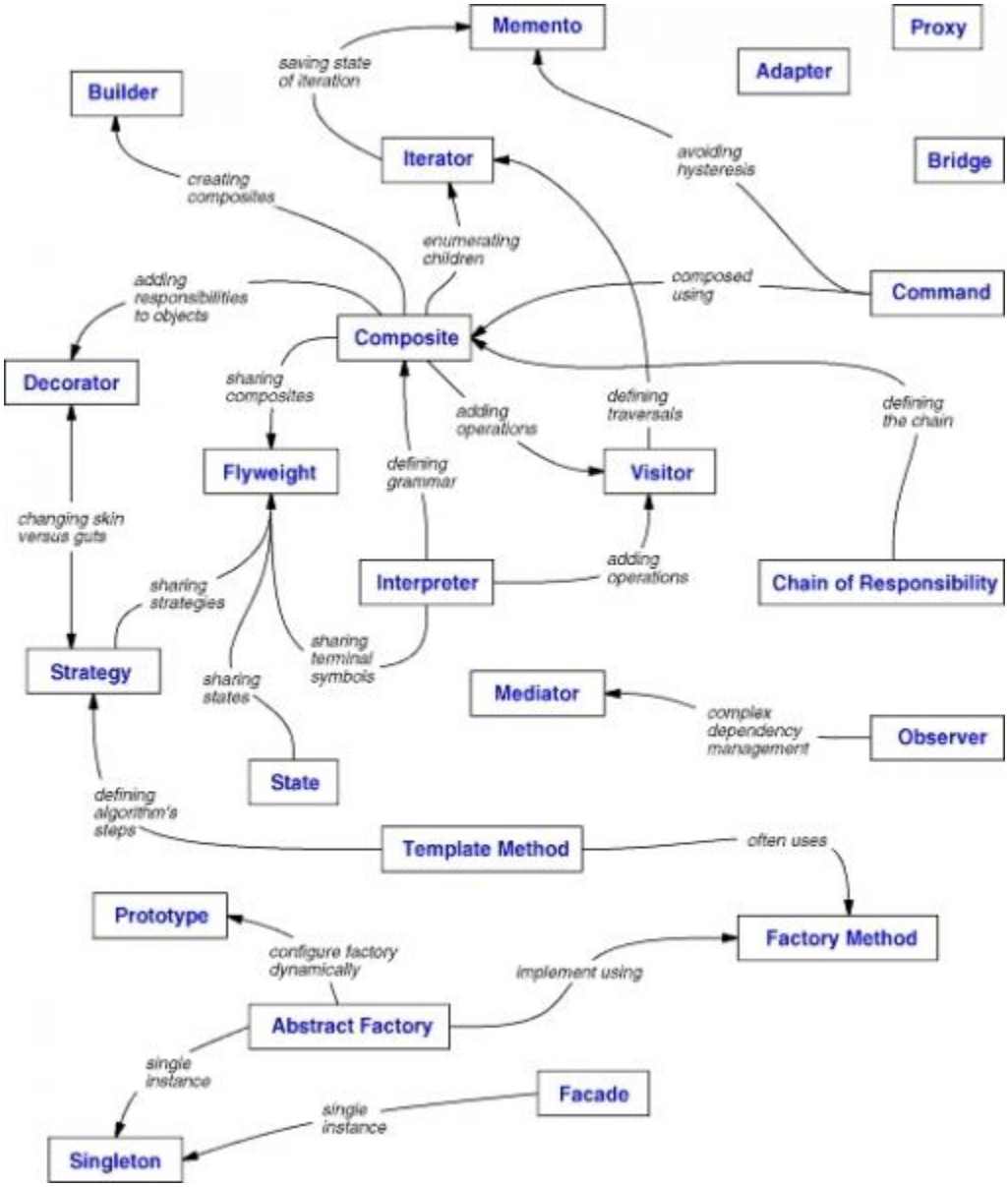
Τα μοτίβα είναι έμπρακτα στη ζωή των προγραμματιστών περίπου 20 χρόνια. Ο πρώτος εμπνευστής των αρχιτεκτονικών χτισιμάτων και της φιλοσοφίας για τη δημιουργία γλώσσας μοτίβων είναι ο Christopher Alexander. Πάνω στην φιλοσοφία του ακολούθησαν ο Kent Beck και ο Ward Cunningham, όπου δημιούργησαν την πρώτη μικρή γλώσσα μοτίβων. Άλλα σημαντικά πρόσωπα πάνω σε αυτές τις

τεχνοτροπίες και από πρακτικής άποψης και από συγγραφικής άποψης είναι ο Grady Booch, ο Erich Gamma, ο Frank Buschmann και φυσικά ο Martin Fowler.

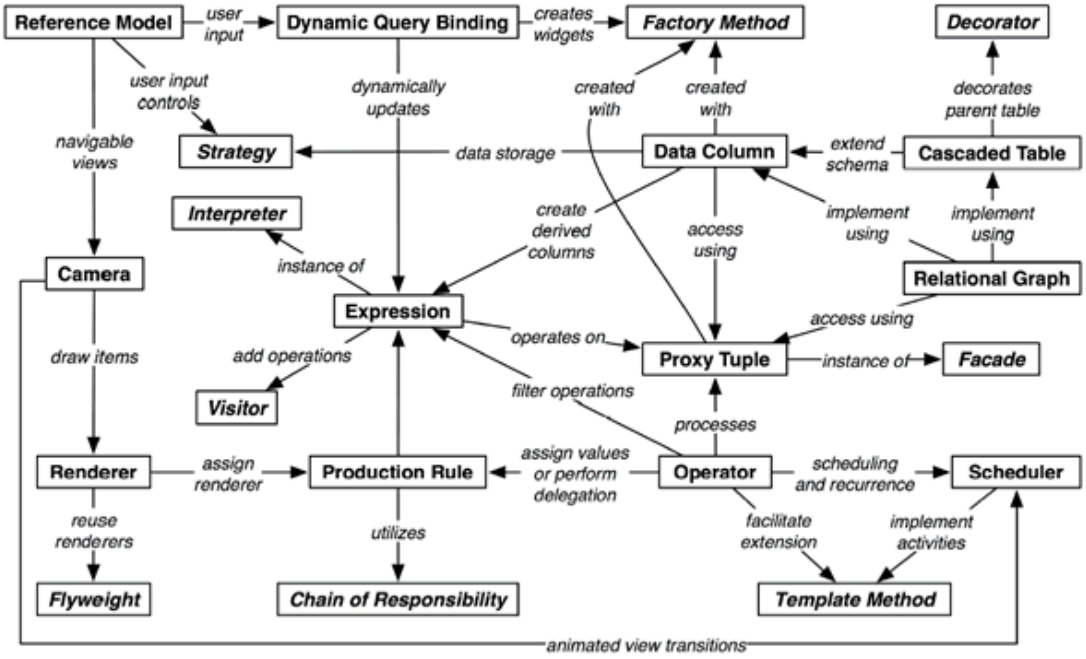
Η κύρια ανάγκη των μοτίβων πλέον μεταφράζεται σε περιπτώσεις που ένα πρόβλημα εμφανίζεται επανειλημμένα σε ένα συγκεκριμένο πλαίσιο. Αντί να χρησιμοποιείται επανειλημμένα ένα απόσπασμα κώδικα που θα γίνεται αντιγραφή επικόλληση σε κάθε σημείο που χρειάζεται, τα μοτίβα έχουν τη δύναμη να επηρεάσουν την σχεδίαση της λύσης. Τα μοτίβα έχουν συμβάλλει τόσο σχεδιαστικά όσο και αρχιτεκτονικά μέσα στα προγράμματα. Από την απλή αντιμετώπιση προβλημάτων έχουν φτάσει σε σημείο να βοηθάνε στην ευελιξία και τη δυνατότητα συντήρησης εφαρμογών έχοντας καλύτερο χειρισμό της πολυπλοκότητας και της χωρητικότητας.

Σε όλους αυτούς τους τομείς που είδαμε τα μοτίβα που εξυπηρετούν τόσες ανάγκες υπολογίζονται να είναι περίπου πάνω από 1.000. Πλέον οι περισσότερες εταιρείες όπως Yahoo, Amazon, Google, IBM, Lucent, Microsoft, Oracle, Siemens και άλλες έχουν συλλογές μοτίβων.

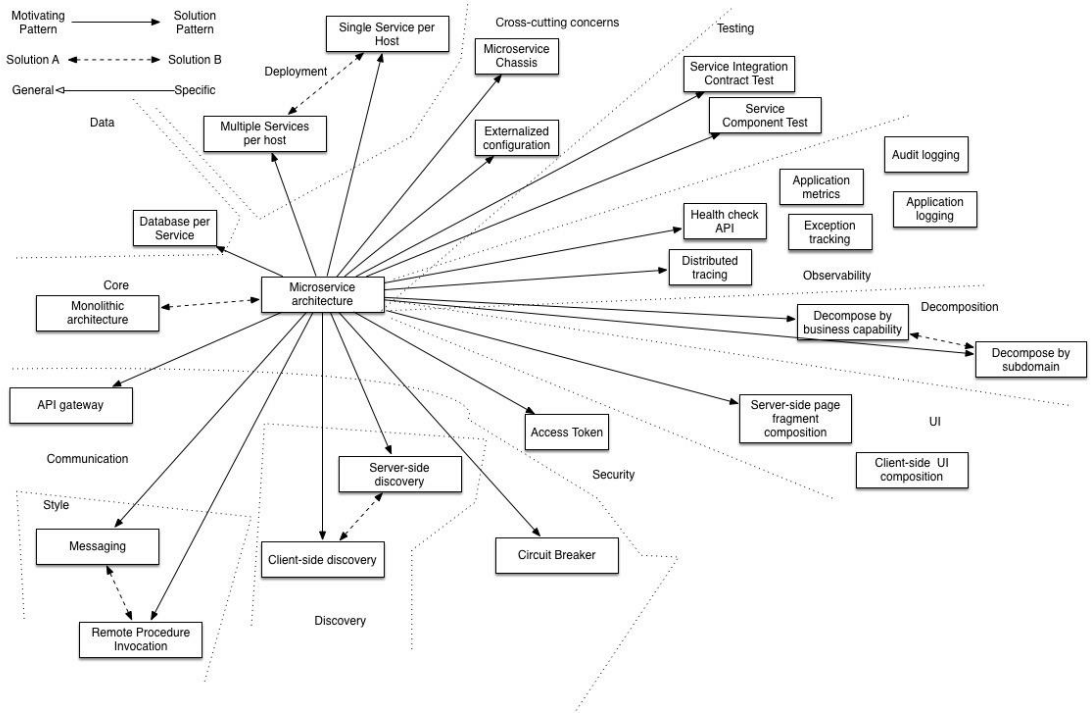
Σχήμα: $1^{η}$ απεικόνιση σχέσεων μεταξύ σχεδιαστικών μοτίβων.



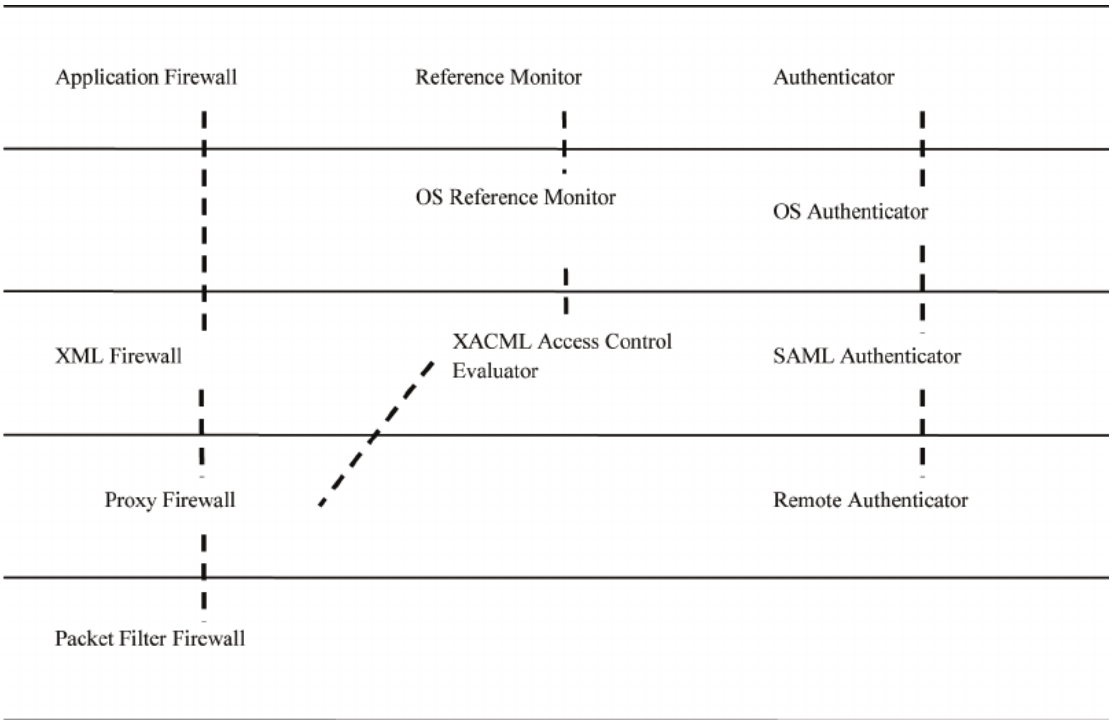
Σχήμα: 2η απεικόνιση σχέσεων μεταξύ σχεδιαστικών μοτίβων.



Σχήμα: Σχέση μοτίβων σε μικρουπηρεσίες.



Σχήμα: Τύποι μοτίβων που βασίζονται σε επίπεδα.



4.2 Η πορεία των μοτίβων από εδώ και πέρα

Η πορεία των μοτίβων από εδώ και πέρα εκτιμάτε ότι θα είναι ανοδική με γρηγορότερο ρυθμό πλέον, διότι η εξοικείωση των προγραμματιστών λογισμικού είναι μεγάλη σε σχέση με πριν από κάποια χρόνια. Είναι λίγο σχετικό και το τι μπορούμε να προβλέψουμε για μελλοντικές υλοποιήσεις πάνω στα μοτίβα. Παρακάτω θα αναφερθούν κάποιοι από τους τομείς που υπάρχουν εικασίες για εξέλιξη μοτίβων πάνω τους.

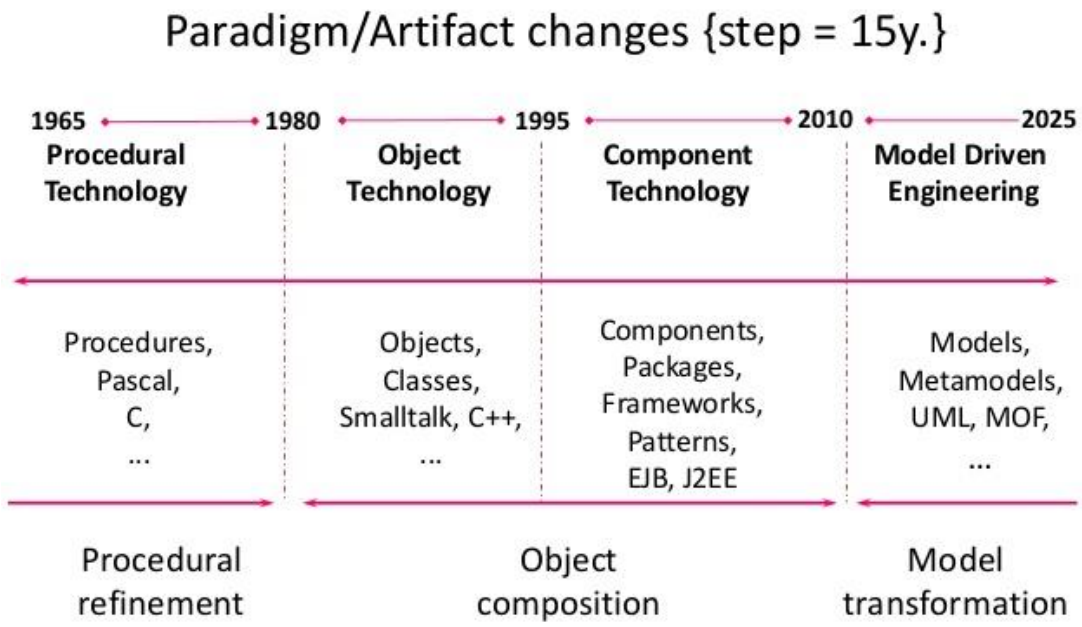
Θα παρουσιάσουμε τους τομείς σαν κάποια μορφή λίστας με μία μικρή περιγραφή:

- Τα δεδομένα ως νόμισμα. Όπως συμβαίνει και τα τελευταία χρόνια να χρησιμοποιείται ηλεκτρικό νόμισμα μέσω κάποιων δεδομένων που παράγεται από λειτουργίες υπολογιστικού συστήματος.
- Πολύγλωσσες αρχιτεκτονικές. Προβλέπεται ότι θα υπάρχει δυνατότητα να χρησιμοποιηθούν διαφόρων τύπων αποθηκευτικοί χώροι δεδομένων για όλα τα είδη συσκευών και όλο το σύνολο αυτών των δεδομένων μέσω διαφορετικών τεχνολογιών και θα λειτουργεί σε οικουμενικές συνθήκες, στο σύννεφο (cloud) και αντίστοιχη τεχνολογία.
- Οι μικρουπηρεσίες. Έχουν υπάρξει μοτίβα όπου ασχολούνται με αυτό το τομέα αλλά υπάρχει εκτίμηση ότι θα γίνουν ακόμα περισσότερα διότι δεν έχει γίνει ακόμα κατανοητός πλήρως ο χώρος των μικρουπηρεσιών.
- Ορισμένες τεχνολογίες πάνω στο SOA (Υπηρεσιοκεντρική αρχιτεκτονική), όπως η μοντελοποίηση επιχειρηματικών διαδικασιών, υπηρεσία

«ενορχήστρωσης» και συστήματα υπερμεγέθους είναι ακόμα ανεξερεύνητες και δεν καλύπτονται ακόμα από κάποια τεκμηριωμένα μοτίβα.

- Γενετικές τεχνολογίες λογισμικού. Τα μοτίβα πλέον εκτός από το να ασχολούνται μόνο με αντικειμενοστραφή λογισμικά γραμμένα από γλώσσες προγραμματισμού, εκτιμάται ότι μπορούν να υποστηρίζουν άλλα λογισμικά, σε συγκεκριμένα αναπτυγμένο αποψιοκεντρικό λογισμικό και λογισμικό καθοδηγούμενο από το μοντέλο.
- Διανεμημένα σε πραγματικό χρόνο και ενσωματωμένα συστήματα. Αυτός ο τομέας αρχίζει και σημειώνει πρόοδο και με τη συμβολή μοτίβων, μπορεί να υπάρξει πραγματική ανάπτυξη, σε υψηλής ποιότητας κατανεμημένων δεδομένων σε πραγματικό χρόνο και ενσωματωμένων συστημάτων.
- Ένας ακόμα τομέας είναι η υποστήριξη της ποιότητας της εξυπηρέτησης για εμπόριο εκτός «ραφιού» κατανεμημένων συστημάτων.
- Φυσικά ένας μεγάλος χώρος ανάπτυξης είναι τα κινητά τηλέφωνα. Υπάρχει μεγάλος χώρος εξέλιξης για τη διαχείριση του χαμηλού και μεταβλητού εύρους ζώνης και της ισχύος, τη συνδεσιμότητα και τη ποιότητα υπηρεσιών, τα διαφορετικά πρωτόκολλα και τη διατήρηση της συνοχής της κρυφής μνήμης για νήματα, ενώ κάποιο κινητό τηλέφωνο είναι αποσυνδεδεμένο από ιστότοπους.
- Ακόμα μεγαλύτερη ανάπτυξη στην αρχιτεκτονική λογισμικού, σε κλάδους που ακόμα δεν έχουν υποστηρίξει τα μοτίβα.
- Ομαδική αλληλεπίδραση. Μοτίβα για αλληλεπίδραση ανθρώπου-μηχανής-ανθρώπου.
- Web νέας γενιάς. Οι ιστότοποι αναπτύσσονται ραγδαία οπότε χρειάζονται παράλληλα υποστήριξη από νέες τεχνολογίες και σίγουρα από νέα μοτίβα.
- Συστήματα επιχειρησιακών συναλλαγών και ηλεκτρονικού εμπορίου. Είδαμε αρκετά μοτίβα σε αυτή την εργασία πάνω σε αυτό το κομμάτι αλλά υπάρχει περιθώριο για μεγάλη εξέλιξη πάνω σε αυτό το τομέα.
- Δομή οργάνωσης και διαδικασιών.

Σχήμα: Η μοντελοποίηση λογισμικού και το μέλλον της μηχανικής



Στο παραπάνω σχήμα φαίνεται η πορεία της εξέλιξης του προγραμματισμού λογισμικού και της μηχανικής, όπου ως στόχο έχει να καταδείξει που μπορούν επίσης να κοιτάξουν μελλοντικά τα μοτίβα.

5. Βιβλιογραφία:

Βιβλία:

Architecting Applications for the Enterprise by Dino Esposito and Andrea Saltarello

Business Patterns for Software Developers by Allan Kelly

Core J2EE Patterns: Best Practices and Design Strategies by Deepak Alur, Dan Malks, John Crupi

Description Logics in Multimedia Reasoning by Leslie F. Sikos

Design Patterns: Elements of Reusable Object-Oriented Software by Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Grady Booch

Domain-Driven Design: Definitions and Pattern Summaries by Eric Evans

Introduction to Concurrency in Programming Languages by Matthew J. Sottile, Timothy G. Mattson, Graig E Rasmussen

Pattern-Oriented Software Architecture: A System of Patterns, Volume 1 by Wiley

Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects, Volume 2 by Wiley

Pattern-Oriented Software Architecture: Patterns for Resource Management, Volume 3 by Wiley

Patterns of Enterprise Application Architecture by Martin Fowler

Presentation Patterns: Techniques for Crafting Better Presentations by Neal Ford, Matthew McCullough, Nathaniel Schutta

Service-oriented Architecture Compass: Business Value, Planning, and Enterprice Roadmap by Norbert Bieberstein, Sanjay Bose, Marc Fiammante, Keith Jones, Rawn Shah

SOA Design Patterns by Thomas Erl

Αναφορές:

Article The Software as a Service Delivery Model Explained posted by Omri Erel in SAAS

Article Panel on design methodology by Reid Smith (October 1987)

Article Using Pattern Languages for Object-Oriented Program by Kent Beck, Ward Cunningham (September 1987)

Article Componentization: The Visitor Example by Bertrand Meyer, Karine Arnout (July 2006)

Article Design in Construction by Steve McConnell (June 2004)

Secure Design Patterns by Chad Dougherty, Kirk Sayre, Robert C. Seacord, David Svoboda, Kazuya Togashi (2009)

Article Collection of User Interface Design Patterns by Sari A. Laakso (2003-09-16)

Link http://wiki3.cosc.canterbury.ac.nz/index.php/Transform_view_pattern

Link https://docs.jboss.org/hibernate/orm/3.5/reference/en/html/inheritance.html about *Hibernate ORM documentation (5.0), Chapter 9. Inheritance mapping*, Retrieved November 3, 2015

Link https://msdn.microsoft.com/en-us/library/ff649595.aspx

Link https://msdn.microsoft.com/en-us/library/ms978717.aspx