Πολυτεχνείο Κρήτης

Τμήμα Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών

Τομέας Πληροφορικής

Σχεδιασμός και Ανάπτυξη Σημασιολογικού Πληροφοριακού Συστήματος με Χρήση Οντολογίας και UML

ΧΡΗΣΤΟΣ Μ. ΨΩΡΟΜΗΤΑΣ

Χανιά 2009

Περίληψη

Ο Παγκόσμιος Ιστός (World Wide Web) έχει αλλάξει τον τρόπο που οι άνθρωποι επικοινωνούν μεταξύ τους . Η ανάπτυξή του κατέστησε το διαδίκτυο προσβάσιμο σε εκατομμύρια χρήστες , επιτρέποντας την απρόσκοπτη δημοσιοποίηση και πρόσβαση σε έγγραφα στο διαδίκτυο. Η εκρηκτική ανάπτυξη του Παγκοσμίου Ιστού δημιούργησε προβλήματα «πληροφοριακής υπερφόρτισης». Έτσι, η παγκόσμια κοινότητα στράφηκε σε μια νέα κατεύθυνση εξέλιξης-επέκτασης του ιστού, η οποία ονομάζεται Σημασιολογικός Ιστός (Semantic Web) και περιλαμβάνει την σαφή αναπαράσταση του νοήματος των πληροφοριών και των εγγράφων, επιτρέποντας την αυτόματη επεξεργασία και ενοποίηση διαδικτυακών πόρων από «έξυπνα» προγράμματα-πράκτορες.

Η προφανέστερη δυνατότητα του Σημασιολογικού Ιστού αφορά στην ανάκτηση πληροφοριών τόσο στα πλαίσια του Παγκόσμιου Ιστού, όσο και σε εκείνα μιας Βάσης Δεδομένων - Ψηφιακής Βιβλιοθήκης . Μάλιστα, για να γίνει εφικτή η σύνδεση κάθε έννοιας με έννοιες που έχουν αναπαρασταθεί σε λεξικά, θησαυρούς (Παγκόσμιος Ιστός) ή και βάσεις δεδομένων , δημιουργήθηκαν οι Οντολογίες .

Οι Οντολογίες συνέβαλλαν στον συσχετισμό των πόρων, διαδικτυακών και μη, παρέχοντας έτσι έναν κοινό τόπο κατανόησης των ανταλλασσόμενων πληροφοριών (διαλειτουργικότητα-interoperability). Έτσι, ο Σημασιολογικός Ιστός αποτελεί επέκταση του Παγκόσμιου Ιστού, παρέχοντας πλούσιες σημασιολογικές φόρμες με την χρήση σημασιολογικών τεχνολογιών .

Στα πλαίσια της παρούσας διπλωματικής εργασίας, γίνεται προσπάθεια αξιοποίησης των δυνατοτήτων της Οντολογίας , σχεδιάζοντας και υλοποιώντας ένα σημασιολογικό πληροφοριακό σύστημα το οποίο αξιοποιεί, μεταβάλλει και επεξεργάζεται μια Οντολογία. Επιπλέον, κάνοντας χρήση της UML επιχειρούμαι να εξάγουμε από την δημιουργηθείσα Οντολογία μια σειρά UML σεναρίων . Επιπροσθέτως, σχεδιάζουμε και υλοποιούμε μια παραμετροποιημένη Berkeley XML βάση δεδομένων με σκοπό την σύγκρισή της με την Οντολογία .

Αυτή η εργασία αφιερώνεται

στην οικογένειά μου...

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή κ. Ευριπίδη Πετράκη για την επίβλεψη και την καθοδήγησή του κατά την διάρκεια της εκπονήσεως της παρούσας διπλωματικής εργασίας.

Θα ήθελα επίσης να ευχαριστήσω τα υπόλοιπα μέλη της τριμελούς επιτροπής, τον επίκουρο καθηγητή κ. Αντώνη Δεληγιαννάκη και τον επίκουρο καθηγητή κ. Βασίλη Σαμολαδά.

Ιδιαίτερα θα ήθελα να ευχαριστήσω τον Λουκά και τον Γιάννη που ήταν πάντα πρόθυμοι να προσφέρουν την βοήθειά τους .

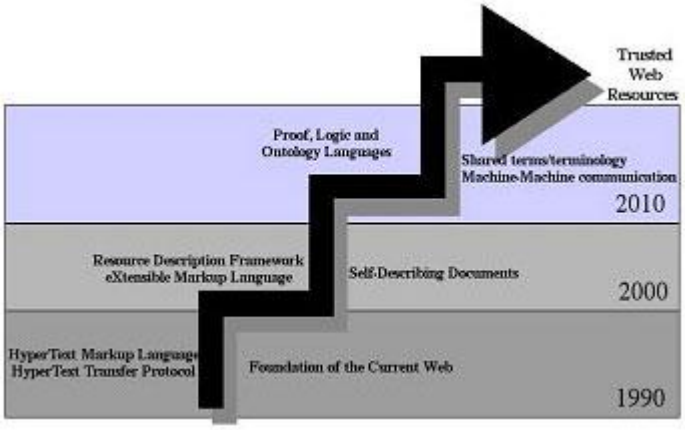
Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου, αλλά και τους φίλους μου που καθ' όλη την διάρκεια της εργασίας αυτής, στέκονταν στο πλευρό μου και με στήριζαν.

Πίνακας Περιεχομένων

1 Εισαγωγή

1.1 Γενικά

Ο Παγκόσμιος Ιστός (World Wide Web) έχει αλλάξει τον τρόπο που οι άνθρωποι επικοινωνούν μεταξύ τους. Η ανάπτυξή του κατέστησε το διαδίκτυο προσβάσιμο σε εκατομμύρια χρήστες, επιτρέποντας την απρόσκοπτη δημοσιοποίηση και πρόσβαση σε έγγραφα στο διαδίκτυο. Η εκρηκτική ανάπτυξη του Παγκοσμίου Ιστού δημιούργησε προβλήματα «πληροφοριακής υπερφόρτισης». Έτσι, η παγκόσμια κοινότητα στράφηκε σε μια νέα κατεύθυνση εξέλιξης-επέκτασης του ιστού, η οποία ονομάζεται Σημασιολογικός Ιστός (Semantic Web) και περιλαμβάνει την σαφή αναπαράσταση του νοήματος των πληροφοριών και των εγγράφων, επιτρέποντας την αυτόματη επεξεργασία και ενοποίηση διαδικτυακών πόρων από «έξυπνα» προγράμματα-πράκτορες.



Εικόνα 1.1 : Η Εξελικτική Πορεία του Παγκόσμιου Ιστού

Η λέξη "Σημασιολογία" έχει ρίζα τις Ελληνικές λέξεις "σημάδι", "σημαίνω" και "σημαντικός" και σήμερα αναφέρεται στο νόημα συχνά σε επίπεδο γλώσσας. Μπορούμε να πούμε ότι ο Σημασιολογικός Ιστός αποτελεί το μεγαλύτερο σε παγκόσμιο επίπεδο έργο ευφυής ενσωμάτωσης συστημάτων ώστε να συνεργάζονται διαλειτουργικά.
Ο Tim Berners-Lee, που επινόησε τον Παγκόσμιο Ιστό το 1989, είχε το όραμα, που τώρα συμμερίζονται πολλοί - ενός ιστού δεδομένων που μπορούν να επεξεργαστούν από μηχανές. Δηλαδή, είχε το όραμα ενός ιστού δεδομένων αυτόματα επεξεργάσιμων από τις εφαρμογές, βάσει του νοήματος και όχι της μορφής της πληροφορίας.

*The **Semantic Web** is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."*
[Berners-Lee, et al., 2001]

Το κέντρο βάρους των περιεχομένων του διαδικτύου μετατοπίζεται συνεχώς από το ελεύθερο κείμενο που είναι πλήρως κατανοητό μόνο από τον άνθρωπο, προς την ημιδομημένη ή και πλήρως δομημένη πληροφορία η οποία μπορεί να γίνει αυτόματα κατανοητή από διαδικτυακές εφαρμογές.
Για να φτάσει το διαδίκτυο στο μέγιστο των δυνατοτήτων του, πρέπει να εξελιχθεί σε μια τέτοια μορφή στην οποία να παρέχει μια παγκοσμίως προσβάσιμη πλατφόρμα που να επιτρέπει σε αυτοματοποιημένα εργαλεία να διαμοιράζονται και να επεξεργάζονται πληροφορίες και δεδομένα για λογαριασμό των ανθρώπων-χρηστών τους.

Σήμερα, η προφανέστερη δυνατότητα του Σημασιολογικού Ιστού αφορά στην ανάκτηση πληροφοριών τόσο στα πλαίσια του Παγκόσμιου Ιστού, όσο και σε εκείνα μιας Βάσης Δεδομένων - Ψηφιακής Βιβλιοθήκης . Μάλιστα, για να γίνει εφικτή η σύνδεση κάθε έννοιας με έννοιες που έχουν αναπαρασταθεί σε λεξικά, θησαυρούς (Παγκόσμιος Ιστός) ή και βάσεις δεδομένων , δημιουργήθηκαν οι Οντολογίες .

Οι Οντολογίες συνέβαλαν στον συσχετισμό των πόρων, διαδικτυακών και μη, παρέχοντας έτσι έναν κοινό τόπο κατανόησης των ανταλλασσόμενων πληροφοριών (διαλειτουργικότητα-interoperability). Οντολογία είναι η αυστηρά μαθηματική περιγραφή ενός πεδίου γνώσης που περιλαμβάνει ένα σύνολο από όρους και τις σημασιολογικές συσχετίσεις μεταξύ τους. Οι όροι της οντολογίας περιγράφουν κλάσεις αντικειμένων, δηλαδή έννοιες – πρότυπα σχετικές με αντικείμενα και οι συσχετίσεις συνήθως αφορούν ιεραρχικές εξαρτήσεις μεταξύ των όρων.

Οι Οντολογίες κωδικοποιούν τη γνώση σε μια περιοχή καθώς και τη γνώση που διαπερνά περιοχές. Με αυτό τον τρόπο, κάνουν αυτή τη γνώση επαναχρησιμοποιήσιμη.

Στην παρούσα φάση η αναζήτηση μέσα σε έγγραφα βασίζεται σε λέξεις-κλειδιά, γεγονός που μειώνει την ποιότητα των αποτελεσμάτων της αναζήτησης και τα έγγραφα που ανακαλούνται μπορεί να χρησιμοποιούν λέξεις με άλλο νόημα από αυτό που αναζητά ο χρήστης. Όμως, όταν οι αναζητήσεις θα γίνονται μέσω οντολογιών, οι λέξεις – κλειδιά θα συνδέονται με το νόημα που θα ήθελε ο χρήστης (με σχέσεις ιεραρχικής συσχέτισης), επιστρέφοντας έτσι μόνο τις επιθυμητές σελίδες. Και οι μηχανές αναζήτησεις θα μπορούν

να επιστρέψουν και έγγραφα που δεν έχουν την συγκεκριμένη λέξη – κλειδί, αλλά κάποια που συνδέεται όμως νοηματικά με την λέξη – κλειδί.

Η έρευνα στην περιοχή των οντολογιών βρίσκεται στα πρώτα στάδιά της, αν και έχουν ήδη εμφανιστεί αρκετές γλώσσες που επιτρέπουν την αναπαράστασή του. Γλώσσες όπως η SHOE, η DAML, η OIL και η υβριδική DAML+OIL[32] αποτελούν επεκτάσεις της RDF δανειζόμενες χαρακτηριστικά από αντίστοιχες γλώσσες αναπαράστασης του πεδίου της τεχνητής νοημοσύνης. Όλες αυτές οι γλώσσες βρίσκονται σε προκαταρκτικό στάδιο, στις οποίες τα τελευταία χρόνια το κυρίαρχο πρότυπο είναι η OWL (Web Ontology Language).
Ένας αριθμός οντολογιών έχει επίσης εμφανιστεί για την διευκόλυνση της ανταλλαγής δεδομένων μεταξύ εμπορικών εφαρμογών τύπου e-Commerce μέσω της αποδοχής κοινής σημασιολογίας για την περιγραφή της πληροφορίας. Αυτές περιλαμβάνουν την CBL (Common Business Library), την cXML (commerce XML), την OCF (Open Catalog Format), την OFX (Open Financial Exchange), το UN/SPSC, το RosetaNet και άλλες. Σημαντικές πρωτοβουλίες έχουν παρθεί για την δημιουργία οντολογιών για επιστημονικά πεδία όπως η Γενετική καθώς και της εφαρμογής τους σε δίκτυα GRID.

Ο σημασιολογικός Ιστός (Semantic Web) αποτελεί μια καινοτομία "εν τη γενέσει" της, η οποία υπόσχεται την ενσωμάτωση όψεων (views) βάσεων δεδομένων, "συναλλαγών" μεταξύ βάσεων δεδομένων (database transactions), λογικών αναπαραστάσεων, συνδέσμων Ιστού (Web links) και αντικειμενοστρεφών αναπαραστάσεων, σε μια σημασιολογική βάση.

Το όραμα της δημιουργίας του Semantic Web στηρίζεται στην επέκταση των υπαρχόντων πλαισίων περιγραφής μετα- δεδομένων και ειδικότερα στην ύπαρξη σημασιολογικού περιεχομένου που είναι δυνατόν να υπόκειται σε αυτόματη επεξεργασία από τον υπολογιστή χωρίς την επέμβαση του ανθρώπινου παράγοντα. Η ερευνητική προσπάθεια έγκειται στην δημιουργία γενικών πλαισίων όπως το UN/SPSC και γλωσσών όπως η OWL τα οποία θα υποστηρίζουν οντολογικά όσο το δυνατόν περισσότερα πεδία εφαρμογών γίνεται. Πρόσφατα, η ομάδα εργασίας οντολογίας του Παγκοσμίου Ιστού εξέδωσε ένα υπό διαμόρφωση προσχέδιο τελικού σχολιασμού των παραδειγμάτων της OWL το οποίο συνοδεύει τον ορισμό της γλώσσας.

1.2 Συνεισφορά Εργασίας

Στα πλαίσια της παρούσας εργασίας αντικείμενο έρευνας είναι η χρήση της οντολογίας στο σχεδιασμό πληροφοριακών συστημάτων σε σχέση με μια παραδοσιακή προσέγγιση που σχετίζεται με σχεσιακό ή XML μοντέλο.

Πιο συγκεκριμένα :

- Με την χρήση της οντολογίας γίνεται κατανοητή η παράσταση από τον άνθρωπο και επιπλέον γίνεται μηχανική, σημασιολογική παράσταση που δείχνει τις έννοιες, τις σχέσεις και τις ιδιότητες των κλάσεων που εμπλέκονται στην σχεδίαση. Αν αντί της οντολογίας υπήρχε XML ή οποιαδήποτε άλλη σχεσιακή βάση, τότε θα ήταν ανάγκη να γραφεί επιπλέον κώδικας για να προκύψουν οι απαραίτητες σχέσεις μεταξύ των κλάσεων. Ακόμη, θα ήταν όλα αυτά ενσωματωμένα στον κώδικα, ενώ με την οντολογία οι σχέσεις, οι ιδιότητες και οι έννοιες των κλάσεων είναι προσβάσιμες στον σχεδιαστή – χρήστη της οντολογίας.
- Το σχήμα της οντολογίας είναι σχεδιαστικά πιο απλό και δίνει by – default την καλύτερη δυνατή παραμετροποίηση, γεγονός που προέκυψε συγκρίνοντας τις δυο διαφορετικές σχεδιαστικές προσεγγίσεις που υλοποιήθηκαν στα πλαίσια της παρούσας εργασίας.
- Με την χρήση της οντολογίας αποφεύχθηκαν τα διάφορα προβλήματα επεκτασιμότητας. Οι διάφορες αλλαγές σε ένα XML αρχείο, επιφέρουν αλλαγές στα XSLT style sheets, με αποτέλεσμα να έχουμε μετατροπή του αρχείου σε άλλα formats. Αυτό με την σειρά του δημιουργεί πρόβλημα στην ενημέρωση όλων των XML εγγράφων και έτσι αυτά δεν είναι συνεπή μετά την νέα προσθήκη.

Αφού λάβαμε υπόψιν όλες τις παραπάνω παραμέτρους και γνωρίζοντας όλα τα πλεονεκτήματα της OWL από την XML(τα οποία αναφέρονται αναλυτικά παρακάτω στην εργασία), προχωρήσαμε στον σχεδιασμό και την υλοποίηση ενός πληροφοριακού συστήματος που αποτελεί μια εφαρμογή για εταιρία ανταλλακτικών αυτοκινήτων.

1.3 Δομή Εργασίας

Στο **δεύτερο κεφάλαιο** θα γίνει ανάλυση των προτύπων, των προδιαγραφών και των τεχνολογιών που χρησιμοποιήθηκαν κατά την διάρκεια της παρούσας διπλωματικής εργασίας.

Στο **τρίτο κεφάλαιο** θα περιγραφεί αναλυτικά ο σχεδιασμός του συστήματος. Στην αρχή θα παρουσιαστεί και θα σχολιαστεί το παραμετροποιημένο σχήμα της Berkeley XML βάσης δεδομένων και μετά θα παρουσιαστεί βήμα προς βήμα η μετάβαση από το XML σχήμα στο γενικό σχήμα της οντολογίας. Στο τέλος του κεφαλαίου αυτού θα γίνει σύγκριση των δυο προαναφερθέντων σχημάτων.

Στο **τέταρτο κεφάλαιο** αναλύεται το σχήμα και τα instances της οντολογίας Car, όπως αυτή υλοποιήθηκε στο Protégé. Στο τέλος του κεφαλαίου αυτού θα γίνει η παρουσίαση τριών σεναρίων που σκοπό έχουν να δείξουν ξεκάθαρα τις part-of και functional σχέσεις της οντολογίας **CAR** .

Στο **πέμπτο κεφάλαιο** θα παρουσιαστεί η υλοποίηση της εφαρμογής. Δηλαδή θα γίνει αναφορά στην πλατφόρμα υλοποίησης της εφαρμογής και την λειτουργικότητάς της. Στο τέλος του κεφαλαίου αυτού θα γίνει αναφορά στην διεπικοινωνία του χρήστη με το σύστημα (user interface), παρουσιάζοντας και σχολιάζοντας μερικά βασικά screen-dumps της εφαρμογής.

2 Σχετικά Πρότυπα, Προδιαγραφές, Τεχνολογίες

2.1 Εισαγωγή

Σε αυτό το κεφάλαιο στόχος είναι να γίνει μια εισαγωγή σε όλα τα πρότυπα και τις τεχνολογίες που χρησιμοποιηθήκαν κατά την υλοποίησης της παρούσας εργασίας. Ειδικότερα:

- Στην Ενότητα 2.2 θα παρουσιαστεί η γλώσσα XML.
- Στην ενότητα 2.3 θα παρουσιαστεί η XML βάση δεδομένων Oracle Berkeley DB XML.
- Στην Ενότητα 2.4 θα παρουσιαστεί η γλώσσα περιγραφής οντολογιών OWL.
- Στην Ενότητα 2.5 θα παρουσιαστεί το Jena Framework.
- Στην Ενότητα 2.6 θα παρουσιαστεί ένα εργαλείο για την Java, το Swing.
- Στην Ενότητα 2.7 θα παρουσιαστεί μια οπτική αντικειμενοστρεφής γλώσσα μοντελοποίησης, η UML.

2.2 XML

Η eXtensible Markup Language (XML) είναι μια γλώσσα ανεξάρτητη από σύστημα και υλικό για την αναπαράσταση δεδομένων και της μορφής τους σε ένα έγγραφο XML (XML document). Ένα έγγραφο XML στην πιο απλή του μορφή είναι ένα αρχείο κειμένου το οποίο περιέχει δεδομένα μαζί με σήμανση η οποία καθορίζει τη δομή των δεδομένων.
Η XML είναι μια παγκοσμίως συμφωνημένη μεταγλώσσα σήμανσης που χρησιμοποιείται πρώτιστα για την ανταλλαγή πληροφοριών. Η ομορφιά της XML βρίσκεται στο γεγονός ότι είναι επεκτάσιμη. Απλά, η XML είναι ένα σύνολο προκαθορισμένων κανόνων (συντακτικό πλαίσιο) που πρέπει να ακολουθήσουμε κατά τη δόμηση των δεδομένων μας.

Για ένα μεγάλο χρονικό διάστημα, οι προγραμματιστές και οι προμηθετές εφαρμογών κατασκεύαζαν εφαρμογές και συστήματα εγκατεστημένα σε μια επιχείρηση τα οποία επεξεργάζονταν δεδομένα τα οποία μπορούσαν να με το δικό τους ιδιωτικό τρόπο. Αλλά καθώς η ανταλλαγή πληροφορίας μεταξύ εφαρμογών και συστημάτων στις επιχειρήσεις επικρατούσε, έγινε πολύ δύσκολο να αναταλλάσεις δεδομένα διότι τα συστήματα δε σχεδιάστηκαν ώστε να δέχονται δεδομένα από εξωτερικά, άγνωστα συστήματα .

Η XML παρέχει μία πρότυπη και κοινή δομή για τη διανομή δεδομένων μεταξύ ανόμοιων συστημάτων. Επιπλέον, η XML έχει ενσωματωμένο ένα μηχανισμό επικύρωσης δεδομένων, ο οποίος εγγυάται ότι η δομή των δεδομένων που λαμβάνεται είναι έγκυρη.

2.2.1 Παράδειγμα αναπαράστασης δεδομένων με XML

Ας δούμε πώς αναπαριστούμε δεδομένα με τη βοήθεια της XML :

```
<employee>
  <shift id= "counter" time="8-12">
    <phone id = "1"> All phone information
      <number>3444333</number >
    </phone>
  </shift >
  <shift id="help_desk" time="1-5">
    <phone id = "2"> All phone information
        <number>332333</number >
    </phone>
  </shift >
  ...
  <home-address>
      <street>3434 Norwalk street</street>
      <city>New York</city>
      <state>NY</state>
  </home-address>
</employee>
```

Εικόνα 2.1 : Παράδειγμα XML εγγράφου

Στο παραπάνω παράδειγμα, αναπαριστούμε τις προσωπικές πληροφορίες και τις πληροφορίες σχετικά με τις βάρδιες ενός υπαλλήλου σε ένα οργανισμό.

Βλέπουμε πώς η XML χρησιμοποιεί τις διακριτικές ετικέτες "<>" και "</>" παρόμοια με τις ετικέτες που χρησιμοποιούνται στην HTML. Αυτό συμβαίνει γιατί η XML είναι μια γλώσσα σήμανσης σαν την HTML.

Η κύρια διαφορά της XML με την HTML είναι ως προς τον σκοπό της κάθε μίας:

- Η XML σχεδιάστηκε για να περιγράφει δεδομένα και να εστιάσει στο τί είναι αυτά τα δεδομένα.
- Η HTML σχεδιάστηκε για να προβάλει δεδομένα και να εστιάσει στο πώς φαίνονται αυτά τα δεδομένα.

Στο παραπάνω παράδειγμα βλέπουμε ότι έχουμε να κάνουμε με καθαρά δεδομένα : ότι ένας υπάλληλος (employee) έχει παραπάνω από μία βάρδιες (shift), για παράδειγμα το πρωί εργάζεται στο ταμείο (counter) και το μεσημέρι στο γραφείο βοήθειας (help desk) και διευθύνσεις.

Οι δύο αρχικές δομικές μονάδες XML που χρησιμοποιούνται στο προηγούμενο παράδειγμα είναι τα elements (στοιχεία) και τα attributes (ιδιότητες).

2.2.2 Elements ή Στοιχεία

Τα στοιχεία (elements) είναι ετικέτες, όπως και στην HTML, και περιέχουν τιμές. Επιπλέον τα elements είναι δομημένα σαν δένδρο. Ώς εκ τούτου έχουμε τα στοιχεία οργανωμένα σε ένα ιεραρχικό τρόπο με ένα στοιχείο-πατέρα και στοιχεία-παιδιά. Τα στοιχεία-παιδιά μπορούν να περιέχουν και αυτά άλλα στοιχεία-παιδιά και ούτω καθ'εξής.

Στο προηγούμενο παράδειγμα, το στοιχείο <employee> είναι το γονικό στοιχείο και έχει το στοιχείο <shift> σαν στοιχείο-παιδί. Παρακάτω το στοιχείο <phone> είναι στοιχείο-παιδί του γονικού στοιχείου <shift>.

Τα στοιχεία έχουν συγκεκριμένα χαρακτηριστικά. Ορισμένα από αυτά είναι :

- Τα στοιχεία μπορεί να περιέχουν δεδομένα, οπως το στοιχείο <number> στο παράδειγμα.
- Αντίστροφα, τα στοιχεία μπορεί να μην περιέχουν δεδομένα αλλά μόνο ιδιότητες, όπως το στοιχείο <shift>.
- Εναλλακτικά, τα στοιχεία μπορεί να περιέχουν ταυτόχρονα και ιδιότητες αλλά και δεδομένα, αλλά επίσης και στοιχεία-παιδιά, όπως το στοιχείο <phone>.

Τα στοιχεία έχουν κάποιους κανόνες :

- Όλα τα στοιχεία πρέπει να έχουν ετικέτα κλεισίματος αντίθετα με την HTML όπου υπάρχουν και ετικέτες που δε χρειάζονται κλείσιμο όπως για παράδειγμα η
.
- Οι ετικέτες των στοιχείων είναι case sensitive δηλαδή υπάρχει διαχωρισμός μεταξύ κεφαλαίων και πεζών και τα ονόματά τους υπακούουν σε κανόνες ονοματολογίας.

- Τα στοιχεία πρέπει να είναι τοποθετημένα σωστά αντίθετα με την HTML.

  HTML : This text is bold and italic

  XML : This text is bold and italic

- Τα έγγραφα της XML πρέπει να έχουν ακριβώς ένα αρχικό στοιχείο (root element).

2.2.3 Attributes ή Ιδιότητες

Οι ιδιότητες (attributes) μας βοηθούν να δώσουμε περισσότερο νόημα και να περιγράψουμε τα στοιχεία μας πιο αποτελεσματικά και με σαφήνεια. Στο προηγούμενο παράδειγμα, το στοιχείο <shift> έχει μία ιδιότητα "id" με τιμές "counter" και "help_desk" . Με τη χρήση τέτοιων ιδιοτήτων, μπορούμε να ξέρουμε άν ένας υπάλληλος εργα΄ζεται στο ταμείο ή στο γραφείο βοήθειας. Αυτό βοηθάει στο να κάνουμε τα δεδομένα σε ένα έγγραφο XML αυτοπεριγραφικά. Πρέπει πάντα να θυμόμαστε ότι ο κύριος σκοπός των ιδιοτήτων είναι να παρέχουν περισσότερη πληροφορία σχετική με ένα στοιχείο και δεν πρέπει να χρησιμοποιούνται για να περιέχουν τα ίδια τα δεδομένα .

Όπως και τα στοιχεία έτσι και οι ιδιότητες έχουν κάποιους κανόνες :

- Οι τιμές των ιδιοτήτων πρέπει να είναι εσωκλείωνται σε " " ή σε ' '.
- Τα ονόματα των ιδιοτήτων ακολουθούν τους ίδιους κανόνες με αυτά των ετικετών.

2.2.4 Καλά διαμορφωμένα έγγραφα (well formed documents)

Ένα «καλά διαμορφωμένο» έγγραφο XML είναι ένα έγγραφο που υπακούει στους κανόνες σύνταξης της XML που αναφέραμε προηγουμένως :

- Τα έγγραφα XML πρέπει να περιέχουν ένα αρχικό στοιχείο.
- Τα στοιχεία XML πρέπει να έχουν ετικέτες κλεισίματος.
- Στις ετικέτες XML υπάρχει διαχωρισμός κεφαλαίων και πεζών.
- Τα στοιχεία XML πρέπει να είναι σωστα τοποθετημένα.
- Οι ιδιότητες XML πρέπει να εσωκλείονται πάντα σε "" ή ''.

2.2.5 DTD

Όπως σε μία γλώσσα προγραμματισμού πρέπει να ξέρουμε τις προδιαγραφές της γλώσσας, με παρόμοιο τρόπο το Document Type Definition (DTD) είναι μία προδιαγραφή, η οποία πρέπει να ακολουθηθεί όταν δημιουργούμε ένα έγγραφο XML. Επίσης, όπως μία από τις εργασίες του μεταγλωττιστή για κάθε γλώσσα προγραμματισμού είναι να ελέγξει αν η προδιαγραφές ακολουθήθηκαν, με παρόμοιο τρόπο υπάρχουν XML parsers οι οποίοι χρησιμοποιούν το DTD για να ελέγξουν την εγκυρότητα ενός εγγράφου XML.

Ένα DTD μας βοηθάει να καθορίσουμε τη δομή ενός εγγράφου XML. Μας παρέχει ένα αυστηρό πλαίσιο και κανόνες οι οποίοι θα ακολουθηθούν όταν δημιουργούμε έγγραφα XML. Επιπρόσθετα, το DTD μπορεί να χρησιμοποιηθεί για τον έλεγχο της εγκυρότητας και της ακεραιότητας των δεδομένων που περιέχονται σε ένα έγγραφο XML.

Μερικά χαρακτηριστικά του DTD είναι τα παρακάτω :

- Το DTD χρησιμοποιείται για να καθορίσει έγκυρα στοιχεία και ιδιότητες που μπορούν να χρησιμοποιηθούν σε ένα έγγραφο XML.
- Με ένα DTD μπορούμε να καθορίσουμε μια ιεραρχική δομή στοιχείων.
- Σε ένα DTD μπορεί επίσης να καθοριστεί η διαδοχική οργάνωση μιας συλλογής στοιχείων-παιδιών τα οποία μπορούν να υπάρχουν σε ένα έγγραφο XML.

Ένα DTD μπορεί να χρησιμοποιηθεί απευθείας μέσα σε ένα έγγραφο XML ή μπορεί να υπάρχει εκτός του εγγράφου XML. Στη δεύτερη περίπτωση θα αναφέρεται με ένα δεσμό μέσα στο έγγραφο XML που δείχνει σε αυτό το DTD.

Βασικά το DTD αποτελείται από τα παρακάτω στοιχεία :

Στοιχείο	Περιγραφή
DTD Element	Μεταδεδομένα για ένα στοιχείο. Καθορίζει τι είδους δεδομένα θα έχει το στοιχείο, τον αριθμό των περιστατικών κάθε στοιχείου, τις σχέσεις μεταξύ των στοιχείων και ούτω καθ'εξής.
DTD Attributes	Καθορίζει διάφορους κανόνες και ορισμούς που σχετίζονται με τα δεδομένα.
DTD Entities	Χρησιμοποιείται για να αναφέρει ένα εξωτερικό αρχείο ή για να παρέχει συντομεύσεις σε κοινό κείμενο.

Πίνακας 2.1 : Τα στοιχεία του DTD

Παράδειγμα : <!ELEMENT employee(shift+,home-address, hobbies*)>

Ένα στοιχείο employee μπορεί να περιέχει ένα ή περισσότερα στοιχεία shift και πρέπει να έχει ένα στοιχείο home-address και μπορεί να έχει μηδέν ή περισσότερα στοιχεία hobbies.

Παράδειγμα : <!ATTLIST shift id CDATA #REQUIRED>

Το στοιχείο shift πρέπει να έχει μία ιδιότητα id.

Εν ολίγοις, ένα DTD χρησιμοποιείται για να καθορίσει μια δομή εγγράφων με τη διευκρίνιση των λεπτομερειών σχετικά με όλα τα στοιχεία και τις ιδιότητες που πρόκειται να χρησιμοποιηθούν σε ένα έγγραφο XML. Ως εκ τούτου μπορεί να χρησιμοποιηθεί για να ελέγξει την εγκυρότητα ενός εγγράφου XML που υποτίθεται ότι ακολουθεί τους κανόνες που καθορίζονται από αυτό το DTD.

2.2.6 XML Schema

Το XML Schema είναι μια πιο προηγμένη έκδοση του DTD. Το DTD έχει πολλά μειονεκτήματα σε σχέση με το schema, όπως το ότι δεν υποστηρίζει ισχυρούς τύπους δεδομένων, έχει σύνταξη διαφορετική από την XML και δεν είναι επεκτάσιμο. Το XML Schema παρουσιάστηκε για να υπερνικήσει αυτά τα μειονεκτήματα.

Οι δύο κύριοι στόχοι του W3c XML Schema working group κατά τη διάρκεια του σχεδιασμού του προτύπου του XML Schema ήταν :

- Να μπορέσουν να εκφράσουν μέσα στο πρότυπο αρχές αντικειμενοστραφούς σχεδιασμού οι οποίες μπορούν να βρεθούν σε όλες τις αντικειμενοστραφείς γλώσσες προγραμματισμού.
- Να παρέχουν υποστήριξη για σύνθετους τύπους δεδομένων παρόμοια με την υποστήριξη που υπάρχει στις περισσότερες σχεσιακές βάσεις δεδομένων.

Τα κυριότερα χαρακτηριστικά του XML Schema είναι τα παρακάτω :

- Η σύνταξη είναι όμοια με της XML. Αυτό σημαίνει ότι μπορούμε να επεξεργαστούμε το schema με οποιοδήποτε επεξεργαστή XML.
- Δεν καθορίζουμε μόνο βασικούς τύπους δεδομένων όπως αλφαρηθμητικό, ακέραιος, πραγματικός και ούτω καθ'εξής αλλά μπορούμε επίσης να καθορίσουμε δικούς μας τύπους δεδομένων.

Για παράδειγμα:

```
<xs:element name="name" type="xs:string" />
```

Εικόνα 2.2 : Παράδειγμα με τύπο δεδομένων

Οι νέοι τύποι που μπορούμε να καθορίσουμε μπορεί να είναι απλοί ή σύνθετοι. Οι σύνθετοι τύποι μπορεί να περιέχουν και άλλα στοιχεία ή και ιδιότητες, ενώ οι απλοί τύποι όχι. Αντίθετα μπορούν να περιέχουν μόνο δεδομένα.

- Το XML Schema παρέχει επικύρωση βασισμένη στο περιεχόμενο (content-based validation) δηλαδή μπορεί να ορίσει την σειρά με την οποία τα στοιχεία-παιδιά εμφανίζονται. Επίσης παρέχει επικύρωση στους ίδιους τους τύπους δεδομένων. Για παράδειγμα μπορούμε να ορίσουμε έναν απλό τύπο "year" με τιμές μεταξύ 2000 και 2100:

```
<xsd:simpleType name="year">
  <xsd:restriction base="xsd:integer">
    <xsd:minInclusive value="2000"/>
    <xsd:maxInclusive value="2100"/>
</xsd:restriction>
</ xsd:simpleType>
```

Εικόνα 2.3 : Παράδειγμα με τύπο δεδομένων

Παρομοίως οι σύνθετοι τύποι μπορεί να ορίσουν τη σειρά με την οποία τα στοιχεία-παιδιά θα εμφανίζονται:

```
<xsd:complexType name="Employee">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:string" />
    <xsd:element name="Address" type="xsd:string" />
    <xsd:element name="Phone" type="xsd:string" />
  <xsd:sequence>
</xsd:complexType>
```

Εικόνα 2.4 : Παράδειγμα σύνθετου τύπου

- Το XML Schema μας παρέχει τη δυνατότητα να επεκτείνουμε άλλα έγγραφα το οποίο δεν είναι τίποτα άλλο παρά κληρονομικότητα. Αυτό σημαίνει ότι μπορούμε να παράγουμε νέους τύπους δεδομένων βάσει παλαιών τύπων.

- Το XML Schema παρέχει υποστήριξη για Namespaces (χρησιμοποιώντας URI). Παρέχει σε κάθε στοιχείο ένα μοναδικό αναγνωριστικό, με το οποίο αποφεύγονται συγκρούσεις ονομάτων μεταξύ των στοιχείων. Αυτό θα μπορούσε να συμβεί, για παράδειγμα, όταν δύο έγγραφα συγχωνεύονταν, και περιείχαν και τα δύο στοιχεία με όνομα "name" τα οποία όμως είχαν διαφορετικό νόημα σε κάθε έγγραφο. Εν ολίγοις βοηθάει στο να ξεχωρίζουν στοιχεία και ιδιότητες με ίδιο όνομα και διαφορετικό νόημα. Μία απλή αναλογία στις γλώσσες προγραμματισμού είναι η χρήση καθολικών και τοπικών μεταβλητών. Μια τοπική μεταβλητή είναι μοναδική μέσα στο πεδίο ισχύος της ενώ μια καθολική μεταβλητή πρέπει να είναι μοναδική σε ολόκληρο το πρόγραμμα. Παρομοίως, με το namespace έχουμε την ελευθερία να να ορίσουμε τύπους χωρίς να ανησυχούμε για συγκρούσεις ονομάτων.
- Τέλος το XML Schema είναι εύκολα επεκτάσιμο για να ενσωματώσει και άλλες λειτουργίες στο μέλλον.

2.3 Berkeley XML DB

Για την αποθήκευση των XML εγγράφων χρησιμοποιήθηκε μία Native Xml Database και συγκεκριμένα η Berkeley DB XML η οποία είναι ένα εργαλείο ανοιχτού λογισμικού (open source). Η Berkeley DB XML αποτελεί μια υψηλών δυνατοτήτων XML βάση δεδομένων που παρέχει υποστήριξη για XQuery γλώσσα αναζήτησης. Χαρακτηρίζεται ως embeddable, δηλαδή έχει δυνατότητες ενσωμάτωσης στην ίδια την εφαρμογή. Ως αποτέλεσμα, τρέχει απευθείας με την εφαρμογή που την χρησιμοποιεί, χωρίς να απαιτείται ανεξάρτητος database server και ανθρώπινη διαχείριση. Διαχειρίζεται τα XML έγγραφα χρησιμοποιώντας XQuery και προσφέρει προχωρημένες υπηρεσίες διαχείρισης δεδομένων, που περιλαμβάνουν ταυτόχρονη πρόσβαση, συνδιαλλαγές και δημιουργία αντιγράφων (replication) για υψηλή διαθεσιμότητα και αντοχή στα λάθη.

Η βασική δομή αποθήκευσης της Berkeley DB XML είναι τα XML έγγραφα. Η συγκεκριμένη βάση δεδομένων οργανώνει τα XML έγγραφα σε συλλογές (containers) και κάθε συλλογή μπορεί να περιέχει έγγραφα τα οποία βασίζονται σε κάποιο συγκεκριμένο σχήμα (σε έναν container, τα αρχεία που αποθηκεύονται, δεν χρειάζεται να υπακούουν όλα στο ίδιο σχήμα).
Σε αντίθεση με τις σχεσιακές βάσεις δεδομένων, οι οποίες αποθηκεύουν δεδομένα σε σχεσιακούς πίνακες, η Berkeley DB Xml έχει ως στόχο να αποθηκεύσει αυθαίρετα δέντρα XML δεδομένων. Στην συνέχεια αυτά μπορούν να αντιστοιχηθούν και να ανακτηθούν, είτε ως πλήρη έγγραφα είτε ως μεμονωμένα τμήματα εγγράφου μέσω μιας XML γλώσσας αναζήτησης, όπως η XQuery.

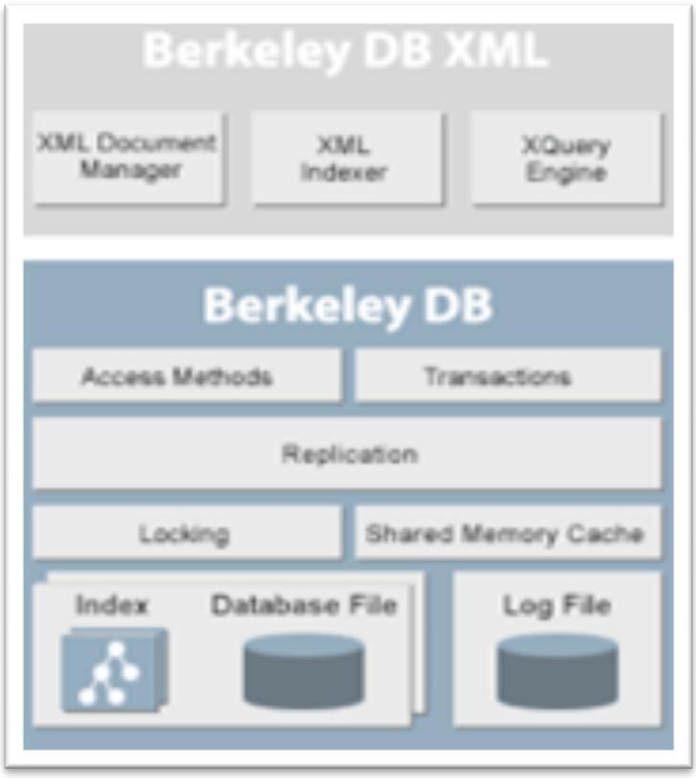
2.3.1 Πλεονεκτήματα της XML Βάσης Δεδομένων

Μια XML βάση δεδομένων έχει διάφορα πλεονεκτήματα σε σχέση με τις σχεσιακές (relational) και αντικειμενοστραφής (object-oriented) βάσεις δεδομένων .

- Xml δεδομένα αποθηκεύονται κατευθείαν στην βάση δεδομένων χωρίς να χρειάζεται περαιτέρω επεξεργασία ή απόσπαση των δεδομένων από κάποιο έγγραφο.
- Τα περισσότερα στοιχεία ενός εγγράφου ,όπως κενά διαστήματα, παραμένουν ανέπαφα κατά την εισαγωγή του στην XML βάση δεδομένων .
- Ερωτήματα (Queries) επιστρέφουν τα έγγραφα XML ή τμήματα τους, το οποίο σημαίνει ότι η ιεραρχική δομή των πληροφοριών XML διατηρείται.

2.3.2 Η Αρχιτεκτονική της Oracle Berkeley DB XML

Η Oracle Berkeley DB XML, είναι χτισμένη στη κορυφή της Oracle Berkeley DB και ως αποτέλεσμα κληρονομεί σημαντικά χαρακτηριστικά και στοιχεία. Ειδικότερα, η ίδια προσθέτει στη κορυφή της Oracle Berkeley DB έναν document parser, έναν XML indexer και μια μηχανή XQuery με στόχο την επίτευξη ταχύτερης και αποδοτικότερης ανάκτησης δεδομένων. Στην εικόνα παρουσιάζεται η αρχιτεκτονική της Oracle Berkeley DB, της Oracle Berkeley DB XML και της Oracle Berkeley DB Java Edition :



Εικόνα 2.5 : Η Αρχιτεκτονική της Berkeley DB και της Berkeley DB XML

Εικόνα 2.6 : Η Αρχιτεκτονική της Oracle Berkeley DB Java Edition

Χάρη στην Oracle Berkeley DB ως την υποκείμενη μηχανή αποθήκευσης, η Oracle Berkeley DB XML κληρονομεί πλήρεις ACID (Atomicity, Consistency, Isolation, Durability) συνδιαλλαγές, αυτόματη αποκατάσταση, κρυπτογράφηση δεδομένων στο δίσκο με AES (Advanced Encryption Standard)και δημιουργία αντιγράφων (replication) για υψηλή διαθεσιμότητα. Επιπρόσθετα, τόσο XML όσο και μη-XML δεδομένα μπορούν να αποθηκευτούν στην Oracle Berkeley DB XML, κάτι που αποτελεί πλεονέκτημα σε κάποιες εφαρμογές.
Η Oracle Berkeley DB XML υποστηρίζει την XQuery 1.0 (Η Xquery αποτελεί πλέον τη κατεξοχήν γλώσσα ερωτήσεων για πρόσβαση σε XML δεδομένα) και την XPath 2.0, XML ονοματοδοσία, έλεγχο εγκυρότητας XML εγγράφων. Πιο συγκεκριμένα, η XQuery μηχανή χρησιμοποιεί ένα εξελιγμένο, βασισμένο στο κόστος, βελτιστοποιητή ερώτησης και υποστηρίζει προ-μεταγλωττισμένη εκτέλεση ερώτησης με ενσωματωμένες μεταβλητές. Μεγάλα έγγραφα μπορούν να αποθηκευτούν ολόκληρα ή κατατμημένα σε κόμβους, επιτυγχάνοντας έτσι αποδοτικότερη ανάκτηση και μερική ενημέρωση των εγγράφων. Επίσης, υποστηρίζει ευέλικτη δεικτοδότηση XML κόμβων, στοιχείων, χαρακτηριστικών, μεταδεδομένων για πιο γρήγορη και αποδοτική ανάκτηση.

2.4 OWL

Η OWL (Web Ontology Language) είναι μια πρότυπη γλώσσα περιγραφής οντολογιών στο Διαδίκτυο. Έχει αναπτυχθεί πάνω από το RDF και χρησιμοποιεί σύνταξη XML . Επεξεργάξεται δεδομένα στον Ιστό και σχεδιάστηκε ώστε να είναι κατανοητή από τους υπολογιστές και όχι από τους ανθρώπους. Η OWL στοχεύει στην παροχή πλήρους υποστήριξης των χαρακτηριστικών των γλωσσών αναπαράστασης γνώσης, επεκτείνοντας με αυτό τον τρόπο την εκφραστικότητα (expressiveness) της DAML+OIL.

2.4.1 Τα Είδη της OWL

Το Web Ontology Working Group του W3C έχει ορίσει την OWL ως τρεις διαφορετικές υπο-γλώσσες, κάθε μια εκ των οποίων στοχεύει να ικανοποιήσει διαφορετικές
πλευρές του πλήρους συνόλου των απαιτήσεων. Τα τρία είδη της OWL είναι τα εξής :

- **OWL Full**
  Η πλήρης γλώσσα καλείται OWL Full. Χρησιμοποιεί όλες τις θεμελιώδεις αρχές των υπογλωσσών OWL και επιτρέπει το συνδιασμό των αρχών αυτών με την RDF και το RDF Schema. Έχει το πλεονέκτημα ότι είναι προς τα πάνω συμβατή με την RDF, τόσο συντακτικά όσο και σημασιολογικά. Δηλαδή, κάθε νόμιμο RDF κείμενο είναι και νόμιμο OWL Full κείμενο και κάθε έγκυρο RDF/RDF Schema συμπέρασμα είναι και έγκυρο OWL συμπέρασμα.
  Το μειονέκτημα της OWL Full είναι ότι η γλώσσα είναι τόσο ισχυρή σε σημείο που παρεμποδίζεται η λήψη αποφάσεων, εξαφανίζοντας κάθε ελπίδα ολοκληρωμένης (ή αποδοτικής) υποστήριξης της συλλογιστικής(reasoning).

- **OWL DL**
  Προκειμένου να βελτιωθεί η υπολογιστική αποδοτικότητα ορίστηκε η OWL DL , μια υπογλώσσα της OWL Full, η οποία περιορίζει τον τρόπο χρήσης των κατασκευαστών (constructors) της OWL και της RDF.
  Ουσιαστικά δεν επιτρέπεται η εφαρμογή ενός OWL κατασκευαστή σε έναν άλλο και επομένως διασφαλίζεται ότι η γλώσσα ανταποκρίνεται σε description logic(Η description logic είναι υποσύνολο της predicate logic και επιτρέπει αποδοτική συλλογιστική).
  Το πλεονέκτημα της OWL DL είναι ότι επιτρέπεται η αποδοτική υποστήριξη της συλλογιστικής. Το μειονέκτημα είναι ότι χάνεται η πλήρης συμβατότητα με την RDF, γιατί ένα RDF κείμενο θα πρέπει να επεκταθεί σε ορισμένα σημεία και να περιορισθεί σε κάποια άλλα, ώστε να είναι νόμιμο OWL DL κείμενο. Γενικώς, κάθε νόμιμο OWL DL κείμενο είναι και νόμιμο RDF κείμενο.

- **OWL Lite**
  Με επιπλέον περιορισμούς η OWL DL περιορίζεται σε ένα υποσύνολο των γλωσσικών κατασκευαστών. Η OWL Lite αποκλείει τις απαριθμημένες κλάσεις (enumerated classes) ,τις δηλώσεις συμβατότητας(disjointness statements) και το αυθαίρετο μέγεθος πεδίου τιμών ιδιοτήτων (cardinality) .

Το πλεονέκτημα είναι ότι η προκύπτουσα γλώσσα είναι εύκολη στην εκμάθηση για τους χρήστες και εύκολα υλοποιήσιμη για τους κατασκευαστές εργαλείων. Το μειονέκτημα της είναι ασφαλώς η περιορισμένη εκφραστικότητα.
Υπάρχουν συγκεκριμένες έννοιες για την ανωφερή συμβατότητα μεταξύ των τριών υπογλωσσών :

- Κάθε νόμιμη OWL Lite οντολογία είναι νόμιμη OWL DL οντολογία.
- Κάθε νόμιμη OWL DL οντολογία είναι νόμιμη OWL Full οντολογία.
- Κάθε έγκυρο OWL Lite συμπέρασμα είναι έγκυρο OWL DL συμπέρασμα.
- Κάθε έγκυρο OWL DL συμπέρασμα είναι έγκυρο OWL Full συμπέρασμα.
- Η OWL εξακολουθεί να χρησιμοποιεί την RDF και το RDF Schema σε μεγάλο βαθμό:
  - Όλα τα είδη της OWL χρησιμοποιούν την RDF για τη σύνταξή τους.
  - Τα στιγμιότυπα ορίζονται όπως στην RDF, χρησιμοποιώντας τις RDF περιγραφές και εισάγοντας την πληροφορία.
  - Constructors της OWL όπως οι owl:Class, owl:DatatypeProperty, και owl:ObjectProperty αποτελούν εξειδικεύσεις των αντίστοιχων της RDF.

2.4.2 Τα Δομικά Στοιχεία της OWL

Τα δομικά στοιχεία της OWL είναι οι **κλάσεις,** οι **ιδιότητες** και τα **άτομα**. Οι OWL κλάσεις ,οι ιδιότητες και τα άτομα ταυτοποιούνται από μοναδικές ταυτότητες που καθορίζονται στο «rdf:ID» γνώρισμά τους.

Οι **OWL κλάσεις** αναπαριστούν σύνολα ατόμων που διαθέτουν κάποιες κοινές ιδιότητες και ανήκουν στην ίδια κατηγορία. Κάθε OWL άτομο είναι μέλος της κλάσης “owl:Thing” και συνεπώς κάθε νέα OWL κλάση που ορίζεται είναι υποκλάση της κλάσης “owl:Thing”. Οι OWL κλάσεις ορίζονται μέσω της δομής “owl:Class” και μπορεί να οριστούν εξ’ αρχής, μέσω των λειτουργιών συνόλων τομή (intersection), ένωση και συμπλήρωμα (με τη χρήση των δομών “owl:intersectionOf”, “owl:unionOf” και “owl:complementOf” αντίστοιχα) και με απαρίθμηση (enumeration) των ατόμων που ανήκουν σε αυτές(μέσω της δομής “owl:oneOf”). Ιεραρχίες κλάσεων σχηματίζονται με τη χρήση της δομής “rdfs:subClassOf”, που δηλώνει ότι μια κλάση “A1” αποτελεί υποκλάση της κλάσης “A”, την οποία εξειδικεύει. Μια OWL κλάση μπορεί να είναι υποκλάση περισσότερων από μιας κλάσεων. Επιπλέον, δυο OWL κλάσεις μπορούν να χαρακτηριστούν ως ισοδύναμες (equivalent) ή ξένες (disjoint), μέσω των δομών αντιστοίχησης (mapping) “owl:equivalentClass” και “owl:disjointWith” αντίστοιχα. Παράδειγμα ορισμού OWL κλάσεων αποτελούν οι κλάσεις “Car” και “Timoni”, ορίζονται σε OWL σύνταξη στην παρακάτω εικόνα.

```
<owl:Class rdf:ID="Car">
< owl:Class rdf:ID="Timoni"/>
        <owl:subClassOf rdf:resource="#Car"/>
</owl:Class>
```

Εικόνα 2.7 : Παράδειγμα ορισμού OWL κλάσεων

Οι **OWL ιδιότητες** (OWL properties) επιτρέπουν τον ισχυρισμό γενικευμένων κοινά αποδεκτών δεδομένων(facts) για τις κλάσεις και συγκεκριμένων κοινά αποδεκτών δεδομένων για τα άτομα των κλάσεων.
Οι ιδιότητες είναι δυαδικές σχέσεις και διακρίνονται σε 2 κατηγορίες:

- **Τις Ιδιότητες Τύπων Δεδομένων** (Datatype Properties), που συσχετίζουν άτομα που ανήκουν στην OWL κλάση που αποτελεί πεδίο ορισμού της ιδιότητας, με τιμές ενός συγκεκριμένου τύπου δεδομένων, που αποτελεί πεδίο τιμών της ιδιότητας. Το πεδίο τιμών μπορεί να είναι κάποιος XML Schema τύπος ή συγκεκριμένες κυριολεκτικές τιμές.

```
<owl:DatatypeProperty rdf:ID="mixaniko">
      <rdfs:domain rdf:resource="#KubotioTaxititon"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XLMSchema
       #string"/>
</owl:DatatypeProperty>
```

Εικόνα 2.8 : Παράδειγμα δημιουργίας Datatype Property

- **Τις Ιδιότητες Αντικειμένων** (Object Properties), που συσχετίζουν άτομα που ανήκουν στην OWL κλάση που αποτελεί πεδίο ορισμού της ιδιότητας, με άτομα που ανήκουν στην OWL κλάση που αποτελεί πεδίο τιμών της ιδιότητας. Οι Ιδιότητες Αντικειμένων ορίζονται μέσω της δομής “owl:ObjectProperty”.

```
<owl:ObjectProperty rdf:ID="hasDiskofrena">
        <owl:domain rdf:resource="#Car"/>
        <owl:range rdf:resource= "#DiskofrenaMembership"/>
</owl:ObjectProperty>
```

Εικόνα 2.9 : Παράδειγμα δημιουργίας Object Property

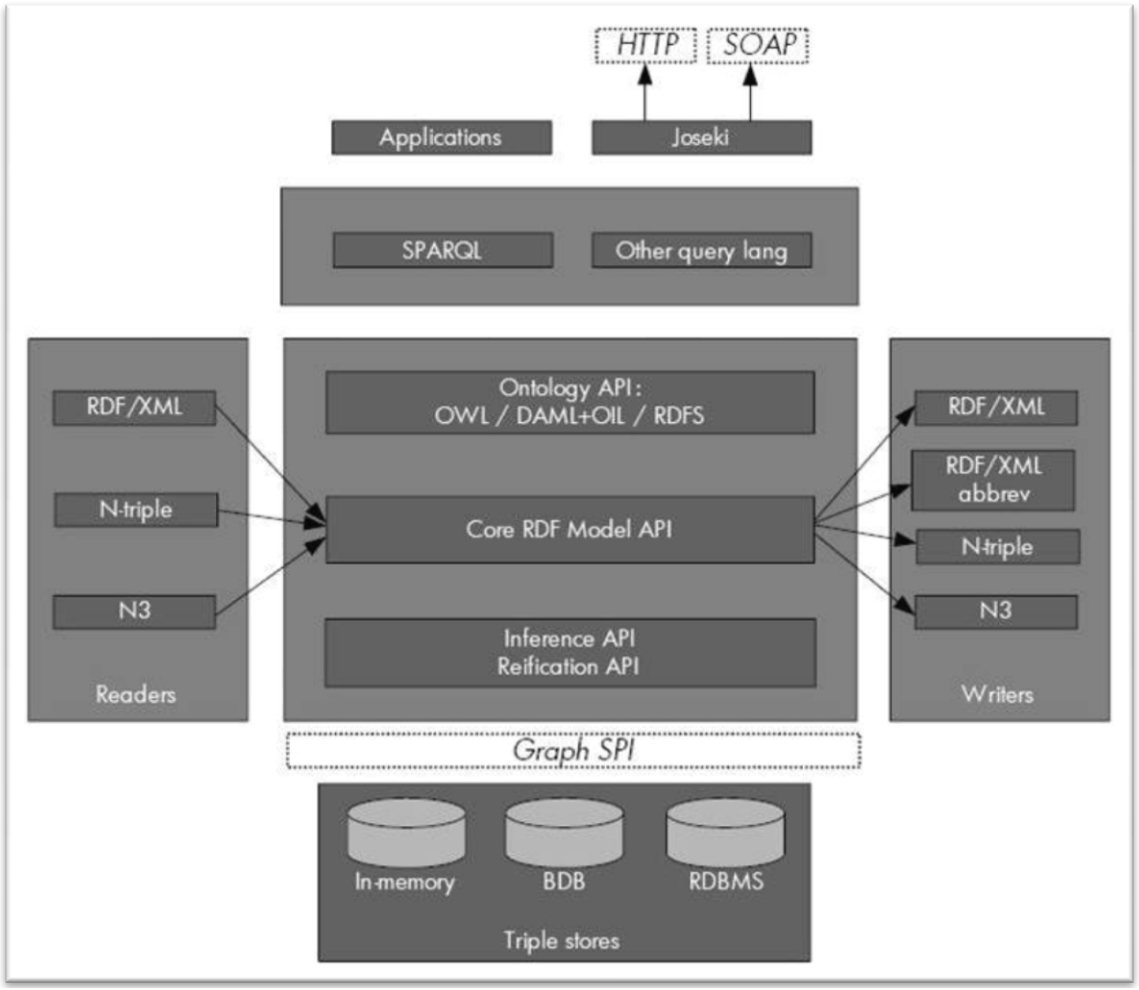
Οι OWL ιδιότητες μπορεί να καθοριστούν ως **συμμετρικές** (symmetric), **μεταβατικές**, **μονοσήμαντες** και **επί** (inverse functional). Επιπλέον, υποστηρίζονται οι σχέσεις αντιστοίχισης μεταξύ ιδιοτήτων **ισοδυναμία** και **αντιστροφή** (inversion), μέσω των δομών “owl:equivalentProperty” και “owl:inverseOf” αντίστοιχα.
Ιεραρχίες ιδιοτήτων μπορούν να σχηματιστούν με τη χρήση της δομής “rdfs:subPropertyOf”, που δηλώνει ότι μια ιδιότητα “κ1” αποτελεί υπο-ιδιότητα της ιδιότητας “κ”, την οποία εξειδικεύει. Μια OWL ιδιότητα μπορεί να είναι υπο-ιδιότητα περισσότερων από μιας ιδιοτήτων.
Υποστηρίζονται επίσης **OWL περιορισμοί** μέσω της δομής “owl:Restriction”, που συμπεριλαμβάνουν περιορισμούς τύπου (μέσω των δομών “owl:allValuesFrom” και “owl:someValuesFrom”), περιορισμούς πολλαπλότητας(μέσω των δομών “owl:cardinality”, “owl:minCardinality” και “owl:maxCardinality”) και περιορισμούς τιμής (μέσω της δομής “owl:hasValue”). Ως παράδειγμα, για να γίνουν κατανοητοί οι OWL περιορισμοί, στον παρακάτω πίνακα θα «τροποποιήσουμε την κλάση «Car». Συγκεκριμένα,με την προσθήκη του περιορισμού η πολλαπλότητα της ιδιότητας “hasDiskofrena” θα είναι ακριβώς 2. Ο περιορισμός αυτός δε μπορεί να εκφραστεί σε RDFS.

```
<owl:Class rdf:about="#DiskofrenaMembership">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource=”#hasDiskofrena”/>
      <owl:Cardinality rdf:datatype=”&xsd;nonNegativeInteger”>2  </owl:Cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Εικόνα 2.10 : Παράδειγμα δημιουργίας Cardinality Restriction

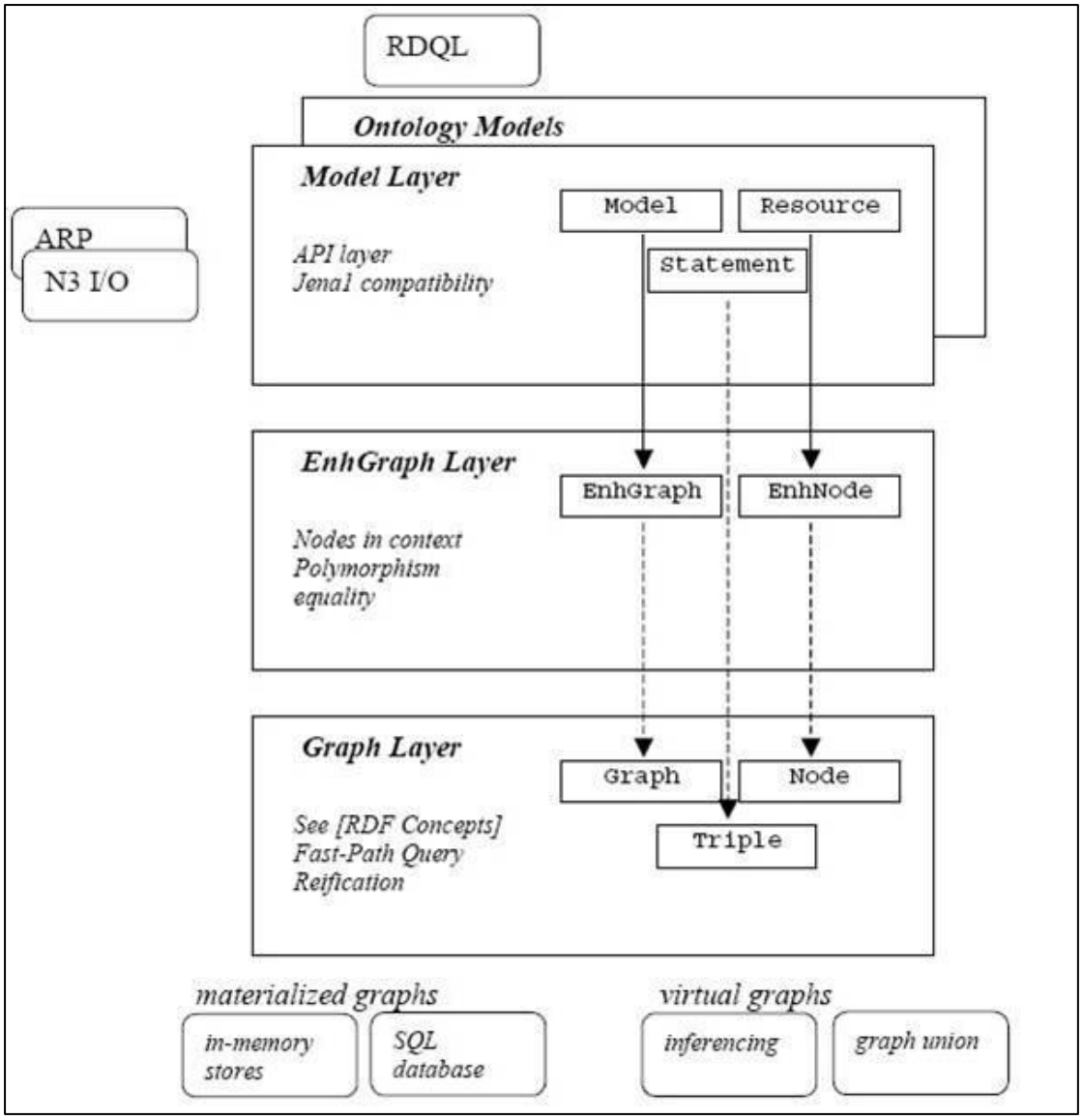
Τα **OWL άτομα**(OWL individual) αποτελούν τα μέλη των OWL κλάσεων. Ένα άτομο μπορεί να ανήκει σε μία ή περισσότερες κλάσεις και συνδέεται με τις κλάσεις στις οποίες ανήκει μέσω της δομής “rdf:type”. Τα άτομα διαθέτουν τις ιδιότητες των OWL κλάσεων όπου ανήκουν και υπακούν στους περιορισμούς που υπάρχουν γι’ αυτές.

2.5 Jena



Εικόνα 2.11 : Outline Jena Architecture

Το Jena είναι ένα open source Semantic Web framework για Java. Παρέχει ένα API για την εξαγωγή και εγγραφή δεδομένων σε RDF γράφους. Οι γράφοι αναπαρίστανται σαν ένα αφηρημένο μοντέλο.
Το μεγάλο πλεονέκτημα του Jena API σε σχέση με άλλα παρεμφερή frameworks (Sesame) είναι η δυνατότητα του για υποστήριξη της γλώσσας OWL.
Το Jena1 αρχικά εκδόθηκε το 2000 ενώ το Jena2 εκδόθηκε τον Αύγουστο του 2003. Η κύρια συνεισφορά του Jena1 είναι το εμπλουτισμένο Model API για τη διαχείριση RDF γραφημάτων. Γύρω από αυτό το API το Jena1 παρέχει πολλά εργαλεία, όπως: ένα RDF/XML parser, μία γλώσσα αναζήτησης (query language), επιπλέον I/O modules για N3 και N-triple και RDF/XML εξόδους. Το Jena1 παρέχει επιπρόσθετο API για το χειρισμό της DAML+OIL.
Το Jena2 παρέχει επιπλέον λειτουργικότητα με την υποστήριξη OWL και RDFS. Υπάρχουν νέα APIs για την προσπέλαση των οντολογιών, και επίσης προσφέρονται 2 νέα σημεία επέκτασης. Το πρώτο επιτρέπει την ανάπτυξη νέων APIs για νέα λειτουργικότητα στους σχεδιαστές εφαρμογών. Το δεύτερο επιτρέπει την ανάπτυξη νέου μηχανισμού τριάδων, όπως εικονικές τριάδες που δημιουργούνται δυναμικά και είναι προϊόντα ορισμένης επεξεργασίας.



Εικόνα 2.12 : The Jena2 Architecture

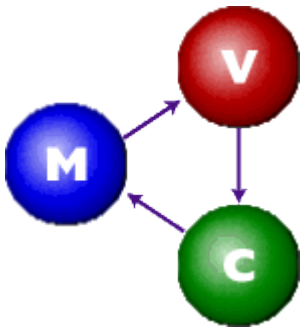
Οι δύο κύριοι στόχοι του Jena2 είναι :

- Πολλαπλές ευέλικτες αναπαραστάσεις RDF γραφημάτων. Αυτό επιτρέπει εύκολη πρόσβαση και διαχείριση των δεδομένων των γραφημάτων επιτρέποντας στον χρήστη – προγραμματιστή την προσπέλαση – πλοήγηση μέσω του μηχανισμού των τριάδων. Πιο συγκεκριμένα, το model API παρουσιάζει τα γραφήματα χρησιμοποιώντας συνθήκες και περιορισμούς από τα RDF recommendations και το Ontology API το οποίο παρουσιάζει τα γραφήματα με τη χρήση της OWL και της RDFS.
- Μία απλουστευμένη προβολή του RDF γραφήματος στο χρήστη με σκοπό την αναπαράσταση των δεδομένων με τη μορφή τριάδων. Αυτό είναι ιδιαίτερα χρήσιμο για τις προσεγγίσεις RDFS και OWL.

2.6 Swing

Το Swing είναι ένα εργαλείο για την Java. Είναι μέρος της Sun Microsystems' Java Foundation Classes(JFC) και συγκεκριμένα ένα API που παρέχει ένα γραφικό περιβάλλον στον χρήστη(graphical user interface-GUI) και τον βοηθά να υλοποιήσει Java προγράμματα και εφαρμογές.

Το Swing αναπτύχθηκε προκειμένου να παράσχει μια πιο σύνθετη σειρά γραφικών εργαλείων-συστατικών(GUI components) σε σχέση με τα ήδη υπάρχοντα AWT(Abstract Window Toolkit). Το Swing παρέχει ένα φυσικό look and feel που προσομοιώνει την εμφάνιση και την αίσθηση των πολλών πλατφόρμων και επίσης υποστηρίζει ένα άμεσα συνδέσιμο look and feel που επιτρέπει στις εφαρμογές να παρέχουν ένα look and feel άσχετο με την υποκείμενη –κάθε φορά – πλατφόρμα.



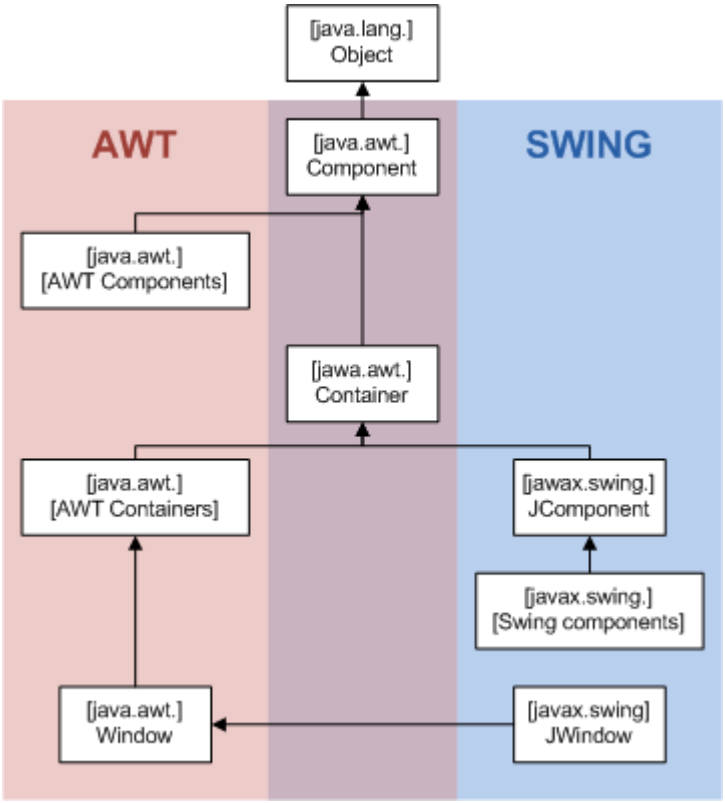
Εικόνα 2.13 : Το MVC μοντέλο

Το Swing είναι ένα εργαλείο –όπως προαναφέραμε-ανεξάρτητο από την υποκείμενη πλατφόρμα και αποτελεί ένα **Model-View-Controller** GUI framework για την Java. Ακολουθεί ένα μονονηματικό προγραμματιστικό μοντέλο.

2.6.1 Σύγκριση Swing με AWT

Τα πλεονεκτήματα του Swing έναντι του AWT είναι τα εξής :

- Περισσότερα συστατικά
- Επεκταμένα χαρακτηριστικά συστατικών
- Καλύτερα εμφάνιση και αίσθηση
- Καλύτερος χειρισμός συμβάντων
- Πιο συμβατά προγράμματα στις διάφορες πλατφόρμες (συστατικά πλήρως υλοποιημένα σε Java).



Εικόνα 2.14 : AWT and Swing class Hierarchy

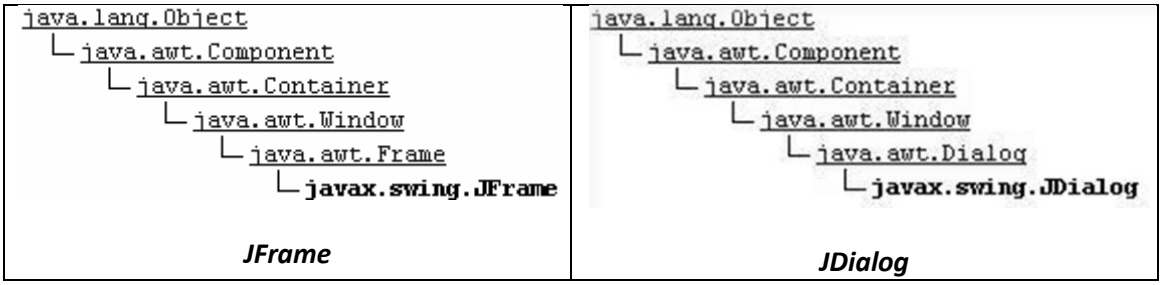
Οι ομοιότητες – διαφορές είναι οι εξής :

- Η βασική δομή του προγράμματος παραμένει.
- Οι βασικές έννοιες «υποδοχέας», «τομέας», «συστατικό» και «διαχειριστής διάταξης» παραμένουν.
- Η διαδικασία προσθήκης συστατικών διαφέρει.
- Η χρήση ενός συστατικού παραμένει ίδια.
- Αλλάζουν ελαφρώς τα ονόματα των κλάσεων (προσθήκη ενός J μπροστά συνήθως).
- Πακέτο : javax.swing

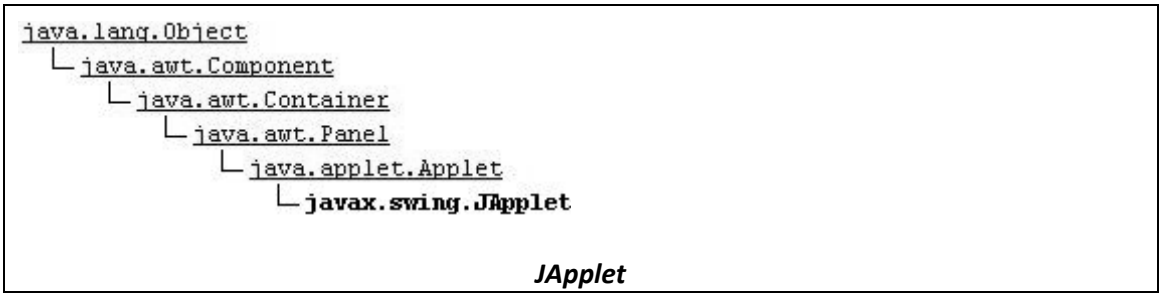
2.6.2 Top-Level Swing Containers και Swing Components

Κάθε γραφική διεπαφή τύπου Swing πρέπει να έχει τουλάχιστον ένα top-level Swing container. Ένα top-level Swing container παρέχει την απαραίτητη υποστήριξη που χρειάζονται τα Swing components για την εμφάνισή τους και την διαχείριση των γεγονότων που αυτά παράγουν. Υπάρχουν τρία top-level Swing containers: το JFrame, το JDialog, και (για applets) το JApplet. Κάθε JFrame αντικείμενο δημιουργεί ένα κύριο γραφικό παράθυρο, κάθε JDialog αντικείμενο δημιουργεί ένα δευτερεύον παράθυρο (δηλ παράθυρο που εξαρτάται από κάποιο άλλο παράθυρο). Κάθε JApplet αντικείμενο δημιουργεί την

περιοχή εμφάνισης ενός applet στο παράθυρο του Web Browser. Η ιεραρχία των top-level Swing containers δίνεται παρακάτω:

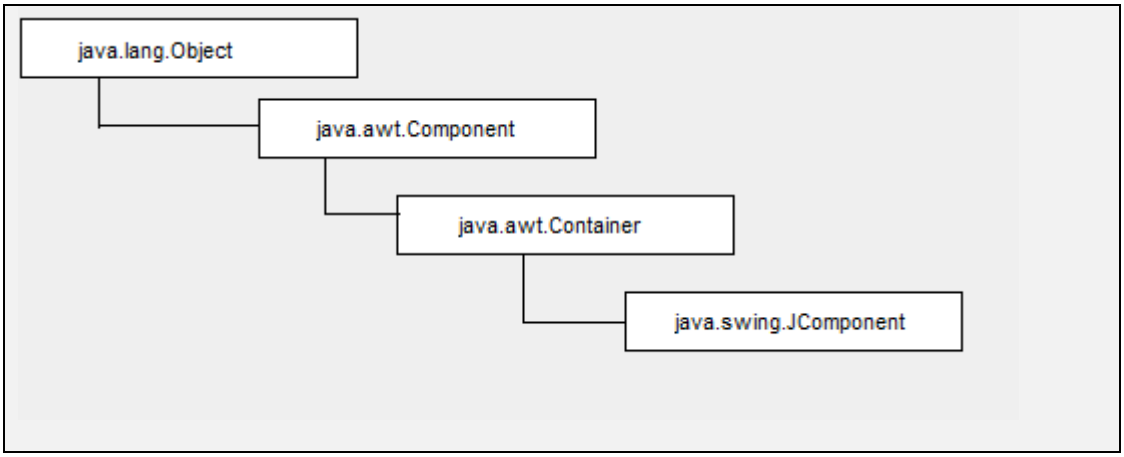


Εικόνα 2.15 : Η Ιεραρχία των JFrame και JDialog



Εικόνα 2.16 : Η Ιεραρχία της JApplet

Με εξαίρεση τα top-level containers, πχ: το JFrame, όλα τα Swing components είναι υποκλάσεις της κλάσης JComponent. Η ιεραρχία της JComponent φαίνεται στο παρακάτω σχήμα:



Εικόνα 2.17 : Η Ιεραρχία της JComponent

Ακολουθεί ένας πίνακας που παρουσιάζει την component-to-model χαρτογράφηση για την Java.

Component	Model Interface	Model Type
JButton	ButtonModel	GUI
JToggleButton	ButtonModel	GUI/data
JCheckBox	ButtonModel	GUI/data
JRadioButton	ButtonModel	GUI/data
JMenu	ButtonModel	GUI
JMenuItem	ButtonModel	GUI
JCheckBoxMenuItem	ButtonModel	GUI/data
JRadioButtonMenuItem	ButtonModel	GUI/data
JComboBox	ComboBoxModel	data
JProgressBar	BoundedRangeModel	GUI/data
JScrollBar	BoundedRangeModel	GUI/data
JSlider	BoundedRangeModel	GUI/data
JTabbedPane	SingleSelectionModel	GUI
JList	ListModel	data
JList	ListSelectionModel	GUI
JTable	TableModel	data
JTable	TableColumnModel	GUI
JTree	TreeModel	data
JTree	TreeSelectionModel	GUI
JEditorPane	Document	data
JTextPane	Document	data
JTextArea	Document	data
JTextField	Document	data
JPasswordField	Document	data

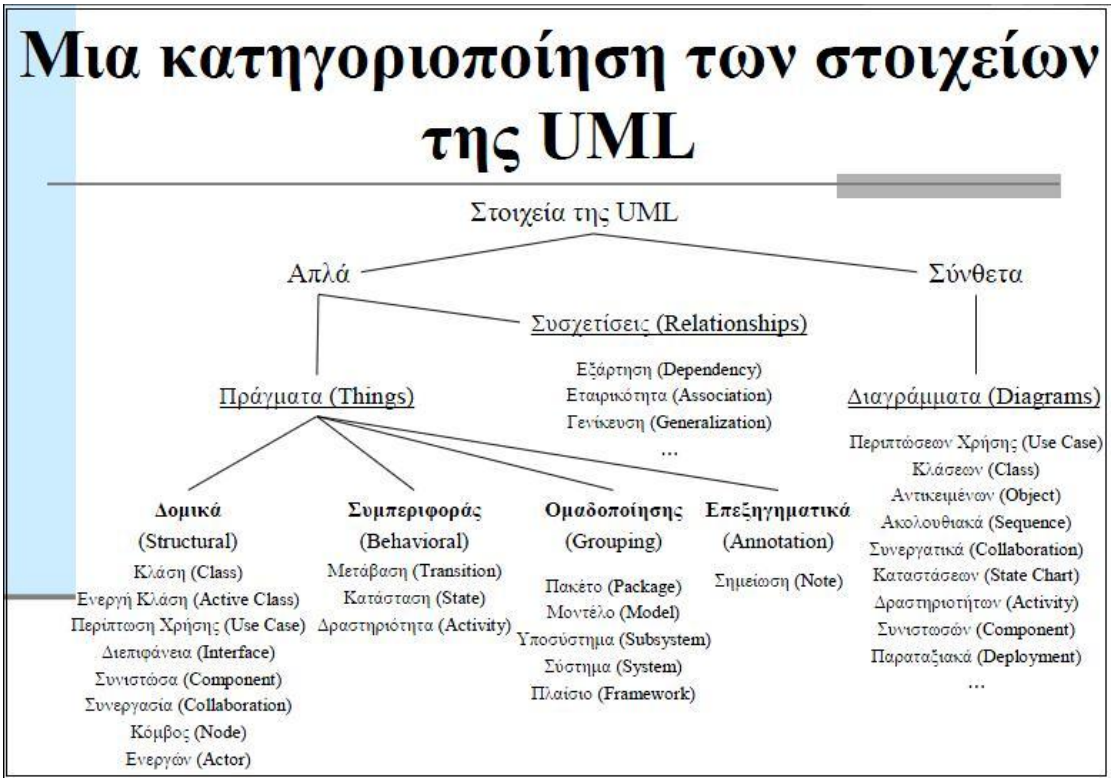
Πίνακας 2.2 : Η Component – to – Model χαρτογράφηση για την Java

2.7 UML

Η UML είναι μια οπτική αντικειμενοστρεφής γλώσσα μοντελοποίησης που χρησιμοποιείται για απεικόνιση (visualization), προδιαγραφή (specification), τεκμηρίωση (documentation) και κατασκευή (construction) των δομικών στοιχείων ενός συστήματος (λογισμικού ή όχι) . Είναι ιδιαίτερα εκφραστική και σημασιολογικά εκτενής γλώσσα, καθώς υποστηρίζει τη σημασιολογία τύπων και μοντέλων για όλα τα μοντέλα ενός συστήματος. Σε επίπεδο βασικών αρχών είναι μικρή και απλή και διακρίνουμε πέντε βασικούς άξονες. Είναι επεκτάσιμη γλώσσα, αφού υπάρχει η δυνατότητα εμπλουτισμού του μετα-μοντέλου με κλάσεις, ιδιότητες και σημασιολογία και είναι επακριβώς ορισμένη με βάση τα δομικά συστατικά ενός αντικειμενοστρεφούς συστήματος. Στα πλεονεκτήματα της συγκαταλέγεται και το γεγονός ότι ενσωματώνει τις ιδέες «καλής πρακτικής» από την βιομηχανία λογισμικού και υλοποιεί την ανάγκη της βιομηχανίας λογισμικού για μια ενιαία γλώσσα μοντελοποίησης.

Συστήματα που μοντελοποιούνται με την UML είναι συστήματα με έμφαση στο λογισμικό, επιχειρησιακά συστήματα και συστήματα που δεν περιέχουν λογισμικό. Οι πέντε βασικοί άξονες της UML είναι :

- Στοιχεία του μοντέλου (model elements)
- Συσχετίσεις (relationships)
- Μηχανισμοί (mechanisms)
- Διαγράμματα (diagrams)
- Αρχιτεκτονικές όψεις (architectural views)



Εικόνα 2.18 : Κατηγοριοποίηση στοιχείων της UML

2.7.1 Τα Δομικά Στοιχεία της UML

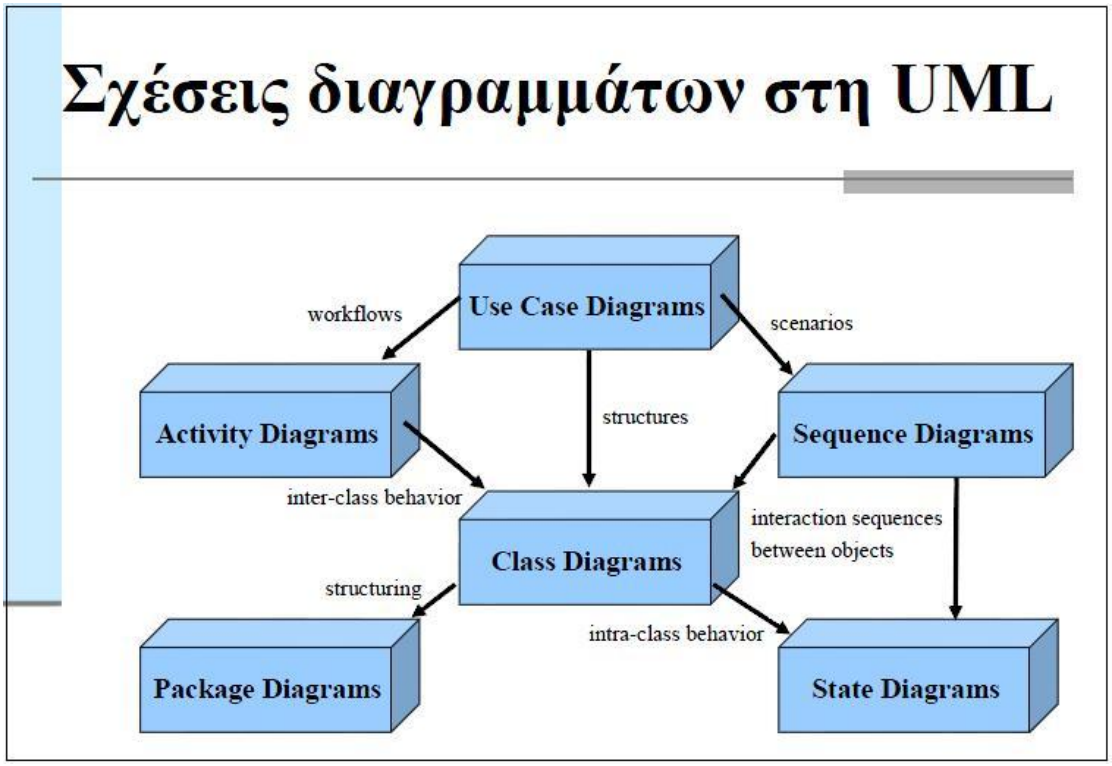
Τα δομικά στοιχεία της UML είναι τα εξής :

- Κλάση (class): Ένα σύνολο αντικειμένων με κοινή δομή και συμπεριφορά.
- Ενεργή κλάση (active class): Μια κλάση που περιγράφει μια διεργασία ή ένα νήμα εκτέλεσης και αλληλεπιδρά με άλλες.
- Περίπτωση χρήσης (use case): Μια λειτουργία που επιτελεί ένα σύστημα και είναι διαθέσιμη στο χρήστη. Είναι συμπεριφορά του συστήματος που συνεπάγεται τη συνεργασία ενός συνόλου αντικειμένων.
- Διεπιφάνεια (interface): Ένα σύνολο από λειτουργίες που ορίζουν την εξωτερική συμπεριφορά ενός αντικειμένου.
- Συνιστώσα (component): Ένα φυσικό και επαναχρησιμοποιήσιμο τμήμα ενός συστήματος, με λογική και φυσική υπόσταση που συνήθως υλοποιεί κάποιες διεπιφάνειες (interfaces).
- Συνεργασία (collaboration): Η περιγραφή μιας διάδρασης μεταξύ ενός συνόλου αντικειμένων.
- Κόμβος (node): Ένας υπολογιστικός πόρος που έχει κάποια μνήμη και υπολογιστική ικανότητα, οπότε εκεί αποθηκεύεται ή/και εκτελείται το λογισμικό.
- Ενεργών (actor): Εξωτερική του συστήματος οντότητα που χρησιμοποιεί τη λειτουργικότητα και τις διεπιφάνειές του.

2.7.2 Τα Στοιχεία Συμπεριφοράς της UML

Τα στοιχεία συμπεριφοράς της UML είναι η κατάσταση (state), η μετάβαση (transition) και η δραστηριότητα, ενώ τα στοιχεία ομαδοποίησης είναι το πακέτο (package), το υποσύστημα (subsystem) και το μοντέλο (model).
Οι τύποι διαγραμμάτων της UML είναι οι παρακάτω :

- Δομικά Διαγράμματα (Structural Diagrams): Περιγράφουν την εσωτερική λογική δομή ενός συστήματος, δηλαδή τα συστατικά του και τις σχέσεις μεταξύ τους. Στα δομικά διαγράμματα ανήκουν τα : Διαγράμματα Κλάσεων (Class Diagram), Διαγράμματα Αντικειμένων (Object Diagram), Διαγράμματα Συνιστωσών (Component Diagram), Παραταξιακό Διάγραμμα (Deployment Diagram).
- Διαγράμματα Συμπεριφοράς (Behavior Diagrams): Περιγράφουν τη δυναμική συμπεριφορά ενός συστήματος, δηλαδή την απόκρισή του σε γεγονότα του περιβάλλοντός του. Στα διαγράμματα συμπεριφοράς ανήκουν τα : Διάγραμμα Περιπτώσεων Χρήσης (**Use Case Diagram**), Διάγραμμα Αλληλουχίας (**Sequence Diagram**), Διάγραμμα _ραστηριοτήτων (**Activity Diagram**), Διάγραμμα Συνεργασίας (**Collaboration Diagram**), Διάγραμμα Καταστάσεων (**Statechart Diagram**).
- Διαγράμματα Διαχείρισης Μοντέλου (**Model Management Diagrams**): Περιγράφουν τη φυσική δομή ενός συστήματος, δηλαδή τις μονάδες λογισμικού που το αποτελούν, σε όρους περιβάλλοντος υλοποίησης. Στα διαγράμματα διαχείρισης μοντέλου ανήκουν τα : Διάγραμμα Πακέτων (**Package Diagram**), Διάγραμμα Υποσυστημάτων (**Subsystem Diagram**), Διάγραμμα Μοντέλων (**Model Diagram**).

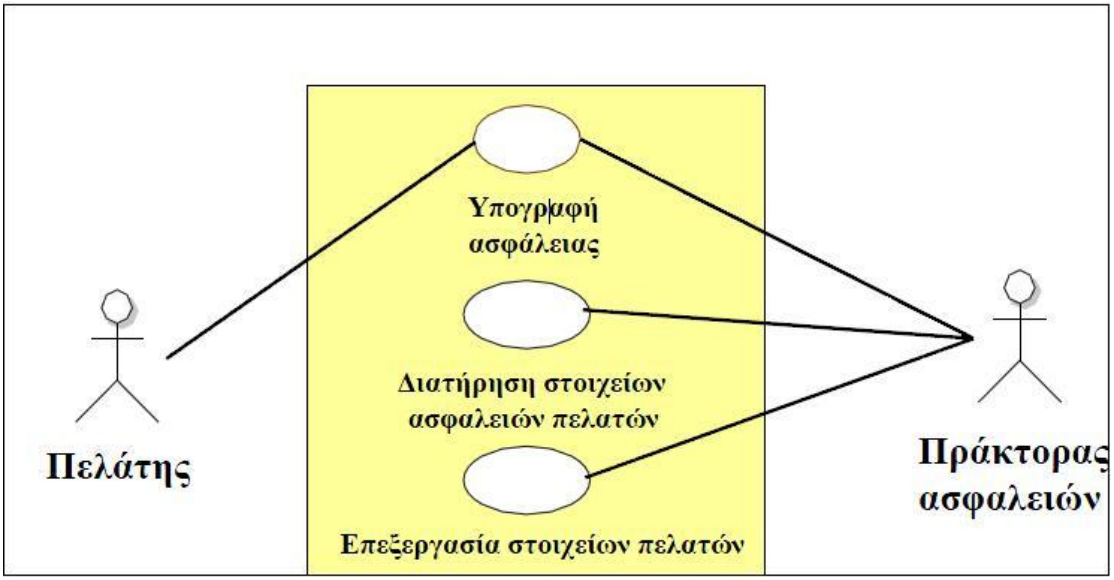


Εικόνα 2.19 : Σχέχεις Διαγραμμάτων στη UML

2.7.3 Παράδειγματα Use Case και Activity Διαγραμμάτων

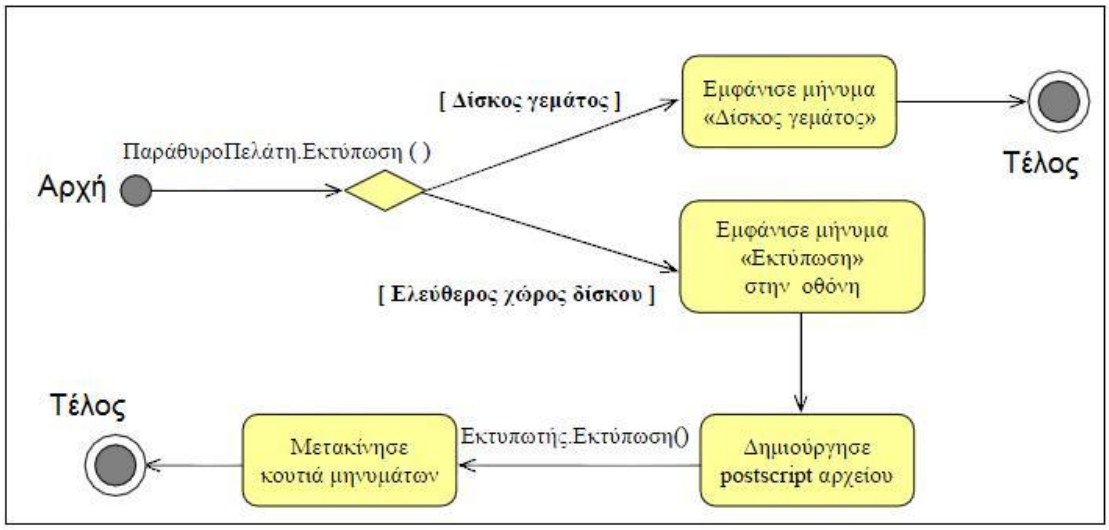
Στην παρούσα διπλωματική εργασία χρησιμοποιήσαμε μόνο τα **Use Case Diagrams** και τα **Activity Diagrams**.

Τα ***Use Case Diagrams*** περιγράφουν τη λειτουργικότητα του συστήματος όπως αυτή γίνεται αντιληπτή από εξωτερικές οντότητες (Actors).



Εικόνα 2.20 : Παράδειγμα ενός Use Case Diagram

Τα ***Activity Diagrams*** παρουσιάζουν την ακολουθιακή ροή των δραστηριοτήτων και περιέχουν προσδιορισμούς των μηνυμάτων που στέλνονται.



Εικόνα 2.21 : Παράδειγμα ενός Activity Diagram

3 Σχεδιασμός Συστήματος

3.1 Εισαγωγή

Η εφαρμογή σχεδιάστηκε με δυο διαφορετικούς τρόπους .

Αρχικά σχεδιάσαμε μια Berkeley XML παραμετροποιημένη βάση δεδομένων ,το σχήμα της οποίας μας έδινε την δυνατότητα να την χρησιμοποιήσουμε για αποθήκευση διαφορετικών προιόντων. Δηλαδή, με το ίδιο σχήμα μπορούσαμε να φτιάξουμε μια βάση για αυτοκίνητα και μια βάση για πόρτες .

Προχωρώντας όμως στην σχεδίαση του συστήματος διαπίστωσαμε ότι την παραμετροποιημένη αυτή XML βάση μπορούσαμε να την αντικαταστήσουμε με την οντολογία. Εξ ορισμού, η οντολογία δίνει την ζητούμενη παραμετροποίηση , επομένως προχώρησαμε στη δημιουργία μιας οντολογίας για το αυτοκίνητο(**CAR)**, με σταδιακή μετάβαση από το XML σχήμα , στο σχήμα της οντολογίας. Η μετάβαση αυτή θα περιγραφεί λεπτομερώς παρακάτω.

3.2 Σχεδιασμός Συστήματος με Berkeley XML DB

3.2.1 Πλατφόρμα Σχεδιασμού της Berkeley XML DB

Η παραμετροποιημένη βάση σχεδιάστηκε σε Berkeley DB XML 2.3.10. Η σχεδίαση του σχήματος της Berkeley DB XML έγινε με την χρήση του NetBeans IDE 6.1 και του εργαλείου που παρέχει για τον συγκεκριμένο σχεδιασμό, DB Visual Architect Professional Edition for Net Beans .

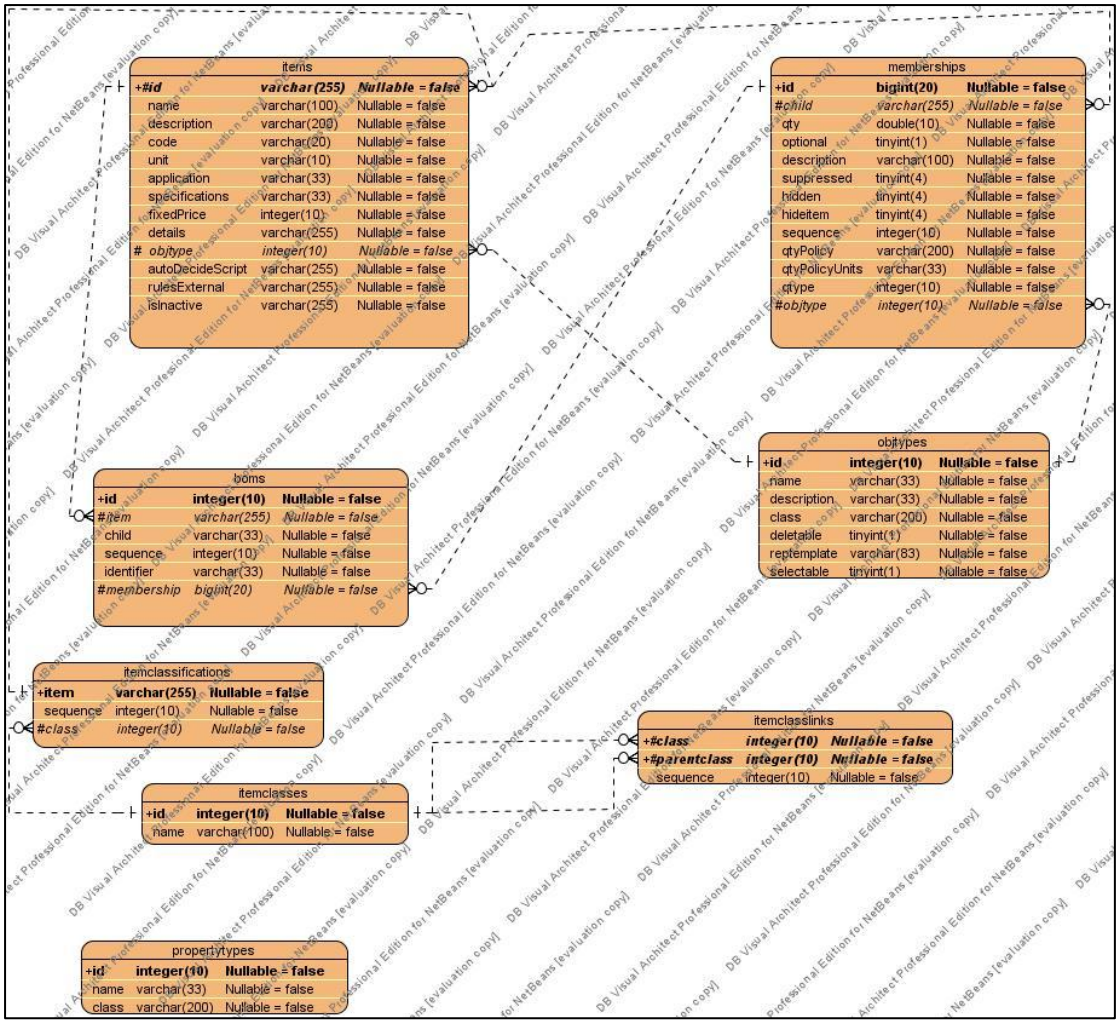
3.2.2 Λειτουργικότητα της Berkeley XML DB

Όπως προαναφέρθηκε στο εισαγωγικό μέρος του παρόντος κεφαλαίου, η Berkeley DB XML που σχεδιάστηκε παρέχει στον χρήστη της παραμετροποίηση. Δηλαδή :

A. Καθιστά δυνατή την δημιουργία μιας βάσης δεδομένων που μπορεί να δεχθεί δυο ή περισσότερα διαφορετικά προιόντα χωρίς να γίνει αλλαγή του σχήματος της βάσης.
B. Καθιστά δυνατή την δημιουργία βάσεων για διαφορετικά προιόντα με τον ίδιο σχήμα.

3.2.3 Παρουσίαση Σχήματος της Berkeley XML DB

Η ακόλουθη εικόνα απεικονίζει το σχήμα της Berkeley XML DB που υλοποιήθηκε για να πετύχουμε την απαιτούμενη παραμετροποίηση.



Εικόνα 3.1 : Το Σχήμα της Berkeley XML DB

Ακολουθεί ένα σχήμα της βάσης δεδομένων (αντίστοιχο του σχεσιακού σχήματος των σχεσιακών βάσεων), χάριν στο οποίο φαίνονται οι οντότητες, οι συσχετίσεις, τα κατηγορήματα, τα κλειδιά και τα foreign κλειδιά. (Με υπογράμμιση εμφανίζονται τα κύρια κλειδιά μίας συσχέτισης ή οντότητας, ενώ με **έντονα γράμματα** τα foreign keys) .

- *items* (id, name, description, code, unit, application, specifications, fixedPrice, details, autoDecideScript, rulesExternal, isInactive, **objtype**).

- *memberships* (id, qty, optional, description, suppressed, hidden, hideitem, sequence, qtyPolicy, qtyPolicyUnits, qtype, **child**, **objtype**).

- *boms* (id, **item**, child, sequence, identifier, **membership**).

- *objtypes* (id, name, description, class, deletable, reptemplate, selectable).

- *itemclassifications* (item, sequence, **class**).

- *itemclasses* (id, name).

- *itemclasslinks* (**class**, **parentclass**, sequence).
- *propertytypes* (id, name, class).

3.3 Σχεδιασμός Συστήματος με Οντολογία

3.3.1 Πλατφόρμα Σχεδιασμού Οντολογίας

Η οντολογία υλοποιήθηκε στο εργαλείο Protégé και στην έκδοση 4.beta.

3.3.2 Λειτουργικότητα Οντολογίας

Η Οντολογία **Car** περιγράφει ένα αυτοκίνητο. Συγκεκριμένα , περιγράφει :

- **Τα μέρη που το απαρτίζουν**. Δηλαδή περιγράφει το τιμόνι, τις πόρτες, το σασί και ούτω καθεξής .
- **Τα χαρακτηριστικά των αντικειμένων που απαρτίζουν το αυτοκίνητο**. Παραδείγματος χάριν : Για την μηχανή μας δίνεται η πληροφορία για τα κυβικά της .
- **Τον τρόπο που τα εξαρτήματα συνδέονται μεταξύ τους**. Παραδείγματος χάριν: Μας δείχνει τον τρόπο που η λυχνία μπαταρίας συνδέεται με την μπαταρία.
- **Τον τρόπο που που τα εξαρτήματα του αυτοκινήτου συνδέονται με το αυτοκίνητο**. Παραδείγματος χάριν: Μας δείχνει τον τρόπο που το τιμόνι και το σασί συνδέονται με το αυτοκίνητο .

3.3.3 Μετάβαση από το σχήμα της Berkeley XML DB στην Οντολογία

Η μετάβαση από το σχεσιακό σχήμα της XLM DB στην οντολογία έγινε σταδιακά με μια σειρά βημάτων. Συγκεκριμένα, τα βήματα που ακολουθήθηκαν είναι τα εξής :

- Αρχικά μη γνωρίζοντας τις by default δυνατότητες του σχήματος της Οντολογίας, προσπαθήσαμε να προσαρμόσουμε στην οντολογία το σχήμα της XLM DB και έτσι προέκυψαν τα παρακάτω δεδομένα .
  Γνωρίζουμε ότι το σχήμα της οντολογίας αποτελείται από :
  - Classes
  - Object Properties
  - Data Properties .

Συγκεκριμένα, τα classes είναι τα : **Item, Itemlocation, Membership, Objtype, Property.** Η περιγραφή της οντολογίας θα γίνει με επίκεντρο τα Classes. Επομένως, ξεκινάμε από το class **Item** .

Το class **Item** έχει τις εξής Superclasses:

- **Thing**
- **hasItemToProperty some Property**
- **inverseOfcontainsItem some Membership**
- **isMadeOf some Membership**
- **classifiedAs only Itemlocation**
- **hasItemToObjtype only Objtype .**

Η υπερκλάση **Thing** είναι η υπερκλάση όλων των κλάσεων που δημιουργούνται στο Protégé 4.

Η υπερκλάση **hasItemToProperty some Property** είναι μια 1-n σχέση που συνδέει την κλάση **Item** με την κλάση **Property .**

Η υπερκλάση **inverseOfcontainsItem some Membership** είναι μια 1-n σχέση που συνδέει την κλάση **Item** με την κλάση **Membership.**

Η υπερκλάση **isMadeOf some Membership** είναι μια 1-n σχέση που συνδέει την κλάση **Item** με την κλάση **Membership.**

Η υπερκλάση **classifiedAs only Itemlocation** είναι μια 1-1 σχέση που συνδέει την κλάση **Item** με την κλάση **Itemlocation.**

Η υπερκλάση **hasItemToObjtype only Objtype** είναι μια 1-1 σχέση που συνδέει την κλάση **Item** με την κλάση **Objtype.**

Το class **Itemlocation** έχει τις εξής Superclasses:

- **Thing**
- **inverseOfClassifiedAs some Item .**

Η υπερκλάση **Thing** είναι η υπερκλάση όλων των κλάσεων που δημιουργούνται στο Protégé 4.

Η υπερκλάση **inverseOfClassifiedAs some Item** είναι μια 1-n σχέση που συνδέει την κλάση **Itemlocation** με την κλάση **Item .**

Το class **Membership** έχει τις εξής Superclasses:

- **Thing**
- **hasMembershipToProperty some Property**
- **containsItem only Item**
- **hasMembershipToObjtype only Objtype**
- **inverseOfisMadeOf only Item .**

Η υπερκλάση **Thing** είναι η υπερκλάση όλων των κλάσεων που δημιουργούνται στο Protégé 4.

Η υπερκλάση **hasMembershipToProperty some Property** είναι μια 1-n σχέση που συνδέει την κλάση **Membership** με την κλάση **Property.**

Η υπερκλάση **containsItem only Item** είναι μια 1-1 σχέση που συνδέει την κλάση **Membership** με την κλάση **Item.**

Η υπερκλάση **hasMembershipToObjtype only Objtype** είναι μια 1-1 σχέση που συνδέει την κλάση **Membership** με την κλάση **Objtype.**

Η υπερκλάση **inverseOfisMadeOf only Item** είναι μια 1-1 σχέση που συνδέει την κλάση **Membership** με την κλάση **Item.**

Το class **Objtype** έχει την εξής Superclass:

- **Thing .**

Η υπερκλάση **Thing** είναι η υπερκλάση όλων των κλάσεων που δημιουργούνται στο Protégé 4.

Το class **Property** έχει την εξής Superclass:

- **Thing .**

Η υπερκλάση **Thing** είναι η υπερκλάση όλων των κλάσεων που δημιουργούνται στο Protégé 4.

Τα Object Properties είναι : **classifiedAs, containsItem, hasItemToObjtype, hasItemToProperty, hasMembershipToObjtype, hasMembershipToProperty, inverseOfClassifiedAs, inverseOfcontainsItem, inverseOfisMadeOf, inverseOfItemClassLinked, isMadeOf, itemClassLinked .**

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Object Properties σε αντιστοιχία με το domain και το range του καθενός. Στόχος είναι να γίνει κατανοητός ο τρόπος που συνδέονται τα classes μεταξύ τους, δηλαδή ποια σχέση – Object Property τα συνδέει.

Object Property	Domain	Range
classifiedAs	Item	Itemlocation
containsItem	Membership	Item
hasItemToObjtype	Item	Objtype
hasItemToProperty	Item	Property
hasMembershipToObjtype	Membership	Objtype
hasMembershipToProperty	Membership	Property
inverseOfClassifiedAs	Itemlocation	Item
inverseOfcontainsItem	Item	Membership
inverseOfisMadeOf	Membership	Item
inverseOfItemClassLinked	Itemlocation	Itemlocation
isMadeOf	Item	Membership
itemClassLinked	Itemlocation	Itemlocation

Πίνακας 3.1 : Object Properties, Domains, Ranges

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Object Properties σε αντιστοιχία με το inverse property και το characteristic του καθενός. Στόχος είναι να περιγραφούν τα επιπλέον χαρακτηριστικά που έχει το κάθε Object Property. Δηλαδή, να γίνει γνωστή η οποιαδήποτε σχέση αντιστροφής έχουν τα Object Properties μεταξύ τους και να γίνουν γνωστά κάποια επιπλέον στοιχεία της λειτουργικότητά τους, όπως το να είναι functional ή reflexive.

Object Property	Inverse Property	Characteristic
classifiedAs	inverseOfClassifiedAs	----------
containsItem	inverseOfcontainsItem	Functional
hasItemToObjtype	----------	----------
hasItemToProperty	----------	----------
hasMembershipToObjtype	----------	----------
hasMembershipToProperty	----------	----------
inverseOfClassifiedAs	classifiedAs	----------
inverseOfcontainsItem	containsItem	Functional
inverseOfisMadeOf	isMadeOf	Functional
inverseOfItemClassLinked	itemClassLinked	Reflexive
isMadeOf	inverseOfisMadeOf	Functional
itemClassLinked	inverseOfItemClassLinked	Reflexive

Πίνακας 3.2 : Object Properties, Inverse Properties, Characteristics

Τα Data Properties είναι : **itemApplication, itemAutoDecideScript, itemClassifiedAs, itemCode, itemContainsItem, itemDescription, itemDetails, itemFixedPrice, itemHasItemToObjtype, itemId, itemInverseOfIsMadeOf, itemIsInactive, itemItemLoc, itemLocClassifiedAs, itemLocId, itemLocName, itemName, itemObjtype, itemRulesExternal, itemSpecifications, itemUnit, memChild, memContainsItem, memDescription, memHasMemToObjtype, memHidden, memHideitem, memId, memInverseOfIsMadeOf, memObjtype, memOptional, memQty, memQtype, memQtyPolicy, memQtyPolicyUnits, memSequence, memSuppressed, objClass, objDeletable, objDescription, objHasItemToObjtype, objHasMemToObjtype, objId, objName, objReptemplate, objSelectable, propClass, propId, propItem, propMem, propName .**

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Data Properties σε αντιστοιχία με το domain και το range του καθενός. Στόχος είναι να συνδεθεί το κάθε Data Property με την αντίστοιχη κλάση στην οποία ανήκει και να δηλωθεί ο τύπος δεδομένων που μπορεί να λάβει το κάθε Data Property.

Data Property	Domain	Range
itemApplication	Item	string
itemAutoDecideScript	Item	string
itemClassifiedAs	classifiedAs **only** Itemlocation	string
itemCode	Item	string
itemContainsItem	containsItem **only** Item	string
itemDescription	Item	string
itemDetails	Item	string
itemFixedPrice	Item	string
itemHasItemToObjtype	hasItemToObjtype **only** Objtype	string
itemId	Item	string
itemInverseOfIsMadeOf	inverseOfIsMadeOf **only** Item	string
itemIsInactive	Item	string
itemItemLoc	Item	string
itemLocClassifiedAs	classifiedAs **only** Itemlocation	string
itemLocId	ItemLocation	string
itemLocName	ItemLocation	string
itemName	Item	string
itemObjtype	Item	string
itemRulesExternal	Item	string
itemSpecifications	Item	string
itemUnit	Item	string
memChild	Membership	string
memContainsItem	containsItem **only** Item	string
memDescription	Membership	string
memHasMemToObjtype	hasMembershipToObjtype **only** Objtype	string

memHidden	Membership	string
memHideitem	Membership	string
memId	Membership	string
memInverseOfIsMadeOf	inverseOfisMadeOf **only** Item	string
memObjtype	Membership	int
memOptional	Membership	int
memQty	Membership	string
memQtype	Membership	int
memQtyPolicy	Membership	string
memQtyPolicyUnits	Membership	string
memSequence	Membership	int
memSuppressed	Membership	int
objClass	Objtype	string
objDeletable	Objtype	int
objDescription	Objtype	string
objHasItemToObjtype	hasItemToObjtype **only** Objtype	int
objHasMemToObjtype	hasMembershipToObjtype **only** Objtype	int
objId	Objtype	int
objName	Objtype	string
objReptemplate	Objtype	string
objSelectable	Objtype	int
propClass	Property	string
propId	Property	int
propItem	Property	string
propMem	Property	string
propName	Property	string

Πίνακας 3.3 : Data Properties, Domains, Ranges

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Data Properties σε αντιστοιχία με το characteristic του καθενός. Στόχος είναι να περιγραφούν τα επιπλέον χαρακτηριστικά που έχει το κάθε Data Property. Δηλαδή, να γίνουν γνωστά κάποια επιπλέον στοιχεία της λειτουργικότητά τους, όπως το να είναι functional.

Data Property	Characteristic
itemApplication	Functional
itemAutoDecideScript	Functional
itemClassifiedAs	Functional
itemCode	Functional
itemContainsItem	Functional
itemDescription	Functional
itemDetails	Functional
itemFixedPrice	Functional
itemHasItemToObjtype	Functional
itemId	Functional
itemInverseOfIsMadeOf	Functional

itemIsInactive	Functional
itemItemLoc	Functional
itemLocClassifiedAs	Functional
itemLocId	Functional
itemLocName	Functional
itemName	Functional
itemObjtype	Functional
itemRulesExternal	Functional
itemSpecifications	Functional
itemUnit	Functional
memChild	Functional
memContainsItem	Functional
memDescription	Functional
memHasMemToObjtype	Functional
memHidden	Functional
memHideitem	Functional
memId	Functional
memInverseOfIsMadeOf	Functional
memObjtype	Functional
memOptional	Functional
memQty	Functional
memQtype	Functional
memQtyPolicy	Functional
memQtyPolicyUnits	Functional
memSequence	Functional
memSuppressed	Functional
objClass	Functional
objDeletable	Functional
objDescription	Functional
objHasItemToObjtype	Functional
objHasMemToObjtype	Functional
objId	Functional
objName	Functional
objReptemplate	Functional
objSelectable	Functional
propClass	Functional
propId	Functional
propItem	Functional
propMem	Functional
propName	Functional

Πίνακας 3.4 : Data Properties, Characteristics

- Κατόπιν, θεωρώντας και διαπιστώνοντας ότι μερικά από τα προαναφερθέντα classes καλύπτονται μέσα από την λειτουργικότητα και τον σχεδιασμό του protégé, το οποίο μας παρέχει την δυνατότητα να χρησιμοποιήσουμε Object Properties και Data Properties, προχωρήσαμε στις τελικές αλλαγές του γενικού σχήματος της οντολογίας .

3.3.4 Παρουσίαση Γενικού Σχήματος Οντολογίας

Οι κλάσεις που αρχικά είχαμε σχεδιάσει ήταν οι εξής : **Item, Itemlocation, Membership, Objtype, Property.**

Thing

- **Item**
- **Membership**
- **Itemlocation**
- **Objtype**
- **Property**

Οι classes όμως που καταλήξαμε είναι οι ακόλουθες: **ItemType, MemType** και **ObjType .** Δηλαδή, αφαιρέθηκαν οι κλάσεις **Itemlocation** και **Property.** Επιπλέον, το **Objtype** από απλή υποκλάση της κλάσης **Thing**, έγινε και υπερκλάση των κλάσεων **ItemType** και **MemType** (οι οποίες είναι οι κλάσεις **Item** και **Membership** που αναφέρθηκαν παραπάνω αντίστοιχα). Αυτό έγινε γιατί το **Objtype** περιγράφει τον τύπο ενός **ItemType** ή ενός **MemType.** Δηλαδή τα **ItemType** και **MemType** είναι instances του **Objtype .** Για να γίνει περισσότερο κατανοητή η αλλαγή που έγινε στο σχήμα της οντολογίας σας παραθέτουμε το ακόλουθο σχήμα :

Thing

- **ObjType**
  - **ItemType**
  - **MemType**

Έτσι βασιζόμενοι σε αυτό το γενικό σχήμα, «χτίσαμε» την οντολογία για το αυτοκίνητο, το σχήμα και τα instances της οποίας περιγράφονται αναλυτικά στο αντίστοιχο κεφάλαιο της εργασίας .

3.4 Συζήτηση – Σύγκριση των δύο Προσεγγίσεων

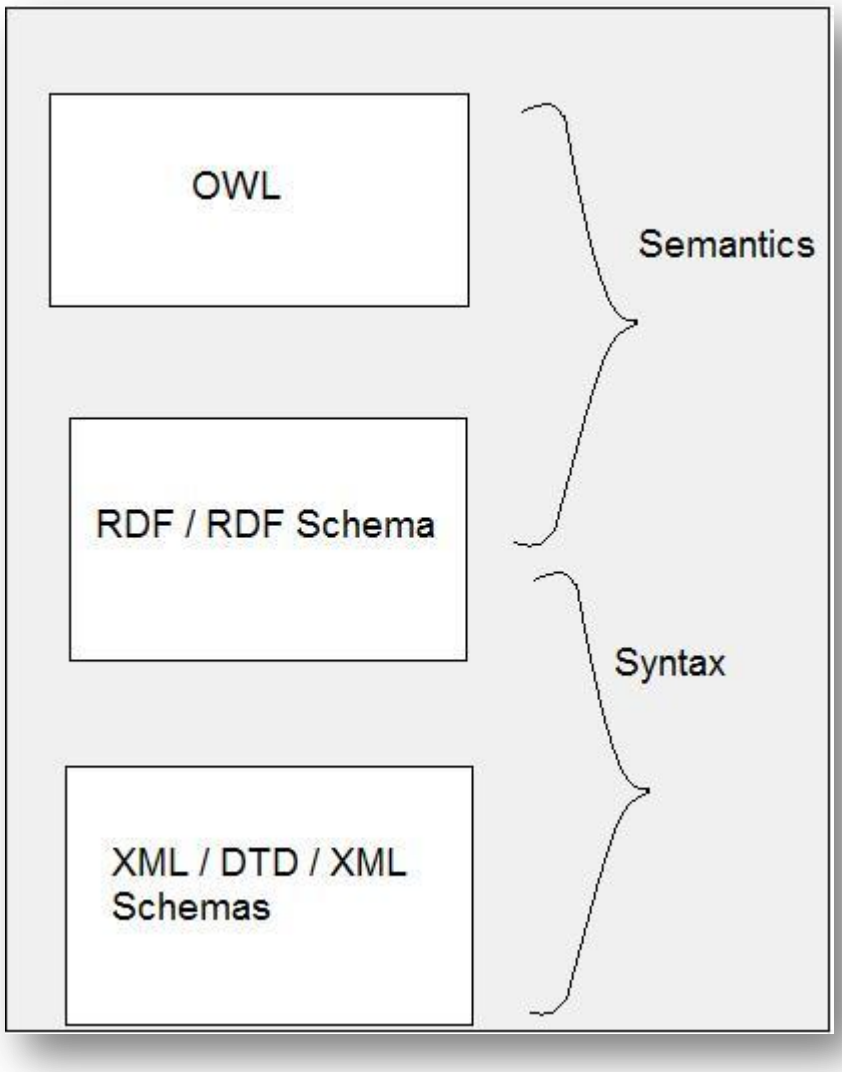
Περιγράφοντας τις διαφορές μεταξύ **XML** και **OWL**, γίνονται εύκολα κατανοητοί οι λόγοι που για την υλοποίηση της τελικής μας εφαρμογής χρησιμοποιήσαμε Οντολογία και όχι Berkeley XML DB.

Οι διαφορές μεταξύ XML και OWL είναι οι παρακάτω:

- Η θεμελιώδης διαφορά μεταξύ ενός XML εγγράφου και ενός RDF εγγράφου είναι στην δομή τους. Η δομή των XML εγγράφων είναι tree-based, ενώ του RDF εγγράφου είναι ένας γράφος. Αυτό οδηγεί σε διαφορετικές ιδιότητες των εγγράφων.
- Ένα σύνολο XML statements από μόνο του δεν μας επιτρέπει να καταλήξουμε σε κάποιο συμπέρασμα σχετικά με οποιαδήποτε άλλα XML statements. Ένα σύνολο όμως OWL statements μπορεί από μόνο του να μας επιτρέψει να καταλήξουμε σε κάποιο συμπέρασμα σχετικά με ένα άλλο OWL statement.
- Μια οντολογία διαφέρει από ένα XML Schema, στο ότι είναι μια αναπαράσταση της γνώσης και όχι μια μορφή μηνύματος. Το XML Schema και τα XML messages εστιάζουν στα συγκεκριμένα δεδομένα και η XML δεν επιτρέπει σε μας να ανακτήσουμε οποιαδήποτε πληροφορία βρίσκεται έξω από τα πλαίσια της συγκεκριμένης χρήσης. Η OWL διαφέρει από το XML Schema, γιατί μας επιτρέπει να καταγράφουμε δεδομένα σχετικά με ένα αντικείμενο, και έξω από τα όρια των συγκεκριμένων συναλλαγών που συνδέονται με τα δεδομένα.
- Για να δημιουργήσουμε ένα παραμετροποιημένο σχήμα για την Berkeley XML βάση δεδομένων, χρειάστηκε η προσθήκη αρκετών κλάσεων και σχέσεων μεταξύ των κλάσεων, ενώ το σχήμα της οντολογίας δίνει by default την καλύτερη δυνατή παραμετροποίηση. Επομένως, χρησιμοποιώντας οντολογία αντί της Berkeley XML βάσης δεδομένων καταφέρνουμε να μειώσουμε την πολυπλοκότητα του σχήματος, καθώς έχουμε λιγότερες κλάσεις και σχέσεις. Με άλλα λόγια καταφέραμε, μεταβαίνοντας από την κλασική προσέγγιση στην οντολογία, να αποφύγουμε το πολύ programming που απαιτούνταν και την πολυπλοκότητα στον σχεδιασμό.
- Από άποψη επεκτασιμότητας είναι προφανές ότι το αρχείο OWL, λόγω και της δομής του, μας επιτρέπει καλύτερη και ευκολότερη επεκτασιμότητα από αυτή που θα μπορούσε να μας προσφέρει ένα XML αρχείο. Αν, δηλαδή, θέλουμε να προσθέσουμε σε ένα σχήμα μια επιλέον κλάση, τότε το XML αρχείο θα πρέπει να τροποποιηθεί και να αλλάξει την δενδρική του δομή. Αυτό μπορεί να έχει δυσάρεστες συνέπειες απαιτώντας αλλαγές στα XSLT-style sheets, μετατρέποντας τελικά το αρχείο σε άλλα formats. Και τι γίνεται με την ενημέρωση όλων των XML εγγράφων προκειμένου να είναι συνεπή μετά τη νέα προσθήκη; Στην περίπτωση της οντολογίας όμως, μας δίνεται η δυνατότητα να προσθέσουμε μια καινούρια κλάση(έναν νέο κόμβο), χωρίς να γίνει καμία αλλαγή στην προηγουμένως δημιουργηθείσα δομή. Συνεπώς, μπορούμε απλά να επεκτείνουμε το σχήμα μας με νέα statements για κάθε αντικείμενο. Μάλιστα δεν είναι καν αναγκαίο να βάλουμε την καινούρια επέκταση

στον ίδιο φάκελο με το αντικείμενο. Μπορούμε να δημιουργήσουμε χωριστό φάκελο δηλώνοντας ότι ο ακόλουθος κατάλογος αντικειμένων έχει αλλάξει και έχει αποκτηθεί καινούρια ιδιότητα.

Άλλες Web γλώσσες όπως η RDFS, προχωρούν ένα βήμα πιο πέρα από την XML και μπορούν να υποστηρίξουν κάποια semantics. Η OWL επίσης, προσφέρει μια σειρά από στάνταρ ιδιότητες όπως η ισοδυναμία.



Εικόνα 3.2 : OWL enables machines to understand data

4 Οντολογία CAR και UML Σενάρια

4.1 Σχήμα Οντολογίας CAR

4.1.1 Classes

Σύμφωνα με την περιγραφή του σχήματος της οντολογίας , μας δίνεται μια υποκλάση της **Thing** , η **ObjType.** Η **ObjType** είναι υπερκλάση των **ItemType** και **MemType.**

Η **ItemType** με την σειρά της έχει τις εξής υποκλάσεις **: Aerosakos, Aerotomi, Anartisi, Car, Diaforiko, Diskofrena, KubotioTaxititon, Mixani, Orofi, Porta, Sasi, Timoni, Troxos, PCcontroller , Mpataria, Ntepozito.**
Η **MemType** έχει τις εξής υποκλάσεις **: AerosakosMembership, AerotomiMembership , AnartisiMembership, DiaforikoMembership, DiskofrenaMembership, Kubotio TaxititonMembership, MixaniMembership, OrofiMembership, PortaMembership, SasiMembership, TimoniMembership, TroxosMembership , PCcontrollerMembership, MpatariaMembership,**
NtepozitoMembership.

Επειδή οι περισσότερες υποκλάσεις της **ItemType** έχουν δικές τους υποκλάσεις και αν προχωρήσουμε στην απλή αναφορά τους το σχήμα της οντολογίας **CAR** θα γίνει δυσνόητο, παραθέτουμε παρακάτω ένα σχεδιάγραμμα με τις κλάσεις της συγκεκριμένης οντολογίας .

Thing

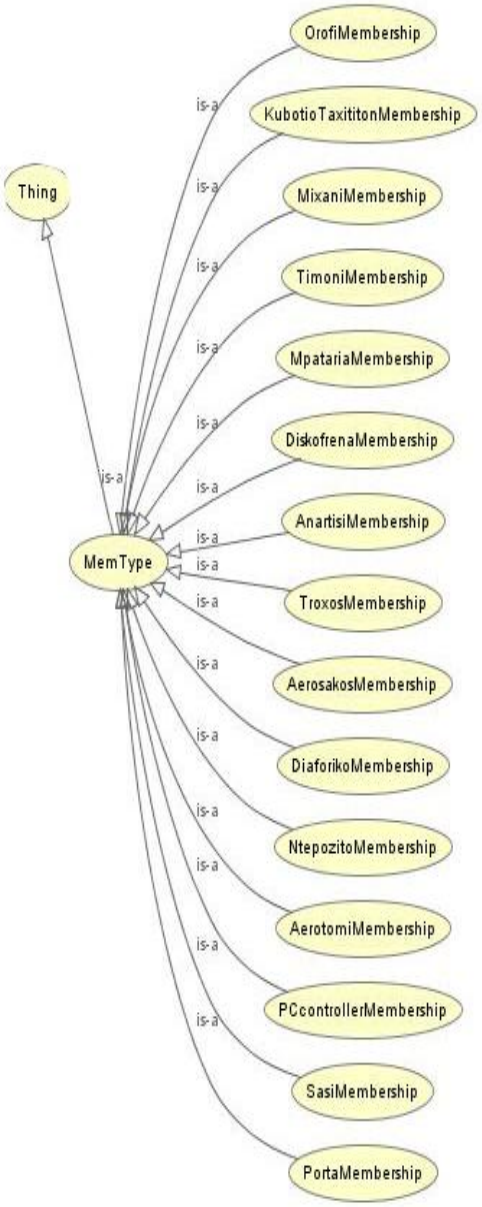
- **ObjType**
  - **ItemType**
    - **Aerosakos**
      - **AerosakosOdigou**
      - **AerosakosOrofis**
      - **AerosakosPleyrikos**
        - **AerosakosPleurikosAristera**
        - **AerosakosPleurikosDeksia**
      - **AerosakosSinodigou**
    - **Aerotomi**
    - **Anartisi**
      - **AnartisiMprosta**
      - **AnartisiPiso**
    - **Car**
    - **Diaforiko**
      - **DiaforikoMprosta**
      - **DiaforikoPiso**
    - **Diskofrena**
      - **DiskofrenaMprosta**
      - **DiskofrenaPiso**
    - **KubotioTaxititon**
      - **Automato**
      - **Mixaniko**
    - **Mixani**
      - **Benzini**
      - **Diesel**
      - **Hlektrokinitiras**
    - **Mpataria**
    - **Ntepozito**
    - **Orofi**
      - **Hliorofi**
      - **MetallikiOrofi**
    - **PCcontroller**
      - **LyxniaAerosakou**
      - **LyxniaDiskofrenon**
      - **LyxniaMixanis**
      - **LyxniaMpatarias**
      - **LyxniaNtepozitou**
      - **LyxniaPortas**
    - **Porta**
      - **Porta3thiro**
      - **Porta5thiro**

- **Sasi**
  - **Coupe**
  - **Kanoniko**
  - **Sedan**
- **Timoni**
- **Troxos**
  - **15Inches**
    - **15InchesAlouminiou**
    - **15InchesSiderou**
  - **16Inches**
    - **16InchesAlouminiou**
    - **16InchesSiderou**
  - **17Inches**
    - **17InchesAlouminiou**
    - **17InchesSiderou**

- **MemType**
  - **AerosakosMembership**
  - **AerotomiMembership**
  - **AnartisiMembership**
  - **DiaforikoMembership**
  - **DiskofrenaMembership**
  - **KubotioTaxititonMembership**
  - **MixaniMembership**
  - **MpatariaMembership**
  - **NtepozitoMembership**
  - **OrofiMembership**
  - **PCcontrollerMembership**
  - **PortaMembership**
  - **SasiMembership**
  - **TimoniMembership**
  - **TroxosMembership**

Ακολουθεί και το σχηματικό διάγραμμα της οντολογίας **CAR** όπως αυτό προέκυψε από τα εργαλεία του Protégé 4 . Για καλύτερη ευκρίνεια της εικόνας, έχει αφαιρεθεί κομμάτι του **MemType** και το παραθέτουμε σε διαφορετική σελίδα.


DiskofrenaPiso
Diskofrena
DiskofrenaMprosta
Mixaniko
KubotioTaxititon
Automato
Sedan
Sasi
Coupe
Kanoniko
AerosakosPleurikosAristera
AerosakosPleurikos
AerosakosPleyrikosDeksia
AerosakosOdigou
Ntepozito
AerosakosOrofis
Aerosakos
AerosakosSinodigou
AnartisiMprosta
Anartisi
AnartisiPiso
LyxniaNtepozitou
LyxniaMpatarias
LyxniaAerosakou
PCcontroller
LyxniaDiskofrenon
LyxniaPortas
LyxniaMixanis
Benzini
Mixani
Hlektrokinitiras
ItemType
Car
Diesel
Diaforiko
DiaforikoMprosta
DiaforikoPiso
Orofi
MetallikiOrofi
Aerotomi
Hliorofi
Porta
Porta5thiro
Troxos
Porta3thiro
17InchesSiderou
17Inches
17InchesAlouminiou
Mpataria
15Inches
15InchesSiderou
Timoni
15InchesAlouminiou
Thing
16Inches
16InchesAlouminiou
MemType
16InchesSiderou
is-a



Εικόνα 4.1 : Σχήμα Οντολογίας CAR

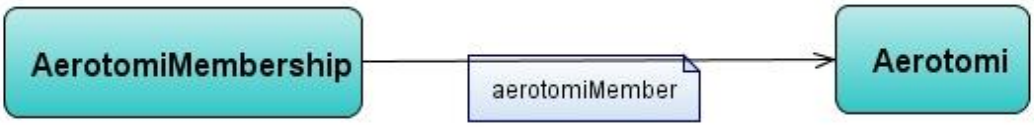
4.1.2 Object Properties

Μετά την παρουσίαση των classes του σχήματος της οντολογίας, σειρά έχουν τα Object Properties.

Τα Object Properties είναι τα εξής : **aerosakosMember, anartisiMember, diaforikoMember, diskofrenaMember, has3Portes, has5Portes, hasAerosakosOdigou, hasAerosakosOrofis, hasAerosakosPleurikosAristera, hasAerosakosPleurikosDeksia, hasAerosakosSinodigou, hasAnartisi, hasDiaforikoMprosta, hasDiaforikoPiso, hasDiskofrena, hasKubotioTaxititon, hasMixani, hasOrofi, hasSasi, hasTimoni, hasTroxos, kubotioTaxititonMember, mixaniMember, orofiMember, portaMember, sasiMember, timoniMember, troxosMember, aerotomiMember, hasAerotomi, hasLyxniaAerosakou, hasLyxniaDiskofrenon, hasLyxniaMixanis, hasLyxniaMpatarias, hasLyxniaNtepozitou, hasLyxniaPortas, hasMpataria, hasNtepozito, lyxniaBlabisGiaAerosakous, lyxniaBlabisGiaDiskofrena, lyxniaBlabisGiaMixani, lyxniaBlabisGiaMpataria, lyxniaBlabisGiaNtepozito, lyxniaBlabisGiaPortes, mpatariaMember, ntepozitoMember, pcControllerMember .**

Ακολουθεί με σχεδιαγράμματα ένα παράδειγμα που δείχνει τον τρόπο σύνδεσης της αεροτομής με το αυτοκίνητο. Τα Object Properties που περιγράφονται στους πίνακες που ακολουθούν μετά τα σχεδιαγράμματα, συνδέονται με τον ίδιο ακριβώς τρόπο .

Εικόνα 4.2 : Το Object Property hasAerotomi



Εικόνα 4.3 : Το Object Property aerotomiMember

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Object Properties σε αντιστοιχία με το domain και το range του καθενός. Στόχος είναι να γίνει κατανοητός ο τρόπος που συνδέονται τα classes μεταξύ τους, δηλαδή ποια σχέση – Object Property τα συνδέει.

Object Property	Domain	Range
hasTimoni	Car	TimoniMembership
timoniMember	TimoniMembership	Timoni
hasTroxos	Car	TroxosMembership
troxosMember	TroxosMembership	Troxos
hasSasi	Car	SasiMembership
sasiMember	SasiMembership	Sasi
has3Portes	Car	PortaMembership
has5Portes	Car	PortaMembership
portaMember	PortaMembership	Porta
hasOrofi	Car	OrofiMembership
orofiMember	OrofiMembership	Orofi
hasMixani	Car	MixaniMembership
mixaniMember	MixaniMembership	Mixani
hasKubotioTaxititon	Car	KubotioTaxititon Membership
kubotioTaxititonMember	KubotioTaxititon Membership	KubotioTaxititon
hasDiskofrena	Car	DiskofrenaMembership
diskofrenaMember	DiskofrenaMembership	Diskofrena
hasDiaforikoMprosta	Car	DiaforikoMembership
hasDiaforikoPiso	Car	DiaforikoMembership
diaforikoMember	DiaforikoMembership	Diaforiko
hasAnartisi	Car	AnartisiMembership
anartisiMember	AnartisiMembership	Anartisi
hasAerosakosOdigou	Car	AerosakosMembership
hasAerosakosOrofis	Car	AerosakosMembership
hasAerosakosPleurikos Aristera	Car	AerosakosMembership
hasAerosakosPleurikos Deksia	Car	AerosakosMembership
hasAerosakosSinodigou	Car	AerosakosMembership
aerosakosMember	AerosakosMembership	Aerosakos
hasAerotomi	Car	AerotomiMembership
aerotomiMember	AerotomiMembership	Aerotomi
hasMpataria	Car	MpatariaMembership
mpatariaMember	MpatariaMembership	Mpataria
hasNtepozito	Car	NtepozitoMembership
ntepozitoMember	NtepozitoMembership	Ntepozito
hasLyxniaAerosakou	Car	PCcontrollerMembership
hasLyxniaDiskofrenon	Car	PCcontrollerMembership
hasLyxniaMixanis	Car	PCcontrollerMembership
hasLyxniaMpatarias	Car	PCcontrollerMembership
hasLyxniaNtepozitou	Car	PCcontrollerMembership
hasLyxniaPortas	Car	PCcontrollerMembership
pcControllerMember	PCcontrollerMembership	PCcontroller

Πίνακας 4.1 : Object Properties, Domains, Ranges

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Object Properties σε αντιστοιχία με το cardinality του καθενός. Στόχος είναι να περιγραφούν τα επιπλέον χαρακτηριστικά που έχει το κάθε Object Property. Δηλαδή, να γίνει γνωστό το πλήθος του κάθε εξαρτήματος που συνδέεται στο αυτοκίνητο(1 τιμόνι, 4 πόρτες κτλ).

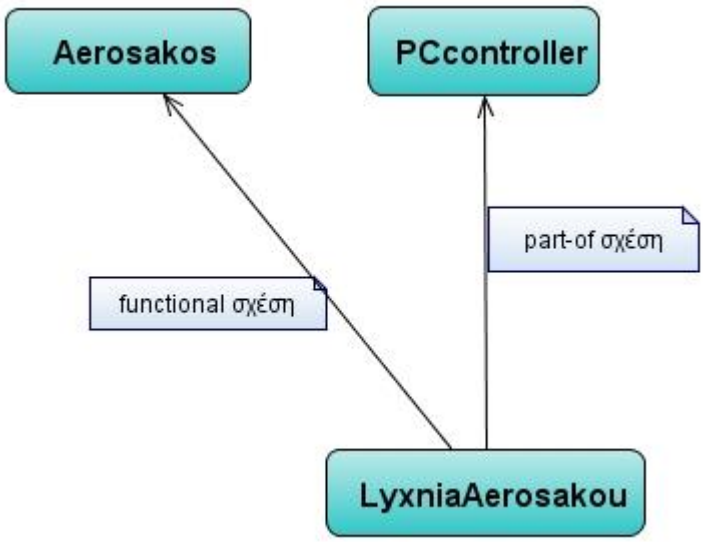
Object Property	Cardinality
hasTimoni	hasTimoni **exactly 1** TimoniMembership
timoniMember	----------
hasTroxos	hasTroxos **exactly 5** TroxosMembership
troxosMember	----------
hasSasi	hasSasi **exactly 1** SasiMembership
sasiMember	----------
has3Portes	has3Portes **exactly 3** PortaMembership
has5Portes	has5Portes **exactly 5** PortaMembership
portaMember	----------
hasOrofi	hasOrofi **exactly 1** OrofiMembership
orofiMember	----------
hasMixani	hasMixani **exactly 1** MixaniMembership
mixaniMember	----------
hasKubotioTaxititon	hasKubotioTaxititon **exactly 1** KubotioTaxititonMembership
kubotioTaxititonMember	----------
hasDiskofrena	hasDiskofrena **exactly 2** DiskofrenaMembership
diskofrenaMember	----------
hasDiaforikoMprosta	hasDiaforikoMprosta **exactly 2** DiaforikoMembership
hasDiaforikoPiso	hasDiaforikoPiso **exactly 2** DiaforikoMembership
diaforikoMember	----------
hasAnartisi	hasAnartisi **exactly 2** AnartisiMembership
anartisiMember	----------
hasAerosakosOdigou	hasAerosakosOdigou **exactly 1** AerosakosMembership
hasAerosakosOrofis	hasAerosakosOrofis **exactly 4** AerosakosMembership
hasAerosakosPleurikos Aristera	hasAerosakosPleurikosAristera **exactly 2** AerosakosMembership
hasAerosakosPleurikos Deksia	hasAerosakosPleurikosDeksia **exactly 2**AerosakosMembership
hasAerosakosSinodigou	hasAerosakosSinodigou **exactly 1** AerosakosMembership
aerosakosMember	----------
hasAerotomi	hasAerotomi **exactly 1** AerotomiMembership
aerotomiMember	----------
hasMpataria	hasMpataria **exactly 1** MpatariaMembership
mpatariaMember	----------
hasNtepozito	hasNtepozito **exactly 1** NtepozitoMembership
ntepozitoMember	----------
hasLyxniaAerosakou	hasLyxniaAerosakou **exactly 1** PCcontrollerMembership
hasLyxniaDiskofrenon	hasLyxniaDiskofrenon **exactly 1** PCcontrollerMembership

hasLyxniaMixanis	hasLyxniaMixanis **exactly 1** PCcontrollerMembership
hasLyxniaMpatarias	hasLyxniaMpatarias **exactly 1** PCcontrollerMembership
hasLyxniaNtepozitou	hasLyxniaNtepozitou **exactly 1** PCcontrollerMembership
hasLyxniaPortas	hasLyxniaPortas **exactly 1** PCcontrollerMembership
pcControllerMember	----------

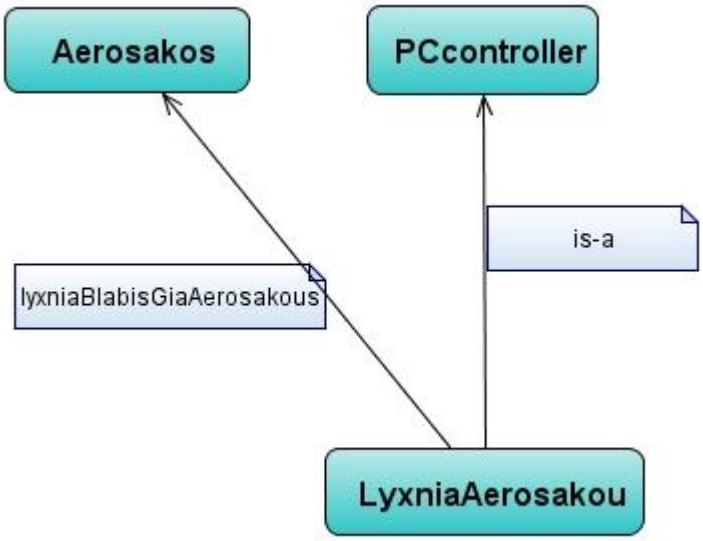
Πίνακας 4.2 : Object Properties, Cardinalities

Ακολουθεί (με σχεδιαγράμματα) ένα παράδειγμα που δείχνει τον ρόλο των υπόλοιπων Object Properties.Τα Object Properties που ακολουθούν μετά τα σχεδιαγράμματα συνδέονται με τον ίδιο ακριβώς τρόπο .

Στο πρώτο διάγραμμα στόχος είναι να αναδειχτούν οι part-of και functional σχέσεις και στο δεύτερο να παρουσιαστούν τα ονόματα αυτών των σχέσεων.



Εικόνα 4.4 : Παράδειγμα με Part – of και Functional Σχέσεις



Εικόνα 4.5 : Παράδειγμα με Part – of και Functional Σχέσεις

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Object Properties σε αντιστοιχία με το domain και το range του καθενός. Στόχος είναι να γίνει κατανοητός ο τρόπος που συνδέονται τα classes μεταξύ τους, δηλαδή ποια σχέση – Object Property τα συνδέει.

Object Property	Domain	Range
lyxniaBlabisGiaAerosakous	LyxniaAerosakou	Aerosakos
lyxniaBlabisGiaDiskofrena	LyxniaDiskofrenon	Diskofrena
lyxniaBlabisGiaMixani	LyxniaMixanis	Mixani
lyxniaBlabisGiaMpataria	LyxniaMpatarias	Mpataria
lyxniaBlabisGiaPortes	LyxniaPortas	Porta
lyxniaBlabisGiaNtepozito	LyxniaNtepozitou	Ntepozito

Πίνακας 4.3 : Object Properties, Domains, Ranges

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Object Properties σε αντιστοιχία με το characteristic του καθενός. Στόχος είναι να περιγραφούν τα επιπλέον χαρακτηριστικά που έχει το κάθε Object Property. Δηλαδή, να γίνουν γνωστά κάποια επιπλέον στοιχεία της λειτουργικότητά τους, όπως το να είναι functional.

Object Property	Characteristic
lyxniaBlabisGiaAerosakous	Functional
lyxniaBlabisGiaDiskofrena	Functional
lyxniaBlabisGiaMixani	Functional
lyxniaBlabisGiaMpataria	Functional
lyxniaBlabisGiaPortes	Functional
lyxniaBlabisGiaNtepozito	Functional

Πίνακας 4.4 : Object Properties, Characteristics

4.1.3 Data Properties

Μετά την παρουσίαση των classes και των Object Properties του σχήματος της οντολογίας, σειρά έχουν τα Data Properties.

Τα Data Properties είναι τα εξής : **eidosDiskofrenon, ippoiMixanis, kubikaMixanis, locationTimoni, sasiCodeNo, sxeseisKibotiouTaxititon, typeAnartisis .**

Στον παρακάτω πίνακα παρουσιάζονται όλα τα Data Properties σε αντιστοιχία με το domain και το range του καθενός. Στόχος είναι να συνδεθεί το κάθε Data Property με την αντίστοιχη κλάση στην οποία ανήκει και να δηλωθεί ο τύπος δεδομένων που μπορεί να λάβει το κάθε Data Property.

Data Property	Domain	Range
eidosDiskofrenon	Diskofrena	string
ippoiMixanis	Mixani	int
kubikaMixanis	Mixani	int
locationTimoni	TimoniMembership	string
sasiCodeNo	Sasi	string
sxeseisKubotiouTaxititon	KubotioTaxititon	int
typeAnartisis	Anartisi	string

Πίνακας 4.5 : Data Properties, Domains, Ranges

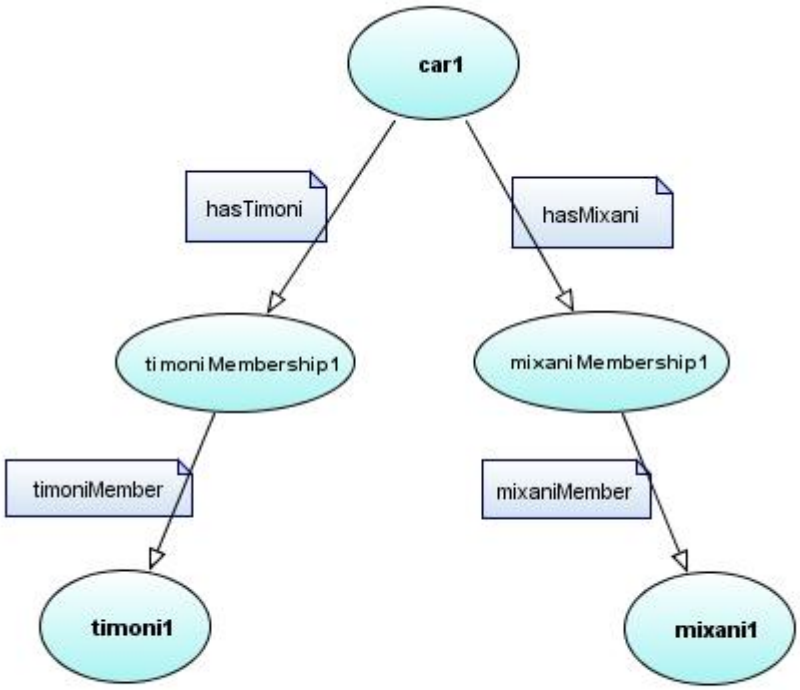
4.2 Instances Οντολογίας CAR

Ως instances στο protégé 4 λέγονται τα individuals . Αφού φτιάξαμε το σχήμα για την οντολογία **CAR** , επόμενο βήμα ήταν η δημιουργία των instances.

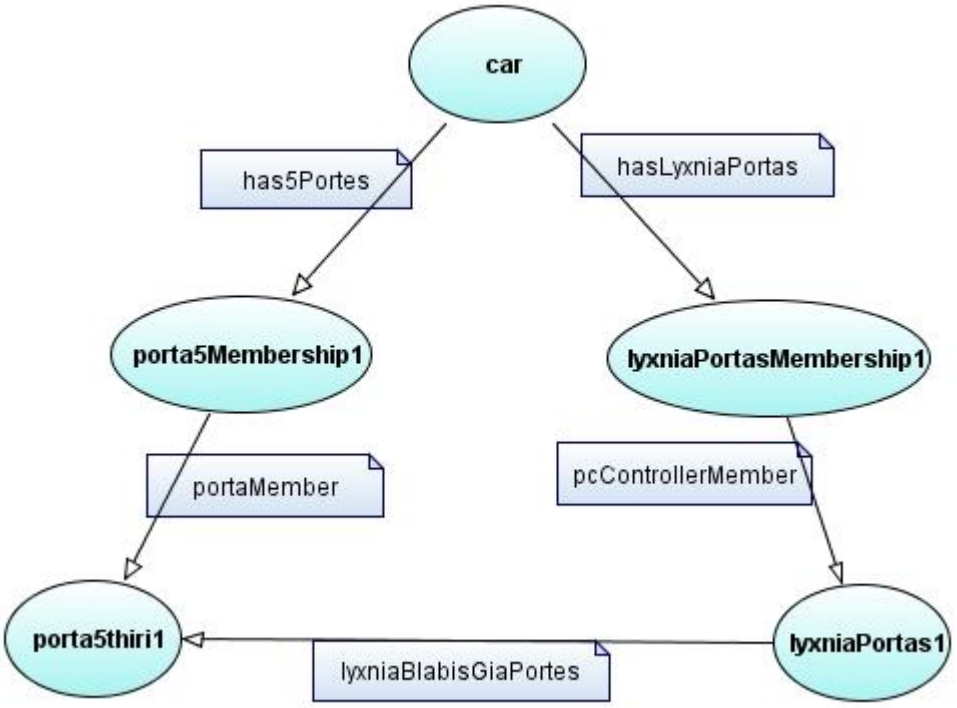
Για παράδειγμα, δημιουργήσαμε ένα individual **car1** που είναι τύπου Car και κατόπιν individuals για τιμόνι, για πόρτες , μηχανή, οροφή και ούτω καθεξής, τα οποία συνδέθηκαν με το **car1 .**

Συγκεκριμένα φτιάξαμε τα εξής individuals : **aerosakosOdigou1, aerosakosOdigouMembership1, aerosakosOrofis1, aerosakosOrofisMembership1, aerosakosPleurikosAristera1, aerosakosPleurikosAristeraMembership1, aerosakosPleurikosDeksia1, aerosakosPleurikosDeksiaMembership1, aerosakosSinodigou1, aerosakosSinodigouMembership1, anartisiMprosta1, anartisiMprostaMembership1, anartisiPiso1, anartisiPisoMembership1, car1, diaforikoMprosta1, diaforikoMprostaMembership1, diaforikoPiso1, diaforikoPisoMembership1, diskofrenaMprosta1, diskofrenaMprostaMembership1, diskofrenaPiso1, diskofrenaPisoMembership1, iliorofi1, kubotioTaxititonAutomato1, kubotioTaxititonAutomatoMembership1, kubotioTaxititonMixaniko1, kubotioTaxititonMixanikoMembership1, metallikiOrofi1, mixaniBenzini1, mixaniBenziniMembership1, mixaniDiesel1, mixaniDieselMembership1, mixaniHlektrokinitiras1, mixaniHlektrokinitirasMembership1, orofiMembershipHliorofi1, orofiMembershipMetallikiOrofi1, porta3Membership1, porta3thiri1, porta5Membership1, porta5thiri1, sasiCoupe1, sasiCoupeMembership1, sasiSedan1, sasiSedanMembership1, sasiKanoniko1, sasiKanonikoMembership1, timoni1, timoniMembership1, troxos15InchesAlouminiou1, troxos15 InchesAlouminiouMembership1, troxos15InchesSiderou1, troxos15 InchesSiderouMembership1, troxos16InchesAlouminiou1, troxos16 InchesAlouminiouMembership1, troxos16InchesSiderou1, troxos16 InchesSiderouMembership1, troxos17InchesAlouminiou1, troxos17 InchesAlouminiouMembership1, troxos17InchesSiderou1, troxos17 InchesSiderouMembership1 , aerotomi1, aerotomiMembership1, lyxniaAerosakou1, lyxniaAerosakouMembership1, lyxniaDiskofrenon1, lyxniaDiskofrenonMembership1, lyxniaMixanis1, lyxniaMixanisMembership1, lyxniaMpatarias1, lyxniaMpatariasMembership1, lyxniaNtepozitou1, lyxniaNtepozitouMembership1, lyxniaPortas1, lyxniaPortasMembership1, mpataria1, mpatariaMembership1, ntepozito1, ntepozitoMembership1 .**

Για να γίνει καλύτερα κατανοητή η σύνδεση όλων των individuals, από τα οποία αποτελείται το αυτοκίνητο, με το individual **car1,** ακολουθεί ένα διάγραμμα που δείχνει τον τρόπο σύνδεσης της μηχανής και του τιμονιού στο αυτοκίνητο και τον τρόπο σύνδεσης της πόρτας και της προειδοποιητικής λυχνίας για ανοιχτή πόρτα, με το αυτοκίνητο.



Εικόνα 4.6 : Παράδειγμα Σύνδεσης των Individuals



Εικόνα 4.7 : Παράδειγμα Σύνδεσης των Individuals

Στους παρακάτω πίνακες γίνεται αναφορά των Individuals που υλοποιήθηκαν και δίνονται κάποιες πληροφορίες γι αυτά. Δηλαδή, κάθε Individual αντιστοιχίζεται με την κλάση της οποίας είναι στιγμιότυπο και κάποια Data Property ή Object Property Assertions .

Individual	Type
aerosakosOdigou1	AerosakosOdigou
aerosakosOdigouMembership1	AerosakosMembership
aerosakosOrofis1	AerosakosOrofis
aerosakosOrofisMembership1	AerosakosMembership
aerosakosPleurikosAristera1	AerosakosPleurikosAristera
aerosakosPleurikosAristeraMembership1	AerosakosMembership
aerosakosPleurikosDeksia1	AerosakosPleurikosDeksia
aerosakosPleurikosDeksiaMembership1	AerosakosMembership
aerosakosSinodigou1	AerosakosSinodigou
aerosakosSinodigouMembership1	AerosakosMembership
anartisiMprosta1	AnartisiMprosta
anartisiMprostaMembership1	AnartisiMembership
anartisiPiso1	AnartisiPiso
anartisiPisoMembership1	AnartisiMembership
diaforikoMprosta1	DiaforikoMprosta
diaforikoMprostaMembership1	DiaforikoMembership
diaforikoPiso1	DiaforikoPiso
diaforikoPisoMembership1	DiaforikoMembership
diskofrenaMprosta1	DiskofrenaMprosta
diskofrenaMprostaMembership1	DiskofrenaMembership

diskofrenaPiso1	DiskofrenaPiso
diskofrenaPisoMembership1	DiskofrenaMembership
iliorofi1	Hliorofi
orofiMembershipHliorofi1	OrofiMembership
metallikiOrofi1	MetallikiOrofi
orofiMembershipMetallikiOrofi1	OrofiMembership
kubotioTaxititonAutomato1	Automato
kubotioTaxititonAutomatoMembership1	KubotioTaxititonMembership
kubotioTaxititonMixaniko1	Mixaniko
kubotioTaxititonMixanikoMembership1	KubotioTaxititonMembership
mixaniBenzini1	Benzini
mixaniBenziniMembership1	MixaniMembership
mixaniDiesel1	Diesel
mixaniDieselMembership1	MixaniMembership
mixaniHlektrokinitiras1	Hlektrokinitiras
mixaniHlektrokinitirasMembership1	MixaniMembership
porta3thiri1	Porta3thiro
porta3Membership1	PortaMembership
porta5thiri1	Porta5thiro
porta5Membership1	PortaMembership
sasiCoupe1	Coupe
sasiCoupeMembership1	SasiMembership
sasiKanoniko1	Kanoniko
sasiKanonikoMembership1	SasiMembership
sasiSedan1	Sedan
sasiSedanMembership1	SasiMembership
timoni1	Timoni
timoniMembership1	TimoniMembership
troxos15InchesAlouminiou1	15InchesAlouminiou
troxos15InchesAlouminiouMembership1	TroxosMembership
troxos16InchesAlouminiou1	16InchesAlouminiou
troxos16InchesAlouminiouMembership1	TroxosMembership
troxos17InchesAlouminiou1	17InchesAlouminiou
troxos17InchesAlouminiouMembership1	TroxosMembership
troxos15InchesSiderou1	15InchesSiderou
troxos15InchesSiderouMembership1	TroxosMembership
troxos16InchesSiderou1	16InchesSiderou
troxos16InchesSiderouMembership1	TroxosMembership
troxos17InchesSiderou1	17InchesSiderou
troxos17InchesSiderouMembership1	TroxosMembership
aerotomi1	Aerotomi
aerotomiMembership1	AerotomiMembership
lyxniaAerosakou1	LyxniaAerosakou
lyxniaAerosakouMembership1	PCcontrollerMembership
lyxniaDiskofrenon1	LyxniaDiskofrenon
lyxniaDiskofrenonMembership1	PCcontrollerMembership
lyxniaMixanis1	LyxniaMixanis
lyxniaMixanisMembership1	PCcontrollerMembership

lyxniaMpatarias1	LyxniaMpatarias
lyxniaMpatariasMembership1	PCcontrollerMembership
lyxniaNtepozitou1	LyxniaNtepozitou
lyxniaNtepozitouMembership1	PCcontrollerMembership
lyxniaPortas1	LyxniaPortas
lyxniaPortasMembership1	PCcontrollerMembership
mpataria1	Mpataria
mpatariaMembership1	MpatariaMembership
ntepozito1	Ntepozito
ntepozitoMembership1	NtepozitoMembership
car1	Car

Πίνακας 4.6 : Individuals, Types

Individual	Object Property Assertion
aerosakosOdigou1	lyxniaBlabisGiaAerosakous lyxniaAerosakou1
aerosakosOdigouMembership1	aerosakosMember aerosakosOdigou1
aerosakosOrofis1	lyxniaBlabisGiaAerosakous lyxniaAerosakou1
aerosakosOrofisMembership1	aerosakosMember aerosakosOrofis1
aerosakosPleurikosAristera1	lyxniaBlabisGiaAerosakous lyxniaAerosakou1
aerosakosPleurikosAristeraMembership1	aerosakosMember aerosakosPleurikosAristera1
aerosakosPleurikosDeksia1	lyxniaBlabisGiaAerosakous lyxniaAerosakou1
aerosakosPleurikosDeksiaMembership1	aerosakosMember aerosakosPleurikosDeksia1
aerosakosSinodigou1	lyxniaBlabisGiaAerosakous lyxniaAerosakou1
aerosakosSinodigouMembership1	aerosakosMember aerosakosSinodigou1
anartisiMprosta1	----------
anartisiMprostaMembership1	anartisiMember anartisiMprosta1
anartisiPiso1	----------
anartisiPisoMembership1	anartisiMember anartisiPiso1
diaforikoMprosta1	----------
diaforikoMprostaMembership1	diaforikoMember diaforikoMprosta1

diaforikoPiso1	----------
diaforikoPisoMembership1	diaforikoMember diaforikoPiso1
diskofrenaMprosta1	lyxniaBlabisGiaDiskofrena lyxniaDiskofrena1
diskofrenaMprostaMembership1	diskofrenaMember diskofrenaMprosta1
diskofrenaPiso1	lyxniaBlabisGiaDiskofrena lyxniaDiskofrena1
diskofrenaPisoMembership1	diskofrenaMember diskofrenaPiso1
iliorofi1	----------
orofiMembershipHliorofi1	orofiMember iliorofi1
metallikiOrofi1	----------
orofiMembershipMetallikiOrofi1	orofiMember metallikiOrofi1
kubotioTaxititonAutomato1	----------
kubotioTaxititonAutomatoMembership1	kubotioTaxititonMember kubotioTaxititonAutomato1
kubotioTaxititonMixaniko1	----------
kubotioTaxititonMixanikoMembership1	kubotioTaxititonMember kubotioTaxititonMixaniko1
mixaniBenzini1	lyxniaBlabisGiaMixani lyxniaMixanis1
mixaniBenziniMembership1	mixaniMember mixaniBenzini1
mixaniDiesel1	lyxniaBlabisGiaMixani lyxniaMixanis1
mixaniDieselMembership1	mixaniMember mixaniDiesel1
mixaniHlektrokinitiras1	lyxniaBlabisGiaMixani lyxniaMixanis1
mixaniHlektrokinitirasMembership1	mixaniMember mixaniHlektrokinitiras1
porta3thiri1	lyxniaBlabisGiaPortes lyxniaPortas1
porta3Membership1	portaMember porta3thiri1
porta5thiri1	lyxniaBlabisGiaPortes lyxniaPortas1
porta5Membership1	portaMember porta5thiri1
sasiCoupe1	----------
sasiCoupeMembership1	sasiMember sasiCoupe1
sasiKanoniko1	----------
sasiKanonikoMembership1	sasiMember sasiKanoniko1
sasiSedan1	----------
sasiSedanMembership1	sasiMember sasiSedan1
timoni1	----------
timoniMembership1	timoniMember timoni1
troxos15InchesAlouminiou1	----------
troxos15InchesAlouminiouMembership1	troxosMember troxos15InchesAlouminiou1
troxos16InchesAlouminiou1	----------
troxos16InchesAlouminiouMembership1	troxosMember troxos16InchesAlouminiou1
troxos17InchesAlouminiou1	----------
troxos17InchesAlouminiouMembership1	troxosMember troxos17InchesAlouminiou1
troxos15InchesSiderou1	----------
troxos15InchesSiderouMembership1	troxosMember troxos15InchesSiderou1
troxos16InchesSiderou1	----------

troxos16InchesSiderouMembership1	troxosMember troxos16InchesSiderou1
troxos17InchesSiderou1	----------
troxos17InchesSiderouMembership1	troxosMember troxos17InchesSiderou1
aerotomi1	----------
aerotomiMembership1	aerotomiMember aerotomi1
lyxniaAerosakou1	----------
lyxniaAerosakouMembership1	pcControllerMember lyxniaAerosakou1
lyxniaDiskofrenon1	----------
lyxniaDiskofrenonMembership1	pcControllerMember lyxniaDiskofrenon1
lyxniaMixanis1	----------
lyxniaMixanisMembership1	pcControllerMember lyxniaMixanis1
lyxniaMpatarias1	----------
lyxniaMpatariasMembership1	pcControllerMember lyxniaMpatarias1
lyxniaNtepozitou1	----------
lyxniaNtepozitouMembership1	pcControllerMember lyxniaNtepozitou1
lyxniaPortas1	----------
lyxniaPortasMembership1	pcControllerMember lyxniaPortas1
mpataria1	lyxniaBlabisGiaMpataria lyxniaMpatarias1
mpatariaMembership1	mpatariaMember mpataria1
ntepozito1	lyxniaBlabisGiaNtepozito lyxniaNtepozitou1
ntepozitoMembership1	ntepozitoMember ntepozito1

Πίνακας 4.7 : Individuals, Object Property Assertions

Individual	Data Property Assertion
aerosakosOdigou1	----------
aerosakosOdigouMembership1	----------
aerosakosOrofis1	----------
aerosakosOrofisMembership1	----------
aerosakosPleurikosAristera1	----------
aerosakosPleurikosAristeraMembership1	----------
aerosakosPleurikosDeksia1	----------
aerosakosPleurikosDeksiaMembership1	----------
aerosakosSinodigou1	----------
aerosakosSinodigouMembership1	----------
anartisiMprosta1	typeAnartisis = “skliri”
anartisiMprostaMembership1	----------
anartisiPiso1	typeAnartisis = “skliri”
anartisiPisoMembership1	----------
diaforikoMprosta1	----------
diaforikoMprostaMembership1	----------
diaforikoPiso1	----------
diaforikoPisoMembership1	----------
diskofrenaMprosta1	eidosDiskofrenon = “takakia”
diskofrenaMprostaMembership1	----------
diskofrenaPiso1	eidosDiskofrenon = “takakia”

diskofrenaPisoMembership1	----------
iliorofi1	----------
orofiMembershipHliorofi1	----------
metallikiOrofi1	----------
orofiMembershipMetallikiOrofi1	----------
kubotioTaxititonAutomato1	sxeseisKubotiouTaxititon = 6
kubotioTaxititonAutomatoMembership1	----------
kubotioTaxititonMixaniko1	sxeseisKubotiouTaxititon = 6
kubotioTaxititonMixanikoMembership1	----------
mixaniBenzini1	kubikaMixanis = 1600 ippoiMixanis = 100
mixaniBenziniMembership1	----------
mixaniDiesel1	kubikaMixanis = 1600 ippoiMixanis = 100
mixaniDieselMembership1	----------
mixaniHlektrokinitiras1	kubikaMixanis = 1600 ippoiMixanis = 100
mixaniHlektrokinitirasMembership1	----------
porta3thiri1	----------
porta3Membership1	----------
porta5thiri1	----------
porta5Membership1	----------
sasiCoupe1	sasiCodeNo = “145eiuyr12”
sasiCoupeMembership1	----------
sasiKanoniko1	sasiCodeNo = “asdxc234ew”
sasiKanonikoMembership1	----------
sasiSedan1	sasiCodeNo = “890iuy56re”
sasiSedanMembership1	----------
timoni1	----------
timoniMembership1	locationTimoni = “aristera”
troxos15InchesAlouminiou1	----------
troxos15InchesAlouminiouMembership1	----------
troxos16InchesAlouminiou1	----------
troxos16InchesAlouminiouMembership1	----------
troxos17InchesAlouminiou1	----------
troxos17InchesAlouminiouMembership1	----------
troxos15InchesSiderou1	----------
troxos15InchesSiderouMembership1	----------
troxos16InchesSiderou1	----------
troxos16InchesSiderouMembership1	----------
troxos17InchesSiderou1	----------
troxos17InchesSiderouMembership1	----------
aerotomi1	----------
aerotomiMembership1	----------
lyxniaAerosakou1	----------
lyxniaAerosakouMembership1	----------
lyxniaDiskofrenon1	----------
lyxniaDiskofrenonMembership1	----------
lyxniaMixanis1	----------
lyxniaMixanisMembership1	----------
lyxniaMpatarias1	----------

lyxniaMpatariasMembership1	----------
lyxniaNtepozitou1	----------
lyxniaNtepozitouMembership1	----------
lyxniaPortas1	----------
lyxniaPortasMembership1	----------
mpataria1	----------
mpatariaMembership1	----------
ntepozito1	----------
ntepozitoMembership1	----------

Πίνακας 4.8 : Individuals, Data Property Assertions

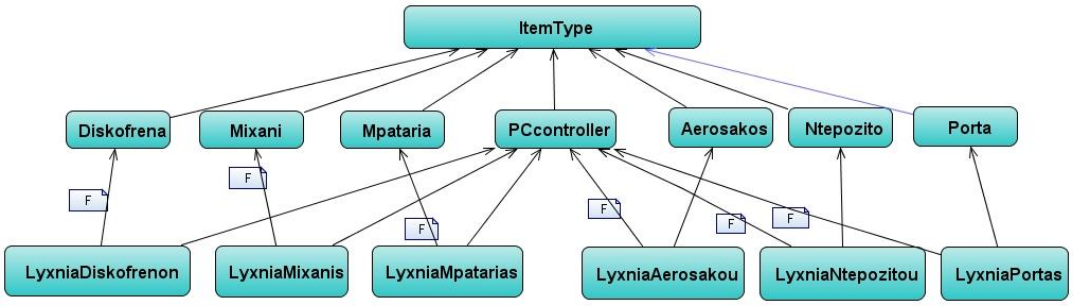
Για το Individual **car1** χρησιμοποιούμε διαφορετικό πίνακα, γιατί σε αυτό συνδέονται όλα τα άλλα Individuals που φτιάξαμε και με αυτό τον τρόπο γίνεται η σύνθεση του αυτοκινήτου. Ο πίνακας θα περιέχει τα εξαρτήματα που χρειάζονται για να φτιαχτεί ένα πλήρες αυτοκίνητο, δηλαδή θα υπάρχει μια επιλογή για οροφή και όχι δυο. Με άλλα λόγια, θα επιλέξει ο σχεδιαστής να βάλει στο αυτοκίνητο ηλιοροφή ή μεταλλική οροφή.

Individual	Object Property Assertion
car1	hasSasi sasiCoupeMembership1
	hasTroxos troxos17InchesAlouminiou Membership1
	hasDiskofrena diskofrenaPisoMembership1
	hasAnartisi anartisiMprostaMembership1
	hasAerosakosPleurikosDeksia aerosakosPleurikosDeksiaMembership1
	hasAerosakosSinodigou aerosakosSinodigouMembership1
	hasKubotioTaxititon kubotioTaxititonMixanikoMembership1
	hasDiaforikoMprosta diaforikoMprostaMembership1
	hasAerosakosOrofis aerosakosOrofisMembership1
	has3Portes porta3Membership1
	hasOrofi orofiMembershipHliorofi1
	hasDiaforikoPiso diaforikoPisoMembership1
	hasTimoni timoniMembership1
	hasAerosakosOdigou aerosakosOdigouMembership1
	hasAnartisi anartisiPisoMembership1
	hasDiskofrena diskofrenaMprostaMembership1
	hasMixani mixaniHlektrokinitirasMembership1
	hasAerosakosPleurikosAristera aerosakosPleurikosAristeraMembership1
	hasNtepozito ntepozitoMembership1
	hasMpataria mpatariaMembership1
	hasLyxniaDiskofrenon lyxniaDiskofrenonMembership1
	hasLyxniaPortas lyxniaPortasMembership1
	hasLyxniaMixanis lyxniaMixanisMembership1
	hasLyxniaAerosakou lyxniaAerosakouMembership1
	hasLyxniaMpatarias lyxniaMpatariasMembership1
	hasLyxniaNtepozitou lyxniaNtepozitouMembership1

Πίνακας 4.9 : Individual car1, Object Property Assertions

Στο σχήμα της οντολογίας και στα instances δεν φαίνονται αρκετά οι functional σχέσεις, όπως τουλάχιστον φαίνονται οι part-of. Γι' αυτό θεώρησαμε απαραίτητο να σας παραθέσουμε το παρακάτω σχήμα που δείχνει καθαρά ποιες είναι οι functional σχέσεις που αναφέρθηκαν παράπανω .

Οι σχέσεις στις οποίες υπάρχει το γράμμα F ως ετικέτα, είναι οι functional σχέσεις, οι υπόλοιπες είναι part-of .



Εικόνα 4.8 : Οι Functional Σχέσεις της Οντολογίας CAR

4.3 UML Σενάρια Οντολογίας

Στο σημείο αυτό θα γίνει η παρουσίαση τριών σεναρίων που σκοπό έχουν να δείξουν ξεκάθαρα τις part-of και functional σχέσεις της οντολογίας **CAR**.

4.3.1 Use Cases

- **Σενάριο 1^{ο}**

Use Case 1	Βλάβη στα φρένα.	
Goal In Context	Αν χαλάσει το σύστημα των φρένων,πρέπει να βρούμε από ποιο μέρος του συστήματος των φρένων προήλθε η βλάβη.	
Scope & Level	Υποσύστημα διαχείρισης Βλαβών,summary.	
Preconditions	Το αυτοκίνητο βρίσκεται σε λειτουργία.	
Success End Condition	Ανιχνεύεται το σημείο από το οποίο προήλθε η βλάβη.	
Failed End Condition	Δεν ανιχνεύεται το σημείο από το οποίο προήλθε η βλάβη.	
Primary, Secondary Actors	Χρήστης, Υποσύστημα διαχείρισης Βλαβών.	
Trigger	Ανίχνευση της αιτίας που προκάλεσαι την βλάβη στα φρένα.	
Description	**Step**	**Action**
	0	Ο χρήστης διαπιστώνει ότι η προειδοποιητική λυχνία βλάβης των φρένων είναι ενεργοποιημένη.
	1	Ο χρήστης αναζητά μέσω του συστήματος την αιτία που προκάλεσε την βλάβη.
	2	Το σύστημα εντοπίζει μέσω της functional σχέσης που υπάρχει μεταξύ της προειδοποιητικής λυχνίας και των δισκόφρενων, ότι η βλάβη προέρχεται από τα δισκόφρενα.
	3	Το σύστημα γνωστοποιεί στον χρήστη μέσω των part-of σχέσεων ποια εξαρτήματα των δισκόφρενων πιθανόν να προκάλεσαν την βλάβη.
	4	Ο χρήστης ελέγχει τα εξαρτήματα αυτά και εντοπίζει την βλάβη.
Extensions	**Step**	**Branching Action**
	2α	Δεν εντοπίζεται η functional σχέση.
		2α.1α Το σύστημα ειδοποιεί τον χρήστη ότι δεν μπορεί να του επιστρέψει απαντήσεις και επιστρέφει στο βήμα 1.
		2α.1β Ο χρήστης ακυρώνει την διαδικασία,η διαδικασία ολοκληρώνεται ανεπιτυχώς και το σύστημα επιστρέφει στην αρχική σελίδα.
	3α	Δεν εντοπίζονται οι part-of σχέσεις.
		3α.1α Το σύστημα ειδοποιεί τον χρήστη ότι δεν μπορεί να του επιστρέψει απαντήσεις και επιστρέφει στο βήμα 1.
		3α.1β Ο χρήστης ακυρώνει την διαδικασία,η διαδικασία ολοκληρώνεται ανεπιτυχώς και το σύστημα επιστρέφει στην αρχική σελίδα.
Sub-Variations	0α	

Πίνακας 4.10 : Υλοποίηση 1^{ου} Use Case

- **Σενάριο 2^{ο}**

Use Case 2	Υπολογισμός της τιμής ενός συστήματος-εξαρτήματος του αυτοκινήτου.	
Goal In Context	Ο χρήστης θέλει να υπολογίσει την τιμή ενός συστήματος του αυτοκινήτου,αθροίζοντας τις τιμές απ' όλα τα κομμάτια που σχετίζονται με αυτό.	
Scope & Level	Υποσύστημα διαχείρισης Συστημάτων-εξαρτημάτων,summary.	
Preconditions	Να έχουν τιμολογηθεί προηγουμένως όλα τα εξαρτήματα.	
Success End Condition	Η τιμή του συστήματος υπολογίζεται επιτυχώς.	
Failed End Condition	Η τιμή του συστήματος δεν υπολογίζεται επιτυχώς.	
Primary, Secondary Actors	Χρήστης,Υποσύστημα διαχείρισης Συστημάτων-εξαρτημάτων.	
Trigger	Υπολογισμός της τιμής ενός συστήματος-εξαρτήματος του αυτοκινήτου, αθροίζοντας τις τιμές απ' όλα τα κομμάτια που σχετίζονται με αυτό.	
Description	**Step**	**Action**
	0	Ο χρήστης επιλέγει ένα σύστημα του αυτοκινήτου.
	1	Ο χρήστης ζητά από το σύστημα να του δώσει την τιμή του συγκεκριμένου συστήματος.
	2	Το σύστημα βρίσκει όλα τα κομμάτια που σχετίζονται με part-of σχέσεις με το συγκεκριμένο σύστημα.
	3	Το σύστημα αθροίζει τις τιμές όλων των σχετιζόμενων με το σύστημα κομματιών.
	4	Το σύστημα δίνει στον χρήστη την τελική τιμή του συγκεκριμένου συστήματος του αυτοκινήτου.
Extensions	**Step**	**Branching Action**
	2α	Δεν εντοπίζονται οι part-of σχέσεις.
		2α.1α Το σύστημα ειδοποιεί τον χρήστη ότι δεν μπορεί να του επιστρέψει απαντήσεις και επιστρέφει στο βήμα 1.
		2α.1β Ο χρήστης ακυρώνει την διαδικασία,η διαδικασία ολοκληρώνεται ανεπιτυχώς και το σύστημα επιστρέφει στην αρχική σελίδα.
	3α	Παρουσιάζεται πρόβλημα στο συγκεκριμένο σημείο.
		3α.1α Το σύστημα ειδοποιεί τον χρήστη ότι δεν μπορεί να του επιστρέψει απαντήσεις και επιστρέφει στο βήμα 1.
		3α.1β Ο χρήστης ακυρώνει την διαδικασία,η διαδικασία ολοκληρώνεται ανεπιτυχώς και το σύστημα επιστρέφει στην αρχική σελίδα.
Sub-Variations	0α	

Πίνακας 4.11 : Υλοποίηση 2^{ου} Use Case

- **Σενάριο 3^{ο}**

Use Case 3	Προτάσεις για προσθήκη κάποιων αξεσουάρ αυτοκινήτου.	
Goal In Context	Το σύστημα βλέποντας τις επιλογές του χρήστη,του προτείνει τα αντίστοιχα αξεσουάρ αυτοκινήτου.	
Scope & Level	Υποσύστημα διαχείρισης Συστημάτων-εξαρτημάτων,summary.	
Preconditions	Ο χρήστης να πρέπει να έχει κάνει συγκεκριμένες επιλογές κατά την δημιουργία του αυτοκινήτου σύμφωνα με τις απαιτήσεις του .	
Success End Condition	Το σύστημα προτείνει στον χρήστη το αξεσουάρ που ταιριάζει με τις επιλογές του χρήστη.	
Failed End Condition	Το σύστημα αποτυγχάνει να προτείνει στον χρήστη κάποιο αξεσουάρ.	
Primary, Secondary Actors	Χρήστης,Υποσύστημα διαχείρισης Συστημάτων-εξαρτημάτων.	
Trigger	Προτάσεις από το σύστημα για προσθήκη κάποιων αξεσουάρ,βάσει των επιλογών του χρήστη κατά την σύνθεση του αυτοκινήτου.	
Description	**Step**	**Action**
	0	Ο χρήστης ξεκινά την σύνθεση του αυτοκινήτου, επιλέγοντας τα διάφορα συστήματα και τα κομμάτια των εξαρτημάτων που τα αποτελούν.
	1	Το σύστημα καταγράφει τις επιλογές του χρήστη.
	2	Ο χρήστης ολοκληρώνει την σύνθεση του αυτοκινήτου.
	3	Το σύστημα προτείνει στον χρήστη κάποια αξεσουάρ για το αυτοκίνητο,σύμφωνα με τις παραπάνω επιλογές του χρήστη.
	4	Ο χρήστης αποφασίζει να δεχτεί την πρόταση του συστήματος.
	5	Το σύστημα σύμφωνα με την απόφαση του χρήστη, συμπεριλαμβάνει τα αξεσουάρ στην τελική σύνθεση του αυτοκινήτου.
Extensions	**Step**	**Branching Action**
	3α	Δεν υπάρχει κάποια πρόταση που να προκύπτει βάσει των επιλογών του χρήστη.
		3α.1 Το σύστημα ενημερώνει τον χρήστη ότι δεν έχει κάποιο αξεσουάρ να του προτείνει και ολοκληρώνει την διαδικασία.
	4α	Η πρόταση δεν γίνεται δεκτή από τον χρήστη.
		4α.1 Ο χρήστης ολοκληρώνει την διαδικασία σύνθεσης του αυτοκινήτου.
	5α	Κάποιο σφάλμα παρουσιάστηκε.
		5α.1α Το σύστημα ενημερώνει τον χρήστη ότι δεν μπορεί να του εμφανίσει την τελική σύνθεση του αυτοκινήτου και τον στέλνει στο βήμα 0.
		5α.1β Ο χρήστης ακυρώνει την διαδικασία,η διαδικασία ολοκληρώνεται ανεπιτυχώς και το σύστημα επιστρέφει στην αρχική σελίδα.
Sub-Variations	0α	

Πίνακας 4.12 : Υλοποίηση 3^{ου} Use Case

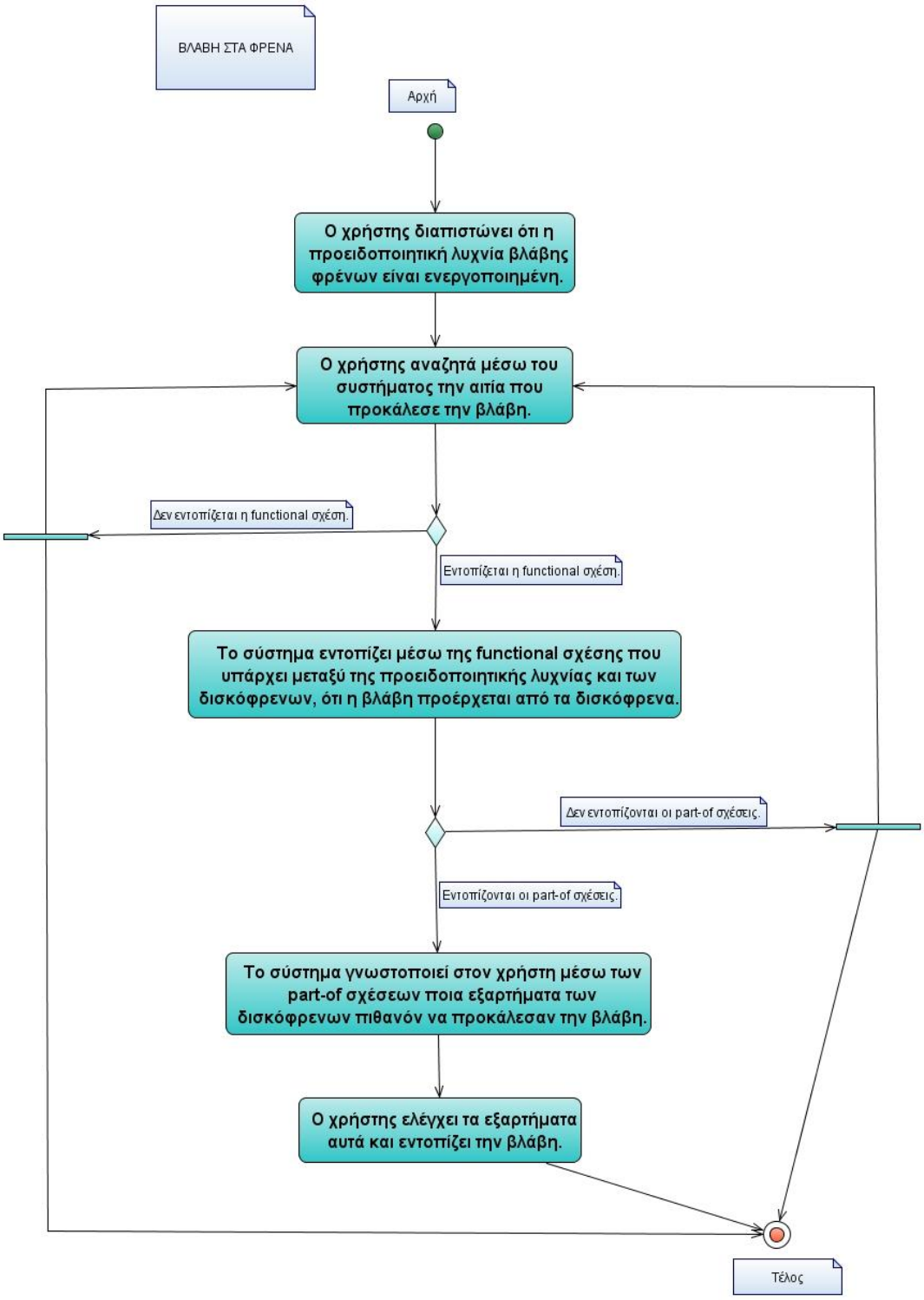
Παρατηρήσεις:

- Στο πρώτο σενάριο περιγράφεται η βλάβη στα φρένα, αν δηλαδή χαλάσει το σύστημα το φρένων , θα πρέπει να βρούμε από ποιο μέρος του συστήματος των φρένων προήλθε η βλάβη. Επομένως, γίνεται φανερό με αυτό τον τρόπο ότι για να πάμε από την λυχνία που είναι προειδοποιητική για τα φρένα μέχρι τα φρένα, κάνουμε χρήση της συγκεκριμένης functional σχέσης που έχει δημιουργηθεί στην οντολογία. Κατόπιν, για να βρούμε ποιο συγκεκριμένο εξάρτημα των φρένων έχει χαλάσει, κάνουμε χρήση των part-of σχέσεων μεταξύ των φρένων και των εξαρτημάτων από τα οποία αποτελούνται. Το ίδιο σενάριο μπορεί να γίνει για τις λυχνίες της πόρτας , του ντεπόζιτου , της μπαταρίας, της μηχανής , των αερόσακων που συνδέονται με functional σχέσεις με την πόρτα, το ντεπόζιτο, την μπαταρία, την μηχανή και τους αερόσακους του αυτοκινήτου αντίστοιχα .
- Στο δεύτερο σενάριο περιγράφεται ο υπολογισμός της τιμής ενός συστήματος του αυτοκινήτου, δηλαδή ο χρήστης θέλει να υπολογίσει την τιμή ενός συστήματος του αυτοκινήτου, αθροίζοντας τις τιμές απ' όλα τα κομμάτια που το απαρτίζουν . Επομένως, γίνεται φανερό με αυτό τον τρόπο ότι για να υπολογίσουμε την τιμή ενός συστήματος , πρέπει πρώτα κάνοντας χρήση των part-of σχέσεων , να βρούμε τα εξαρτήματα από τα οποία αποτελείται το συγκεκριμένο σύστημα και στην συνέχεια ανάλογα με τις τιμές του κάθε εξαρτήματος ,αλλά και τον αριθμό τους για την σωστή σύνδεση του συστήματος(ποσότητα πχ: 5 βίδες) , να υπολογίσουμε την τελική τιμή του συγκεκριμένου συστήματος του αυτοκινήτου . Το ίδιο σενάριο μπορεί να γίνει για όλα τα συστήματα του αυτοκινήτου, παραδείγματος χάριν: για την πόρτα, την μηχανή κτλ.
- Στο τρίτο σενάριο περιγράφεται η πρόταση του συστήματος για προσθήκη κάποιων αξεσουάρ αυτοκινήτου, δηλαδή το σύστημα βλέποντας τις επιλογές του χρήστη, του προτείνει να προσθέσει στην σύνθεση του αυτοκινήτου του κάποιο αξεσουάρ . Το συγκεκριμένο σενάριο είναι γενικό και για να γίνει περισσότερο κατανοητό θα προχωρήσουμε στην παράθεση ενός παραδείγματος :
  - ❖ Έστω ότι κατά την σύνθεση του αυτοκινήτου του, ο χρήστης έχει επιλέξει :
    1. Για οροφή , την **ηλιοροφή** .
    2. Για αριθμό θυρών του αμαξιού, το **τρίθυρο** μοντέλο .
    3. Για σασί , το **coupe** .
    4. Για διάμετρο ζάντας, τις **17Inches** .

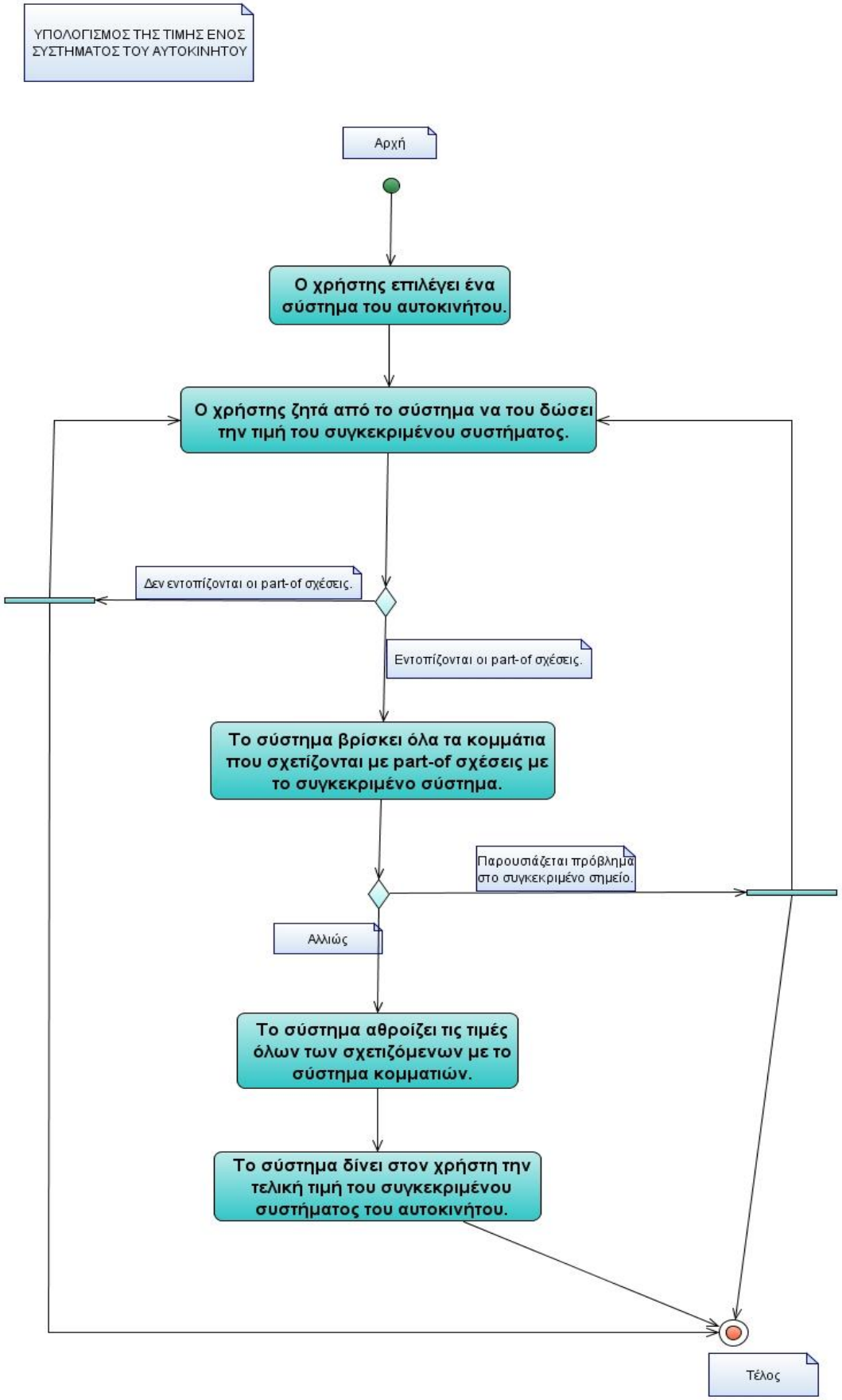
  Τότε το σύστημα «βλέποντας» τις παραπάνω επιλογές, προτείνει στον χρήστη την προσθήκη της **αεροτομής**, που είναι ένα ,επιπρόσθετο και όχι αναγκαίο, αξεσουάρ του αυτοκινήτου.

4.3.2 Activity Diagrams

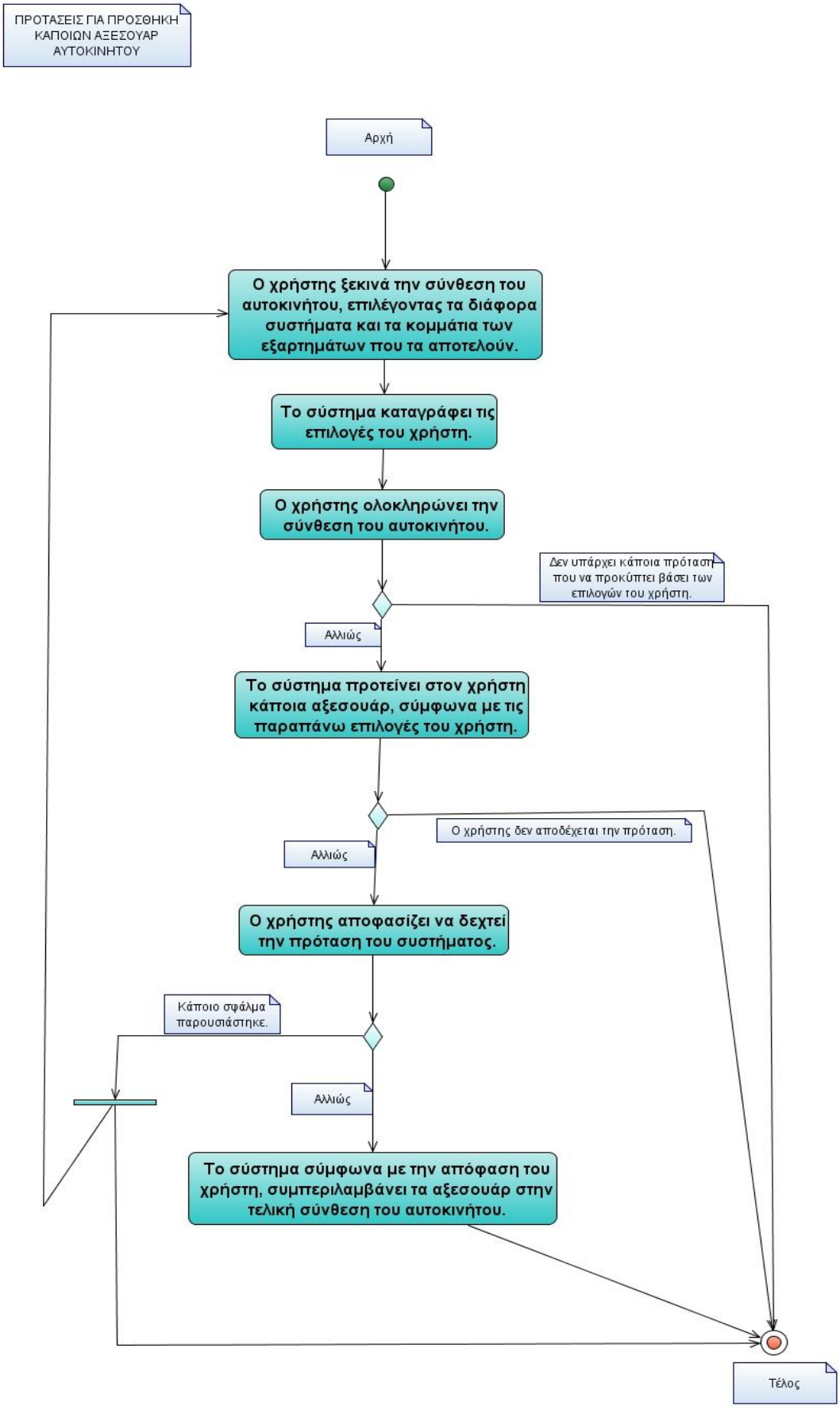
Ακολουθούν τα activity diagrams των προαναφερθέντων use cases.



Εικόνα 4.9 : Υλοποίηση 1^{ου} Activity Diagram



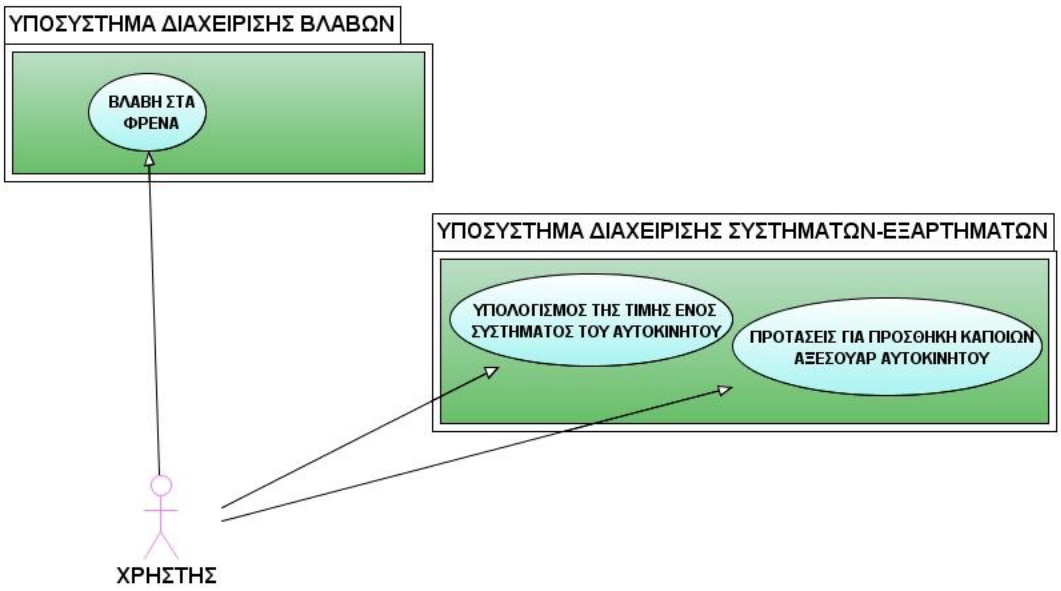
Εικόνα 4.10 : Υλοποίηση 2^{ου} Activity Diagram



Εικόνα 4.11 : Υλοποίηση 3^{ου} Activity Diagram

4.3.3 Use Case Diagram

Ακολουθεί ένα use case diagram που ικανοποιεί τα παραπάνω UML σενάρια .



Εικόνα 4.12 : Υλοποίηση του Use Case Diagram

5 Υλοποίηση Εφαρμογής

5.1 Πλατφόρμα Υλοποίησης της Εφαρμογής

Η εφαρμογή υλοποιήθηκε με την χρήση του εργαλείου NetBeans IDE 6.1 . Για την σύνδεση της οντολογίας (δηλαδή του owl αρχείου που είχε δημιουργηθεί στο Protégé) και της java , χρησιμοποιήθηκε το jena 2.5.7 και για το σχεδιασμό του συστήματος χρησιμοποιήθηκε το εργαλείο της java, Swing .

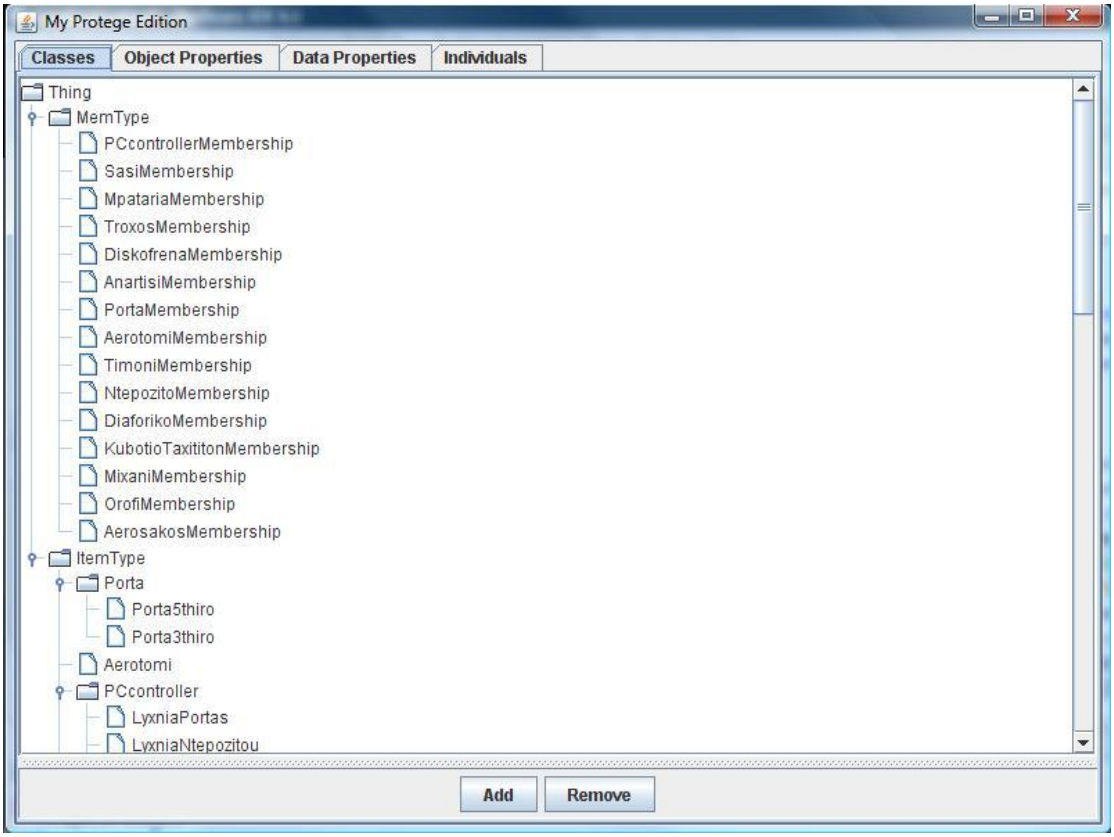
5.2 Λειτουργικότητα Εφαρμογής

Η εφαρμογή υλοποιεί μια desktop εφαρμογή που την χρησιμοποιεί ο σχεδιαστής του συστήματος για να διαχειριστεί την οντολογία του. Με άλλα λόγια, αυτή η εφαρμογή αντικαθιστά την λειτουργικότητα που παρέχει στον σχεδιαστή το εργαλείο σχεδιασμού οντολογιών Protégé . Με τα ακόλουθα screen dumps και τον σχολιασμό που τα συνοδεύει γίνεται αντιληπτή η ακριβής λειτουργικότητα του συστήματος.

5.3 Διεπικοινωνία Χρήστη – Συστήματος

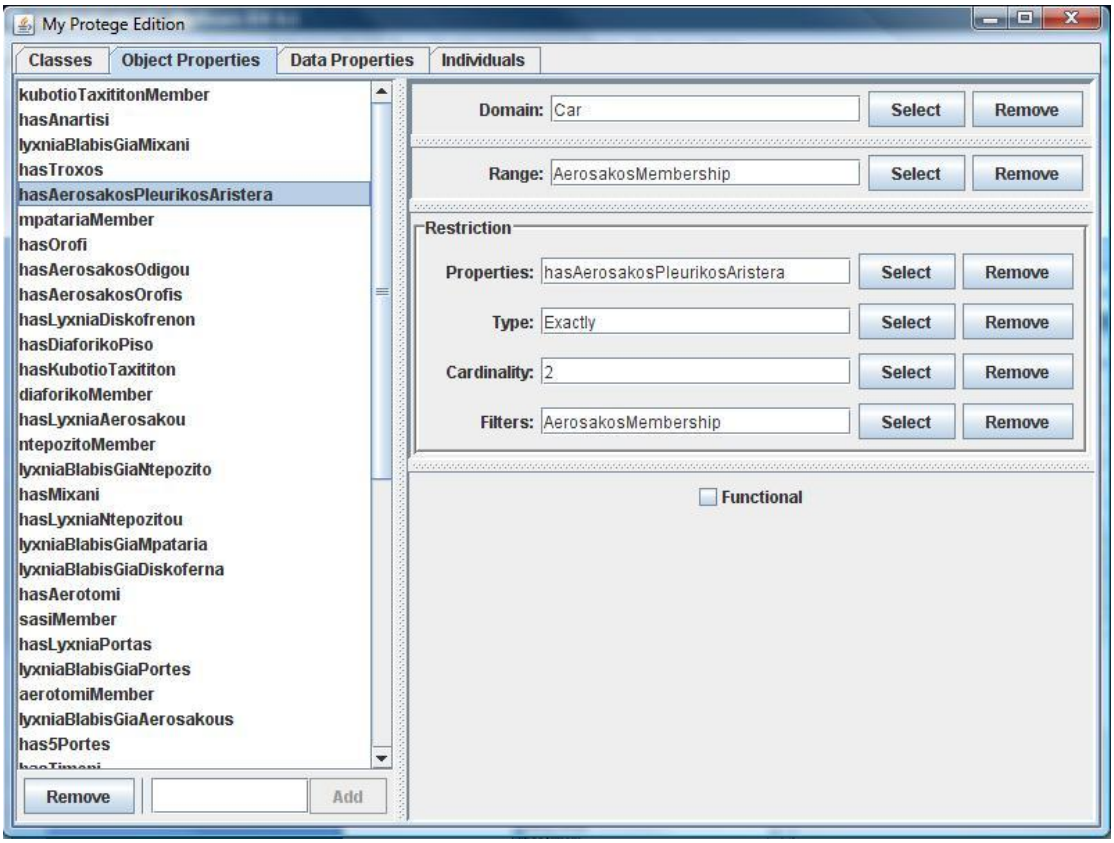
5.3.1 Περιγραφή – Παρουσίαση των Screen – Dumps

Στο ακόλουθο screen dump φαίνονται τα **classes** που υπάρχουν στο owl αρχείο που χρησιμοποιήσαμε ως είσοδο στην εφαρμογή . Δηλαδή, έχουμε δημιουργήσει ένα JTree που αναπαριστά το δέντρο των κλάσεων της οντολογίας . Επιπλέον δίνεται η δυνατότητα στον σχεδιαστή να επέμβει στο αρχείο της οντολογίας , καθώς με την χρήση των κουμπιών add και remove μπορεί να προσθέσει ή να απομακρύνει κάποια κλάση από το δέντρο αντίστοιχα .



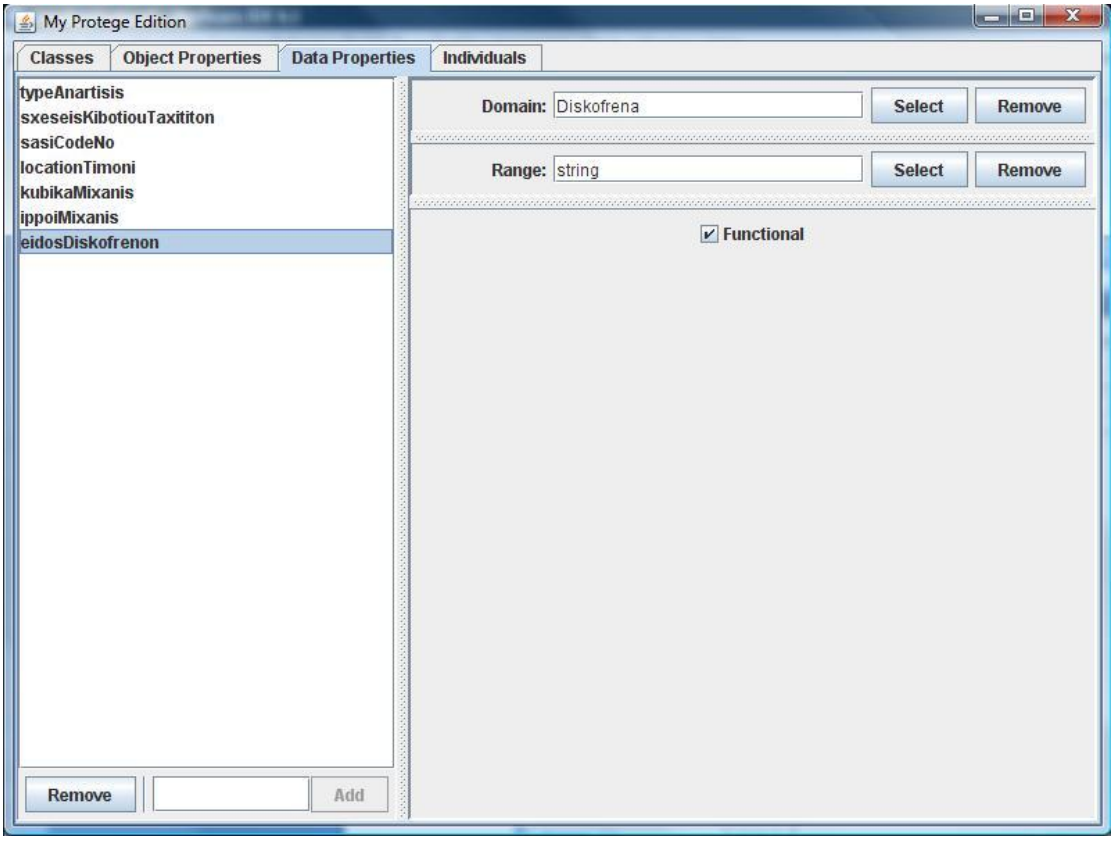
Εικόνα 5.1 : Η καρτέλα Διαχείρισης των Classes

Στο ακόλουθο screen dump φαίνονται τα **object properties** που υπάρχουν στο owl αρχείο που χρησιμοποιήσαμε ως είσοδο στην εφαρμογή. Δηλαδή έχουμε δημιουργήσει JList και JScrollPanel με σκοπό να αναπαριστήσουμε τα object properties (με τα αντίστοιχα domains,ranges,restrictions και την functionality των) της οντολογίας . Επιπλέον δίνεται η δυνατότητα στον σχεδιαστή να επέμβει στο αρχείο της οντολογίας , καθώς με την χρήση των κουμπιών select και remove μπορεί να προσθέσει ή να απομακρύνει κάποιο domain,range ή restriction αντίστοιχα . Ακόμη, έχει την δυνατότητα ο σχεδιαστής με τα κουμπιά add και remove να προσθέσει ή να αφαιρέσει κάποιο object property και με αντίστοιχα selects στα υπόλοιπα πεδία να ορίσει πλήρως το αντίστοιχο object property και τέλος επιλέγοντας το κουτάκι functional ,να δηλώσει το συγκεκριμένο object property ως functional .



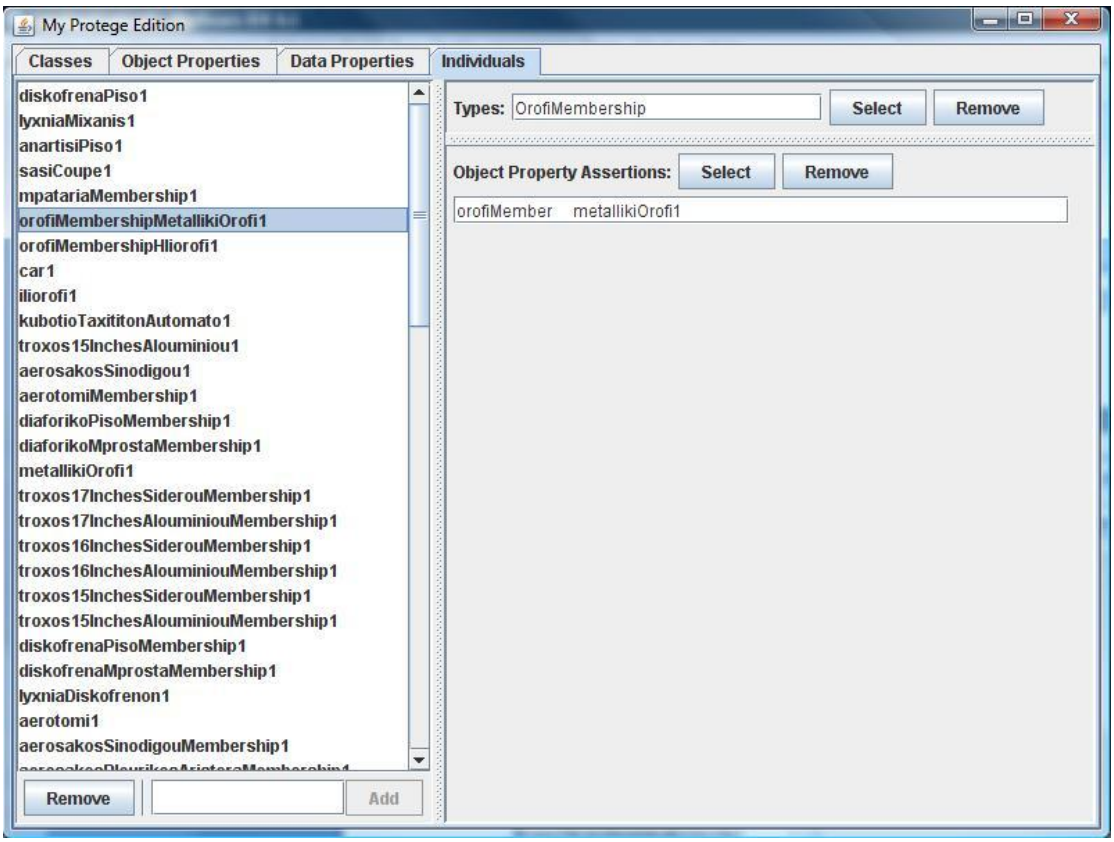
Εικόνα 5.2 : Η καρτέλα Διαχείρισης των Object Properties

Στο ακόλουθο screen dump φαίνονται τα **data properties** που υπάρχουν στο owl αρχείο που χρησιμοποιήσαμε ως είσοδο στην εφαρμογή . Δηλαδή έχουμε δημιουργήσει JList και JScrollPanel με σκοπό να αναπαριστήσουμε τα data properties (με τα αντίστοιχα domains,ranges και την functionality των) της οντολογίας . Επιπλέον δίνεται η δυνατότητα στον σχεδιαστή να επέμβει στο αρχείο της οντολογίας , καθώς με την χρήση των κουμπιών select και remove μπορεί να προσθέσει ή να απομακρύνει κάποιο domain,range αντίστοιχα. Ακόμη, έχει την δυνατότητα ο σχεδιαστής με τα κουμπιά add και remove να προσθέσει ή να αφαιρέσει κάποιο data property και με αντίστοιχα selects στα υπόλοιπα πεδία να ορίσει πλήρως το αντίστοιχο data property και τέλος επιλέγοντας το κουτάκι functional ,να δηλώσει το συγκεκριμένο data property ως functional .



Εικόνα 5.3 : Η καρτέλα Διαχείρισης των Data Properties

Στο ακόλουθο screen dump φαίνονται τα **individuals** που υπάρχουν στο owl αρχείο που χρησιμοποιήσαμε ως είσοδο στην εφαρμογή . Δηλαδή έχουμε δημιουργήσει JList και JScrollPanel με σκοπό να αναπαριστήσουμε τα individuals (με τα αντίστοιχα types,object property assertions) της οντολογίας . Επιπλέον δίνεται η δυνατότητα στον σχεδιαστή να επέμβει στο αρχείο της οντολογίας , καθώς με την χρήση των κουμπιών select και remove μπορεί να προσθέσει ή να απομακρύνει κάποιο type,object property assertion αντίστοιχα . Ακόμη, έχει την δυνατότητα ο σχεδιαστής με τα κουμπιά add και remove να προσθέσει ή να αφαιρέσει κάποιο individual και με αντίστοιχα selects στα υπόλοιπα πεδία να ορίσει πλήρως το αντίστοιχο individual .



Εικόνα 5.4 : Η καρτέλα Διαχείρισης των Individuals

Λίστα Εικόνων

Λίστα Πινάκων

Αναφορές

1. Έμπειρα Συστήματα και Εφαρμογές στην Ρομποτική. Scholl of Computer and Electrical Engineering.http://courses.dbnet.ntua.gr/el/empeira_systimata_kai_efarmoges_sth_rompotiki/dialejeis.html
2. Σημασιολογικός Ιστός, Βασιλειάδης Νικόλαος. http://lpis.csd.auth.gr/mtpx/sw/sw-info-fr.htm#teaching
3. DSMC – Εργαστήριο Ψηφιακών Συστημάτων και Επεξεργασίας Ψηφιακών Μέσων, Ευφυής Πρόσβαση στην Ψηφιακή Πληροφορία. http://dsmc.eap.gr/semweb.php
4. Αναπαράσταση Γνώσης στον Παγκόσμιο Ιστό – XML Schema, Ιωάννης Χατζηλυγερούδης. http://mmlab.ceid.upatras.gr/aigroup/undergrad/krweb/docs/XML-Schema.pdf
5. Επιχειρηματικές Εφαρμογές των Τεχνολογιών Internet. Information Management Unit/ICCS of NTUA. http://www.imu.iccs.gr/courses/MIS/MIS-lecture3.ppt
6. Web Services Theory. http://www.it.uom.gr/project/soap/Theory/xml.html
7. Web Ontology Language : OWL. A Semantic Web Primer. Grigoris Antoniou – Frank van Harmelen. http://www.ics.forth.gr/isl/swprimer/presentations/Chapter4.ppt
8. Web Ontology Language : OWL. DERN – Digital Enterprise Research Network. Σπυρος Ντιούδης. http://www.imu.iccs.gr/projects/dern/files/OWL-Ntioudis.pdf
9. Wikipedia – Swing. http://en.wikipedia.org/wiki/Swing_(Java)
10. Γραφικές Διεπαφές Χρήστη – Swing. http://mmlab.ceid.upatras.gr/aigroup/undergrad/java/docs/jav12a-GUI2_SWING_EVENTS.pdf
11. Sun Developer Network. A Swing Architecture Overview. http://www.j2ee.me/products/jfc/tsc/articles/architecture/index.html
12. Εισαγωγή στη UML. Γλώσσες Προδιαγραφής. http://courses.cn.ntua.gr/mod/resource/view.php?id=359
13. Oracle Berkeley DB Java Edition – Getting Started with Berkeley DB Java Edition, Release 3.3 .
14. XML Summer School 2005. A Practical Introduction to Ontologies and OWL. Duncan Hull and Nick Drummond.
15. Αποθήκες Δεδομένων και Εξόρυξης Γνώσης. Κώστας Καρπούζης.
16. Jena Framework. Κωνσταντίνος Κανάρης.
17. The Jena RDF Framework. Κωνσταντίνος Τζώνας.
18. Οντολογίες και Σημασιολογικός Ιστός. Πανεπιστήμιο Αιγαίου. http://www.icsd.aegean.gr/kotis/OE&SW'07
19. Introduction to OWL. Πανεπιστήμιο Αιγαίου. http://www.icsd.aegean.gr/kotis/OE&SW'07

20. A practical Guide to Building OWL Ontologies Using Protégé 4 and CO – ODE Tools Edition 1.1 . Matthew Horridge, Simon Jupp, Georgina Moulton, Alan Rector, Robert Stevens, Chris Wroe. The University of Manchester.

21. RDFS and Quering. Πανεπιστήμιο Αιγαίου.
http://www.icsd.aegean.gr/kotis/OE&SW'07

22. Μοντελοποίηση Περιπτώσεων Χρήσης (Use Case Modeling).
http://www.rational.com/uml
http://www.sdmagazine.com/uml

23. Σημασιολογικό Διαδίκτυο. Τεχνητή Νοημοσύνη – Β' Έκδοση. Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου.

24. Ανάλυση Συστημάτων. Εισαγωγή στη UML.