



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

**ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ
ΥΠΟΛΟΓΙΣΤΩΝ**

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ

**Πρότυπο Σύστημα Αποθήκευσης και Διαχείρισης
Σχημάτων RDFS**

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΖΩΗΣ ΚΑΟΥΔΗ

Επιβλέπων : Τιμολέων Σελλής
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2004



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ

Πρότυπο Σύστημα Αποθήκευσης και Διαχείρισης Σχημάτων RDFS

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΖΩΗΣ ΚΑΟΥΔΗ

Επιβλέπων : Τιμολέων Σελλής
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 13^η Ιουλίου 2004.

.....
Τιμολέων Σελλής
Καθηγητής Ε.Μ.Π.

.....
Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

.....
Νεκτάριος Κοζύρης
Επίκουρος Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2004

(Υπογραφή)

.....

ΖΩΗ ΚΑΟΥΔΗ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

© 2004 – All rights reserved

Ευχαριστίες

Η διπλωματική αυτή εκπονήθηκε στο Εργαστήριο Συστημάτων Βάσεων Γνώσεων και Δεδομένων του Εθνικού Μετσόβιου Πολυτεχνείου. Ήταν μια πολύ καλή και ενδιαφέρουσα επαφή με το χώρο των συστημάτων διαχείρισης δεδομένων και με τον προγραμματισμό. Στο σημείο αυτό, θα ήθελα να ευχαριστήσω τον καθηγητή κ. Τιμολέοντα Σελλή για την επίβλεψη της διπλωματικής και τη δυνατότητα που μου έδωσε να εκπονήσω τη διπλωματική μου εργασία στο Εργαστήριο Συστημάτων Βάσεων Γνώσεων και Δεδομένων. Επίσης, θα ήθελα να ευχαριστήσω ιδιαίτερα το Δρ. Θοδωρή Δαλαμάγκα για τη στήριξη και το χρόνο που διέθεσε για την επίλυση προβλημάτων που δημιουργήθηκαν κατά τη διάρκεια εκπόνησης της διπλωματικής. Η βοήθεια του ήταν καταλυτική για το πέρας της εργασίας και η συνεργασία μας υπήρξε άψογη.

Στους γονείς μου, Κώστα και Άννα

Περίληψη

Ο Σημασιολογικός Ιστός είναι το επόμενο βήμα του Παγκόσμιου Ιστού, όπου η πληροφορία αποκτά δομή και σημασιολογία ώστε να υποστηριχθεί η αποδοτική αναζήτηση, επεξεργασία και ενοποίηση δεδομένων. Το πλαίσιο RDF θεωρείται σημαντική συνεισφορά για την καθιέρωση του Σημασιολογικού Ιστού. Πρόκειται για ένα μοντέλο περιγραφής και επεξεργασίας μεταδεδομένων, δηλαδή δεδομένων για τα δεδομένα. Σε ένα κείμενο RDF, ορίζονται δηλώσεις για πόρους του Ιστού, δηλαδή ιστοσελίδες, συγγραφείς, προγράμματα κλπ. Το RDF σχήμα είναι ένας μηχανισμός περιγραφής σχετικών πόρων και των σχέσεων μεταξύ τους, και αποτελεί σημασιολογική επέκταση του πλαισίου RDF. Το RDF σχήμα βασίζεται σε κλάσεις και ιδιότητες και θυμίζει σύστημα τύπων για αντικειμενοστρεφείς γλώσσες προγραμματισμού.

Ο στόχος της διπλωματικής είναι η ανάπτυξη ενός πρότυπου συστήματος διαχείρισης RDF σχημάτων. Το σύστημα αυτό παρέχει γλώσσα ερωτήσεων με τέσσερις βασικούς τελεστές: *επιλογή*, *ένωση*, *τομή*, *διαφορά*. Ο τελεστής *επιλογή* εφαρμόζεται σε ένα RDF σχήμα και προβάλλει κατά κάποιο τρόπο ένα μέρος του σχήματος αυτού. Ο τελεστής *ένωση* εφαρμόζεται σε δύο RDF σχήματα και τα συγχωνεύει σε ένα. Ο τελεστής *τομή* εφαρμόζεται επίσης σε δύο RDF σχήματα και βρίσκει το κοινό τους κομμάτι. Τέλος, ο τελεστής *διαφορά* εφαρμόζεται σε δύο RDF σχήματα και βρίσκει μέρος του πρώτου σχήματος που δεν υπάρχει στο δεύτερο. Η γλώσσα ερωτήσεων του συστήματος υποστηρίζεται με γραφικό περιβάλλον ερώτησης-με-χρήση-παραδείγματος (QBE).

Λέξεις Κλειδιά:

Σημασιολογικός Ιστός, RDF/S, γράφοι RDF, RQL, επιλογή, ένωση, τομή, διαφορά.

Abstract

The Semantic Web is the next step of the current Web, in which information is given well-defined meaning to support effective data discovery, automation and integration. The RDF framework is a key issue for the Semantic Web. It is a foundation for processing metadata, that is data for the meaning of data. In an RDF document, one can make statements about particular Web resources, that is Web pages, page authors, scripts, etc. The RDF Schema provides mechanisms for describing groups of related resources and the relationships between these resources, acting as a semantic extension of RDF. The RDF Schema description language is based on classes and properties, and is similar to the type systems of object-oriented programming languages.

This Diploma Thesis describes a prototype system for RDF schema management. The system provides a query language with four operators: *selection*, *union*, *intersection*, *difference*. The *selection* operator is applied on a RDF schema and extracts a part of it. The *union* operator is applied on two RDF schemas and merges them. The *intersection* operator is also applied on two RDF schemas and extracts their common part. Finally, the *difference* operator is applied on two RDF schemas, too, and extracts the part of the first schema which is not present in the second. The user can pose queries using a graphical interface that supports the Query-by-Example (QBE) paradigm.

Keywords:

Semantic Web, RDF/S, RDF graphs, RQL, selection, union, intersection, difference.

Πίνακας περιεχομένων

1	Εισαγωγή.....	17
1.1	Αντικείμενο της διπλωματικής	18
1.2	Οργάνωση του τόμου.....	20
2	Περιγραφή θέματος.....	21
2.1	Θεωρητικό υπόβαθρο.....	21
2.1.1	RDF	21
2.1.2	RDF Schema.....	26
2.1.3	Η γλώσσα RQL (RDF Query Language).....	31
2.2	Σχετικές εργασίες.....	37
2.2.1	Το σύστημα RONDO	37
2.2.2	Διαχείριση ιεραρχικών σχημάτων στο Σημασιολογικό Ιστό.....	38
2.3	Στόχος	41
3	Θεωρητική μελέτη	43
3.1	RDF σχήματα και γράφοι.....	43
3.1.1	Ορολογία και Ορισμός RDFS.....	43
3.1.2	Ορισμός ευρύτερου πεδίου ορισμού και τιμών	45
3.1.3	Ορισμός υποσυνόλου ενός RDFS.....	46
3.1.4	Καθολικό σχήμα	47
3.2	Τελεστές.....	48
3.2.1	Τελεστής επιλογής.....	48
3.2.2	Τελεστής ένωσης	50
3.2.3	Τελεστής τομής.....	52
3.2.4	Τελεστής διαφοράς	52
4	Ανάλυση και σχεδίαση	55
4.1	Ανάλυση – περιγραφή αρχιτεκτονικής	55
4.1.1	Διαχωρισμός υποσυστημάτων	55
4.1.2	Περιγραφή υποσυστημάτων	57
4.1.2.1	Υποσύστημα διαχείρισης RQL ερωτήσεων.....	57

4.1.2.2	Υποσύστημα αποθήκευσης επιλεγμένων κλάσεων	58
4.1.2.3	Υποσύστημα εφαρμογής των τελεστών των πράξεων.....	58
4.1.2.4	Υποσύστημα δημιουργίας RDF σχήματος.....	59
4.1.2.5	Υποσύστημα αποθήκευσης RDF σχήματος.....	59
4.1.2.6	Υποσύστημα απεικόνισης RDF σχήματος.....	60
4.1.2.7	Υποσύστημα διαπροσωπείας χρήστη.....	61
4.2	Σχεδίαση του συστήματος.....	61
4.2.1	Εφαρμογές	61
4.2.1.1	Εφαρμογή διαχείρισης βάσης	63
4.2.1.2	Εφαρμογή αλλαγής καθολικού σχήματος.....	63
4.2.1.3	Εφαρμογή τελεστών.....	64
4.2.1.4	Εφαρμογή αλγορίθμου επιλογής.....	64
4.2.1.5	Εφαρμογή απεικόνισης RDF σχημάτων	65
5	Υλοποίηση.....	67
5.1	Πλατφόρμες και προγραμματιστικά εργαλεία	67
5.1.1	Γενικά	67
5.1.2	Εγκατάσταση της RQL και του RSSDB.....	68
5.1.2.1	RQL.....	68
5.1.2.2	RSSDB.....	69
5.1.3	Εγκατάσταση του Graphviz και του RDFSViz	69
5.1.3.1	Graphviz.....	69
5.1.3.2	RDFSViz.....	70
5.2	Λεπτομέρειες υλοποίησης.....	70
5.2.1	Αλγόριθμοι	70
5.2.1.1	Αλγόριθμος επιλογής	70
5.2.1.2	Αλγόριθμος ένωσης	76
5.2.1.3	Αλγόριθμος τομής.....	77
5.2.1.4	Αλγόριθμος διαφοράς	78
5.2.1.5	Αλγόριθμος για την παρουσίαση της ιεραρχίας των κλάσεων	79
5.2.1.6	Αλγόριθμος για την παρουσίαση των ιδιοτήτων.....	80
5.2.1.7	Αλγόριθμος για την παρουσίαση των κυριολεκτικών	82

5.2.2	Περιγραφή κλάσεων	83
5.2.2.1	public class ui	83
5.2.2.2	public class Operation	85
5.2.2.3	public class RQLQuery	86
5.2.2.4	public class db	86
5.2.2.5	public class ReadRDFFile	87
5.2.2.6	public class CreateRdfsFile	87
5.2.2.7	public class StoreToDB	88
5.2.2.8	public class Begin	88
6	Έλεγχος.....	91
6.1	Μεθοδολογία ελέγχου	91
6.2	Αναλυτική παρουσίαση ελέγχου	91
6.2.1	Νέα ερώτηση	92
6.2.2	Αλλαγή καθολικού σχήματος	99
6.2.3	Αλλαγή βάσης.....	100
6.2.4	Απεικόνιση αποθηκευμένων σχημάτων σε γράφο	100
6.2.5	Απεικόνιση αποθηκευμένων σχημάτων σε μορφή αρχείου.....	102
7	Επίλογος	105
7.1	Σύνοψη και συμπεράσματα.....	105
7.2	Μελλοντικές επεκτάσεις	106
8	Βιβλιογραφία	107

1

Εισαγωγή

Το Διαδίκτυο αποτελεί σήμερα τη μεγαλύτερη πηγή πληροφορίας. Μεγάλοι όγκοι δεδομένων αναζητούνται, ανταλλάσσονται και επεξεργάζονται μέσω του Παγκόσμιου Ιστού. Επειδή, όμως, ο όγκος των δεδομένων του Ιστού έχει πάρει μεγάλες διαστάσεις χωρίς να υπάρχει ενιαίος τρόπος οργάνωσης, η ανταλλαγή και η επεξεργασία τους είναι πολύ δύσκολη. Ο Σημασιολογικός Ιστός έρχεται ακριβώς να εξυπηρετήσει την ανάγκη για ενιαία οργάνωση των δεδομένων, ώστε ο Ιστός να γίνει μια παγκόσμια πλατφόρμα ανταλλαγής και επεξεργασίας δεδομένων από ετερογενείς πηγές πληροφορίας. Ο Σημασιολογικός Ιστός δίνει δομή, οργάνωση και σημασιολογία στα δεδομένα, ώστε να είναι κατανοητά σε επίπεδο μηχανής.

Οι δύο πιο γνωστές τεχνολογίες που έχουν αναπτυχθεί μέχρι στιγμής και υποστηρίζουν τη λειτουργία του Σημασιολογικού Ιστού είναι η γλώσσα XML και το πλαίσιο RDF. Χρησιμοποιώντας τη γλώσσα XML, οι σελίδες και τα δεδομένα του Ιστού σημαδεύονται με ετικέτες. Οι ετικέτες βοηθούν τα προγράμματα Ιστού να ανακαλύψουν τα δεδομένα ευκολότερα, δίνοντας σημασιολογία και δομή στα δεδομένα. Για παράδειγμα, πληροφορία σχετική με τον συγγραφέα αυτής της διπλωματικής κωδικοποιείται σε ένα αρχείο XML με τις ετικέτες `undergraduate`, `firstname`, `lastname`, `age`, `email`, ως εξής:

```
<undergraduate>
  <firstname> Zoi </firstname>
  <lastname> Kaoudi </lastname>
  <age> 22 </age>
  <email> zkaoudi@dbnet.ntua.gr </email>
```

</undergraduate>

Η γλώσσα XML δίνει μια κωδικοποίηση στα δεδομένα που βοηθά την ανταλλαγή και επαναχρησιμοποίησή τους από πολλές εφαρμογές.

Ακόμα και με την γλώσσα XML, η δομή και οργάνωση που παρέχεται είναι στο επίπεδο των δεδομένων. Στο Σημασιολογικό Ιστό, η επικοινωνία εφαρμογών με διαφορετικά σχήματα απαιτεί δυνατότητες ορισμού και διαχείρισης μεταδεδομένων. Το πλαίσιο RDF είναι ένα τέτοιο εργαλείο αναπαράστασης "δεδομένων για τα δεδομένα". Σε ένα RDF αρχείο ορίζονται δηλώσεις για αντικείμενα του Ιστού, όπως σελίδες, συγγραφείς, προγράμματα, κ.λ.π. Για παράδειγμα, είναι δυνατή η δήλωση ότι η συγγραφέας της ιστοσελίδας www.dblab.ece.ntua.gr/~zkaoudi είναι προπτυχιακή φοιτήτρια, ηλικίας 22 χρονών. Μια επέκταση του πλαισίου RDF είναι το RDF Schema (RDFS), το οποίο και παρέχει μηχανισμούς περιγραφής σχετικών αντικειμένων του Ιστού και των σχέσεων μεταξύ τους. Το RDF Schema βασίζεται σε κλάσεις και ιδιότητες, έννοιες γνωστές από τον χώρο των αντικειμενοστρεφών συστημάτων. Η βασική διαφορά είναι ότι οι ιδιότητες είναι οντότητες πρώτης τάξης και ορίζονται ανεξάρτητα από τις κλάσεις. Για παράδειγμα, μπορούμε να ορίσουμε την τάξη student και τις υποκλάσεις postgraduate και undergraduate για να ομαδοποιήσουμε τις ιστοσελίδες των φοιτητών σε προπτυχιακές και μεταπτυχιακές, καθώς και να συσχετίσουμε τις τάξεις αυτές με χαρακτηριστικά (ηλικία, ενδιαφέροντα, κ.λ.π.).

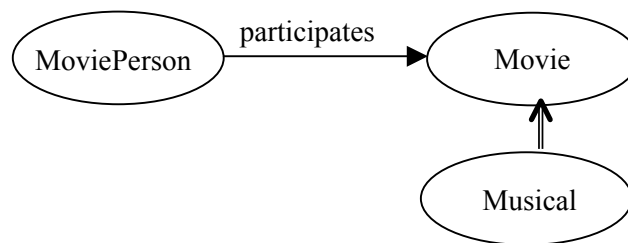
Το πλαίσιο RDF είναι πολύ σημαντικό στοιχείο του Σημασιολογικού Ιστού. Βελτιώνει τις δυνατότητες των μηχανών αναζήτησης στον Ιστό, περιγράφει καταλόγους θεματικών ιεραρχιών σε πύλες και ψηφιακές βιβλιοθήκες, υποστηρίζει την ανταλλαγή γνώσης μεταξύ πρακτόρων λογισμικού, κ.λ.π.

1.1 Αντικείμενο της διπλωματικής

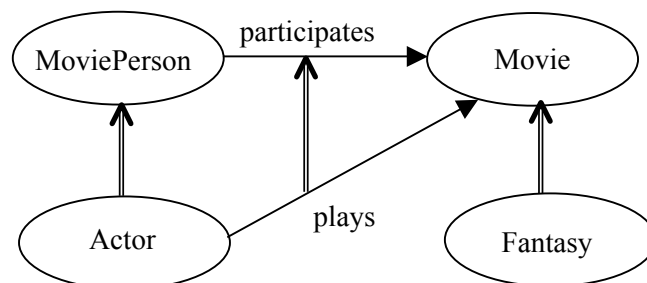
Το αντικείμενο της διπλωματικής είναι η ανάπτυξη ενός πρότυπου συστήματος διαχείρισης RDF σχημάτων. Το σύστημα αυτό παρέχει γλώσσα ερωτήσεων με τέσσερις βασικούς τελεστές: *επιλογή*, *ένωση*, *τομή*, *διαφορά*. Ο τελεστής *επιλογή* εφαρμόζεται σε ένα RDF σχήμα και προβάλλει κατά κάποιο τρόπο ένα μέρος του σχήματος αυτού. Ο τελεστής *ένωση* εφαρμόζεται σε δύο RDF σχήματα και τα συγχωνεύει σε ένα. Ο τελεστής *τομή* εφαρμόζεται επίσης σε δύο RDF σχήματα και βρίσκει το κοινό τους κομμάτι. Τέλος, ο τελεστής *διαφορά* εφαρμόζεται σε δύο RDF σχήματα και βρίσκει μέρος του πρώτου σχήματος που δεν υπάρχει στο δεύτερο. Η γλώσσα ερωτήσεων του συστήματος υποστηρίζεται με γραφικό περιβάλλον ερώτησης-με-χρήση-παραδείγματος (QBE).

Το πεδίο εφαρμογής των παραπάνω λειτουργιών διαχείρισης RDF σχημάτων είναι οι κόμβοι-πύλες Ιστού (portals) και οι ηλεκτρονικές αγορές. Μέσω της γλώσσας ερωτήσεων του συστήματος, ο χρήστης μπορεί, για παράδειγμα, να διαχειρίζεται ιεραρχίες πυλών Ιστού και να δημιουργεί καινούριες, εκτελώντας πράξεις επιλογής, ένωσης, τομής και διαφοράς σύμφωνα με τις επιλογές του.

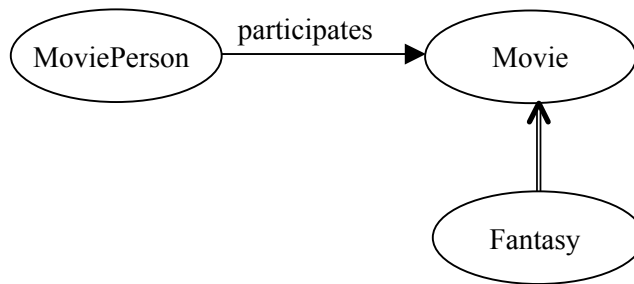
Έστω, λοιπόν, για παράδειγμα, ότι έχουμε καταλόγους με θέμα τη δημιουργία κινηματογραφικών ταινιών. Στα Σχήματα 1.1 και 1.2 φαίνονται δύο τέτοιοι κατάλογοι με τη μορφή RDF σχήματος. Καταρχήν, μπορούμε να προβάλλουμε μέρος του ενός σχήματος και να πάρουμε ένα νέο σχήμα. Για παράδειγμα, εάν από το Σχήμα 1.2 επιλέξουμε τους κόμβους MoviePerson, Movie και Fantasy, θα δημιουργήσουμε το RDF σχήμα που φαίνεται στο Σχήμα 1.3. Επίσης, εάν θέλουμε να πάρουμε το κοινό κομμάτι των RDFS που απεικονίζονται στα Σχήματα 1.1 και 1.2, μπορούμε να χρησιμοποιήσουμε τον τελεστή *τομή* παίρνοντας το RDF σχήμα που φαίνεται στο Σχήμα 1.4.



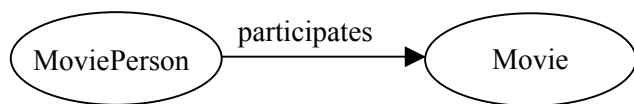
Σχήμα 1.1



Σχήμα 1.2



Σχήμα 1.3



Σχήμα 1.4

1.2 Οργάνωση του τόμου

Η εργασία αυτή χωρίζεται σε 8 κεφάλαια. Στο 2^ο Κεφάλαιο γίνεται μια περιγραφή του θέματος, δίνοντας το θεωρητικό υπόβαθρο που χρειάζεται και σχετικές εργασίες πάνω στο συγκεκριμένο θέμα. Στο 3^ο Κεφάλαιο παρουσιάζεται η θεωρητική μελέτη της διπλωματικής εργασίας. Δίνονται βασικοί ορισμοί πάνω στα RDF σχήματα καθώς και οι ορισμοί των πράξεων που υλοποιήσαμε. Στη συνέχεια, στο 4^ο Κεφάλαιο γίνεται η σχεδίαση και ανάλυση του συστήματος. Περιγράφονται τα υποσυστήματα του συστήματος όσο αναφορά την αρχιτεκτονική του και οι εφαρμογές του. Οι λεπτομέρειες υλοποίησης του συστήματος παρουσιάζονται στο 5^ο Κεφάλαιο. Εκεί δίνονται λεπτομέρειες για την εγκατάσταση όλων των εργαλείων που χρειάζονται, καθώς επίσης βασικοί αλγόριθμοι και ο σκελετός του κώδικα του προγράμματος. Στο 6^ο Κεφάλαιο γίνεται ο έλεγχος του συστήματος με βάση ένα σενάριο χρήσης του συστήματος. Στη συνέχεια, στο 7^ο Κεφάλαιο παρουσιάζεται μια σύνοψη της εργασίας και τυχόν μελλοντικές επεκτάσεις. Τέλος, η σχετική βιβλιογραφία δίνεται στο 8^ο Κεφάλαιο.

2

Περιγραφή θέματος

Στο κεφάλαιο αυτό παρουσιάζεται το θεωρητικό υπόβαθρο της διπλωματικής εργασίας, καθώς και σχετικές εργασίες και έρευνες που έχουν διεξαχθεί σε συναφή θέματα. Τέλος, παρουσιάζεται η συνεισφορά της παρούσας διπλωματικής πάνω σε αυτόν το θεματικό τομέα.

2.1 Θεωρητικό υπόβαθρο

2.1.1 RDF

Ο Σημασιολογικός Ιστός (*Semantic Web*) [7] είναι το επόμενο σημαντικό βήμα του Παγκόσμιου Ιστού. Εφαρμογές του Σημασιολογικού Ιστού όπως είναι οι πύλες γνώσης (*Knowledge portals*) και οι ηλεκτρονικές αγορές (*E-Marketplaces*) απαιτούν τη διαχείριση τεράστιων όγκων μεταδεδομένων, δηλαδή πληροφορίας που περιγράφει τα δεδομένα. Έτσι, με τις εφαρμογές αυτές τεράστιες ποσότητες πόρων ιστού (*web resources*), όπως δεδομένα, έγγραφα, προγράμματα, θα είναι διαθέσιμες μαζί με διαφόρων ειδών μεταπληροφορία. Η καλύτερη γνώση για το νόημα, τη χρησιμότητα, την προσπελασιμότητα ή την ποιότητα των πηγών αυτών θα διευκολύνει την αυτόματη επεξεργασία των διαθέσιμων υπηρεσιών στον Ιστό.

Το πλαίσιο RDF (*Resource Description Framework*) [6] επιτρέπει τη δημιουργία και την ανταλλαγή μεταδεδομένων των πόρων όπως και οποιαδήποτε άλλα δεδομένα στον Ιστό.

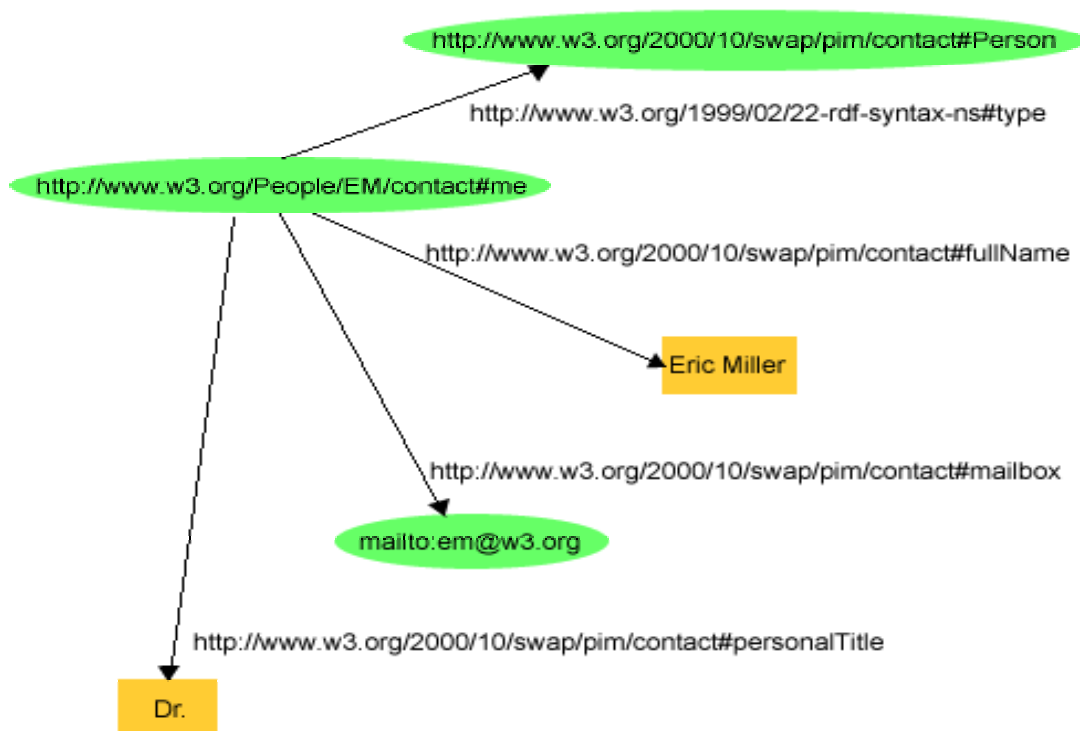
Είναι ιδιαίτερα προοριζόμενο για την αντιπροσώπευση μεταδεδομένων για τους πόρους του Ιστού, όπως τίτλος, συντάκτης και ημερομηνία τροποποίησης μιας ιστοσελίδας, πνευματικά δικαιώματα και πληροφορία χορήγησης αδειών για ένα έγγραφο του Ιστού, ή πρόγραμμα διαθεσιμότητας για κάποιο κοινό πόρο. Παρόλα αυτά, με τη γενίκευση της έννοιας "πόρος Ιστού", το RDF μπορεί επίσης να χρησιμοποιηθεί για να αντιπροσωπεύσει πληροφορίες για πράγματα που μπορούν να προσδιοριστούν στον Ιστό ακόμα και όταν δεν μπορούν να ανακτηθούν άμεσα από αυτόν. Παραδείγματα περιλαμβάνουν πληροφορίες για αντικείμενα διαθέσιμα από ηλεκτρονικές αγορές (πχ., πληροφορίες για τις προδιαγραφές, τις τιμές και τη διαθεσιμότητα) ή περιγραφή των προτιμήσεων ενός χρήστη του Ιστού για την παράδοση πληροφορίας.

Το μοντέλο RDF στοχεύει σε καταστάσεις στις οποίες αυτή η πληροφορία πρέπει να υποβληθεί σε επεξεργασία από εφαρμογές και όχι μόνο για να επιδειχθεί στους ανθρώπους. Επίσης, παρέχει ένα κοινό πλαίσιο για την πληροφορία αυτή έτσι ώστε να μπορεί να ανταλλαχθεί μεταξύ των εφαρμογών χωρίς απώλεια της σημασίας. Δεδομένου αυτού, οι σχεδιαστές εφαρμογών μπορούν να εκμεταλλευτούν τη διαθεσιμότητα κοινών RDF parsers και εργαλείων επεξεργασίας. Η δυνατότητα να ανταλλαχθεί πληροφορία μεταξύ διαφορετικών εφαρμογών σημαίνει ότι πληροφορία μπορεί να τεθεί στη διάθεση και άλλων εφαρμογών εκτός από εκείνες για τις οποίες δημιουργήθηκε αρχικά.

Το RDF είναι βασισμένο στην ιδέα του προσδιορισμού των πραγμάτων χρησιμοποιώντας προσδιοριστικά του Ιστού (web identifiers, αποκαλούμενα Uniform Resource Identifiers ή URIs) και της περιγραφής των πόρων από την άποψη απλών ιδιοτήτων και τιμών των ιδιοτήτων αυτών (*properties and property values*). Αυτό επιτρέπει στα RDF να αντιπροσωπεύουν απλές δηλώσεις για πόρους ως ένα γράφο από κόμβους και τόξα που αντιπροσωπεύουν τους πόρους, τις ιδιότητες και τις τιμές τους. Για να είμαστε πιο συγκεκριμένοι, η φράση «υπάρχει ένας άνθρωπος προσδιορισμένος από το <http://www.w3.org/People/EM/contact#me> του οποίου το όνομα είναι Eric Miller, η ηλεκτρονική διεύθυνση em@w3.org και ο τίτλος του είναι "Dr" » μπορεί να παρουσιαστεί σε ένα γράφο RDF όπως φαίνεται στο Σχήμα 2.1.

Το σχήμα αυτό δείχνει ότι το RDF χρησιμοποιεί URIs για να προσδιορίσει άτομα όπως ο Eric Miller που προσδιορίζεται από το <http://www.w3.org/People/EM/contact#me>, είδη πραγμάτων, π.χ. Person που προσδιορίζεται από <http://www.w3.org/2000/10/swap/pim/contact#Person>, ιδιότητες εκείνων των πραγμάτων, π.χ. ταχυδρομική θυρίδα που προσδιορίζεται από <http://www.w3.org/2000/10/swap/pim/contact#mailbox>, τιμές εκείνων των ιδιοτήτων, π.χ. <mailto:em@w3.org> ως τιμή της ιδιότητας ταχυδρομικής θυρίδας (το RDF χρησιμοποιεί επίσης συμβολοσειρές όπως "Eric Miller" και τιμές από άλλα

τύπους δεδομένων (*datatypes*) όπως ακέραιοι αριθμοί και ημερομηνίες, ως τιμές των ιδιοτήτων).



Σχήμα 2.1

Επίσης, το RDF παρέχει μια σύνταξη σε XML μορφή (RDF/XML) για την καταγραφή και την ανταλλαγή τέτοιων γράφων. Το παρακάτω παράδειγμα δείχνει το παραπάνω σχήμα στη μορφή αυτή:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```

Όπως στην HTML, το RDF/XML είναι μηχανή επεξεργάσιμη και χρησιμοποιώντας URIs μπορεί να συνδέσει κομμάτια πληροφοριών στον Ιστό. Εντούτοις, αντίθετα από το συμβατικό υπερκείμενο (*hypertext*), τα URIs μπορούν να αναφερθούν σε οποιοδήποτε ευπροσδιόριστο

πράγμα, συμπεριλαμβανομένων και εκείνων που μπορεί να μην είναι άμεσα ανακτήσιμα από τον Ιστό (όπως το πρόσωπο Eric Miller). Το αποτέλεσμα είναι ότι εκτός από την περιγραφή πραγμάτων όπως ιστοσελίδων, τα RDF μπορούν επίσης να περιγράψουν αυτοκίνητα, επιχειρήσεις, ανθρώπους, γεγονότα ειδήσεων κλπ. Επιπλέον, οι ιδιότητες στα RDF έχουν URIs, για να προσδιορίσουν ακριβώς τις σχέσεις που υπάρχουν μεταξύ των συνδεμένων στοιχείων.

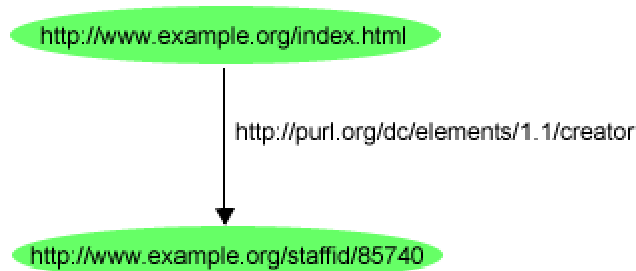
Γενικότερα, το RDF είναι βασισμένο στην ιδέα ότι τα πράγματα που περιγράφονται έχουν ιδιότητες (*properties*) που έχουν τιμές και ότι οι πόροι μπορούν να περιγραφούν με εκφράσεις οι οποίες διευκρινίζουν τις ιδιότητες αυτές και τις τιμές τους. Το RDF χρησιμοποιεί μια ιδιαίτερη ορολογία για τα διάφορα μέρη των εκφράσεων. Συγκεκριμένα, το μέρος που προσδιορίζει το πράγμα για το οποίο είναι η πρόταση (μια ιστοσελίδα για παράδειγμα) καλείται υποκείμενο (*subject*). Το μέρος που προσδιορίζει την ιδιότητα ή το χαρακτηριστικό του υποκειμένου που η πρόταση διευκρινίζει (δημιουργός, δημιουργία-ημερομηνία, ή γλώσσα σε αυτά τα παραδείγματα) καλείται κατηγορημα (*predicate*) και το μέρος που προσδιορίζει την τιμή εκείνης της ιδιότητας καλείται αντικείμενο (*object*). Έτσι, παίρνοντας την αγγλική πρόταση : `http://www.example.org/index.html has a creator whose value is John Smith` θα έχουμε:

```
subject: http://www.example.org/index.html
predicate: http://purl.org/dc/elements/1.1/creator
object: http://www.example.org/staffid/85740
```

Το RDF μοντελοποιεί τέτοιες εκφράσεις ως κόμβους και τόξα σε ένα γράφο. Με αυτήν την μοντελοποίηση μια πρόταση αντιπροσωπεύεται από:

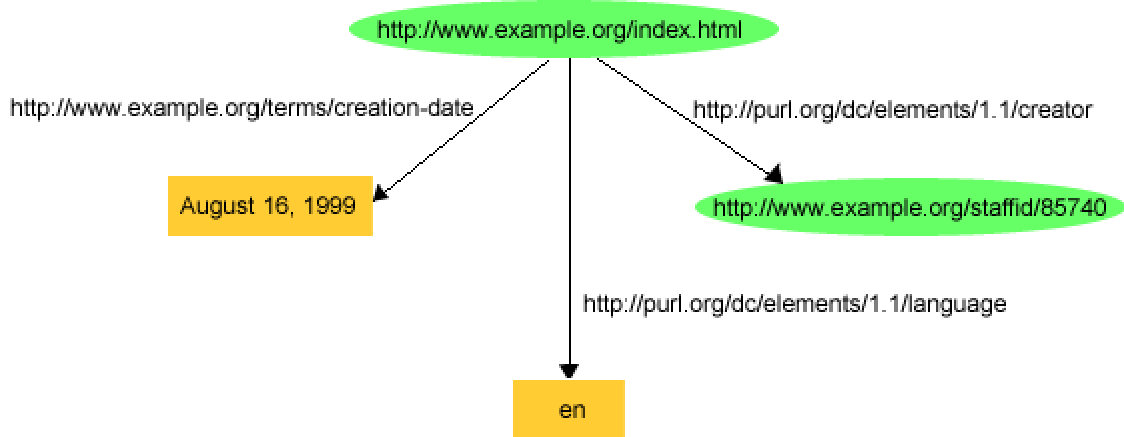
- ένα κόμβο (*node*) για το υποκείμενο
- ένα κόμβο (*node*) για το αντικείμενο
- ένα τόξο (*arc*) για το κατηγορημα κατευθυνόμενο από το υποκείμενο στο αντικείμενο

Έτσι, η παραπάνω πρόταση θα μπορούσε να αναπαρασταθεί με τον γράφο που φαίνεται στο Σχήμα 2.2.



Σχήμα 2.2

Στο παράδειγμα του Σχήματος 2.3 μπορούμε να διαπιστώσουμε ότι τα αντικείμενα του RDF μπορούν να είναι είτε URIs είτε σταθερές τιμές που λέγονται κυριολεκτικά (*literals*) και αναπαριστούνται από συμβολοσειρές προκειμένου να αντιπροσωπεύσουν ορισμένα είδη ιδιοτήτων. Τα κυριολεκτικά δεν μπορούν να χρησιμοποιηθούν ως υποκείμενα ή ως κατηγορήματα σε RDF εκφράσεις.



Σχήμα 2.3

Ένας εναλλακτικός τρόπος να αναπαραστήσουμε ένα RDF είναι γράφοντας κάθε έκφραση που προκύπτει, δηλαδή σε μορφή triples. Έτσι, το παραπάνω σχήμα θα μπορούσε να γραφεί ως εξής:

```

<http://www.example.org/index.html>
<http://purl.org/dc/elements/1.1/creator>
<http://www.example.org/staffid/85740>.

<http://www.example.org/index.html>
<http://www.example.org/terms/creation-date> "August 16, 1999".

<http://www.example.org/index.html>
<http://purl.org/dc/elements/1.1/language> "en".
  
```

Για μεγαλύτερη ευκολία, αντί να γράφονται πλήρως τα URIs το RDF δίνει τη δυνατότητα για χρήση των namespaces. Έτσι, έχοντας ορίσει τα namespaces

```

ex: http://www.example.org/
dc: http://purl.org/dc/elements/1.1/
exterm: http://www.example.org/terms/
exstaff: http://www.example.org/staffid/
  
```

Το παράδειγμα αυτό θα μπορούσε να γραφτεί ως εξής:

```

ex:index.html dc:creator exstaff:85740.
ex:index.html exterm:creation-date "August 16, 1999".
ex:index.html dc:language "en".
  
```

Τέλος, το παραπάνω RDF θα μπορούσε να γραφτεί σε μορφή RDF/XML, όπως προαναφέραμε, ως εξής:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:extterms="http://www.example.org/terms/">

  <rdf:Description rdf:about="http://www.example.org/index.html">
    <extterms:creation-date>August 16, 1999</extterms:creation-date>
  </rdf:Description>

</rdf:RDF>
```

2.1.2 RDF Schema

Το πλαίσιο RDF παρέχει έναν τρόπο να εκφραστούν απλές εκφράσεις για πόρους, χρησιμοποιώντας ονομασμένες ιδιότητες και τιμές. Παρόλα αυτά, οι κοινότητες χρηστών RDF χρειάζονται επίσης τη δυνατότητα να καθορίσουν το λεξιλόγιο (όρους) που αυτοί σκοπεύουν να χρησιμοποιήσουν σε εκείνες τις εκφράσεις, και συγκεκριμένα, να δείξουν ότι περιγράφουν συγκεκριμένα είδη ή κατηγορίες πόρων και γι' αυτό θα χρησιμοποιήσουν συγκεκριμένες ιδιότητες στην περιγραφή εκείνων των πόρων. Για παράδειγμα, άνθρωποι ενδιαφερόμενοι για την περιγραφή βιβλιογραφικών πόρων θα ήθελαν να περιγράψουν κατηγορίες όπως `ex:Book` ή `ex:MagazineArticle` και να χρησιμοποιήσουν ιδιότητες όπως `ex:author`, `ex:title` και `ex:subject` για να τις περιγράψουν. Το ίδιο το RDF δεν παρέχει κανένα μέσο για τον προσδιορισμό κλάσεων και ιδιοτήτων. Αντί αυτού, τέτοιες κλάσεις και ιδιότητες περιγράφονται ως ένας RDF λεξιλόγιο χρησιμοποιώντας επεκτάσεις του RDF που είναι το RDF Schema ή RDFS.

Το RDFS παρέχει τις προϋποθέσεις που απαιτούνται για να περιγράψει κανείς τέτοιες κλάσεις και ιδιότητες και για να προσδιορίσει ποιες κλάσεις και ιδιότητες αναμένονται να χρησιμοποιηθούν μαζί. Με άλλα λόγια, το RDF σχήμα παρέχει ένα σύστημα τύπων για RDF. Το σύστημα αυτό για το RDFS είναι παρόμοιο κατά κάποιον τρόπο με τα συστήματα τύπων των αντικειμενοστρεφών γλωσσών προγραμματισμού όπως η Java. Παραδείγματος χάριν, το RDF σχήμα επιτρέπει στους πόρους να είναι ορισμένοι ως στιγμιότυπα μιας ή περισσότερων κλάσεων. Επιπλέον, επιτρέπει στις κλάσεις να είναι οργανωμένες με έναν ιεραρχικό τρόπο. Για παράδειγμα, μια κλάση `ex:Dog` μπορεί να οριστεί ως υποκλάση της `ex:Mammal` που είναι υποκλάση της `ex:Animal`, σημαίνοντας ότι οποιοσδήποτε πόρος που είναι στιγμιότυπο της κλάσης `ex:Dog` είναι επίσης έμμεσα στιγμιότυπο της κλάσης `ex:Animal`. Εντούτοις, οι κλάσεις και οι ιδιότητες του RDF είναι κατά κάποιον τρόπο πολύ διαφορετικές από τους τύπους των γλωσσών προγραμματισμού. Η περιγραφή των RDF κλάσεων και ιδιοτήτων δεν δημιουργούν ένα καλούπι στο οποίο οι πληροφορίες πρέπει να ανήκουν, αλλά παρέχουν πρόσθετη πληροφορία για τους πόρους RDF που περιγράφουν.

Ένα βασικό βήμα σε οποιοδήποτε είδος διαδικασίας περιγραφής είναι ο προσδιορισμός των διαφόρων πραγμάτων που περιγράφονται. Το RDF σχήμα αναφέρεται σε αυτά τα "πράγματα" ως κλάσεις. Μια κλάση (class) αντιστοιχεί στη γενική έννοια ενός τύπου ή μιας κατηγορίας, όπως την έννοια μιας κλάσης στις αντικειμενοστρεφείς γλώσσες προγραμματισμού. Οι κλάσεις μπορούν να χρησιμοποιηθούν για να αντιπροσωπεύσουν σχεδόν οποιαδήποτε κατηγορία πράγματος, όπως ιστοσελίδες, ανθρώπους, τύπους εγγράφων, βάσεις δεδομένων ή αφηρημένες έννοιες. Οι κλάσεις στο RDFS περιγράφονται χρησιμοποιώντας τους πόρους `rdfs:Class` και `rdfs:Resource` και τις ιδιότητες `rdf:type` και `rdfs:subClassOf`.

Ας πάρουμε για παράδειγμα μια εταιρία που έχει οχήματα (vehicles). Σε αυτό το παράδειγμα μπορούμε να ορίσουμε μια ιεραρχία όπως φαίνεται στο παρακάτω σχήμα:



Σχήμα 2.4

Αυτή η ιεραρχία μπορεί να γραφτεί σε RDF/XML μορφή με τη χρήση namespaces ως εξής:

```

<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd
"http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://example.org/schemas/vehicles">

  <rdfs:Class rdf:ID="MotorVehicle"/>

  <rdfs:Class rdf:ID="PassengerVehicle">
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdfs:Class>

  <rdfs:Class rdf:ID="Truck">
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdfs:Class>

```

```

<rdfs:Class rdf:ID="Van">
  <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdfs:Class>

<rdfs:Class rdf:ID="MiniVan">
  <rdfs:subClassOf rdf:resource="#Van"/>
  <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
</rdfs:Class>

</rdf:RDF>

```

Το RDF/XML στο παράδειγμα αυτό εισάγει ονόματα, όπως το `MotorVehicle`, για τους πόρους (κλάσεις) που περιγράφει χρησιμοποιώντας το `rdf:ID`. Αυτό είναι χρήσιμο εδώ επειδή συντομεύει το `URIref` και παρέχει έναν πρόσθετο έλεγχο ότι η τιμή του `rdf:ID` είναι μοναδική σε όλη την τρέχουσα βάση URI (συνήθως το έγγραφο URI). Αυτό βοηθά να πάρει τις επαναλαμβανόμενες τιμές `rdf:ID` κατά την καθορισμό των ονομάτων των κλάσεων και των ιδιοτήτων στα RDF σχήματα. Σχετικά `URIrefs` βασισμένα σε αυτά τα ονόματα μπορούν έπειτα να χρησιμοποιηθούν σε άλλους ορισμούς κλάσεων μέσα στο ίδιο σχήμα (πχ. όπως το `#MotorVehicle` χρησιμοποιείται στη περιγραφή των άλλων κλάσεων). Το πλήρες `URIref` αυτής της κλάσης, που υποθέτει ότι το ίδιο το σχήμα ήταν ο πόρος `http://example.org/schemas/vehicles`, θα ήταν `http://example.org/schemas/vehicles#MotorVehicle`. Για να εξασφαλιστεί ότι οι αναφορές σε αυτές τις κατηγορίες σχημάτων θα διατηρηθούν με συνέπεια ακόμα κι αν το σχήμα μετατοπιστεί ή αντιγραφτεί (ή για να ορίσει απλά μια βάση `URIref` για τις κλάσεις σχημάτων χωρίς να υποθέσουμε ότι όλες υπάρχουν σε μια μοναδική θέση), οι περιγραφές των κλάσεων μπορούν επίσης να περιλάβουν τη δήλωση `xml:base="http://example.org/schemas/vehicles"`.

Εκτός από την περιγραφή συγκεκριμένων κλάσεων πραγμάτων που θέλουν να περιγράψουν, οι κοινότητες χρηστών πρέπει επίσης να είναι σε θέση να περιγράψουν συγκεκριμένες ιδιότητες που χαρακτηρίζουν εκείνες τις κλάσεις (όπως `rearSeatLegRoom` για να περιγράψει ένα επιβατικό όχημα (passenger vehicle)). Στο RDF σχήμα, οι ιδιότητες (properties) περιγράφονται χρησιμοποιώντας την RDF κλάση `rdf:Property` και τις ιδιότητες σχημάτων RDF `rdfs:domain`, `rdfs:range`, και `rdfs:subPropertyOf`. Επίσης, παρέχει λεξιλόγιο για να περιγράψει πώς οι ιδιότητες και οι κλάσεις προορίζονται για να χρησιμοποιηθούν μαζί στα RDF δεδομένα. Η σημαντικότερη πληροφορία αυτού του είδους παρέχεται με τη χρησιμοποίηση των ιδιοτήτων `rdfs:range` και `rdfs:domain` για να περιγράψουν περαιτέρω τις ιδιότητες που έχουν οριστεί από εφαρμογή.

Το `rdfs:range` χρησιμοποιείται για να δείξει ότι οι τιμές μιας συγκεκριμένης ιδιότητας είναι στιγμιότυπα μιας οριζόμενης κλάσης. Παραδείγματος χάριν, εάν θέλαμε να δείξουμε ότι η ιδιότητα `ex:author` έχει τιμές που είναι στιγμιότυπα της κλάσης `ex:Person`, θα γράφαμε τα RDF triples:

```

ex:Person    rdf:type    rdfs:Class.
ex:author    rdf:type    rdf:Property.
ex:author    rdfs:range  ex:Person.

```

Μια ιδιότητα (*property*), για παράδειγμα το `ex:hasMother`, μπορεί να έχει ένα ή περισσότερα από ένα `range properties`. Εάν το `ex:hasMother` δεν έχει κανένα `range` τότε τίποτα δεν λέγεται για τις τιμές της ιδιότητας `ex:hasMother`. Εάν το `ex:hasMother` έχει ένα `range`, για παράδειγμα το `ex:Person`, αυτό λέει ότι οι τιμές της ιδιότητας `ex:hasMother` είναι στιγμιότυπα της κλάσης `ex:Person`. Εάν το `ex:hasMother` έχει περισσότερα από ένα `ranges`, για παράδειγμα το `ex:Person` και το `ex:Female`, αυτό λέει ότι οι τιμές του `ex:hasMother` είναι πόροι που είναι στιγμιότυπα όλων των κλάσεων που είναι `ranges` δηλ., ότι οποιαδήποτε τιμή του `ex:hasMother` είναι και `ex:Female` και `ex:Person`. Τέλος, το `rdfs:range` μπορεί επίσης να χρησιμοποιηθεί για να δείξει ότι η τιμή μιας ιδιότητας μπορεί να δίνεται και από κυριολεκτικό (*literal*).

Το `rdfs:domain` χρησιμοποιείται για να δείξει ότι μια συγκεκριμένη ιδιότητα ισχύει για μια οριζόμενη κλάση. Παραδείγματος χάριν, εάν θέλαμε να δείξουμε ότι η ιδιότητα `ex:author` ισχύει για τα στιγμιότυπα της κλάσης `ex:Book`, θα γράφαμε τα RDF triples:

```

ex:Book      rdf:type    rdfs:Class.
ex:author    rdf:type    rdf:Property.
ex:author    rdfs:domain ex:Book.

```

Μια δεδομένη ιδιότητα, για παράδειγμα `ex:weight`, μπορεί να έχει ένα ή περισσότερα από ένα `domain`. Εάν το `ex:weight` δεν έχει κανένα `domain`, τότε τίποτα δεν λέγεται για τους πόρους με τους οποίους το `ex:weight` μπορεί να χρησιμοποιηθεί. Εάν το `ex:weight` έχει ένα `domain`, για παράδειγμα το `ex:Book`, αυτό λέει ότι η ιδιότητα `ex:weight` ισχύει για τα στιγμιότυπα της κλάσης `ex:Book`. Εάν το `ex:weight` έχει περισσότερα από ένα `domain`, για παράδειγμα το `ex:Book` και το `ex:MotorVehicle`, αυτό λέει ότι οποιοσδήποτε πόρος που έχει μια ιδιότητα `ex:weight` είναι στιγμιότυπο όλων των κλάσεων που είναι `domain` του `property` αυτού, δηλ. οποιοσδήποτε πόρος που έχει ιδιότητα `ex:weight` είναι και ένα `ex:Book` και ένα `ex:MotorVehicle`.

Το RDF σχήμα παρέχει έναν τρόπο να ειδικευτούν οι ιδιότητες, όπως και οι κλάσεις. Αυτή η σχέση ειδίκευσης μεταξύ δύο ιδιοτήτων περιγράφεται χρησιμοποιώντας την προκαθορισμένη ιδιότητα `rdfs:subPropertyOf`. Παραδείγματος χάριν, εάν τα `ex:primaryDriver` και `ex:driver` είναι και οι δύο ιδιότητες, θα μπορούσαμε να περιγράψουμε αυτές τις ιδιότητες, και το γεγονός ότι το `ex:primaryDriver` είναι μια ειδίκευση του `ex:driver`, με το γράψιμο των RDF δηλώσεων:

```

ex:driver      rdf:type    rdf:Property.
ex:primaryDriver  rdf:type    rdf:Property.
ex:primaryDriver  rdfs:subPropertyOf  ex:driver.

```

Μια ιδιότητα (property) μπορεί να είναι ένα υπο-ιδιότητα (subproperty) καμίας, μίας ή περισσότερων ιδιοτήτων. Τα `rdfs:range` και `rdfs:domain` του RDFS που ισχύουν για μια ιδιότητα ισχύουν επίσης για κάθε μία από τις υπο-ιδιότητές της. Έτσι, στο παραπάνω παράδειγμα, το σχήμα RDF καθορίζει το `ex:primaryDriver` επίσης να έχει `rdfs:domain` το `ex:MotorVehicle`, λόγω της σχέσης `subproperty` με το `ex:driver`.

Παρακάτω φαίνεται ένα ολοκληρωμένο παράδειγμα σε μορφή RDF/XML:

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd
"http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://example.org/schemas/vehicles">

  <rdfs:Class rdf:ID="MotorVehicle"/>

  <rdfs:Class rdf:ID="PassengerVehicle">
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdfs:Class>

  <rdfs:Class rdf:ID="Truck">
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdfs:Class>

  <rdfs:Class rdf:ID="Van">
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdfs:Class>

  <rdfs:Class rdf:ID="MiniVan">
    <rdfs:subClassOf rdf:resource="#Van"/>
    <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
  </rdfs:Class>

  <rdfs:Class rdf:ID="Person"/>

  <rdfs:Datatype rdf:about="xsd:integer"/>

  <rdf:Property rdf:ID="registeredTo">
    <rdfs:domain rdf:resource="#MotorVehicle"/>
    <rdfs:range rdf:resource="#Person"/>
  </rdf:Property>

  <rdf:Property rdf:ID="rearSeatLegRoom">
    <rdfs:domain rdf:resource="#PassengerVehicle"/>
    <rdfs:range rdf:resource="xsd:integer"/>
  </rdf:Property>

  <rdf:Property rdf:ID="driver">
    <rdfs:domain rdf:resource="#MotorVehicle"/>
  </rdf:Property>

  <rdf:Property rdf:ID="primaryDriver">
    <rdfs:subPropertyOf rdf:resource="#driver"/>
  </rdf:Property>

</rdf:RDF>
```

Στο σημείο αυτό, πρέπει να πούμε ότι παρόλο που το σύστημα τύπων του RDFS είναι παρόμοιο με αυτό των αντικειμενοστραφών γλωσσών προγραμματισμού, υπάρχουν σημαντικές διαφορές σε αρκετά σημεία. Πρώτον, το RDF Schema, αντί να περιγράφει τις κλάσεις σαν να έχουν κάποιες συγκεκριμένες ιδιότητες, περιγράφει τις ιδιότητες σαν να εφαρμόζονται σε συγκεκριμένους πόρους κλάσεις ως πεδίο ορισμού ή πεδίο τιμών. Για παράδειγμα, αν είχαμε μια κλάση Book με ένα χαρακτηριστικό author και μια άλλη κλάση SoftwareModule που επίσης είχε χαρακτηριστικό author, αυτά τα δύο χαρακτηριστικά θα θεωρούνταν διαφορετικά στις γλώσσες προγραμματισμού. Με άλλα λόγια, το πεδίο δράσης ενός χαρακτηριστικού καθορίζεται μέσα στην κλάση που το ορίζει. Αντίθετα, στο RDFS οι ιδιότητες είναι ανεξάρτητες και έχουν καθολικό πεδίο δράσης (εκτός αν οριστούν να εφαρμόζονται σε συγκεκριμένες κλάσεις χρησιμοποιώντας το πεδίο ορισμού).

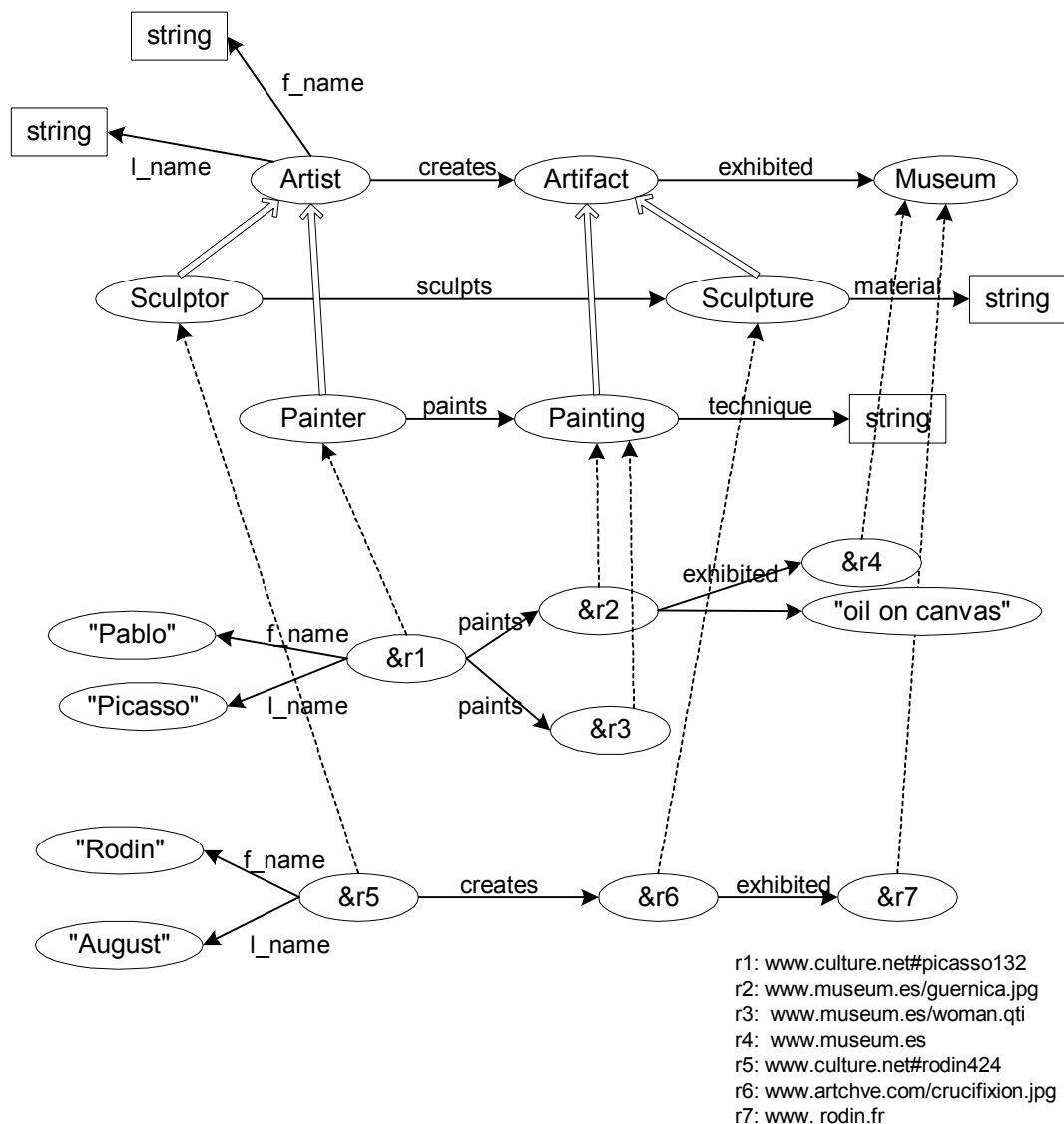
Επιπλέον, οι ιδιότητες στο RDF μπορούν να κληρονομηθούν σε αντίθεση με τον αντικειμενοστρεφή προγραμματισμό που κάτι τέτοιο δεν ισχύει. Δηλαδή οι ιδιότητες μπορούν να κληρονομούν χαρακτηριστικά από άλλες ιδιότητες δημιουργώντας έτσι μια ιεραρχία ιδιοτήτων, όπως και οι κλάσεις.

Τέλος, μια εξίσου σημαντική διαφορά είναι ότι στο RDF τα στιγμιότυπα μιας ιδιότητας είναι προαιρετικά. Δηλαδή είναι δυνατό να υπάρχει ένα στιγμιότυπο μιας κλάσης χωρίς να υπάρχει στιγμιότυπο της ιδιότητας που εφαρμόζεται ως πεδίο ορισμού. Για παράδειγμα, στον προγραμματισμό, ένα στιγμιότυπο της κλάσης Book δεν επιτρέπεται να μην έχει το χαρακτηριστικό author, και επίσης, εάν το χαρακτηριστικό author έπαιρνε τιμές που είναι στιγμιότυπα της κλάσης Person, δε θα επιτρεπόταν να μην έχει τιμή. Αντίθετα, το RDF παρέχει πληροφορία για το σχήμα ως κάτι επιπρόσθετο χωρίς να επιβάλλει περιορισμούς στο πώς να χρησιμοποιηθεί η πληροφορία αυτή.

2.1.3 Η γλώσσα RQL (RDF Query Language)

Η RQL [1] είναι μια συναρτησιακή γλώσσα και στηρίζεται σε ένα επίσημο πρότυπο για κατευθυνόμενους γράφους με ετικέτες που επιτρέπει την ερμηνεία των περιγραφών πόρων με τη βοήθεια ενός ή περισσότερων RDF σχημάτων. Η RQL προσαρμόζει την λειτουργία των ημιδομημένων/XML γλωσσών διατύπωσης ερωτήσεων στις ιδιαιτερότητες του RDF αλλά, κυρίως, επιτρέπει επερωτήσεις ομοιόμορφα και στα δεδομένα και στα σχήματα των πόρων.

Ως παράδειγμα για την καλύτερη κατανόηση της RQL θα θεωρήσουμε έναν κατάλογο πυλών (*portal catalog*) για μουσεία. Στο κάτω μέρος του Σχήματος 2.5 βλέπουμε να αναπαριστούνται τα δεδομένα για τις δύο ιστοσελίδες των μουσείων (&t4 και &t7). Στο επάνω μέρος του σχήματος βλέπουμε ένα RDF σχήμα για ειδικούς μουσείων.



Σχήμα 2.5

Η RQL καθορίζεται από ένα σύνολο βασικών ερωτήσεων που μπορούν να χρησιμοποιηθούν για να χτίσουν νέες ερωτήσεις μέσω της λειτουργικής σύνθεσης. Υποστηρίζει γενικευμένες εκφράσεις μονοπατιών (*generalized path expressions*) που χαρακτηρίζουν μεταβλητές στις ετικέτες και των κόμβων (δηλ., κλάσεων) και των ακμών (δηλ., ιδιοτήτων). Ο ομαλός συνδυασμός εκφράσεων μονοπατιών στα σχήματα και στα δεδομένα είναι το κύριο χαρακτηριστικό για την ικανοποίηση των αναγκών διάφορων σημασιολογικών εφαρμογών του Ιστού όπως οι πύλες γνώσης και οι ηλεκτρονικές αγορές.

Ο πυρήνας των RQL ερωτήσεων παρέχει ουσιαστικά τα μέσα να προσεγγιστούν οι βάσεις περιγραφής RDF με την ελάχιστη γνώση του εν λόγω σχήματος. Αυτές οι ερωτήσεις μπορούν να χρησιμοποιηθούν σε μια απλή διεπαφή για αναζήτηση στις βάσεις περιγραφής RDF. Παραδείγματος χάριν, στις πύλες γνώσης, για κάθε θέμα (δηλαδή κλάση), κάποιος μπορεί να

πλοηγήσει τα επιμέρους θέματά του (δηλαδή υποκλάσεις) και να ανακαλύψει τελικά τους πόρους (ή το συνολικό αριθμό τους) που είναι άμεσα ταξινομημένοι κάτω από αυτούς.

Για να διασχίσουμε τις ιεραρχίες κλάσεων / ιδιοτήτων που καθορίζονται σε ένα σχήμα, η RQL παρέχει λειτουργίες όπως `subClassOf` (για τις μεταβατικές υποκλάσεις) και `subClassOf^` (για τις άμεσες υποκλάσεις). Παραδείγματος χάριν, η ερώτηση `subClassOf^(Artist)` επιστρέφει ένα bag με τα ονόματα κλάσεων `Painter` και `Sculptor`. Παρόμοιες λειτουργίες υπάρχουν και για τις ιδιότητες (δηλ., `subPropertyOf` και `subPropertyOf^`). Κατόπιν, για μια συγκεκριμένη ιδιότητα μπορούμε να βρούμε τον ορισμό της με την εφαρμογή της συνάρτησης `domain` και `range`. Παραδείγματος χάριν, το `domain(creates)` επιστρέφει το όνομα κλάσης `Artist`.

Μπορούμε να έχουμε πρόσβαση στα δεδομένα των κλάσεων με το να γράψουμε το όνομά τους. Παραδείγματος χάριν, η ερώτηση `Artist` επιστρέφει ένα bag που περιέχει τα URIs `www.culture.net#rodin424 (&r5)` και `www.culture.net#picasso132 (&r1)`, δεδομένου ότι αυτοί οι πόροι ανήκουν στην επέκταση της κλάσης `Artist`. Πρέπει να τονιστεί ότι, εξ' ορισμού η RQL χρησιμοποιεί μια εκτεταμένη ερμηνεία κλάσης (ή ιδιότητας), δηλαδή την ένωση του συνόλου κατάλληλων στιγμιότυπων μιας κλάσης με εκείνα όλων των υποκλάσεων της. Προκειμένου να ληφθούν τα κατάλληλα στιγμιότυπα μιας κλάσης (δηλ., μόνο οι κόμβοι επονομαζόμενοι με το όνομα κλάσης αυτής), η RQL παρέχει τον ειδικό χειριστή ("[^]"): π.χ.. `^Artist`.

Επιπλέον, η RQL χρησιμοποιεί ως σημεία εισόδου σε μια περιγραφή RDF όχι μόνο τα ονόματα των κλάσεων αλλά και τα ονόματα των ιδιοτήτων. Παραδείγματος χάριν, θεωρώντας τις ιδιότητες ως δυαδικές σχέσεις, η απλή ερώτηση `creates` επιστρέφει ένα bag με διατεταγμένα ζεύγη των πόρων που ανήκουν στη εκτεταμένη ερμηνεία του `creates`:

<i>source</i>	<i>target</i>
&r5	&r6
&r1	&r2
&r1	&r3

Για τις περιπτώσεις που χρησιμοποιούνται τα ίδια ονόματα σε διαφορετικά σχήματα, κάποιος μπορεί να χρησιμοποιήσει μια πρόταση `namespace` για να επιλύσει προβλήματα σύγκρουσης των ονομάτων π.χ.:

```
ns:title
Using Namespace ns=&www.olcl.org/$schema2.rdf#
```

Παρακάτω θα επικεντρωθούμε σε RQL ερωτήσεις πάνω στο RDF Schema του Σχήματος 2.5.

Παραδείγματος χάριν, στην ακόλουθη ερώτηση, λαμβάνοντας υπόψη μια συγκεκριμένη ιδιότητα του σχήματος θέλουμε να βρούμε όλες τις σχετικές κλάσεις:

Q1: Ποιες κλάσεις μπορούν να υπάρξουν ως πεδίο ορισμού και πεδίο τιμών της ιδιότητας *creates*?

```
SELECT $C1, $C2
FROM   {$C1}creates{$C2}
```

\$C1	\$C2
Artist	Artifact
Artist	Painting
Artist	Sculpture
Painter	Artifact
Painter	Painting
Painter	Sculpture
Sculptor	Artifact
Sculptor	Painting
Sculptor	Sculpture

Στην πρόταση *from*, χρησιμοποιούμε μια βασική έκφραση μονοπατιών σε σχήμα (*schema path expression*) που αποτελείται από το όνομα του property *creates* (δηλ., μια ετικέτα ακμής) και δύο μεταβλητές κλάσης *\$C1* και *\$C2* (δηλ., μεταβλητές πάνω σε ετικέτες κόμβων). Ο συμβολισμός *{ }* χρησιμοποιείται στις εκφράσεις μονοπατιών για να εισάγει κατάλληλες μεταβλητές σχημάτων ή δεδομένων. Δεδομένου ότι οι ιδιότητες του RDF μπορούν να εφαρμοστούν σε οποιαδήποτε υποκλάση του πεδίου ορισμού και τιμών τους (λόγω του πολυμορφισμού), η έκφραση *{ \$C1 }creates{ \$C2 }* απλά δείχνει ότι τα *\$C1* και *\$C2* επαναλαμβάνονται πάνω στο *subclassof(domain(creates))* και *subclassof(range(creates))* αντίστοιχα (συμπεριλαμβανομένης της ρίζας). Πρέπει να τονιστεί ότι ένα τέτοιο είδος RQL εκφράσεων μονοπατιών μπορούν να αποτελούνται όχι μόνο από τις ετικέτες ακμών όπως το *creates*, αλλά και από τις ετικέτες κόμβων όπως το *Artist*. Η έκφραση *Artist{ \$C }* είναι ένας συντομότερος τρόπος για το *subclassof(Artist){ C }* (συμπεριλαμβανομένου και της ρίζας *Artist*).

Η πρόταση *select* καθορίζει μια προβολή πάνω στις μεταβλητές ενδιαφέροντος (π.χ., *\$C1*, *\$C2*). Επιπλέον, μπορούμε να χρησιμοποιήσουμε "*select **" για να περιλάβουμε στο αποτέλεσμα τις τιμές όλων των μεταβλητών που περιέχονται στο *from*. Αυτή η προβολή θα

κατασκευάσει μια διαταγμένη εγγραφή (δηλ., μια ακολουθία) της οποίας τα πεδία εξαρτούνται από τον αριθμό των χρησιμοποιούμενων μεταβλητών.

Ας δούμε τώρα πώς μπορούμε να πάρουμε όλες τις σχετικές ιδιότητες του σχήματος για μια συγκεκριμένη κλάση:

Q2: Βρες όλες τις ιδιότητες και το πεδίο ορισμού τους που μπορούν να εφαρμοστούν στην κλάση *Painter*.

```
SELECT @P, range(@P)
FROM   {C}@P
WHERE  C = Painter
```

@P	range(@P)
creates	Artifact
paints	Painting
lname	string
fname	string

Στην πρόταση `from` της Q2, χρησιμοποιούμε μια άλλη έκφραση μονοπατιών πάνω στο σχήμα που αποτελείται από μια μεταβλητή κλάσης `C` και μια μεταβλητή ιδιότητας `@P`. Γενικά, οι μεταβλητές ιδιοτήτων προτάσσονται από `@`. Για κάθε πιθανή τιμή της `@P`, η μεταβλητή κλάσης `C` παίρνει τιμές από το `subclassof(domain(p))`. Ο όρος `where` θα φιλτράρει τις αξιολογήσεις του `@P` για να κρατήσει μόνο εκείνες τις ιδιότητες για τις οποίες η κλάση *Painter* είναι ίση με το `domain` τους (π.χ., `paints`) ή είναι μια έγκυρη υποκλάση του `domain` τους (π.χ., `creates`, `lname`, `fname`). Σημειώστε ότι στο αποτέλεσμα Q2, το `range` μπορεί να είναι είτε κλάση είτε κυριολεκτικό δεδομένου ότι οι ιδιότητες των δεδομένων μπορούν να κυμανθούν σε κλάσεις (δηλ., αντιπροσωπεύουν σχέσεις) και κυριολεκτικά τύπων (δηλ., αντιπροσωπεύουν ιδιότητες).

Επίσης, στις εκφράσεις μονοπατιών υπάρχει ο συμβολισμός `{x;C}` όπου φιλτράρει τους κόμβους `x` (δηλ., πόροι) που ονομάζονται με ένα όνομα κλάσης `C`. Με άλλα λόγια, είναι ισοδύναμο με τον όρο `'C in typeof(x)'`. Κατ' επέκταση, το `{;C}` απλά δείχνει έναν όρο φιλτραρίσματος κόμβων (δηλ., κλάσεων) που προσδιορίζεται από ένα όνομα κλάσης `C` και λαμβάνει υπόψη και τις συνδέσεις `rdfs:SubClassOf`. Παραδείγματος χάριν, στην έκφραση `{;Painter}@P` το πεδίο ορισμού του `@P` μπορεί να είναι *Painter* ή οποιαδήποτε από τις υπερκλάσεις του και υπονοεί τον όρο φιλτραρίσματος `'Painter>= domain(@P)'`.

Για να δείξουμε την εκφραστική δύναμη των RQL ερωτήσεων σχήματος που συνδυάζονται με τη λειτουργική σημασιολογία του, ας δούμε την παρακάτω ερώτηση:

Q3: Βρες όλη την πληροφορία που σχετίζεται με την κλάση *Painter* (δηλαδή τις υπερκλάσεις της, καθώς επίσης και τις άμεσες ή κληρονομούμενες ιδιότητες).

```
seq( Painter, superclassof^(Painter), (SELECT @P, domain(@P), range(@P)
                                     FROM    {;Painter}@P)
    )
```

Για να συλλέξουμε όλες τις σχετικές πληροφορίες κατασκευάζουμε ρητά στην Q3 μια ακολουθία με τρία στοιχεία. Το πρώτο στοιχείο είναι μια σταθερά (*Painter*) που ερμηνεύεται από το σύστημα τύπων RQL ως όνομα κλάσης. Το δεύτερο στοιχείο είναι ένα σύνολο (bag) που περιέχει τα ονόματα των άμεσων υπερκλάσεων του *Painter*. Το τρίτο στοιχείο είναι ένα σύνολο ακολουθιών με τα εξής τρία στοιχεία: το πρώτο τύπου *property* και τα άλλα δύο τύπου που προκύπτει από την ένωση κλάσεων και κυριολεκτικών.

Τέλος, μια πιο πολύπλοκη ερώτηση θα μας δείξει πως μπορούμε να πάρουμε πληροφορία με πλοήγηση του σχήματος.

Q4: Ποιες ιδιότητες μπορούν να βρεθούν σε ένα βήμα από τις κλάσεις πεδία τιμών του *creates*?

```
SELECT $Y, @P, range(@P)
FROM    creates{$Y}.@P
```

$\$Y$	$@P$	$range(@P)$
Artifact	exhibited	Museum
Painting	exhibited	Museum
Sculpture	exhibited	Museum
Painting	technique	String
Sculpture	material	String

Στην Q4, ο συμβολισμός "." υπονοεί μια συνένωση (join) μεταξύ των κλάσεων πεδίο τιμών της ιδιότητας *creates* και του πεδίου ορισμού του $@P$: για κάθε όνομα κλάσης Y στο πεδίο τιμών του *creates*, ψάχνουμε όλες τις ιδιότητες των οποίων πεδίο ορισμού είναι το $\$Y$ ή υπερκλάση αυτού: $\$Y \leq domain(@P)$ και $\$Y \leq range(creates)$. Με άλλα λόγια, αυτή η συνένωση μας επιτρέπει να ακολουθήσουμε ιδιότητες που μπορούν να εφαρμοστούν σε κλάσεις πεδία τιμών του *creates* (είτε επειδή καθορίζονται άμεσα ή επειδή κληρονομούνται) σε οποιαδήποτε υποκλάση του πεδίου τιμών του *creates*.

2.2 Σχετικές εργασίες

Στην ενότητα αυτήν περιγράφονται συνοπτικά θέματα που καλύπτουν οι παρακάτω εργασίες:

- Σύστημα Rondo
- Διαχείριση ιεραρχικών σχημάτων στο Σημασιολογικό Ιστό

2.2.1 Το σύστημα RONDO

Το RONDO [2] είναι ένα πρότυπο σύστημα στο οποίο εφαρμόζονται υψηλού επιπέδου τελεστές σε μοντέλα (π.χ. σε σχεσιακά μοντέλα, αντικειμενοστρεφή, XML schemas, κ.λ.π.) καθώς και ταιριάσματα μεταξύ των μοντέλων, ως οντότητες στην ολότητά τους και όχι ως σύνολα από μεμονωμένα στοιχεία. Το RONDO στηρίζεται σε τρεις βασικές έννοιες:

1. Μοντέλα (*models*). Τα μοντέλα αναπαρίστανται ως κατευθυνόμενοι γράφοι με ετικέτες. Οι κόμβοι τους υποδηλώνουν βασικά στοιχεία αναπαράστασης, όπως τάξεις, σχέσεις, τύπους, κ.λ.π., ενώ οι ακμές υποδηλώνουν συσχετίσεις μεταξύ των βασικών στοιχείων. Κάθε μοντέλο που εισάγεται στο σύστημα RONDO θα πρέπει να μετασχηματιστεί σε έναν τέτοιο γράφο.
2. Ταιριάσματα (*morphisms*). Τα ταιριάσματα είναι 1-1 αντιστοιχίσεις μεταξύ των κόμβων σε γράφους που αναπαριστούν τα μοντέλα.
3. Επιλογείς (*selectors*). Οι επιλογείς είναι σύνολα βασικών στοιχείων από τα μοντέλα, δηλαδή κόμβων από τους αντίστοιχους γράφους.

Μοντέλα, ταιριάσματα και επιλογείς είναι οι είσοδοι (και τα αποτελέσματα) των τελεστών όπως Εξαγωγή (*Extract*), Διαγραφή (*Delete*), Ταίρι (*Match*), Συγχώνευση (*Merge*):

1. Οι τελεστές Εξαγωγή και Διαγραφή επιλέγουν ή διαγράφουν μέρη από τα μοντέλα, αντίστοιχα. Η είσοδος και για τους δύο είναι ένα μοντέλο M και ένας επιλογέας που δηλώνει ποια στοιχεία του μοντέλου θα εξαχθούν ή θα διαγραφούν, αντίστοιχα. Το αποτέλεσμα είναι ένα μοντέλο M' , μέρος του μοντέλου M .
2. Ο τελεστής Ταίρι δέχεται ως είσοδο δύο μοντέλα και επιστρέφει το ταίριασμά τους.
3. Ο τελεστής Συγχώνευση δέχεται ως είσοδο δύο μοντέλα $M1$, $M2$ και το ταίριασμά τους και επιστρέφει ένα νέο μοντέλο M , ικανό να αναπαραστήσει πληροφορία που περιέχεται και στα δύο μοντέλα $M1$, $M2$, καθώς και δύο νέα ταιριάσματα: το ένα μεταξύ του M και του $M1$, και το άλλο μεταξύ του M και του $M2$.

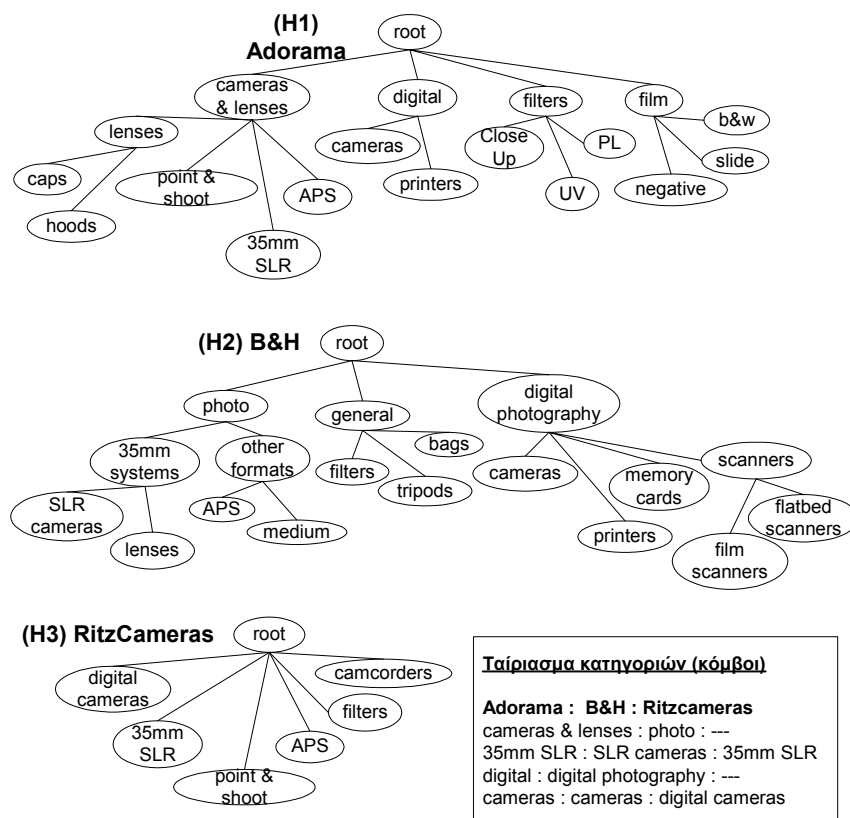
Στο σύστημα RONDO, οι παραπάνω τελεστές έχουν υλοποιηθεί για γράφους που αναπαριστούν σχεσιακά μοντέλα και σχήματα XML. Ειδικά για τον τελεστή Ταίρι, πέρα από

τον αλγόριθμο που το ίδιο το σύστημα RONDO προσφέρει, διάφοροι άλλοι αλγόριθμοι ταιριάσματος σχημάτων μπορούν να ενσωματωθούν.

2.2.2 Διαχείριση ιεραρχικών σχημάτων στο Σημασιολογικό Ιστό

Η συγκεκριμένη διδακτορική διατριβή [3] προτείνει ένα πλαίσιο διαχείρισης ιεραρχιών για τον Σημασιολογικό Ιστό. Μέχρι τώρα οι ιεραρχίες έχουν χρησιμοποιηθεί ως απλοί σημασιολογικοί οδηγοί για διάσχιση ή αναζήτηση πληροφορίας με γλώσσες όπως οι XQuery και RQL. Όμως, στο περιβάλλον του Ιστού, όπου η αναζήτηση πληροφορίας σε ένα πεδίο γνώσης απαιτεί επεξεργασία σε πολλές πηγές σχετικές με το πεδίο αυτό, προσδιορίζεται η ανάγκη της ανάδειξης των ιεραρχιών σε οντότητες πρώτης τάξης, έτσι ώστε να διαχειρίζονται ως ολοκληρωμένες οντότητες και όχι ως σύνολα από μεμονωμένους κόμβους. Παρακάτω φαίνονται οι απαιτήσεις αυτές με παραδείγματα μιας ειδικής κατηγορίας πυλών του Ιστού.

Τα Adorama, B&H και RitzCamera είναι τρεις ηλεκτρονικές αγορές φωτογραφικού εξοπλισμού. Οι δύο πρώτες παρέχουν θεματικές ιεραρχίες με κατηγορίες και υποκατηγορίες που υποστηρίζουν αναζήτηση και διάσχιση, ενώ η τρίτη παρέχει μια απλή ιεραρχία δύο επιπέδων με δυνατότητες διάσχισης.

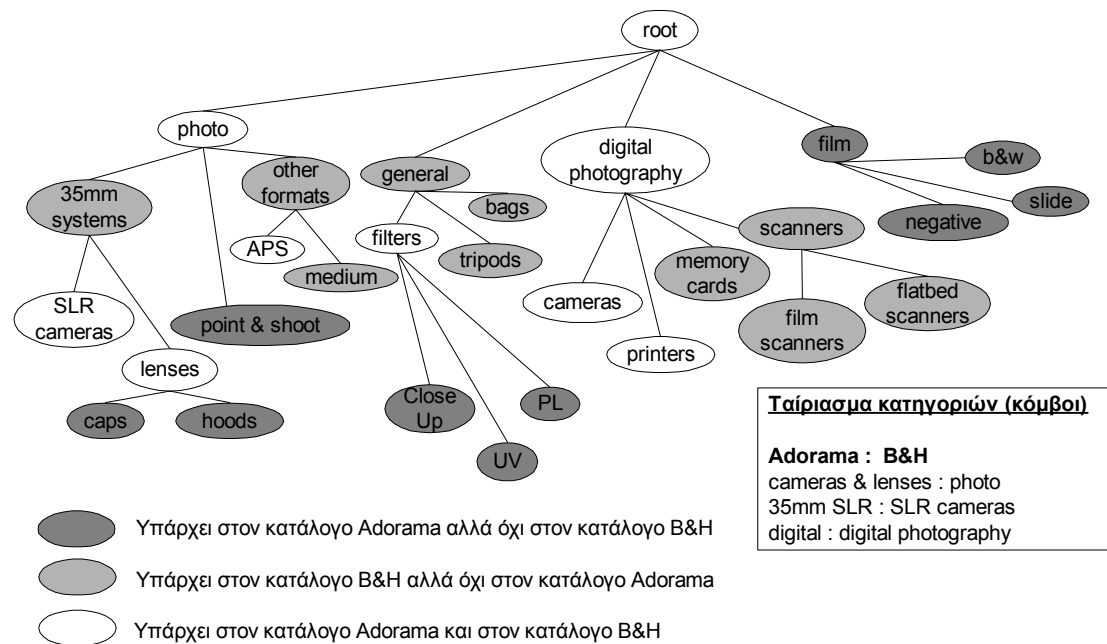


Σχήμα 2.6

Το Σχήμα 2.6 δείχνει μέρη των ιεραρχιών τους. Οι ηλεκτρονικές αυτές αγορές είναι παραδείγματα πύλων καταλόγων που παρέχουν δεδομένα οργανωμένα σε ιεραρχίες με κατηγορίες και υποκατηγορίες. Οι πύλες καταλόγων είναι συνήθως κάθετες, αφορούν δηλαδή συγκεκριμένο θεματικό πεδίο, π.χ. φωτογραφικό εξοπλισμό, και παίζουν σημαντικό ρόλο στον Ιστό, καθότι είναι κόμβοι που οργανώνουν μεγάλο όγκο δεδομένων.

Αντιμετωπίζοντας τους καταλόγους αυτούς ως ένα σύνολο από ομόλογες ιεραρχίες που οργανώνουν υλικό σχετικό με φωτογραφικό εξοπλισμό, προσδιορίζονται τρεις τελεστές:

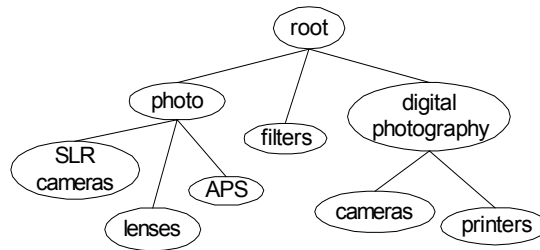
- Τελεστής με σημασιολογία ένωσης. Το Σχήμα 2.7 απεικονίζει μια ιεραρχία που συγχωνεύει τις ιεραρχίες του Adorama και του B&H. Στην καινούρια αυτή ιεραρχία υπάρχουν όλες οι κατηγορίες από το Adorama και το B&H. Οι δομικές σχέσεις της μπορεί να μην υπάρχουν σε όλες τις αρχικές ιεραρχίες. Για παράδειγμα, στην καινούρια ιεραρχία υπάρχουν οι caps και hoods για τον κόμβο lenses, μια κατηγοριοποίηση που υπάρχει μόνο στο Adorama και όχι στο B&H. Από την άλλη, η κατηγορία APS είναι κοινή και στις δύο ιεραρχίες.



Σχήμα 2.7

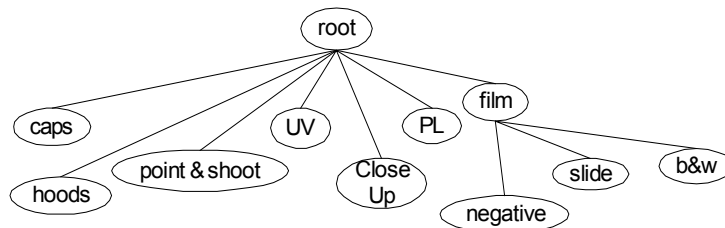
- Τελεστής με σημασιολογία τομής. Το Σχήμα 2.8 δείχνει το κοινό μέρος των ιεραρχιών για τους καταλόγους Adorama και B&H. Στην καινούρια αυτή ιεραρχία υπάρχουν κατηγορίες κοινές και στο Adorama και στο B&H. Οι δομικές σχέσεις της υπάρχουν σε όλες τις αρχικές ιεραρχίες. Για παράδειγμα, στην καινούρια ιεραρχία υπάρχουν οι APS και lenses για τον κόμβο photo, μια κατηγοριοποίηση που υπάρχει και στο Adorama (ακριβώς κάτω από τον κόμβο cameras & lenses, τον αντίστοιχο

κόμβο του photo) και στο Β&Η (ακολουθώντας το μονοπάτι από τον κόμβο photo κάτω προς τα φύλλα).



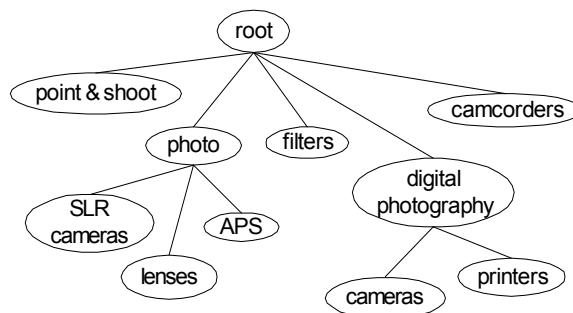
Σχήμα 2.8

- Τελεστής με σημασιολογία διαφοράς. Το Σχήμα 2.9 δείχνει το μέρος της ιεραρχίας του Adorama που δεν υπάρχει στο Β&Η. Η δομική πληροφορία του μέρους αυτού περιέχεται μόνο στην ιεραρχία του Adorama. Για παράδειγμα, η καινούρια ιεραρχία έχει τους κόμβους negative, slide και b&w για τον κόμβο film, μια κατηγοριοποίηση που υπάρχει μόνο στο Adorama.



Σχήμα 2.9

Μια ακολουθία τέτοιων τελεστών μπορεί να δώσει αποτέλεσμα όπως αυτό που φαίνεται στο Σχήμα 2.10: την ιεραρχία που παράγεται συγχωνεύοντας την ιεραρχία του καταλόγου RitzCamera με το κοινό κομμάτι των ιεραρχιών των Adorama και Β&Η.



Σχήμα 2.10

Ο στόχος αυτής της εργασίας είναι η διαχείριση ιεραρχιών με τελεστές χαμηλού επιπέδου, αντίστοιχους με αυτούς της συνολοθεωρίας. Δίνει έμφαση στην ύπαρξη συγκεκριμένων αλγεβρικών ιδιοτήτων και σημασιολογίας που να δίνουν δυνατότητες μετασχηματισμού, απλοποίησης και βελτιστοποίησης σειρών πράξεων.

2.3 Στόχος

Οι εργασίες που παρουσιάστηκαν στην προηγούμενη ενότητα παρέχουν ένα τρόπο να διαχειρίζονται πληροφορία η οποία αναπαρίσταται σε πολλά σχήματα παρόμοιας μορφής. Ας φανταστούμε, λοιπόν, ότι η πληροφορία αυτή είναι δομημένη σε RDF Schemas και ότι αναπαριστά μεταπληροφορία διαφόρων πυλών που διαχειρίζονται την ίδια θεματική ενότητα. Εμείς θέλουμε να μπορέσουμε να συγκρίνουμε τα σχήματα αυτά ώστε να μπορέσουμε να προβάλλουμε μέρος κάποιου σχήματος, να πάρουμε όλη την πληροφορία που μας παρέχουν, την κοινή πληροφορία ή την πληροφορία που μας δίνει κάποιο σχήμα και όχι κάποιο άλλο.

Ο στόχος της διπλωματικής εργασίας έγκειται στο να ορίσουμε και να υλοποιήσουμε κάποιες πράξεις πάνω σε RDF σχήματα ώστε να επιτύχουμε τα παραπάνω σενάρια. Οι πράξεις αυτές θα βασίζονται στις κύριες πράξεις της συνολοθεωρίας και είναι η ένωση, η τομή και η διαφορά. Εκτός από αυτές τις πράξεις θα υλοποιήσουμε και ένα είδος επιλογής πάνω σε ένα σχήμα που ουσιαστικά θα προβάλλει ένα κομμάτι του σχήματος αυτού.

3

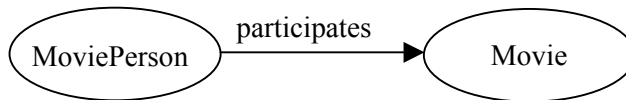
Θεωρητική μελέτη

Στο κεφάλαιο αυτό θα παρουσιάσουμε την ορολογία για τα RDF σχήματα και τους γράφους που τα αναπαριστούν. Επίσης, θα ορίσουμε πράξεις πάνω σε αυτά τα σχήματα. Οι πράξεις αυτές υλοποιούνται με βάση τους τελεστές: *ένωση*, *τομή* και *διαφορά* δύο σχημάτων. Επιπλέον, παρουσιάζεται ο τελεστής *επιλογή* πάνω σε ένα σχήμα, ο οποίος ουσιαστικά προβάλλει μέρος του σχήματος.

3.1 RDF σχήματα και γράφοι

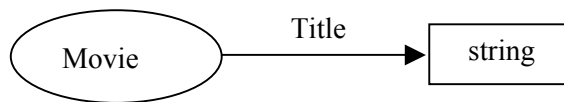
3.1.1 Ορολογία και Ορισμός RDFS

Όπως είδαμε και στο προηγούμενο κεφάλαιο, τα RDF σχήματα μπορούν πολύ εύκολα να αναπαρασταθούν σε μορφή γράφου. Κάθε *κλάση* (*class*) του RDFS απεικονίζεται ως κόμβος του γράφου και κάθε *ιδιότητα* (*property*) ως κατευθυνόμενη ακμή με ετικέτα το όνομά της. Η κλάση από την οποία ξεκινάει η ιδιότητα αποκαλείται *πεδίο ορισμού* (*domain*) της ιδιότητας αυτής και η κλάση στην οποία καταλήγει αποκαλείται *πεδίο τιμών* (*range*) της ιδιότητας αυτής. Στο Σχήμα 3.1 φαίνεται ένα παράδειγμα, όπου η ιδιότητα *participates* έχει πεδίο ορισμού την κλάση *MoviePerson* και πεδίο τιμών την κλάση *Movie*.



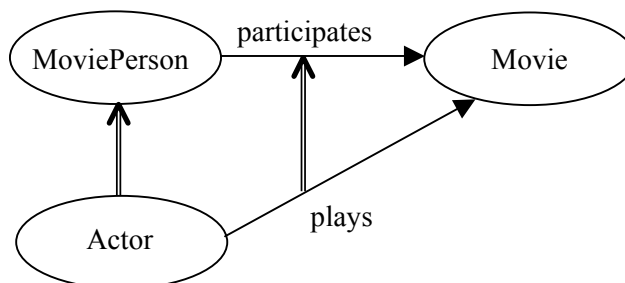
Σχήμα 3.1

Όταν τώρα το πεδίο τιμών μιας ιδιότητας είναι τύπος δεδομένων, όπως συμβολοσειρά, ακέραιος, ημερομηνία κλπ, τότε λέμε ότι αυτός ο κόμβος είναι *κυριολεκτικό (literal)* και συμβολίζεται με ορθογώνιο κόμβο πάνω στο γράφο. Μια ιδιότητα μπορεί να έχει μόνο ως πεδίο τιμών ένα κυριολεκτικό και ποτέ ως πεδίο ορισμού της. Ακολουθεί ένα παράδειγμα στο Σχήμα 3.2:



Σχήμα 3.2

Τέλος, το RDFS υποστηρίζει έναν ιεραρχικό τρόπο αναπαράστασης τόσο των κλάσεων όσο και των ιδιοτήτων. Δηλαδή, μια κλάση μπορεί να είναι υποκλάση (*subclass*) ή υπερκλάση (*superclass*) μιας άλλης καθώς και μια ιδιότητα να είναι υπο-ιδιότητα (*subproperty*) ή υπερ-ιδιότητα (*superproperty*) μιας άλλης. Για παράδειγμα, μια κλάση Actor είναι (*isA*) MoviePerson και άρα είναι υποκλάση της κλάσης αυτής, ενώ η ιδιότητα plays είναι υπο-ιδιότητα της ιδιότητας participates. Το παράδειγμα αυτό φαίνεται στο Σχήμα 3.3 όπου οι ακμές isA που δηλώνουν αυτήν την ιεραρχία φαίνονται με τόξα διπλής γραμμής.



Σχήμα 3.3

Λαμβάνοντας υπόψη μας, λοιπόν, όλα τα παραπάνω μπορούμε να δώσουμε έναν επίσημο ορισμό του RDF Schema αφού πρώτα ορίσουμε μια καλά-ορισμένη ιεραρχία κλάσεων και ιδιοτήτων.

Στον παρακάτω ορισμό, με το σύμβολο " \prec " αναπαριστούμε τη σχέση *isA* κλάσεων και ιδιοτήτων. Δηλαδή το $C_1 \prec C_2$ δηλώνει ότι η κλάση C_1 είναι υποκλάση της κλάσης C_2 και το $p_1 \prec p_2$ δηλώνει ότι η ιδιότητα p_1 είναι υποιδιότητα της p_2 .

Ορισμός 3.1: Ορίζουμε ως καλά-ορισμένη ιεραρχία κλάσεων και ιδιοτήτων τη συνάρτηση $H = (N, \prec)$, όπου N είναι το σύνολο κλάσεων και ιδιοτήτων. Η συνάρτηση αυτή είναι καλά-ορισμένη εάν το \prec δίνει τη μικρότερη μερική διάταξη τέτοια ώστε: εάν $p_1, p_2 \in P$ και $p_1 \prec p_2$, τότε το πεδίο_ορισμού(p_1) \leq πεδίο_ορισμού(p_2) και πεδίο_τιμών(p_1) \leq πεδίο_τιμών(p_2).

Ακολουθεί ο ορισμός του RDF σχήματος.

Ορισμός 3.2: Ένα RDF σχήμα είναι μια πεντάδα από στοιχεία: $R = (C, P, \psi, \lambda, H)$ όπου: C είναι το σύνολο των κόμβων, δηλαδή κλάσεων, του σχήματος και P είναι το σύνολο των ακμών, δηλαδή ιδιοτήτων, του σχήματος. Η συνάρτηση H αποτελεί μια καλά-ορισμένη ιεραρχία κλάσεων και ιδιοτήτων $H = (N, \prec)$ όπου N είναι το σύνολο κλάσεων και ιδιοτήτων ($N = C \cup P$). Η συνάρτηση λ είναι μια συνάρτηση που δίνει ετικέτες σε κόμβους και ακμές, δηλαδή $\lambda : C \cup P \rightarrow \{\text{σύνολο των στοιχείων του RDFS}\}$. Τέλος, ψ είναι μια συνάρτηση $\psi : C \rightarrow P \times P$ όπου ουσιαστικά καθορίζει το πεδίο ορισμού και το πεδίο τιμών των ιδιοτήτων παίρνοντας υπόψη τις δηλώσεις του RDFS *rdfs:domain* και *rdfs:range* αντίστοιχα.

3.1.2 Ορισμός ευρύτερου πεδίου ορισμού και τιμών

Στην ενότητα αυτή θα δώσουμε τον ορισμό της έννοιας του ευρύτερου πεδίου ορισμού και τιμών μιας ιδιότητας.

Ορισμός 3.3: Ορίζουμε ως ευρύτερο πεδίο ορισμού μιας ιδιότητας p το σύνολο των κλάσεων C που μπορούν να εφαρμοστούν στην ιδιότητα αυτή. Δηλαδή το σύνολο που δημιουργείται από την κλάση πεδίο ορισμού της ιδιότητας και τις υποκλάσεις της. Άρα θα έχουμε:

$$C = \text{domain}(p) \cup \text{subclass}(\text{domain}(p)).$$

Στον παραπάνω ορισμό, με *subclass*(C_1) δηλώνουμε όλους τους απογόνους της κλάσης C_1 .

Ορισμός 3.4: Ορίζουμε ως ευρύτερο πεδίο τιμών μιας ιδιότητας p το σύνολο των κλάσεων C από τις οποίες μπορεί να πάρει τιμή η ιδιότητα αυτή. Δηλαδή το σύνολο που δημιουργείται από την κλάση πεδίο τιμών της ιδιότητας και τις υποκλάσεις της. Άρα θα έχουμε:

$$C = \text{range}(p) \cup \text{subclass}(\text{range}(p)).$$

Στον παραπάνω ορισμό, με $\text{subclass}(C1)$ δηλώνουμε όλους τους απογόνους της κλάσης $C1$.

3.1.3 Ορισμός υποσυνόλου ενός RDFS

Παρακάτω ορίζουμε πότε ένα RDF σχήμα αποτελεί υποσύνολο ενός άλλου RDF σχήματος.

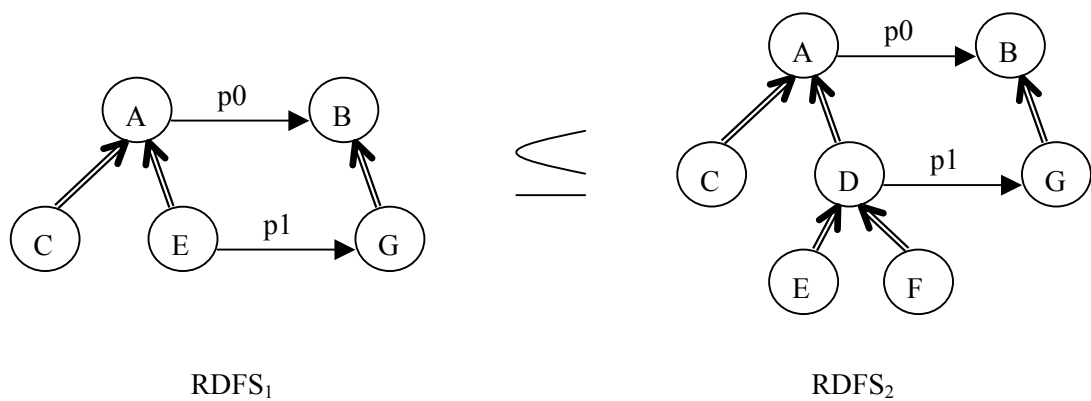
Ορισμός 3.5: Ένα RDF σχήμα $R_i = (C_i, P_i, \psi_i, \lambda_i, H_i)$ αποτελεί υποσύνολο ενός άλλου RDF σχήματος $R_j = (C_j, P_j, \psi_j, \lambda_j, H_j)$ και το συμβολίζουμε με $R_i \subseteq R_j$, εάν ισχύουν τα παρακάτω:

1. το σύνολο C_i των κλάσεων του σχήματος R_i είναι υποσύνολο του συνόλου C_j των κλάσεων του σχήματος R_j , δηλαδή $C_i \subseteq C_j$
2. το σύνολο P_i των ιδιοτήτων του R_i είναι υποσύνολο του συνόλου P_j των ιδιοτήτων του R_j , δηλαδή $P_i \subseteq P_j$
3. το ευρύτερο πεδίο ορισμού και το ευρύτερο πεδίο τιμών κάθε ιδιότητας που ανήκει στο P_i τα οποία καθορίζονται από τη συνάρτηση ψ_i ισχύουν και με βάση τη συνάρτηση ψ_j , δηλαδή $\forall p \in P_i$ εάν με βάση την ψ_i ισχύει $\text{ευρύτερο_πεδίο_ορισμού}_i(p) = D$ και $\text{ευρύτερο_πεδίο_τιμών}_i(p) = R$ τότε πρέπει να ισχύει σύμφωνα με την ψ_j $D \subseteq \text{ευρύτερο_πεδίο_ορισμού}_j(p)$ και $R \subseteq \text{ευρύτερο_πεδίο_τιμών}_j(p)$.
4. τα ονόματα των κλάσεων και των ιδιοτήτων που ορίζει η συνάρτηση λ_i είναι ίδια με τα ονόματα των αντίστοιχων κλάσεων και ιδιοτήτων που ορίζει η συνάρτηση λ_j .
5. η ιεραρχία που ορίζει η συνάρτηση H_i αποτελεί έγκυρη ιεραρχία σύμφωνα με την ιεραρχία που ορίζει η συνάρτηση H_j . Δηλαδή,

$\forall x_1, x_2 \in N_i$ εάν ισχύει $x_1 \prec x_2$ με βάση τη συνάρτηση H_i τότε ισχύει το ίδιο ($x_1 \prec x_2$) και με βάση τη συνάρτηση H_j και

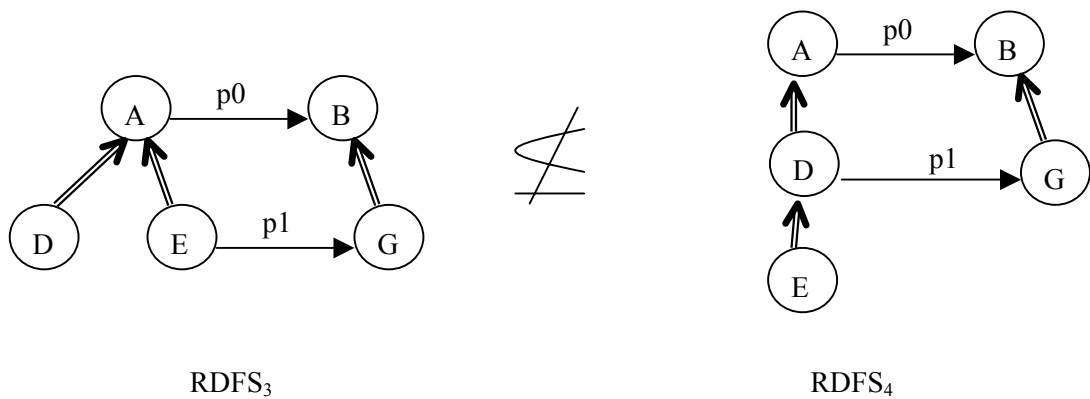
$\forall x_1, x_2 \in N_i$ εάν ισχύει $x_1 \prec x_2$ με βάση τη συνάρτηση H_j τότε ισχύει το ίδιο και με βάση τη συνάρτηση H_i , όπου $N_i = C_i \cup P_i$.

Στο Σχήμα 3.4 φαίνεται ένα παράδειγμα, όπου το RDFS_1 αποτελεί υποσύνολο του RDFS_2 .



Σχήμα 3.4

Αντίθετα, στο Σχήμα 3.5 φαίνεται ένα παράδειγμα όπου το RDFS₃ δεν είναι υποσύνολο του RDFS₄. Οι κλάσεις D και E παρόλο που είναι υποκλάσεις της A, σύμφωνα με το RDFS₄ η D είναι υπερκλάση της E πράγμα το οποίο δε φαίνεται στο RDFS₃. Έτσι, παραβιάζεται η 5^η συνθήκη του ορισμού.



Σχήμα 3.5

3.1.4 Καθολικό σχήμα

Στα πλαίσια της διπλωματικής αυτής υποθέτουμε τη χρήση της έννοιας του καθολικού σχήματος (*global schema*). Όλα τα επιμέρους σχήματα που θα χρησιμοποιούμε για να εφαρμόσουμε τους τελεστές θα είναι υποσύνολα (σύμφωνα με τον ορισμό που δώσαμε παραπάνω) ενός καθολικού σχήματος. Επίσης, τα αποτελέσματα που προκύπτουν από τις πράξεις θα είναι υποσύνολα του ίδιου καθολικού σχήματος.

Με την έννοια αυτή, υποθέτουμε ένα RDF σχήμα ως το καθολικό και με βάση το οποίο εκτελούμε όλες τους τελεστές που θέλουμε.

3.2 Τελεστές

3.2.1 Τελεστής επιλογή

Παρακάτω ορίζουμε έναν ειδικό τελεστή *επιλογή* ο οποίος παίρνει ως είσοδο ένα σύνολο κλάσεων από το καθολικό σχήμα και βγάζει ως έξοδο το ελάχιστο σε κλάσεις υποσύνολο του σχήματος αυτού. Το τελικό αποτέλεσμα είναι ένα RDF σχήμα, το οποίο περιέχει όλες τις κλάσεις εισόδου, και αποτελεί έγκυρο υποσύνολο του καθολικού σχήματος.

Ορισμός 3.6: Έστω ότι C είναι το σύνολο κλάσεων που έχουν επιλεγεί από ένα RDF σχήμα. Ορίζουμε τον τελεστή *επιλογή* να δημιουργεί ένα νέο RDF σχήμα σύμφωνα με τον παρακάτω αλγόριθμο:

Αλγόριθμος:

Για κάθε κλάση μέσα στο C

 Πάρε όλες τις ιδιότητες που έχουν ως ευρύτερο πεδίο ορισμού την κλάση αυτή

 Για κάθε ιδιότητα της κλάσης αυτής

 Πάρε το ευρύτερο πεδίο τιμών της ιδιότητας αυτής

 Κράτα μόνο τις κλάσεις του πεδίου τιμών που ανήκουν στο C

Για κάθε ιδιότητα που βρήκαμε παραπάνω

 /* Εύρεση μοναδικής κλάσης του πεδίου ορισμού της ιδιότητας */

 Εάν υπάρχει μόνο μία κλάση στο πεδίο ορισμού τότε κρατάμε αυτήν την κλάση

 Εάν υπάρχουν πάνω από μία κλάσεις τότε

 Ελεγχος αν οι κλάσεις αυτές ανήκουν σε ένα μονοπάτι ιεραρχίας:

 Εάν ναι τότε κρατάμε την κλάση που βρίσκεται πιο ψηλά στην ιεραρχία

 Εάν όχι τότε βρίσκουμε τον πιο κοντινό πρόγονο κλάση που ενώνει τις κλάσεις αυτές, κρατάμε αυτήν ως πεδίο ορισμού της ιδιότητας και την προσθέτουμε στο C

 /* Εύρεση μοναδικής κλάσης του πεδίου τιμών της ιδιότητας */

 Εάν υπάρχει μόνο μία κλάση στο πεδίο τιμών τότε κρατάμε αυτήν την κλάση

Εάν υπάρχουν πάνω από μία κλάσεις τότε

Έλεγχος αν οι κλάσεις αυτές ανήκουν σε ένα μονοπάτι ιεραρχίας:

Εάν ναι τότε κρατάμε την κλάση που βρίσκεται πιο ψηλά στην ιεραρχία

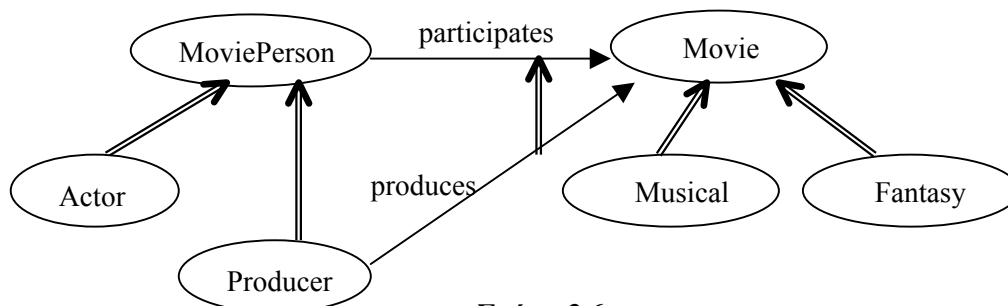
Εάν όχι τότε βρίσκουμε τον πιο κοντινό πρόγονο κλάση που ενώνει τις κλάσεις αυτές, κρατάμε αυτήν ως πεδίο τιμών της ιδιότητας και την προσθέτουμε στο C

Ακολουθεί ένα παράδειγμα του τελεστή της επιλογής.

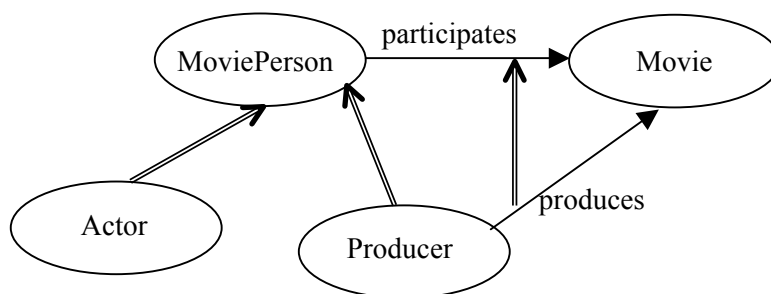
Έστω ότι έχω ένα RDF σχήμα όπως φαίνεται στο Σχήμα 3.6 και επιλέγω από αυτό τις κλάσεις Actor, Producer και Movie. Τότε σύμφωνα με τον αλγόριθμο θα έχω:

- Actor
ιδιότητες = {participates}
 - ο participates
ευρύτερο πεδίο τιμών = {Movie, Musical, Fantasy}
κρατώ μόνο το Movie
- Producer
ιδιότητες = {produces, participates}
 - ο produces
ευρύτερο πεδίο τιμών = {Movie, Musical, Fantasy}
κρατώ μόνο το Movie
 - ο participates
ευρύτερο πεδίο τιμών = {Movie, Musical, Fantasy}
κρατώ μόνο το Movie
- Movie
ιδιότητες = { }
- participates
ευρύτερο πεδίο ορισμού = {Actor, Producer}
οι κλάσεις δεν ανήκουν στο ίδιο μονοπάτι ιεραρχίας άρα παίρνουμε τον κοντινότερο πρόγονο που είναι η κλάση MoviePerson και την προσθέτουμε στο C.
ευρύτερο πεδίο τιμών = {Movie}
μια μόνο κλάση άρα κρατάμε αυτήν.
- produces
ευρύτερο πεδίο ορισμού = {Actor, Producer, MoviePerson}
μια ιεραρχία κρατάμε την κλάση MoviePerson.
ευρύτερο πεδίο τιμών = {Movie}
μια μόνο κλάση άρα κρατάμε αυτήν.

Το RDF σχήμα που προκύπτει σύμφωνα με τον παραπάνω αλγόριθμο φαίνεται στο Σχήμα 3.7.



Σχήμα 3.6



Σχήμα 3.7

Όπως φαίνεται από το παραπάνω σχήμα του αποτελέσματος, εκτός από τις τρεις κλάσεις που διαλέξαμε στο αποτέλεσμα συμπεριλαμβάνεται άλλη μία, η *MoviePerson*. Αυτό έγινε διότι το ευρύτερο πεδίο ορισμού της ιδιότητας *participates* είναι και η κλάση *Actor* και η κλάση *Producer* και σύμφωνα με τον αλγόριθμο, αφού δεν ανήκουν στο ίδιο μονοπάτι ιεραρχίας, προσθέτουμε τον κοντινότερο κοινό πρόγονο, δηλαδή την κλάση *MoviePerson*. Ο αλγόριθμος αυτός υλοποιήθηκε με αυτόν τον τρόπο για δύο λόγους. Πρώτον, σύμφωνα με τους κανονισμούς της RQL δεν επιτρέπεται να υπάρχουν πάνω από μία κλάση είτε ως πεδίο ορισμού είτε ως πεδίο τιμών μιας ιδιότητας. Δεύτερον, όπως έχουμε αναφέρει σε προηγούμενο κεφάλαιο, όταν κάποια ιδιότητα έχει πάνω από μία κλάση ως πεδίο ορισμού σημαίνει ότι τα στιγμιότυπα που ικανοποιούν την ιδιότητα αυτή πρέπει να είναι στιγμιότυπα όλων των κλάσεων του πεδίου ορισμού της πράγμα το οποίο δεν θέλουμε να συμβαίνει. Δηλαδή θα έπρεπε ένα στιγμιότυπο που ικανοποιεί την ιδιότητα *participates* να είναι και *Actor* και *Producer*, το οποίο δεν ισχύει.

Τέλος, παρατηρούμε ότι το αποτέλεσμα είναι υποσύνολο του αρχικού (σύμφωνα με τον ορισμό 3.5) αφού οποιαδήποτε πληροφορία την παίρνουμε από το αρχικό σχήμα. Πρώτον, το σύνολο κλάσεων και ιδιοτήτων το παίρνουμε από το αρχικό και μάλιστα με τα ίδια ονόματα (συνθήκες 1,2,4) και τα πεδία ορισμού και τιμών είναι επίσης σύμφωνα με το αρχικό σχήμα (συνθήκη 3). Τέλος, η ιεραρχία που προκύπτει στο τελικό σχήμα είναι έγκυρη με βάση την ιεραρχία του αρχικού σχήματος (συνθήκη 5).

3.2.2 Τελεστής ένωση

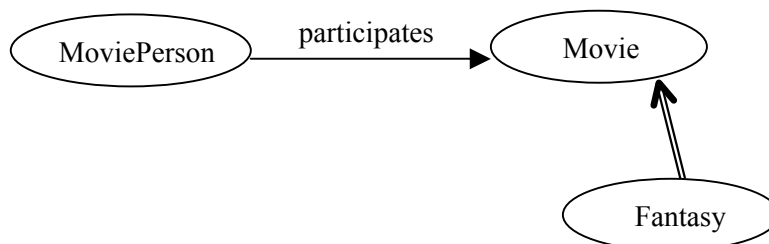
Παρακάτω ορίζουμε τον τελεστή *ένωση* πάνω σε δύο σχήματα. Ο τελεστής αυτός παίρνει ως είσοδο δύο σχήματα RDF, τα οποία είναι υποσύνολα του ίδιου καθολικού σχήματος, και βγάζει ως έξοδο ένα RDF σχήμα, του οποίου το σύνολο των κλάσεων περιέχει την ένωση των συνόλων των κλάσεων των δύο σχημάτων, και το οποίο αποτελεί υποσύνολο του καθολικού σχήματος.

Ορισμός 3.7: Έστω δύο RDF σχήματα R_1 και R_2 με σύνολα κλάσεων C_1 και C_2 αντίστοιχα, τα οποία είναι υποσύνολα ενός καθολικού σχήματος R_G , δηλαδή ισχύει $R_1 \subseteq R_G$ και $R_2 \subseteq R_G$. Ορίζουμε τον τελεστή ένωσης ως το RDF σχήμα που προκύπτει αφού εφαρμόσουμε τον τελεστή επιλογή στο σύνολο της ένωσης C_U των κλάσεων των δύο σχημάτων, με βάση του καθολικού σχήματος R_G .

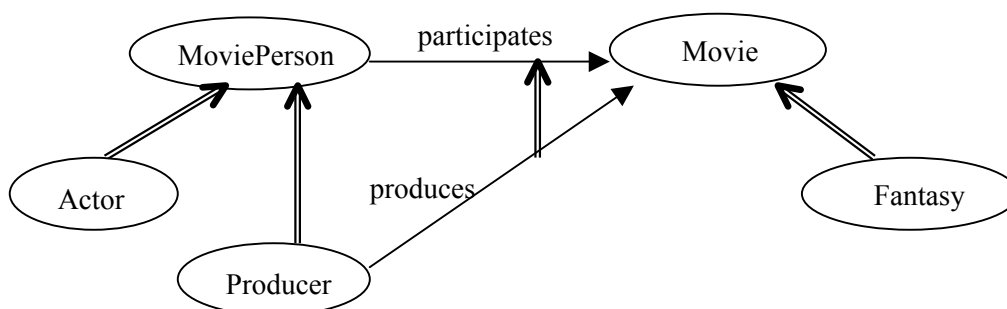
Στο τελικό αποτέλεσμα θα ισχύει:

1. $(C_1 \cup C_2) \subseteq C_U$
2. $R_U \subseteq R_G$

Έστω για παράδειγμα ότι θεωρούμε ως καθολικό σχήμα το RDFS που φαίνεται στο Σχήμα 3.6 και δύο υποσύνολα του τα R_1 και R_2 που φαίνονται στα Σχήματα 3.7 και 3.8 αντίστοιχα. Έτσι, το σύνολο της ένωσης των κλάσεων αυτών των δύο σχημάτων είναι $\{Actor, Producer, MoviePerson, Movie, Fantasy\}$. Τότε το αποτέλεσμα της ένωσης των δύο αυτών σχημάτων εφαρμόζοντας τον αλγόριθμο της επιλογής σε αυτό το σύνολο απεικονίζεται στο Σχήμα 3.9. Το αποτέλεσμα αυτό είναι υποσύνολο του καθολικού σχήματος.



Σχήμα 3.8



Σχήμα 3.9

3.2.3 Τελεστής τομή

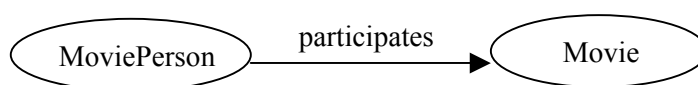
Παρακάτω ορίζουμε τον τελεστή *τομή* πάνω σε δύο σχήματα. Ο τελεστής αυτός παίρνει ως είσοδο δύο σχήματα RDF, τα οποία είναι υποσύνολα του ίδιου καθολικού σχήματος, και βγάζει ως έξοδο ένα RDF σχήμα, του οποίου το σύνολο των κλάσεων περιέχει την τομή των συνόλων των κλάσεων των δύο σχημάτων, και το οποίο αποτελεί υποσύνολο του καθολικού σχήματος.

Ορισμός 3.8: Έστω δύο RDF σχήματα R_1 και R_2 με σύνολα κλάσεων C_1 και C_2 αντίστοιχα, τα οποία είναι υποσύνολα ενός καθολικού σχήματος R_G , δηλαδή ισχύει $R_1 \subseteq R_G$ και $R_2 \subseteq R_G$. Ορίζουμε τον τελεστή *τομή* ως το RDF σχήμα που προκύπτει αφού εφαρμόσουμε τον τελεστή *επιλογή* στο σύνολο της τομής C_{\cap} των κλάσεων των δύο σχημάτων, με βάση του καθολικού σχήματος R_G .

Στο τελικό αποτέλεσμα θα ισχύει:

1. $(C_1 \cap C_2) \subseteq C_{\cap}$
2. $R_{\cap} \subseteq R_G$

Έστω για παράδειγμα ότι θεωρούμε ως καθολικό σχήμα το σχήμα που φαίνεται στο Σχήμα 3.6 και δύο υποσύνολα του τα R_1 και R_2 που φαίνονται στα Σχήματα 3.7 και 3.8 αντίστοιχα. Έτσι, το σύνολο της τομής των κλάσεων αυτών των σχημάτων είναι $\{\text{MoviePerson}, \text{Movie}\}$. Τότε το αποτέλεσμα της τομής των δύο αυτών σχημάτων εφαρμόζοντας τον αλγόριθμο της επιλογής σε αυτό το σύνολο απεικονίζεται στο Σχήμα 3.10. Το αποτέλεσμα αυτό είναι υποσύνολο του καθολικού σχήματος.



Σχήμα 3.10

3.2.4 Τελεστής διαφορά

Παρακάτω ορίζουμε τον τελεστή *διαφορά* μεταξύ δύο σχημάτων. Ο τελεστής αυτός παίρνει ως είσοδο δύο σχήματα RDF, τα οποία είναι υποσύνολα του ίδιου καθολικού σχήματος, και βγάζει ως έξοδο ένα RDF σχήμα, του οποίου το σύνολο των κλάσεων περιέχει τη διαφορά των

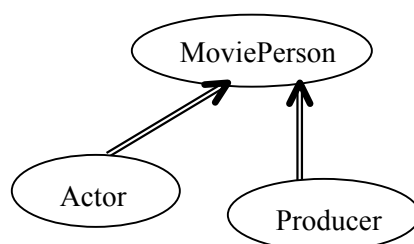
συνόλων των κλάσεων των δύο σχημάτων, και το οποίο αποτελεί υποσύνολο του καθολικού σχήματος.

Ορισμός 3.9: Έστω δύο RDF σχήματα R_1 και R_2 με σύνολα κλάσεων C_1 και C_2 αντίστοιχα, τα οποία είναι υποσύνολα ενός καθολικού σχήματος R_G , δηλαδή ισχύει $R_1 \subseteq R_G$ και $R_2 \subseteq R_G$. Ορίζουμε τον τελεστή διαφορά ως το RDF σχήμα που προκύπτει αφού εφαρμόσουμε τον τελεστή επιλογή στο σύνολο της τομής C_D των κλάσεων των δύο σχημάτων, με βάση του καθολικού σχήματος R_G .

Στο τελικό αποτέλεσμα θα ισχύει:

1. $(C_1 - C_2) \subseteq C_D$
2. $R_D \subseteq R_G$

Έστω για παράδειγμα ότι θεωρούμε ως καθολικό σχήμα το σχήμα που φαίνεται στο Σχήμα 3.6 και δύο υποσύνολα του τα R_1 και R_2 που φαίνονται στα Σχήματα 3.7 και 3.8 αντίστοιχα. Έτσι, το σύνολο της διαφοράς των κλάσεων αυτών των σχημάτων είναι $\{Actor, Producer\}$. Τότε το αποτέλεσμα της διαφοράς των δύο αυτών σχημάτων εφαρμόζοντας τον αλγόριθμο της επιλογής σε αυτό το σύνολο απεικονίζεται στο Σχήμα 3.11. Όπως βλέπουμε πάλι, η κλάση `MoviePerson` προστέθηκε στο τελικό αποτέλεσμα, παρόλο που το σύνολο της διαφοράς των κλάσεων δεν την περιείχε. Αυτό οφείλεται στον αλγόριθμο του τελεστή επιλογή που περιγράψαμε παραπάνω. Το αποτέλεσμα αυτό είναι υποσύνολο του καθολικού σχήματος.



Σχήμα 3.11

4

Ανάλυση και σχεδίαση

Στο κεφάλαιο αυτό παρουσιάζεται η μελέτη που έγινε για την υλοποίηση του συστήματος. Αρχικά φαίνεται ο διαχωρισμός του συστήματος σε επιμέρους υποσυστήματα και μια σύντομη περιγραφή του καθενός. Τέλος, στη σχεδίαση του συστήματος, παρουσιάζονται οι εφαρμογές του.

4.1 Ανάλυση – περιγραφή αρχιτεκτονικής

Στην ενότητα αυτή παρουσιάζεται η ανάλυση του συστήματος και ο διαχωρισμός του σε επιμέρους υποσυστήματα όσο αναφορά την αρχιτεκτονική.

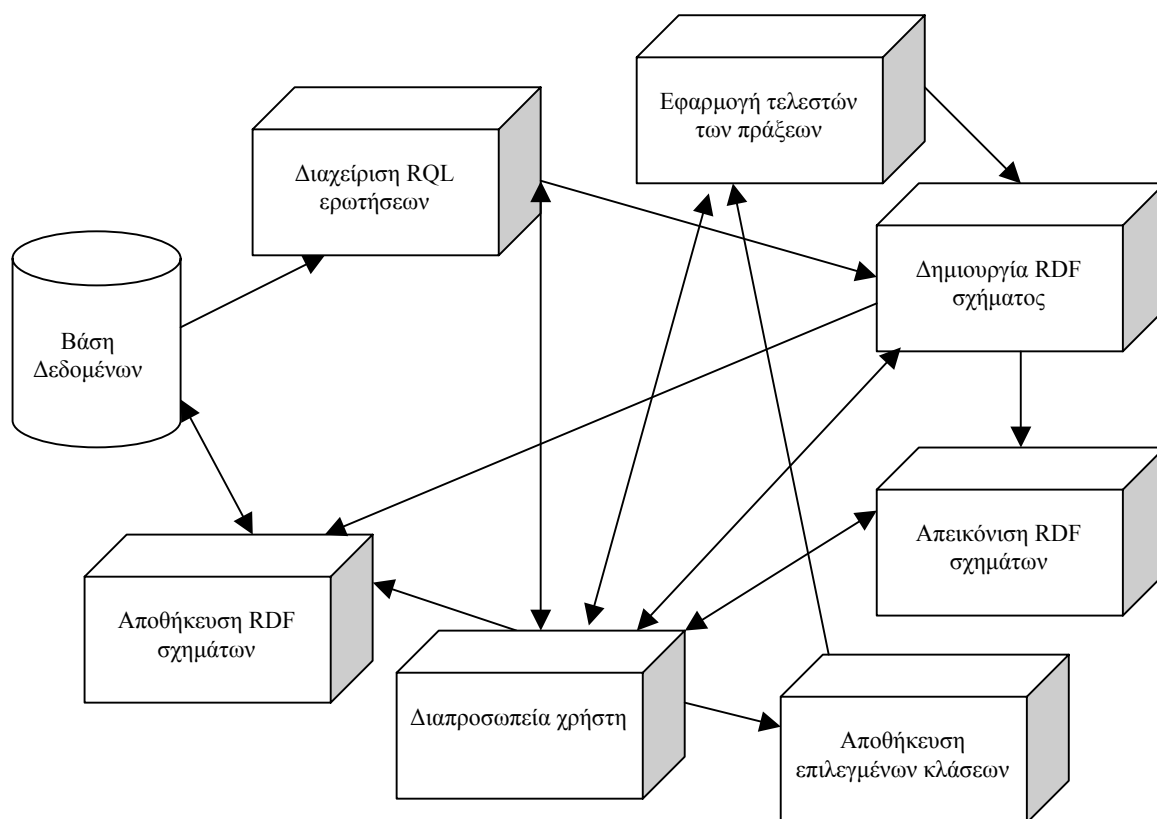
4.1.1 Διαχωρισμός υποσυστημάτων

Το σύστημα αποτελείται από τα εξής υποσυστήματα:

- Υποσύστημα διαχείρισης RQL ερωτήσεων
- Υποσύστημα αποθήκευσης επιλεγμένων κλάσεων
- Υποσύστημα εφαρμογής των τελεστών των πράξεων
- Υποσύστημα δημιουργίας RDF σχήματος
- Υποσύστημα αποθήκευσης σχημάτων στη βάση

- Υποσύστημα απεικόνισης RDF σχημάτων
- Υποσύστημα διαπροσωπείας του χρήστη

Το Σχήμα 4.1 απεικονίζει την αρχιτεκτονική του συστήματος, όπου φαίνονται τα υποσυστήματα που αναφέραμε και η επικοινωνία μεταξύ τους. Μέσω της διαπροσωπείας χρήστη γίνεται η επιλογή των επιθυμητών σχημάτων, της επιθυμητής πράξης και το σύνολο των κλάσεων που θέλουμε να πάρει μέρος στην πράξη αυτή. Όλα αυτά αποθηκεύονται στη μνήμη μέσω του υποσυστήματος αποθήκευσης επιλεγμένων κλάσεων και δίνονται στο υποσύστημα εφαρμογής των τελεστών. Στη συνέχεια, με τη βοήθεια δεδομένων που ανακτούμε μέσω RQL ερωτήσεων στη βάση δημιουργούμε το τελικό RDF αρχείο. Κατόπιν, έχουμε δυνατότητα απεικόνισής του σε μορφή γράφου, καθώς και αποθήκευσής του στη συγκεκριμένη βάση δεδομένων. Τέλος, υπάρχει και η δυνατότητα απεικόνισης ήδη αποθηκευμένων RDF σχημάτων σε μορφή γράφου ή σε μορφή αρχείου καθώς και η γραφική απεικόνιση σχημάτων που προκύπτουν σε ενδιάμεσα στάδια πράξεων απευθείας μέσω της διαπροσωπείας χρήστη.



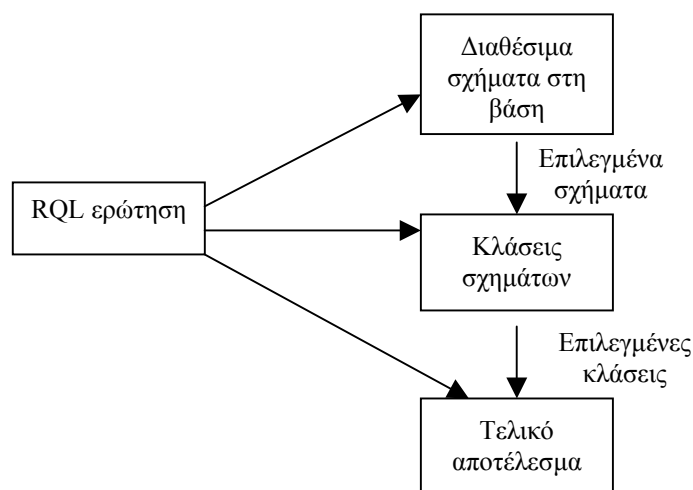
Σχήμα 4.1

4.1.2 Περιγραφή υποσυστημάτων

Παρακάτω δίνεται λεπτομερέστερη περιγραφή για καθένα από τα υποσυστήματα που αναφέραμε προηγουμένως.

4.1.2.1 Υποσύστημα διαχείρισης RQL ερωτήσεων

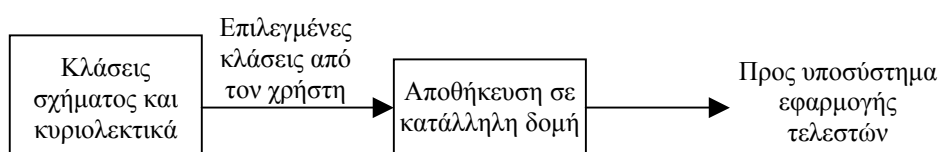
Το υποσύστημα αυτό είναι υπεύθυνο για την διεξαγωγή RQL ερωτήσεων στη βάση δεδομένων. Αυτό είναι απαραίτητο για την ανάκτηση πληροφορίας είτε για κάποιο σχήμα αποθηκευμένο στη βάση είτε για το καθολικό σχήμα. Αρχικά, όταν ο χρήστης επιλέξει ότι θέλει να κάνει κάποια πράξη τότε με τη βοήθεια του διαχειριστή RQL ερωτήσεων εμφανίζονται τα διαθέσιμα σχήματα της βάσης. Αφού επιλέξει κάποια σχήματα πάνω στα οποία θέλει να εφαρμόσει πράξεις, τότε χρησιμοποιώντας αυτό το υποσύστημα παίρνουμε την πληροφορία που χρειάζεται για αυτά τα σχήματα (όπως ποιες είναι οι κλάσεις του). Ακόμα, για να δημιουργήσουμε το τελικό RDF σχήμα χρειαζόμαστε πληροφορία για το καθολικό σχήμα, ώστε να συνδέσουμε κατάλληλα τις επιλεγμένες κλάσεις. Το υποσύστημα αυτό επικοινωνεί με τη βάση δεδομένων για να πάρει την πληροφορία, καθώς και με τα υποσυστήματα εφαρμογής τελεστών των πράξεων και δημιουργίας του RDF σχήματος για να δώσει την πληροφορία που αυτά χρειάζονται. Ακόμα επικοινωνεί με τη διαπροσωπεία για να γίνουν γνωστά στο χρήστη τα διαθέσιμα σχήματα καθώς και οι κλάσεις που περιέχουν τα σχήματα που θα επιλέξει. Στο Σχήμα 4.2 φαίνεται το διάγραμμα του υποσυστήματος αυτού.



Σχήμα 4.2

4.1.2.2 Υποσύστημα αποθήκευσης επιλεγμένων κλάσεων

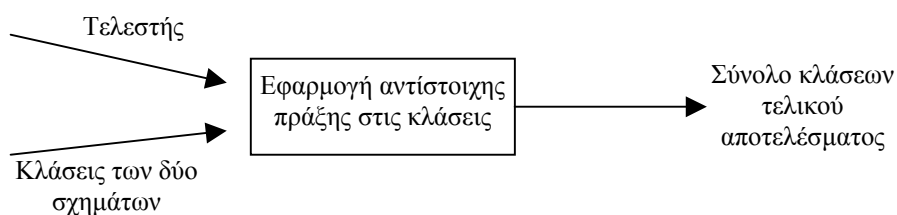
Το υποσύστημα αυτό αποθηκεύει τις κλάσεις που έχει επιλέξει ο χρήστης για να συμμετάσχουν στην πράξη. Για κάθε κλάση ενός σχήματος εμφανίζονται τυχόν ιδιότητες που ξεκινούν από την κλάση αυτή και καταλήγουν σε κυριολεκτικό. Όταν ο χρήστης επιλέξει, μέσω της διαπροσωπείας, ποιες κλάσεις και ποια κυριολεκτικά από αυτές τις κλάσεις επιθυμεί, η διαπροσωπεία επικοινωνεί με το υποσύστημα αποθήκευσης επιλεγμένων κλάσεων για την αποθήκευση σε μια κατάλληλη δομή των κλάσεων και των κυριολεκτικών αυτών. Τότε το υποσύστημα αυτό δίνει την αποθηκευμένη αυτή δομή στο σύστημα εφαρμογής τελεστών. Στο Σχήμα 4.3 φαίνεται το διάγραμμα ροής του υποσυστήματος αυτού.



Σχήμα 4.3

4.1.2.3 Υποσύστημα εφαρμογής των τελεστών των πράξεων

Το υποσύστημα αυτό είναι υπεύθυνο για την εφαρμογή ενός από τους τελεστές ανάλογα με την επιθυμία του χρήστη. Οι τελεστές αυτοί είναι οι γνωστοί από τη συνολοθεωρία τελεστές, ένωση, τομή και διαφορά. Διαβάζοντας την επιθυμητή πράξη αυτό πράττει ανάλογα με την πράξη αυτή και βγάζει κάποια αποτελέσματα που χρειάζονται για τη δημιουργία του τελικού σχήματος. Το υποσύστημα αυτό επικοινωνεί με το υποσύστημα αποθήκευσης επιλεγμένων κλάσεων για να πάρει τον επιθυμητό τελεστή και τις επιλεγμένες κλάσεις των σχημάτων, τα οποία θα συμμετάσχουν στην πράξη αυτή. Κατόπιν, εφαρμόζεται η κατάλληλη πράξη και επιστρέφεται το αποτέλεσμα, δηλαδή το σύνολο των κλάσεων που θα περιέχει το σχήμα του αποτελέσματος, στο υποσύστημα δημιουργίας RDF σχήματος για τη δημιουργία του τελικού αποτελέσματος ή στη διαπροσωπεία για τυχόν συνέχιση των πράξεων. Στο Σχήμα 4.4 φαίνεται το διάγραμμα ροής της διαδικασίας που ακολουθεί το υποσύστημα αυτό.

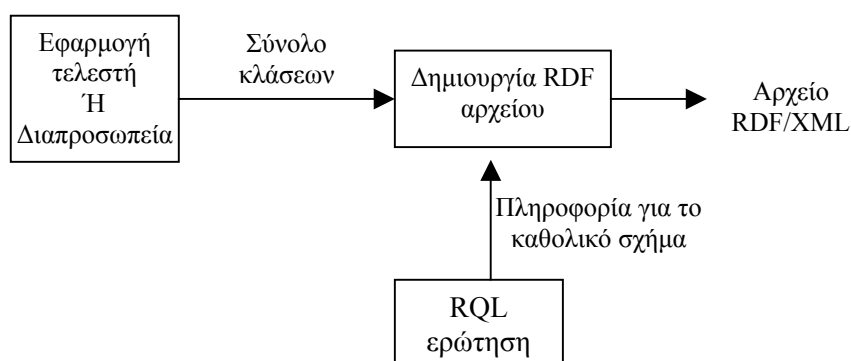


Σχήμα 4.4

4.1.2.4 Υποσύστημα δημιουργίας RDF σχήματος

Το υποσύστημα αυτό αποτελεί το βασικότερο μέρος του συστήματος. Με αυτό δημιουργείται το τελικό RDF σχήμα και άρα και αρχείο. Το υποσύστημα αυτό, εφόσον ο χρήστης έχει επιλέξει πάνω από ένα σχήματα για κάποια πράξη, επικοινωνεί με το υποσύστημα εφαρμογής των τελεστών για να πάρει τα αποτελέσματα και κατόπιν, αφού πάρει πληροφορία για το καθολικό σχήμα μέσω του υποσυστήματος διαχείρισης RQL ερωτήσεων, δημιουργεί το τελικό σχήμα. Το σχήμα αυτό, ύστερα από επιθυμία του χρήστη, το μεταδίδει στα υποσυστήματα απεικόνισης RDF σχήματος και αποθήκευσης RDF σχήματος για την απεικόνισή του σε μορφή γράφου και την αποθήκευσή του στη βάση δεδομένων αντίστοιχα.

Εάν ο χρήστης έχει επιλέξει ένα μόνο σχήμα το υποσύστημα δε χρειάζεται να επικοινωνήσει με το υποσύστημα εφαρμογής τελεστών αφού δεν μπορεί να εφαρμοστεί κάποιος τελεστής σε ένα σχήμα. Σε αυτήν την περίπτωση για να πάρει το σύνολο των κλάσεων επικοινωνεί με τη διαπροσωπεία, όπου ο χρήστης έχει επιλέξει τις κλάσεις που θέλει. Στη συνέχεια, όταν δημιουργηθεί το τελικό σχήμα, το υποσύστημα δημιουργίας σχήματος, επικοινωνεί, ύστερα από επιθυμία του χρήστη, με τα συστήματα απεικόνισης και αποθήκευσης RDF σχήματος. Τέλος, πρέπει να αναφέρουμε ότι το υποσύστημα αυτό επικοινωνεί με τη διαπροσωπεία χρήστη και στην περίπτωση απεικόνισης κάποιου σχήματος που έχει προκύψει από ενδιάμεσο στάδιο πολλών πράξεων. Τότε, το σύνολο των κλάσεων αυτού του ενδιάμεσου σχήματος αποτελεί την είσοδο στο υποσύστημα δημιουργίας RDF σχήματος και με πληροφορία πάλι από το καθολικό σχήμα είναι δυνατή η απεικόνιση του σε μορφή γράφου. Παρακάτω φαίνεται το διάγραμμα ροής του υποσυστήματος αυτού.

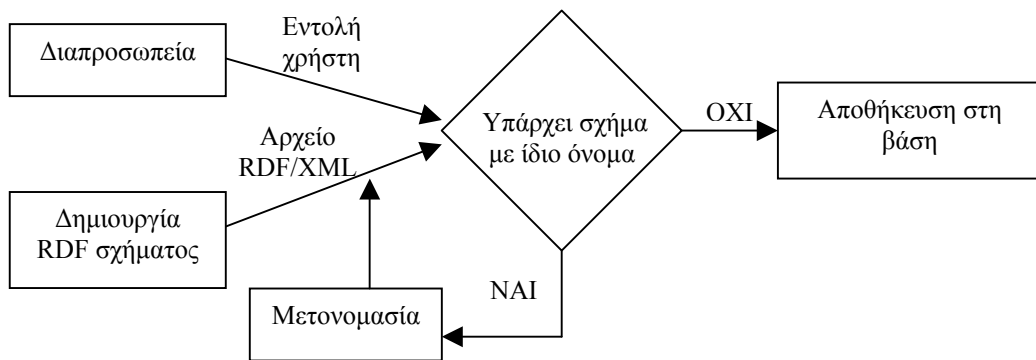


Σχήμα 4.5

4.1.2.5 Υποσύστημα αποθήκευσης RDF σχήματος

Το υποσύστημα αυτό είναι υπεύθυνο για την αποθήκευση ενός RDF σχήματος, το οποίο έχει προκύψει μετά από μια σειρά πράξεων, στη βάση δεδομένων. Αυτό παίρνει το RDF αρχείο

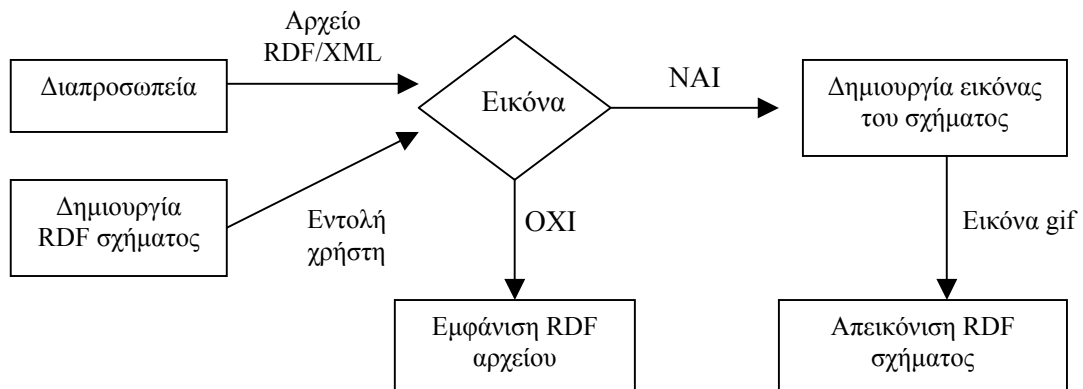
και εφόσον δεν υπάρχει σχήμα με το ίδιο όνομα στη βάση το αποθηκεύει. Έτσι, το υποσύστημα επικοινωνεί με τη βάση για έλεγχο τυχούσας σύγκρουσης ονόματος και για την αποθήκευση του σχήματος αυτού. Επίσης, επικοινωνεί με το υποσύστημα διαπροσωπείας χρήστη, όπου ο χρήστης καθορίζει εάν θέλει να το αποθηκεύσει ή όχι. Τέλος, επικοινωνεί με το υποσύστημα δημιουργίας RDF σχήματος ώστε να πάρει το τελικό αρχείο και να το αποθηκεύσει. Στο Σχήμα 4.6 απεικονίζεται το διάγραμμα ροής του υποσυστήματος αυτού.



Σχήμα 4.6

4.1.2.6 Υποσύστημα απεικόνισης RDF σχήματος

Το υποσύστημα αυτό είναι υπεύθυνο για την απεικόνιση ενός RDF σχήματος είτε σε μορφή γράφου είτε σε μορφή αρχείου. Για την απεικόνιση του τελικού σχήματος σε μορφή γράφου, παίρνει το RDF αρχείο από το υποσύστημα δημιουργίας RDF σχήματος και το απεικονίζει κατόπιν επιθυμίας του χρήστη μέσω της διαπροσωπείας. Επίσης, το σύστημα παρέχει τη δυνατότητα απεικόνισης σχημάτων ήδη αποθηκευμένων στη βάση ή σχημάτων που προκύπτουν από ενδιάμεσα στάδια πράξεων του χρήστη. Αυτό γίνεται δυνατό μέσω της επικοινωνίας του υποσυστήματος με το υποσύστημα διαπροσωπείας χρήστη. Στο Σχήμα 4.7 φαίνεται το διάγραμμα ροής του υποσυστήματος αυτού.



Σχήμα 4.7

4.1.2.7 Υποσύστημα διαπροσωπείας χρήστη

Το υποσύστημα αυτό μαζί με το υποσύστημα απεικόνισης RDF σχήματος αποτελούν τη διεπαφή όλου του συστήματος. Μέσω της διαπροσωπείας είναι δυνατή η επικοινωνία με το χρήστη. Μόλις ο χρήστης επιλέξει να κάνει μια ερώτηση, το υποσύστημα διαχείρισης RQL ερωτήσεων φέρνει όλα τα σχήματα που υπάρχουν στη βάση και η διαπροσωπεία τα εμφανίζει στον χρήστη. Κατόπιν, ο χρήστης διαλέγει τα σχήματα που θα συμμετέχουν στις πράξεις. Αρχικά δύο από αυτά του εμφανίζονται με τις κλάσεις που περιέχουν. Για κάθε κλάση εμφανίζεται και η αντίστοιχη ιδιότητα, αν υπάρχει, που ξεκινά από αυτήν την κλάση και καταλήγει σε κυριολεκτικό. Τον τελεστή της πράξης που θέλει να κάνει ο χρήστης μαζί με τους κόμβους κάθε σχήματος και τα κυριολεκτικά του κάθε κόμβου που θέλει να πάρουν μέρος, τα δίνει στο υποσύστημα αποθήκευσης επιλεγμένων κλάσεων για να αποθηκευτούν με τον κατάλληλο τρόπο. Εφόσον έχει δημιουργηθεί το τελικό σχήμα, ο χρήστης έχει τη δυνατότητα μέσω της διαπροσωπείας να επιλέξει την απεικόνισή του σε μορφή γράφου ή/και την αποθήκευσή του στη βάση. Αυτό επιτυγχάνεται με την επικοινωνία της διαπροσωπείας χρήστη με τα υποσύστημα απεικόνισης και αποθήκευσης RDF σχήματος αντίστοιχα. Τέλος, η διαπροσωπεία παρέχει τη δυνατότητα απεικόνισης RDF σχημάτων, μέσω του αντίστοιχου υποσυστήματος, ήδη αποθηκευμένων στη βάση σχημάτων και σχημάτων που προκύπτουν σε ενδιάμεσα στάδια κάποιων πράξεων. Στο Σχήμα 4.8 φαίνεται το διάγραμμα ροής της διαπροσωπείας χρήστη.

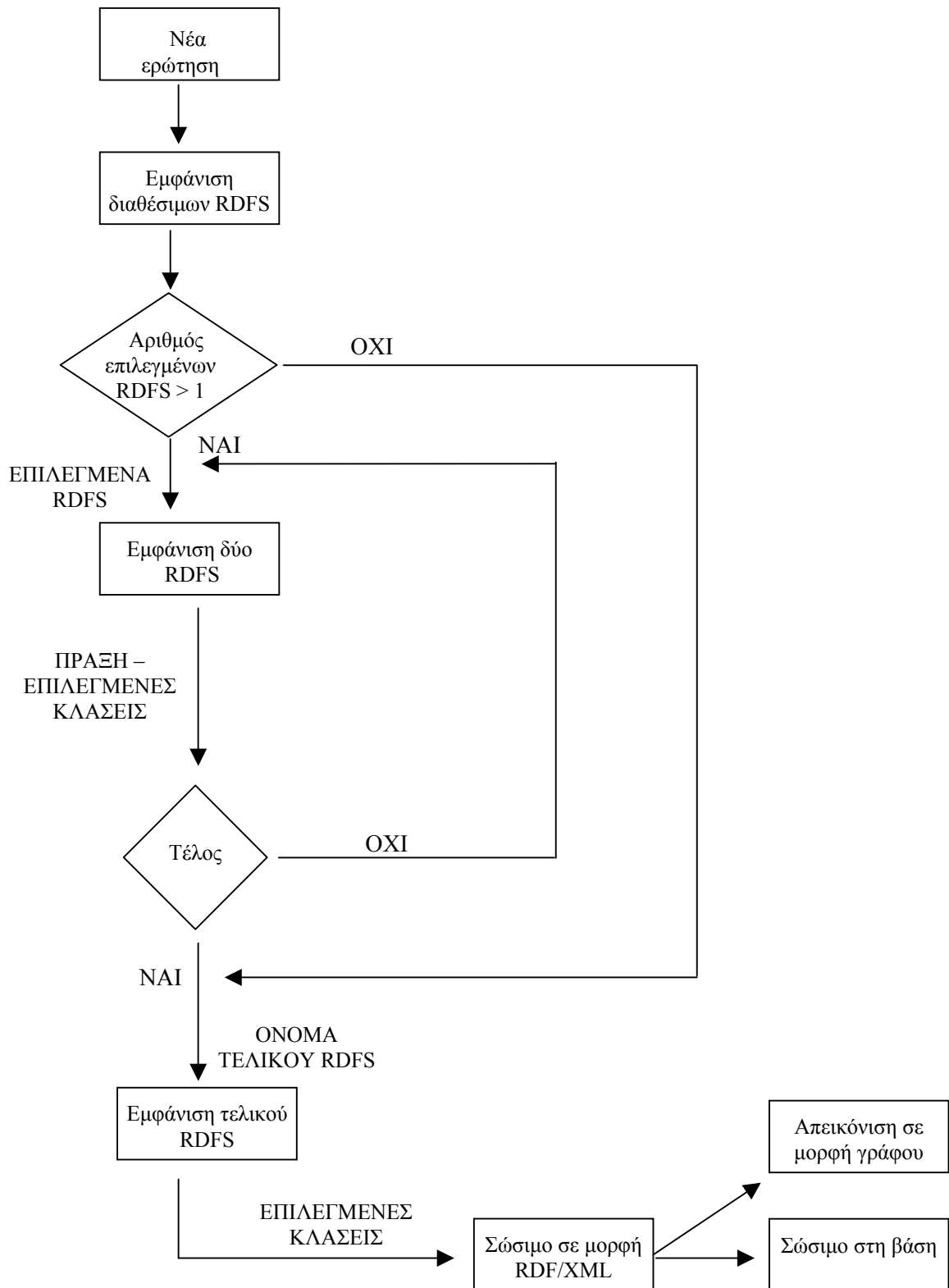
4.2 Σχεδίαση του συστήματος

Στην ενότητα αυτή παρουσιάζεται η σχεδίαση του συστήματος. Το σύστημα επιτελεί κάποιες λειτουργίες και παρακάτω δίνονται οι εφαρμογές του.

4.2.1 Εφαρμογές

Παρακάτω παρουσιάζονται οι εφαρμογές που διαθέτει το σύστημα. Αυτές είναι:

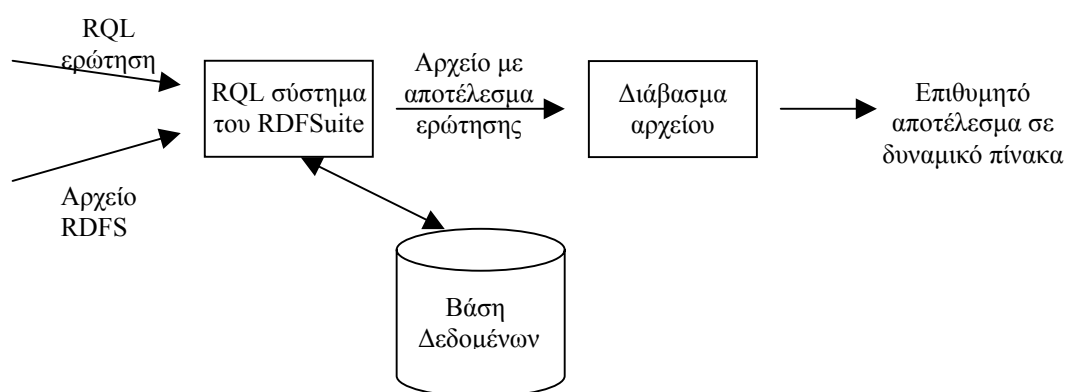
- Εφαρμογή διαχείρισης RQL ερωτήσεων
- Εφαρμογή διαχείρισης καθολικού σχήματος
- Εφαρμογή τελεστών
- Εφαρμογή αλγορίθμου επιλογής
- Εφαρμογή απεικόνισης RDF σχημάτων



Σχήμα 4.8

4.2.1.1 Εφαρμογή διαχείρισης βάσης

Η εφαρμογή αυτή διαχειρίζεται τη βάση του συστήματος. Εκτελεί ερωτήσεις RQL στη βάση για να πάρει πληροφορία για κάποιο αποθηκευμένο σχήμα και αποθηκεύει σχήματα στη βάση που έχουν δημιουργηθεί μετά από πράξεις. Η εφαρμογή χρησιμοποιεί την γλώσσα RQL και κώδικα φτιαγμένο για ερωτήσεις RQL καθώς και αποθήκευση σχημάτων στη βάση. Στο Σχήμα 4.9 φαίνεται η διαδικασία ερώτησης και αποθήκευσης ενός σχήματος.



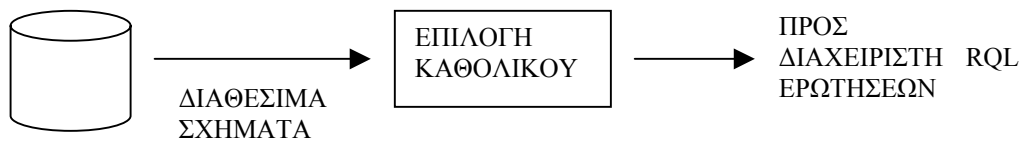
Σχήμα 4.9

Όσο αναφορά την ερώτηση RQL, αυτή γίνεται περνώντας σε string την ερώτηση μέσω μιας συνάρτησης που έχει υλοποιήσει το RDFSuite για την RQL. Το αποτέλεσμα δίνεται μέσα σε ένα μη διατεταγμένο σύνολο (bag) και γράφεται σε ένα αρχείο. Επομένως, διαβάζουμε το αρχείο με τη βοήθεια του SAX Parser που προσφέρει η Java και αποθηκεύουμε τα αποτελέσματα σε ένα πίνακα.

Αντίστοιχα, για την αποθήκευση ενός RDF σχήματος, έχοντας δημιουργήσει το αρχείο rdfs, δίνουμε ως παράμετρο σε μια συνάρτηση του συστήματος της RQL το μονοπάτι που βρίσκεται το αρχείο μαζί με τη βάση και η συνάρτηση αυτή κάνει τα απαραίτητα ώστε να αποθηκευτεί στη βάση το σχήμα.

4.2.1.2 Εφαρμογή αλλαγής καθολικού σχήματος

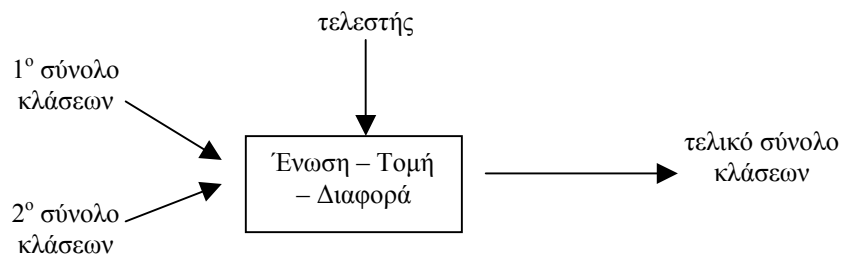
Το σύστημα δίνει τη δυνατότητα στο χρήστη να επιλέξει το καθολικό σχήμα με βάση το οποίο θα γίνουν οι πράξεις. Η επιλογή αυτή γίνεται ανάμεσα από τα διαθέσιμα σχήματα που υπάρχουν στη βάση. Έτσι, ο χρήστης μπορεί να κάνει διάφορες πράξεις και να πάρει διαφορετικά αποτελέσματα ανάλογα με το καθολικό σχήμα που έχει επιλέξει κάθε φορά. Στο σχήμα 4.10 απεικονίζεται η εφαρμογή αυτή.



Σχήμα 4.10

4.2.1.3 Εφαρμογή τελεστών

Η εφαρμογή αυτή είναι υπεύθυνη για τους τελεστές που θα εφαρμοστούν στα σύνολα κλάσεων δύο σχημάτων. Έχοντας τα σύνολα αυτά ανάλογα με τον τελεστή δημιουργεί ένα τελικό σύνολο κλάσεων. Στην περίπτωση του τελεστή ένωση, η εφαρμογή ενώνει τα δύο σύνολα και επομένως το αποτέλεσμα περιλαμβάνει όλες τις κλάσεις των δύο συνόλων. Στην περίπτωση του τελεστή τομή, η εφαρμογή βρίσκει την τομή των δύο συνόλων και το αποτέλεσμα περιέχει τις κοινές τους κλάσεις. Τέλος, στην περίπτωση του τελεστή διαφορά, η εφαρμογή βρίσκει τη διαφορά των δύο συνόλων. Το αποτέλεσμα περιέχει τις κλάσεις που υπάρχουν στο πρώτο σύνολο και δεν υπάρχουν στο δεύτερο. Παρακάτω ακολουθεί ένα διάγραμμα της εφαρμογής αυτής.

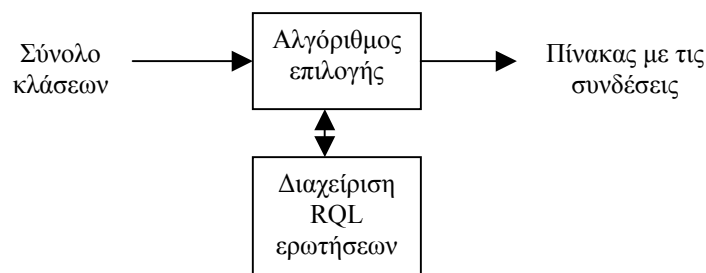


Σχήμα 4.11

4.2.1.4 Εφαρμογή αλγορίθμου επιλογής

Η εφαρμογή αυτή είναι υπεύθυνη για τον αλγόριθμο του τελεστή επιλογής που περιγράψαμε στο Κεφάλαιο 3. Έχοντας το σύνολο των κλάσεων που θέλουμε να προβάλλουμε στο καθολικό σχήμα, η εφαρμογή αυτή υλοποιεί τον αλγόριθμο βγάζοντας έτσι σε μια δομή ποιες κλάσεις πρέπει να συνδεθούν και με ποιες ιδιότητες. Κάνοντας RQL ερωτήσεις πάνω στο καθολικό σχήμα βγαίνει το συμπέρασμα για το πώς θα γίνει η σύνδεση των κλάσεων. Το

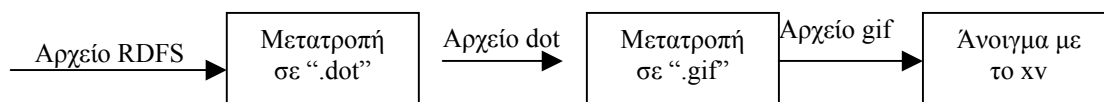
αποτέλεσμα αποθηκεύεται σε μια δομή δυναμικού πίνακα, ο οποίος για κάθε ιδιότητα έχει δίνει το πεδίο ορισμού και τιμών της. Το Σχήμα 4.12 απεικονίζει την εφαρμογή αυτή.



Σχήμα 4.12

4.2.1.5 Εφαρμογή απεικόνισης RDF σχημάτων

Το σύστημα δίνει τη δυνατότητα στο χρήστη να δει τα αποθηκευμένα σχήματα της βάσης είτε σε μορφή RDF γράφου είτε σε μορφή αρχείου. Με αυτόν τον τρόπο βλέπει τα σχήματα ώστε να του είναι πιο εύκολη η επιλογή αυτών πάνω στα οποία θέλει να εφαρμόσει τους τελεστές. Επίσης, η γραφική απεικόνιση μπορεί να γίνει και κατά τη διάρκεια των πράξεων για τα δύο σχήματα τα οποία εμφανίζονται για την εφαρμογή κάποιου τελεστή. Τα σχήματα αυτά μπορεί να είναι είτε από τα αρχικά σχήματα της βάσης είτε ενδιάμεσα σχήματα που έχουν προκύψει από προηγούμενα στάδια εφαρμογής των τελεστών. Στο παρακάτω σχήμα απεικονίζεται η εφαρμογή αυτή.



Σχήμα 4.13

Όσο αναφορά τη γραφική απεικόνιση, χρησιμοποιώντας το εργαλείο RDFSviz μετατρέπουμε το αρχείο rdfs σε μια ειδική μορφή αρχείου “.dot”. Το τελευταίο αυτό αρχείο χρησιμοποιώντας το εργαλείο Graphviz το μετατρέπουμε σε εικόνα “.gif”. Στη συνέχεια, καλούμε το πρόγραμμα xv του linux και απεικονίζεται ο γράφος αυτός.

Για την απεικόνιση σε μορφή αρχείου, απλά διαβάζουμε το αρχείο γραμμή προς γραμμή και το εμφανίζουμε σε ένα πλαίσιο.

5

Υλοποίηση

Στο κεφάλαιο αυτό περιγράφουμε πως υλοποιήθηκε το σύστημα με βάση τη θεωρητική μελέτη που περιγράψαμε. Αρχικά, δίνονται η πλατφόρμα και τα προγραμματιστικά εργαλεία που χρησιμοποιήσαμε. Στη συνέχεια δίνονται περισσότερες λεπτομέρειες όσο αναφορά τους αλγορίθμους και τις κλάσεις που υλοποιήσαμε.

5.1 Πλατφόρμες και προγραμματιστικά εργαλεία

Στην ενότητα αυτή παρουσιάζονται οι πλατφόρμες πάνω στις οποίες στήθηκε το σύστημα καθώς και εργαλεία που χρησιμοποιήσαμε για την υλοποίησή του.

5.1.1 Γενικά

Η υλοποίηση του συστήματος έγινε σε περιβάλλον Suse Linux με τη βοήθεια της γλώσσας προγραμματισμού Java. Για τη βάση δεδομένων χρησιμοποιήθηκε η PostgreSQL 7.3.2. Επίσης, βασικό εργαλείο για τη διαχείριση της βάσης ήταν η γλώσσα ερωτήσεων RQL (RDF Query Language) καθώς και το εργαλείο RSSDB του RDFSuite [11] για την αποθήκευση στη βάση RDF σχημάτων. Επίσης, για την απεικόνιση των RDF σχημάτων σε μορφή γράφου χρησιμοποιήθηκαν δύο εργαλεία: το πρώτο, το RDFSviz [10] μετέτρεπε ένα RDF σχήμα σε μια μορφή ειδικού αρχείου “.dot”, ενώ το δεύτερο, το GraphViz [9], μετέτρεπε το αρχείο

αυτό σε εικόνα “.gif”. Έτσι, χρησιμοποιώντας το εργαλείο xv του Linux ανοίγαμε τις εικόνες αυτές.

Πρέπει ακόμα να επισημάνουμε ότι η εκπόνηση της διπλωματικής έγινε στο Εργαστήριο Γνώσεων και Βάσεων Δεδομένων. Δουλεύαμε σε ένα τοπικό υπολογιστή του τοπικού δικτύου σε περιβάλλον Windows 2000 και συνδεόμασταν σε ένα άλλο μηχάνημα Linux χρησιμοποιώντας το SSH. Για την απεικόνιση εφαρμογών από το μηχάνημα Linux χρειάστηκε να εγκαταστήσουμε στο τοπικό υπολογιστή το πρόγραμμα X-win32 και να αρχικοποιήσουμε τη μεταβλητή περιβάλλοντος DISPLAY με το ip του τοπικού μηχανήματος.

5.1.2 Εγκατάσταση της RQL και του RSSDB

5.1.2.1 RQL

Για την εγκατάσταση της RQL χρειαστήκαμε την Postgresql 7.3.2, το gcc 2.98.3 και το jdk 1.4.2 της γλώσσας προγραμματισμού Java. Αφού κατεβάσαμε την RQL και αποσυμπιέσαμε το αρχείο, ακολουθήσαμε τα εξής βήματα:

- Αρχικοποίηση της μεταβλητής περιβάλλοντος LD_LIBRARY

Αρχικοποιήσαμε τη μεταβλητή LD_LIBRARY να περιέχει το μονοπάτι που βρίσκεται η βιβλιοθήκη της rql

- Δημιουργία του Makefile

Για τη δημιουργία του Makefile εκτελέσαμε την εντολή `configure` και μάλιστα με τέτοιο τρόπο ώστε να είναι δυνατή η εκτέλεση RQL ερωτήσεων μέσα από πρόγραμμα Java. Συγκεκριμένα, η εντολή που εκτελέσαμε είναι η εξής: `./configure --with-jni = μονοπάτι που βρίσκεται το include file της Java`

- Μεταγλώττιση της RQL

Για την μεταγλώττιση της RQL, αφού τρέξαμε την εντολή `make`, αποφασίσαμε να την εγκαταστήσουμε ως ανεξάρτητη εφαρμογή (standalone application) και γι αυτό το λόγο τη μεταγλωττίσαμε με την εντολή: `gmake local_client`. Τέλος, για να συμπεριλάβουμε και τη συνεργασία της με τη Java εκτελέσαμε την εντολή: `gmake jnilib`.

5.1.2.2 RSSDB

Αφού εγκαταστήσαμε την RQL, χρησιμοποιήσαμε και το RSSDB από το RDFSuite για την αποθήκευση RDF αρχείων στη βάση. Για να είναι δυνατή η χρήση αυτού του εργαλείου ακολουθήσαμε τα εξής βήματα:

- Αποσυμπίεση του αντίστοιχου αρχείου
- Αρχείο postgres.conf

Στο αρχείο αυτό της Postgres επιβεβαιώσαμε ότι η μεταβλητή tcpip_socket είναι on ώστε να είναι εφικτή η επικοινωνία, καθώς επίσης είδαμε ότι η μεταβλητή port είχε τιμή 5432.

- Αρχείο pg_hba.conf

Στο αρχείο αυτό της Postgres καθορίσαμε τον host να είναι το ip του μηχανήματος που εγκαταστήσαμε την RQL και υλοποιήσαμε το σύστημα.

- Αρχικοποίηση των μεταβλητών JAVA_HOME και RSSDB_HOME μέσα στο script
- Αρχικοποιήσαμε σε αυτές τις μεταβλητές το μονοπάτι που βρίσκεται ο φάκελος bin της Java και το μονοπάτι που βρίσκεται το αρχείο που κατεβάσαμε αντίστοιχα.
- Τρέξιμο του script

Το script run_rssdb το τρέξαμε με την εντολή: `source ./run_rssdb`

5.1.3 Εγκατάσταση του Graphviz και του RDFSviz

Τα εργαλεία αυτά τα χρησιμοποιήσαμε για την απεικόνιση RDFS αρχείων σε μορφή γράφου.

5.1.3.1 Graphviz

Το εργαλείο αυτό παίρνει ένα αρχείο “.dot” το οποίο έχει μια συγκεκριμένη μορφή και βγάζει μια εικόνα γράφου από το αρχείο αυτό. Για την εγκατάσταση του Graphviz εκτελέσαμε τις ακόλουθες ενέργειες:

- `./configure`
- `make`
- βάλουμε στη μεταβλητή περιβάλλοντος PATH το μονοπάτι των καταλόγων `dotty`, `lefty` και `dotneato` που χρειάζονται για την απεικόνιση του γράφου

Κατόπιν υποθέτοντας ότι έχουμε ένα αρχείο “x.dot” η εντολή που εκτελούμε για την εφαρμογή του εργαλείου αυτού ώστε να μετατρέψει το αρχείο αυτό σε “.gif” είναι:

```
dot -Tgif x.dot -o x.gif
```

5.1.3.2 RDFSviz

Το εργαλείο αυτό παίρνει ένα αρχείο RDFS και το μετατρέπει σε αρχείο “.dot” το οποίο έχει τη μορφή που χρειάζεται το Graphviz. Το εργαλείο αυτό είναι υλοποιημένο σε και η χρήση του είναι πολύ απλή. Για το τρέξιμο του εργαλείου αυτού εκτελέσαμε την παρακάτω εντολή:

```
java
-jar lib/RDFSschemaGrapher.jar
-rdfs file:/// x.rdfs
-RDFSNamespace http://www.w3.org/2000/01/rdf-schema#
-config config.xml
-o x.dot
```

όπου

```
-jar δηλώνει τη βιβλιοθήκη του εργαλείου
-rdfs το αρχείο RDFS που θέλουμε να μετατρέψουμε
-RDFSNamespace το namespace του RDF specification
-config ένα αρχείο xml που έχει ρυθμίσεις σε σχέση με τα χρώματα και
τα σχήματα
-o το αρχείο εξόδου.
```

5.2 Λεπτομέρειες υλοποίησης

Στην ενότητα αυτή παρουσιάζονται περισσότερες λεπτομέρειες για την υλοποίηση του συστήματος. Αρχικά αναφέρονται οι βασικοί αλγόριθμοι που αναπτύχθηκαν και στη συνέχεια περιγράφονται συνοπτικά οι κλάσεις του προγράμματος.

5.2.1 Αλγόριθμοι

Παρακάτω δίνονται οι κυριότεροι αλγόριθμοι που υλοποιήσαμε για την εφαρμογή και συγκεκριμένα για τη δημιουργία του τελικού RDF σχήματος. Μεταξύ αυτών είναι και ο αλγόριθμος που υλοποιεί τον ειδικό τελεστή επιλογή, τον οποίο περιγράψαμε στο Κεφάλαιο 3.

5.2.1.1 Αλγόριθμος επιλογής

Αυτός ο αλγόριθμος είναι ο αλγόριθμος που αναφέραμε και στο Κεφάλαιο 3 και υλοποιεί τον τελεστή επιλογή βρίσκοντας έτσι από ένα σύνολο κλάσεων ποιες κλάσεις πρέπει να συνδεθούν και με ποιες ιδιότητες, πάντα με βάση RQL ερωτήσεις στο καθολικό σχήμα. Δέχεται ως είσοδο ένα σύνολο κλάσεων που έχουν επιλεγεί από ένα σχήμα και έναν πίνακα ο οποίος έχει τις ιδιότητες με το πεδίο ορισμού και το πεδίο τιμών. Παρακάτω φαίνεται η επίσημη διατύπωση του αλγορίθμου.

Είσοδος: πίνακας με τις επιλεγμένες κλάσεις, classes

Έξοδος: πίνακας με τις ιδιότητες που θα συμπεριληφθούν στο τελικό σχήμα και το πεδίο ορισμού και τιμών της κάθε μίας. Ο πίνακας αυτός, result, έχει την εξής μορφή:

```
[ [ιδιότητα1, πεδίο_ορισμού1, πεδίο_τιμών1], [ιδιότητα2, πεδίο_ορισμού2, πεδίο_τιμών2], ...]
```

Αλγόριθμος:

```
/* κατασκευή του πίνακα bigVector που έχει μορφή: [ [ιδιότητα1,
πεδίο_ορισμού1, [πεδία_τιμών1], [ιδιότητα2, πεδίο_ορισμού2,
[πεδία_τιμών2], ... ] */
```

Για κάθε στοιχείο του classes, δηλαδή για κάθε κλάση

 Βάλε στον πίνακα property όλες τις ιδιότητες που έχουν ως πεδίο ορισμού την κλάση αυτή

```
    (RQL: "select @P from {;classes[i]}@P{$C}")
```

 Για κάθε στοιχείο του property, δηλαδή για κάθε ιδιότητα

 Βάλε στον πίνακα ranges όλες τις κλάσεις που είναι πεδία τιμών της ιδιότητας αυτής

```
        (RQL: "select $C from property[i]{$C}")
```

 Για κάθε στοιχείο του ranges

 Εάν ο πίνακας classes δεν περιέχει το στοιχείο αυτό τότε

 Αφαίρεσε το στοιχείο αυτό από τον ranges

 Εάν ο πίνακας ranges δεν είναι άδειος τότε

 Πρόσθεσε στον πίνακα propDomRange το στοιχείο του πίνακα property

 Πρόσθεσε στον πίνακα propDomRange το στοιχείο του πίνακα classes

 Πρόσθεσε στον πίνακα propDomRange όλο τον πίνακα ranges

 Εάν ο πίνακας propDomRange δεν είναι άδειος τότε

 Πρόσθεσε στον πίνακα bigVector όλον τον πίνακα propDomRange

```
/* κατασκευή του πίνακα finalVector που έχει μορφή: [ [ιδιότητα1,
[πεδία_ορισμού1], [πεδία_τιμών1]], [ιδιότητα2, [πεδία_ορισμού2],
[πεδία_τιμών2]], ... ] */
```

Για κάθε στοιχείο του bigVector

 Εάν ο πίνακας properties περιέχει το μηδενικό στοιχείο του πίνακα-στοιχείου του bigVector, δηλαδή την ιδιότητα αυτή, τότε

 Στο στοιχείο του πίνακα finalVector με την ίδια ιδιότητα πρόσθεσε το πρώτο στοιχείο του πίνακα bigVector, δηλαδή το πεδίο ορισμού

 Αλλιώς

 Πρόσθεσε στον finalVector έναν πίνακα με πρώτο στοιχείο την ιδιότητα, δεύτερο στοιχείο έναν πίνακα με το πεδίο ορισμού και τρίτο στοιχείο έναν πίνακα με τα πεδία τιμών

 Πρόσθεσε στον properties την ιδιότητα αυτή

```
/* κατασκευή του πίνακα result που έχει μορφή: [ [ιδιότητα1,
πεδίο_ορισμού1, πεδίο_τιμών1], [ιδιότητα2, πεδίο_ορισμού2,
πεδίο_τιμών2], ... ] */
```

Για κάθε στοιχείο του finalVector

Πρόσθεσε στον πίνακα domains όλα τα στοιχεία του δεύτερου στοιχείου του πίνακα-στοιχείου του finalVector, δηλαδή όλα τα πεδία ορισμού

Πρόσθεσε στον πίνακα ranges όλα τα στοιχεία του τρίτου στοιχείου του πίνακα-στοιχείου του finalVector, δηλαδή όλα τα πεδία τιμών

Πρόσθεσε στον πίνακα v το μηδενικό στοιχείο του πίνακα-στοιχείου του finalVector, δηλαδή την ιδιότητα

/* εύρεση μιας κλάσης για πεδίο ορισμού της ιδιότητας */

Εάν το μέγεθος του domains είναι 1, δηλαδή ένα πεδίο ορισμού, τότε

 Πρόσθεσε στον πίνακα v το μοναδικό στοιχείο του domains, δηλαδή το πεδίο ορισμού

Αλλιώς

 Βρες το πεδίο ορισμού της ιδιότητας αυτής

 (RQL: "domain(ιδιότητα[i])")

 Εάν το άμεσο πεδίο ορισμού της ιδιότητας ανήκει στον domains τότε

 Πρόσθεσε στον v το στοιχείο αυτό

Αλλιώς

 Για κάθε στοιχείο του πίνακα domains

 Βρες τις υπερκλάσεις του στοιχείου αυτού

 (RQL: "superClassOf(domains[i])")

 Εάν δεν υπάρχει η υπερκλάση αυτού του στοιχείου μέσα στον πίνακα classes τότε

 Πρόσθεσε το στοιχείο αυτό στον πίνακα domain1

 Εάν το μέγεθος του domain1 είναι 1 τότε

 /* όλες οι κλάσεις του domains ανήκουν σε ένα μονοπάτι ιεραρχίας */

 Πρόσθεσε στον v το στοιχείο αυτό

Αλλιώς

 /* πρέπει να προσθέσουμε έναν κόμβο που δεν υπάρχει μέσα στον πίνακα classes */

 ncaOld=κοντινότερος κοινός πρόγονος των δύο πρώτων στοιχείων του πίνακα domain1

 (RQL: "nca(domain1[0], domain1[1])")

 Για κάθε στοιχείο του domain1

 ncaNew=κοντινότερος κοινός πρόγονος του στοιχείου αυτού και του προηγούμενού του

 (RQL:"nca(domain1[i], domain1[i-1])")

 Εάν ncaOld≠ncaNew

 Βάλτε στον πίνακα subclass τις υποκλάσεις του ncaNew

 Εάν ο subclass περιέχει το ncaOld τότε

 ncaOld=ncaNew

 Πρόσθεσε στον classes τον ncaNew

 Πρόσθεσε στον v το ncaNew

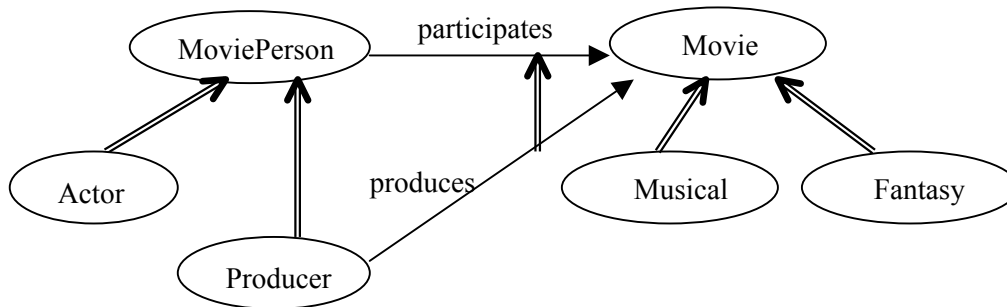

```

/* εύρεση μιας κλάσης για πεδίο τιμών της ιδιότητας */
Εάν το μέγεθος του ranges είναι 1, δηλαδή ένα πεδίο τιμών, τότε
    Πρόσθεσε στον πίνακα v το μοναδικό στοιχείο του ranges,
    δηλαδή το πεδίο τιμών
Αλλιώς
    Βρες το πεδίο τιμών της ιδιότητας αυτής
    (RQL: "range(ιδιότητας[i])")
Εάν το άμεσο πεδίο τιμών της ιδιότητας ανήκει στον ranges
    τότε
        Πρόσθεσε στον v το στοιχείο αυτό
Αλλιώς
    Για κάθε στοιχείο του πίνακα ranges
        Βρες τις υπερκλάσεις του στοιχείου αυτού
        (RQL: "superClassOf(ranges[i])")
Εάν δεν υπάρχει η υπερκλάση αυτού του
    στοιχείου μέσα στον πίνακα classes τότε
        Πρόσθεσε το στοιχείο αυτό στον πίνακα
        rangel
Εάν το μέγεθος του rangel είναι 1, δηλαδή
    όλες οι κλάσεις του ranges ανήκουν σε ένα
    μονοπάτι ιεραρχίας, τότε
        /* όλες οι κλάσεις του ranges ανήκουν
        σε ένα μονοπάτι ιεραρχίας */
        Πρόσθεσε στον v το στοιχείο αυτό
Αλλιώς
    /* πρέπει να προσθέσουμε έναν κόμβο που
    δεν υπάρχει μέσα στον πίνακα classes */
    ncaOld=κοντινότερος κοινός πρόγονος των
    δύο πρώτων στοιχείων του πίνακα rangel
    (RQL: "nca(rangel[0], rangel[1])")
    Για κάθε στοιχείο του rangel
        ncaNew=κοντινότερος κοινός
        πρόγονος του στοιχείου αυτού και
        του προηγούμενού του
        (RQL:"nca(rangel[i],
            rangel[i-1])")
    Εάν ncaOld≠ncaNew
        Βάλε στον πίνακα subclass
        τις υποκλάσεις του ncaNew
        Εάν ο subclass περιέχει το
        ncaOld τότε
            ncaOld=ncaNew
        Πρόσθεσε στον classes τον ncaNew
        Πρόσθεσε στον v το ncaNew
    Πρόσθεσε στον πίνακα result όλον τον πίνακα v

```

*Σημείωση: όταν λέμε ότι μια ιδιότητα έχει πολλά πεδία ορισμού εννοούμε ότι εκτός από την άμεση κλάση-πεδίο ορισμού από το οποίο ξεκινάει η ιδιότητα παίρνουμε και τα παιδιά της κλάσης αυτής, δηλαδή εννοούμε το ευρύτερο πεδίο ορισμού. Το ίδιο ισχύει και για τα πεδία τιμών.

Παρακάτω δίνουμε ένα παράδειγμα εφαρμογής του αλγορίθμου. Έστω ότι έχουμε το σχήμα που φαίνεται στο Σχήμα 5.3 και επιλέγουμε τις κλάσεις Actor, Producer και Movie.



Σχήμα 5.3

Σύμφωνα με τον αλγόριθμο θα έχουμε:

classes = [Actor, Producer, Movie]

➤ Actor

property = [participates]

○ participates

ranges = [Movie, Musical, Fantasy]

αφαιρούμε τις κλάσεις που δεν ανήκουν στον classes και μένει

ranges = [Movie]

propDomRange = [participates, Actor, [Movie]]

bigVector = [[participates, Actor, [Movie]]]

➤ Producer

property = [produces, participates]

○ produces

ranges = [Movie, Musical, Fantasy]

αφαιρούμε τις κλάσεις που δεν ανήκουν στον classes και μένει

ranges = [Movie]

propDomRange = [produces, Producer, [Movie]]

bigVector = [[participates, Actor, [Movie]], [produces, Producer, [Movie]]]

○ participates

ranges = [Movie, Musical, Fantasy]

αφαιρούμε τις κλάσεις που δεν ανήκουν στον classes και μένει

ranges = [Movie]

propDomRange = [participates, Producer, [Movie]]

bigVector = [[participates, Actor, [Movie]], [produces, Producer, [Movie]], [participates, Producer, [Movie]]]

➤ Movie

property = []

finalVector = [[participates, [Actor, Producer], [Movie]], [produces, [Producer], [Movie]]]

➤ participates

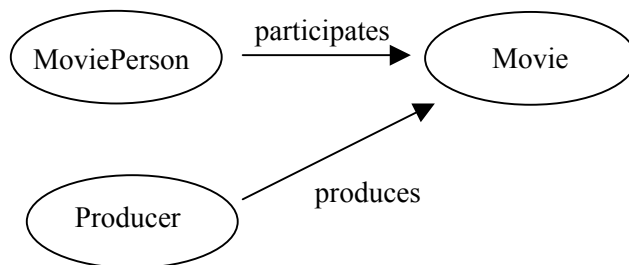
domains = [Actor, Producer]

```

ranges = [Movie]
v = [participates]
domains.size > 1
  RQL: domain(participates) = MoviePerson
  MoviePerson δεν ανήκει στον domains
  RQL: superClassOf(Actor) = MoviePerson, Resource
  καμμία δεν ανήκει στον domains
  domain1 = [Actor]
  RQL: superClassOf(Producer) = MoviePerson, Resource
  καμμία δεν ανήκει στον domains
  domain1 = [Actor, Producer]
  domain1.size > 1
  κοντινότερος πρόγονος (Actor, Producer) = MoviePerson
  v = [participates, MoviePerson]
  classes = [Actor, Producer, Movie, MoviePerson]
ranges.size = 1
  v = [participates, MoviePerson, Movie]
  result = [ [participates, MoviePerson, Movie] ]
➤ produces
domains = [Producer]
ranges = [Movie]
v = [produces]
domains.size = 1
  v = [produces, Producer]
ranges.size = 1
  v = [produces, Producer, Movie]
  result = [ [participates, MoviePerson, Movie], [produces, Producer, Movie] ]

```

Επομένως, ο αλγόριθμος έβγαλε το αποτέλεσμα που φαίνεται στο Σχήμα 5.4. Όσο αναφορά τις ιεραρχίες υλοποιούμε τους αλγορίθμους που δίνονται παρακάτω.



Σχήμα 5.4

5.2.1.2 Αλγόριθμος ένωσης

Ο αλγόριθμος αυτός εξηγεί πως υλοποιήσαμε την πράξη της ένωσης δύο συνόλων από κλάσεις. Για κάθε κλάση έχουμε και τις ιδιότητες που ξεκινούν από αυτήν και καταλήγουν σε κάποιο κυριολεκτικό.

Είσοδος: οι πίνακες $v1$, $v2$ με το σύνολο των κλάσεων του RDFS και τα κυριολεκτικά τους. Δηλαδή οι πίνακες αυτοί έχουν την εξής μορφή:

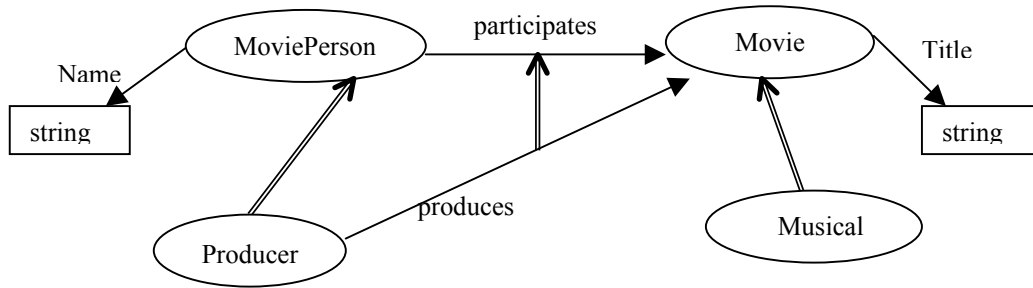
[[κλάση1,κυριολεκτικό11,κυριολεκτικό12,...], [κλάση2, κυριολεκτικό21, κυριολεκτικό22,...], ...]

Έξοδος: πίνακας με τις τελικές κλάσεις που θα συμμετέχουν στο τελικό RDFS και τα κυριολεκτικά τους. Ο πίνακας αυτός, *result*, έχει τη μορφή που έχουν και οι πίνακες εισόδου.

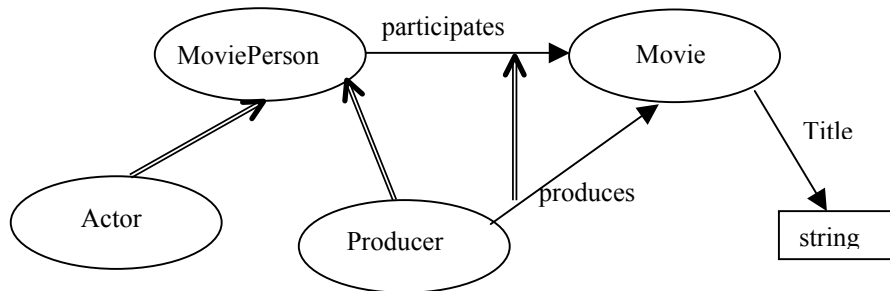
Αλγόριθμος:

```
Για κάθε πίνακα-στοιχείο του πίνακα  $v1$ 
    Πρόσθεσε το στοιχείο στον πίνακα result
/* ο result έχει όλες τις κλάσεις του πρώτου σχήματος */
Για κάθε πίνακα-στοιχείο του πίνακα  $v2$ 
    Εάν ο πίνακας result δεν περιέχει το στοιχείο(πίνακα) αυτό τότε
        /* ο result δεν περιέχει το ίδιο στοιχείο-πίνακα */
        Για κάθε στοιχείο του πίνακα result
            Εάν οι κλάσεις είναι οι ίδιες, δηλαδή τα στοιχεία
            που βρίσκονται στη μηδενική θέση των πινάκων
            στοιχείων που εξετάζουμε τότε
                /* τα ονόματα των κλάσεων ίδια, μία φορά στο
                βρόχο */
                Για κάθε στοιχείο του πίνακα-στοιχείου του
                πίνακα  $v2$ 
                    Εάν ο πίνακας-στοιχείο του πίνακα
                    result δεν περιέχει το στοιχείο του
                    πίνακα-στοιχείου του  $v2$  τότε
                        /* το στοιχείο-πίνακα του result
                        δεν περιέχει το κυριολεκτικό */
                        Πρόσθεσε το κυριολεκτικό αυτό
                    break;
            Εάν δεν έχει βρεθεί καμία όμοια κλάση στο result
            τότε
                Πρόσθεσε το στοιχείο αυτό του  $v1$  στο result
```

Παρακάτω φαίνεται ένα παράδειγμα εφαρμογής του αλγορίθμου αυτού. Στα Σχήματα 5.1 και 5.2 φαίνονται δύο RDF γράφοι. Ας υποθέσουμε ότι θέλουμε να ενώσουμε αυτούς τους γράφους.



Σχήμα 5.1



Σχήμα 5.2

Τότε, σύμφωνα με τον αλγόριθμο που περιγράψαμε παραπάνω, θα είχαμε

```
v1 = [ [MoviePerson, Name], [Producer], [Movie, Title], [Musical] ]
```

```
v2 = [ [MoviePerson], [Actor], [Producer], [Movie, Title] ]
```

και το αποτέλεσμα ύστερα από εφαρμογή του αλγορίθμου θα ήταν το εξής:

```
result = [ [MoviePerson, Name], [Actor], [Producer], [Movie, Title] ]
```

5.2.1.3 Αλγόριθμος τομής

Ο αλγόριθμος αυτός εξηγεί πως υλοποιήσαμε την πράξη της τομής δύο συνόλων από κλάσεις. Για κάθε κλάση έχουμε και τις ιδιότητες που ξεκινούν από αυτήν και καταλήγουν σε κάποιο κυριολεκτικό.

Είσοδος: οι πίνακες $v1$, $v2$ με το σύνολο των κλάσεων του RDFS και τα κυριολεκτικά τους.

Δηλαδή οι πίνακες αυτοί έχουν την εξής μορφή:

```
[[κλάση1,κυριολεκτικό11,κυριολεκτικό12,...], [κλάση2, κυριολεκτικό21, κυριολεκτικό22,... ], ... ]
```

Έξοδος: πίνακας με τις τελικές κλάσεις που θα συμμετέχουν στο τελικό RDFS και τα κυριολεκτικά τους. Ο πίνακας αυτός, $result$, έχει τη μορφή που έχουν και οι πίνακες εισόδου.

Αλγόριθμος:

```
Για κάθε στοιχείο(πίνακα) του v1
  Για κάθε στοιχείο(πίνακα) του v2
    Εάν τα στοιχεία αυτά είναι ίσα τότε
      Πρόσθεσε το στοιχείο αυτό στον πίνακα result
    Αλλιώς εάν είναι ίσα τα πρώτα στοιχεία των στοιχείων-
    πινάκων τότε
      /* ίδια ονόματα κλάσεων και διαφορετικά
      κυριολεκτικά */
      Για κάθε στοιχείο του στοιχείου πίνακα του v1
        Εάν το στοιχείο πίνακα του πίνακα v2 δεν
        περιέχει το στοιχείο του στοιχείου πίνακα του
        v1 τότε
          /* ο πίνακας του v1 δεν περιέχει το
          κυριολεκτικό του v2 */
          Αφαίρεσε από το στοιχείο πίνακα του
          πίνακα v1 το στοιχείο αυτό
      Πρόσθεσε στον result το στοιχείο πίνακα του v1
```

Παρακάτω δίνουμε ένα παράδειγμα εφαρμογής του αλγορίθμου αυτού. Έστω πάλι ότι έχουμε τους γράφους των Σχημάτων 5.1 και 5.2. σύμφωνα με τον αλγόριθμο εφαρμογής του τελεστή τομής θα είχαμε:

```
v1 = [ [MoviePerson, Name], [Producer], [Movie, Title], [Musical] ]
```

```
v2 = [ [MoviePerson], [Actor], [Producer], [Movie, Title] ]
```

και το αποτέλεσμα της τομής θα ήταν το εξής:

```
result = [ [[MoviePerson], [Producer], [Movie, Title] ]
```

5.2.1.4 Αλγόριθμος διαφοράς

Ο αλγόριθμος αυτός εξηγεί πως υλοποιήσαμε την πράξη της διαφοράς δύο συνόλων από κλάσεις. Για κάθε κλάση έχουμε και τις ιδιότητες που ξεκινούν από αυτήν και καταλήγουν σε κάποιο κυριολεκτικό.

Είσοδος: οι πίνακες v1, v2 με το σύνολο των κλάσεων του RDFS και τα κυριολεκτικά τους.

Δηλαδή οι πίνακες αυτοί έχουν την εξής μορφή:

```
[[κλάση1,κυριολεκτικό11,κυριολεκτικό12,...], [κλάση2, κυριολεκτικό21, κυριολεκτικό22,... ],
... ]
```

Έξοδος: πίνακας με τις τελικές κλάσεις που θα συμμετέχουν στο τελικό RDFS και τα κυριολεκτικά τους. Ο πίνακας αυτός, result, έχει τη μορφή που έχουν και οι πίνακες εισόδου.

Αλγόριθμος:

```
Για κάθε στοιχείο-πίνακα του v1
```

```

Πρόσθεσε το στοιχείο αυτό στον result
Για κάθε στοιχείο-πίνακα του v2
    Εάν το πρώτο στοιχείο του πίνακα του v1 είναι ίσο με το
    πρώτο στοιχείο του πίνακα του v2 τότε
        /* ίδια ονόματα κλάσεων */
        Αφαίρεσε από το result το όλο το στοιχείο-πίνακα
        του v1

```

Παρακάτω δίνουμε το αντίστοιχο παράδειγμα εφαρμογής του αλγορίθμου διαφοράς. Έστω ότι θέλαμε να βρούμε τη διαφορά Σχήμα 5.1 μείον το Σχήμα 5.2. Τότε θα έχουμε:

```
v1 = [ [MoviePerson, Name], [Producer], [Movie, Title], [Musical] ]
```

```
v2 = [ [MoviePerson], [Actor], [Producer], [Movie, Title] ]
```

και το αποτέλεσμα της τομής θα ήταν το εξής:

```
result = [ [Musical] ]
```

5.2.1.5 Αλγόριθμος για την παρουσίαση της ιεραρχίας των κλάσεων

Ο αλγόριθμος αυτός δείχνει πως βρήκαμε τη σχέση μεταξύ των κλάσεων όσο αναφορά την ιεραρχία. Έτσι, γράψαμε στο αρχείου του σχήματος τις σωστές σχέσεις.

Είσοδος: πίνακας με όλες τις επιλεγμένες κλάσεις και τις κλάσεις που έχει προσθέσει ο αλγόριθμος επιλογής, classes.

Έξοδος: μέρος του RDFS αρχείου που αναπαριστά ποιες κλάσεις υπάρχουν και τη σχέση έχουν μεταξύ τους.

Αλγόριθμος:

```

Για κάθε στοιχείο του πίνακα classes, δηλαδή για κάθε κλάση
    Βάλε στον πίνακα superClass τις υπερκλάσεις της κλάσης αυτής
    Για κάθε στοιχείο του πίνακα superClass
        Εάν ο classes περιέχει το στοιχείο αυτό τότε
            Γράψε στο αρχείο ως κλάση το στοιχείο του classes
            και ότι είναι υποκλάση του στοιχείου του superClass
            break
        Εάν δεν υπάρχει καμία υπερκλάση του στοιχείου του classes
        τότε
            Γράψε στο αρχείο ως κλάση μόνο το στοιχείο του
            classes

```

Παρακάτω δίνουμε ένα παράδειγμα το οποίο αφορά το παράδειγμα του Σχήματος 5.3 έχοντας επιλέξει τις κλάσεις Actor, Producer και Movie και έχοντας εφαρμόσει πρώτα τον αλγόριθμο επιλογής που περιγράψαμε παραπάνω. Έτσι, έχουμε:

classes = [Actor, Producer, Movie, MoviePerson]

➤ Actor

superClass = [MoviePerson, Resource]

- MoviePerson

ανήκει στον classes

Actor → MoviePerson

τέλος

➤ Producer

superClass = [MoviePerson, Resource]

- MoviePerson

ανήκει στον classes

Producer → MoviePerson

τέλος

➤ Movie

superClass = [Resource]

- Resource

δεν ανήκει στον classes

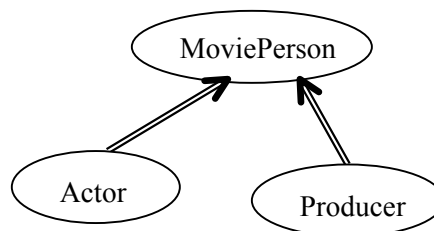
➤ MoviePerson

superClass = [Resource]

- Resource

δεν ανήκει στον classes

Άρα συμπεράναμε ότι οι ιεραρχίες που υπάρχουν είναι αυτές που φαίνονται στο Σχήμα 5.5



Σχήμα 5.5

5.2.1.6 Αλγόριθμος για την παρουσίαση των ιδιοτήτων

Ο αλγόριθμος αυτός δείχνει πως υλοποιήσαμε μέρος του τελικού αρχείου του RDF σχήματος όσο αναφορά τις ιδιότητες. Σ' αυτό βοήθησε και ο αλγόριθμος που παρουσιάσαμε στην Ενότητα 5.2.1.4, ο οποίος επιστρέφει τον πίνακα result.

Είσοδος: ο πίνακας result ο οποίος έχει την εξής μορφή: [[ιδιότητα1, πεδίο_ορισμού1, πεδίο_τιμών1], [ιδιότητα2, πεδίο_ορισμού2, πεδίο_τιμών2], ...]

Έξοδος: μέρος του RDFS αρχείου που αναπαριστά τις ιδιότητες που υπάρχουν με το πεδίο ορισμού και τιμών τους καθώς και τη σχέση που έχουν μεταξύ τους οι ιδιότητες.

Αλγόριθμος:

Για κάθε στοιχείο του πίνακα result

Γράψε το μηδενικό στοιχείο του πίνακα-στοιχείου του result, δηλαδή την ιδιότητα

Γράψε το πρώτο στοιχείο του πίνακα-στοιχείου του result, δηλαδή το πεδίο ορισμού

Γράψε το πρώτο στοιχείο του πίνακα-στοιχείου του result, δηλαδή το πεδίο ορισμού

Βάλε στον πίνακα superProperty τις υπεριδιότητες της ιδιότητας αυτής

Για κάθε στοιχείο του πίνακα superProperty

Εάν ο πίνακας finalProperties περιέχει το στοιχείο αυτό τότε

Γράψε ότι η ιδιότητα του result είναι υποιδιότητα του στοιχείου του superProperty

Πρόσθεσε στον finalProperties την ιδιότητα αυτή

Παρακάτω δίνουμε τη συνέχεια του παραδείγματος που ξεκινήσαμε στην Ενότητα 5.2.1.5 για τον αλγόριθμο επιλογής. Έτσι είχαμε:

```
result = [ [participates, MoviePerson, Movie], [produces, Producer, Movie] ]
```

➤ participates

```
superProperty = [ ]
```

```
finalproperties = [participates]
```

➤ produces

```
superProperty = [participates]
```

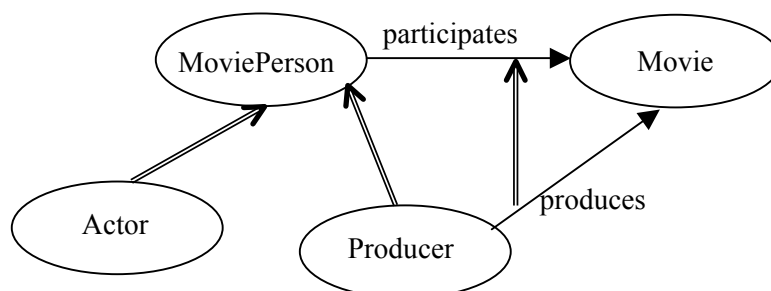
ανήκει στον finalproperties

```
produces → participates
```

```
finalproperties = [participates, produces]
```

επομένως, βρήκαμε ότι η ιδιότητα produces είναι υποιδιότητα της participates.

Τελικά αφού έχουμε εφαρμόσει όλους τους αλγορίθμους από την Ενότητα 5.2.1.4 ως και 5.2.1.6 βλέπουμε το τελικό αποτέλεσμα στο Σχήμα 5.6.



Σχήμα 5.6

5.2.1.7 Αλγόριθμος για την παρουσίαση των κυριολεκτικών

Ο αλγόριθμος αυτός δείχνει τον τρόπο με τον οποίο υλοποιήσαμε μέρος του αρχείου RDFS όσο αναφορά τις ιδιότητες που έχουν πεδίο τιμών κυριολεκτικά.

Είσοδος: πίνακας που περιέχει τις κλάσεις και τα κυριολεκτικά κάθε μίας κλάσης. Ο πίνακας `rdfsSelected` έχει τη μορφή: `[[κλάση1,κυριολεκτικό11,κυριολεκτικό12,...], [κλάση1,κυριολεκτικό11,κυριολεκτικό12,...], ...]`

Έξοδος: μέρος του αρχείου του τελικού RDFS σχήματος που αναπαριστά τις ιδιότητες που καταλήγουν σε κυριολεκτικά.

Αλγόριθμος:

Για κάθε στοιχείο του πίνακα `rdfsSelected`

Εάν ο πίνακας-στοιχείο του `rdfsSelected` έχει μέγεθος πάνω από 1 τότε

Για κάθε στοιχείο του πίνακα-στοιχείου του `rdfsSelected` από τη θέση 1 και μετά

Βάλτε στο αλφαριθμητικό `range` το πεδίο τιμών του στοιχείου αυτού, δηλαδή του κυριολεκτικού

Γράψτε ως ιδιότητα το κυριολεκτικό αυτό

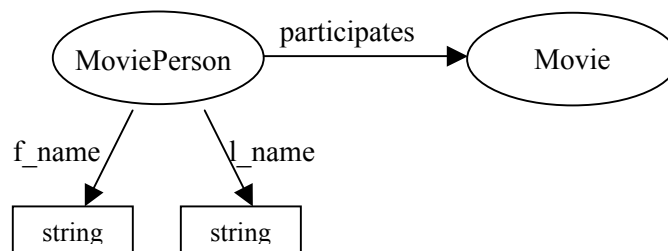
Γράψτε ως πεδίο ορισμού το μηδενικό στοιχείο του πίνακα-στοιχείου του `rdfsSelected`

Γράψτε ως πεδίο τιμών το `range`

*Σημείωση: πρέπει να σημειώσουμε ότι λέγοντας κυριολεκτικό εννοούμε την ιδιότητα που έχει ως πεδίο τιμών κυριολεκτικό, δηλαδή τύπο δεδομένων.

Για παράδειγμα ας δούμε το Σχήμα 5.7. Ο πίνακας εισόδου για τον αλγόριθμο αυτό θα ήταν σε αυτήν την περίπτωση ο εξής:

`rdfsSelected = [[MoviePerson, Name], [Movie]]`



Σχήμα 5.7

Επομένως θα είχαμε:

`[MoviePerson, Name].size > 1`

➤ `f_name`

```
MoviePerson -> f_name
➤ l_name
MoviePerson -> l_name
[Movie].size = 1
```

Άρα θα συνδέαμε έτσι τις κλάσεις με τα κυριολεκτικά τους.

5.2.2 Περιγραφή κλάσεων

Παρακάτω δίνεται μια πιο λεπτομερή ανάλυση των κλάσεων που δημιουργήσαμε για την υλοποίηση του συστήματος της εφαρμογής. Για κάθε κλάση μια σύντομη περιγραφή των πεδίων και μεθόδων της.

5.2.2.1 *public class ui*

Η κλάση αυτή υλοποιεί όλη τη διαπροσωπία του συστήματος.

Πεδία

- `private final String dbName`
Το όνομα της βάσης
- `private String globalName`
Το όνομα του καθολικού σχήματος
- `private RQLQuery rquery`
Αντικείμενο της κλάσης RQLQuery
- `private db d`
Αντικείμενο της κλάσης db
- `private JFrame frame`
Το αρχικό frame της εφαρμογής
- `private JPanel panel`
Το αρχικό Panel της εφαρμογής
- `private GridBagLayout blayout`
Το layout για όλα τα Panel
- `private GridBagConstraints con`
Τα constraints για το layout
- `private JMenuBar bar`
Το μενού της εφαρμογής
- `private JMenu fileMenu`
Το μενού file
- `private JMenu viewMenu`
Το μενού view
- `private JMenuItem exitItem`
Το κομμάτι Exit του μενού file για έξοδο από την εφαρμογή
- `private JMenuItem queryItem`
Το κομμάτι New Query του μενού file για την έναρξη καινούριας ερώτησης
- `private JMenuItem globalItem`
Το κομμάτι Set global Schema του μενού file για την αλλαγή του καθολικού σχήματος
- `private JMenuItem viewItem`
Το κομμάτι View RDF Schemas του μενού view για την εμφάνιση του RDF σχήματος σε μορφή γράφου

- `private JMenuItem viewRdfsItem`
Το κομμάτι View RDF Files του μενού view για την εμφάνιση του RDF σχήματος σε μορφή αρχείου
- `private Icon arrowL`
Εικονίδιο για το κουμπί Next
- `private Icon tick`
Εικονίδιο για το κουμπί Finish
- `private Icon saveIcon`
Εικονίδιο για το κουμπί Save
- `private Icon demoIcon`
Εικονίδιο για το κουμπί View Result
- `private Icon eyes`
Εικονίδιο για το κουμπί View
- `private Vector rdfsNames`
Όλα τα rdf σχήματα που υπάρχουν στη βάση
- `private Vector rdfsVector`
Τα επιλεγμένα από τον χρήστη rdf σχήματα
- `private Vector rdfsClasses1`
Οι κλάσεις του σχήματος στην αριστερή λίστα
- `private Vector rdfsClasses2`
Οι κλάσεις του σχήματος στην δεξιά λίστα
- `private Vector rdfsClass1Selected`
Οι επιλεγμένες κλάσεις στην αριστερή λίστα
- `private Vector rdfsClass2Selected`
Οι επιλεγμένες κλάσεις στην δεξιά λίστα
- `private Vector rdfsAllClasses1`
Όλες οι κλάσεις του σχήματος του σχήματος του αριστερού ComboBox
- `private Vector rdfsAllClasses2`
Όλες οι κλάσεις του σχήματος του σχήματος του δεξιού ComboBox
- `private Vector rdfsSelected1`
Πίνακας πινάκων με τις κλάσεις που έχουν επιλεγεί και τα κυριολεκτικά που έχουν επιλεγεί για κάθε κλάση από το σχήμα του αριστερού ComboBox
- `private Vector rdfsSelected2`
Πίνακας πινάκων με τις κλάσεις που έχουν επιλεγεί και τα κυριολεκτικά που έχουν επιλεγεί για κάθε κλάση από το σχήμα του δεξιού ComboBox
- `private Vector rdfsResult`
Πίνακας με τις κλάσεις και τα κυριολεκτικά τους μετά από την πράξη
- `private Vector rdfsResultsVector`
Πίνακας με όλους τους rdfsResult πίνακες όλων των ενδιάμεσων αποτελεσμάτων
- `private Vector literal`
Τα κυριολεκτικά της κλάσης που είναι επιλεγμένη στην αριστερή λίστα
- `private Vector literal2`
Τα κυριολεκτικά της κλάσης που είναι επιλεγμένη στην δεξιά λίστα
- `private Vector box1`
Πίνακας με τα checkboxes του αριστερού Panel
- `private Vector box2`
Πίνακας με τα checkboxes του δεξιού Panel
- `private Vector resultVector`
Πίνακας με τις κλάσεις του αποτελέσματος
- `private String resultName`
Όνομα του τελικού σχήματος
- `private String resultNameCut`
Όνομα του τελικού σχήματος χωρίς την κατάληξη
- `private int counter`
Μετρητής με το πόσες φορές έχει εμφανιστεί το Panel των πράξεων
- `private Vector processVector = new Vector();`
Πίνακα με τις διεργασίες για την εμφάνιση των γράφων

Μέθοδοι

- `public ui()`
Κατασκευαστής για αρχικοποίηση του καθολικού σχήματος και φόρτωση της βιβλιοθήκης για την RQL.
- `private void createFrame()`
Δημιουργία του πρώτου frame της εφαρμογής
- `private void createPanel()`
Δημιουργία του Panel για το πρώτο frame της εφαρμογής
- `private void viewAllRdfs(final int x)`
Δημιουργία Panel για τη δυνατότητα εμφάνισης του επιθυμητού σχήμα είτε σε μορφή γράφου (x=0) είτε σε μορφή αρχείου (x=1)
- `private void createListFrame()`
Δημιουργία Frame με λίστα διαθέσιμων σχημάτων στη βάση και δυνατότητα επιλογής από αυτά
- `private void createOperatFrame(int index)`
Δημιουργία Frame για τις πράξεις. Το index καθορίζει ποιο σχήμα θα βρίσκεται στο πρώτο ComboBox
- `private void finalResultFrame()`
Δημιουργία τελικού Frame με το αποτέλεσμα και δυνατότητα στο χρήστη να το σώσει σε αρχείο, στη βάση ή να το δει σε μορφή γράφου.
- `private void addComponent(Component component, Container container, int row, int column, int width, int height)`
Τοποθετεί το component πάνω στο container στην επιθυμητή θέση
- `private void clearAllVectors1()`
Καθαρίζει όλους τους πίνακες που χρησιμοποιούνται στο αριστερό Panel
- `private void clearAllVectors2()`
Καθαρίζει όλους τους πίνακες που χρησιμοποιούνται στο δεξιό Panel

5.2.2.2 *public class Operation*

Η κλάση αυτή υλοποιεί τις πράξεις ένωση, τομή και διαφορά των επιλεγμένων από το χρήστη κλάσεων των δύο RDF σχημάτων.

Πεδία

- `private String oper`
Ο επιθυμητός τελεστής
- `Vector v1`
Οι κλάσεις με τα επιλεγμένα κυριολεκτικά του ενός RDF σχήματος
- `Vector v2`
Οι κλάσεις με τα επιλεγμένα κυριολεκτικά του άλλου RDF σχήματος
- `Vector result`
Οι κλάσεις με τα κυριολεκτικά τους μετά την εφαρμογή του τελεστή

Μέθοδοι

- `public Operation(Object oper, Vector v1, Vector v2)`
Κατασκευαστής που αρχικοποιεί όλα τα πεδία και αποφασίζει τελικά ποια πράξη πρέπει να εκτελεστεί ανάλογα με τη μεταβλητή oper
- `private void union()`
Τελεστής ένωσης πάνω στις κλάσεις
- `private void intersection()`
Τελεστής τομής πάνω στις κλάσεις
- `private void difference()`
Τελεστής διαφοράς πάνω στις κλάσεις

- `public Vector getResult()`
Επιστρέφει πίνακα με τις σωστές κλάσεις μετά την εφαρμογή του τελεστή

5.2.2.3 *public class RQLQuery*

Η κλάση αυτή δημιουργεί σύνδεση με την Postgres και με τη βιβλιοθήκη της RQL και επιτρέπει την εκτέλεση ερώτησης RQL.

Πεδία

- `RQLDatabaseConnection dbc`
Μεταβλητή για σύνδεση με τη βάση
- `String queryfile`
Το όνομα αρχείου με το αποτέλεσμα της ερώτησης

Μέθοδοι

- `public RQLQuery(String dbName)`
αρχικοποίηση των πεδίων, σύνδεση με τη βάση και φόρτωμα της βιβλιοθήκης της RQL
- `public void executeQuery(String s)`
Εκτέλεση της RQL ερώτησης που δίνεται από το `s`

5.2.2.4 *public class db*

Η κλάση αυτή υλοποιεί συναρτήσεις που επιστρέφουν το αποτέλεσμα ερωτήσεων RQL

Πεδία

- `private RQLQuery rquery`
Αντικείμενο της κλάσης RQLQuery για ερωτήσεις RQL
- `private Vector rdfsVector`
Τα επιλεγμένα rdf σχήματα
- `private String global`
Το όνομα του καθολικού σχήματος

Μέθοδοι

- `public db(RQLQuery rquery, Vector rdfsVector)`
Αρχικοποίηση των πεδίων με τις αντίστοιχες παραμέτρους
- `public void setGlobalSchema(String global)`
Θέτει το καθολικό σχήμα ίσο με την τιμή της παραμέτρου
- `public Vector getNamespacesFromDB()`
Επιστρέφει όλα ονόματα των σχημάτων που υπάρχουν στη βάση
- `public Vector getClassesFromDB(int index)`
Επιστρέφει όλες τις κλάσεις του σχήματος που βρίσκεται στη θέση `index` του πίνακα `rdfsVector`
- `public Vector getLiteralFromDB(String s, int index)`
Επιστρέφει όλα τα κυριολεκτικά της κλάσης, που δηλώνει η παράμετρος `s`, από το σχήμα που βρίσκεται στη θέση `index` του πίνακα `rdfsVector`
- `public Vector getSuperClassFromDB(String className)`
Επιστρέφει τις υπερκλάσεις της κλάσης, που δηλώνει η παράμετρος, από το καθολικό σχήμα
- `public Vector getSubClassFromDB(String s)`
Επιστρέφει τις υποκλάσεις της κλάσης, που δηλώνει η παράμετρος, από το καθολικό σχήμα
- `public Vector getSuperPropertyFromDB(String s)`
Επιστρέφει τις υπεριοιότητες της ιδιότητας, που δηλώνει η παράμετρος, από το καθολικό σχήμα

- `public Vector getPropertyFromDB(String clas)`
Επιστρέφει τις ιδιότητες από το καθολικό σχήμα οι οποίες έχουν πεδίο ορισμού την κλάση που δηλώνει η παράμετρος
- `public Vector getAllRangesFromDB(String property)`
Επιστρέφει τις κλάσεις από το καθολικό σχήμα που αποτελούν πεδίο τιμών της ιδιότητας που δηλώνει η παράμετρος
- `public String getNCAFromDB(String s1, String s2)`
Επιστρέφει το κοντινότερο κοινό πρόγονο των κλάσεων που δηλώνουν οι παράμετροι από το καθολικό σχήμα

5.2.2.5 *public class ReadRDFFile*

Η κλάση αυτή αποτελεί ένα είδος Parser που διαβάζει ένα αρχείο αποτελέσματος μιας ερώτησης RQL και επιστρέφει το αποτέλεσμα σε πίνακα.

Πεδία

- `public static Vector v`
Τα στοιχεία του αποτελέσματος μετά από το διάβασμα του αρχείου
- `ReadRDFFileHandler rgh`
Αντικείμενο της εσωτερικής κλάσης `ReadRDFFileHandler`

Μέθοδοι

- `ReadRDFFile()`
Κενός κατασκευαστής
- `ReadRDFFile(String xmlFile, String tagName, String attributeName)`
Κατασκευαστής που χρησιμοποιεί τον SAX Parser
- `public Vector getVectorAttributes()`
Επιστρέφει τον πίνακα με το αποτέλεσμα
- `public void addToVector(Object o)`
Προσθέτει στον πίνακα `v` το αντικείμενο `o`

5.2.2.6 *public class CreateRdfsFile*

Η κλάση αυτή δημιουργεί ένα αρχείο RDF, ουσιαστικά ένα RDF σχήμα, έχοντας γνωστές τις κλάσεις του σχήματος αυτού και χρησιμοποιώντας τον αλγόριθμο που περιγράψαμε παραπάνω.

Πεδία

- `private FileOutputStream outputFile`
Το αρχείο που θα γραφτεί το RDF σχήμα
- `private PrintStream out`
Stream για το γράψιμο του αρχείου
- `private String filename`
Το όνομα του αρχείου και άρα του σχήματος
- `private db dbase`
Το όνομα της βάσης
- `private Vector rdfsSelected`
Πίνακας με τις επιλεγμένες κλάσεις και τα κυριολεκτικά της κάθε μίας
- `private Vector classes`
Πίνακας με τις επιλεγμένες κλάσεις
- `private Vector properties`

Πίνακας με τις ιδιότητες που θα έχουν ως πεδίο ορισμού τις κλάσεις του πίνακα classes

- `private Vector finalProperties`

Πίνακας με τις τελικές ιδιότητες που θα χρησιμοποιηθούν

- `private Vector result`

Πίνακας με στοιχεία πίνακες που αποτελούνται από την ιδιότητα, το πεδίο ορισμού της και το πεδίο τιμών της

- `public static String newline`

Αλλαγή γραμμής στο αρχείο

Μέθοδοι

- `public CreateRdfsFile(String filename, Vector rdfsSelected, db dbase)`

Κατασκευαστής που αρχικοποιεί τα πεδία με τις τιμές των παραμέτρων και τον πίνακα classes με τη βοήθεια της παραμέτρου rdfsSelected

- `private void algorithm()`

Υλοποίηση του αλγορίθμου που περιγράψαμε σε παραπάνω ενότητα

- `public void writeToFile()`

Γράφει το τελικό αρχείο γραμμή – γραμμή σύμφωνα με τα αποτελέσματα του αλγορίθμου

5.2.2.7 *public class StoreToDB*

Η κλάση αυτή αποθηκεύει στη βάση το τελικό RDF σχήμα που προέκυψε μετά τις επιθυμητές πράξεις του χρήστη. Χρησιμοποιεί κλάσεις που παρέχει η RQL για την αποθήκευση τέτοιων σχημάτων.

Πεδία

- `private SpecReprDBStorage storage`
Αντικείμενο της κλάσης SpecReprDBStorage της RQL
- `private String dbname`
Όνομα της βάσης
- `private String filename`
Όνομα του αρχείου που θέλουμε να αποθηκεύσουμε

Μέθοδοι

- `public StoreToDB(String dbname, String filename)`

Κατασκευαστής που αρχικοποιεί το όνομα της βάσης και του αρχείου

- `public void store()`

Καλεί τις κατάλληλες μεθόδους της RQL για την αποθήκευση του σχήματος

5.2.2.8 *public class Begin*

Η κλάση αυτή ξεκινά την εφαρμογή μας καθώς περιέχει τη συνάρτηση main.

Πεδία

- `public ui u`
Αντικείμενο της κλάσης ui

Μέθοδοι

- `public Begin()`
Κατασκευαστής της κλάσης που αρχικοποιεί το αντικείμενο της κλάσης υί καλώντας τον κατασκευαστή της
- `public static void main(String[] args)`
Ξεκινά την εφαρμογή και καλεί τον κατασκευαστή της ίδιας της κλάσης

6

Έλεγχος

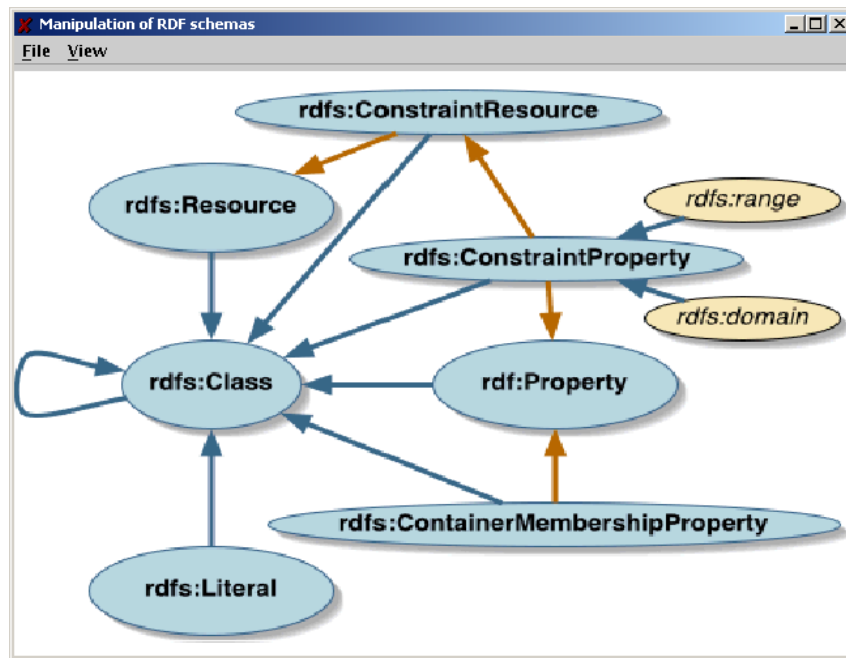
Στο κεφάλαιο αυτό ολοκληρώνουμε την υλοποίηση του συστήματος με το λεπτομερή έλεγχο.

6.1 Μεθοδολογία ελέγχου

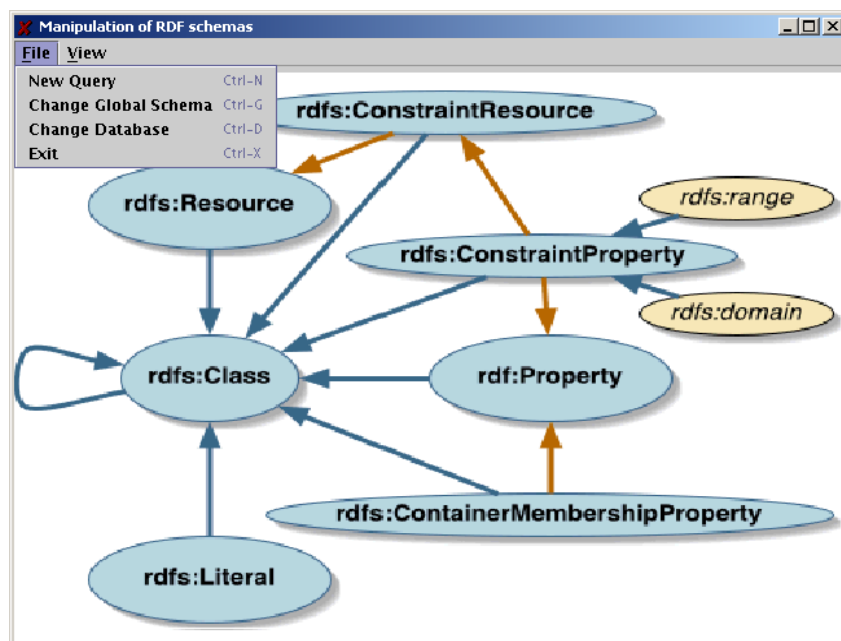
Ο έλεγχος του συστήματος αυτού πραγματοποιήθηκε με τη χρήση σεναρίων λειτουργίας. Με βάση τα χαρακτηριστικά του συστήματος, συντάσσονται σεναρία ελέγχου που χρησιμοποιούν όλες τις λειτουργίες που προσφέρει το σύστημα. Επειδή δεν είναι εύκολο να δώσουμε όλα τα σεναρία που καλύπτουν όλες τις δυνατές περιπτώσεις ελέγχου του συστήματος, παρακάτω παρουσιάζονται τα πιο βασικά για όλες τις λειτουργίες του συστήματος.

6.2 Αναλυτική παρουσίαση ελέγχου

Στην ενότητα αυτή περιγράφεται ένα σενάριο χρήσης του συστήματος με όλες τις πιθανές αποφάσεις που μπορεί να πάρει ο χρήστης. Η αρχική οθόνη του συστήματος φαίνεται στο Σχήμα 6.1. οι δυνατότητες που παρέχει το σύστημα όπως τις αναφέραμε και στο Κεφάλαιο 4.2 δίνονται από τις επιλογές File και View του Μενού. Αυτές φαίνονται στα Σχήματα 6.2 και 6.3 αντίστοιχα.



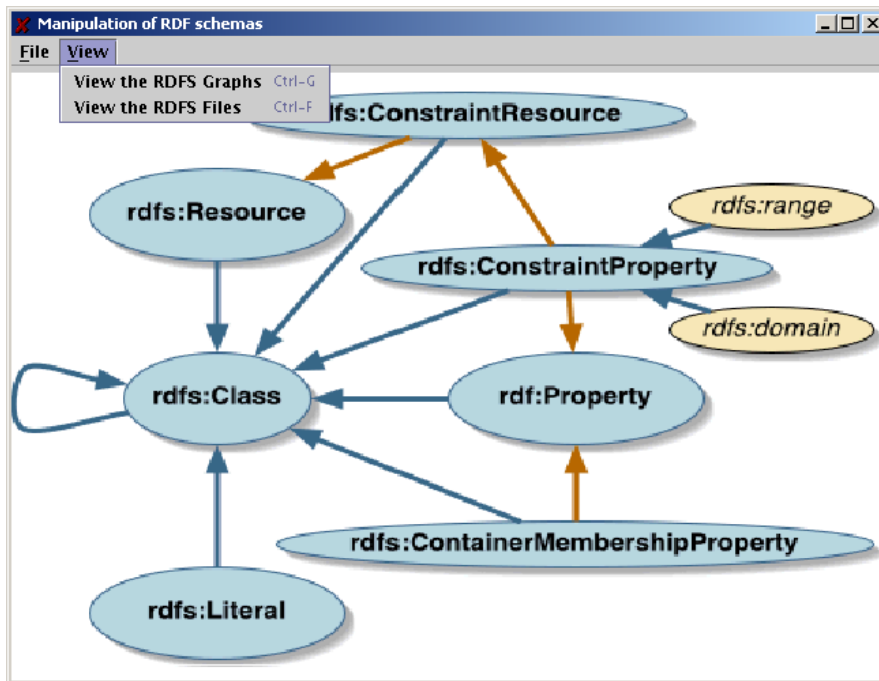
Σχήμα 6.1



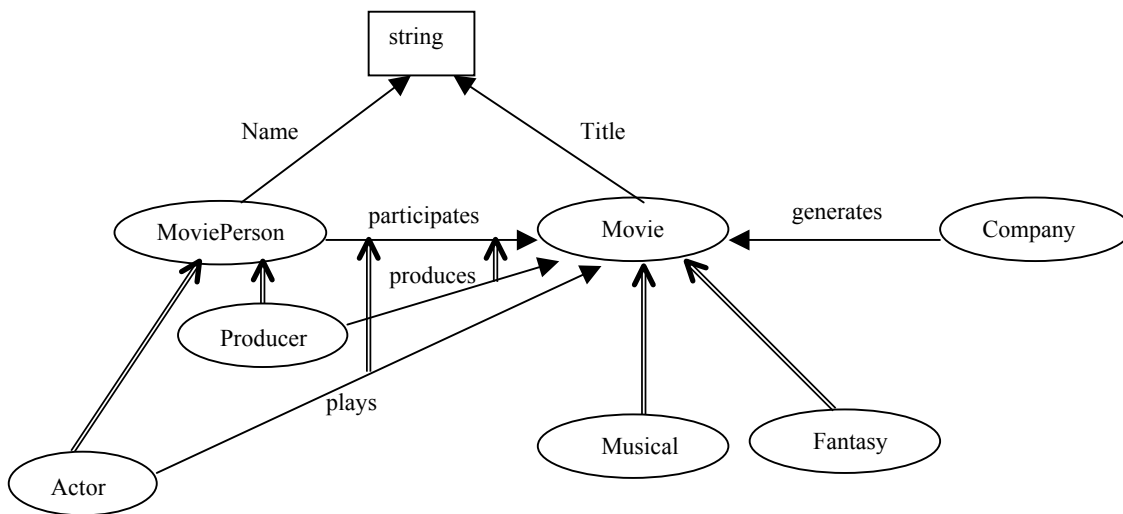
Σχήμα 6.2

6.2.1 Νέα ερώτηση

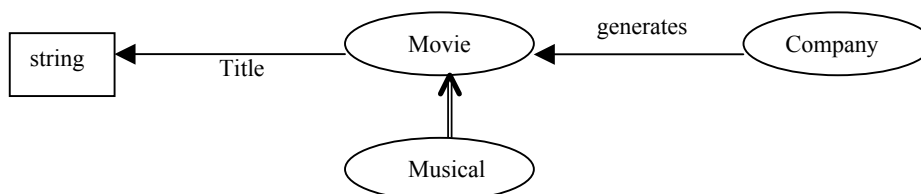
Ως σενάριο σε αυτήν την περίπτωση θα χρησιμοποιήσουμε τρία σχήματα RDF: το καθολικό `movieGlobal.rdfs` και άλλα δύο τα οποία είναι υποσύνολα του καθολικού, `movie1.rdfs` και `movie7.rdfs`. Τα σχήματα αυτά φαίνονται στα Σχήματα 6.4, 6.5 και 6.6 αντίστοιχα.



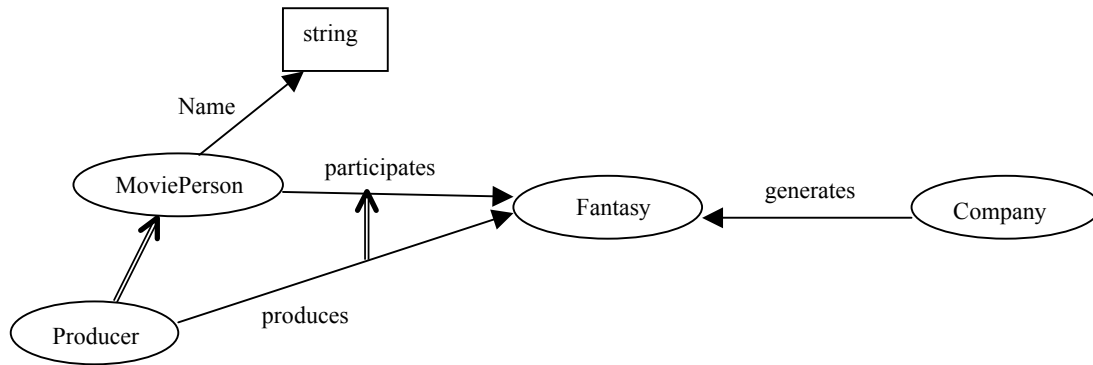
Σχήμα 6.3



Σχήμα 6.4

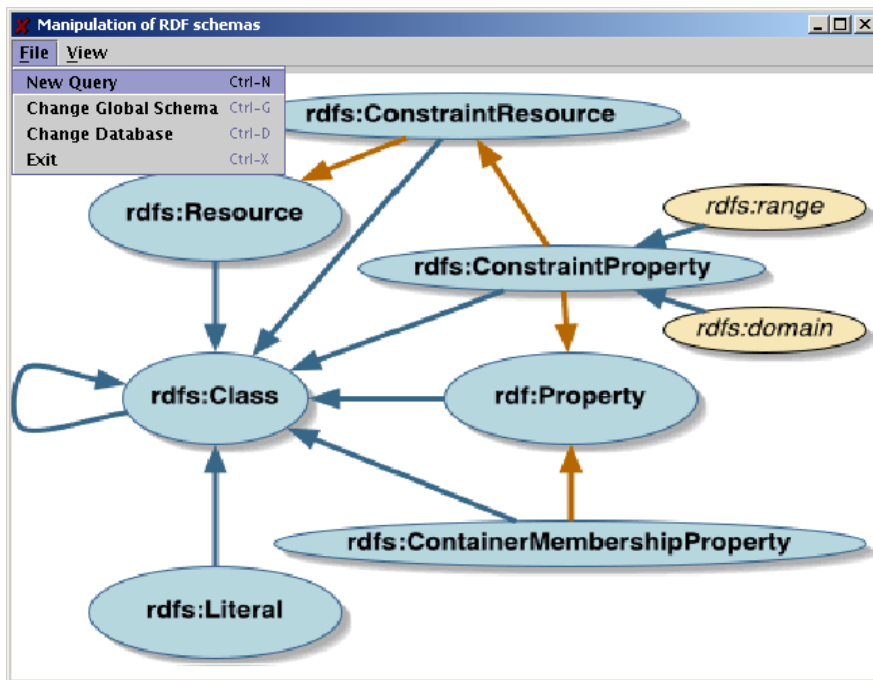


Σχήμα 6.5



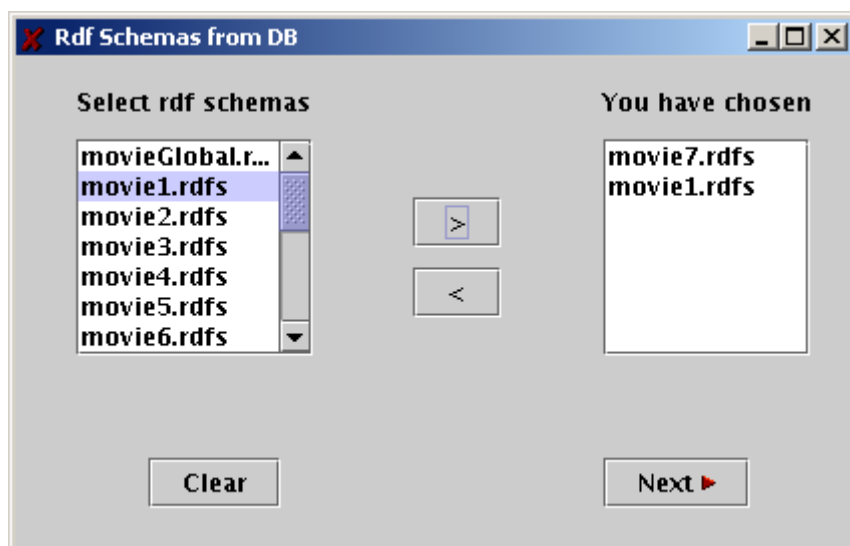
Σχήμα 6.6

Παρουσιάζουμε παρακάτω τις οθόνες που εμφανίζονται για την υλοποίηση μιας νέας ερώτησης.



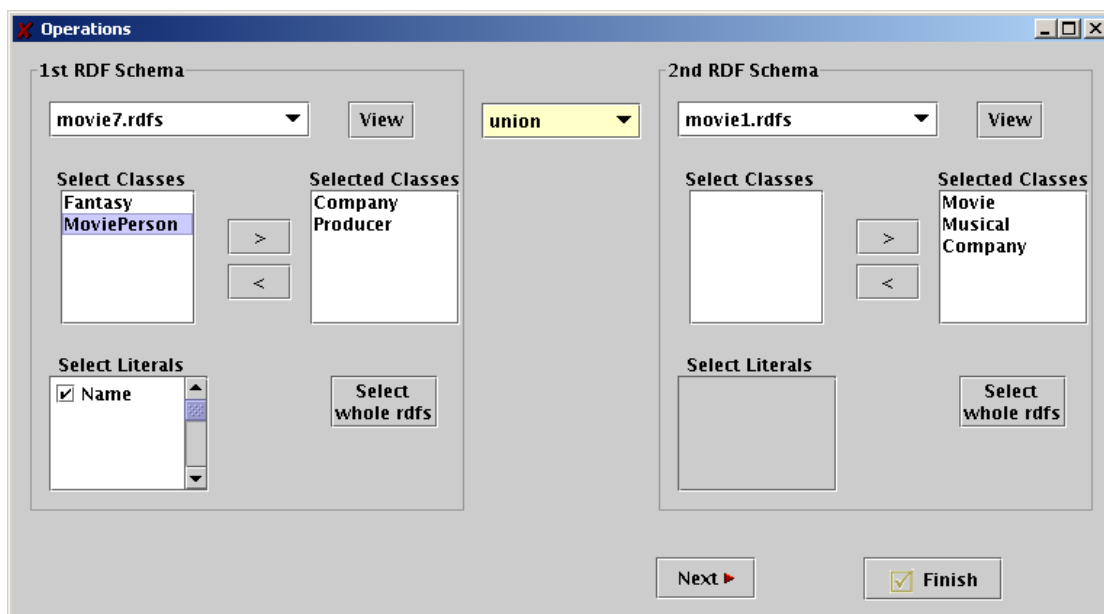
Σχήμα 6.7

Μόλις ο χρήστης πατήσει New Query του εμφανίζεται στην οθόνη το πλαίσιο που φαίνεται στο Σχήμα 6.8 και μπορεί να επιλέξει όποια σχήματα θέλει για να κάνει τις πράξεις του. Αυτό γίνεται με το να διαλέγει από την αριστερή λίστα σχήματα και να τα περνάει στη δεξιά με τη βοήθεια του κουμπιού με το βέλος προς τα δεξιά. Εάν μετανιώσει για κάποια επιλογή του, μπορεί να επιλέξει από την αριστερή λίστα το σχήμα που δε θέλει και να το στείλει ξανά στη δεξιά. Τέλος, ο χρήστης μπορεί να καθαρίσει όλες του τις επιλογές πατώντας το κουμπί Clear.



Σχήμα 6.8

Μόλις πατήσει το κουμπί Next τα δύο αυτά σχήματα είναι διαθέσιμα μαζί με τις κλάσεις που περιέχουν καθώς και οι πράξεις που θα διαλέξει. Η οθόνη αυτή φαίνεται στο Σχήμα 6.9. Στην οθόνη αυτή αν πατήσουμε το κουμπί View μπορούμε να δούμε σε γράφο τα σχήματα που είναι επιλεγμένα στα αντίστοιχα ComboBox. Επίσης, έχουμε τη δυνατότητα να επιλέξουμε ποιες κλάσεις για κάθε σχήμα θέλουμε να συμμετέχουν στην πράξη αυτή. Αυτό γίνεται με τον ίδιο μηχανισμό που είδαμε πριν στο προηγούμενο παράθυρο, ενώ όταν επιλέγουμε μια κλάση φαίνονται από κάτω με CheckBoxes οι ιδιότητες που φεύγουν από την κλάση αυτή και καταλήγουν σε κυριολεκτικά. Τέλος, υπάρχει το κουμπί Select whole rdfs για να επιλέξουμε όλο το σχήμα, δηλαδή όλες τις κλάσεις και τα κυριολεκτικά.



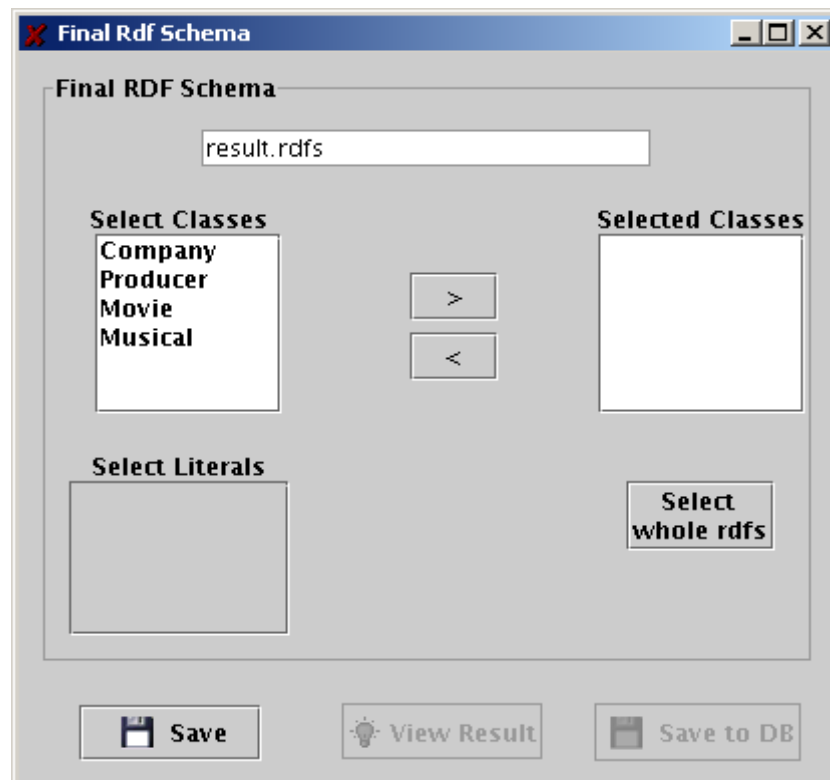
Σχήμα 6.9

Έστω ότι από το σχήμα movie1.rdfs ο χρήστης έχει επιλέξει τις κλάσεις {Movie, Musical, Company} και από το σχήμα movie7.rdfs τις κλάσεις {Company, Producer}, και επιθυμεί να κάνει την πράξη της ένωσης (union) πάνω στα δύο σχήματα. Τότε εάν πατήσει το κουμπί Finish θα του εμφανιστεί το παράθυρο που φαίνεται στο Σχήμα 6.10 για να δώσει όνομα στο τελικό αποτέλεσμα και κατόπιν το παράθυρο που φαίνεται στο Σχήμα 6.11.

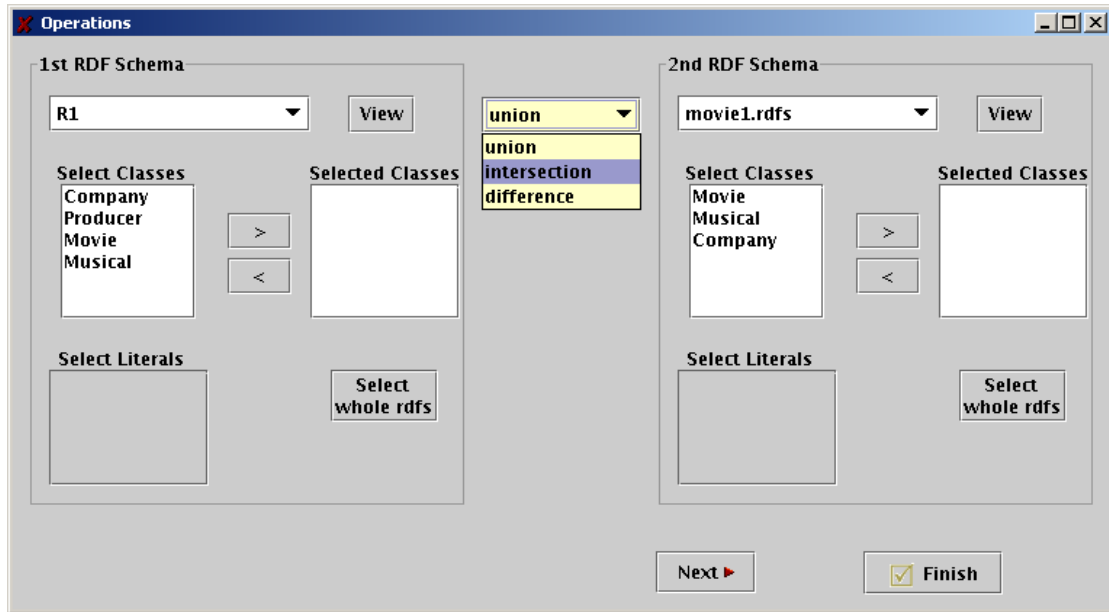


Σχήμα 6.10

Διαφορετικά, εάν πατήσει το κουμπί Next θα υπολογιστεί το αποτέλεσμα της πράξης αυτής και θα ανοίξει το ίδιο παράθυρο αλλά με ένα παραπάνω σχήμα, το R1, το οποίο είναι το αποτέλεσμα της πράξης αυτής. Εδώ έχει τις ίδιες δυνατότητες που περιγράψαμε πριν για επιλογή κλάσεων, κυριολεκτικών και τελεστή. Η οθόνη αυτή φαίνεται στο Σχήμα 6.12. Κατόπιν, εάν πατήσει το κουμπί Finish θα συμβούν όλα όσα περιγράψαμε παραπάνω και θα εμφανιστεί το παράθυρο του Σχήματος 6.11.

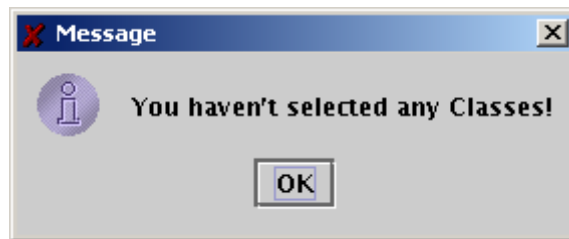


Σχήμα 6.11

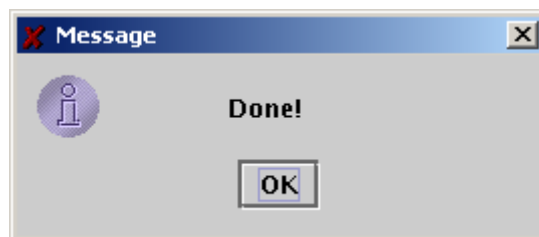


Σχήμα 6.12

Τότε ο χρήστης έχει πάλι τη δυνατότητα να κάνει επιλογή πάνω στις κλάσεις του τελικού αποτελέσματος. Εάν δεν επιλέξει καμία κλάση και πατήσει το κουμπί Save τότε του εμφανίζεται ένα μήνυμα λάθους όπως φαίνεται στο Σχήμα 6.12. Έστω ότι πατάμε το κουμπί Select whole rdfs οπότε επιλέγονται όλες οι κλάσεις του τελικού αποτελέσματος, δηλαδή όλο το τελικό σχήμα. Τότε μόλις πατήσουμε το κουμπί Save αφού σωθεί το αρχείο και εμφανιστεί ένα μήνυμα επιτυχίας, όπως φαίνεται στο Σχήμα 6.13, είναι δυνατή η απεικόνιση του αρχείου και η αποθήκευσή του στη βάση δεδομένων με τα κουμπιά View Result και Save to DB αντίστοιχα, όπως φαίνεται στο Σχήμα 6.14.

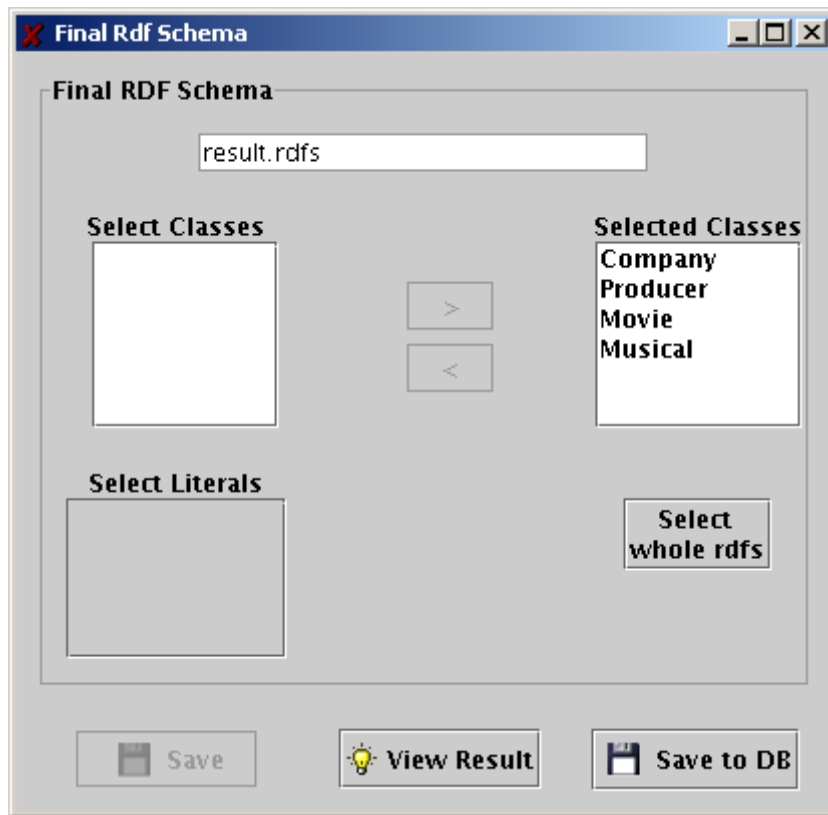


Σχήμα 6.12

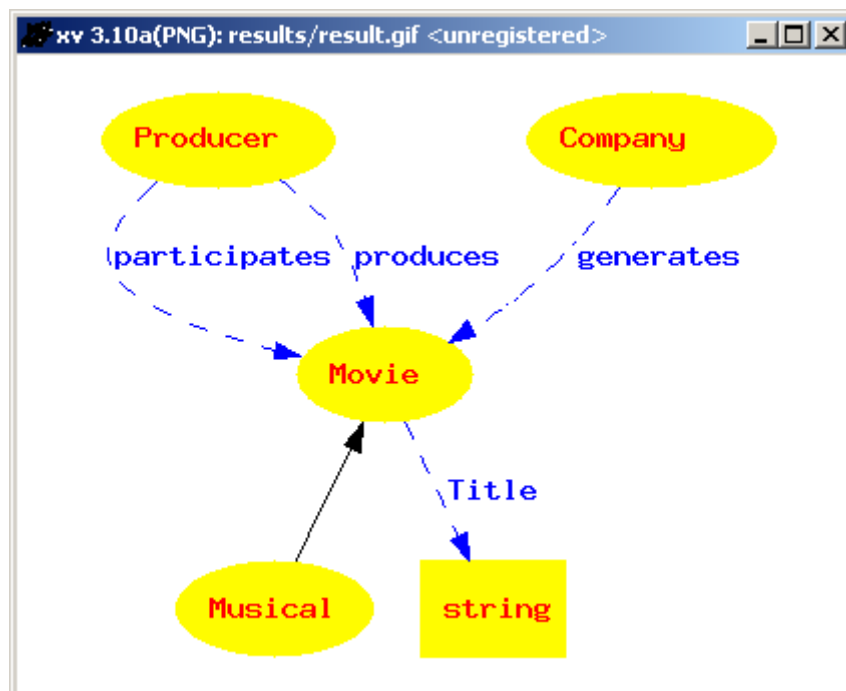


Σχήμα 6.13

Τέλος, εάν ο χρήστης πατήσει το κουμπί View Result θα του ανοίξει ένα παράθυρο το οποίο θα έχει το σχήμα όπως είναι σε μορφή γράφου. Αυτό φαίνεται στο Σχήμα 6.15. εάν πατήσει το κουμπί Save to DB θα του εμφανίσει το παράθυρο που φαίνεται στο Σχήμα 6.13.



Σχήμα 6.14

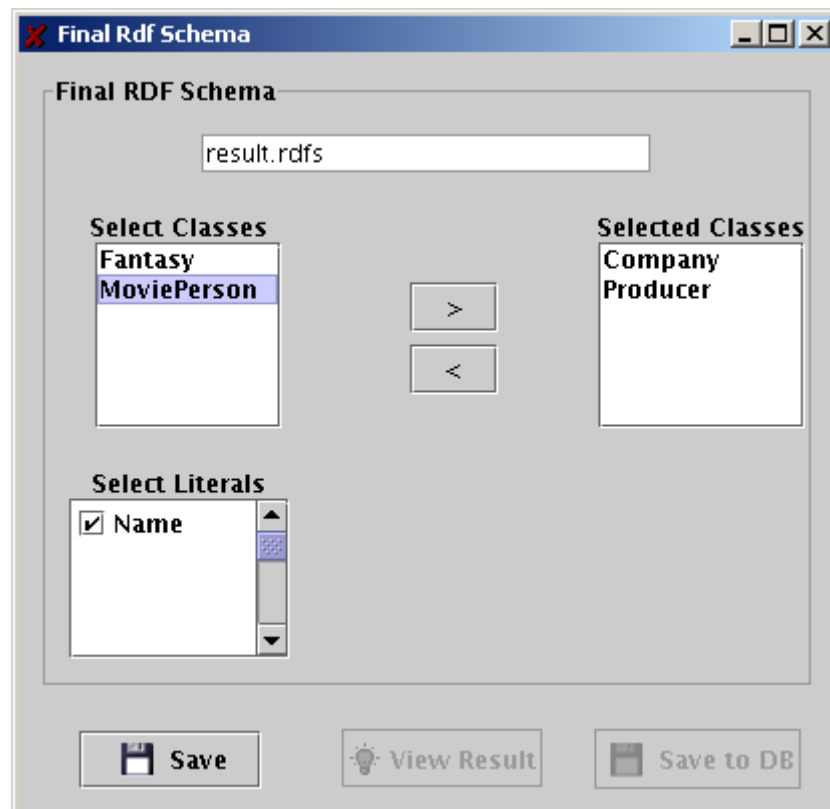


Σχήμα 6.15

Στην περίπτωση τώρα που ο χρήστης επιλέξει ένα μόνο σχήμα από το παράθυρο του Σχήματος 6.8 και πατήσει το κουμπί Next, τότε το εμφανίζεται το παράθυρο του Σχήματος 6.10 για να δώσει κατευθείαν όνομα στο τελικό σχήμα και κατόπιν το παράθυρο του Σχήματος 6.11 ώστε να μπορεί να κάνει επιλογή σε κάποιες κλάσεις και να δημιουργήσει ένα νέο RDF σχήμα. Σε αυτήν την περίπτωση, όμως, λείπει από το παράθυρο το κουμπί Select whole rdfs διότι δεν έχει νόημα να θέλει να επιλέξει όλες τις κλάσεις και να παράγει ένα ίδιο σχήμα. Το παράθυρο φαίνεται στο Σχήμα 6.16, όπου έστω ότι είχαμε διαλέξει μόνο το σχήμα movie7.rdfs.

6.2.2 Αλλαγή καθολικού σχήματος

Το σύστημα δίνει τη δυνατότητα στο χρήστη να αλλάξει το καθολικό σχήμα και να καθορίσει αυτός όποιο σχήμα επιθυμεί. Για να γίνει αυτό ο χρήστης αρκεί να επιλέξει από το File Menu το Change Global Schema οπότε του εμφανίζεται το παράθυρο που φαίνεται στο Σχήμα 6.17. Βέβαια πρέπει να τονίσουμε ότι χρειάζεται προσοχή στην επιλογή του καθολικού σχήματος καθώς πρέπει τα υπόλοιπα σχήματα που θα κάνουμε τις πράξεις να είναι υποσύνολα του καθολικού.



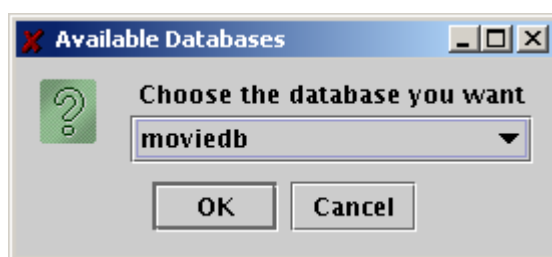
Σχήμα 6.16



Σχήμα 6.17

6.2.3 Αλλαγή βάσης

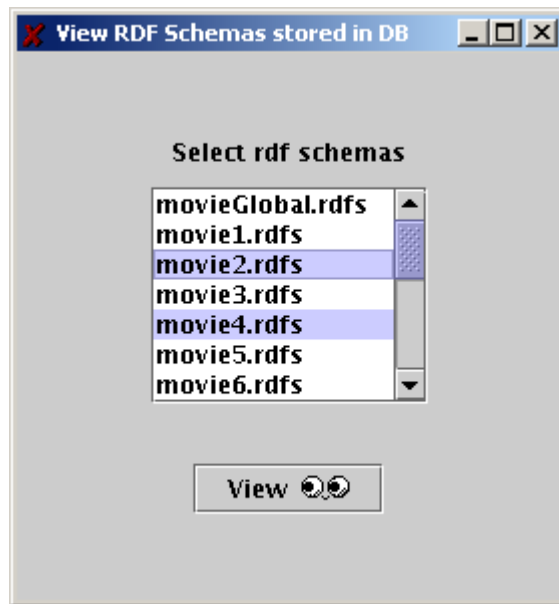
Το σύστημα δίνει τη δυνατότητα στο χρήστη να καθορίσει από ποια βάση θέλει να πάρει RDF σχήματα και να κάνει πράξεις. Αρκεί να επιλέξει από το File Menu το Change Database οπότε και του εμφανίζονται οι διαθέσιμες βάσεις που περιέχουν RDF σχήματα. Το παράθυρο αυτό φαίνεται στο Σχήμα 6.18.



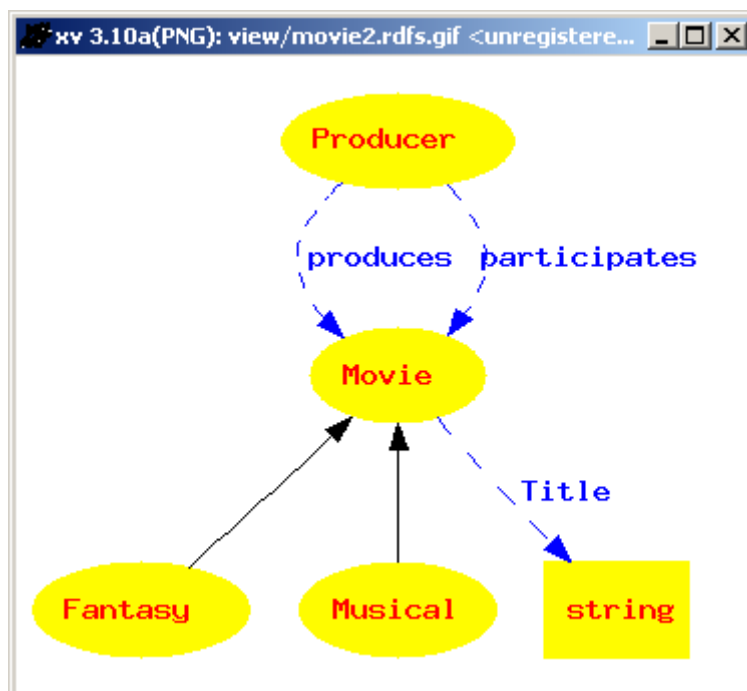
Σχήμα 6.18

6.2.4 Απεικόνιση αποθηκευμένων σχημάτων σε γράφο

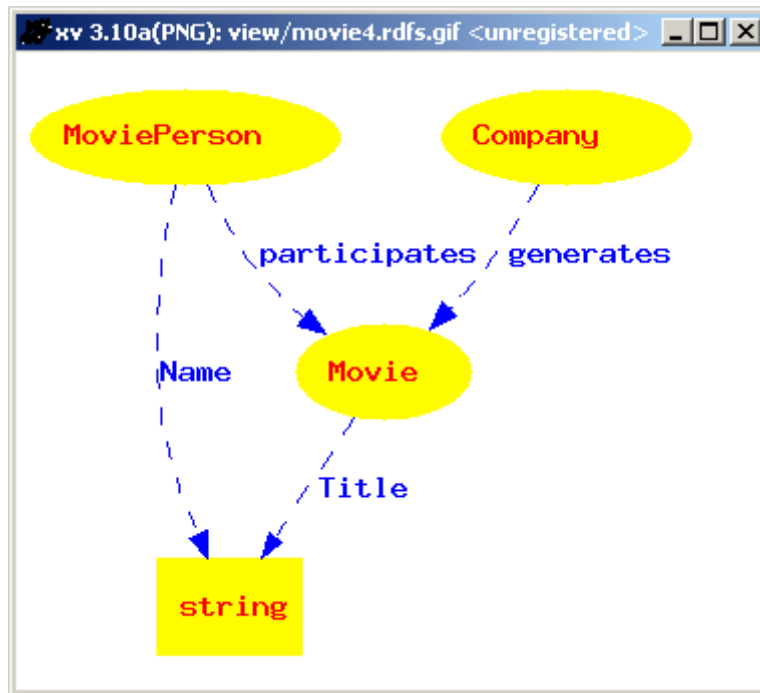
Το σύστημα δίνει τη δυνατότητα για απεικόνιση σχημάτων, που είναι ήδη αποθηκευμένα στη βάση, σε μορφή γράφου. Αυτό γίνεται αρκεί ο χρήστης να επιλέξει από το View Menu το View RDFS Graphs οπότε εμφανίζεται το παράθυρο που φαίνεται στο Σχήμα 6.19. Τότε μπορεί να επιλέξει ένα ή περισσότερα σχήματα για να απεικονίσει. Έστω ότι επιλέξαμε τα σχήματα movie2.rdfs και movie4.rdfs και πατήσουμε το κουμπί View οπότε ανοίγουν τα παράθυρα που φαίνονται στα Σχήματα 6.20 και 6.21.



Σχήμα 6.19



Σχήμα 6.20



Σχήμα 6.21

6.2.5 Απεικόνιση αποθηκευμένων σχημάτων σε μορφή αρχείου

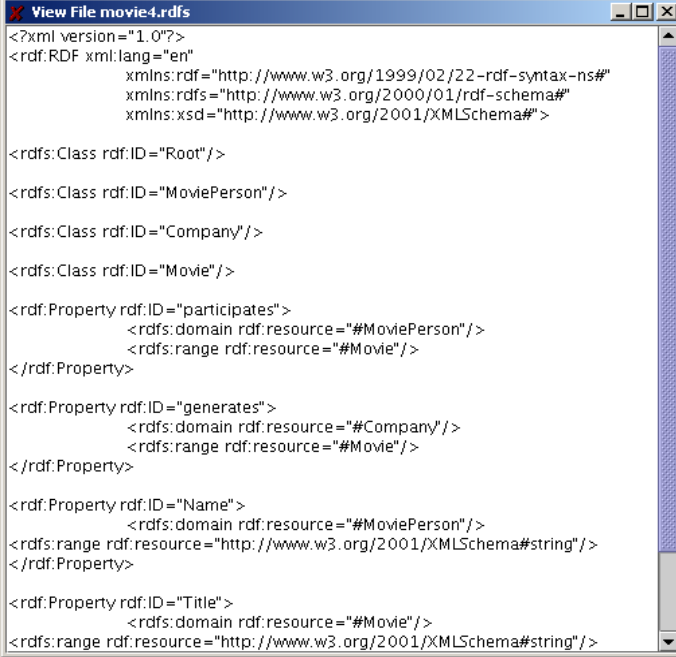
Τέλος, το σύστημα δίνει τη δυνατότητα για απεικόνιση σε μορφή αρχείου RDF/XML σχημάτων αποθηκευμένων στη βάση. Αυτό γίνεται αρκεί ο χρήστης να επιλέξει από το View Menu το View RDFS Files οπότε εμφανίζεται και πάλι το παράθυρο που φαίνεται στο Σχήμα 6.19. εάν επιλέξουμε πάλι τα ίδια σχήματα movie2.rdfs και movie4.rdfs και πατήσουμε το κουμπί View τότε ανοίγουν τα παράθυρα που φαίνονται στα Σχήματα 6.22 και 6.23.

```

View File movie2.rdfs
<?xml version="1.0"?>
<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#" >
  <rdfs:Class rdf:ID="Root"/>
  <rdfs:Class rdf:ID="Producer"/>
  <rdfs:Class rdf:ID="Movie"/>
  <rdfs:Class rdf:ID="Musical">
    <rdfs:subClassOf rdf:resource="#Movie"/>
  </rdfs:Class>
  <rdfs:Class rdf:ID="Fantasy">
    <rdfs:subClassOf rdf:resource="#Movie"/>
  </rdfs:Class>
  <rdf:Property rdf:ID="participates">
    <rdfs:domain rdf:resource="#Producer"/>
    <rdfs:range rdf:resource="#Movie"/>
  </rdf:Property>
  <rdf:Property rdf:ID="produces">
    <rdfs:domain rdf:resource="#Producer"/>
    <rdfs:range rdf:resource="#Movie"/>
    <rdfs:subPropertyOf rdf:resource="#participates"/>
  </rdf:Property>
  <rdf:Property rdf:ID="Title">

```

Σχήμα 6.22



```
<?xml version="1.0"?>
<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#">

<rdfs:Class rdf:ID="Root"/>

<rdfs:Class rdf:ID="MoviePerson"/>

<rdfs:Class rdf:ID="Company"/>

<rdfs:Class rdf:ID="Movie"/>

<rdf:Property rdf:ID="participates">
  <rdfs:domain rdf:resource="#MoviePerson"/>
  <rdfs:range rdf:resource="#Movie"/>
</rdf:Property>

<rdf:Property rdf:ID="generates">
  <rdfs:domain rdf:resource="#Company"/>
  <rdfs:range rdf:resource="#Movie"/>
</rdf:Property>

<rdf:Property rdf:ID="Name">
  <rdfs:domain rdf:resource="#MoviePerson"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</rdf:Property>

<rdf:Property rdf:ID="Title">
  <rdfs:domain rdf:resource="#Movie"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</rdf:Property>
```

Σχήμα 6.23

7

Επίλογος

Με το κεφάλαιο αυτό ολοκληρώνεται η παρούσα διπλωματική. Παρακάτω παρουσιάζονται τα συμπεράσματα και τυχόν μελλοντικές επεκτάσεις που μπορούν να γίνουν.

7.1 Σύνοψη και συμπεράσματα

Η καθιέρωση του Σημασιολογικού Ιστού ως το επόμενο βήμα στον παγκόσμιο Ιστό και ο όλο και μεγαλύτερος όγκος δεδομένων που ανταλλάσσονται και επεξεργάζονται δημιουργεί μια ανάγκη για καλύτερη οργάνωση των δεδομένων με τη βοήθεια του μοντέλου RDF. Σκοπός της διπλωματικής εργασίας αυτής ήταν η υλοποίηση ενός πρότυπου συστήματος για τη διαχείριση RDF σχημάτων με σκοπό τη σύγκριση δικτυακών τόπων οργανωμένων με ένα κοινό πρότυπο RDF σχήματος και τη δημιουργία καινούριων. Αυτό επιτεύχθηκε με τη χρήση τεσσάρων τελεστών: *επιλογή, ένωση, τομή και διαφορά*.

Αρχικά παρουσιάστηκε το θεωρητικό υπόβαθρο που χρειάστηκε, όπως το RDF/S και η γλώσσα ερωτήσεων RQL. Επίσης, παρουσιάστηκαν σχετικές εργασίες πάνω στο θέμα αυτό, όπως το σύστημα RONDO και η διδακτορική διατριβή με θέμα τη διαχείριση ιεραρχικών σχημάτων στο Σημασιολογικό Ιστό.

Στη συνέχεια, παρουσιάστηκε η θεωρητική μελέτη που έγινε στα πλαίσια αυτής της διπλωματικής εργασίας. Δόθηκαν βασικοί ορισμοί, όπως ο ορισμός του RDF Schema, τι αποτελεί υποσύνολο ενός RDF σχήματος και τι είναι το καθολικό σχήμα. Επίσης, ορίσαμε

τον τελεστή *επιλογή*, όπου δοθέντος ενός συνόλου κλάσεων από ένα σχήμα επιστρέφει το ελάχιστο σε κλάσεις σχήμα που είναι υποσύνολο του αρχικού. Τέλος, με βάση τον τελεστή *επιλογή* ορίσαμε τους τελεστές *ένωση*, *τομή* και *διαφορά* δύο RDF σχημάτων. Ο τελεστής *ένωση*, που εφαρμόζεται σε δύο RDF σχήματα, συγχωνεύει τα σχήματα αυτά παίρνοντας όλη την πληροφορία που παρείχαν. Ο τελεστής *τομή*, που εφαρμόζεται επίσης σε δύο RDF σχήματα, βρίσκει το κοινό κομμάτι τους. Τέλος, ο τελεστής *διαφορά*, που εφαρμόζεται επίσης σε δύο RDF σχήματα, βρίσκει το κομμάτι του πρώτου που δεν υπήρχε στο δεύτερο σχήμα.

Κατόπιν, παρουσιάστηκε η ανάλυση, σχεδίαση και υλοποίηση του συστήματος. Στην ανάλυση της αρχιτεκτονικής του συστήματος περιγράφηκαν τα υποσυστήματα που το αποτελούν και πώς επικοινωνούν μεταξύ τους, ενώ στη σχεδίαση περιγράφηκαν οι εφαρμογές του. Στη συνέχεια, παρουσιάστηκε η υλοποίηση του συστήματος σε επίπεδο εγκατάστασης βασικών εργαλείων αλλά και σε επίπεδο παρουσίασης των βασικών αλγορίθμων και του σκελετού του κώδικα του προγράμματος. Τέλος, έγινε μια παρουσίαση της λειτουργίας του συστήματος μέσω ενός σεναρίου χρήσης.

Συμπερασματικά, αυτή η διπλωματική εργασία υλοποίησε ένα πρότυπο σύστημα διαχείρισης RDF σχημάτων με τέτοιο τρόπο ώστε να δημιουργεί νέα σχήματα, παρέχοντας μια γλώσσα ερωτήσεων-με-χρήση-παραδείγματος με τον τελεστή *επιλογή* και τους βασικούς τελεστές της συνολοθεωρίας *ένωση*, *τομή* και *διαφορά*.

7.2 Μελλοντικές επεκτάσεις

Η διπλωματική εργασία αυτή αποτελεί μια καλή βάση για περαιτέρω ανάπτυξη πάνω σε θέματα διαχείρισης RDF σχημάτων. Παρακάτω απαριθμούμε μια λίστα με πράγματα που θα είχε ενδιαφέρον να γίνουν στο μέλλον για επέκταση και αναβάθμιση αυτής.

- Ενσωμάτωση των τελεστών *ένωση*, *τομή* και *διαφορά* καθώς και του ειδικού τελεστή *επιλογή* στη γλώσσα RQL [1].
- Δυνατότητα *επιλογής* των ιδιοτήτων που έχουν πεδίο τιμών κυριολεκτικά (string, integer κλπ.) και στις υποκλάσεις των κλάσεων που είναι πεδίο ορισμού των ιδιοτήτων αυτών. Δηλαδή εάν από μια κλάση A ξεκινάει μια ιδιότητα p που καταλήγει σε string, να είναι δυνατή η *επιλογή* της ιδιότητας αυτής από την κλάση B, υποκλάση της A, ακόμα και όταν η κλάση A δεν επιλέγεται.
- Προσθήκη *επιλογής* στο Μενού για εισαγωγή στη βάση δεδομένων έτοιμων RDF αρχείων, ώστε να μην είναι απαραίτητη η χρήση του εργαλείου RSSDB του RDFSuite [11].

8

Βιβλιογραφία

- [1] Gregory Karvounarakis, Sofia Alexaki, Vassilis Christophides, Dimitris Plexousakis, RQL: A Declarative Query Language for RDF, WWW2002, May 7-11, 2002, Honolulu, Hawaii, USA
- [2] Sergey Melnik, Erhard Rahm, Philip A. Bernstein, Rondo: A Programming Platform for Generic Model Management, SIGMOD 2003, June 9-12, 2003, San Diego, CA.
- [3] Θεόδωρος Δαλαμάγκας, Διαχείριση Ιεραρχικών Σχημάτων στο Σημασιολογικό Ιστό, Διδακτορική Διατριβή, 2004
- [4] Αλεξάνδρα Μέλιου, Μοντελοποίηση και Διερεύνηση των Αλγεβρικών Ιδιοτήτων Δενδρικών Ιεραρχικών Δομών, Διπλωματική Εργασία, 2003
- [5] T. Dalamagas, A. Meliou, T. Sellis, Modelling and manipulating the structure of portal catalogs, Tech. report, KDBS Lab, Dept of Electrical and Computer Eng., NTU Athens, Jan 2003
- [6] RDF Primer, W3C Recommendation 10 February 2004

- [7] Semantic Web, available at <http://www.semanticweb.org>
- [8] Deitel&Deitel, Java How to Program, 4th Edition

- [9] Graphviz - open source graph drawing software, available at <http://www.research.att.com/sw/tools/graphviz/>

- [10] The FRODO RDFSviz Tool, available at <http://www.dfki.uni-kl.de/frodo/RDFSviz/>

- [11] RDFSuite, <http://139.91.183.30:9090/RDF/index.html>